

## **Lab 5: Group Project: Fast, Reliable File Transfer**

**Suela Buzi**

**Esha Desai**

**Leon Aburime Leon**

**Harsh Gupta**

## Aim:

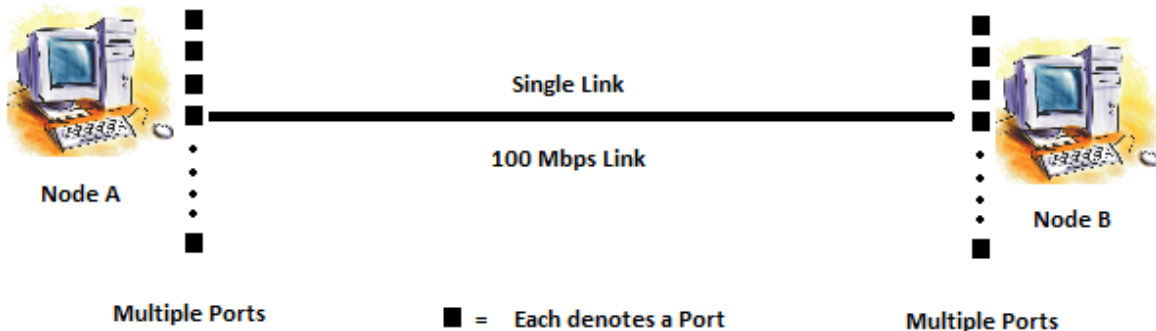
Our aim was to build a fast reliable file transfer protocol using the base of UDP protocol to exploit its advantage of fast transfer while still aiming for 100% reliability similar to TCP.

## Design:

We started implementing our designed protocol using **Python language**.

An overview of our design:

First, the client requests the name of the file it requires to the server. This request is like a connection request made to the server. The server uses multi processing on different sockets to send the file and the clients are communicating with their respective servers. Each file is splitted into the number of ports at the server each used for sending the file. So each server is handling equal number of packets for sending to the client. It can be visualized as shown below in the figure.



We have kept the maximum data size of payload to be 1400 bytes. We have reserved a 100 bytes for the header which contains the metadata information. So the total packet length is 1500 bytes. The metadata includes file information like the sequence number of packets, number of packets, datalength and file size. The number of packets is the splitted file size divided by 1400 bytes. We are using UDP sockets. The port number is passed in the 1<sup>st</sup> argument on the Command Line Interface.

First we start the server. The server waits on different ports for the client to make a request to it. The client waits in a while loop till it receives all the packets. Then these packets are sequentially written into their respective files on the disk. And later we are concatenating all these files into the file which is the same as the original file at the server.

## Observations:

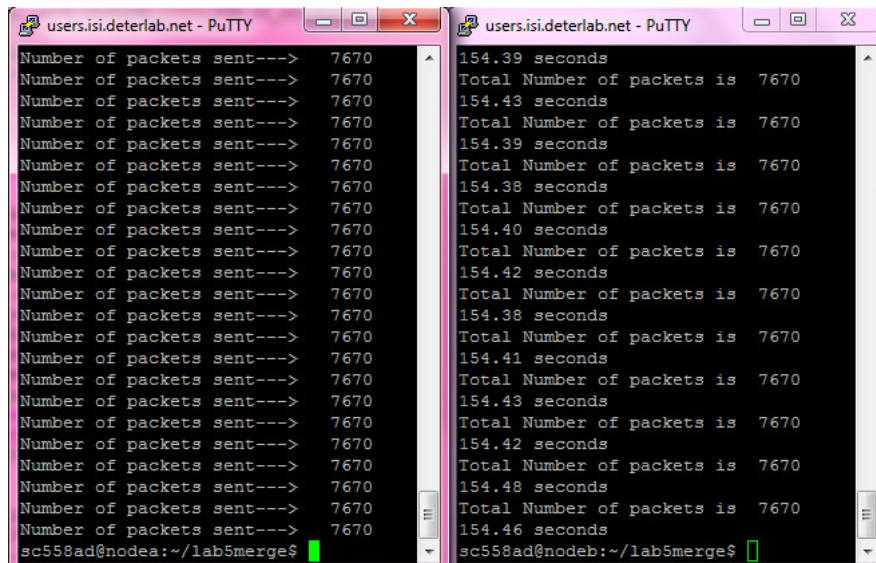
We are testing our code on a 100 Mbps link. The file which we used for our testing was 1073741824 bytes which is 1 Gigabyte file. The time on sending the first packet on the server side was noted and the time on receiving the last packet on the client side was noted and the total transferred time for the whole file to be received by the client turned out to be 154.6 seconds. The **throughput** at 0ms delay was calculated as  $(1073741824 * 8 / 154.4) =$  **"55.664Mbps."** We are running using 100 processes.

Table of observations at different delay links:

Delay in the link RTT (ms)	Time measured for file transfer (seconds)
0	154.46
10	154.49
200	154.86

Here are the screen shots :

At 0ms delay in the link:

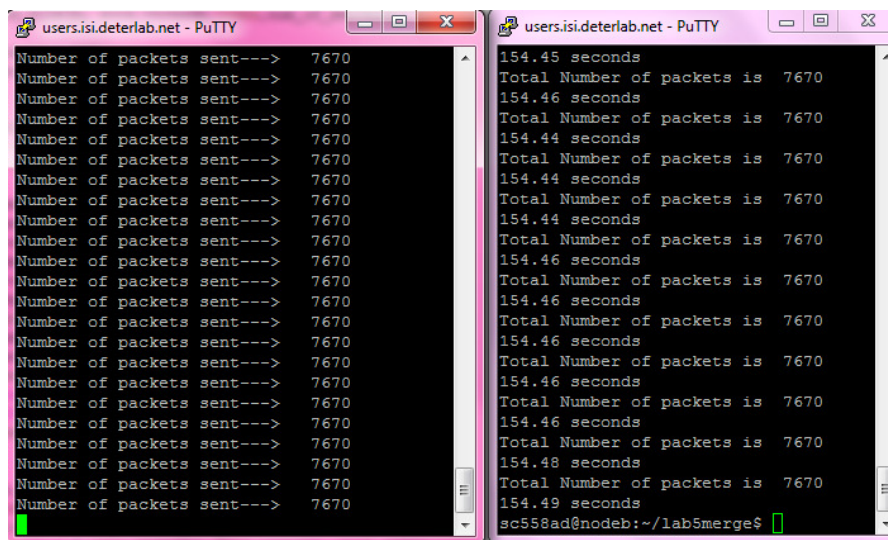


The image shows two side-by-side PuTTY terminal windows. The left window, titled 'users.isi.deterlab.net - PuTTY', displays a list of 20 lines, each showing 'Number of packets sent---> 7670'. The right window, also titled 'users.isi.deterlab.net - PuTTY', displays a list of 20 lines, each showing a time value followed by 'Total Number of packets is 7670'. The time values range from 154.38 to 154.48 seconds. Both windows have a green cursor at the bottom.

```
users.isi.deterlab.net - PuTTY
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
sc558ad@nodea:~/lab5merge$

users.isi.deterlab.net - PuTTY
154.39 seconds
Total Number of packets is 7670
154.43 seconds
Total Number of packets is 7670
154.39 seconds
Total Number of packets is 7670
154.38 seconds
Total Number of packets is 7670
154.40 seconds
Total Number of packets is 7670
154.42 seconds
Total Number of packets is 7670
154.38 seconds
Total Number of packets is 7670
154.41 seconds
Total Number of packets is 7670
154.43 seconds
Total Number of packets is 7670
154.42 seconds
Total Number of packets is 7670
154.48 seconds
Total Number of packets is 7670
154.46 seconds
sc558ad@nodeb:~/lab5merge$
```

At 5ms delay in the link:

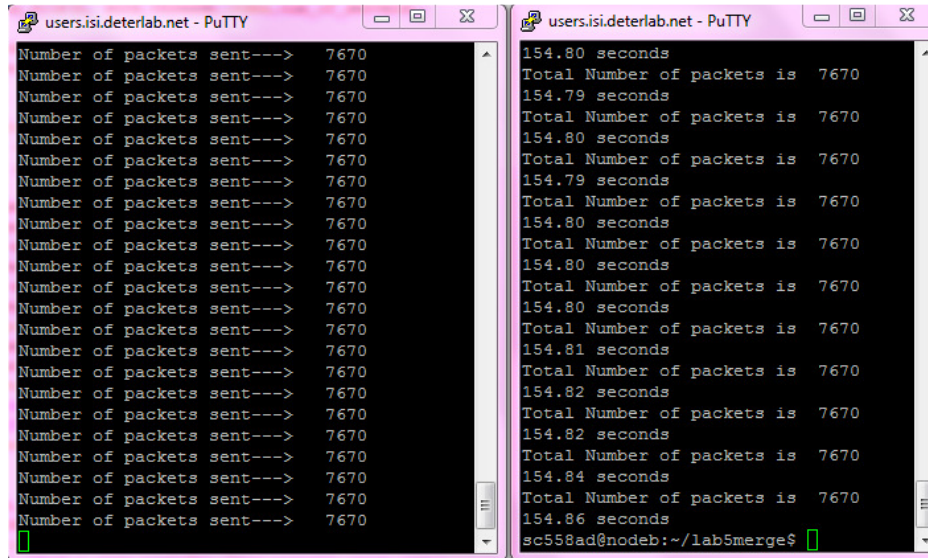


The image shows two side-by-side PuTTY terminal windows. The left window, titled 'users.isi.deterlab.net - PuTTY', displays a list of 20 lines, each showing 'Number of packets sent---> 7670'. The right window, also titled 'users.isi.deterlab.net - PuTTY', displays a list of 20 lines, each showing a time value followed by 'Total Number of packets is 7670'. The time values range from 154.44 to 154.49 seconds. Both windows have a green cursor at the bottom.

```
users.isi.deterlab.net - PuTTY
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
Number of packets sent---> 7670
sc558ad@nodea:~/lab5merge$

users.isi.deterlab.net - PuTTY
154.45 seconds
Total Number of packets is 7670
154.46 seconds
Total Number of packets is 7670
154.44 seconds
Total Number of packets is 7670
154.44 seconds
Total Number of packets is 7670
154.44 seconds
Total Number of packets is 7670
154.46 seconds
Total Number of packets is 7670
154.46 seconds
Total Number of packets is 7670
154.46 seconds
Total Number of packets is 7670
154.46 seconds
Total Number of packets is 7670
154.48 seconds
Total Number of packets is 7670
154.49 seconds
sc558ad@nodeb:~/lab5merge$
```

At 200ms delay in the link:



The image shows two side-by-side PuTTY terminal windows connected to users.isi.deterlab.net. The left window displays a list of 20 'Number of packets sent' entries, each followed by '7670'. The right window displays a series of 'Total Number of packets is 7670' messages interspersed with timing measurements in seconds, ranging from 154.80 to 154.86. The terminal windows have a black background with white text and standard PuTTY window controls at the top.

```
users.isi.deterlab.net - PuTTY
Number of packets sent--> 7670
Number of packets sent--> 7670
Number of packets sent--> 7670
Number of packets sent--> 7670
Number of packets sent--> 7670
Number of packets sent--> 7670
Number of packets sent--> 7670
Number of packets sent--> 7670
Number of packets sent--> 7670
Number of packets sent--> 7670
Number of packets sent--> 7670
Number of packets sent--> 7670
Number of packets sent--> 7670
Number of packets sent--> 7670
Number of packets sent--> 7670
Number of packets sent--> 7670
Number of packets sent--> 7670
Number of packets sent--> 7670
Number of packets sent--> 7670
Number of packets sent--> 7670

users.isi.deterlab.net - PuTTY
154.80 seconds
Total Number of packets is 7670
154.79 seconds
Total Number of packets is 7670
154.80 seconds
Total Number of packets is 7670
154.79 seconds
Total Number of packets is 7670
154.80 seconds
Total Number of packets is 7670
154.80 seconds
Total Number of packets is 7670
154.80 seconds
Total Number of packets is 7670
154.81 seconds
Total Number of packets is 7670
154.82 seconds
Total Number of packets is 7670
154.82 seconds
Total Number of packets is 7670
154.84 seconds
Total Number of packets is 7670
154.86 seconds
sc558ad@nodeb:~/lab5merge$
```

## Difficulties Faced:

We started off with forking multiple server threads and multiple client threads for sending and receiving of the parts of files. But the threads in python did not seem to work in parallel and so we switched to multi processing.

We tried using the “pickle” module of python for serializing the data for changing the object(an array in our case) to a string to transmit over the link. But the pickle could not seem to handle the special non-printable characters which might be occurring in the 1 GB temp file generated using the ‘dd’ command. We tried sending a text file of 12Megabytes where the code with pickle seemed to work fine. But with a 12MB file generated by a ‘dd’ command failed to transfer. Now we are sending the array of packets with each of them being sent as a string over the link.

Initially we were writing to a file on the disk each time a packet was received by the client which increased a lot of delay. We then changed our code to write into a buffer each time a packet was received in the client code and later, on receiving all the packets we concatenated all the respective files to a single output file.