ChaCha20 Private Messenger

Software Project Management Plan

Dominic Oertel
Towson University

April 27, 2021

Welcome to the SPMP of the CC20 Private Messenger. The purpose of this document will be to walk the reader through the various elements of the software development process, organizational attributes, and offer a general preview of the finished product.

This software is being developed with the understanding that we are living in a world where surveillance has increasingly become the norm of everyday life and true privacy is a dying concept. Digital technology has afforded us many benefits since the beginning of the information age, and it is the view of some, including the developer of this project, that as citizens of a free republic we have a fundamental right to confidentiality, integrity, and accessibility in our technology.

Tables of contents

Executive Summary

The ChaCha20 Private Messenger will not in itself be a messenger application. Instead, it is designed to be used in tandem with Discord, which handles all message transfers between user devices. Should they so choose, users may use this software to add extra layers of security to their discord messages in a quick and reliable manner, making it very difficult for Discord and other 3rd parties to be able to snoop in on their data. Intuitively, this software is also required on the receiving end of messages as the only way to decrypt user messages in a reasonable amount of time using a key shared only between users.

The user will be walked through each step of setting up this software, making it very simple to use while offering some insight into basic cryptographic concepts. Prerequisites include a computer complete with Python 3, its standard libraries, as well as one dependency: A library by the name of PyCryptodome, which will supply the backend program with the ChaCha20 encryption scheme. Additionally, users who indent to use this tool to its full extent will need the Discord client, NodeJS, and a text editor installed on their machine.

Deliverables & Milestones

Project Diagrams – 2/31/2021

SPMP – 3/16/2021

Back-end Translator and Key Establishment program – 3/23/2021

Message transfer code and protocols – 4/13/2021

Final Testing and Submission – 4/27/2021

Glossary

Developer – The author of this SPMP and project, Dominic Oertel

Supervisor – The professor who mandated this project development, Rebecca Broadwater

CC20 PM – ChaCha20 Private Messenger, the subject of this report

Security – the inability of third parties to intrude on user data

En- Decryption – translation of readable messages into almost unbreakable code and back

ChaCha20 – Encryption algorithm used by Secure Messenger Application

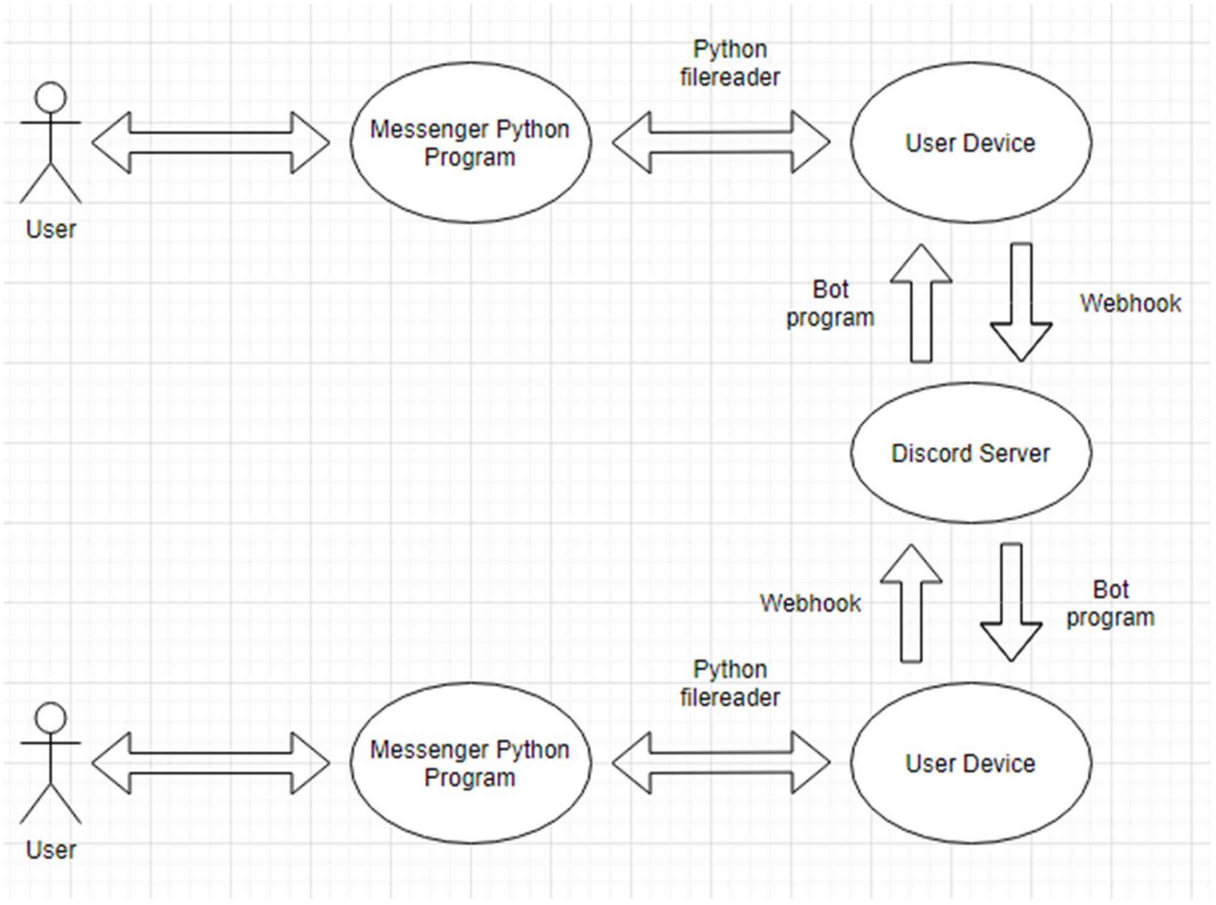
SPMP – Software Project Management Plan

API – Application Programming Interface, used to carry out application functions via source code

Bot – an automated program capable of carrying out specialized tasks for the user

<u>Project Organization</u>

The project contains two main programs. First, the encryption/decryption Python code, which will serve the software's primary purpose of taking user input, performing numerous complex bit-by-bit operations on it, and returning the corresponding output. This program will also be responsible for key establishment, an advanced cryptographical procedure using a well-known process called the Diffie Helman Protocol. In other words, this first program will be the one performing all mathematical operations as well as ensuring that data generated for the end user will be secure and safe to send over insecure channels.



Second, to implement this translation program, Discord's flexible API will be used to transfer messages between users quickly without needing to waste time directly interact with the client itself. Instead, users are strongly recommended to use a predesigned bot program included in the project repository to automatically receive and download messages from their correspondent, giving the Python code instant access to decrypt its contents. Furthermore, included in the Python code will be a webhook, also meant to use the Discord API. Messages encrypted by CC20PM will automatically be sent over whichever Discord server is designated by the user.

All responsibilities for this project will be handled by the sole developer. Responsibilities are comprised of designing the program, determining its organizational layout, compiling adequate documentation and diagrams, completing source code for the front- and backend Python code, debugging to ensure the software operates as outlined, and most importantly, the developer will be responsible for ensuring total and inalienable privacy for the end user of the CC20 Messenger. Absolutely no data will be collected from users of the end-product, as that would defeat the software's purpose.

Managerial Process

The philosophy going into this project was to emphasize convenience for the end user as much as possible. This means that users must be able to install it and its dependencies quickly, and that they are able to quickly grasp its intuitive design. Naturally, functionality is the top objective of development. The software must function properly as advertised with minimal to no bugs or interrupts. After functionality, convenience for the end user as described is the next highest objective.

This project's development is based on the assumption that there is demand for truly private and secure communication between end users. Although far less convenient than using secure messengers as they are, this application seeks to put the means of unquestionable security and confidentiality back into the hands of the consumer. It does this by walking users through each step of the process of key establishment, which is the make-or-break aspect of confidentiality. When using messengers such as SMS, WhatsApp, Snapchat, etc., even under the assumption that the companies in control of these applications are not secretly collecting and storing messages from their users, there is still the problem that the keys they are using are not in the hands of the user alone. Despite their promises, corporations are responsible for user rights and privacy, which have been infringed upon on numerous occasions in the past. The consumer must be the only one with access to their private decryption key to be absolutely certain that there is no chance of it being misused.

Risks of note include the following: bugs, Discord outage or API changing, and user irresponsibility. First and foremost, there is always a chance that bugs may slip through testing and may end up in the final product. In the event of this happening, the problems must be made clear to the developer as soon as possible. Depending on the nature of the glitch that occurred, it may be fixed in anywhere from a couple minutes to a few days. In any case, problems with the software will be handled immediately if they are discovered and made known.

Furthermore, in the unlikely event of Discord servers going down or the API changing to make this software incompatible, changes will be made to handle whichever scenario occurs. Should Discord change so thoroughly that this messenger will no longer work, the developer will ensure that the necessary changes are made to keep the software up to date and working perfectly. Unfortunately, in the event of a total outage, there is little to be done except wait for Discord's staff to handle the issue.

Finally, user irresponsibility is the most complex problem to solve and another one out of the hands of the developer. This software includes a built-in default key for the user to use if they wish, however, it is not recommended as the cryptanalysts generally agree a different key should be used for each session of use.

Technical Process

This project will be programmed using the Python 3 programming language and JavaScript for all source code. Almost all functions and modules within the Python program will rely on its standard libraries. The one exception will be the encryption scheme library, pycryptodome. ChaCha20 is a module of that library, and it will be the one used to encrypt and decrypt user messages. Key establishment will be done using standard python mathematical functions and message transfer on the sending end will also be handles by Python webhooks. The task of receiving and downloading messages will primarily be handled by the Discord bot, programmed using Discord.js, Discord's official JavaScript API module.

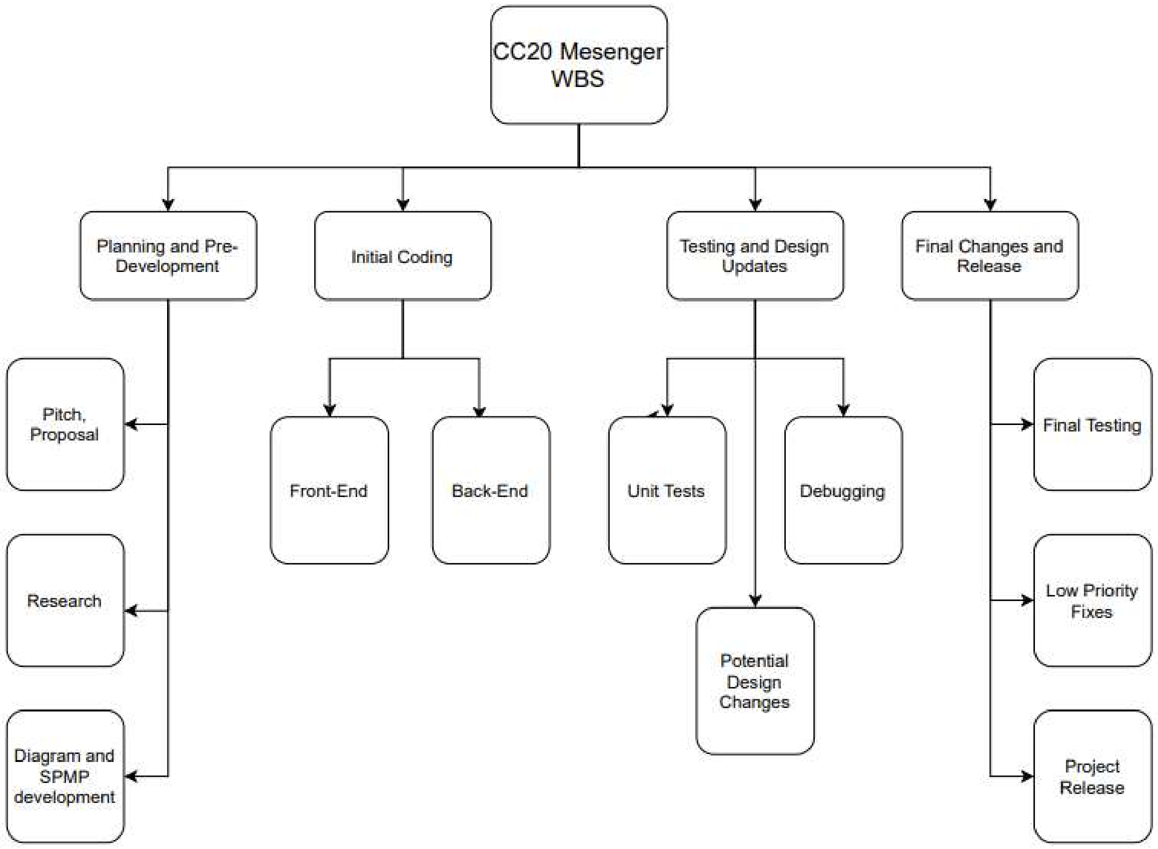
Planning and research of the CC20 PM development process began in early March of 2021. It was decided early on that the end product would be an encryption – decryption program that is open for anyone to use and offers accessibility to those not familiar with more complex encryption programs, which often need knowledge of the command line to use. Intuitively, to offer the greatest ease of access to users, it was decided shortly after that some method would be included to make transfer of encoded messages fast and reliable. It took several weeks of research and initial testing to determine whether Python 3 and pycryptodome would be used for this project, or Java 8 and java.security encryption. In the end, Python won out because it offers more powerful and simplified encryption schemes, being a newer language with the same cross-platform aspects that Java has boasted for the past 25 years.

It was decided that the Messenger would not transfer messages between end users on its own. Such a software would have required skills and time outside the scope of this semester. Instead, Discord was chosen as a developer-friendly and convenient method to send messages. The hardest part was finding a way to automatically up- and download the encrypted messages, but eventually it was discovered that webhooks can send messages very easily through Python and a bot can signal to the Python program what messages to download to decrypt. The creation of the bot is the most complicated part for the end user but a detailed tutorial on setup is included in the repository.

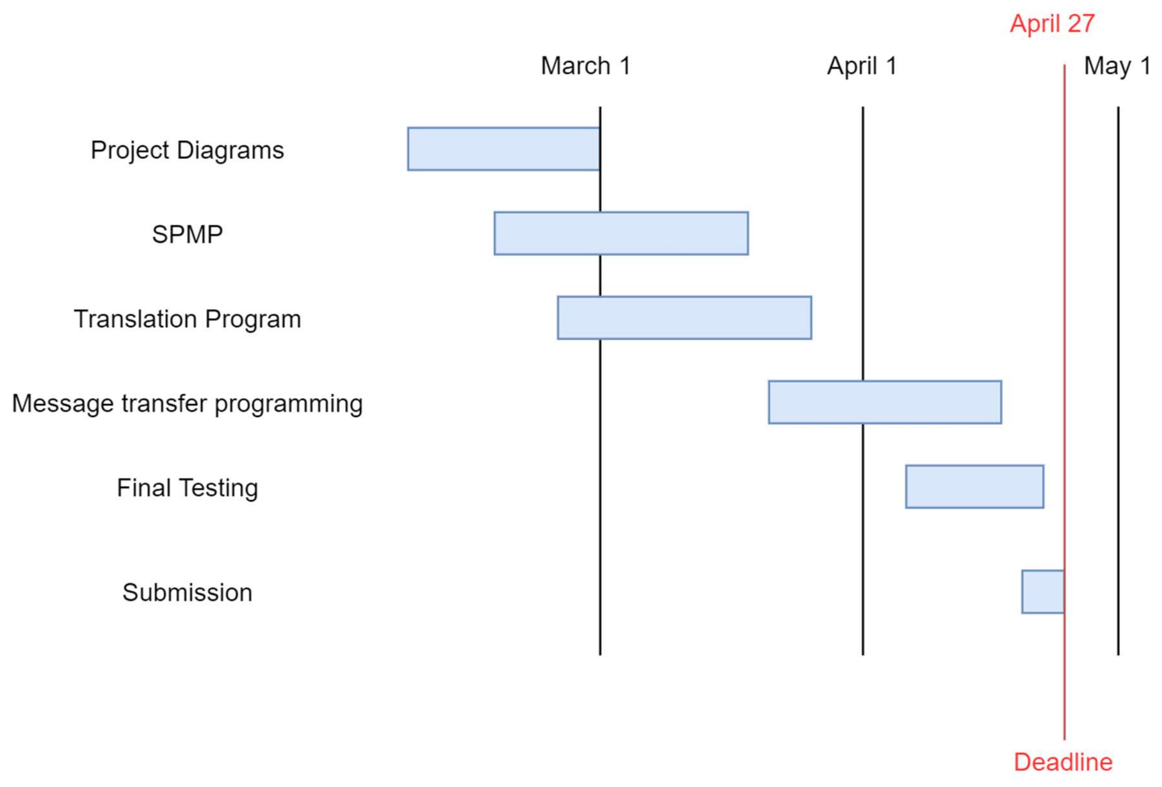
The first coding for this project took place near the end of March. It was decided that back-end programming was first and foremost the top priority and should thus be completed first. This entailed creating the CC20 program, one which would en- and decrypt user messages with sufficient security so they may not be broken within a short amount of time by snooping 3rd parties, but simple enough to be translated instantaneously. After this program was created and thoroughly tested, incorporation of the Discord API began in early April, which led to the inclusion of JavaScript in the necessary code.

All Python programming is being done using the JetBrains PyCharm Integrated Development Environment. JetBrains offers all students enrolled in university a license to their software packages free of charge. PyCharm offers an extremely ergonomic experience, providing users with code analysis, a graphical debugger, an integrated unit tester, integration with version control systems, and supported web development via Django and data science via Anaconda. All JavaScript programming and interaction with GitHub was done using Microsoft Visual Studio Code. Diagrams for this project were created primarily using draw.io and documents such as this were made in Microsoft Office's Word 2019.

Work Elements



Schedule



Project Diagrams:	February 10 – March 1
SPMP:	February 18 – March 16
Translator Program:	February 25 – March 23
Message Transfer:	March 19 – April 13
Final Testing:	April 4 – April 25
Submission:	April 27