Hanoi University of Science and Technology

Project I Report
Offline Password Manager

Bui Hai Duong 20214953
Instructor: Tran Vinh Duc

January 2025

Table of Content

1. Overview

This project is a command-line based password manager that allows users to securely store, retrieve, and manage their passwords. The application uses encryption to protect stored passwords and provides various features to manage them efficiently.

2. Features

- Add/Update Passwords: Users can add new passwords or update existing ones for different websites.
- Retrieve Passwords: Users can look up stored passwords for specific websites.
- Delete Passwords: Users can delete passwords for specific websites.
- Delete All Passwords: Users can delete the entire database of stored passwords.
- Delete All Data: Users can delete all data including the master password and stored passwords.
- Generate Passwords: Users can generate complex passwords of a specified length.

3. Libraries Used

- sys: Provides access to some variables used or maintained by the interpreter and to functions that interact strongly with the interpreter.
- getpass: Provides a way to securely handle password prompts where the input is not echoed.
- pyperclip: A cross-platform clipboard module for copying and pasting text.
- termcolor: Provides ANSI color formatting for output in the terminal.
- json: Provides methods to parse JSON formatted data.
- os: Provides a way of using operating system dependent functionality like reading or writing to the file system.
- random: Implements pseudo-random number generators for various distributions.
- string: Provides a collection of string constants.
- Crypto.Cipher: Part of the PyCryptodome library, used for encryption and decryption.
- hashlib: Provides a common interface to many secure hash and message digest algorithms.

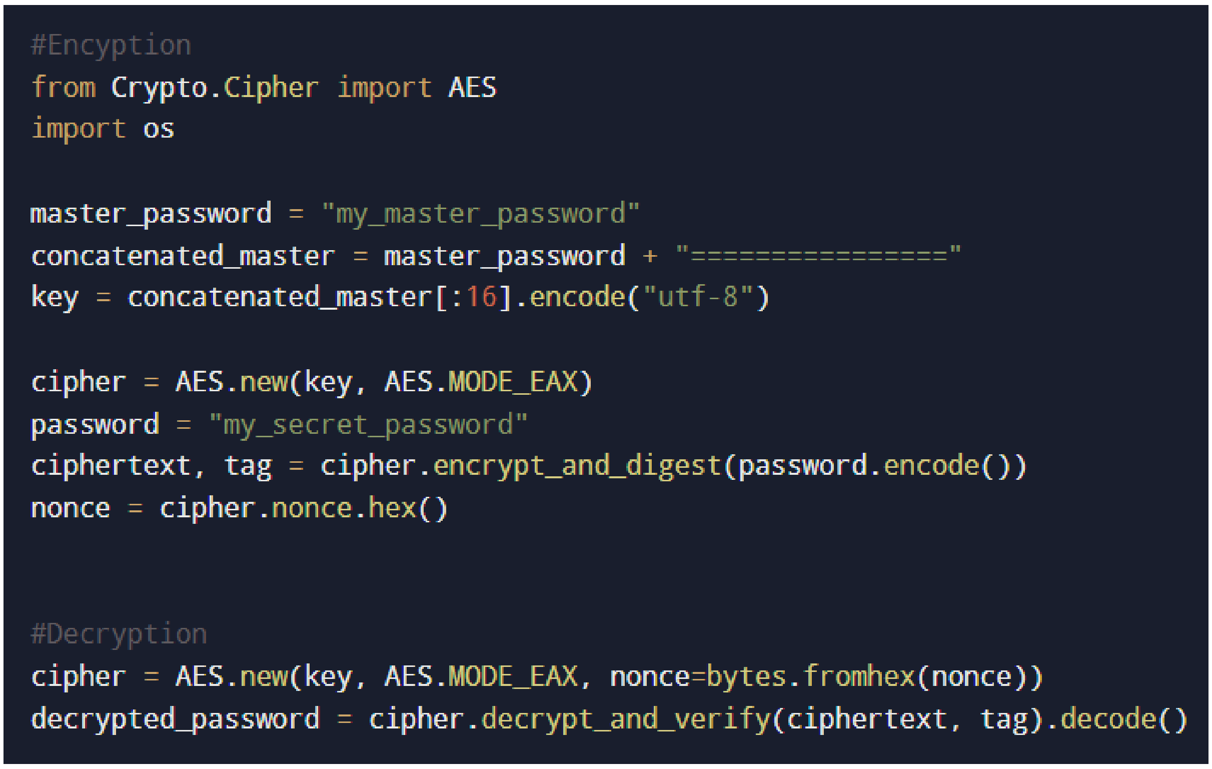
4. Encryption Mechanism

- The application uses the PyCryptodome library for encryption and decryption of passwords. The master password is used as a key to encrypt and decrypt the stored passwords. This ensures that even if the password file is accessed by an unauthorized user, the passwords remain secure.

- Key Derivation: The master password is concatenated with extra characters to ensure it is at least 16 characters long. The first 16 characters are then used as the encryption key. A nonce is generated and used in the encryption process.

- Encryption: When a password is added or updated, it is encrypted using the derived key and a symmetric encryption algorithm like AES. The encrypted password is then stored in the password file.

- Decryption: When a password is retrieved, the stored encrypted password is decrypted using the same derived key and symmetric encryption algorithm. The decrypted password is then displayed to the user.



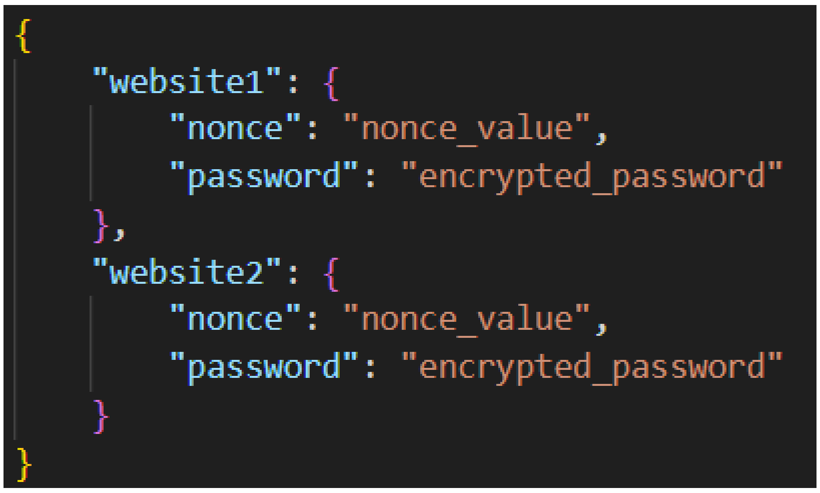
```python
#Encyption
from Crypto.Cipher import AES
import os

master_password = "my_master_password"
concatenated_master = master_password + "================"
key = concatenated_master[:16].encode("utf-8")

cipher = AES.new(key, AES.MODE_EAX)
password = "my_secret_password"
ciphertext, tag = cipher.encrypt_and_digest(password.encode())
nonce = cipher.nonce.hex()


#Decryption
cipher = AES.new(key, AES.MODE_EAX, nonce=bytes.fromhex(nonce))
decrypted_password = cipher.decrypt_and_verify(ciphertext, tag).decode()
```

- Passwords are stored in a JSON file in an encrypted format. Each entry in the JSON file contains the website, the encrypted password, and the nonce used during encryption. The structure of the JSON file is as follows:



```
{
    "website1": {
        "nonce": "nonce_value",
        "password": "encrypted_password"
    },
    "website2": {
        "nonce": "nonce_value",
        "password": "encrypted_password"
    }
}
```

5. Potential Vulnerabilities

- Weak Master Password: If the master password is weak, it can be easily guessed or brute-forced, compromising the security of all stored passwords.

- Local File Access: If an attacker gains access to the local file system, they can potentially access the encrypted password file and attempt to decrypt it.

- Clipboard Sniffing: When passwords are copied to the clipboard, other applications can potentially read the clipboard contents, leading to password exposure.

- Malware: If the user's system is infected with malware, it could potentially capture keystrokes or access the password manager's data.

6. Future Improvements

- Key Derivation Function (KDF): Implement a more secure key derivation function like PBKDF2 or Argon2 to derive the encryption key from the master password.

- Password Strength Meter: Add a feature to evaluate the strength of user-generated passwords and provide feedback.

- Two-Factor Authentication (2FA): Integrate two-factor authentication to add an extra layer of security.

- User Interface: Develop a graphical user interface (GUI) to make the application more user-friendly.

- Cloud Sync: Implement a feature to sync passwords with a cloud service for backup and access across multiple devices.