Software Requirements Specification
TauNet Client Application v1.0

---

1. Introduction

1.1. Purpose
The purpose of this document is to outline the minimum features required of the TauNet client application (the application) that is to be built by Jacob Martin (the developer). Note that the application functions both as client and server, but is referred to as a client application in this document for simplicity's sake. The application may implement additional features at the discretion of the developer so long as these minimum requirements are met.

1.2. Product Overview
The application is intended to allow its single user (the user) to be able to send messages to and receive messages from other client applications while fully implementing and adhering to the referenced TauNet Communications Protocol (the protocol).

---

2. References
TauNet Communications Protocol v0.2. CS 300 Section 1 Fall 2015.
https://docs.google.com/document/d/1juKX1KE8FnpVBpb9S6RHwLm2DpCJv0RnuKHOfOK-h7Y/edit

---

3. Hardware and Environment Requirements

3.1. Intended Hardware
The application shall run on the Raspberry Pi 2 Model B computer system (the hardware). Application behavior is undefined on other systems.

3.2. Operating System
The hardware shall be outfitted with the Raspbian Jessie Linux distribution based on Debian Jessie. The application is to be installed on this operating system.

3.3. Dependencies
The application shall depend externally only upon software packages included in the Debian source lists accessible through the installed apt repositories and approved of by the customer. Any external dependencies not available through the source lists shall be included with the final application so as to not require the end user to retrieve them.

---

4. Application Requirements

4.1. Installation of Client Table
The application shall enable a single client table to be installed. This client table shall have been distributed according to (and fulfill all requirements specified by) the protocol.

4.2. Uninstallation of Client Table
If a client table has been installed, there shall be a method for uninstalling said client table.

4.3. Interface
The application shall have an interface that will be implemented to allow the user to send and receive messages while the application is running.

4.4. Sending of Messages
The user of the application shall be able to attempt to send a message to any of the clients on the installed client table. The user shall, through the interface, select or specify the username of the intended recipient and then type the message to send. The interface shall inform the user in some way if the user's message exceeds the maximum length specified by the protocol or otherwise limit the user's input to that maximum length.

The application shall encrypt the message according to the protocol and attempt to connect to the intended recipient according to the protocol. If the connection succeeds, the encrypted message shall be transferred via the connection to the intended recipient. If the connection fails, the user shall be notified of the failure through the interface.

4.5. Receiving of Messages
As long as the application is running, the application shall listen on the port specified in the protocol for incoming messages. Once an incoming message is received, the application shall decrypt the message according to the protocol. When the incoming message is decrypted, the application shall make the user aware of the message's arrival through the interface.

5. Usability Requirements

5.1. Accessibility
The application will be able to be ran while the user is accessing the hardware through a monitor, keyboard, and mouse. Remote accessibility is not defined as a requirement at this point.

5.2. Readability
Messages and notifications shall be displayed in a clearly readable manner.

5.3. Resources
The application is not to consume excessive hardware system resources as compared to similar industry-grade messaging applications. In other words, the footprint of the application shall be considered normal in the eyes of a reasonable software engineer.

---

6. Use Cases

6.1. Sender Use Case
Before initiating this use case, the user has already installed a client table.

1. The user launches or gives focus to the application and chooses to send a message through the interface.
2. The interface presents a list of potential recipients from the list of usernames populating the installed client table.
3. The user selects or specifies a single recipient from the presented list.
4. The interface allows the user to type in a message to the single recipient.
5. The user types in a message.
6. The system attempts to send the message to the recipient adhering to the protocol and notifies the user via the interface if the attempt fails.

6.2. Recipient Use Case
Before initiating this use case, the user has already installed a client table and launched the application.

1. The user continues to allow the application to run.
2. While the system is listening, an incoming message is received and parsed according to the protocol.
3. The user is notified via the interface of the received message's sender username, the time the message was sent, and the message contents.

7. Verification

7.1. Sending Message Failure Notification

1. Assert that one of the clients on the client table is offline or nonexistent.
2. Send a message to that offline/nonexistent user.
3. Assert that the interface displays a notification that the message failed to be delivered.

7.2. Sending/Receiving Messages

1. Orchestrate having two clients (client A and B) on the same TauNet that can be simultaneously monitored.
2. Have client A send a message to client B.
3. Assert that the message displays properly on client B's interface.
4. Have client B send a message to client A.
5. Assert that the message displays properly on client A's interface.