Subject : Software Engineering

Subject Code : CS3273

Section: Gx

Topic : Exploring Software Tool

Group -2

Assignment – 4

Enrollment no.	Name
2020CSB007	ARATRIKA PAL
2020CSB009	MAYANK MANAVENDRA KUMAR
2020CSB010	GOURAV KUMAR SHAW
2020CSB011	TAMOGHNA ROY
2020CSB013	ABDUL KHAZMUDDIN
2020CSB014	SIDDARTHA JANA

1. Purpose of the tool

Iperf is a tool used for measuring network performance by testing the bandwidth, throughput, and latency of a network connection. It can be used to test both TCP and UDP protocols. Client sends data to the server, the server receives the data and calculates bandwidth.

- Bandwidth - maximum rate of data transfer. In each second how much datawe send/receive
- Throughput - rate of message delivery over a communication channel.

2. The path for software download

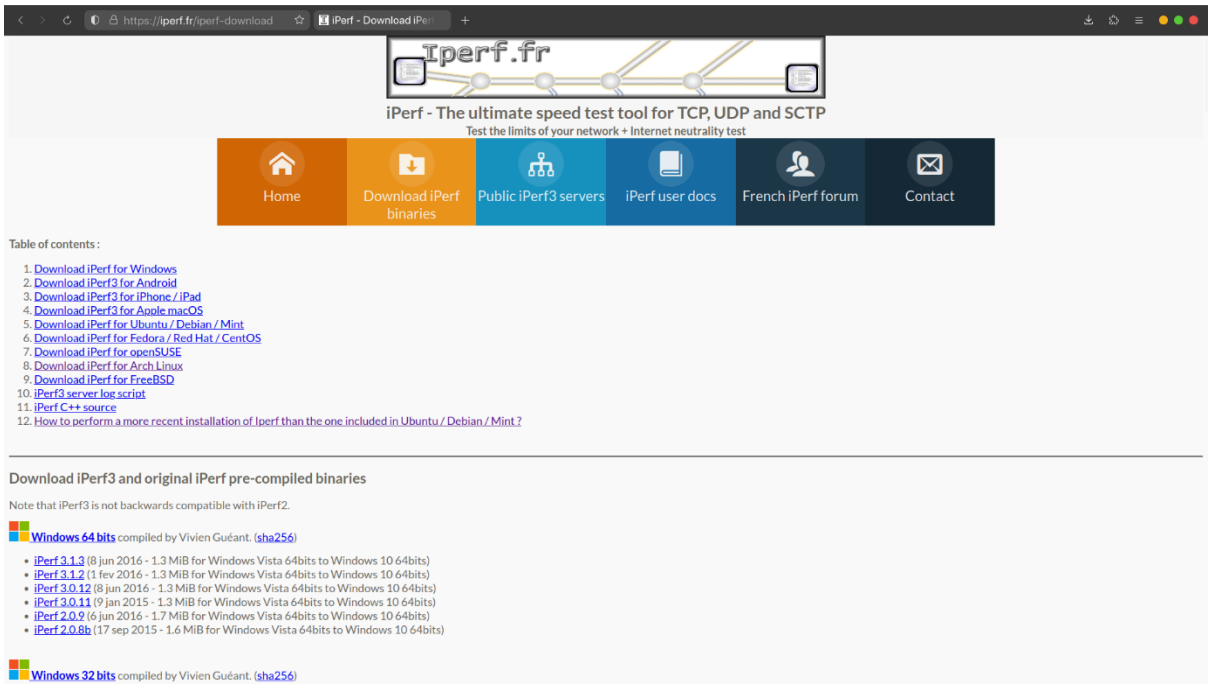
The iPerf software can be downloaded from the official iPerf website (https://iperf.fr/iperf-download.php). The website provides download links (to binaries) for various operating systems, including Windows, Linux, and macOS. The website also provides instructions for downloading and installing iPerf on each platform.

Windows/ macOS:

On the iPerf website, click on the download link for Windows. It will take you to a page where you can download the latest version of iPerf. The file will be in a .zip format, and you can extract it to a folder of your choice. To run iPerf , open the Terminal app (Command Prompt or PowerShell in the case for Windows) and navigate to the folder where you extracted

iPerf. Then use the iPerf commands to configure and run the tool.

To check out the most recent code, clone the git repository at: https://github.com/esnet/iperf.git .You may use the commands: "./configure; make; make install" to build from source(cloned / downloaded git repository).



When it comes to Linux distributions, iPerf can be installed on a variety of distributions including Ubuntu, Fedora, NixOs, and Arch-based distributions. Here is a brief overview of the installation process for each distribution that does not make use of binaries available on the aforementioned website:

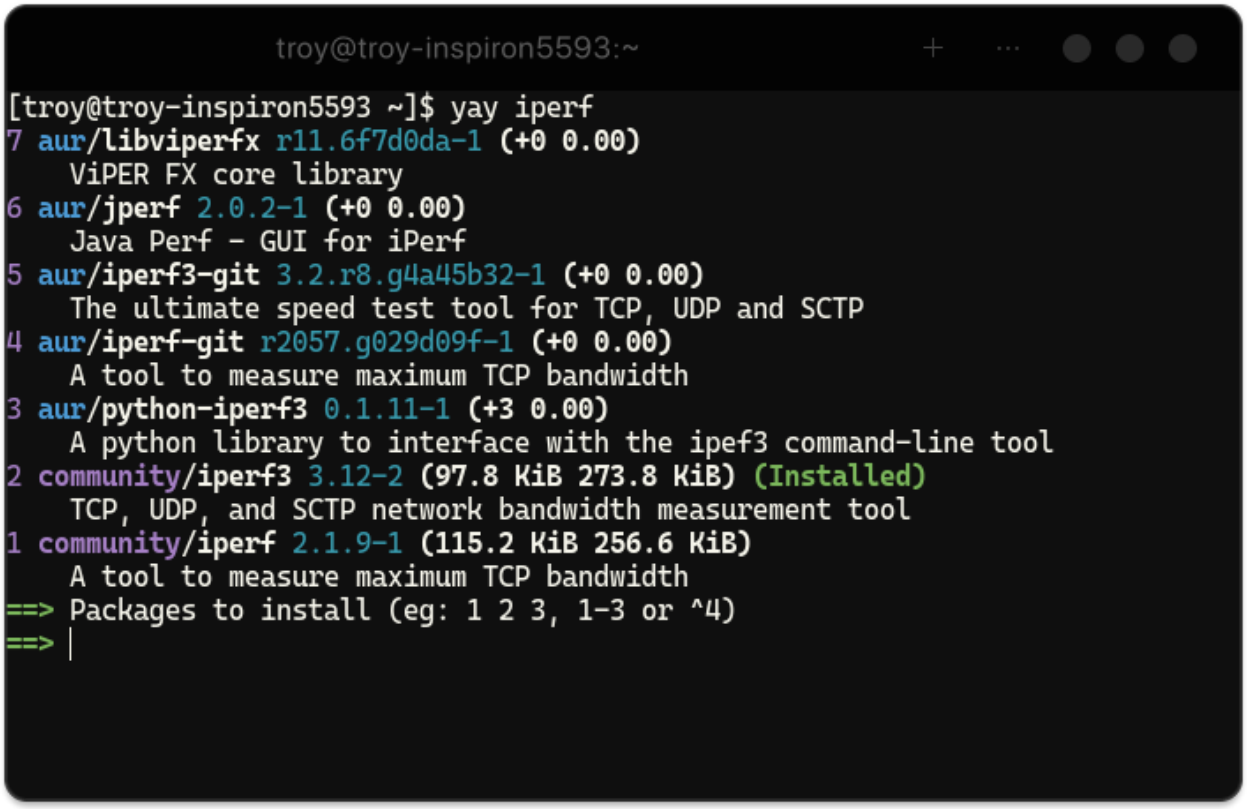
- Ubuntu: iPerf is available in the Ubuntu repository, and can be installed using the apt package manager. Open a

terminal and enter the command "**sudo apt-get install iperf**".

- Fedora: iPerf is available in the Fedora repository, and can be installed using the dnf package manager. Open a terminal and enter the command "**sudo dnf install iperf**".
- NixOS: iPerf is available in the Nix package manager, and thus can be installed using it . Open a terminal and enter the command "**nix-env -iA nixos.iperf**" for NixOS and "**nix-env -iA nixpkgs.iperf**" on Non NixOS distributions.
- Arch-based distributions: iPerf is available on the Arch community repository and can be installed either using the pacman package manager or an Arch User Repository helper such as yay / paru. Open a terminal and enter the command "**sudo pacman -S iperf**" or "**yay -S iperf**".

The downloaded binary may be present in /bin, usr/bin, /home/user/bin, etc.

It is worth noting that the installation process for iPerf may differ slightly between different Linux distributions. Users may need to consult the documentation or community forums for their specific distribution for more detailed installation instructions.



3. Target platform and installation procedure

TARGET PLATFORM:

Target Platform means the technical parameters of the system for which a piece of software

Is being developed including CPU, RAM, hard disk, storage, operating system, network and security configuration.

iPerf is a cross platform opensource tool(written in C) for active measurements of the maximum achievable bandwidth on IP networks. It supports tuning of various parameters

related to timing, buffers and protocols (TCP, UDP, SCTP with IPv4 and IPv6). For each test it reports the bandwidth, loss, and other parameters.

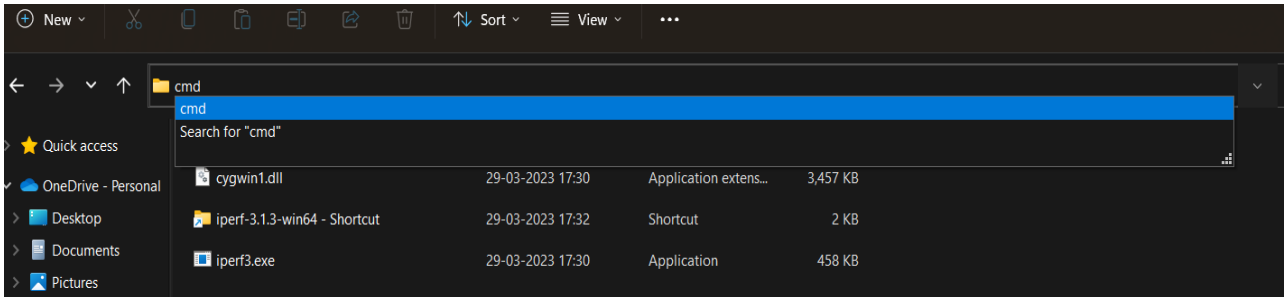
iPerf features

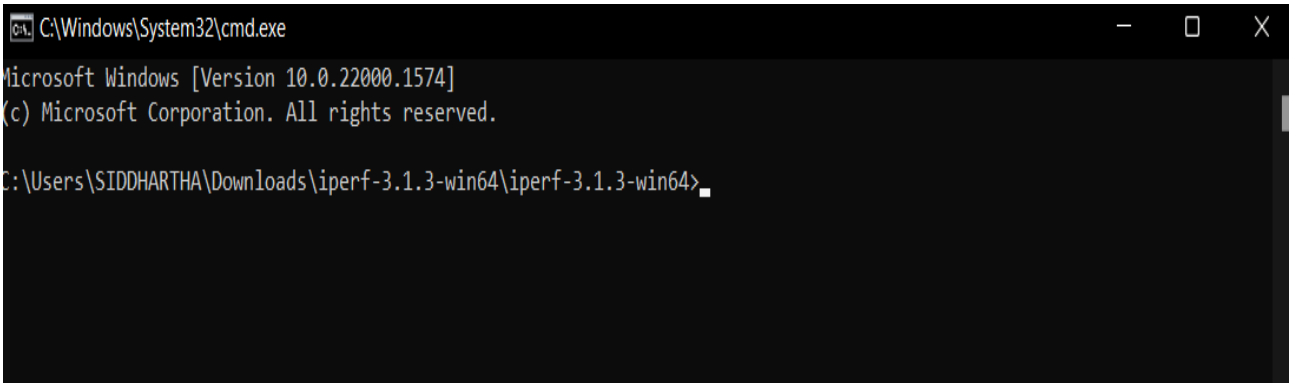
- TCP and SCTP
  - Measure bandwidth
  - Report MSS/MTU size and observed read sizes.
  - Support for TCP window size via socket buffers.
- UDP
  - Client can create UDP streams of specified bandwidth.
  - Measure packet loss
  - Measure delay jitter
  - Multicast capable
- Cross-platform: Windows, Linux, Android, MacOS X, FreeBSD, OpenBSD, NetBSD, Solaris,...
- Client and server can have multiple simultaneous connections (-P option).
- Server handles multiple connections, rather than quitting after a single test.

Installation Procedure:

After downloading the zip file go to File Explorer and in downloads extract the content.

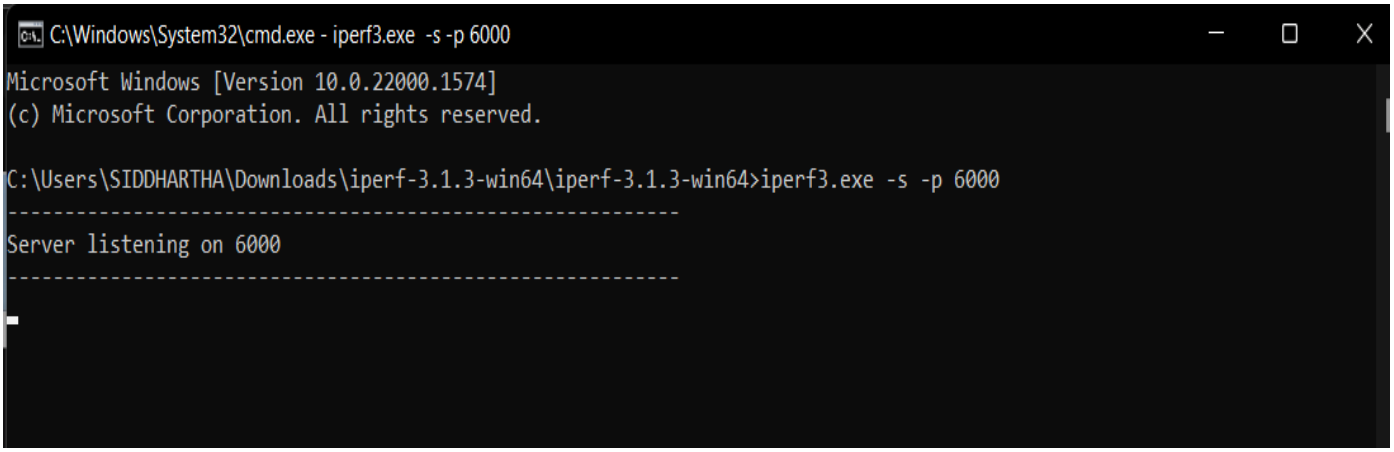
Navigate to the executable and in the name bar enter “cmd” and press enter.



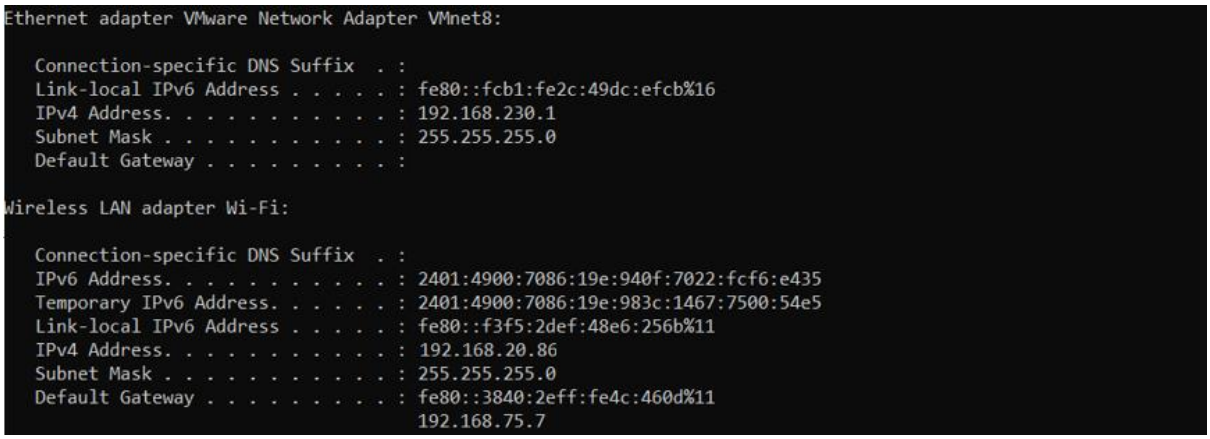


Then install the same tool in another computer which will be client.

Activate Iperf by assigning the desired port with this command **“iperf3.exe -s -p #port”.**



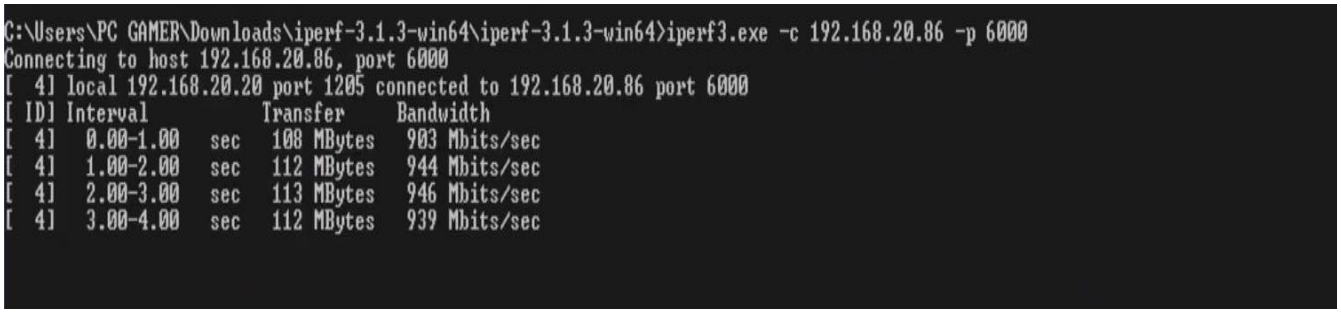
Open another CMD console and run “ipconfig” to know the IP address



Ip address of wi-fi adapter is 192.168.20.86

On the client, in CMD, check the network status with the remote server with this command:

“iperf3.exe -c IP_ADRRESS -p #port.”



4.Commands to configure and run the tool

Iperf is a widely used command-line tool for measuring network performance by generating TCP and UDP data streams and measuring the throughput, jitter, and packet loss between two endpoints. It can be used to test the bandwidth of a network connection, as well as to troubleshoot network issues. Iperf works in server client model that means there should be one server and one client to start Iperf traffic flow.

Here is a step-by-step guide on how to configure and run Iperf on a Linux machine:

1.Install Iperf:

We can install Iperf using the package manager of our Linux distribution.

For example, on Ubuntu, you can install it by running the following command:

```
aratrika@aratrika:~$ sudo apt-get install iperf
[sudo] password for aratrika:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  iperf
0 upgraded, 1 newly installed, 0 to remove and 234 not upgraded.
Need to get 121 kB of archives.
After this operation, 315 kB of additional disk space will be used.
Get:1 http://in.archive.ubuntu.com/ubuntu jammy/universe amd64 iperf amd64 2.1.5+dfsg1-1 [121 kB]
Fetched 121 kB in 2s (80.4 kB/s)
Selecting previously unselected package iperf.
(Reading database ... 212753 files and directories currently installed.)
Preparing to unpack .../iperf_2.1.5+dfsg1-1_amd64.deb ...
Unpacking iperf (2.1.5+dfsg1-1) ...
Setting up iperf (2.1.5+dfsg1-1) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for ufw (0.36.1-4build1) ...
aratrika@aratrika:~$
```

2.Choose the endpoint:

Decide which endpoint will act as the client and which one will act as the server. The server endpoint will listen for incoming connections, while the client endpoint will initiate a connection to the server.

3.Run the Iperf server:

On the server endpoint, run the following command to start the Iperf server in server mode:

```
aratrika@aratrika:~$ iperf -s
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size:  128 KByte (default)
------------------------------------------------------------
[  1] local 10.0.2.15 port 5001 connected with 10.0.2.15 port 34230
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-9.9997 sec  23.1 GBytes  19.8 Gbits/sec
[  2] local 10.0.2.15 port 5001 connected with 10.0.2.15 port 40690
[ ID] Interval       Transfer     Bandwidth
[  2] 0.0000-8.9539 sec  15.3 GBytes  14.7 Gbits/sec
```

This will start the Iperf server in default mode listening on TCP port 5001.

4.Run the Iperf client:

On the client endpoint, run the following command to initiate a connection to the server:

```
aratrika@aratrika:~$ iperf -c 10.0.2.15
------------------------------------------------------------
Client connecting to 10.0.2.15, TCP port 5001
TCP window size: 2.50 MByte (default)
------------------------------------------------------------
[  1] local 10.0.2.15 port 34230 connected with 10.0.2.15 port 5001
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-10.0026 sec  23.1 GBytes  19.8 Gbits/sec
```

This will start the Iperf client in default mode and send TCP traffic to the server.

5.Interpret the results:

Once the test is complete, Iperf will output the results, which include the throughput, jitter, and packet loss of the connection.

For example, the following output shows a test with a bandwidth of 100 Mbps:

```
[ ID] Interval       Transfer     Bandwidth
[  2] 0.0000-8.9539 sec  15.3 GBytes  14.7 Gbits/sec
```

This means that the connection has a throughput of 14.7Gbps.

We can also use different options and parameters to customize the Iperf test. Here are some examples:

To run the test for a specific duration, use the -t option:

```
^Caratrika@aratrika:~iperf -c 10.0.2.15 -t 30
------------------------------------------------------------
Client connecting to 10.0.2.15, TCP port 5001
TCP window size: 2.50 MByte (default)
------------------------------------------------------------
[  1] local 10.0.2.15 port 44842 connected with 10.0.2.15 port 5001
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-30.0039 sec  64.0 GBytes  18.3 Gbits/sec
```

This will run the test for 30 seconds.

To use UDP instead of TCP, use the -u option:

```
aratrika@aratrika:~$ iperf -c 10.0.2.15 -u
------------------------------------------------------------
Client connecting to 10.0.2.15, UDP port 5001
Sending 1470 byte datagrams, IPG target: 11215.21 us (kalman adjust)
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  1] local 10.0.2.15 port 48779 connected with 10.0.2.15 port 5001
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-10.0210 sec  1.25 MBytes  1.05 Mbits/sec
[  1] Sent 896 datagrams
[  1] Server Report:
[ ID] Interval           Transfer     Bandwidth        Jitter   Lost/Total  Latency avg/min/max/stdev PPS NetPwr
[  1] 0.0000-0.0000 sec  4.00 GBytes  -nan bits/sec   0.000 ms 4294966296/0 (0%) -/-/-/- ms 0 pps
```

To specify a different port number, use the -p option:

```
^Caratrika@aratrika:~/6th_Sem_CN$ iperf -s -p 1234
------------------------------------------------------------
Server listening on TCP port 1234
TCP window size:  128 KByte (default)
------------------------------------------------------------
```

This will start the Iperf server listening on port 1234.

```
aratrika@aratrika:~/6th_Sem_CN$ iperf -c 10.0.2.15 -u -p 1234
------------------------------------------------------------
Client connecting to 10.0.2.15, UDP port 1234
Sending 1470 byte datagrams, IPG target: 11215.21 us (kalman adjust)
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  1] local 10.0.2.15 port 33236 connected with 10.0.2.15 port 1234
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-10.0073 sec  1.25 MBytes  1.05 Mbits/sec
```

5. Case studies – the kind of experiments done using this tool.

Iperf is a commonly used tool for measuring network performance. It is an open-source tool that can be used to measure the bandwidth and the quality of a network connection. Iperf allows to generate and measure network traffic to test the performance of a network.

Here are a few examples of case studies that can be conducted using Iperf:

1. Testing network bandwidth:

Iperf can be used to measure the maximum available bandwidth of a network connection. We can use Iperf to generate network traffic between two hosts and measure the bandwidth. This test can help us to identify if there are any network bottlenecks that are affecting performance.

To test network bandwidth using Iperf, we can use the following command on the client machine:

- `iperf -c <server_ip_address> -t <test_duration> -i <interval> -P <num_threads>`

- This command will instruct Iperf to connect to the server at `<server_ip_address>`, run the test for `<test_duration>` seconds, report results every

`<interval>` seconds, and use `<num_threads>` threads to generate network traffic.

- For example, to test the bandwidth between a client machine with IP address `192.168.1.10` and a server machine with IP address `192.168.1.20` for 60 seconds, reporting results every 10 seconds, and using 4 threads, we can run the following command on the client machine:

- `iperf -c 192.168.1.20 -t 60 -i 10 -P 4`

- This test will measure the maximum available bandwidth between the client and server machines.

- Testing network bandwidth refers to the process of measuring the amount of data that can be transmitted over a network connection between two devices in a given amount of time.

- The bandwidth is usually measured in bits per second (bps), kilobits per second (Kbps), megabits per second (Mbps), or gigabits per second (Gbps).

```
gouravkr@Ubuntu:~/Desktop$ iperf -c 192.168.89.211 -t 60 -i 10 -P 2
------------------------------------------------------------
Client connecting to 192.168.89.211, TCP port 5001
TCP window size: 2.50 MByte (default)
------------------------------------------------------------
[  1] local 192.168.89.211 port 53818 connected with 192.168.89.211 port 5001
[  2] local 192.168.89.211 port 53832 connected with 192.168.89.211 port 5001
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-10.0000 sec  46.4 MBytes  38.9 Mbits/sec
[  2] 0.0000-10.0000 sec  2.50 MBytes  2.10 Mbits/sec
[SUM] 0.0000-10.0000 sec  48.9 MBytes  41.0 Mbits/sec
[  1] 10.0000-20.0000 sec  39.6 MBytes  33.2 Mbits/sec
[  2] 10.0000-20.0000 sec  0.000 Bytes  0.000 bits/sec
[SUM] 10.0000-20.0000 sec  39.6 MBytes  33.2 Mbits/sec
[  1] 20.0000-30.0000 sec  39.8 MBytes  33.3 Mbits/sec
[  2] 20.0000-30.0000 sec  0.000 Bytes  0.000 bits/sec
[SUM] 20.0000-30.0000 sec  39.8 MBytes  33.3 Mbits/sec
[  1] 30.0000-40.0000 sec  39.6 MBytes  33.2 Mbits/sec
[  2] 30.0000-40.0000 sec  0.000 Bytes  0.000 bits/sec
[SUM] 30.0000-40.0000 sec  39.6 MBytes  33.2 Mbits/sec
[  1] 40.0000-50.0000 sec  40.5 MBytes  34.0 Mbits/sec
[  2] 40.0000-50.0000 sec  0.000 Bytes  0.000 bits/sec
[SUM] 40.0000-50.0000 sec  40.5 MBytes  34.0 Mbits/sec
[  1] 50.0000-60.0000 sec  40.5 MBytes  34.0 Mbits/sec
[  2] 50.0000-60.0000 sec  0.000 Bytes  0.000 bits/sec
[SUM] 50.0000-60.0000 sec  40.5 MBytes  34.0 Mbits/sec
[  1] 60.0000-122.2212 sec   128 KBytes  16.9 Kbits/sec
[  1] 0.0000-122.2212 sec   247 MBytes  16.9 Mbits/sec
[  2] 0.0000-122.2205 sec  2.50 MBytes   172 Kbits/sec
[SUM] 0.0000-61.5414 sec   249 MBytes  33.9 Mbits/sec
[ CT] final connect times (min/avg/max/stdev) = 0.059/0.113/0.168/0.077 ms (tot/err) = 2/0
gouravkr@Ubuntu:~/Desktop$
```

2. Evaluating network performance under different conditions:

Iperf can be used to test network performance under different network conditions. For example,we can use Iperf to measure network performance when using different protocols, such as TCP and UDP, or when using different packet sizes.

- To evaluate network performance under different conditions using Iperf, we can change the test parameters to simulate different network conditions. For example, to test network performance with different protocols, we can use

the `-u` option to switch to UDP protocol instead of the default TCP protocol.

- UDP is useful for measuring network performance under conditions with high packet loss, while TCP is useful for measuring performance under normal network conditions.

- To run a UDP test for 60 seconds with a packet size of 1000 bytes and report results every 10 seconds, we can run the following command:

- `iperf -c 192.168.1.20 -u -b 100M -l 1000 -t 60 -i 10`

This test will measure the network performance using UDP protocol with a 100 Mbps bitrate and a 1000 byte packet size.

```
gouravkr@Ubuntu:~/Desktop$ iperf -c 192.168.89.211 -u -b 100M -l 1000 -t 60 -i 10
------------------------------------------------------------
Client connecting to 192.168.89.211, UDP port 5001
Sending 1000 byte datagrams, IPG target: 76.29 us (kalman adjust)
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  1] local 192.168.89.211 port 51861 connected with 192.168.89.211 port 5001
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-10.0000 sec   125 MBytes   105 Mbits/sec
[  1] 10.0000-20.0000 sec   125 MBytes   105 Mbits/sec
[  1] 20.0000-30.0000 sec   125 MBytes   105 Mbits/sec
[  1] 30.0000-40.0000 sec   124 MBytes   104 Mbits/sec
[  1] 40.0000-50.0000 sec   126 MBytes   105 Mbits/sec
[  1] 50.0000-60.0000 sec   125 MBytes   105 Mbits/sec
[  1] 0.0000-60.0000 sec   750 MBytes   105 Mbits/sec
[  1] Sent 786433 datagrams
```

3. Testing network performance with different network configurations:

Iperf can also be used to test network performance with different network configurations. For example,we can use Iperf to test the *performance of a network* with different levels of congestion or with different routing configurations.

- To test network performance with different network configurations using Iperf,we can change the network configuration on the server machine and run the test again.
- For example, we can test the performance of a network with different levels of congestion by running the test during a period of high network traffic.
- To simulate a congested network, we can run a file transfer on the server machine during the Iperf test. To do this, we can use the following command on the server machine:
- `dd if=/dev/zero bs=1M count=1000 | nc -l 1234`

- This command creates a stream of 1000 MB (1 GB) of zeros using the `dd` command and sends it to port 1234 using `nc` (netcat) command. The `|` character pipes the output of the dd command as input to the `nc` command.

- The `nc` command listens on port 1234 and waits for a client to connect. Once a client is connected, it will send the stream of zeros to the client.

Running this command during the Iperf test will simulate a congested network and allow us to test network performance under these conditions.

4. Evaluating network performance with different hardware configurations:

Iperf can also be used to test network performance with different hardware configurations. For example, we can use Iperf to test the performance of a network with different network interface cards (NICs) or with different CPU configurations.

- To evaluate network performance with different hardware configurations using Iperf, we can change the hardware configuration on the client or server machine and run the test again.

- For example, we can test the performance of a network with different network interface cards (NICs) by switching the NIC on the server machine and running the test again.

- To switch the NIC on the server machine, we can use the following command on the server machine:
- `ifconfig eth0 down`
  `ifconfig eth1 up`

This command will disable the `eth0` NIC and enable the `eth1` NIC. Running the Iperf test again after this change will allowus to test network performance with the new NIC.

6. Drawback of the tool (if any)

● Only supports a limited number of protocols, including TCP, UDP, and SCTP. This can be a limitation if you need to test other protocols.

● primarily designed to test bandwidth between two endpoints, which may not accurately reflect real-world network scenarios. For example,
it may not account for network congestion or latency.

● NICs with limited buffer sizes may cause TCP throughput to drop.

● Iperf's performance is impacted by the size of the kernel buffer. The size of the buffer which depends on the OS impacts Iperf's performance for UDP tests.

7. Any other aspects you would like to cover those are useful to the context of the tool

Visualization: iperf can generate output in several formats, including plain text, CSV, JSON, and XML. The tool also supports visualization through tools like Grafana and InfluxDB, allowing users to create custom dashboards and charts to monitor network performance.