Reliable UDP Chat Room

Networking Project Spring 2019

Nathan Martin

Computer Science Department

University of Memphis

Memphis, TN US

nmrtin11@memphis.edu

Nathan Mar†  
 Department Name  
 Institution/University Name  
 City State Country  
 email@email.com

ABSTRACT

In this project, the main purpose is to create a chat room that uses UDP socket programming; but making sure that it provides reliable data transfer along with no data loss and no data corruption.

This project was proposed by Xing Gao, a professor at the University of Memphis that he assigned COMP 3825 students – or Networking and Information Assurnace.

The idea of the project is for me to learn the importance of TCP along with how TCP is implemented and how the different parts of TCP work with everyday use when it comes to networks and communications.

The idea is to use a UDP socket, but for me to implement my own forms of TCP on top of the UDP socket so that there is reliable message transfer, file transfer, image transfer, a GUI, message charting, along with allowing multiple users to be in the chat room at the same time.

While I did not fully finish all of the materials required, they were attempted (except the GUI), and some of them (file transfer go-back-N), I did not have success in finishing.

Notable successes of the program is – multiple users can connect to a chat room and communicate with each other, emojis, file charting, name customization, and attempted reliable message and file transferring.

The project itself is very interesting to me and I plan to continue learning more about the importance of TCP in the use of everyday life on the web, instant messaging, and much more.

There is a README.md that will describe how to download the programs along with running them for anyone’s testing.

The link to the project is found [here](https://github.com/nathanjmartin/UDP-Chat-Room).

CCS CONCEPTS

• None

KEYWORDS

UDP – User Datagram Protocol (unreliable communications protocol)

TCP – Transmission Control Protocol (reliable communications protocol)

Emoji – emoticon to represent a user’s emotion

GUI – graphical user interface

Go-back-N – algorithm/process of providing reliable data transfer with the user of a timer/timeout function along with sequence/acknowledgment numbers so that if data is lost, it will be sent again until the program acknowledges that it has been received.

1 Problem, Motivation, and Design Goals

The problem, motivation, and design goals behind this project was given by Professor Xing Gao from the University of Memphis, who is teaching Networking and Information Assurance during the Fall 2019 semester.

He provided detailed goals along with explaining the reasoning behind a project like this is – which will be broken down below in the following sections.

These sections will explain the problem with the project, the motivation behind a project like this, and all of the design goals that were required for this project.

1.1 Problem

UDP is not a reliable transport service. The problem with this project is that we have to use UDP to send messages, emojis, and files, which will for sure have data loss. The reason this is a problem is that since UDP is not reliable, it is unlikely that the messages, emojis, and files will be delivered consistently with no problems; this is a problem because normally, with a chatting application, it is generally important to consistently deliver messages with no data loss, no matter what the data is.

1.2 Motivation

The motivation behind this project is to better understand how TCP works in todays world to provide reliable data transfer compared to that of UDP. Along with this, the motivation has been for students to try and program reliable data transfer on top of UDP; in other words, to create our own version of TCP on top of UDP so that we can understand the processes of TCP, which include the handshake, go-back-N, along with others.

1.3 Design Goals

The Design Goals within this project was to have a Basic UI, support both text, emoji, pictures. Along with these, there is to be file transfer, and charting history (where a user can see all their messages or all previous messages within the chat room!)

Some more design goals were to create a form of a TCP handshake over top of UDP to establish connections between the clients and the servers, along with implementing a Go-Back-N algorithm that will reliably make sure that all data is sent and received without any data loss or corruption. A bonus was to have multi-user chat room that would allow multiple users to chat at the same time. This project could have been done alone or with a group of 2 and from my understanding groups of 2 would have to have a nicer GUI to go along with the project.

2 Approach and Technique

The Approach and Technique that I took seemed logical to me. My initial idea was to first establish a basic connection between a client program and a server program. My idea was that once I got this working, I could begin doing more complex stuff with both the client and the server programs. This would lead me to begin to start working on the handshake process, file transfer, emojis, Go-Back-N implementations, along with the file charting and other requirements needed for the project.

3 Evaluation plan

For my evaluation plan for the project and the requirements needed for it; I just planned on trying to implement a certain feature and then testing it to see if there were any bugs that were caused by me implementing that feature. If there was a bug that occurred, I would continue changing and testing that functionality until it worked quickly. This was my approach throughout the entire time I spent on the project.

I originally had basic file transfer working and I tried implementing a go-back-N with the file transfer, however that has started causing problems and I am still trying to test and see why that is not working correctly. I am thinking the problem has to do with the server not handling the sequence numbers correctly.

Another evaluation I had to do was with the emojis; my program only supports 3 emojis, and my GUI has just been the terminal windows. Because of this, the 3 emojis show up as a box with a question mark in the middle as my terminal does not support the emoji’s format.

Other than those, message charting works, messaging works, handshaking works, but still have those problems as listed above.

4 Conclusion and future work

In conclusion, I have most of the project itself in tact and working for a group of one. The requirements are to have python 3.7.5 running on your PC, and to start the server.py program before starting the chat.py programs (client program); this is because the server has to be up and running before the chat programs can begin the handshake process.

File transferring with go-back-N has caused me some issues, which is why I did not end up using Tkinter for my GUI so that emojis will be displayed correctly. To use an emoji, you just need to type on of the following: ‘: )’, ‘xD’, ‘:D’. The program will add the emoji next to those symbols whenever you send one to all other users.

As for future work, I plan on continuing to try and figure out how to make this file transfer work with go-back-N so that I can support files that are the size of multiple gigabytes as it is an interesting concept to me. I also want to eventually keep working on this interesting project idea and to develop a GUI and possibly send the program to my friends so that we can all talk to each other while using it.

REFERENCES

[1] The Python Standard Library <https://docs.python.org/3/library/>

Conference Name:ACM Woodstock conference

Conference Short Name:WOODSTOCK’18

Conference Location:El Paso, Texas USA

ISBN:978-1-4503-0000-0/18/06

Year:2018

Date:June

Copyright Year:2018

Copyright Statement:rightsretained

DOI:10.1145/1234567890

RRH: F. Surname et al.

Price:$15.00