Software Requirements Specification

Version 1.0 <>

November 1, 2015

CS 300 Students

Submitted in partial fulfillment of the requirements of

CS 300 Software Engineering

Table of Contents

1.0. Introduction

1.1. Purpose

The purpose of the this document to present a detailed description of TauNet; a pi to pi private and encrypted communications network. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, and the constraints under which it will operate.

As more comes to light on project specifics this document may be amended to comply with any changes that are made.

1.2. Project Scope

The TauNet Communications system will be a medium of private and secure communication for people with raspberry pi's who are registered with the system. Access to TauNet will be by invitation only. TauNet will run on specific hardware, namely the Raspberry Pi 2b. And users who wish to join the network must acquire their own and download the appropriate software application.

2.0. Overall Description

2.1. System Environment

TauNet is a solution to creating a secure and pre-defined network of communication nodes for a private group of people that can piggy-back on the internet and be used for secure communication. Security will be accomplished by encrypting outgoing messages and decrypting incoming messages using a single encryption key and method that only users will have access to. In it's initial implementation, TauNet will be capable of communication between any two users with access to it's network.

3.0. Functional Requirements Specification

This section is a general outline of how the system will function.

3.1. The TauNet network

Two or more users must be on the same network for TauNet to work. From this point on network users will be referred to as members. Each member will have a direct link to other members of the network for one on one communication.

3.2. Joining TauNet

Joining is by invitation only. The invitee will be given the network's encryption key and the list of other members of the network and will store them on their raspberry pi.

3.3. Communicating on TauNet

Communication will be between two members of the network. Both members must have their Pi's turned on and be running the program to send and receive messages. Communication will be initiated by typing the name or handle of the person being contacted followed by a message. Sender hits return to send and waits for reply.

3.4. Use Cases

Case: Joining TauNet

1. Acquire Raspberry Pi
2. Download TauNet Program
3. Upload list of current users, IP addresses, and the encryption key to the TauNet program
4. Start messaging.

Case: Sending a message

1. Enter Senders TauNet user name
2. Enter TauNet user name of recipient
3. Type message
4. If recipient is running TauNet message is received and he can respond

Header:

from: Senders TauNet user name

to: Recipients TauNet user name

<Message Body>

Case: Receiving a Message

1. Have TauNet running on Raspberry Pi
2. Program will listen on specified ssh channel
3. If message is received it will print on users screen.

Case: Replying

1. To reply user will enter their TauNet username

2. User enters the recipients TauNet username

3. User enters message and hits enter

4. If message is not sent, ‘not sent’ message will appear

4.0. TauNet Protocol

4.1. Description

The TauNet protocol is a layered atop TCP/IP for the exchange of messages between TauNet nodes. A plaintext message shall be marked with header information: the header and message body shall be encrypted using RC4 encryption, and the resulting cipher-text sent to its destination via an ephemeral TCP connection.

4.2. Encryption

Messages shall be encrypted with RC4 encryption. The general encryption approach is as described in the CipherSaber document. For each message, a 16-byte IV will be appended to the TauNet key to create the message key. The IV will be sent as plaintext as the first 16 bytes of the message stream. The IV should vary from message to message. Recommended ways to do this include hardware random generation or the use of a node-internal RC4 stream keyed with a non-shared key to generate IVs.

Once keyed, 200 iterations of the RC4 generator shall be performed and outputs discarded before using the RC4 key-stream. Beyond this, each successive key-stream byte shall be xor-ed with the next byte of the message plaintext to create a cipher-text byte; this byte shall be transmitted via TCP to the destination.

4.3. Message Format

Each TauNet message shall begin with a header section. The end of the header section will be marked by a blank line. Each header will consist of an identifying keyword, a colon, a single space, and the header payload. There are several standard headers, which shall all appear in the order given:

5.0. References

1. Software Requirements Specification, Version 1.0, April 2014

2. Interview with customer, Professor Bart Massey

3. TauNet Protocol, Version 0.1 - Professor Bart Massey CS300