The Following Document contains the Source Code of the Proxy Project

Source Name	Description	Remarks
CPingClient.cpp	A Simple Ping Client implementation	Levarages ClientSocket
CPingClient.h	Header file for Ping Client	
CProtocolServer.cpp	Has got implementation of Protocol Server and Proxy Server	
CProtocolServer.h	Header file for Protocol and Proxy Server	Leverages ServerSpcket
ClientSocket.cpp	A Simple S0cket Client	Leveraged in ProxyServer
ClientSocket.h	Header file for Socket Client	
ServerSocket.cpp	The Core Server Code	
ServerSocket.h	The Core Server Header	
Utils.cpp	Utility/Environment Functions	
Utils.h	Header files	
First.cpp	A Simple Passthrough Proxy for Filer Transfer	
makefile	The makefile	Make -f makefile
Spawner.cpp	A GNU Linux program	Not used now

Let us compile the Code to generate the Proxy

```
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Project_Engine$ make
g++ -c -std=c++17  Utils.cpp  -o Utils.o
g++ -c -std=c++17  ServerSocket.cpp -o ServerSocket.o
g++ -c -std=c++17  ClientSocket.cpp -o ClientSocket.o
g++ -c -std=c++17  CPingClient.cpp  -o CPingClient.o
g++ -c -std=c++17  CProtocolServer.cpp -o CProtocolServer.o
g++ -std=c++17 spawner.cpp Utils.o  ServerSocket.o ClientSocket.o  CProtocolServer.o  -lpthread -
o spawner.exe
g++ -std=c++17 first.cpp ServerSocket.o ClientSocket.o CPingClient.o CProtocolServer.o   -lpthread
-o first.exe
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Project_Engine$
```

Let us move to the directory which contains the File Transfer Protocol written by Midhun A Darvin in a Cross Platform Manner. Works on Windows and Linux.

```
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Midhun_Code$ ls -l
total 104
-rw-r--r-- 1 pai pai  7310 May  6 16:11 Client.cpp
-rwxr-xr-x 1 pai pai 28280 May  6 16:55 Client.dmp
-rw-r--r-- 1 pai pai  2746 May  6 16:48 Client.h
-rw-r--r-- 1 pai pai 28280 May  6 18:15 DEST_WIRE.out
```

```
-rw-r--r-- 1 pai pai  4206 May  6 16:23 Payload.cpp
-rw-r--r-- 1 pai pai     0 May  6 16:22 Playload.cpp
-rw-r--r-- 1 pai pai   786 May  6 16:11 README.md
-rw-r--r-- 1 pai pai 11226 May  6 16:24 Server.cpp
-rw-r--r-- 1 pai pai  2992 May  6 16:11 Server.h
-rw-r--r-- 1 pai pai    96 May  6 16:29 makefile
-rwxrwxrwx 1 pai pai    95 May  6 16:50 makefile.sh
-rw-r--r-- 1 pai pai     0 May  6 16:23 payload.cpp
-rw-r--r-- 1 pai pai  2316 May  6 16:21 payload.h
-rw-r--r-- 1 pai pai    11 May  6 16:11 sample.txt
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Midhun_Code$
```

Now Let us Compile, the Server.exe and Client.exe

```
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Midhun_Code$ g++ Server.cpp Payload.cpp -lpthread -
o Server.exe
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Midhun_Code$ g++ Client.cpp Payload.cpp -lpthread -
o Client.exe
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Midhun_Code$ ls *.exe
Client.exe  Server.exe
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Midhun_Code$
```

Now Let us run the stuff without Proxy

```
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Midhun_Code$ ./Server.exe 4000 &
[2] 1217
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Midhun_Code$ ./Client.exe Client.dmp localhost 4000
```

Now let us kill all the Server.exe process using the Kill command

```
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Midhun_Code$ ps
  PID TTY          TIME CMD
   72 tty2     00:00:00 bash
  997 tty2     01:46:50 Server.exe
 1217 tty2     00:02:45 Server.exe
 1232 tty2     00:00:00 ps
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Midhun_Code$ kill 997 1217
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Midhun_Code$ ps
  PID TTY          TIME CMD
   72 tty2     00:00:00 bash
 1233 tty2     00:00:00 ps
[1]-  Terminated              ./Server.exe 9000
[2]+  Terminated              ./Server.exe 4000
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Midhun_Code$
```

The Proxy is written in such a way that Proxy does port forwarding to the Port 9000. So, next time Server.exe should listen on Port 9000

```
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Midhun_Code$ ./Server.exe 9000 &
[1] 1245
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Midhun_Code$ About to Bind.. .....
Listen. .....
Creating a POSIX Thread..... for Listener

Entered the Listener Thread....

pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Midhun_Code$
```

Now, Let us start the Proxy on the Port 5000. Any request which comes to the Proxy will be forwarded to port 9000.

In the CProxyServer.cpp, look at the following line

```
////////////////////////////////////////
// Get a Default Endpoint
//
END_POINT * GetDefaultEndpoint() {
   return  new END_POINT { "127.0.0.1" ,9000 , 1,"",0,"  "};
}
```

Let us start the Proxy Server, at the Port 5000

```
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Project_Engine$ ./first.exe 5000
Received from Command line 5000

First Thread  =><str>
Started First Listening Thread
Entered the Listener Thread....
node info => string(info.node_info)
.....
```

Now, start the Client and hit at the Port 5000. The Proxy will receive the request and forward it to Port 9000. The File is getting transferred correctly.

```
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Midhun_Code$ ./Client.exe Client.dmp localhost 5000
File name: Client.dmp   File size: 28280
.....After the Accept.....
.....Accepted a new Connection.....
```

The Screen dump of the Proxy is given below

```
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Project_Engine$ ./first.exe 5000
Received from Command line 5000

First Thread  =><str>
Started First Listening Thread
Entered the Listener Thread....
node info => string(info.node_info)
.............

.....................After the Accept.........

.....................Accepted a new Connection.........
B4 callint Client Threaed => ClientData.node_info
Remote IP address == 127.0.0.1
.............
Resolved 127.0.0.1   9000
Entered Nested Loop
==================================
Received a Client packet.....................
Length of Packet is 12
Packet Type = 1
==================================
==================================
Received a Server packet.....................
Length of Packet is 12
Packet Type = 2
==================================
==================================
Received a Client packet.....................
Length of Packet is 280
Packet Type = 3
==================================
==================================
Received a Server packet.....................
Length of Packet is 12
Packet Type = 2
==================================
==================================
Received a Client packet.....................
Length of Packet is 5016
Packet Type = 4
==================================
==================================
Received a Client packet.....................
Length of Packet is 30104
Packet Type = 4
==================================
==================================
Received a Server packet.....................
Length of Packet is 12
Packet Type = 2
```

```
====================================
```

The Screen dump of the Server.exe and Client.exe is given below

```
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Midhun_Code$ ./Client.exe Client.dmp localhost 5000
File name: Client.dmp   File size: 28280
.....After the Accept.....
.....Accepted a new Connection.....
Acquired the Lock....
1
Finished Sending the Acknowledgement ... bytes send= 12

Ready to Transfer File .....

 file name = Client.dmp size = 28280finished .....sending acknowledgement.....transferring files
Waiting for the content
Now, we will iterate to send the content of the file.....
Opened File.....Client.dmp
Sequence      ..... 1
Sending Bytes ..... 4096
Sequence      ..... 2
Sending Bytes ..... 4096
Sequence      ..... 3
Sending Bytes ..... 4096
Sequence      ..... 4
Packet Type = 4
Sending Bytes ..... 4096
Packet sequence ..... 1
Sequence      ..... 5
Sending Bytes ..... 4096
Sequence      ..... 6
Sending Bytes ..... 4096
Residue = 3704
Coming out of the Send Loop Send Residue
Finished Sending File, About to send EOF
Finished Sending EOF
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Midhun_Code$ Packet Type = 4
Packet sequence ..... 2
Packet Type = 4
Packet sequence ..... 3
Packet Type = 4
Packet sequence ..... 4
Packet Type = 4
Packet sequence ..... 5
Packet Type = 4
Packet sequence ..... 6
Packet Type = 4
Packet sequence ..... 7
Packet Type = 5
End of File Received
```

```
pai@LAPTOP-QIG6F4HD:~/RecursiveMake/Midhun_Code$
```

To Run the Program, we started two instances of WSL terminal. One Terminal was used to run Server.exe (in the background using &) and Client.exe in the foreground. In the Sercond terminal, we used Proxy (.first.exe)