Department of Electrical and Computer Engineering

University of Massachusetts Dartmouth

ECE 369 Computer Networks

©Prof. Hong Liu

Project: Network Programming Framework

Project Progress Report

Group Coordinator: Marcel Vieira

Members: Robert Sheran, Andrew, AJ Gabriel, Darian Kavu

1. Understand Socket Programming: Three Experiments (Python & Wireshark)

2. Making Design Choice: Client/Server or Peer2Peer

Our group decided to make the choice to use Peer2Peer connectivity due to how it could simultaneously provide a reasonably stable connection for a chatting app as well as how it could be easily made secure using a series of public and private data encryption keys.

3. Integrating Socket Codes: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (another language) besides Python

Our group decided to use C++ as our second language because it is a common programing language and has plenty of socket support. Also, throughout the years the team has had plenty of exposure to it through other classes. It also allows for the creation of objects which may be useful in the creation of our applicaiton

4. Inventing Own App:

Problem Statement revision

Create a secure chatting app using Peer-to-Peer connection and encryption keys to ensure safe and reliable connections.

State of the Art: One Key Paper & your annotation

Technical Approach refined:

Our biggest challenges are going to be the creation of the peer-to-peer network and the encryption of messages. In order to create the peer-to-peer connection we are going to use 2 sockets one for receiving messages and one for sending messages and the processes to send and receive will be threaded as to allow them to run at the same time. If another host would like to be added another set of sockets will be created for this user as well. Each user will be connected to 2 other users and will ensure they have at least 1 different user in the network. All messages will then be communicated between connection and compared for accuracy and updated if needed.

As for the encryption we will simply use an encryption module to generate asymmetric keys which will then be used to encrypt messages before they are sent out and the resulting encrypted string will be encoded into bytes and sent to the users the current user is connected to.

Preliminary Results demonstration:

As of the time of this progress report, our group’s code is segmented into sections and has not been tested altogether but all sections are nearing completion except for the encryption which will be applied towards the end of the process, we just wanted to get the core functionality operational. As a result, the group has not generated a set of official preliminary results.

5. Individual Accountability:

Marcel: Assisted with design of most security related aspects as well as designing how connections would be passed from central server to hosts in a Peer2Peer connection. I also, began working on simplified Peer2Peer coded and some basic functionality operational. Also, assisted in documentation including proposal, scripts, and graph.

Robert: Assisted in the designing of the Peer2Peer connection as well as how it may interact with the security aspects of the code. Assisted in the writing and designing of course-related documents such as the proposal and demonstration scripts. Will Continue to assist in Peer2Peer related issues and further debugging in other areas as the project continues.

Andrew: looking into threging and setting up functions in python to facilitate it.

Darian: Developed a simple and user-friendly User interface for the program which will enable users to send and receive messages within a group. Assisted with the development of a function which enables messages to be received from the server and displayed on screens of all users.

AJ: Will work on app backend, such as temporary secure data storage, data aggregation, and temporary connection storage for the purposes of maintaining a persistent connection throughout a single, secure chat. All data should be wiped on chat completion, excepting for essential housekeeping functions and user-specified personalization (if that’s something we want to implement.) Data storage will consist of a temporary chat log, only insofar as the messages will be visible until manually deleted or the chat has concluded. Data will be passed to the front-end as needed and stored only if it is required. Possibly investigate systems to ensure no data fragments are left behind after deletion (common security hole when deleting data.) If data space is marked as empty without changing the contents, it can be retrieved with the appropriate software. Recommend manual overwriting of all data with garbage data or nulls upon completion.

6. Next Step:

Currently we have managed to successfully create the beginning of our GUI and have managed to connect 2 users with a peer-to-peer connection but, the connection is not threaded. The next steps are going to be to thread the peer-to-peer connection, and attempt to get in a 3rd user in the connection as well as work on controlling load on an individual user.

