CS301 ASSIGNMENT-1

Prajapati Harsh Pareshkumar
12241300

[Solution 1]

1.) **ifconfig** is used to configure and view the current status of the network interfaces in linux operating systems.
In the result below, We can see 4 network interfaces in my machine.
Docker0 : This interface is used for container communications.
Eth0 : This interface is used for wired network connection.
Lo : This interface is used to use services on localhost.
Wlan0 : This interface is used for wireless local area network connection.

```
┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ ifconfig
docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:21:3c:f3:9c  txqueuelen 0  (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

eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 9c:2d:cd:16:48:ef  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 0×10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 102860  bytes 40737480 (38.8 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 102860  bytes 40737480 (38.8 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.10.214.150  netmask 255.255.254.0  broadcast 10.10.215.255
        inet6 fe80::b16:4f38:e1ea:a650  prefixlen 64  scopeid 0×20<link>
        ether 38:7a:0e:c2:74:b4  txqueuelen 1000  (Ethernet)
        RX packets 1108  bytes 268462 (262.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 765  bytes 107903 (105.3 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

The output of the command shows the network interfaces which are up on the host.There are multiple tuples in the network interfaces like mtu, inet (IPv4), netmask, broadcast, inet6 (IPv6), ether (MAC Address) and etc.

MTU(Maximum Transmission Unit) : This shows the maximum size of the packet which we can transmit over an interface.

inet(IP address version 4)/inet6(IP address version 6) : inet is the 32 bit IP (Internet Protocol) address assigned to the host by the local network to uniquely identify the host on the network and inet6 is the upgraded version of IP address version 4.

Netmask : Netmask divides the 32 bit address into network and host portion.1's denote the network part and 0's denote the host part of the IP address.

Broadcast : This address is used to broadcast over a network interface.

Ether : This shows the Physical address/MAC(Media Access Control) Address of the host.

2.) We can use up, down, <interface name>, name, some flags, mtu, netmask and some other options with the ifconfig command.

Some options are shown below.

```
┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ ifconfig -a
br-93696f1cfe85: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        inet 172.18.0.1  netmask 255.255.0.0  broadcast 172.18.255.255
        ether 02:42:45:4f:3a:9a  txqueuelen 0  (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

docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:21:3c:f3:9c  txqueuelen 0  (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

eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 9c:2d:cd:16:48:ef  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 0×10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 114726  bytes 44760123 (42.6 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 114726  bytes 44760123 (42.6 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.10.214.150  netmask 255.255.254.0  broadcast 10.10.215.255
        inet6 fe80::b16:4f38:e1ea:a650  prefixlen 64  scopeid 0×20<link>
        ether 38:7a:0e:c2:74:b4  txqueuelen 1000  (Ethernet)
        RX packets 1194  bytes 294529 (287.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 874  bytes 125326 (122.3 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

ifconfig -a command lists all the network interfaces on the host whether it is up or down.

In the above figure, we can see the interface br-93696f1cfe85 which was down and not shown in the previous output.

We can also use **-s** flag for shortlist output and **-v** flag for verbose output with the ifconfig command.

```
┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ ifconfig wlan0
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.10.214.150  netmask 255.255.254.0  broadcast 10.10.215.255
        inet6 fe80::b16:4f38:e1ea:a650  prefixlen 64  scopeid 0x20<link>
        ether 38:7a:0e:c2:74:b4  txqueuelen 1000  (Ethernet)
        RX packets 1242  bytes 304433 (297.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 927  bytes 131754 (128.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0


┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ sudo ifconfig wlan0 10.10.215.171

┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ ifconfig wlan0
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.10.215.171  netmask 255.0.0.0  broadcast 10.255.255.255
        inet6 fe80::b16:4f38:e1ea:a650  prefixlen 64  scopeid 0x20<link>
        ether 38:7a:0e:c2:74:b4  txqueuelen 1000  (Ethernet)
        RX packets 1250  bytes 305069 (297.9 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 935  bytes 132540 (129.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

We can change our machine's inet address using the **ifconfig <interface> <IP address>** command.

Now, just like changing IP addresses, We can also change the MAC(Media Access Control)/Physical address of our host machine for the network interface with **ifconfig <interface> hw ether <address_you_want>.**

```
┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ ifconfig wlan0
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 2000
        inet 10.10.214.150  netmask 255.255.254.0  broadcast 10.10.215.255
        inet6 fe80::b16:4f38:e1ea:a650  prefixlen 64  scopeid 0×20<link>
        ether 38:7a:0e:c2:74:b4  txqueuelen 1000  (Ethernet)
        RX packets 7106  bytes 5378241 (5.1 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2872  bytes 708733 (692.1 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0


┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ sudo ifconfig wlan0 down

┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ sudo ifconfig wlan0 hw ether 00:11:22:33:44:55

┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ sudo ifconfig wlan0 up

┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ ifconfig wlan0
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 2000
        inet 10.10.69.80  netmask 255.255.240.0  broadcast 10.10.79.255
        inet6 fe80::b16:4f38:e1ea:a650  prefixlen 64  scopeid 0×20<link>
        ether 00:11:22:33:44:55  txqueuelen 1000  (Ethernet)
        RX packets 7138  bytes 5384688 (5.1 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2915  bytes 713447 (696.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

We can change the name of the network interface in our machine as we want. This can be done by the command **ifconfig <interface> name <Chosen_name>**.

```
┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ sudo ifconfig wlan0 name wifi

┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ ifconfig
docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:21:3c:f3:9c  txqueuelen 0  (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

eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 9c:2d:cd:16:48:ef  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 0×10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 148265  bytes 58559096 (55.8 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 148265  bytes 58559096 (55.8 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wifi: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.10.215.171  netmask 255.0.0.0  broadcast 10.255.255.255
        inet6 fe80::b16:4f38:e1ea:a650  prefixlen 64  scopeid 0×20<link>
        ether 38:7a:0e:c2:74:b4  txqueuelen 1000  (Ethernet)
        RX packets 1328  bytes 332917 (325.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1014  bytes 144497 (141.1 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

[Solution 2]

1.) The **ping** is used to test the connectivity between our host device and the target device.It sends ICMP(Internet Control Message Protocol) Echo requests to the destination whether it is reachable or not and reports the time of packets to reach the destination and return back .

2.) (a)

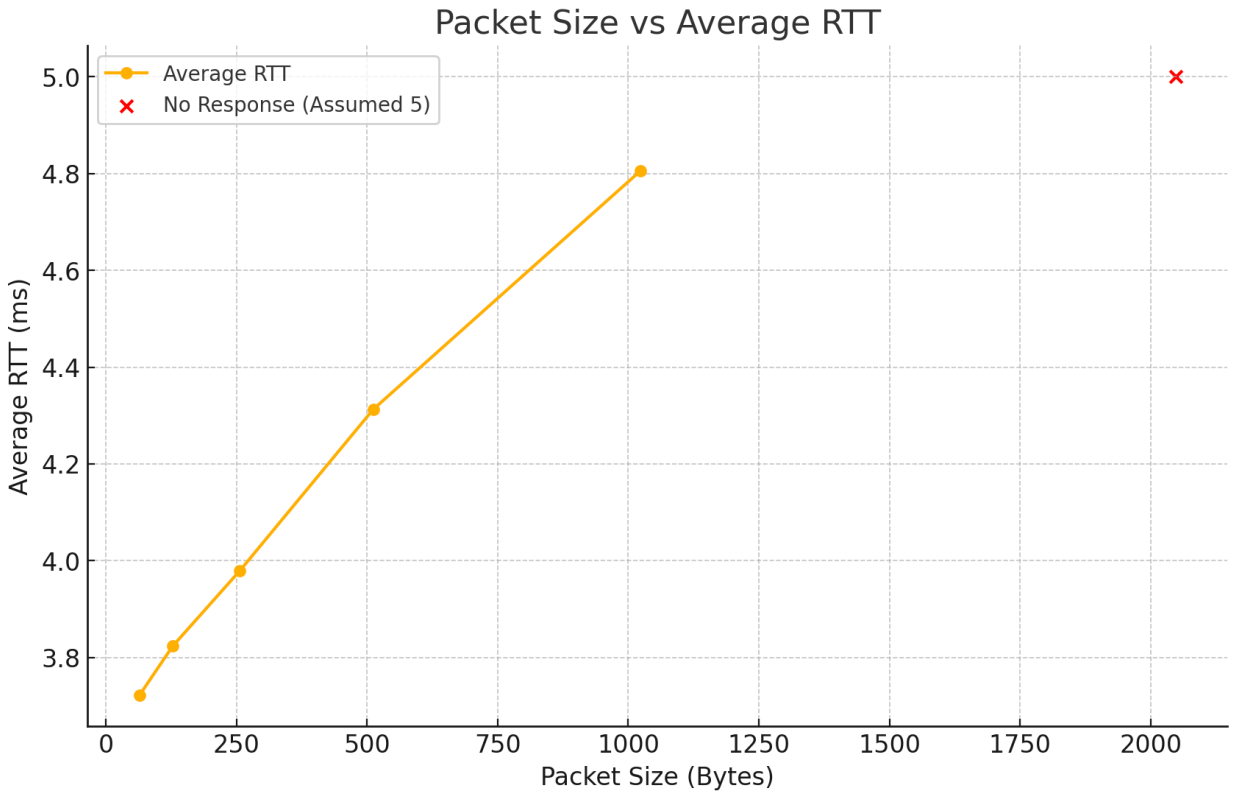
	RTT1	RTT2	RTT3
IIT BHILAI	4.310	5.336	2.067
IIT DELHI	59.653	61.397	58.184
GOOGLE	26.593	21.311	21.882

After looking at the above results, we can define the correlation between RTT and geographical distance between source and destination that **RTT increases as the distance between source and destination increases**.
Among the three hosts,**IIT Bhilai's server is nearest** to me hence it has shown the least average RTT all the time.
IIT Delhi's server is farther than IIT Bhilai which has shown higher RTT.
Google has deployed its servers in many countries to provide better service that's why it has shown lower RTT than IIT Delhi or it may happen due to use of advanced technology.

(b)

Packet Size	Average RTT
64	3.722
128	3.824
256	3.979
512	4.313
1024	4.806
2048	No Response

I was unable to get response for the packets of the size 2048 bytes since most of the network interfaces have MTU Limit of 1500 Bytes.Also There may be ICMP payload limitations or it may happen due to firewall restrictions for the packets.



(c)

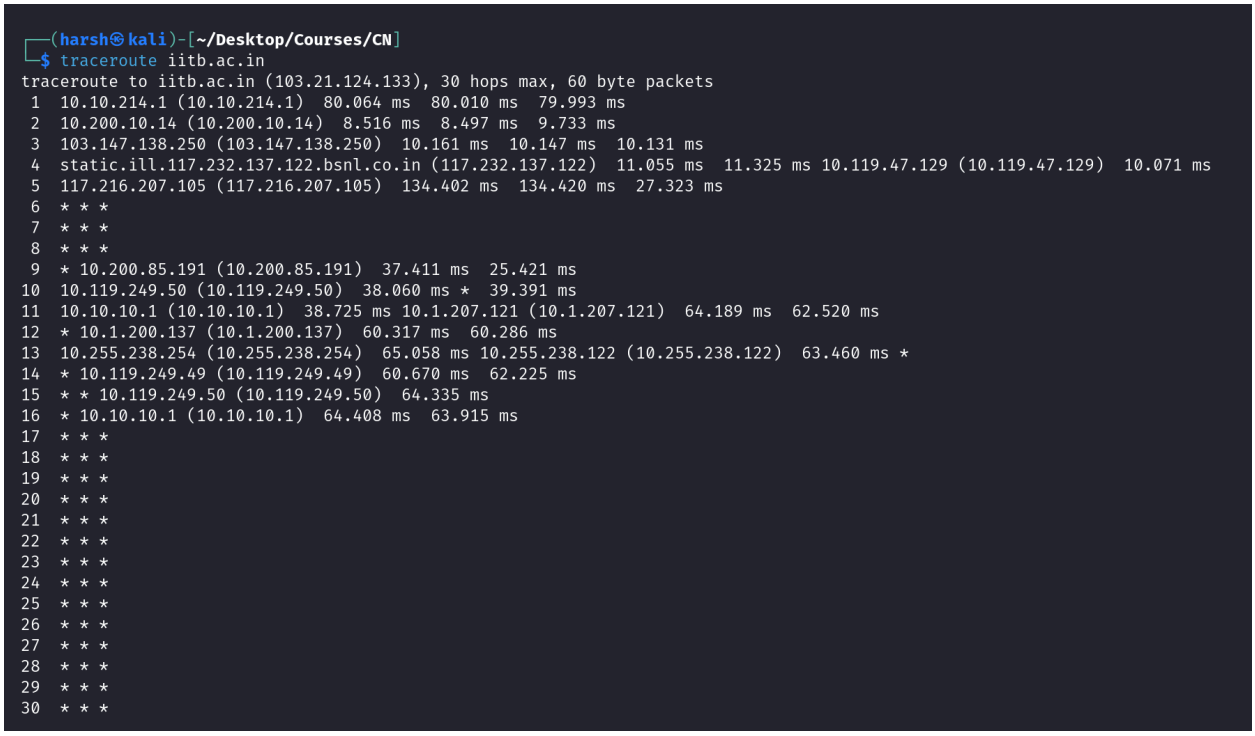
Impact of Packet Size on RTT : Generally, Larger packet size can result into higher RTT due to transmission delay. Sometimes it does packet fragmentation if the packet's size exceeds MTU. Whereas, smaller packets can be transmitted faster.But it may vary based on the traffic/congestion on the network at that point in time.

Impact of Time on RTT : Peak Hours of the day/High Network Usage may lead to high RTT because of congestion,some transmission delays and packet loss.When Network traffic is high, Queuing Delay may lead to frequent packet drops which can cause high RTT.In Off-Peak Hours, average RTT will be lesser than Peak Hours.

[Solution 3]

1.) The **traceroute** command is used to trace the path of the packets to reach the specific destination and also it can be used to **monitor the packet loss** or the **delays** on the specific route.

2.)



In the above image, We tried to reach IIT Bombay's Website and trace the path of the packets but after reaching to the host **10.10.10.1** , we got all the responses as *** * ***.

This can happen in some cases :

(i) Asymmetric Path : Sometimes The return path of teh packet may vary from the outgoing path which causes packet loss.

(ii) Firewalls/Security : Packets are blocked by Firewall.

(iii) Rate Limiting : Sometimes it can be dropped by the router for the prevention of Network form DoS Attack.

(iv) Network Congestion : If there is high traffic on the network,packets can also be dropped.

3.) Yes, it is possible to find the route to a specific host which fails to respond to the ping command.

Reason :

Ping command uses ICMP Echo Request Messages and expects the destination to return an ICMP Echo Reply.There are some cases possible where ping command does not work but traceroute can.

(i) Some hosts may be configured to block ICMP Echo requests but can respond to other network traffic.

(ii) Some routers or firewalls may block ICMP requests but respond to TCP or UDP which can give the path to the destination using traceroute command.

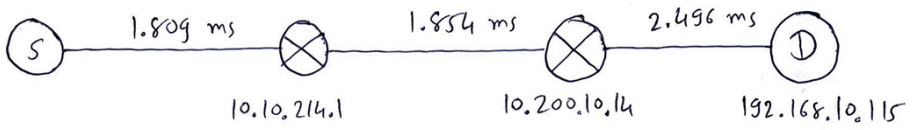
4.) The path traced by the traceroute command for Three hosts is given below.

```
┌──(harsh㉿kali)-[~/Desktop/Courses/CN]
└─$ traceroute iitbhilai.ac.in
traceroute to iitbhilai.ac.in (192.168.10.115), 30 hops max, 60 byte packets
 1  10.10.214.1 (10.10.214.1)  1.809 ms  1.977 ms  1.902 ms
 2  10.200.10.14 (10.200.10.14)  1.853 ms  1.834 ms  1.814 ms
 3  192.168.10.115 (192.168.10.115)  2.510 ms  2.496 ms  2.480 ms
```

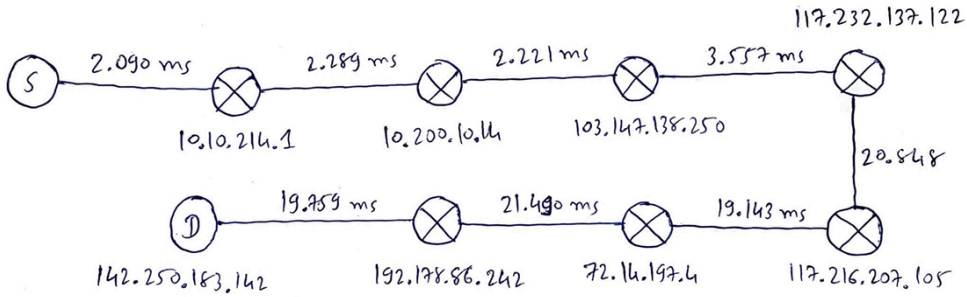
```
┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ traceroute google.com
traceroute to google.com (142.250.183.142), 30 hops max, 60 byte packets
 1  10.10.214.1 (10.10.214.1)  23.864 ms  25.388 ms  25.370 ms
 2  10.200.10.14 (10.200.10.14)  25.345 ms  25.155 ms  26.032 ms
 3  103.147.138.250 (103.147.138.250)  26.122 ms  26.105 ms  26.086 ms
 4  static.ill.117.232.137.122.bsnl.co.in (117.232.137.122)  26.568 ms  26.550 ms  26.534 ms
 5  117.216.207.105 (117.216.207.105)  78.012 ms  77.995 ms  77.977 ms
 6  72.14.197.4 (72.14.197.4)  77.211 ms  42.735 ms  40.890 ms
 7  * * *
 8  216.239.47.148 (216.239.47.148)  41.936 ms 216.239.54.84 (216.239.54.84)  41.915 ms 74.125.251.132 (74.125.251.132)  44.633 ms
 9  142.250.214.113 (142.250.214.113)  41.894 ms 192.178.110.204 (192.178.110.204)  40.723 ms 192.178.110.106 (192.178.110.106)  43.569 ms
10  192.178.110.245 (192.178.110.245)  40.690 ms bom07s31-in-f14.1e100.net (142.250.183.142)  39.205 ms  21.195 ms
```

```
┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ traceroute kali.org
traceroute to kali.org (104.18.4.159), 30 hops max, 60 byte packets
 1  10.10.214.1 (10.10.214.1)  6.535 ms  6.457 ms  7.580 ms
 2  10.200.10.14 (10.200.10.14)  7.575 ms  7.549 ms  7.525 ms
 3  103.147.138.250 (103.147.138.250)  7.565 ms  7.541 ms  7.519 ms
 4  10.119.47.129 (10.119.47.129)  8.884 ms static.ill.117.232.137.122.bsnl.co.in (117.232.137.122)  8.865 ms 10.119.47.129 (10.119.47.129)  7.363 ms
 5  * 117.216.207.105 (117.216.207.105)  22.340 ms *
 6  * * *
 7  * * *
 8  * * *
 9  115.247.69.85 (115.247.69.85)  51.342 ms  51.323 ms  50.094 ms
10  * 117.216.207.107 (117.216.207.107)  19.288 ms *
11  * 162.158.226.122 (162.158.226.122)  49.559 ms 103.27.170.48 (103.27.170.48)  20.899 ms
12  162.158.226.77 (162.158.226.77)  33.526 ms 162.158.226.79 (162.158.226.79)  21.486 ms  21.195 ms
13  104.18.4.159 (104.18.4.159)  21.798 ms  33.962 ms  34.326 ms
```

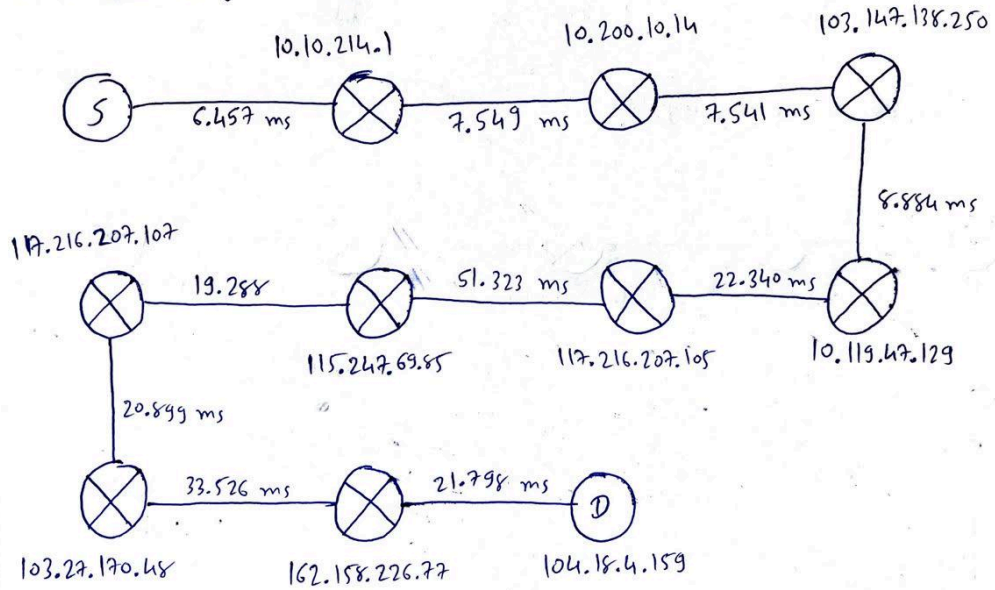
(i) iitbhilai.ac.in (192.162.10.115)



(ii) google.com (142.250.183.142)



(iii) kali.org (104.18.4.159)



[Solution 4]

1.) **Nmap** is the widely used network scanning tool for **Host Discovery** and **Service Discovery** on the Network. It can also be used to **identify open ports**, **running services**, and **potential security vulnerabilities**.

Host Discovery : To discover which hosts are up and down on the network.
Service Discovery : Which services are running on the hosts which are up.
Port Scanning : To identify if there are any unnecessary ports open on the hosts which may lead to an attack on the host.

2.) We can run nmap iitbhilai.ac.in to find open ports on the network.
Here, we will get the list of all open ports on the server.
Now, we can use nmap command with -sV flag to scan for the Versions of the running services on the host.
We will scan with command **nmap -sV iitbhilai.ac.in** .which is shown below.

```
┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ sudo nmap -sV iitbhilai.ac.in
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-14 22:33 IST
Nmap scan report for iitbhilai.ac.in (192.168.10.115)
Host is up (0.019s latency).
Not shown: 996 closed tcp ports (reset)
PORT     STATE SERVICE    VERSION
22/tcp   open  ssh        OpenSSH 7.4 (protocol 2.0)
80/tcp   open  http       Apache httpd 2.4.6 ((CentOS) OpenSSL/1.0.2k-fips PHP/5.4.16)
443/tcp  open  ssl/http   Apache httpd 2.4.6 ((CentOS) OpenSSL/1.0.2k-fips PHP/5.4.16)
5666/tcp open  tcpwrapped

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 13.85 seconds
```

We can see from the above image that there are 4 TCP ports in open state among which 3 are the standard TCP ports.
Nmap has also tried to determine the version of the service running on each open port.

Port 22 (SSH)
Service : SSH (Secure Shell)
Version : OpenSSH 7.4 (protocol 2.0)
Usage : It allows secure remote login and command execution.

Port 80 (HTTP)
Service : Apache HTTP Server
Version : Apache httpd 2.4.6
Usage : This port is used for serving web pages over HTTP.

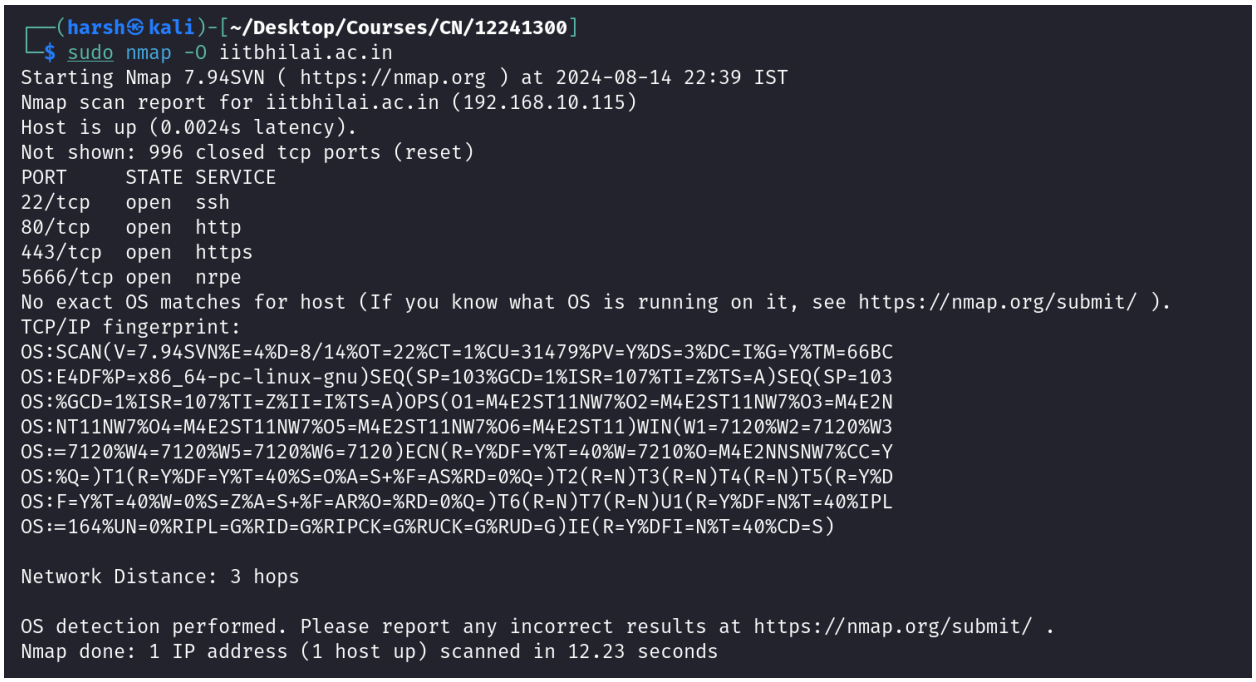
Port 443 (HTTPS)

Service : Apache HTTP Server

Version : Apache httpd 2.4.6

Usage : This port is used for serving web pages over HTTPS, which is the secure version of HTTP since it uses Encryption for Data Transmission.

3.) We can use the **'-O'** flag with the nmap command to determine the OS running on the host.
Command : **nmap -O <hostname>**



As we can see in the image, OS detection using nmap did not give any proper idea about OS match.But from the open ports like 22, 80, 443, we can guess that the host must be using a Unix-based operating system.

But to figure out which OS is actually running on the server, I used the Netcat command.
In this experiment,I connected to the server on open HTTP port using **nc iitbhilai.ac.in 80** .
After establishing connection, I sent an HTTP request to the server but it was not a valid HTTP request hence the server sent an acknowledgement for Bad Request but in the packet header it also sent some useful information about OS and Content-length and etc.

This has given the idea about the server that it is using **CentOS version 2.4.6**.

```
┌──(prajapati㉿kali)-[~/CN_12241300]
└─$ nc iitbhilai.ac.in 80
asdf
HTTP/1.1 400 Bad Request
Date: Mon, 19 Aug 2024 12:37:34 GMT
Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips PHP/5.4.16
Content-Length: 226
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>400 Bad Request</title>
</head><body>
<h1>Bad Request</h1>
<p>Your browser sent a request that this server could not understand.<br />
</p>
</body></html>
```

4.) (a)host : www.iitd.ac.in

```
┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ sudo nmap -sV www.iitd.ac.in
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-14 22:47 IST
Nmap scan report for www.iitd.ac.in (103.27.9.24)
Host is up (0.049s latency).
Other addresses for www.iitd.ac.in (not scanned): 2001:df4:e000:29::212
Not shown: 997 filtered tcp ports (no-response)
PORT     STATE  SERVICE  VERSION
80/tcp   open   http     Apache httpd
443/tcp  open   ssl/http Apache httpd
1723/tcp closed pptp

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 23.13 seconds
```

Open Ports : 80 (Version : Apache httpd) , 443 (Version : Apache httpd)

```
┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ sudo nmap -O --osscan-limit www.iitd.ac.in
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-14 22:57 IST
Nmap scan report for www.iitd.ac.in (103.27.9.24)
Host is up (0.051s latency).
Other addresses for www.iitd.ac.in (not scanned): 2001:df4:e000:29::212
Not shown: 997 filtered tcp ports (no-response)
PORT     STATE  SERVICE
80/tcp   open   http
443/tcp  open   https
1723/tcp closed pptp
Device type: general purpose
Running (JUST GUESSING): FreeBSD 6.X (85%)
OS CPE: cpe:/o:freebsd:freebsd:6.2
Aggressive OS guesses: FreeBSD 6.2-RELEASE (85%)
No exact OS matches for host (test conditions non-ideal).

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.76 seconds
```

OS Detection : No matches found using -O flag hence we used **–osscan-limit.**

–osscan-limit flag is used when there is at least one open TCP port and one closed TCP port found.
This guessed that the server is using **FreeBSD 6.X OS with 85% probability**.

I tried the same method using netcat command to find the OS but it did not work.

(b) host: google.com

```
┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ sudo nmap -sV google.com
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-14 23:02 IST
Nmap scan report for google.com (142.250.183.142)
Host is up (0.021s latency).
Other addresses for google.com (not scanned): 2404:6800:4009:824::200e
rDNS record for 142.250.183.142: bom07s31-in-f14.1e100.net
Not shown: 998 filtered tcp ports (no-response)
PORT    STATE SERVICE   VERSION
80/tcp  open  http      gws
443/tcp open  ssl/https gws
```

Open Ports : 80 (Version : gws), 443 (Version : gws)

```
┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ sudo nmap -O google.com
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-14 23:06 IST
Nmap scan report for google.com (142.250.183.142)
Host is up (0.021s latency).
Other addresses for google.com (not scanned): 2404:6800:4009:824::200e
rDNS record for 142.250.183.142: bom07s31-in-f14.1e100.net
Not shown: 998 filtered tcp ports (no-response)
PORT    STATE SERVICE
80/tcp  open  http
443/tcp open  https
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
OS fingerprint not ideal because: Missing a closed TCP port so results incomplete
No OS matches for host

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.79 seconds
```

OS Detection : No matches found
It could not get any reliable guess since there was not any closed TCP port.

Connection on the HTTP port using netcat did not give any idea about OS,too.

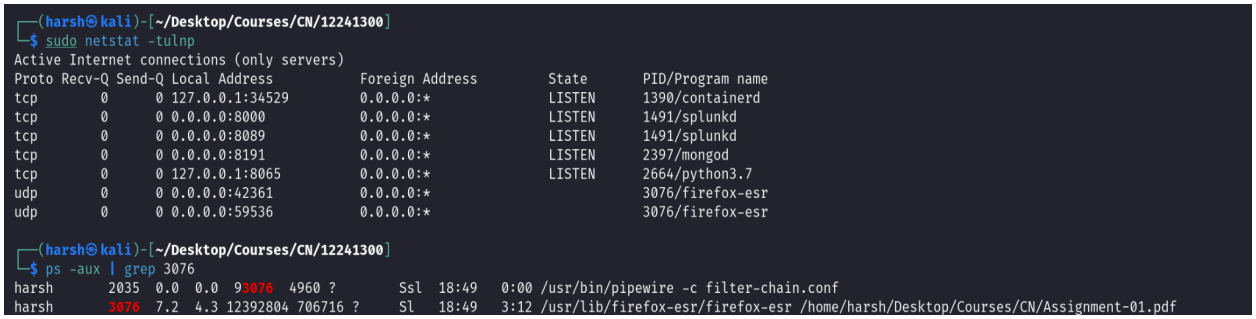
[Solution 5]

1.) **Netstat** is the network utility tool used to display network connections, routing tables, interface statistics, masquerade connections, and multicast memberships.

2.) To find all the active TCP ports on the system,
Run, **netstat -tn** command.
Where -t is used for tcp ports and -n for showing the numeric address of the hosts.

```
┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ sudo netstat -tn
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.0.1:56842         127.0.0.1:8191          ESTABLISHED
tcp        0      0 127.0.0.1:8191          127.0.0.1:56826         ESTABLISHED
tcp        0      0 10.10.214.150:45758     142.250.66.14:443       ESTABLISHED
tcp        0      0 10.10.214.150:39300     172.217.160.170:443     ESTABLISHED
tcp        0      0 127.0.0.1:8191          127.0.0.1:56754         ESTABLISHED
tcp        0      0 127.0.0.1:8191          127.0.0.1:56856         ESTABLISHED
tcp        0      0 127.0.0.1:8191          127.0.0.1:56858         ESTABLISHED
tcp        0      0 127.0.0.1:8191          127.0.0.1:56828         ESTABLISHED
tcp        0      0 127.0.0.1:8191          127.0.0.1:56770         ESTABLISHED
tcp        0      0 10.10.214.150:48904     104.18.32.47:443        ESTABLISHED
tcp        0      0 127.0.0.1:8191          127.0.0.1:56794         ESTABLISHED
tcp        0      0 127.0.0.1:56828         127.0.0.1:8191          ESTABLISHED
tcp        0      0 127.0.0.1:8191          127.0.0.1:56872         ESTABLISHED
tcp        0      0 127.0.0.1:56856         127.0.0.1:8191          ESTABLISHED
tcp        0      0 127.0.0.1:56882         127.0.0.1:8191          ESTABLISHED
tcp        0      0 127.0.0.1:56858         127.0.0.1:8191          ESTABLISHED
tcp        0      0 10.10.214.150:59196     142.251.42.42:443       ESTABLISHED
tcp        0      0 127.0.0.1:56854         127.0.0.1:8191          ESTABLISHED
tcp        0      0 127.0.0.1:56872         127.0.0.1:8191          ESTABLISHED
tcp        0      0 127.0.0.1:8191          127.0.0.1:56854         ESTABLISHED
tcp        0      0 10.10.214.150:35520     216.58.196.69:443       ESTABLISHED
tcp        0      0 10.10.214.150:43556     142.250.183.67:443      ESTABLISHED
tcp        0      0 10.10.214.150:52240     34.117.188.166:443      ESTABLISHED
tcp        0      0 10.10.214.150:48906     104.18.32.47:443        TIME_WAIT
tcp        0      0 127.0.0.1:8191          127.0.0.1:56788         ESTABLISHED
tcp        0      0 10.10.214.150:53074     104.18.32.47:443        ESTABLISHED
tcp        0      0 127.0.0.1:56788         127.0.0.1:8191          ESTABLISHED
tcp        0      0 10.10.214.150:43402     142.250.67.234:443      ESTABLISHED
tcp        0      0 127.0.0.1:56826         127.0.0.1:8191          ESTABLISHED
tcp        0      0 10.10.214.150:59508     142.250.183.67:443      ESTABLISHED
tcp        0      0 127.0.0.1:56770         127.0.0.1:8191          ESTABLISHED
tcp        0      0 127.0.0.1:56754         127.0.0.1:8191          ESTABLISHED
tcp        0      0 10.10.214.150:41778     172.64.41.4:443         ESTABLISHED
tcp        0      0 127.0.0.1:8191          127.0.0.1:56882         ESTABLISHED
tcp        0      0 127.0.0.1:56794         127.0.0.1:8191          ESTABLISHED
tcp        0      0 127.0.0.1:8191          127.0.0.1:56842         ESTABLISHED
```

To identify ports and PIDs for the web browser,
We use **netstat -tulnp** command.
Use of flags,
[-t] :: To show TCP ports
[-u] :: To show UDP ports
[-l] :: To show all the ports which are listening
[-n] :: To show numeric addresses of the hosts
[-p] :: To show the PID



In the above picture, we can see the services running on TCP and UDP along with PIDs and program name. To identify the port number and PID of the particular service, we used **ps -aux** command and then used **grep <PID>** command to show the process ID 3076.

This experiment has shown that we were running Assignment-01.pdf on the port number 59536 with PID 3076.

We used **-l flag** for getting the clear idea of the experiment by listing only listening ports to shorten the output and then we determined the particular process.
We would have been able to do it with the **-tunp flag** but the output of netstat command would have been unnecessarily long.

Now, let us find out if any of the services running on our system is using any of the standard ports from HTTP, DHCP, DNS, SMTP, and FTP.

Generally, HTTP runs on port 80, DHCP on 67/68 , DNS on 53, SMTP on 25 and FTP on 21.

For that we will run **sudo netstat -tunp | grep -E ':80|:67|:68|:53|:25|:21'** command.

```
┌──(harsh㉿kali)-[~]
└─$ sudo netstat -tunp | grep -E ':80|:67|:68|:53|:25|:21'
tcp        0      0 127.0.0.1:54648         127.0.0.1:8000          ESTABLISHED 2840/firefox-esr
tcp        0      0 127.0.0.1:54662         127.0.0.1:8000          ESTABLISHED 2840/firefox-esr
tcp        0      0 127.0.0.1:8000          127.0.0.1:54662         ESTABLISHED 1512/splunkd
tcp        0      0 127.0.0.1:8000          127.0.0.1:54644         ESTABLISHED 1512/splunkd
tcp        0      0 127.0.0.1:8000          127.0.0.1:54630         ESTABLISHED 1512/splunkd
tcp        0      0 127.0.0.1:54630         127.0.0.1:8000          ESTABLISHED 2840/firefox-esr
tcp        0      0 127.0.0.1:8000          127.0.0.1:54648         ESTABLISHED 1512/splunkd
tcp        0      0 127.0.0.1:54644         127.0.0.1:8000          ESTABLISHED 2840/firefox-esr
tcp        0      0 127.0.0.1:54672         127.0.0.1:8000          ESTABLISHED 2840/firefox-esr
tcp        0      0 127.0.0.1:8000          127.0.0.1:54672         ESTABLISHED 1512/splunkd
tcp6       0      0 127.0.0.1:80            127.0.0.1:47950         TIME_WAIT   -
udp        0      0 10.10.215.145:68        10.200.10.250:67        ESTABLISHED 1317/NetworkManager
```

The last connection in the above picture is using **ports 67 and 68** which shows that this service is running **DHCP**.
The last second connection shows an **HTTP connection on port 80**.
No services were running on SMTP, DNS and FTP protocols hence we couldn't find it in the above output.

3.) **netstat -su** command can show the statistics of all the UDP connections on the system.

```
┌──(harsh㉿kali)-[~/Desktop/Courses/CN/12241300]
└─$ netstat -su
IcmpMsg:
    InType3: 43
    OutType3: 41
Udp:
    6410 packets received
    41 packets to unknown port received
    153 packet receive errors
    3388 packets sent
    153 receive buffer errors
    0 send buffer errors
    IgnoredMulti: 9
UdpLite:
IpExt:
    OutMcastPkts: 4
    InBcastPkts: 9
    OutBcastPkts: 9
    InOctets: 72711932
    OutOctets: 53233060
    OutMcastOctets: 160
    InBcastOctets: 702
    OutBcastOctets: 702
    InNoECTPkts: 152090
MPTcpExt:
```