

Encryption and Decryption Tool

Jack Sprague, Ben Yaker, Michael Shteynberg

Computing Fundamentals – 2140



Table of contents

01

Introduction

02

Literature Review

03

Methodology

04

Analysis and Results

05

Discussion

06

Conclusion

07

References



Introduction

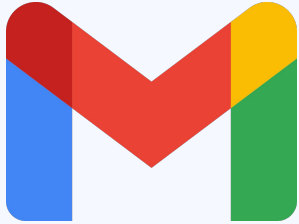
Objectives

- Create a quick and easy encryption tool for users to send messages and secrets
- Create a decryption tool that takes the encrypted messages and makes them readable for users

Project Scope

- Symmetric encryption that uses a single password for encryption and decryption

Literature Review



Gmail

Gmail uses TLS to encrypt emails while they're sent and keeps them secure when stored



NordVPN

NordVPN uses TLS to encrypt your internet connection and keeps your data secure while browsing



WhatsApp

WhatsApp uses end-to-end encryption with the Signal Protocol to keep messages and calls secure

How does our project relate?

- Our project also uses end to end encryption like some of those tools
- We use a single passkey which is symmetric encryption
- The same passkey is used for encryption and decryption
- We are different because the user enters their own passkey which is more secure and makes our project more trustworthy

Methodology

Methods and Techniques Used



Existing Libraries

Python cryptography library

- Used for the encryption function and key derivation function

Tkinter

- Used to create the GUI and enable user interaction



Python Modules

- Divide the code into parts
- Increase readability



Classes

- Used to organize code
- Consistent with the principles of OOP.

Cryptographic concepts

Encryption algorithm

- A symmetric encryption algorithm called Fernet was used.
- Encrypts text using a 32-byte key

Key Derivation Function

- Converts a plaintext password into a 32-byte key.
- Uses **hashing** - a one-way algorithm that converts text to a password hash.

Pseudocode

User Input

1. Initialize GUI with
2. Select encryption or decryption.
3. Input password
4. Input text to encrypt/decrypt

Key Derivation

Use KDF function from cryptography library to convert password to 32-byte key.

Encryption

Using the Fernet encryption algorithm, encrypt the text using the key.

OR

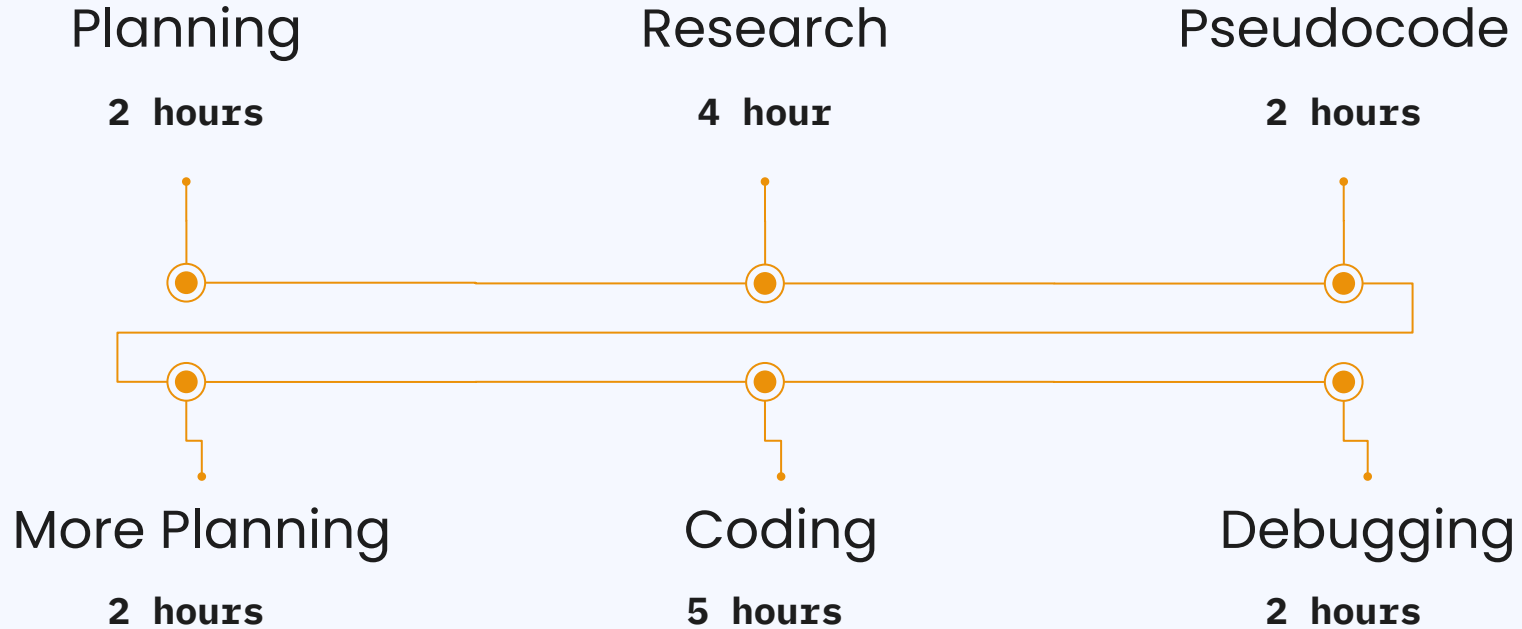
Output

Display result of the encryption/decryption for the user to copy.

Decryption

Uses the inverse Fernet algorithm to decrypt the token using the key.

Timeline



Key Findings



Symmetric

Single passkey encryption



Asymmetric

Public and Private passkey encryption



GUI

Graphical User Interface - environment for encryption



Fernet

Basic Symmetric Encryption algorithm



KDF

Passkey converter for encryption tool



Salt

Security tool for passkey hashing

Results

Secret123	Secret123	Python	Python1
HelloWorld	gAAAAABnPP4UhSDry OZPAqMFkkkOPd_IPe -J3YWImnH9vKXRjMT -ICSIBctaYPm5DCHfi mLoBa36LcqMe8iSzjl uKv7o_m7SA==	Coding is Fun	gAAAAABnPP-tZeUX8rDBG8 COWrXYL9xGvmp0NgFuz50I tFTivQL5E2vzg94fLqQLQG3X eC1Ap3HvrkKvCRIHQ2smvxr o6-cOFw==
gAAAAABnPP4UhSDry OZPAqMFkkkOPd_IPe -J3YWImnH9vKXRjMT -ICSIBctaYPm5DCHfi mLoBa36LcqMe8iSzjl uKv7o_m7SA==	HelloWorld	gAAAAABnPP-tZeUX8rD BG8COWrXYL9xGvmp0N gFuz50ItFTivQL5E2vzg94 fLqQLQG3XeC1Ap3HvrkK vCRIHQ2smvxro6-cOFw ==	Error!!!!!!!

Interpretation of Results

01 — Functionality — The tool successfully encrypts input text and decrypts it back to the original using the single passkey

02 — Accuracy — Decrypted outputs consistently match the original inputs, confirming the reliability of the encryption-decryption process

03 — Security Implications — The results demonstrate that the encryption is only as secure as the secrecy of the passkey

Discussion

Implications

• Security

Encryption algorithms are very secure and important on the internet.

• Freedom

End-to-end encryption ensures absolute freedom of communication.

Limitations

• Sym. Encryption

Sender and recipient must both know the password.

• Scalability

Only supports text.
Slower on larger files.

Project Conclusions

01



Importance of Encryption

The project reinforces the importance of encryption in protecting sensitive data

02



KDF and Salt

Using KDFs and Salt strengthens passkey security and prevents common vulnerabilities

03



GUIs Improve Usability

A user-friendly GUI makes tools more accessible for non-technical users.

04



Organized Code is Key:

Dividing code into classes improves readability, maintainability, and scalability.

Future Work

Asymmetric Encryption

Implement public-private key encryption to enhance security and enable secure sharing of encrypted data

Login Page

Develop a login system to manage user authentication and improve passkey security

Improved GUI

Redesign the interface to be more streamlined, visually appealing, and accessible for all users

File Encryption

Expand the tool to encrypt various file types, such as images and documents

References

- [Python Cryptography](#) - explains how to use cryptography tools like Fernet in Python
- [Cryptography Salt](#) - explains what salt and password hashing is
- [GUI](#) - explains what GUI is and how to implement it in Python
- [Tkinter](#) - explains what Tkinter does and how to use it in Python



"I love python"

– **Michael Shteynberg**
(someone famous)

