COMP 4985
Assignment 2
TCP/UDP Protocol Analysis

Jaegar Sarauer
A00925935

Contents

Purpose

The purpose of the TCP/UDP Client/Server program is to use Winsock 2 API with asynchronous sockets to study the speed and reliability of both TCP and UDP protocols. The ability to test both a client and a server of both protocols is wrapped into one program for the user to easily select. When starting the program, the user will be able to select the setup for the program, then be brought to the main interface for sending and receiving data.
Upon choosing a program type, (Client or Server), and a protocol (TCP or UDP), the main UI will appear for the user. The UI includes:

Input Fields:

- A field for an IP address. This field is for entering an IP for the client to use to connect to a server.
- A field for a port number. This field is for binding the client/server to a port, to listen for connections.
- A field for packet size. This is size of packet the client will be sending (if client program), or the size of packets the server will read in if it is a server program.
- A field for packet count. This will indicate how many “junk data” packets to send to the server if the client decides to select this option. It is unused by the server.
- Send False Data Check Box. This check box is for a client program. If the client decides to send “junk data” to the server, this box should be checked.
- Show Data Check Box. This box is for server and client optimization. It will disable the Data field, and some other operations which take extra resources when sending or receiving data.
- Data Field. This field will show the data read from a client (if a server), or show data loaded from a file, or typed in from the user (if a client program).
- Statistics field: This field shows data on incoming and outgoing packets (depending on the program type), transfer time, and current status of the program.

Buttons:

- (Server) Allow Connections: This button will open a socket, bind to it, and begin accepting connections from a server.
- (Client) Send: This button will open a socket, bind to it, and attempt to send data to a running server, defined either by the data field, or the junk data selection.
- Clear Data: This button will remove all data in the Data box, and reset statistics.
- Open File: This will allow the user to load a file from somewhere on the computer.
- Save File: This will allow the user to save a file to somewhere on the computer.
- Exit: This will close the program, closing connections and sockets associated with the ending program.

Please see the Usage section for details on how to use this program.

Abstract

The Transport Control Protocol (TCP) and User Datagram Protocol (UDP) are widely known as the two major protocols in data communication. This document studies the difference between the two, and compares the findings with previous information regarding each protocol. Testing of the protocols have been carried out by the TCP/UDP Client/Server program, explained later in this document. The studies are conducted over a wired Local Area Network connection. Results are expected to differ on types of networks used.

After conducting the study using the TCP/UDP Client/Server program, TCP and UDP operate very similar on a wired LAN. The only noticeable differences is the packet loss for UDP on very small packet sizes.

Introduction

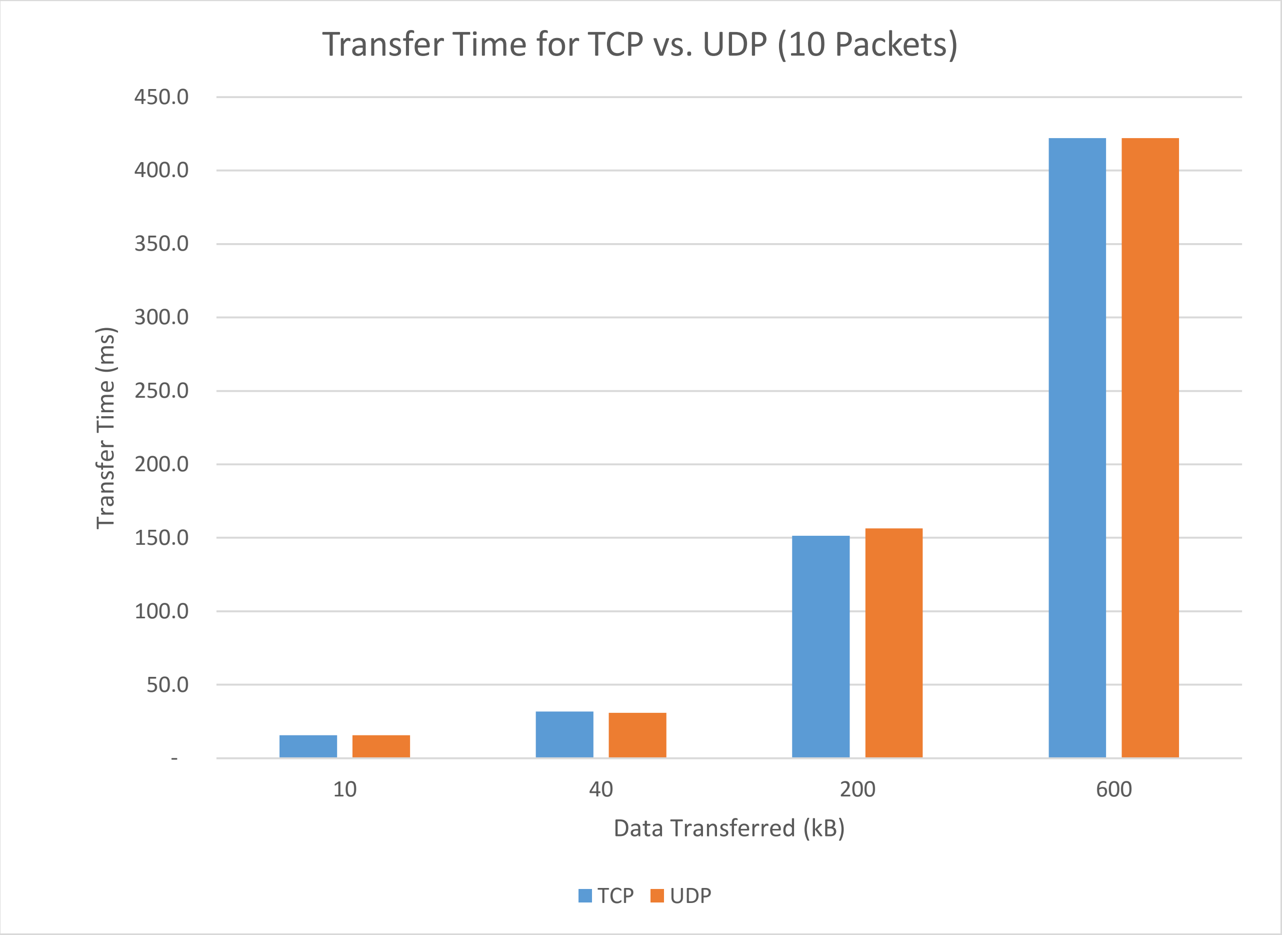
In order to test the protocols and expect varying results to find an informative conclusion, the program was required to be dynamic in forms data would be sent in. By allowing the user to specify sizes and amounts of data to send, studies could be conducted with varying constraints. These varying data sizes were: 1KB, 4KB, 20KB, and 60KB. Data sizes were tested with both 10 and 100 packets, for both UDP and TCP. Tests were then conducted 3 times for each specification, and their averages were taken for final studies.

Unfortunately, data may be found to be slower than expected, due to a UI and active statistics which must be updated as the program runs. Attempts to optimize this was done by allowing the user to disable visual representation of the data being loaded client sided, and read server sided. This change led to a significant change in speed, however, statistics still add a noticeable overhead to the connection speed.

Analysis

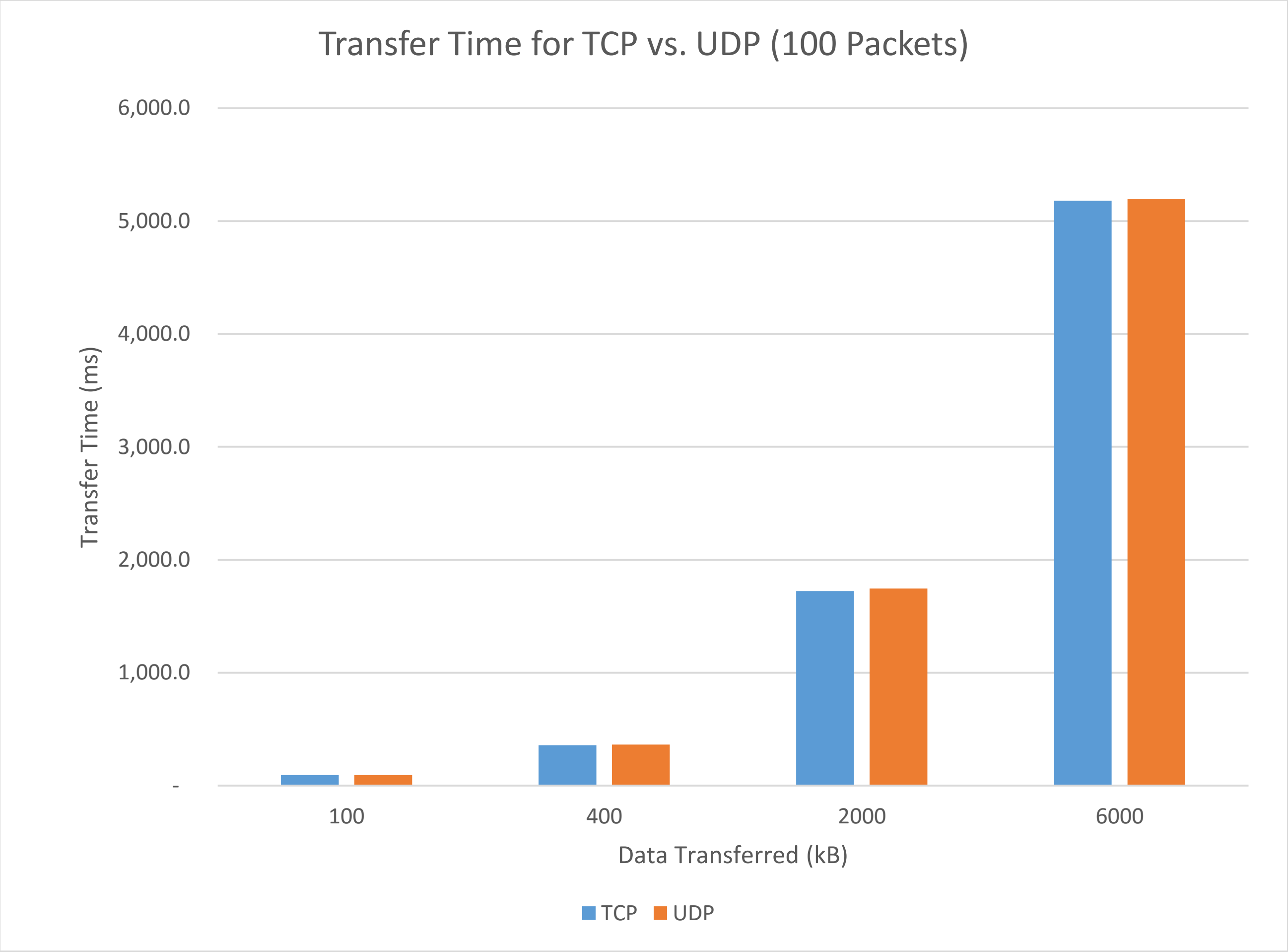
Data analysis can be found for 1KB, 4KB, 20KB, and 60KB packet sizes, in counts of 10 and 100 packets each. Data analysis will be outlined with tables and their results discussed. The end of the analysis discussion will have raw tablet data for exact numbers.

Transfer Time for Packet Size of 10



This table represents the average transfer time in milliseconds from sending 10 packets each of size 1kB, 4kB, 20kB, and 60kB. As seen above, TCP and UDP are very similar in speed, where TCP shows a slightly faster speed in some cases. There wasn't a significant difference found in speed or packet loss, as data returned no packet loss for both TCP and UDP.

Transfer Time for Packet Size of 100



This table represents the average transfer time in milliseconds from sending 100 packets each of size 1kB, 4kB, 20kB, and 60kB. Similar to the test on 10 packets, the speed at which the packets for TCP and UDP are very similar. TCP again shows a slightly faster speed, which may depend solely on program overhead, as transfer speeds are milliseconds apart. Again both UDP and TCP witnessed no packet loss while sending each packet size.

Tabulated Data of All Tests and Average

The below table contains data for TCP data transfers and calculated averages on the results.

	Protocol	TCP							
	Packets Sent	10				100			
	Packet Size	1024	4096	20480	61440	1024	4096	20480	61440
Test 1	Transfer Time (ms)	16.0	32.0	141.0	406.0	94.0	359.0	1,719.0	5,203.0
	Data Transferred (kb)	10.0	40.0	200.0	600.0	100.0	400.0	2,000.0	6,000.0
	Packets Lost	-	-	-	-	-	-	-	-
Test 2	Transfer Time (ms)	15.0	32.0	156.0	438.0	93.0	360.0	1,718.0	5,178.0
	Data Transferred (kb)	10.0	40.0	200.0	600.0	100.0	400.0	2,000.0	6,000.0
	Packets Lost	-	-	-	-	-	-	-	-
Test 3	Transfer Time (ms)	16.0	31.0	157.0	422.0	94.0	359.0	1,734.0	5,156.0
	Data Transferred (kb)	10.0	40.0	200.0	600.0	100.0	400.0	2,000.0	6,000.0
	Packets Lost	-	-	-	-	-	-	-	-
Average	Transfer Time (ms)	15.7	31.7	151.3	422.0	93.7	359.3	1,723.7	5,179.0
	Data Transferred (kb)	10.0	40.0	200.0	600.0	100.0	400.0	2,000.0	6,000.0
	Packets Lost	-	-	-	-	-	-	-	-

The below table contains data for TCP data transfers and calculated averages on the results.

	Protocol	UDP							
	Packets Sent	10				100			
	Packet Size	1024	4096	20480	61440	1024	4096	20480	61440
Test 1	Transfer Time (ms)	16.0	31.0	156.0	422.0	94.0	359.0	1,734.0	5,188.0
	Data Transferred (kb)	10.0	40.0	200.0	600.0	100.0	400.0	2,000.0	6,000.0
	Packets Lost	-	-	-	-	-	-	-	-
Test 2	Transfer Time (ms)	16.0	31.0	157.0	406.0	94.0	359.0	1,734.0	5,204.0
	Data Transferred (kb)	10.0	40.0	200.0	600.0	100.0	400.0	2,000.0	6,000.0
	Packets Lost	-	-	-	-	-	-	-	-
Test 3	Transfer Time (ms)	15.0	31.0	156.0	438.0	94.0	375.0	1,766.0	5,188.0
	Data Transferred (kb)	10.0	40.0	200.0	600.0	100.0	400.0	2,000.0	6,000.0
	Packets Lost	-	-	-	-	-	-	-	-
Average	Transfer Time (ms)	15.7	31.0	156.3	422.0	94.0	364.3	1,744.7	5,193.3
	Data Transferred (kb)	10.0	40.0	200.0	600.0	100.0	400.0	2,000.0	6,000.0
	Packets Lost	-	-	-	-	-	-	-	-

Conclusion

Following the data taken using the TCP/UDP Client/Server conclude that there is little difference between UDP and TCP on a local network where packet loss is unlikely. The unlikely packet loss allows UDP to communicate without packet loss, increasing speed of the connection.

TCP may appear similar to UDP due to ignoring the setup of connection with three-way handshake. This method of ensuring connection should require 3 extra packets which TCP must account for while UDP doesn't (as it doesn't operate under a connection).

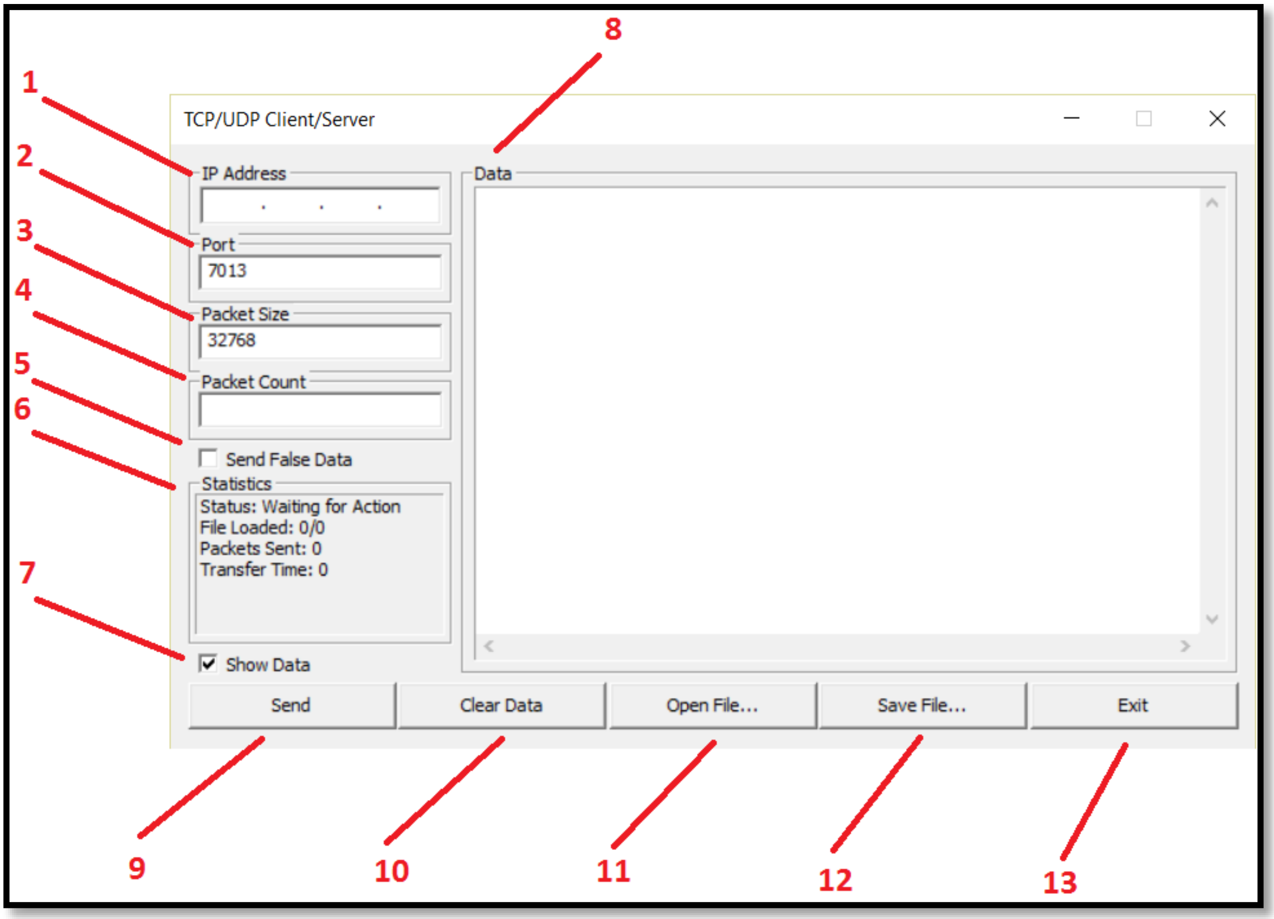
Another possibility is the UI of the program is limiting the speed of the two connections, where UDP may be able to read packets faster if it is able to. However, this was tested by disabling most UI elements (such as displaying packet data), and while results did increase significantly, TCP and UDP still matched in speeds.

Regarding the lack of packet loss in both connects led to speculation that UDP may not be operating correctly as "true" UDP. However, this speculation was discarded after a test with extremely small packet sizes were communicated over TCP and UDP (64 bytes, 10,000 packets). The results ended with a nearly 50% packet loss on UDP, while a 0% packet loss in TCP. This new indication in data shows that UDP is operating correctly, but my not be reaching its maximum threshold for reading speeds. The graph below represents testing lower packet sizes with high packet count.

	Protocol	TCP		UDP	
	Packets Sent	10,000		10,000	
	Packet Size	16	64	16	64
Test 1	Transfer Time (ms)	984.0	984.0	156.0	735.0
	Data Transferred (kb)	156.3	625.0	156.3	625.0
	Packets Lost	-	-	5,012.0	4,993.0
Test 2	Transfer Time (ms)	985.0	990.0	141.0	768.0
	Data Transferred (kb)	156.3	625.0	156.3	625.0
	Packets Lost	-	-	5,013.0	5,012.0
Test 3	Transfer Time (ms)	991.0	987.0	141.0	766.0
	Data Transferred (kb)	156.3	625.0	156.3	625.0
	Packets Lost	-	-	5,012.0	5,013.0
Average	Transfer Time (ms)	986.7	987.0	146.0	756.3
	Data Transferred (kb)	156.3	625.0	156.3	625.0
	Packets Lost	-	-	5,012.3	5,006.0

Usage

This section of the manual explains the functions of the program and how to use them. Below is an image of the program upon start up. Each element of the program is labelled with a number, which will be later referenced for an explanation.



IP Address: . . .

Port: 7013

Packet Size: 32768

Packet Count:

□ Send False Data

Statistics

```
Status: Waiting for Action
File Loaded: 0/0
Packets Sent: 0
Transfer Time: 0
```

☑ Show Data

1- IP Address Field

This field is for entering an IP for the client to use to connect to a server. It is not useable by the server.

2- Port Field

This field is for binding the client/server to a port, to listen for connections.

3- Packet Size Field

This is size of packet the client will be sending (if client program), or the size of packets the server will read in if it is a server program.

4- Packet Count Field

This will indicate how many "junk data" packets to send to the server if the client decides to select this option. It is unused by the server.

5- False Data Check Box

This check box is for a client program. If the client decides to send "junk data" to the server, this box should be checked.

6- Statistics Area Box

This un-editable field shows common statistics on the program. The statistics will include the current status of the program, the amount of characters loaded from a file (if it is a client), the amount of packets sent/received (depends if the program is a client or server), and the data communication speed of sending/receiving the packets.

7- Data Check Box

This box is for server and client optimization. It will disable the Data field, and some other operations which take extra resources when sending or receiving data.

8- Data Field

This field will show the data read from a client (if a server), or show data loaded from a file, or typed in from the user (if a client program).

9- Preform Action Button

(Server) Allow Connections: This button will open a socket, bind to it, and begin accepting connections from a server.

(Client) Send: This button will open a socket, bind to it, and attempt to send data to a running server, defined either by the data field, or the junk data selection.

10- Clear Data Button

This button will remove all data in the Data box, and reset statistics.

11- Open File Button

This will allow the user to load a file from somewhere on the computer.

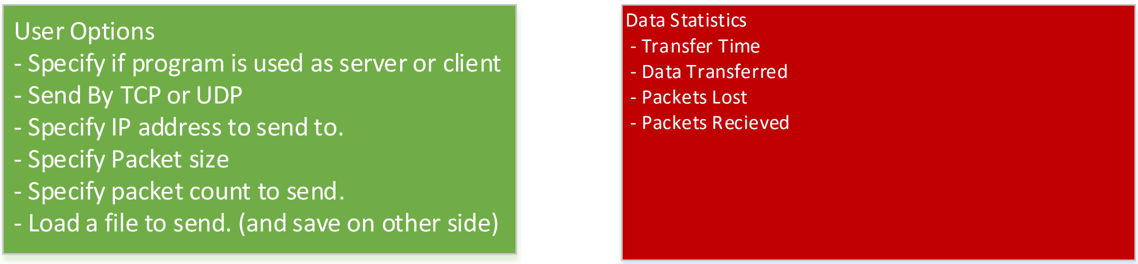
12- Save File Button

This will allow the user to save a file to somewhere on the computer.

13- Exit Button

This will close the program, closing connections and sockets associated with the ending program.

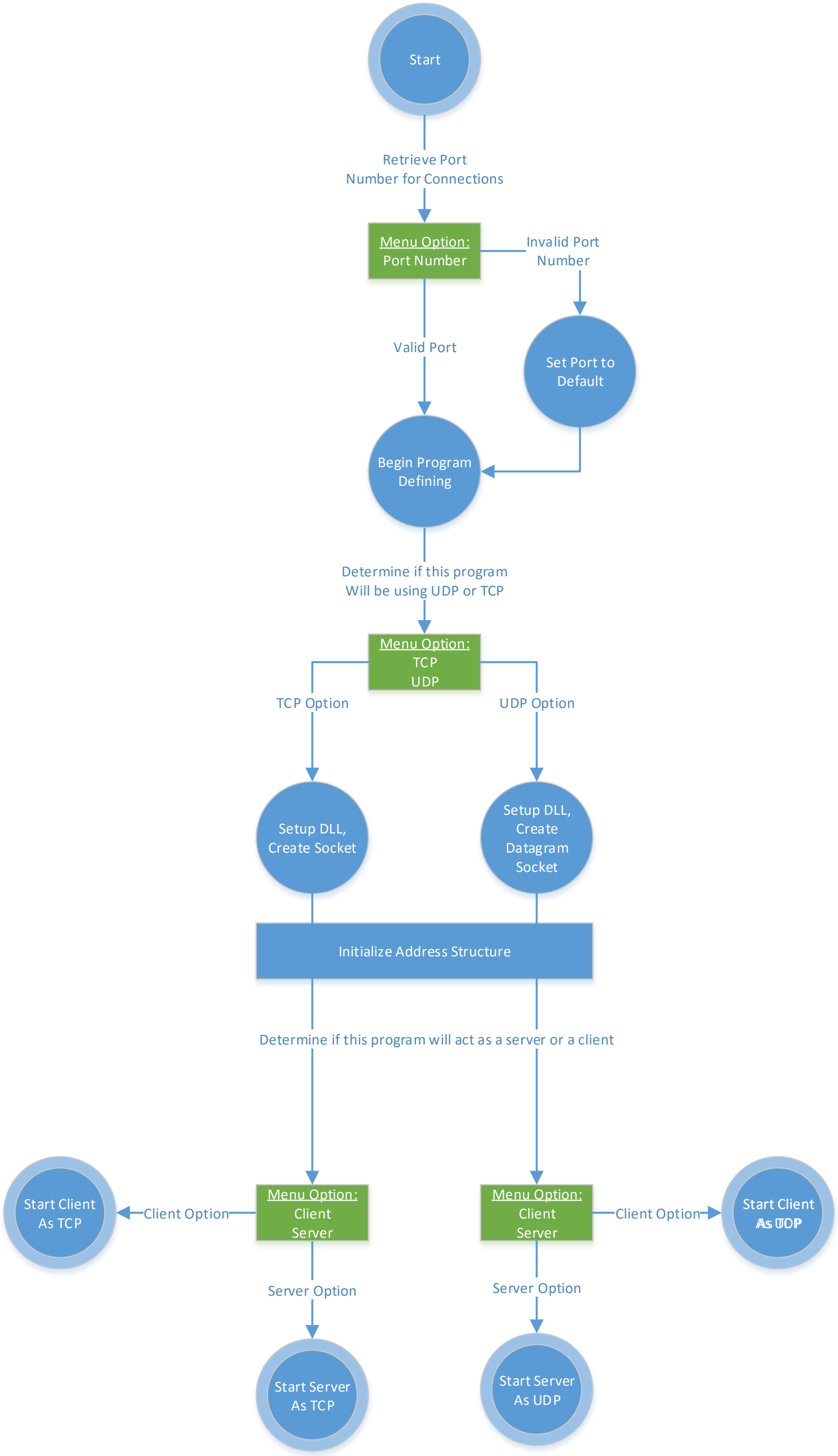
Program Design – Diagram



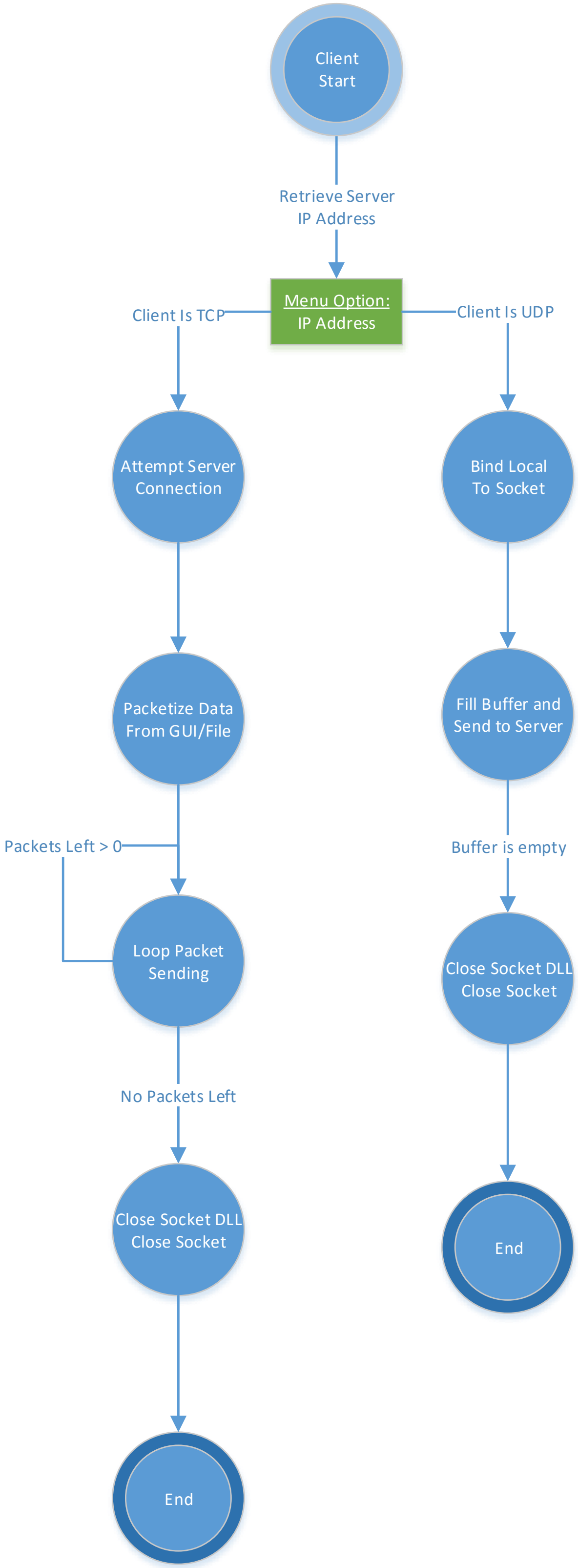
User Options
- Specify if program is used as server or client
- Send By TCP or UDP
- Specify IP address to send to.
- Specify Packet size
- Specify packet count to send.
- Load a file to send. (and save on other side)

Data Statistics
- Transfer Time
- Data Transferred
- Packets Lost
- Packets Recieved

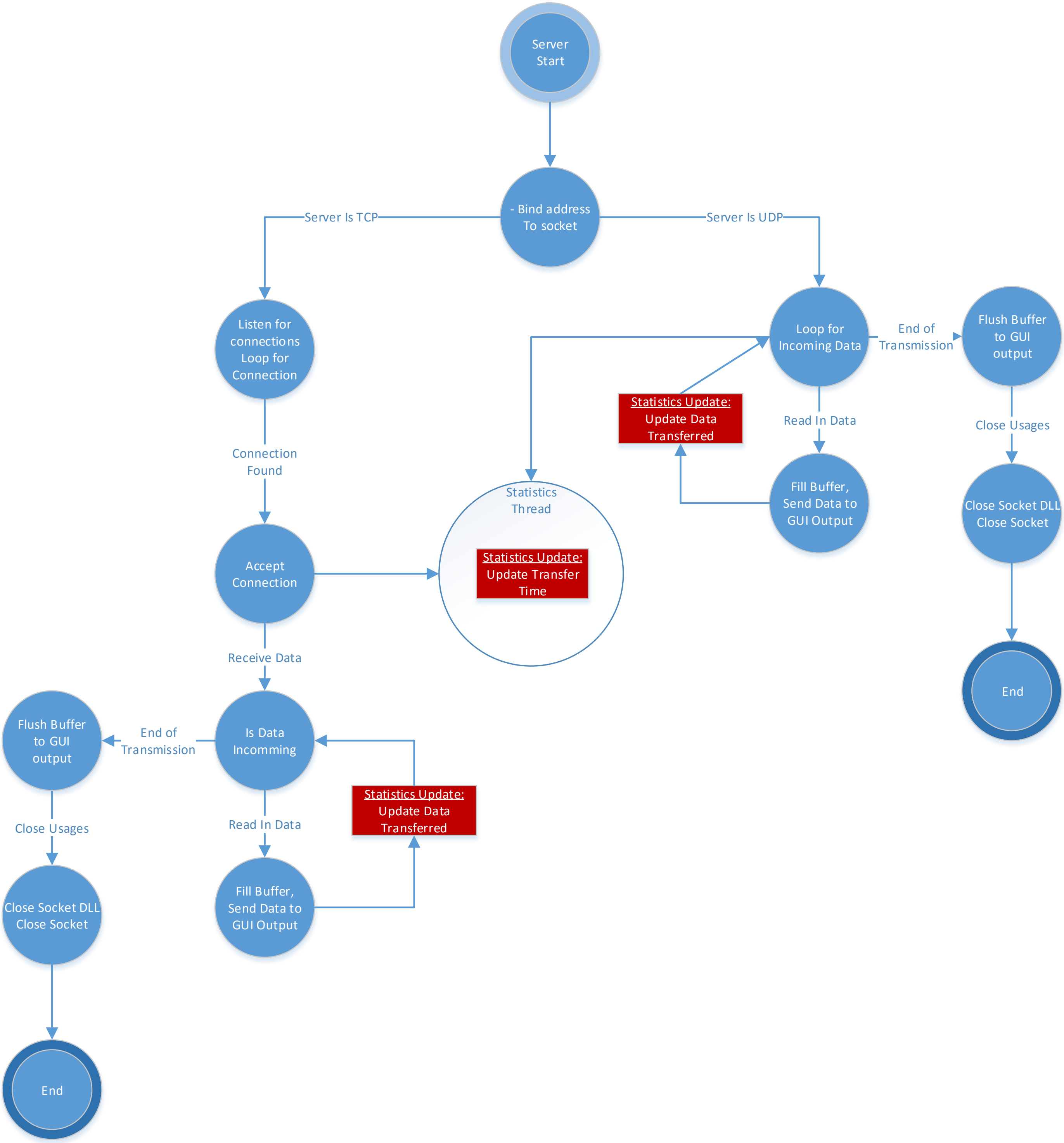
Main Program Setup



Client Program


Client Start
Retrieve Server IP Address
Menu Option: IP Address
Client Is TCP
Client Is UDP
Attempt Server Connection
Bind Local To Socket
Packetize Data From GUI/File
Fill Buffer and Send to Server
Packets Left > 0
Buffer is empty
Loop Packet Sending
Close Socket DLL Close Socket
No Packets Left
Close Socket DLL Close Socket
End
End

Server Program



Program Design – Pseudo Code

(1) START

```
Start WinMain
    Instantiate Setup Dialog
    Instantiate statistics struct
    Start WndProc Loop
```

(2) Create Default Socket

```
On Program Select Message:
    User selects program type
    User selects protocol type
    Instantiate default socket (protocol type, program type)
    Start Main Window
```

(3) Create Connections

```
Create socket
Bind socket
If program is client
    Connect to host
    If TCP
        Attempt to connect to server
    If UDP
        Attempt to send data to server
Else if program is server
    Create Read Thread
    If TCP
        loop for connection
            If connection
                Read from connection to buffer
                Update statistics
                Show buffer to screen
                If transmission end
                    Close client connection
    If UDP
        Loop for read to buffer
        If characters read
            Update statistics
            Show buffer to screen
```

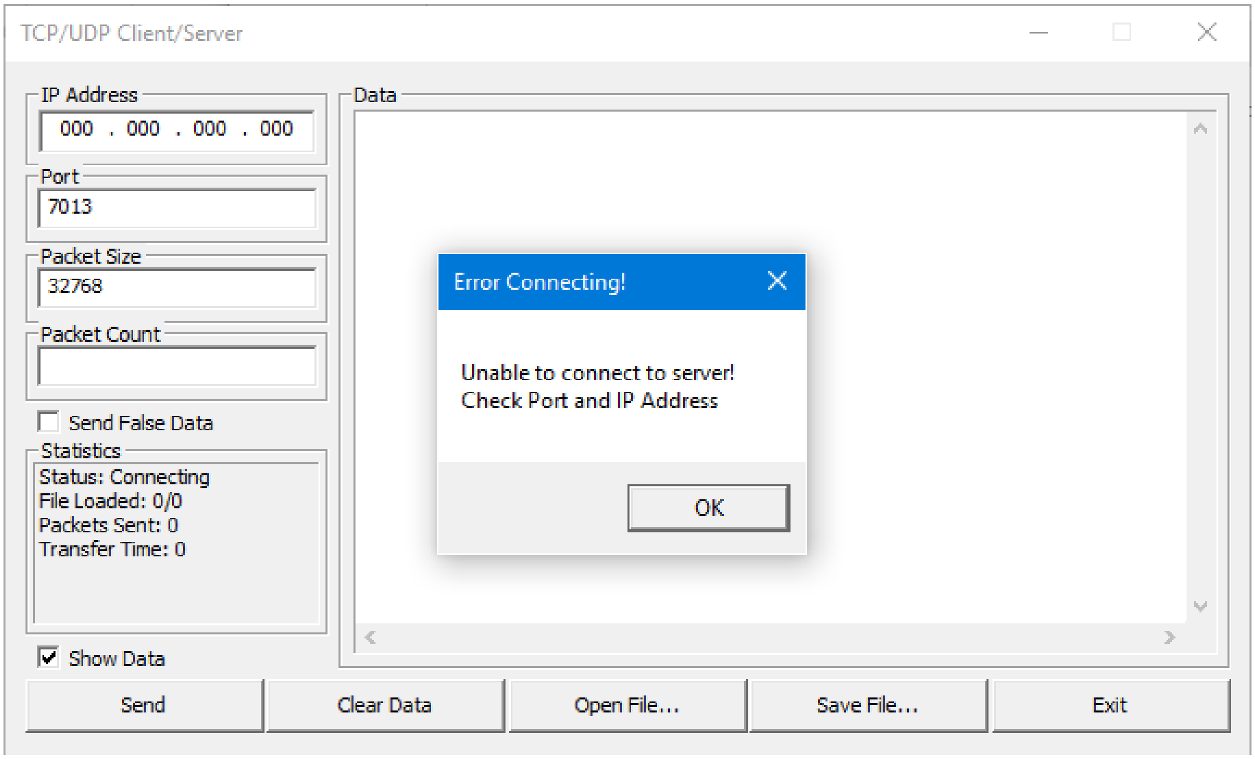
(5) WndProc Message Listener

```
If message open file
    User open file
        Read file to UI
If message send
    Create connections for program type
If message save file
    User open file location
        Save file from UI to disk
If message clear
    Delete data from UI
    Clear statistics
If message quit/exit
    Close connection
    Close socket
    End program
```

Test Cases

Test #	Test Description	Test Result	Proof	Pass?
1	Entering an incorrect IP address Client sided.	Upon entering an incorrect IP address and attempting to connect to a server using the “Send” button, a prompt will tell the user that the host name isn’t accessible.	See Fig 1.	YES
2	Entering an incorrect IP address Client sided.	Upon entering an incorrect port number and attempting to connect to a server using the “Send” button, a prompt will tell the user that the host isn’t accessible.	See Fig 2.	YES
3	Entering an incredibly small packet size, Client or Server sided.	The program will not allow the user to enter a packet size smaller than 4, in order to allow important characters for receiving to be placed in the buffer. A prompt will appear for the user.	See Fig 3.	YES
4	Opening a file.	Clicking the “Open File…” button will open a windows dialog which allows the user to look for a file and open it.	See Fig 4.	YES
5	Saving a file.	Clicking the “Save File…” button will open a windows dialog which allows the user to look for a directory to save a file to.	See Fig 5.	YES
7	Receiving data from a client.	Sending data after a successful connection on both UDP and TCP are possible.	See Fig 6.	YES
8	Sending data to a server.	Receiving data from a client after accepting their connection on both TCP and UDP are possible.	See Fig 7.	YES

Test 1 – Figure 1



IP Address: 000 . 000 . 000 . 000

Port: 7013

Packet Size: 32768

Packet Count:

☐ Send False Data

Statistics

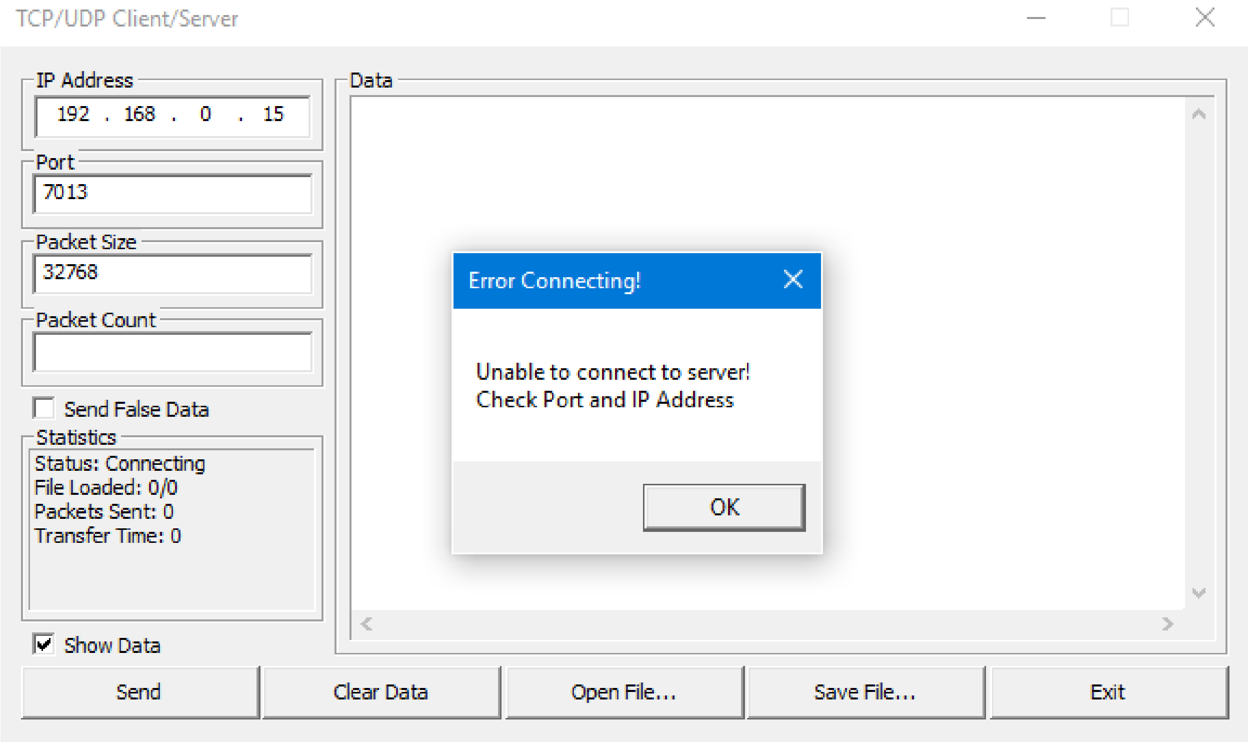
Status: Connecting
File Loaded: 0/0
Packets Sent: 0
Transfer Time: 0

☑ Show Data

Error Connecting!

Unable to connect to server!
Check Port and IP Address

Test 2 – Figure 1



IP Address: 192 . 168 . 0 . 15

Port: 7013

Packet Size: 32768

Packet Count:

☐ Send False Data

Statistics

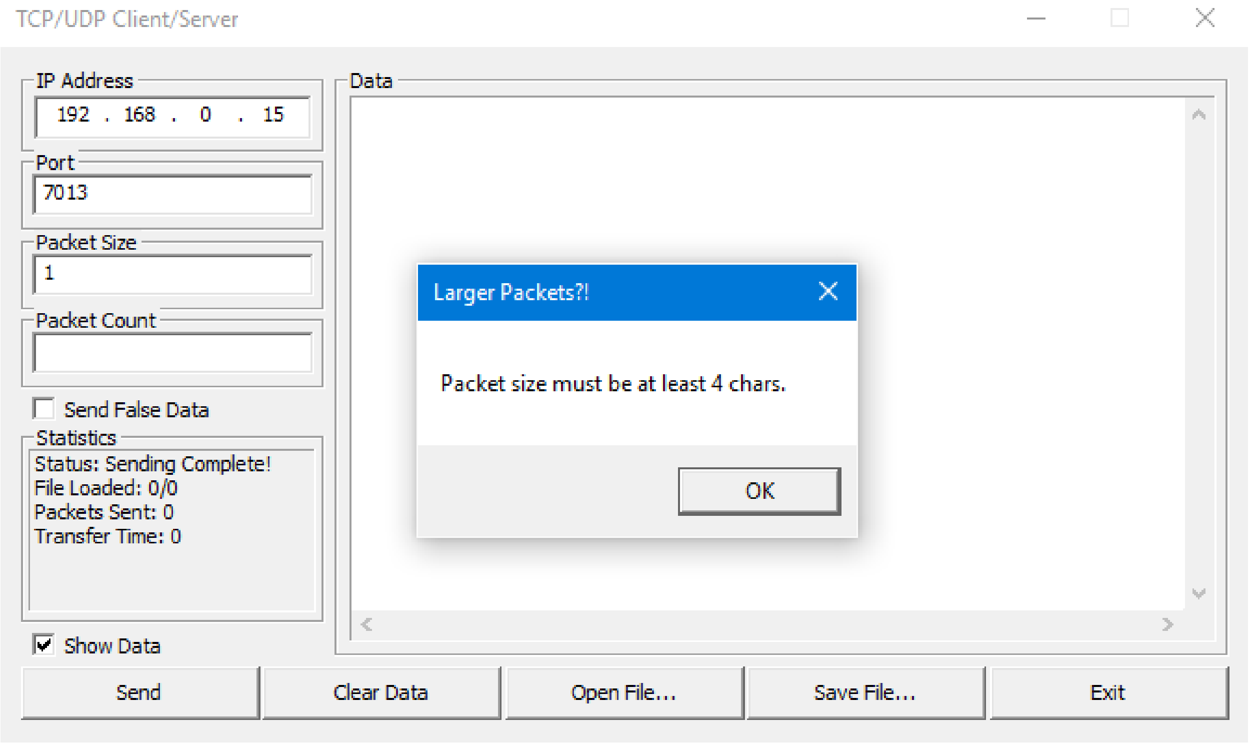
Status: Connecting
File Loaded: 0/0
Packets Sent: 0
Transfer Time: 0

☑ Show Data

Error Connecting!

Unable to connect to server!
Check Port and IP Address

Test 3 – Figure 1



TCP/UDP Client/Server

IP Address: 192 . 168 . 0 . 15

Port: 7013

Packet Size: 1

Packet Count:

□ Send False Data

Statistics

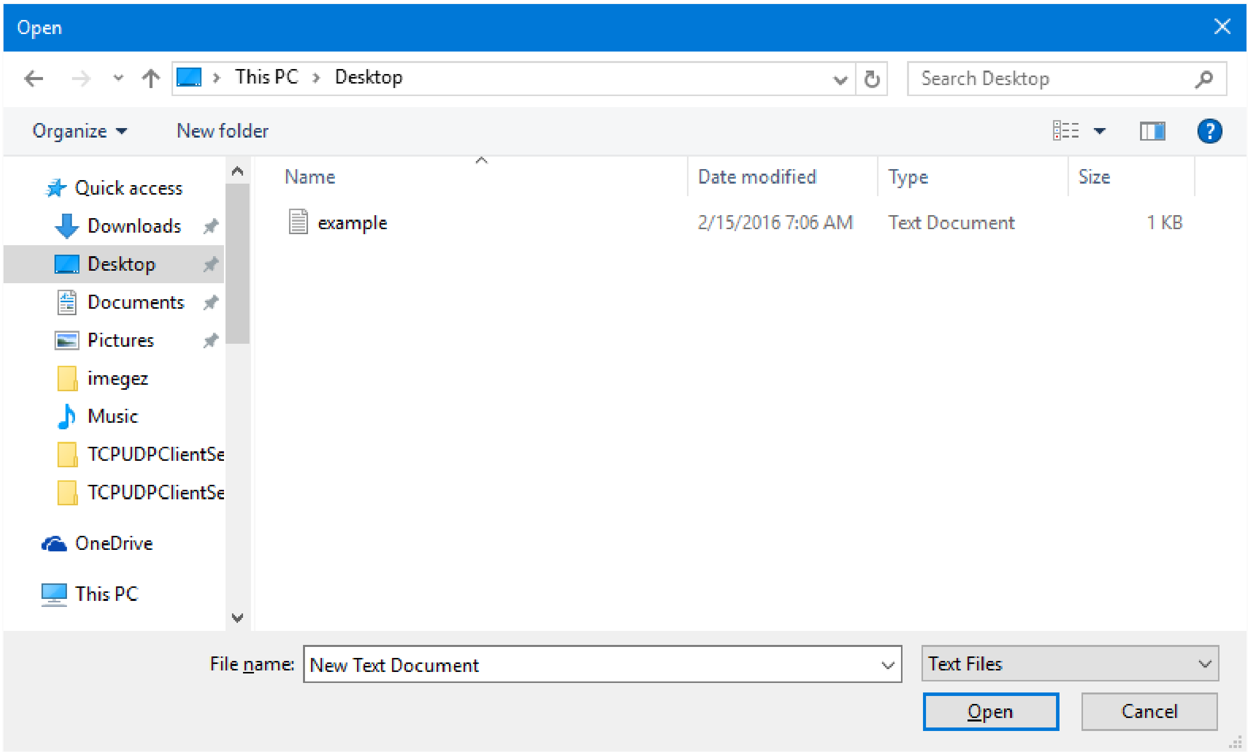
Status: Sending Complete!
File Loaded: 0/0
Packets Sent: 0
Transfer Time: 0

☑ Show Data

Larger Packets?!

Packet size must be at least 4 chars.

Test 4 – Figure 1



Open

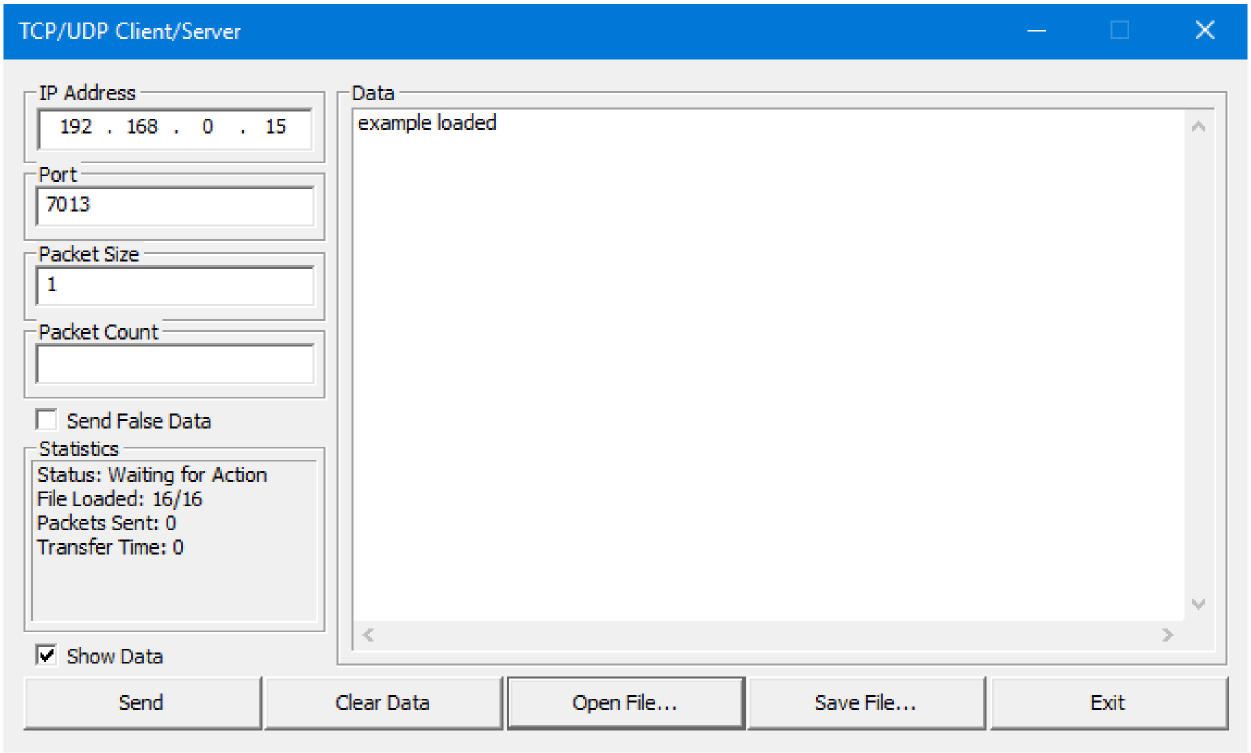
This PC › Desktop

Name	Date modified	Type	Size
example	2/15/2016 7:06 AM	Text Document	1 KB

File name: New Text Document

Text Files

Test 4 – Figure 2



TCP/UDP Client/Server

IP Address: 192 . 168 . 0 . 15

Port: 7013

Packet Size: 1

Packet Count:

☐ Send False Data

Statistics

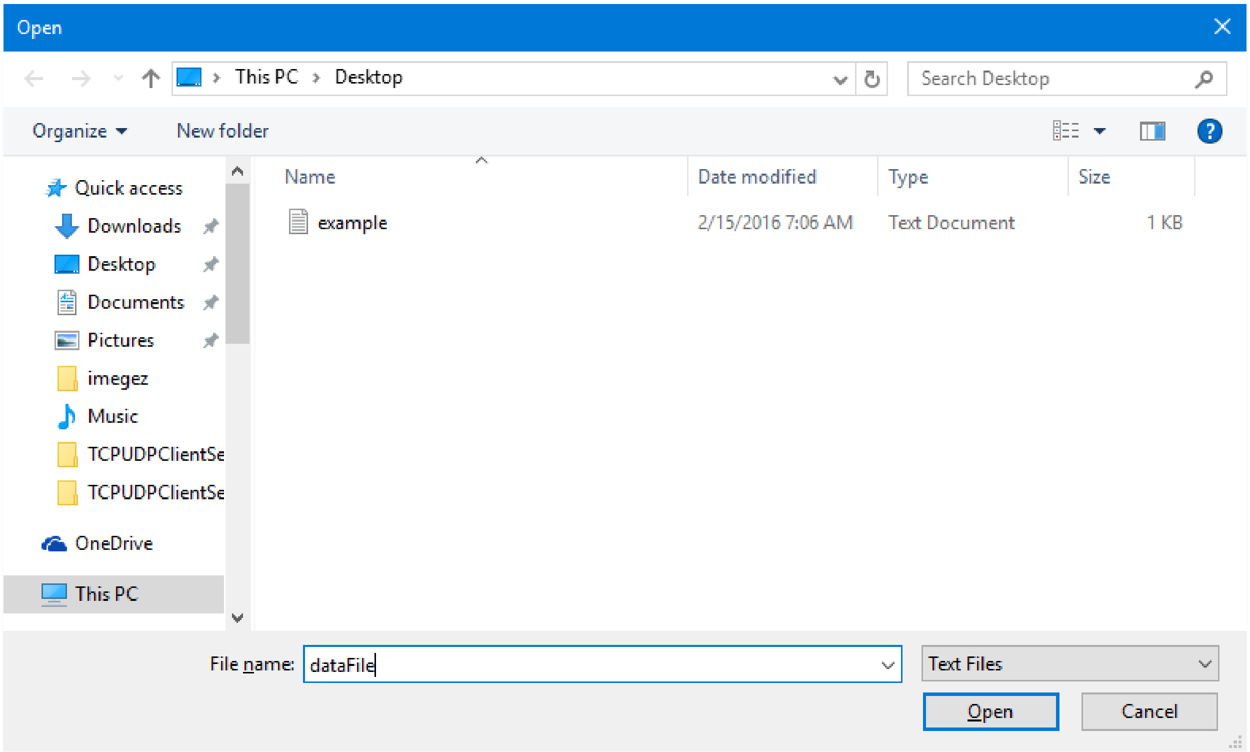
Status: Waiting for Action
File Loaded: 16/16
Packets Sent: 0
Transfer Time: 0

☑ Show Data

Data

example loaded

Test 5 – Figure 1



Open

This PC › Desktop

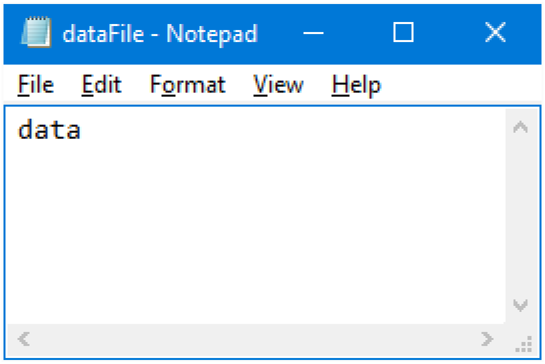
Search Desktop

Name	Date modified	Type	Size
example	2/15/2016 7:06 AM	Text Document	1 KB

File name: dataFile

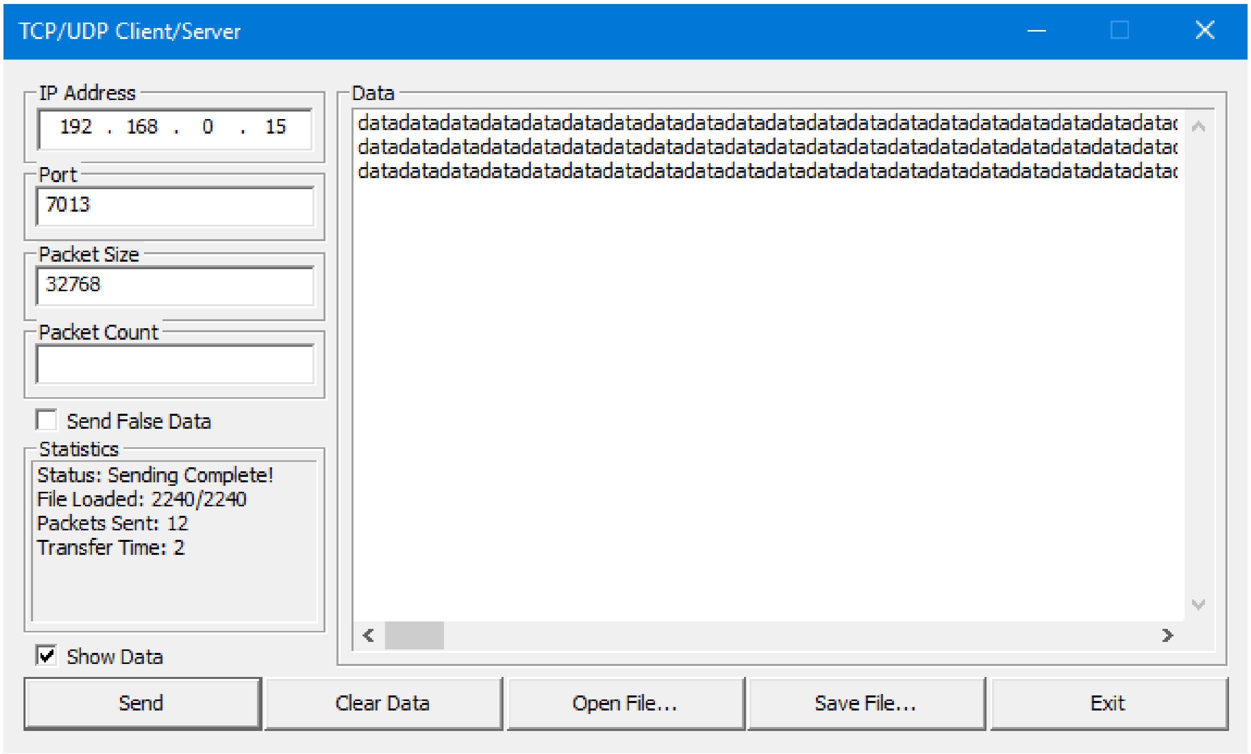
Text Files

Test 5 – Figure 2



data

Test 6 – Figure 1



IP Address: 192 . 168 . 0 . 15

Port: 7013

Packet Size: 32768

Packet Count:

□ Send False Data

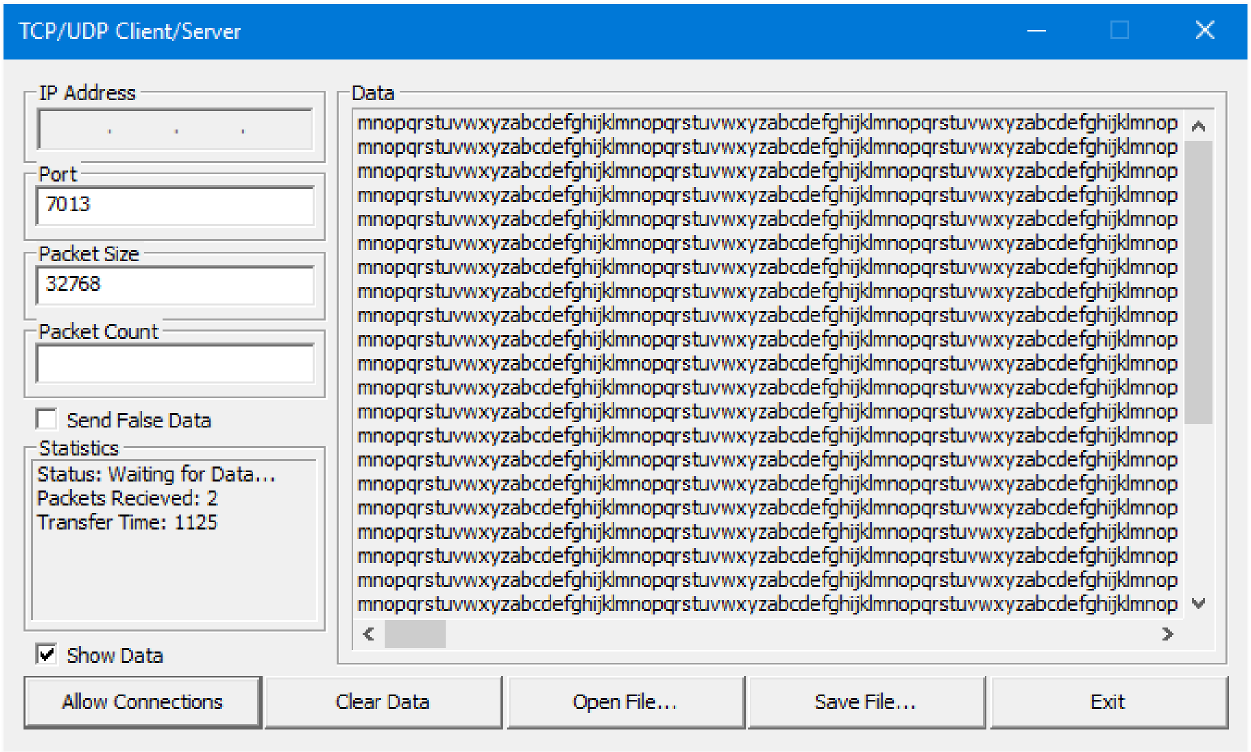
Statistics

```
Status: Sending Complete!
File Loaded: 2240/2240
Packets Sent: 12
Transfer Time: 2
```

☑ Show Data

```
datadatadatadatadatadatadatadatadatadatadatadatadatadatadatadatadatadatadatadata
datadatadatadatadatadatadatadatadatadatadatadatadatadatadatadatadatadatadatadata
datadatadatadatadatadatadatadatadatadatadatadatadatadatadatadatadatadatadatadata
```

Test 7 – Figure 1



IP Address: . . .

Port: 7013

Packet Size: 32768

Packet Count:

☐ Send False Data

Statistics

Status: Waiting for Data…
Packets Recieved: 2
Transfer Time: 1125

☑ Show Data

Data

```
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
mnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop
```