Project Evaluation Report
TauNet Client Application v1.0

---

1. Purpose

The purpose of this document is to report on the status of the TauNet application at the point of its version 1.0 release, to describe what the process of building the application was like, and to make notes for the future. Version 1.0 is the point where the application is being tagged and delivered to the customer.

---

2. The Process

Because the design was well thought-out and documented ahead of time, the process of actually implementing the application was very smooth and followed the design exactly. If the design had made assumptions or been too specific we could have easily painted ourselves into a corner, but we managed to avoid that.

Using the English description of the CipherSaber-2 algorithm in addition to using Bart's pseudocode made the process of implementing it in C very straight-forward. The code turned out to be extremely efficient, though it does have the downside of loading the entire input files into memory before beginning to cipher them (I did this because it was simplest and our messages do currently have a protocol-defined size limit so it is not currently an issue). This could be rewritten to cipher and output, say, 1024 bytes at a time in order to allow the encrypting and decrypting of much larger files and messages without using more memory than necessary.

---

3. Self Tests Outcomes - Success

Self tests involve any tests that can be performed on a single Raspberry Pi by communicating with itself via the application's receive and send commands.

3.1. Installation - Success

The installation process works as documented.

```
leng@cherub:~$ git clone git@github.com:PSU-CS-300-Fall2015/Martin_Jacob_TauNet.git
Cloning into 'Martin_Jacob_TauNet'...
remote: Counting objects: 233, done.
remote: Compressing objects: 100% (30/30), done.
remote: Total 233 (delta 10), reused 0 (delta 0), pack-reused 203
Receiving objects: 100% (233/233), 524.77 KiB | 0 bytes/s, done.
Resolving deltas: 100% (103/103), done.
Checking connectivity... done.
leng@cherub:~$ cd Martin_Jacob_TauNet/
leng@cherub:~/Martin_Jacob_TauNet$ gcc cs2.c -o cs2
leng@cherub:~/Martin_Jacob_TauNet$ cp clients.json.example clients.json
leng@cherub:~/Martin_Jacob_TauNet$ vim clients.json
leng@cherub:~/Martin_Jacob_TauNet$
```

3.2. Missing Client Table - Success

The error message displays appropriately in both commands.

```
leng@cherub:~/Martin_Jacob_TauNet$ mv clients.json somewhere-else.json
leng@cherub:~/Martin_Jacob_TauNet$ ./send.py
The client table file is missing.
leng@cherub:~/Martin_Jacob_TauNet$ ./receive.py
The client table file is missing.
leng@cherub:~/Martin_Jacob_TauNet$
```

3.3. Sending and Receiving Messages - Success

The application is able to successfully send and receive messages correctly while using the commands on the same machine.

```
leng@cherub:~/Martin_Jacob_TauNet$ ./send.py leng "Testing: one, two, three."
leng@cherub:~/Martin_Jacob_TauNet$ ./send.py leng "This is a multiple
> line
> message."
leng@cherub:~/Martin_Jacob_TauNet$
```

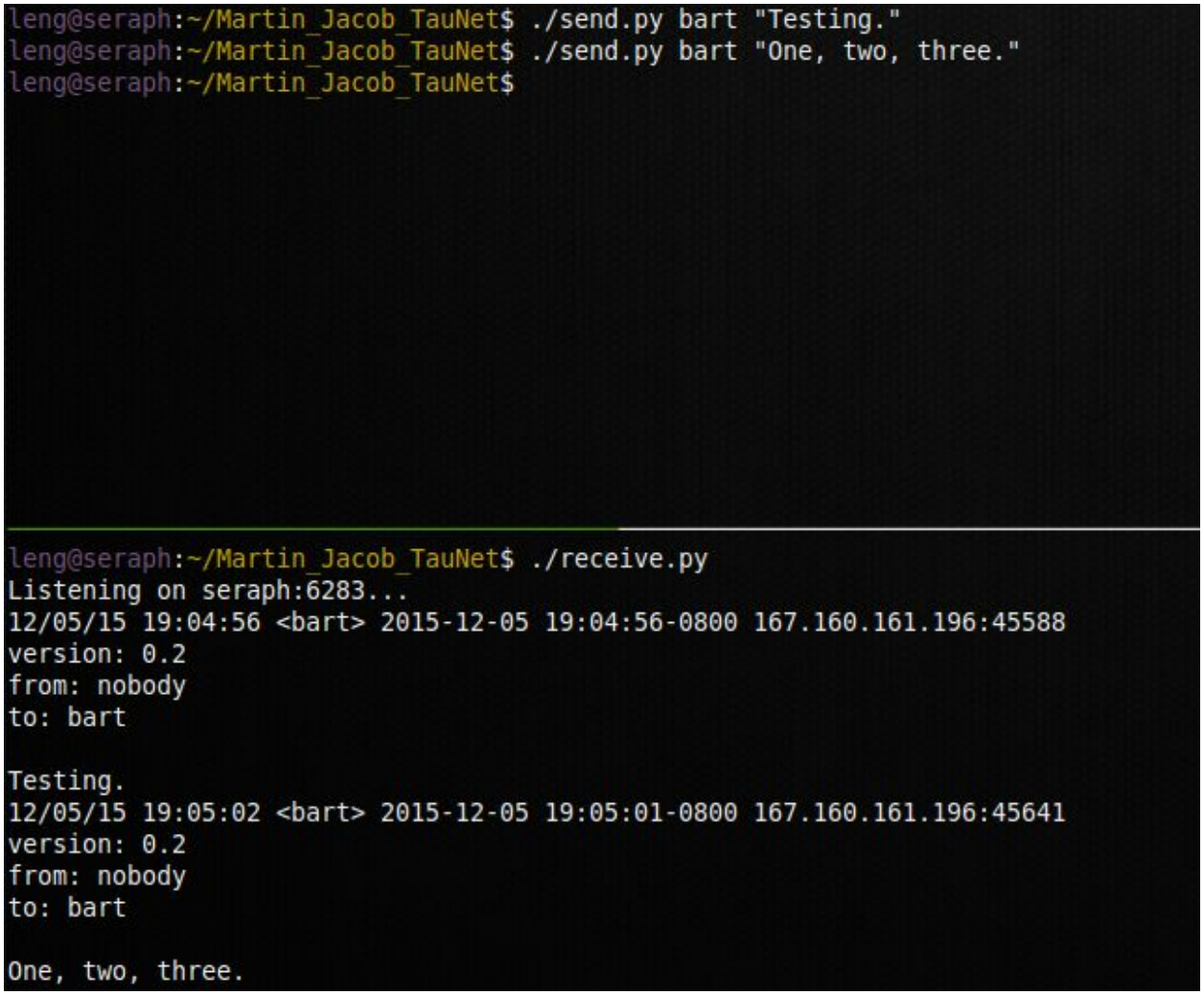
```
leng@cherub:~/Martin_Jacob_TauNet$ ./receive.py
Listening on cherub:6283...
12/05/15 16:49:54 <leng> Testing: one, two, three.
12/05/15 16:50:07 <leng> This is a multiple
line
message.
```

4. Community Tests Outcomes - Success

These tests involve connecting to and communicating with external TauNet clients.

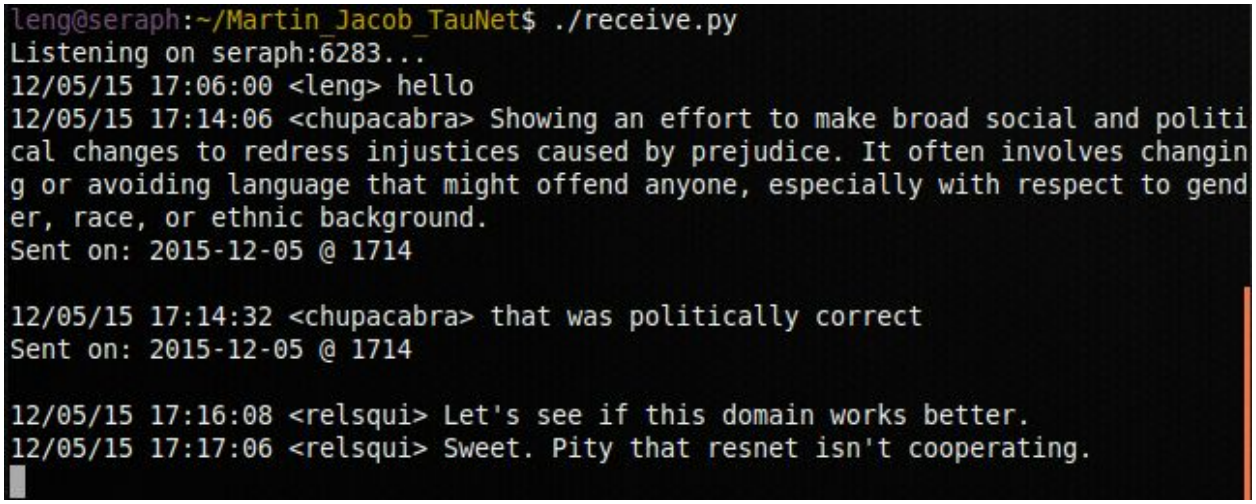
4.1. Sending Messages

Tested this using Bart's echo server; all went according to plan (received responses, as well).



4.2. Receiving Messages

Received, decrypted, and decoded several messages from users on the TauNet.



---

5. Future Concerns

The application may become a lot more user-friendly if it is implemented with an interface that leverages the **GNU ncurses** package in order to combine the sending and receiving of messages into one simple, intuitive interface. This way, terminal multiplexing (or starting multiple terminals) would no longer be required in order to send and receive messages simultaneously. More attention could be paid to the interface in this regard to make using the application more enjoyable.

The application also does not currently perform any kind of logging or storing of outgoing messages whose delivery is failed for any reason. In addition to this, because the send command's message is passed through a command-line argument, the messages are stored in the bash history of the Linux user using the TauNet application. This could potentially be a security issue that can be addressed in a later version if necessary.