Computer Science 354 Group 35: Project 2 Report

Rivaldo Ponnadu - 20691815

Lyndon Fooy – 21028540

Report Overview:
In this project we were tasked with creating a reliable TCP and UDP file transfer system.
In this report we share how we developed and implemented the program as well as what the program can and cannot do and a brief analysis of the data structures and designs implemented throughout development.

Unimplemented features:
We experienced some issues when sending large files over TCP. We were unable to remove this file size limit for TCP by the due date.

Additional Features Implemented:
We added a statistical printout measuring the time taken to send the file, the throughput and number of packages that had to be resent in a UDP transfer. The TCP implementation only prints out the number of bytes sent.

Description of files:
We have six java files directly related to the submissions functionality as well as a generic timer class.

1. Client.java is our terminal implementation of the UDP sender.
2. Server.java is the corresponding terminal implementation of a UDP receiver.
3. TCPClient.java is as the name suggests a TCP sender implemented for terminal use.
4. TCPServer.java pairs with the above file as a terminal implemented TCP receiver.
5. ClientGUI.java is a very simple GUI capable of sending files over both TCP and UDP protocols using the above files.
6. ServerGUI.java is capable of receiving files over TCP and UDP and presents the user with an extremely simple GUI.

Program Description:

To use the terminal implementation of either TCP or UDP a user must first run the relevant server. For UDP file sending the user need not add any additional arguments however for TCP file sending the user must add an argument for the name the received file will be saved as. This server will then wait for a client to connect. Once connection occurs the file will be sent in datagram packets

Experiments:
We performed a few experiments and found that TCP file sending had higher throughput and shorter sending times than UDP sending. TCP also sends fewer packets and is more reliable overall.

Issues Encountered:
We encountered issues in sending large files over TCP protocols. We suspect this has to do with our TCP client implementation. More specifically how the bytePkg is processed in the while loop and how the count variable might be limited in accessing possible values stemming from the while loop implementation as mentioned before.

Compilation:
Using terminal locate the src file of the project and enter the command “make”. This will create all relevant java classes. Removing the created classes can be done by typing the “make clean” command into a src terminal.

Execution:
For terminal execution of the UDP protocol first run a server. The server requires no arguments. The sender must then run a client from terminal with the arguments of server IP, name or location of file to be sent and name of file on the server side. The client should then connect to the server and send the selected file.

The TCP terminal system also requires a server to be run first with the filename for the received file as an argument. The corresponding client must then be run with arguments for the server IP and name or location of the file being sent. The client should then connect to the server and send the file.

For GUI implementation the user can run either client or server GUI’s in any order. However before a file can be sent over this system a user should first interact with the server GUI to initialise a server for the sending protocol being used. In the case of UDP simply clicking the button will suffice however with TCP a user should first fill the filename textfield with the desired name the received file should be saved as.
After this a user should interact with the client side GUI and fill in the IP textfield with the IP of the receiving server as well as the name or location of the file to be sent. When using the TCP protocol filling in these fields is sufficient however when sending over UDP a user must also fill out the final text field for the name the file will receive at the server location.

Libraries:

Java.util
Java.io
Java.net
Java.awt
Javax.swing