Greg Cernera

Professor Arias

CMPT 220

7 April 2017

Milestone

For my project, I am constructing an instant messaging system that is specifically designed for sending encrypted messages between users and maintain anonymity. This program will be made for users who wish to keep their conversations private for whatever reason. I also have desires for this program to also be a sharing file system that encrypts and decrypts files sent between the users. However, if the instant messaging tool is not completed with ample time to add the file sharing feature, then the addition will not be implemented and the program will solely be an encrypted instant messaging program.

I wanted to create this application because of my interest in cryptography and secrecy on the internet. I wanted to build a program with the intention of making a tool that hackers would use to relay messages to other computers secretly, and not have to run the risk of exposure by using programs such as iMessage, Google, or any other monitored messaging system. In order for a program like this to work and function in the real world, the program would have to set up a server and send a direct signal to an IP address and port number that the client user provides. In the case that someone does intercept the signal between the two hosts, the instant messages being sent will still be encrypted prior to sending, and would have to be decrypted when the other user receives the message.

In order to understand the encryption and decryption part of it, one must understand how the program is actually constructed. There are two separate projects: the Server project and the Client project. This program follows the server/client setup, similar to that of an Internet Relay Chat. One user (User1) will play the role of the server, and the other user (User2) will be the client that has to connect to User1's server. The Server project will be initiated first, which will set up a server on User1's computer and begin searching for a signal that points to the server's IP address and port number. Once the server is setup and searching for possible clients, User2 will open the Client project and provide the IP address and port number that User1's server is transmitting (similar to the PuTTY setup). Once the server recognizes and accepts the client, the two users are now connected and can send encrypted messages. The Server project has classes, MyServer and MyServerTest, and the Client project has classes, MyClient and MyClientTest. MyServer runs and monitors the server and encrypts all messages being sent by User1, while MyClient runs and monitors the client and encrypts all messages being sent by User2. The "test" classes simply executed their respective classes.

In other words, one end has to setup the server by simply running the program, and I believe that this part of the project works. Currently, I am working on the client portion of the project and have almost completed the MyClient class. I have to get the class to connect to the server given the IP address and port number, and then the instant messaging should be established. I also have to create "encrypt" and "decrypt" methods for each class that will encrypt messaging being sent and decrypt messages being received. I will have to research the best encryption methods and implement a strategy into my program.

In order for a user to properly use this system, he or she must follow these simple protocols. First one user (User1) must have the “Server” package on their computer and the other user (User2) must have the “Client” package on their computer. User1 will execute the “Server” application first, because if User2 initiated the “Client” application first, the client would have nothing to connect to and be stalled. So once the server is set up, User2 will start their program and will have to input the IP address and port number that they wish to connect to. Once the client sends this information, the server will pick up this information and connect the two users. Now the two computers can communicate with each other with the safety of encrypted messages. When a user wants to leave, they must simply type “Exit” in the message box, or just close the window. It is a simple step-by-step process that allows users to start communicating as easily and simply as possible.

Thus, this system gives users the possibility of maintaining complete anonymity, for the most part. This project is not only an exciting and useful application, but also a learning process for me. I have never created a networking application before, but many days of research and reading have given me a good amount of knowledge to get started quickly. This system will be very promising and will hopefully be a tool that someone will find useful.

References/Bibliography

http://cs.lmu.edu/~ray/notes/javanetexamples/

http://searchexchange.techtarget.com/definition/Internet-Relay-Chat

https://www.tutorialspoint.com/java/java_networking.htm

MyServer
- chat : JTextArea - txtbox: JTextField - outputStream : ObjectOutputStream - inputStream : ObjectInputStream - server : ServerSocket - socket : Socket
+ MyServer() + start() + waitForConnection() + establishConnection() + continueProcessing() + end() + sendMessage(msg : String) + showMessage(msg : String) + canType(flag : boolean) + encrypt(message : String) + decrypt(message : String)

MyClient
- chat : JTextArea - txtbox: JTextField - outputStream : ObjectOutputStream - inputStream : ObjectInputStream - server : ServerSocket - socket : Socket - IP : String - message : String
+ MyClient() + start() + connect() + establishConnection() + continueProcessing() + end() + sendMessage(msg : String) + showMessage(msg : String) + canType(flag : boolean)


+ encrypt(message : String)
+ decrypt(message : String)