CS-302

Computer Networks Lab Report

Assignment - 6

BTech 5th Semester 2021

Department of Computer Science and Engineering,

National Institute of Technology Karnataka, Surathkal

October 2021

Submitted By:
Deepta Devkota - 191CS117
Akanksha More - 191CS106

Part A

1. Print the list of network interfaces, their MAC addresses, and their assigned IP addresses, if any.

Ans:

In order to list down the network interfaces, the MAC addresses, and assigned IP addresses, we use the command **ifconfig.**

ifconfig (interface configurator) command is used to initialize an interface, assign IP Address to interface and enable or disable interface on demand. We can view the **IP Address** and **Hardware / MAC address** assigned to the interface and also MTU (Maximum transmission unit) size.

```
dell@dell-Inspiron-3593:~$ ifconfig
enp2s0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 98:e7:43:3a:6c:e7  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
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 20900  bytes 2190001 (2.1 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 20900  bytes 2190001 (2.1 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlp3s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.105  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 2400:1a00:b050:ec6c:1b94:3b2e:5c5e:27a0  prefixlen 64  scopeid 0x0<global>
        inet6 2400:1a00:b050:ec6c:62a0:3dfa:c489:f9a9  prefixlen 64  scopeid 0x0<global>
        inet6 fe80::6bc0:fa5c:fe5d:62fd  prefixlen 64  scopeid 0x20<link>
        ether ac:d5:64:90:7c:6d  txqueuelen 1000  (Ethernet)
        RX packets 1355751  bytes 1339558081 (1.3 GB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 648713  bytes 206348954 (206.3 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

The respective network interfaces, with their MAC and IP address, are listed down below:

1. **<u>enp2s0:</u>**

   The first network interface is the ethernet interface.

   The following info can be deduced from enp2s0

   1. en stands for ethernet
   2. p2 stands for bus number, here bus number is 2
   3. s0 stands for slot number, here slot number is 0

   MAC address: 98:e7:43:3a:6c:e7

2. **lo:**
   lo stands for lookback. The loopback network interface is a software loopback mechanism which may be used for performance analysis, software testing, and/or local communication. It should be the last interface configured, as protocols may use the order of configuration as an indication of priority.
   IP address: 127.0.0.1

3. **wlp3s0:**
   It is the **WLAN** interface. The following info can be deduced from wlp3s0.
   a. wl stands for WLAN
   b. p3 stands for bus number, here bus number is 3
   c. s0 stands for slot number, here slot number is 0

   MAC address: ac:d5:64:90:7c:6d
   IP address: 192.168.1.105

---

2. Calculate the latency between mininet vm and www.rutgers.edu for 10 packets. Repeat the result for stanford.edu and www.google.co.in and compare the difference in latency.

Ans:

Network latency is the time it takes for data or a request to go from the source to the destination. Latency in networks is measured in milliseconds. The closer the latency is to zero, the better.

Common reasons for high latency:

- Long time to send data.
- Long time to access the servers or web-applications
- Websites don't load

Latency can be measured as the **Round Trip Time (RTT)** or **Time to First Byte (TTFB)**. For the following question we have measured the latency in terms of RTT.

RTT(Round Trip Time): Amount of time to send a request and receive response.

For the following question we have used the ping(Packet INternet Groper) command to calculate the latency between mininet vm and specified destinations. ping command is used to test connectivity between two nodes.

Format: ping -c N ip_address (N: number of packets)

a.Mininet vm and ww.rutgers.edu

```
dell@dell-Inspiron-3593:~$ ping -c 10 www.rutgers.edu
PING www.rutgers.edu (128.6.46.88) 56(84) bytes of data.
64 bytes from www.rutgers.edu (128.6.46.88): icmp_seq=1 ttl=240 time=268 ms
64 bytes from www.rutgers.edu (128.6.46.88): icmp_seq=2 ttl=240 time=276 ms
64 bytes from www.rutgers.edu (128.6.46.88): icmp_seq=3 ttl=240 time=346 ms
64 bytes from www.rutgers.edu (128.6.46.88): icmp_seq=4 ttl=240 time=271 ms
64 bytes from www.rutgers.edu (128.6.46.88): icmp_seq=5 ttl=240 time=263 ms
64 bytes from www.rutgers.edu (128.6.46.88): icmp_seq=6 ttl=240 time=264 ms
64 bytes from www.rutgers.edu (128.6.46.88): icmp_seq=7 ttl=240 time=337 ms
64 bytes from www.rutgers.edu (128.6.46.88): icmp_seq=8 ttl=240 time=357 ms
64 bytes from www.rutgers.edu (128.6.46.88): icmp_seq=9 ttl=240 time=269 ms
64 bytes from www.rutgers.edu (128.6.46.88): icmp_seq=10 ttl=240 time=301 ms

--- www.rutgers.edu ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9004ms
rtt min/avg/max/mdev = 263.403/295.226/356.646/35.488 ms
```

As we can see, there are 10 packets being sent to the server.

For each packet we can observe the following:
from: destination and its ip address
icmp_seq: sequence number of each ICMP packet
ttl: Time to live value i.e. number of network hops a packet can take before a router discards it.
time: the time it took a packet to reach the destination and come back to the source.

We can see that there is 0% packet loss and the ping timeout is 9004ms.

minimum RTT: 263.403
average RTT: 295.226
maximum RTT: 356.646

b. mininet vm and stanford.edu

```
dell@dell-Inspiron-3593:~$ ping -c 10 www.stanford.edu
PING www.stanford.edu(2a04:4e42:48::645 (2a04:4e42:48::645)) 56 data bytes
64 bytes from 2a04:4e42:48::645 (2a04:4e42:48::645): icmp_seq=1 ttl=54 time=342 ms
64 bytes from 2a04:4e42:48::645 (2a04:4e42:48::645): icmp_seq=2 ttl=54 time=371 ms
64 bytes from 2a04:4e42:48::645 (2a04:4e42:48::645): icmp_seq=3 ttl=54 time=313 ms
64 bytes from 2a04:4e42:48::645 (2a04:4e42:48::645): icmp_seq=4 ttl=54 time=313 ms
64 bytes from 2a04:4e42:48::645 (2a04:4e42:48::645): icmp_seq=5 ttl=54 time=350 ms
64 bytes from 2a04:4e42:48::645 (2a04:4e42:48::645): icmp_seq=6 ttl=54 time=365 ms
64 bytes from 2a04:4e42:48::645 (2a04:4e42:48::645): icmp_seq=7 ttl=54 time=383 ms
64 bytes from 2a04:4e42:48::645 (2a04:4e42:48::645): icmp_seq=8 ttl=54 time=406 ms
64 bytes from 2a04:4e42:48::645 (2a04:4e42:48::645): icmp_seq=9 ttl=54 time=318 ms
64 bytes from 2a04:4e42:48::645 (2a04:4e42:48::645): icmp_seq=10 ttl=54 time=366 ms

--- www.stanford.edu ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9002ms
rtt min/avg/max/mdev = 313.043/352.796/406.180/29.804 ms
```

We can see that there is 0% packet loss and the ping timeout is 9002ms.

minimum RTT: 313.043
average RTT: 352.769
maximum RTT: 406.180

c. mininet vm and www.google.co.in

```
dell@dell-Inspiron-3593:~$ ping -c 10 www.google.co.in
PING www.google.co.in(bom12s17-in-x03.1e100.net (2404:6800:4009:82a::2003)) 56 data bytes
64 bytes from bom12s17-in-x03.1e100.net (2404:6800:4009:82a::2003): icmp_seq=1 ttl=114 time=52.9 ms
64 bytes from bom12s17-in-x03.1e100.net (2404:6800:4009:82a::2003): icmp_seq=2 ttl=114 time=61.2 ms
64 bytes from bom12s17-in-x03.1e100.net (2404:6800:4009:82a::2003): icmp_seq=3 ttl=114 time=54.2 ms
64 bytes from bom12s17-in-x03.1e100.net (2404:6800:4009:82a::2003): icmp_seq=4 ttl=114 time=54.2 ms
64 bytes from bom12s17-in-x03.1e100.net (2404:6800:4009:82a::2003): icmp_seq=5 ttl=114 time=54.6 ms
64 bytes from bom12s17-in-x03.1e100.net (2404:6800:4009:82a::2003): icmp_seq=6 ttl=114 time=60.5 ms
64 bytes from bom12s17-in-x03.1e100.net (2404:6800:4009:82a::2003): icmp_seq=7 ttl=114 time=53.2 ms
64 bytes from bom12s17-in-x03.1e100.net (2404:6800:4009:82a::2003): icmp_seq=8 ttl=114 time=54.4 ms
64 bytes from bom12s17-in-x03.1e100.net (2404:6800:4009:82a::2003): icmp_seq=9 ttl=114 time=54.0 ms
64 bytes from bom12s17-in-x03.1e100.net (2404:6800:4009:82a::2003): icmp_seq=10 ttl=114 time=54.0 ms

--- www.google.co.in ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9012ms
rtt min/avg/max/mdev = 52.895/55.328/61.241/2.813 ms
```

We can see that there is 0% packet loss and the ping timeout is 9012ms.

minimum RTT: 52.895
average RTT: 55.328
maximum RTT: 61.241

Thus, we can observe that out of all three destinations, www.google.co.in has the lowest RTT and correspondingly lower latency and hence has a better performance.

Part B

Create a simple two-node network using "sudo mn" and do the following

Ans:

Running **sudo mn** command in the command prompt creates **two namespaces h1, h2,** and the **switch s1** with virtual ethernet links connecting the switch and the namespaces. We have two links h1-s1 and h2-s1.

```
dell@dell-Inspiron-3593:~$ sudo mn
[sudo] password for dell:
*** No default OpenFlow controller found for default switch!
*** Falling back to OVS Bridge
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller

*** Starting 1 switches
s1 ...
*** Starting CLI:
```

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

The command **h1 ifconfig** | **grep** ether lists the MAC address of a namespace.

MAC address of h1:

```
mininet> h1 ifconfig | grep ether
        ether fa:ee:90:64:23:ab  txqueuelen 1000  (Ethernet)
```

To further observe the interface of the MAC address we run the command

h1 ifconfig

Interface for the h1 MAC address:

```
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
        inet6 fe80::f8ee:90ff:fe64:23ab  prefixlen 64  scopeid 0x20<link>
        ether fa:ee:90:64:23:ab  txqueuelen 1000  (Ethernet)
        RX packets 37  bytes 4774 (4.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 11  bytes 866 (866.0 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
        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
```

The MAC address of host h1 is fa:ee:90:64:23:ab

The command **“s1 ifconfig** | **grep ether”** lists the MAC address of the namespace.

MAC address of s1:

```
mininet> s1 ifconfig | grep ether
        ether 98:e7:43:3a:6c:e7  txqueuelen 1000  (Ethernet)
        ether 26:b7:e5:2b:df:ed  txqueuelen 1000  (Ethernet)
        ether c6:03:22:d6:6c:51  txqueuelen 1000  (Ethernet)
        ether ac:d5:64:90:7c:6d  txqueuelen 1000  (Ethernet)
```

Interfaces for the s1 MAC addresses:

To further observe the interface of the MAC address we run the command

h1 ifconfig

```
mininet> s1 ifconfig
enp2s0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 98:e7:43:3a:6c:e7  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
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 21087  bytes 2209799 (2.2 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 21087  bytes 2209799 (2.2 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s1-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::24b7:e5ff:fe2b:dfed  prefixlen 64  scopeid 0x20<link>
        ether 26:b7:e5:2b:df:ed  txqueuelen 1000  (Ethernet)
        RX packets 12  bytes 936 (936.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 40  bytes 5117 (5.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s1-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::c403:22ff:fed6:6c51  prefixlen 64  scopeid 0x20<link>
        ether c6:03:22:d6:6c:51  txqueuelen 1000  (Ethernet)
        RX packets 12  bytes 936 (936.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 39  bytes 5047 (5.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlp3s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.105  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 2400:1a00:b050:ec6c:1b94:3b2e:5c5e:27a0  prefixlen 64  scopeid 0x0<global>
        inet6 2400:1a00:b050:ec6c:62a0:3dfa:c489:f9a9  prefixlen 64  scopeid 0x0<global>
        inet6 fe80::6bc0:fa5c:fe5d:62fd  prefixlen 64  scopeid 0x20<link>
        ether ac:d5:64:90:7c:6d  txqueuelen 1000  (Ethernet)
        RX packets 1392854  bytes 1387012561 (1.3 GB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 658082  bytes 209165722 (209.1 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

As seen in the result above, in switch s1 the MAC address of the:

a. **ethernet interface** is **6e:09:7f:83:92:8e.**
b. **s1-eth1 interface** is **26:b7:e5:2b:df:ed**
c. **s1-eth2 interface** is **c6:03:22:d6:6c:51**
d. **wlp3s0 interface** is **ac:d5:64:9-:7c:6d**

Interfaces if s1 can be obtained from the i**fconfig** command:

Command : s1 ifconfig

The various interfaces are listed below:

```
mininet> s1 ifconfig
enp2s0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 98:e7:43:3a:6c:e7  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
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 21087  bytes 2209799 (2.2 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 21087  bytes 2209799 (2.2 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s1-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::24b7:e5ff:fe2b:dfed  prefixlen 64  scopeid 0x20<link>
        ether 26:b7:e5:2b:df:ed  txqueuelen 1000  (Ethernet)
        RX packets 12  bytes 936 (936.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 40  bytes 5117 (5.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s1-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::c403:22ff:fed6:6c51  prefixlen 64  scopeid 0x20<link>
        ether c6:03:22:d6:6c:51  txqueuelen 1000  (Ethernet)
        RX packets 12  bytes 936 (936.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 39  bytes 5047 (5.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlp3s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.105  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 2400:1a00:b050:ec6c:1b94:3b2e:5c5e:27a0  prefixlen 64  scopeid 0x0<global>
        inet6 2400:1a00:b050:ec6c:62a0:3dfa:c489:f9a9  prefixlen 64  scopeid 0x0<global>
        inet6 fe80::6bc0:fa5c:fe5d:62fd  prefixlen 64  scopeid 0x20<link>
        ether ac:d5:64:90:7c:6d  txqueuelen 1000  (Ethernet)
        RX packets 1392854  bytes 1387012561 (1.3 GB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 658082  bytes 209165722 (209.1 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

The command **net**, shows the nodes, the interfaces, and the connections:

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

The description of the interfaces is as listed below:

1. **enp2s0:**
   It is the ethernet interface. The following info can be deduced from enp2s0
   d. en stands for ethernet
   e. p2 stands for bus number, here bus number is 2
   f. s0 stands for slot number, here slot number is 0
2. **lo:**
   lo stands for lookback. The loopback network interface is a software loopback mechanism which may be used for performance analysis, software testing, and/or local communication. It should be the last interface configured, as protocols may use the order of configuration as an indication of priority. As seen in the above figure the IP address is **127.0.0.1**

3. **s1-eth1:**
   As observed in the above result, s1-eth1 is the interface for the virtual ethernet link between h1 and s1.
4. **s1-eth2:**
   As observed in the above result, s1-eth2 is the interface for the virtual ethernet link between h2 and s1.
5. **wlp3s0:**
   It is the WLAN interface. The following info can be deduced from wlp3s0.
   a. wl stands for WLAN
   b. p3 stands for bus number, here bus number is 3
   c. s0 stands for slot number, here slot number is 0
   The IP address for this interface is **192.168.1.105**

---

b. Ping h1 from h2 and view the ARP entries stored at hosts h1 and h2. Measure the TCP throughput from h1 to h2 using iperf.

Ans:

Ping h1 from h2:

```
mininet> h2 ping -c 10 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=0.698 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.089 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.075 ms
64 bytes from 10.0.0.1: icmp_seq=4 ttl=64 time=0.079 ms
64 bytes from 10.0.0.1: icmp_seq=5 ttl=64 time=0.069 ms
64 bytes from 10.0.0.1: icmp_seq=6 ttl=64 time=0.070 ms
64 bytes from 10.0.0.1: icmp_seq=7 ttl=64 time=0.072 ms
64 bytes from 10.0.0.1: icmp_seq=8 ttl=64 time=0.049 ms
64 bytes from 10.0.0.1: icmp_seq=9 ttl=64 time=0.077 ms
64 bytes from 10.0.0.1: icmp_seq=10 ttl=64 time=0.080 ms

--- 10.0.0.1 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9208ms
rtt min/avg/max/mdev = 0.049/0.135/0.698/0.187 ms
```

Thus, **we pinged h1 from h2 by limiting the number of packets sent to 10**. We can observe the minimum, average, and maximum RTT as follows:

minimum RTT: 0.049

average RTT: 0.135

maximum RTT: 0.698

ARP:

Address Resolution Protocol (ARP) is a procedure for mapping a **dynamic IP address to a permanent MAC address** in a local area network (LAN).

We can observe the **arp entries** for the nodes h1 and h2 as follows:

```
mininet> h1 arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.0.2                 ether   5e:8d:9f:92:5b:c6   C                     h1-eth0
```

```
mininet> h2 arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.0.1                 ether   fa:ee:90:64:23:ab   C                     h2-eth0
```

arp, the command generates an ARP table of hosts with the ARP entries of each other. For two hosts h1 and h2, using the command gives the h2 Mac address inside the ARP table of h1 and vice versa.

In the ARP table as shown above:
Address: IP address of another device connected to the same network
HWtype: Tells *arp* which class of entries it should check for, by default value is ether
HWaddress: MAC address of the device connected to the same network
Flags Mask: helps in understanding what type of entry is being placed in the memory as seen in the ARP table flags.
Iface: name of the interface that is listed.

Thus we can observe that, in the ARP table of h1, the entry corresponds to h2, and similarly for h2, the entry corresponds to h1.

For both the entries, the **Flag Mask is of type 'C'** which means the entries are dynamically learned by the ARP protocol.

TCP throughput from h1 to h2:
Throughput tells us how much data was transferred from a source at any given time.
For testing the TCP and analyzing the TCP throughput, the **iperf command** is used as follows:

```
mininet> iperf h1 h2
*** Iperf: testing TCP bandwidth between h1 and h2
*** Results: ['48.8 Gbits/sec', '48.9 Gbits/sec']
```

In the Results we can observe:
First result: Bandwidth output from the iperf server
Second result: Bandwidth output from the iperf client
Bandwidth tells you how much data *could* theoretically be transferred from a source at any given time.

Thus we can observe the two bandwidths as **48.8 Gbits/sec** and **48.9 Gbits/sec.** The bandwidth is often limited by **TCP's self-clocking and congestion control algorithm.**