This is my submission for the final evaluation of the CONIKS for Tor Messenger project which I started with my awesome mentors Marcela S. Melara from Princeton University, Arlo Breault from The Tor Project, and Ismail Khoffi from EPFL. They always had a smile on their face and gave me many valuable comments/feedback. It has been my greatest journey. Everything related to GSoC was so awesome. I'm really glad I got such supportive mentors. Thanks a lot to The Tor Project and GSoC!

1. **Project description:**

Tor Messenger is a cross-platform chat program that aims to be secure by default and sends all of its traffic over Tor. CONIKS is an end-user key management and verification system for end-to-end secure communication services, which improves upon existing key management systems by providing both strong security and better usability using a model called key transparency.

This project consists of two parts: a CONIKS key server for key management and a CONIKS client plugin, which performs the key verifications, for the Tor Messenger client. The functionalities include key registrations, key changes, lookups and monitoring.

2. **Achievement:**

We have a prototype server running that accepts the registrations and enables the client to perform the protocol.

3. **Project commits:**

**CONIKS-Go repo:**

Commits already merged into master:
https://github.com/coniks-sys/coniks-go/commits?author=c633

Pull requests which are under reviewing:
https://github.com/coniks-sys/coniks-go/pulls/c633

**ctypes-otr repo:**

Commits already merged into master:
https://github.com/arlolra/ctypes-otr/commits/master?author=c633

Pull requests which are under reviewing:
https://github.com/arlolra/ctypes-otr/pulls/c633

4. **Work done:**

- Implemented a CONIKS library in Go which includes the core CONIKS protocol implementation, a Merkle prefix tree implementation, an implementation of CONIKS' crypto primitives, cgo library for the client, as well as a reference key server with registration, key lookup and monitoring functions.
    - The Merkle prefix tree is one of the most important components of the CONIKS protocol. We implemented this data structure separately as a library to help other developers use it in their implementation easily.
    - Currently, the format of messages exchanged by CONIKS clients and servers is JSON.
    - The connection between the key server and clients is secured using TLS.
    - PRs: #29, #71 & #72 (key server implementation); #64 (protocol implementation); #1 & #8 (Merkle prefix tree library)
- Implemented the CONIKS preferences panel for the ctypes-otr addon (which is an existing OTR add-on for Tor Messenger).
    - The CONIKS preferences panel in the ctypes-otr addon is written using XUL technology. I also tried to use new ES6 syntaxes in my implementation.
    - PR: #82
- Started integrating the cgo library (libconiks) into the ctypes-otr addon.
    - The client library was compiled to a shared library using cgo. This library is then wrapped using the js-ctypes, so that it can be called from the add-on. Unfortunately, the client library currently works on Linux and macOS only . The approach for this client library is as follows: it takes the raw server response message that is a JSON string. Then it will do all unmarshalling and verifications and return the error code to the addon. The add-on (Javascript part) now just

sends & receives the messages, and calls the library functions and writes down the verified result into files.
    - PRs: #39, #83, #84
- Designed a novel registration protocol, which would be used by third party of CONIKS (who does not provide the communication services, e.g., Tor Messenger). The detailed documentation of this protocol is available here.
- Implemented the registration bots which is an specific implementation of the account verification protocol for Twitter.
    - The bot and key server would be run in the same machine, and the key server accepts the registration requests from the bot only. Thus, we have established an Unix socket between the bot and the key server.
    - PR: #33

**5. Work left:**

Much work remains:

- The client registration function is under review.
- The client key lookups and monitorings are under development.
- Documentation for the project is incomplete. However, we have many discussions on the Github issues/pull requests, which can be a part of the documentation.

There are also many open issues that need to be solved before this project can go to the production stage (coniks-go repo & ctypes-otr repo).

**6. Future work:**

I will absolutely keep working on this project after the end of GSoC. We plan to have a release before the end of August, and this release will be published on CONIKS's mailing list (https://coniks.cs.princeton.edu/subscribe.html). We would like to receive feedback from the community on our implementation as well as the current open issues.