COMP.SEC.300-2021-2022-1 Secure Programming

Programming project document

Kokkonen Jesse

## General description

The program is a password manager. It is used to manage user's passwords in a way that he only needs to remember one password, so called master password. The user can generate new or save existing passwords with the program. The passwords are stored encrypted using 256-bit AES, with each key derived from the master password using scrypt with different salt for each key. The decryption happens in the same way, and the decrypted password is automatically copied to the clipboard for easy pasting.

The user interface is a CLI. The interface has options for the user to either retrieve a password, list all password titles, generate new password, add an existing password, update an existing password, delete an existing password, or quit the program.

On all platforms the program requires pycryptodome python package. On linux platforms, xsel is required for the automatic copying to clipboard to function. According to the documentation of pyperclip it should work out of the box on windows and mac.

## Structure of the program

The program is divided into 5 different modules. The modules are main, pw_handler, file_handler, cipher and utils.

The main module is responsible for running the user interface. I aimed to keep other functionality minimal in this module.

The pw_handler module contains all the functionality for interacting with passwords. These operations are generating new password, adding new passwords, and retrieving, updating or deleting passwords.

The file_handler module has operations for accessing files. This includes reading or writing to a file.

The cipher module contains all the encryption related functionalities. It can derive the key for encryption, encrypt and decrypt the password files.

The utils module has utilities. These include functions such as listing all the password titles saved and checking whether a file exists or not.

## Secure programming solutions

The biggest secure programming features the program deals with are key derivation, encryption and decryption. These issues have been solved with the pycryptodome library, which includes readymade functions for deriving keys and ciphers. These features are used in the cypher.py module in their respective functions.

## Security testing

The program deals with user input, and aware user could input his own file to the program. All user input is read using the default input function of python. Since the python version used is 3, input is not evaluated and it poses no risks, and I was unable to abuse this in any way.

An aware user could make a malicious file, using malicious.pw as an example here, and get it into the program when retrieving a password. Again, python3 does not execute anything read from a file, so it is safe to use.

The program can delete files too. This is where the file extension .pw comes in. The program always appends the extension to the file the user wishes to delete, so there is no way for the user to delete something else.

## Vulnerabilities

There is currently one vulnerability, and it is that padding for the passwords has not been implemented. This means that the length of the password can be read straight from the file. This reduces the amount of brute-force attempts required to crack the passwords.

## Improvements

- Implement the padding
- Graphical user interface
- Store the master password for more usability and less security, currently master password is asked at start, and if it is wrong, it is only found out after trying to decrypt the password (or if somehow luckily the decrypted bitstream is utf-8 encodeable, when user tries to input password to the service he is using)
- Make the application web-based, allows access to passwords on any device able to connect to internet