Taunet

Software Design Document

Brodie Herrick

1-Dec-2015

Table of Contents

1. Introduction

1.1 Purpose

This software design document describes the architecture and system design of Taunet. This document is aimed at developers who may add to or refine the current Taunet implementation.

1.2 Scope

Taunet is a secure communication program utilizing an implementation of ciphersaber-2. The goal of Taunet is to transmit all communication securely, in a method that prevents a third-party from reading an intercepted communication.

1.3 Overview

Taunet is a solo project for a CS300 Software Engineering course at Portland State University.

2. System Overview

While Taunet is only aimed at working with the Raspberry Pi computer system, it's current implementation is as cross-platform as the Python programming language. The design is rather straightforward, using a class structure to make a singly-linked list of nodes that the user enters upon starting the program.

3. System Architecture

3.1 Architectural Design

The client program has a few methods, which all serve to send out messages. The client method initializes the classes, variables, and prepares to open sockets to communicate with other nodes. The menu method is pretty self-explanatory. The send_msg method takes the host given by the user's input and opens a connection to that node, allowing for communication. The encipher, rc4, and b2a methods all work together to encode the plaintext message into a ciphertext, which is then passed back to the send_msg method to be… well, sent.

The server program is a bit simpler, as it lacks any need for data structures. Upon running, it asks the user for the network passphrase, sets itself up to listen on the specified port, and then listens for messages to be sent in. Once it receives a message, the message is passed through the decipher, rc4, and a2b methods, which decode it. Once decoded, it is displayed in the terminal.

4. Data Design

4.1 Data Description

Each time a user elects to add a new node, a new object is created that contains the node's owner's name and it's internet-reachable address. The user can also elect to delete nodes from their network at will.

4.2 Data Dictionary

node: contains the name and address of the taunet node
slist: a linked list, and functions related to it, of taunet nodes.

5. Human Interface Design

5.1 Overview of Human Interface

The system starts with asking for the user's name and their network key. Once that data has been gathered, it displays a menu that the user can utilize to navigate the program. Each option has been simplified as much as possible.

6. Requirements Matrix

Chat Program	\
Secure	RC4 encrypted chat program
RC4 Encrypted	/
No Single Point failure	Decentralized system
at least 300 char messages	Current system allows for 500+
Synchronous	Effectively instantaneous
Headers	version, from, and to data in header
Support for at least 12 nodes	Can support as many nodes as the user inputs
Be expansible	User can add nodes at will
Be able to remove compromised units	User can remove nodes at will