# COMP 4985

## Assignment 2

Robin Hsieh A00657820

# Table of Contents

# Analysis & Report

## Abstract

An analysis on TCP and UDP protocols to see the reliability when sending datagram packets between two Window workstations. This report will show that TCP and UDP are very similar on a wired LAN (Local Area Network), but when introduced with wireless LAN, and WAN (Wide Area Network), we will discover that UDP becomes more unreliable as the distance between the client and the server increases. The only downside of TCP is the connection speed compared to UDP.

## Introduction

The objective of this assignment was to design and implement a Windows program, which will generate TCP and UDP datagrams, and transfer the data using the TCP/IP protocol API calls. In this report, I will demonstrate and attempt to prove the widely known facts about TCP and UDP connections through data collected, and to confirm those facts. Theoretically, TCP are known to be a slower connection than UDP, but is much more reliable, and vice versa. This Windows application will allow the user to choose between hosting a TCP/UDP server, or become the client that will connect to a TCP/UDP server. If the client has been chosen, the user will be able to input IP address, port number, protocol selection, packet size, and the amount of times the packet will be sent. Once everything has been filled out, the user will be able to either send a packet filled with randomized characters, or a file pre-selected. If the server has been chosen, the user will be able to change the port number and the protocol selection for the server to be hosted on. In addition, the server will also automatically save the incoming packets and be written to a file.

This report will also include the analysis on both connections to determine what the difference in speed were, and which is more reliable. The testing inputs that were used was a set of different packet sizes (including 1kB, 4kB, 20kB, and up to 60kB). The packets were also sent in two different multiplicity (10 and 100) to see if this will take an effect on how the speed and reliability factor. The conclusion from this report will show that TCP is more reliable compared to UDP, but will take longer for all the packets to arrive at the destination. Furthermore, distance will play a significant role between TCP and UDP, while the UDP speed remains at a very high level, the packet loss becomes known when introduced with more noise factors.

## Analysis

The purpose of this application is to test the speed and reliability of the two protocols being utilized (TCP/UDP).

(WARNING: The charts below may not be completely accurate, and the delay might be a little shifted due to WINDOWS SYSTEM time may be different between the client and the server workstations. Tried to work around this using the call "NET TIME \\local_ip_address /SET /YES", but still is not completely accurate. The data being presented was captured on a home network, so the results may not be inaccurate due to network speed, and bandwidth issue.)

First, we will analyze **wired LAN connections**:

| Protocol | Packet Size (kB) | Times Sent | Client Start Time | Server End Time | Delay | Client Sent (B) | Server Received (B) | Throughput |
|---|---|---|---|---|---|---|---|---|
| TCP | 1 | 10 | 44.788 | 44.967 | 0.179 | 10240 | 10240 | 100.00% |
| UDP | 1 | 10 | 45.16 | 45.335 | 0.175 | 10240 | 10240 | 100.00% |
| TCP | 1 | 100 | 46.928 | 48.405 | 1.477 | 102400 | 102400 | 100.00% |
| UDP | 1 | 100 | 47.982 | 48.877 | 0.895 | 102400 | 86016 | 84.00% |
| TCP | 4 | 10 | 18.889 | 19.041 | 0.152 | 40960 | 40960 | 100.00% |
| UDP | 4 | 10 | 53.84 | 53.948 | 0.108 | 40960 | 32768 | 80.00% |
| TCP | 4 | 100 | 22.061 | 22.87 | 0.809 | 409600 | 409600 | 100.00% |
| UDP | 4 | 100 | 58.535 | 59.509 | 0.974 | 409600 | 319488 | 78.00% |
| TCP | 20 | 10 | 36.262 | 36.426 | 0.164 | 204800 | 204800 | 100.00% |
| UDP | 20 | 10 | 23.854 | 23.931 | 0.077 | 204800 | 122880 | 60.00% |
| TCP | 20 | 100 | 39.625 | 40.476 | 0.851 | 2048000 | 2027520 | 99.00% |
| UDP | 20 | 100 | 26.622 | 27.306 | 0.684 | 2048000 | 1556480 | 76.00% |

From the chart, we can see that TCP connect is quite reliable, throughput is always very high through all the different stress tests it goes through. But as expected, the delay for TCP is generally always higher than UDP. Also notice that UDP almost always beat TCP with speed, but starts to decrease in throughput as soon as it starts sending more data.

Second, we will analyze **wireless LAN connections**:

| Protocol | Packet Size (kB) | Times Sent | Client Start Time | Server End Time | Delay | Client Sent (B) | Server Received (B) | Throughput |
|---|---|---|---|---|---|---|---|---|
| TCP | 1 | 10 | 1.75 | 1.765 | 0.015 | 10240 | 10240 | 100.00% |
| UDP | 1 | 10 | 29.737 | 29.774 | 0.037 | 10240 | 10240 | 100.00% |
| TCP | 1 | 100 | 6.37 | 6.434 | 0.064 | 102400 | 100352 | 98.00% |
| UDP | 1 | 100 | 22.763 | 22.897 | 0.134 | 102400 | 43008 | 42.00% |
| TCP | 4 | 10 | 14.05 | 14.08 | 0.03 | 40960 | 28672 | 70.00% |
| UDP | 4 | 10 | 3.749 | 3.774 | 0.025 | 40960 | 24576 | 60.00% |
| TCP | 4 | 100 | 18.939 | 19.051 | 0.112 | 409600 | 188416 | 46.00% |
| UDP | 4 | 100 | 12.001 | 12.113 | 0.112 | 409600 | 139264 | 34.00% |
| TCP | 20 | 10 | 57.003 | 57.027 | 0.024 | 204800 | 61440 | 30.00% |
| UDP | 20 | 10 | 48.289 | 48.321 | 0.032 | 204800 | 102400 | 50.00% |
| TCP | 20 | 100 | 3.552 | 3.627 | 0.075 | 2048000 | 266240 | 13.00% |
| UDP | 20 | 100 | 51.297 | 51.381 | 0.084 | 2048000 | 327680 | 16.00% |

From this chart, we can see, on a wireless network, there are a lot more chances for data/packets to get lost in transmission, even though this is still at a LAN. Even though our data might be a little corrupted as stated above, that getting the SYSTEMTIME on client and server might not completely be synchronous, in this case, UDP is slower than TCP in multiple counts. But the overall throughput still declares TCP the winner.

Lastly, we will analyze **WAN**:

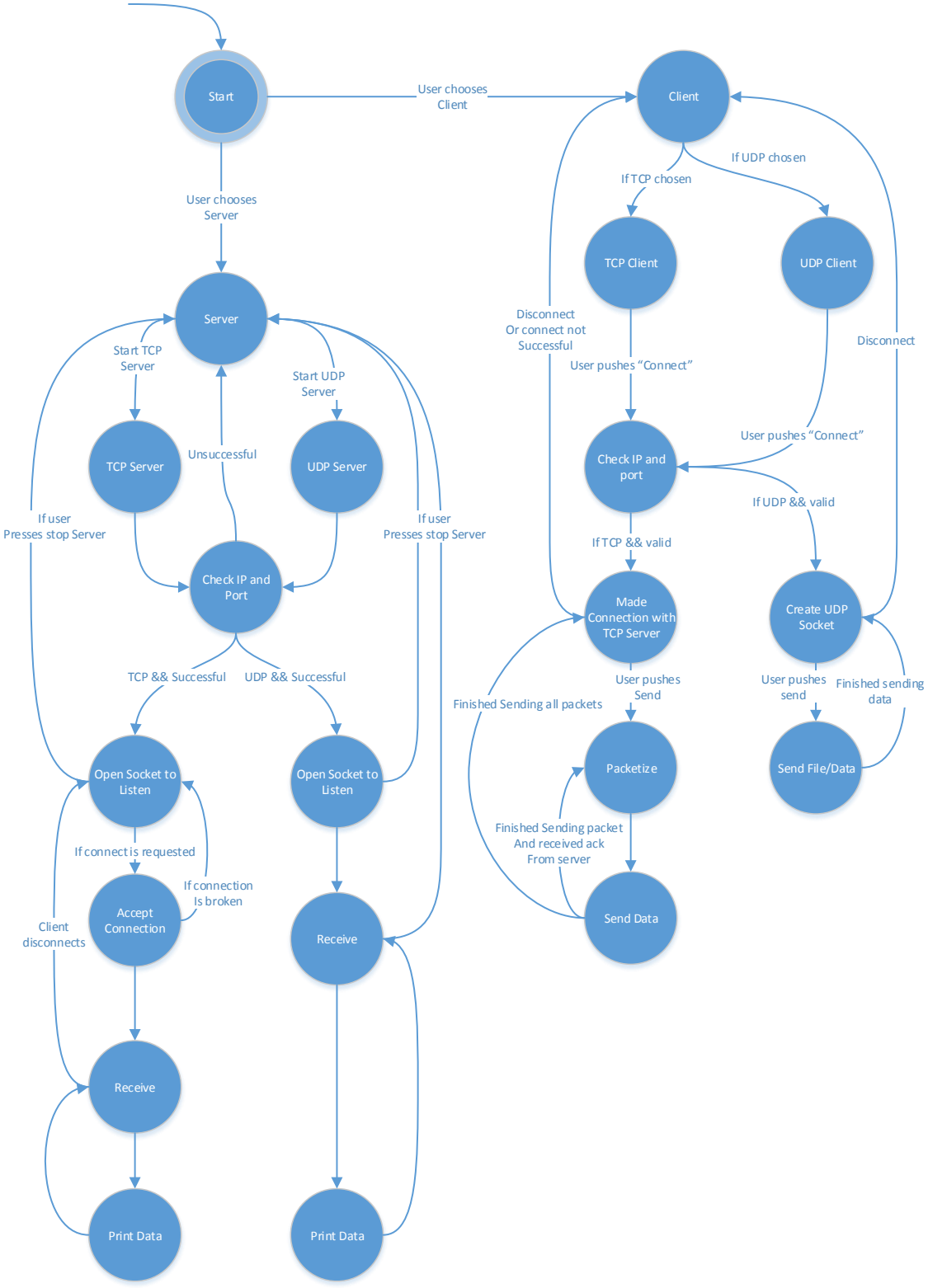
| Protocol | Packet Size (kB) | Times Sent | Client Start Time | Server End Time | Delay | Client Sent (B) | Server Received (B) | Throughput |
|---|---|---|---|---|---|---|---|---|
| TCP | 1 | 10 | 56.67 | 56.698 | 0.028 | 10240 | 10240 | 100.00% |
| UDP | 1 | 10 | 26.21 | 26.253 | 0.043 | 10240 | 10240 | 100.00% |
| TCP | 1 | 100 | 59.838 | 59.961 | 0.123 | 102400 | 101376 | 99.00% |
| UDP | 1 | 100 | 29.907 | 30.018 | 0.111 | 102400 | 27648 | 27.00% |
| TCP | 4 | 10 | 24.363 | 24.391 | 0.028 | 40960 | 28672 | 70.00% |
| UDP | 4 | 10 | 35.855 | 35.891 | 0.036 | 40960 | 32768 | 80.00% |
| TCP | 4 | 100 | 29.214 | 29.347 | 0.133 | 409600 | 204800 | 50.00% |
| UDP | 4 | 100 | 41.007 | 41.143 | 0.136 | 409600 | 122880 | 30.00% |
| TCP | 20 | 10 | 48.621 | 48.646 | 0.025 | 204800 | 61440 | 30.00% |
| UDP | 20 | 10 | 29.505 | 29.545 | 0.04 | 204800 | 102400 | 50.00% |
| TCP | 20 | 100 | 53.169 | 53.264 | 0.095 | 2048000 | 307200 | 15.00% |
| UDP | 20 | 100 | 32.415 | 32.518 | 0.103 | 2048000 | 327680 | 16.00% |

Here from the chart, the trend will continue, with the throughput from TCP generally better than UDP, with the delays in favor of UDP.

## Conclusion

Through this application, we have tested using TCP and UDP to send data to each respective servers, and confirmed that most times than other, TCP has a better overall throughput, while UDP is generally better on speed. To have this knowledge is very useful when talking about data communications. Whenever we're going to be writing a program in the future that involves a networking portion, the two different protocol will be one of the bigger decisions that needs to be made. This will all depend on how reliable we will need the data to go through, as well as how fast we want the data to pass from client to server, back and forth.

# Design


Start
User chooses Client
Client
User chooses Server
If TCP chosen
If UDP chosen
TCP Client
UDP Client
Server
Disconnect Or connect not Successful
Disconnect
User pushes "Connect"
Start TCP Server
Start UDP Server
TCP Server
Unsuccessful
UDP Server
Check IP and port
User pushes "Connect"
If UDP && valid
If user Presses stop Server
If user Presses stop Server
If TCP && valid
Check IP and Port
Made Connection with TCP Server
Create UDP Socket
TCP && Successful
UDP && Successful
User pushes Send
User pushes send
Finished sending data
Finished Sending all packets
Open Socket to Listen
Open Socket to Listen
Packetize
Send File/Data
If connect is requested
Finished Sending packet And received ack From server
If connection Is broken
Accept Connection
Client disconnects
Send Data
Receive
Receive
Print Data
Print Data

# Function Prototypes

## Main.cpp

```
LRESULT CALLBACK WndProc (HWND hwnd, UINT message, WPARAM wParam, LPARAM lParam);
void setup_InputFields(int cxChar, int cyChar, HWND hwnd, LPARAM lParam);
void startup_enable_inputFields(HWND hwnd);
void draw_text_title(HDC hdc, int cxChar, int cyChar);
void server_selected(HWND hwnd, HMENU hMenu);
void client_selected(HWND hwnd, HMENU hMenu);
BOOL OpenFile(HWND hwnd);
BOOL FileRead(HWND hwnd, LPTSTR pstrFileName);
```

## Server.cpp

```
BOOL SetupServer(HWND hwnd, SOCKET* lpListenSocket, int port, BOOL protocol_Selected);
BOOL SetupTCPServer(SOCKET* lpListenSocket, SOCKADDR_IN serverAddr);
BOOL SetupUDPServer(SOCKET* lpListenSocket, SOCKADDR_IN serverAddr);
BOOL AcceptTCPConnections(HWND hwnd, SOCKET* lpListenSocket, SOCKET* lpAcceptSocket);
void TCPRead(SOCKET* lpAcceptSocket);
void UDPRead(SOCKET* lpListenSocket);
void stopServer(SOCKET serverSocket);
```

## Client.cpp

```
BOOL Check_IP(TCHAR* host, struct hostent* hp);
BOOL UDPCreateSocket(HWND hwnd, SOCKET* lpClientSocket, TCHAR* host, int port, struct hostent* hp);
BOOL ConnectToTCPServer(HWND hwnd, SOCKET* lpConnectSocket, TCHAR* host, int port);
void SendFileToUDP(SOCKET* lpUDPSendSocket, TCHAR* host, int port, int packetSize, int timesToSend);
void SendFileToTCP(SOCKET* lpConnectedSocket, int packetSize, int timesToSend);
void SendUDPRandomData(SOCKET* lpDirectedSocket, TCHAR* host, int port, int packetSize, int timesToSend);
void SendTCPRandomData(SOCKET* lpConnectedSocket, int packetSize, int timesToSend);
void RandomizePacket(int packetSize, TCHAR* packet);
```

## EditResultBox.cpp

```
void resetResultBox();
void addToResultBox(const CHAR* s);
void AddTCPServerStats(BOOL ClientConnected, int bytesReceived);
void AddTCPClientStats(BOOL connected, int bytesSent);
void AddUDPServerStats(BOOL ListeningForClients, int bytesReceived);
void AddUDPClientStats(BOOL connected, int bytesSent);
```

## Pseudo Code

```
MainWindow()
{
   Register window properties
   Create Window
   Set up TextBoxes and ListBox for user inputs and printing result in

   Wait for User Input
   if SERVER is selected (pass in TCP or UDP)
      call Server funtions
   if CLIENT is selected (pass in TCP or UDP)
      call Client functions
}

Server()
{
   if UDP
      SetupUDPServer()
      ReadUDPSocket()
      PrintData()

   if TCP
      SetupTCPServer()
      AcceptTCPConnections()
      ReadTCPSocket()
      PrintData()
}

SetupUDPServer()
{
   Open Socket to listen on
   Use the WSAAsyncSelect call
   bind socket
}

ReadUDPSocket()
{
   do WSARecvFrom to get the message
}

SetupTCPServer()
{
   Open Socket to listen on
   Use the WSAAsyncSelect call
   bind socket
   listen on the socket
}
```

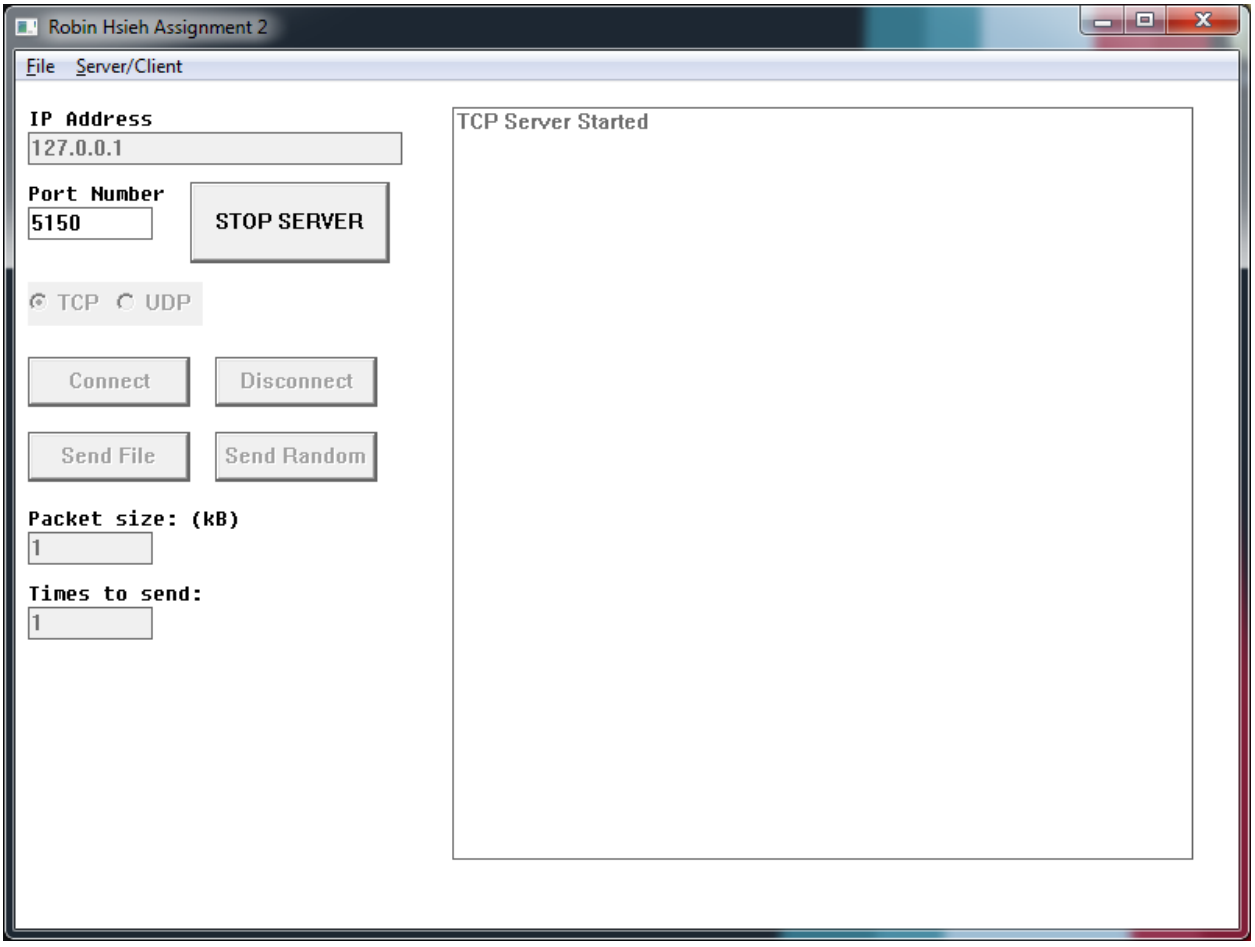
```
AcceptTCPConnections()
{
   If there is a connection being requested
      do WSAAccept on that socket
      Use the WSAAsyncSelect call to accept for reading, or to close
}
ReadTCPSocket()
{
   do WSARecv to read off of the socket
}
PrintData()
{
   Print how many bytes have been read
}

Client()
{
   if UDP
      CreateUDPClientSocket()
      SendUDPRandomData()
   if TCP
      CreateTCPClientSocket()
      ConnectToTCPServer()
      SendTCPRandomData()
}

CreateUDPClientSocket()
{
   check if IP is valid
   do WSASocket call
}

SendUDPRandomData()
{
   Get the specified packetsize and the the times to be sent from user input
   create random data of specified packetsize

   do WSASendTo * the times to be sent set by user
}

ConnectToTCPServer()
{
   check if IP is valid
   Open Socket
   Connect to that socket
}
```

## Test Documents

| Test | Test Description | Expected Result | Pass/Fail | Screen Shot |
|---|---|---|---|---|
| 1 | Start TCP server | Successfully started TCP server | Pass | Figure 01 |
| 2 | Start UDP server | Successfully started UDP server | Pass | Figure 02 |
| 3 | Client connecting to TCP server | Connection established | Pass | Figure 03a, 03b |
| 4 | Client set up to send to certain port for UDP server | Ready to send data | Pass | Figure 04 |
| 5 | Send a file over TCP | Sending properly through TCP | Pass | Figure 05a |
| 6 | Send a file over UDP | Sending properly through UDP | Pass | Figure 06a |
| 7 | Send random packets over TCP | Sending properly through TCP | Pass | Figure 05a |
| 8 | Send random packets over UDP | Sending properly through UDP | Pass | Figure 06a |
| 9 | Receive on TCP server | Able to receive through TCP | Pass | Figure 05b |
| 10 | Receive on UDP server | Able to receive through UDP | Pass | Figure 06b |
| 11 | Save file coming into TCP/UDP server | File is saved | Pass | Figure 07a, 07b |

# Figures

## Figure 01



## Figure 02

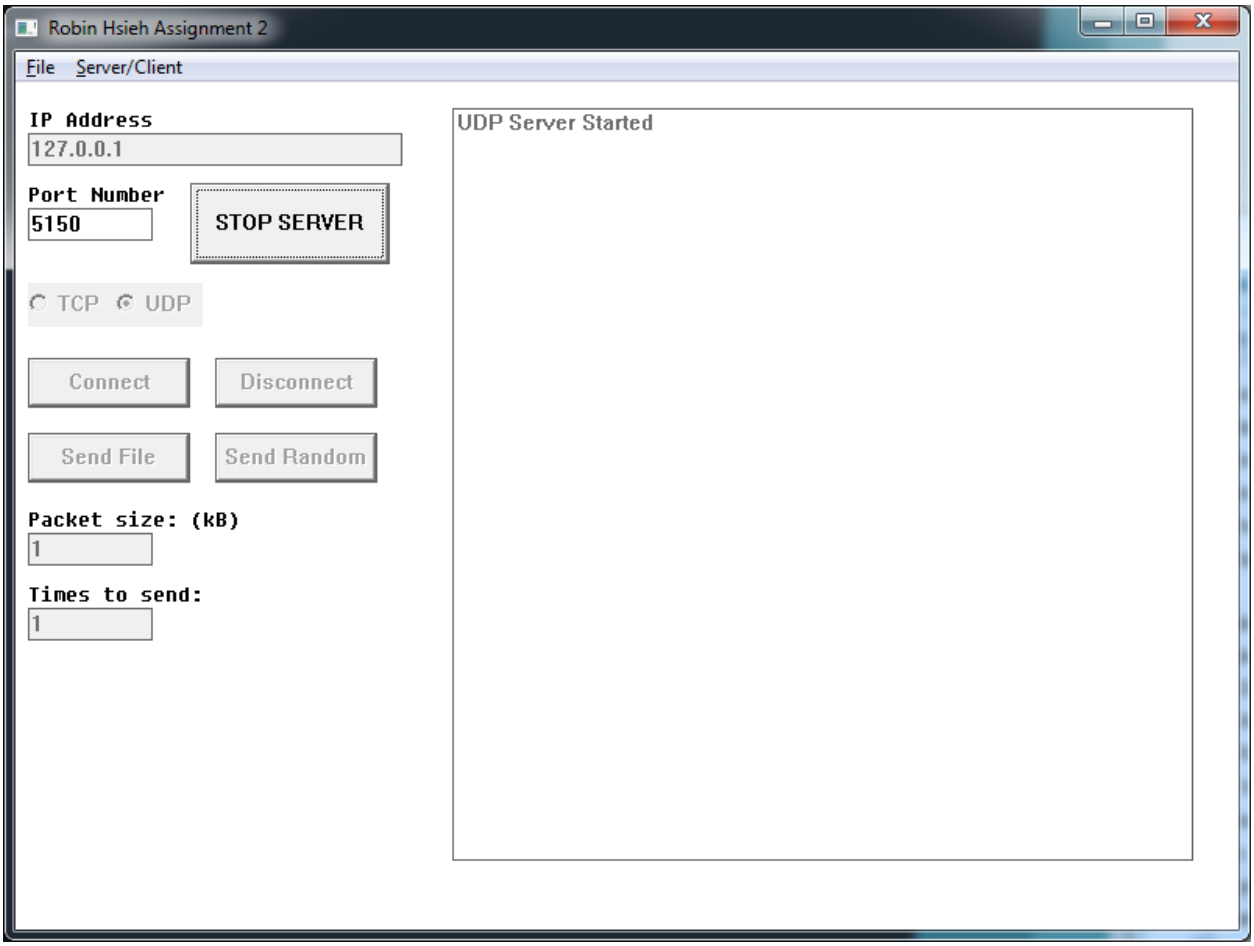

Robin Hsieh Assignment 2
File
Server/Client
IP Address
127.0.0.1
Port Number
5150
STOP SERVER
TCP
UDP
Connect
Disconnect
Send File
Send Random
Packet size: (kB)
1
Times to send:
1
UDP Server Started

Figure 03a

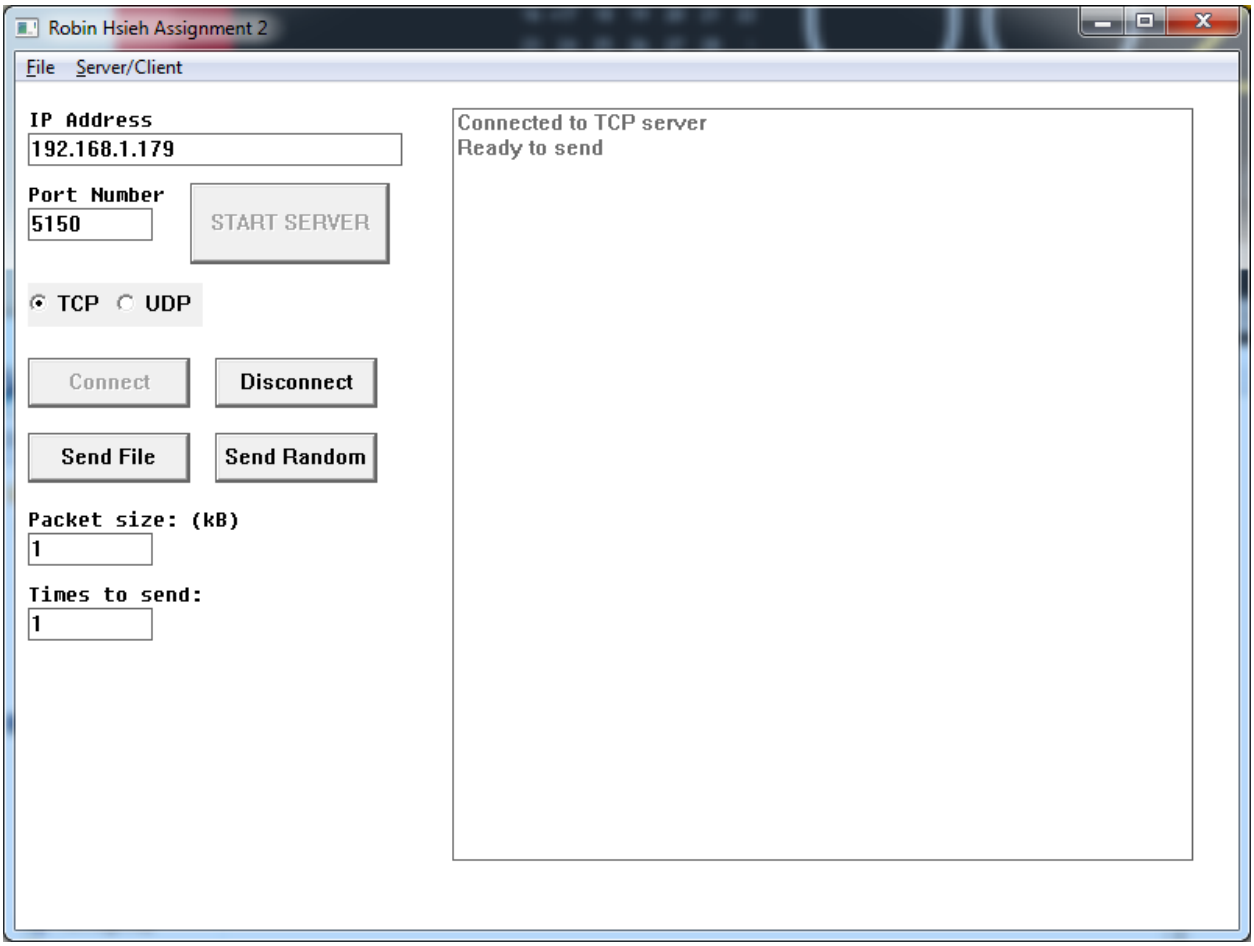

Robin Hsieh Assignment 2
File
Server/Client
IP Address
192.168.1.179
Port Number
5150
START SERVER
TCP
UDP
Connect
Disconnect
Send File
Send Random
Packet size: (kB)
1
Times to send:
1
Connected to TCP server
Ready to send

Figure 03b

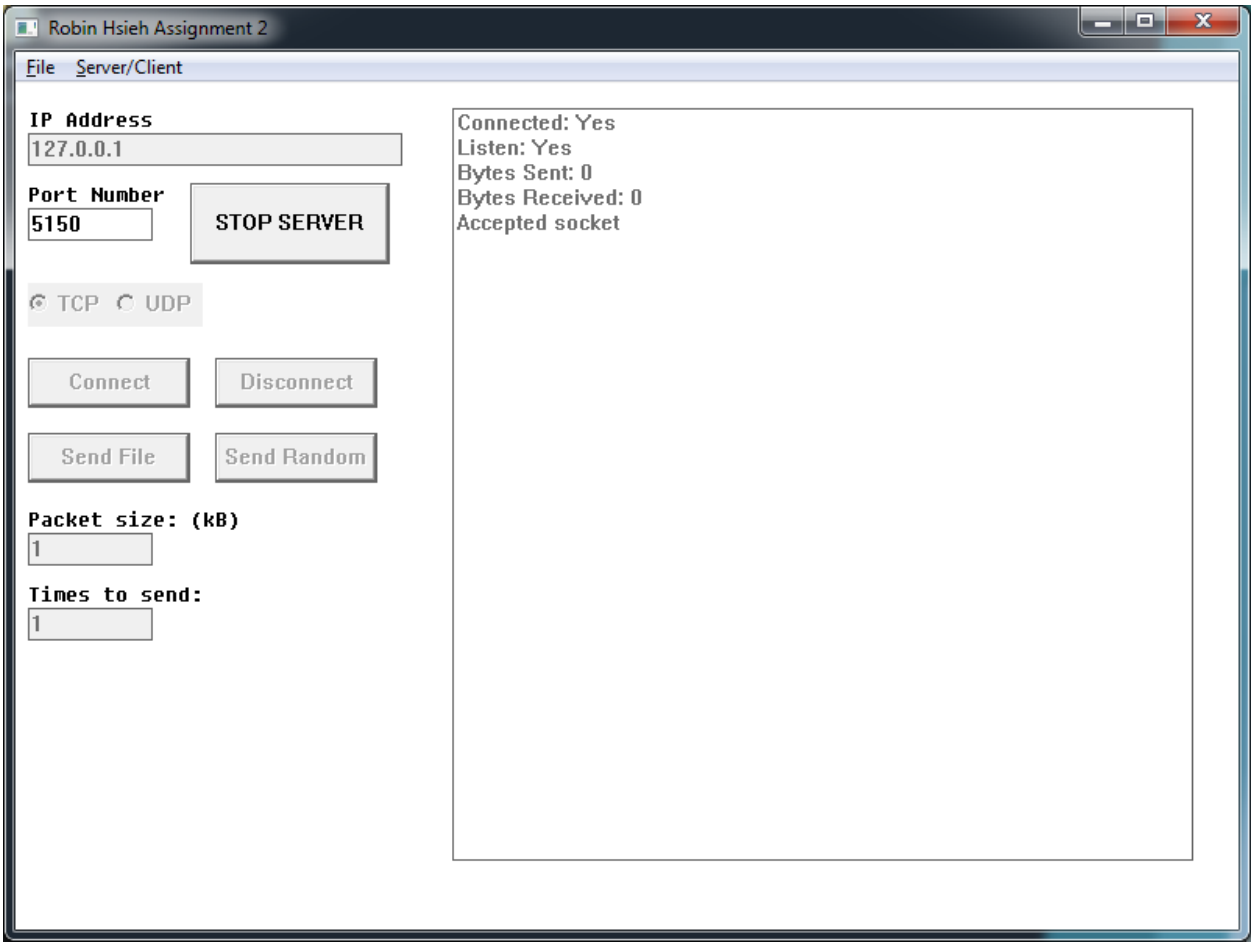

Robin Hsieh Assignment 2
File
Server/Client
IP Address
127.0.0.1
Port Number
5150
STOP SERVER
TCP
UDP
Connect
Disconnect
Send File
Send Random
Packet size: (kB)
1
Times to send:
1
Connected: Yes
Listen: Yes
Bytes Sent: 0
Bytes Received: 0
Accepted socket

Figure 04

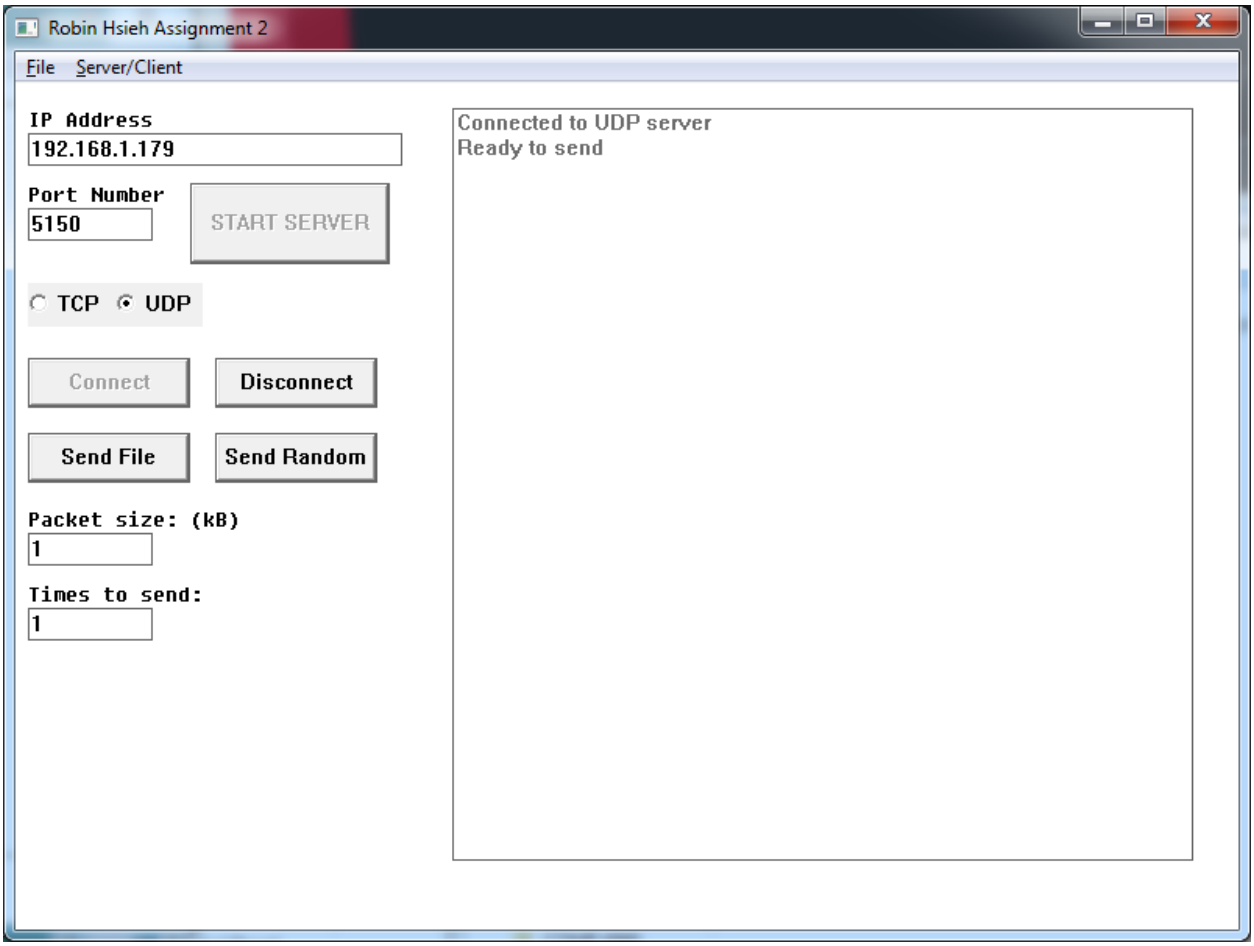

Robin Hsieh Assignment 2
File
Server/Client
IP Address
192.168.1.179
Port Number
5150
START SERVER
TCP
UDP
Connect
Disconnect
Send File
Send Random
Packet size: (kB)
1
Times to send:
1
Connected to UDP server
Ready to send

Figure 05a

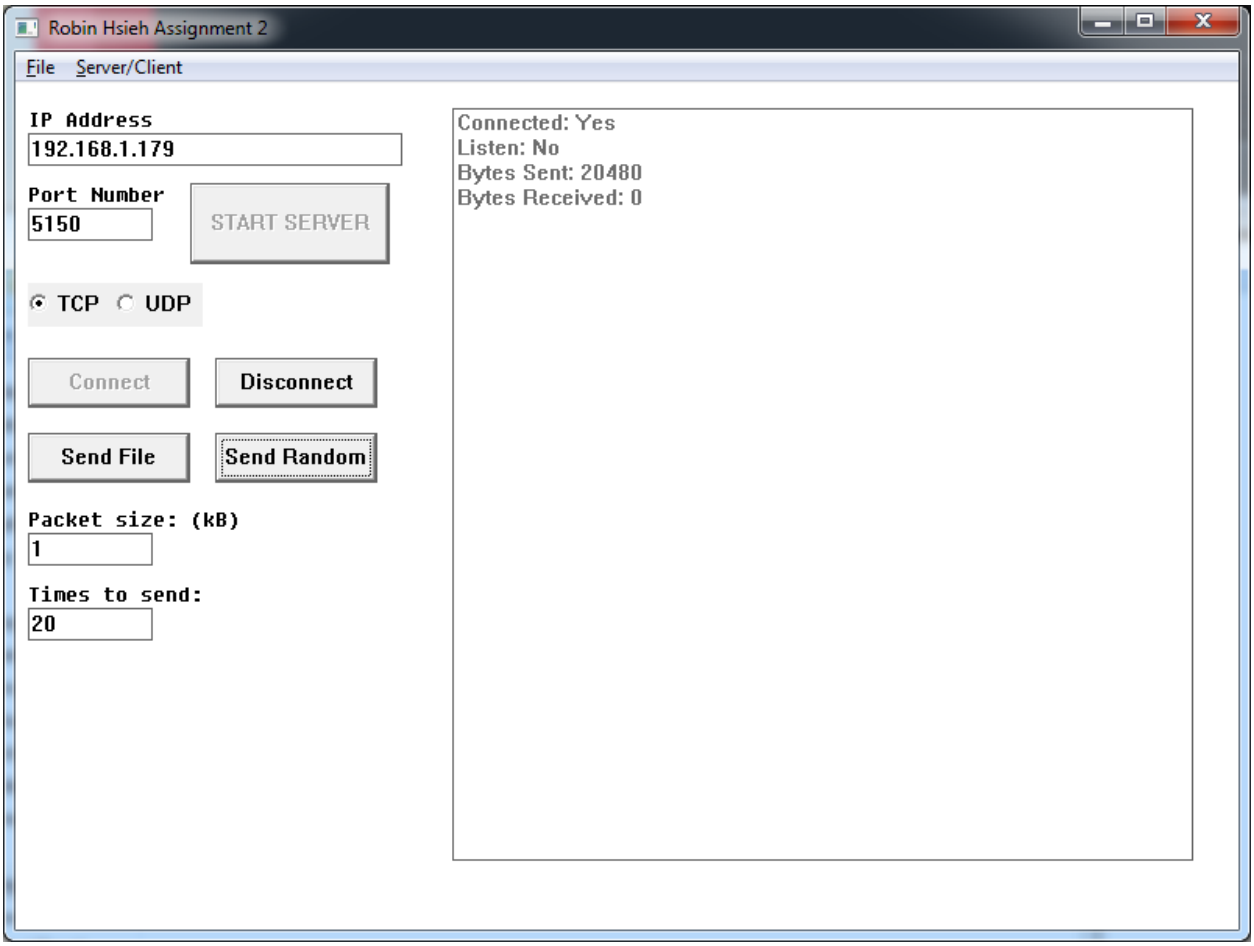

Robin Hsieh Assignment 2
File
Server/Client
IP Address
192.168.1.179
Port Number
5150
START SERVER
TCP
UDP
Connect
Disconnect
Send File
Send Random
Packet size: (kB)
1
Times to send:
20
Connected: Yes
Listen: No
Bytes Sent: 20480
Bytes Received: 0

Figure 05b

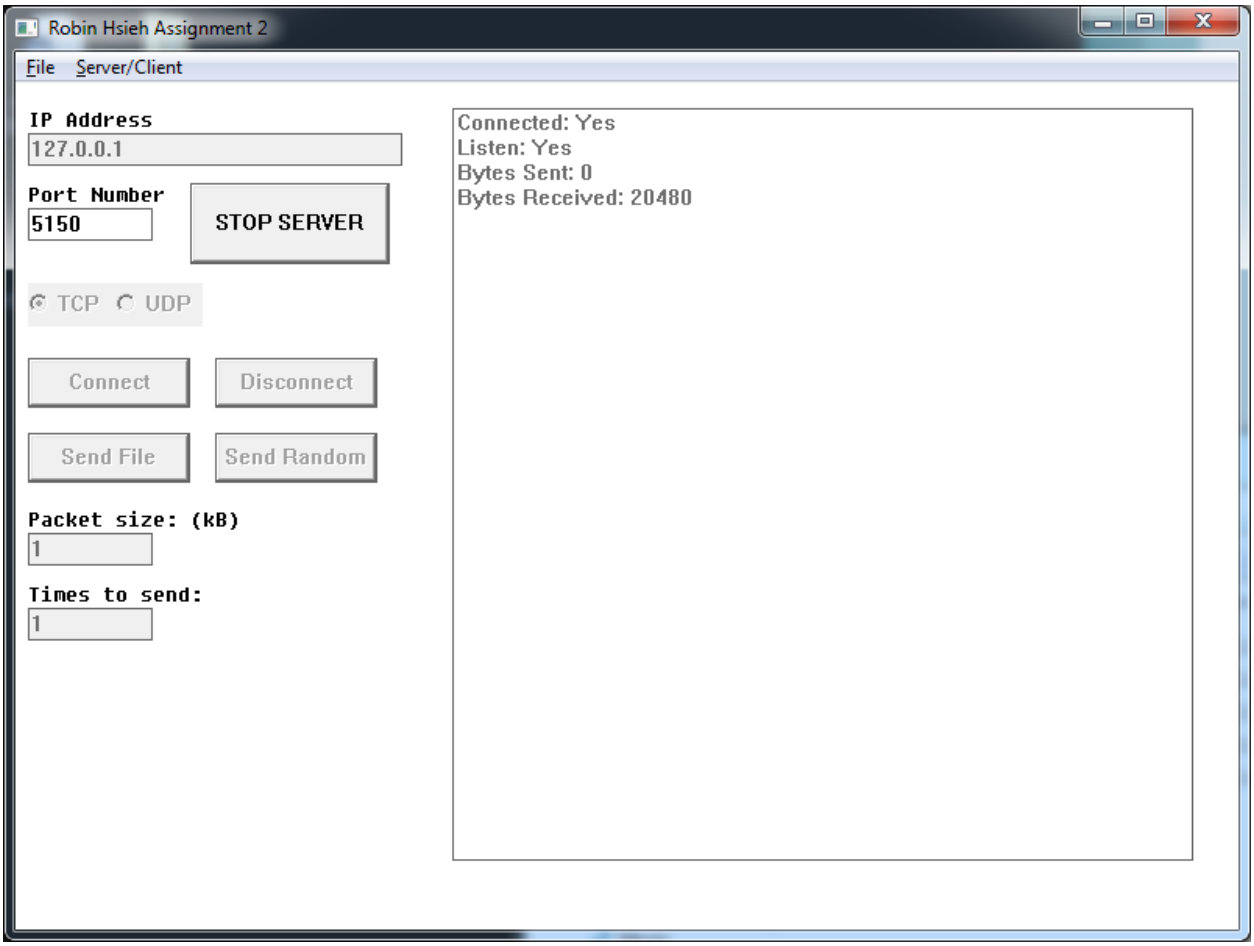

Robin Hsieh Assignment 2
File
Server/Client
IP Address
127.0.0.1
Port Number
5150
STOP SERVER
TCP
UDP
Connect
Disconnect
Send File
Send Random
Packet size: (kB)
1
Times to send:
1
Connected: Yes
Listen: Yes
Bytes Sent: 0
Bytes Received: 20480

Figure 06a

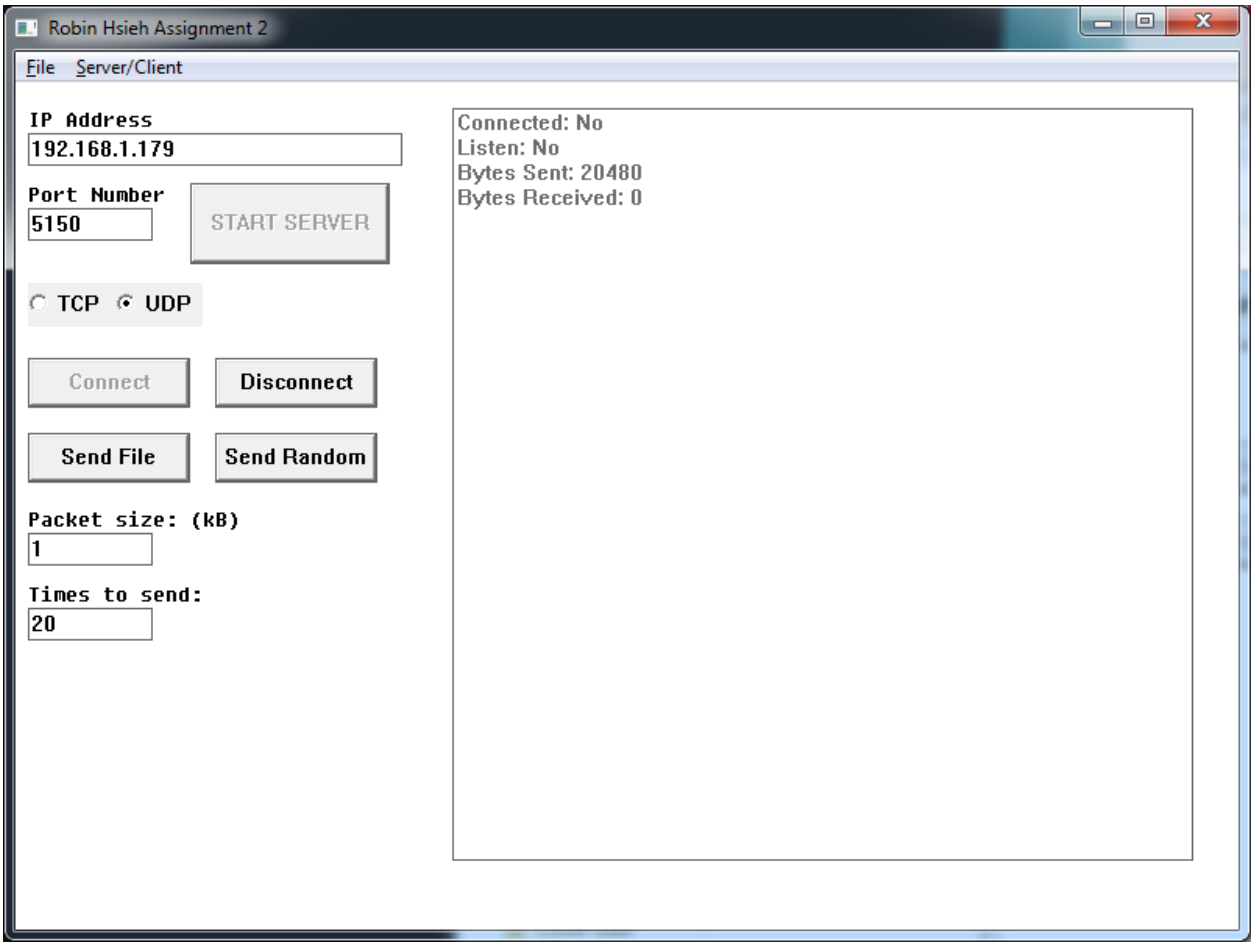

Robin Hsieh Assignment 2
File
Server/Client
IP Address
192.168.1.179
Port Number
5150
START SERVER
TCP
UDP
Connect
Disconnect
Send File
Send Random
Packet size: (kB)
1
Times to send:
20
Connected: No
Listen: No
Bytes Sent: 20480
Bytes Received: 0

Figure 06b

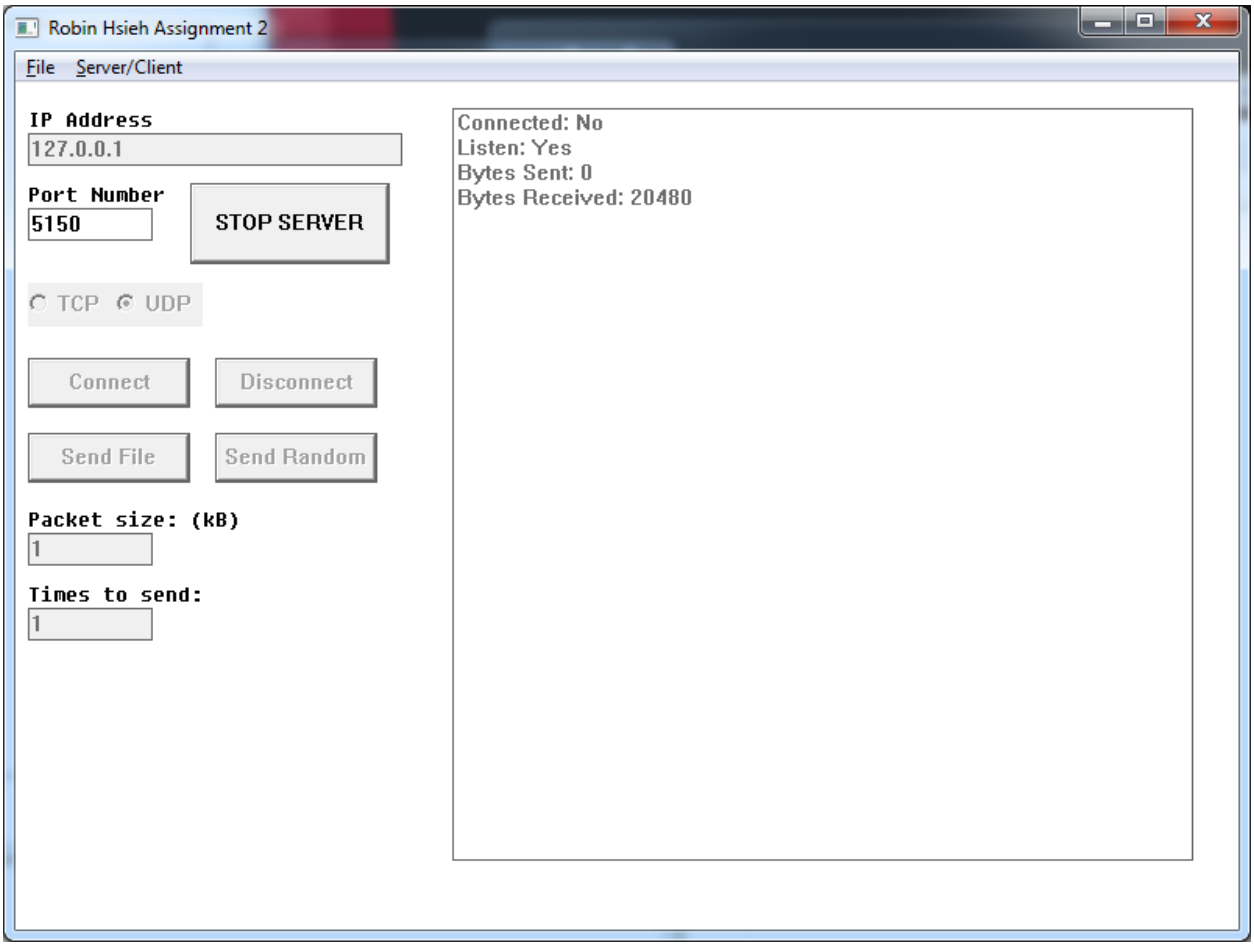
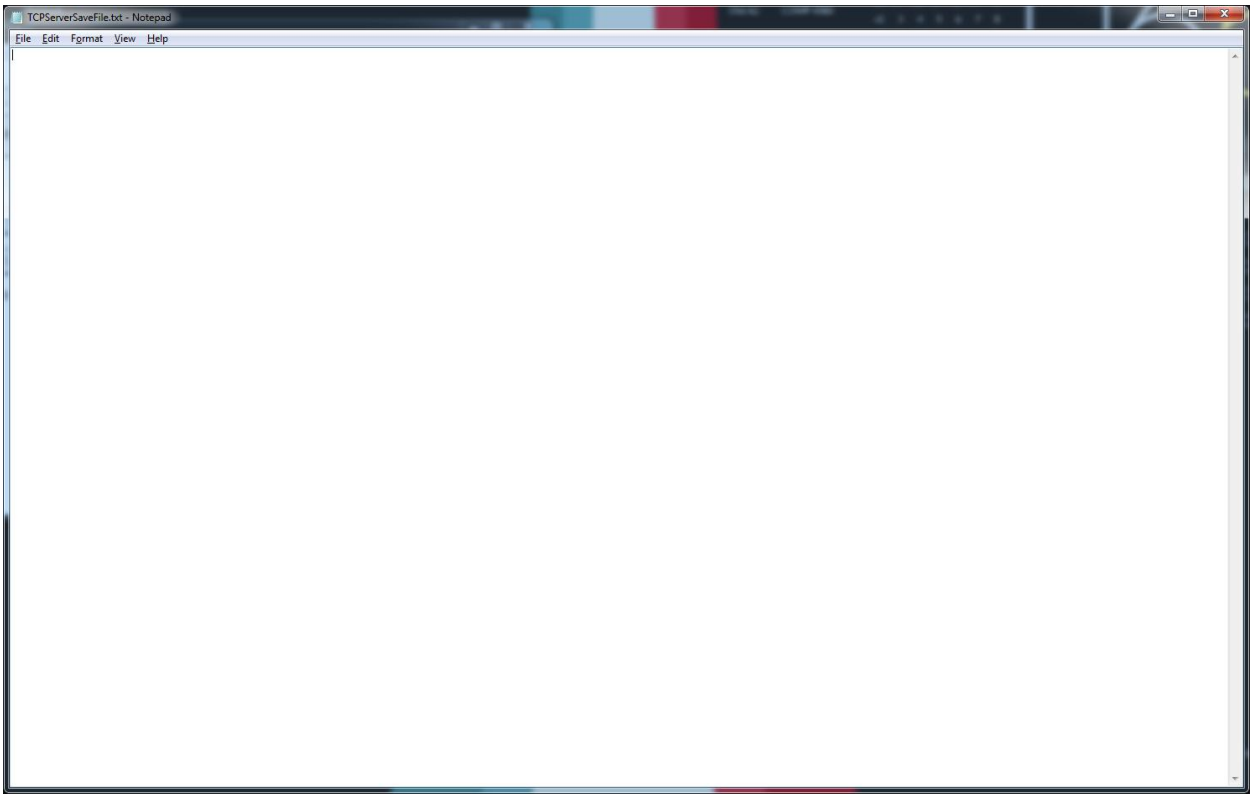

Robin Hsieh Assignment 2
File
Server/Client
IP Address
127.0.0.1
Port Number
5150
STOP SERVER
TCP
UDP
Connect
Disconnect
Send File
Send Random
Packet size: (kB)
1
Times to send:
1
Connected: No
Listen: Yes
Bytes Sent: 0
Bytes Received: 20480

Figure 07a



## Figure 07b