

Password Manager Program

Introduction

Nearly every page you visit insists you create a user account and think up a password, from dating apps to hyper-secure banking sites. The human memory can't keep up with dozens and dozens of these. Some users get the bright idea to use the simplest possible passwords, things that are easy to remember, like "12345678" or "password." Others memorize one superbly random password and use it for everything. Either path is likely to make you the latest victim of identity theft.

In order to avoid being prey to such attackers, a password manager is a candidate solution. With a password manager you don't have to remember that strong, unique password for every website. The password manager takes care of that, and can even help you come up with random passwords.

Problem Definition

In this project I will be designing and implementing a simple password manager with a random 10 character password generator using the cryptographic python library, Pycryptodome.

Existing Password Managers

The typical password manager installs as a browser plug-in to handle password capture and replay. When you log in to a secure site, it offers to save your credentials. When you return to that site, it offers to automatically fill in those credentials. If you've saved multiple logins for the same site, the password manager offers you multiple account login options. Most also offer a browser toolbar menu of saved logins, so you can go straight to a saved site and log in automatically.

Some products detect password-change events and offer to update the existing record. Some even record your credentials during the process of signing up for a new secure website. On the flip side, a password manager that *doesn't* include password capture and replay automation needs to offset that lack with significant other assets. Some of the existing password managers are listed below.

Dashlane

Dashlane can syncs across all Windows, macOS, Android, and iOS devices. It does offer all essential and advanced password management features. Includes VPN protection, Scans Dark Web for compromised accounts and can captures online shopping receipts. However, It is quite

expensive, especially if you already have a VPN. It also cannot choose VPN server country. It has no special handling for nonstandard logins and has limited support for Internet Explorer.

Keeper

Keeper supports all popular platforms and browsers and also supports the Two-factor authentication. It has secure password sharing and inheritance. It has an optional secure file storage and messaging feature while retaining full history of passwords and files. It however, has some limitations like Web form filling is somewhat limited and does not fully automate password updates.

Bitwarden

Bitwarden also like the previously mentioned supports all popular platforms and browsers, Two-factor authentication using Yubikey or FIDO. It can also Generate TOTP codes for 2FA-supporting sites. It also has the analyze passwords feature and somewhat inexpensive. It does have its limitations nonetheless. Edge extension does not work correctly. Its support for iOS is somewhat limited and Full-scale secure sharing costs extra.

LastPass

Lastpass is one of the most popular password manager because of its free version. It supports enhanced multifactor authentication choices and include a 1GB of secure online file storage. However, it doesn't add enough to what you can get for free and the premium is quite costly.

LogmeOnce

LogmeOnce syncs across Windows, macOS, Linux, iOS, and Android with an easy to use new, streamlined interface. It boasts of vast number of features, many of them unique and patented. Nonetheless it has some limitations. Some features cost extra. All-features installation quite expensive and the vast number of features may overwhelm users.

Password Boss

Password Boss like others can sync across Windows, macOS, iOS, and Android devices. It also support secure sharing and password inheritance, Two-factor authentication, security Dashboard and can auto-fill web forms. However, it does not have online access to stored passwords and some configuration settings could be more flexible.

Sticky Password

Sticky Password does all the basics of a password manager and syncs passwords across devices. It support secure no-cloud Wi-Fi sync and Two-factor authentication. It however, does not recover master password.

1Password

1password has apps for Windows, macOS, Android, and iOS. It has secure yet simple authentication when adding new devices. It also supports Two-factor authentication and has extensions for most browsers. All-platform Chrome extension but it is not compatible with Internet Explorer and has limited import options.

All the mentioned password manager above are only few of the existing ones. Some others include, Kaspersky Password Manager, Intuitive Password, F-Secure, Zoho Vault, True Key, RoboForm 8 Everywhere, Authentic8 Silo, password Genie, Remembear, SaferPass, SplashData SplashID, Trend Micro and many more.

Methodology

In this project, the Python(version 3.x) programming language is selected as it contains rich cryptograhic packages sufficient for this project like the Cryptography library, the Pycryptodome, Libsodium and many more. Among all, Pycryptodome is selected for this project. Pypcryptodome is a fork of the original PyCrypto python cryptography library that appears to be maintained, and contains algorithms that the has been missing in the Pycrypto library. This library was selected not only because it is recommended for the project, it is also easy to understand and implement and support vast amount of encryption options.

One must avoid having both PyCrypto and Pycryptodome installed at the same time, as they will interfere with each other. When Pycryptodome is installed sing the '`pip install pycryptodome`' command, all the modules are installed under the Crypto package which in a way interferes with the original Pycrypto(no longer maintained) library.

To avoid this, it is recommended that the original Pycrypto library be uninstalled before installing Pycryptodome or the new installation be done using the '`pip install pycryptodomex`' command. This is a Pycryptodome installation that is independent of the original pycrypto library and in this case, all the modules are installed under the Cryptodome package not the Crypto package.

A master password is taken from the user, and it is combined with a 16 byte 'salt' value to generate a key of 32bits using the PBKDF2 function of the Pycryptodome library. The user is then asked to supply the website to which it wants to retrieve its password or save in the database.

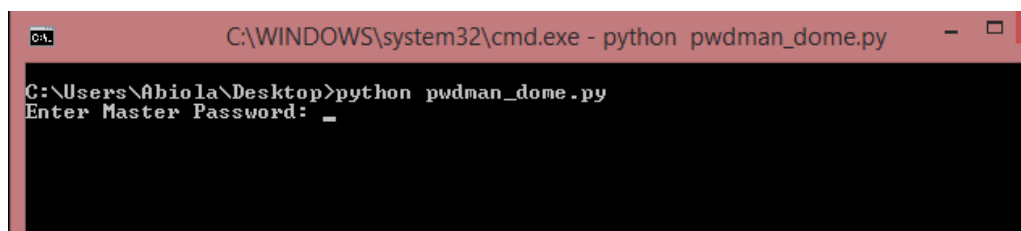
The supplied website by the users is checked against the database, if the account is found, then the website plus the password is retrieved for the user which can be copied to the platform where it is needed but if not found in the database, the user is given an option to either supply a password to be saved for the website or automatically generate a 10 digit random strong password composed of alpha-numeric characters including special characters. The website and the password is then saved in the password file and the file gets encrypted using the Cipher Feedback(CFB) mode of the AES encrypt function in pycryptodome.

The encrypted password file is decrypted whenever a website is to be checked in the database and re-encrypted after a new account is added.

Results and Discussion

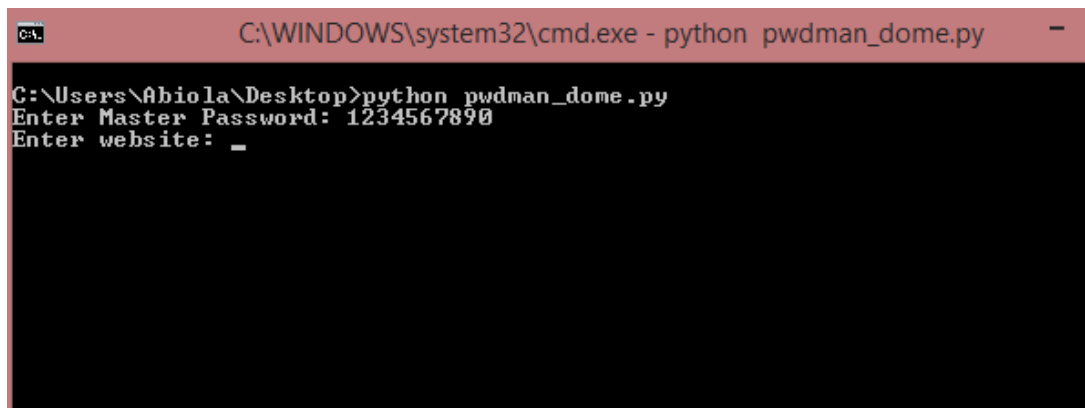
To compile and run this program, the python library Pycryptodomex needs to be installed on the computer using the command 'pip install pycryptodomex'.

On a first run of the program, the user is requested for a master password with which the program will use to decrypt and encrypt on subsequent run of the program.



```
C:\WINDOWS\system32\cmd.exe - python pwdman_dome.py
C:\Users\Abiola\Desktop>python pwdman_dome.py
Enter Master Password: _
```

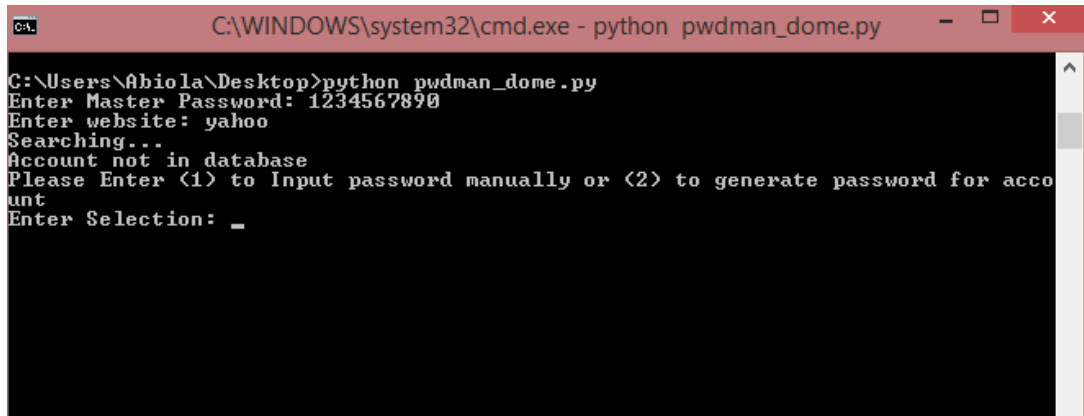
After supplying a password of at least 10 characters long, the user is requested for the website which he wants to retrieve the password.



```
C:\WINDOWS\system32\cmd.exe - python pwdman_dome.py
C:\Users\Abiola\Desktop>python pwdman_dome.py
Enter Master Password: 1234567890
Enter website: _
```

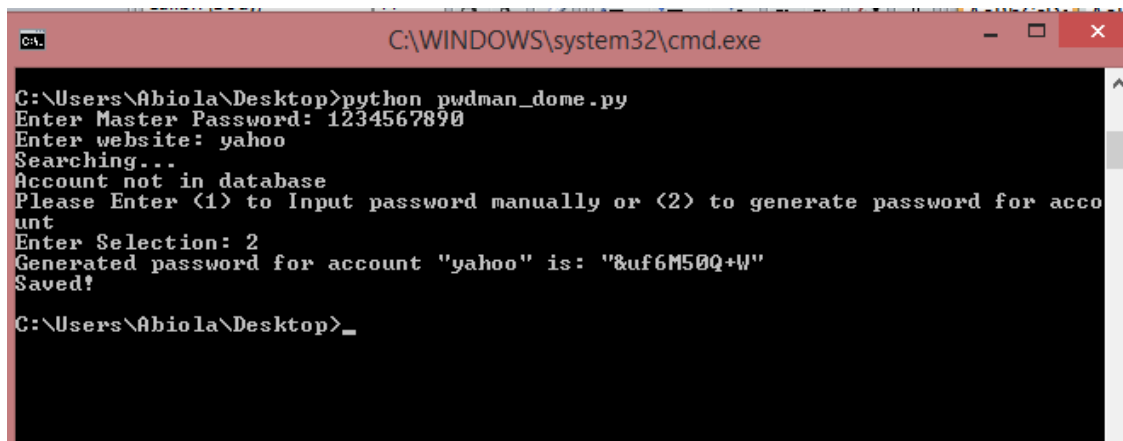
Let's take that the users enters the website 'yahoo'. This input from the user is searched within the password file database and if not found, the user is required to choose to manually supply a

password to be saved with the website to complete the account in the database or allow the program to automatically generate a password with 10 alpha-numeric characters.



```
C:\WINDOWS\system32\cmd.exe - python pwdman_dome.py
C:\Users\Abiola\Desktop>python pwdman_dome.py
Enter Master Password: 1234567890
Enter website: yahoo
Searching...
Account not in database
Please Enter <1> to Input password manually or <2> to generate password for account
Enter Selection: _
```

For this example, let the user reply with '2'. The program automatically generates the password, displays it for the user's immediate use and also saves it in the database. Then encrypts the password file.



```
C:\WINDOWS\system32\cmd.exe
C:\Users\Abiola\Desktop>python pwdman_dome.py
Enter Master Password: 1234567890
Enter website: yahoo
Searching...
Account not in database
Please Enter <1> to Input password manually or <2> to generate password for account
Enter Selection: 2
Generated password for account "yahoo" is: "&uf6M50Q+W"
Saved!
C:\Users\Abiola\Desktop>_
```

On subsequent run of the program, if the user supplies a wrong master password, he is denied access but if the master password is correct, and the website 'yahoo' is supplied again. The program will this time decrypt the password file, then search and retrieve the website in addition to its password saved in the database. In the screenshot below, it will be noticed that the password retrieved is the same as the one that was saved in the previous shot above.

```
C:\WINDOWS\system32\cmd.exe

Enter Master Password: 1234567890
Enter website: yahoo
Searching...
Account not in database
Please Enter (1) to Input password manually or (2) to generate password for account
Enter Selection: 2
Generated password for account "yahoo" is: "&uf6M50Q+W"
Saved!

C:\Users\Abiola\Desktop>python pwdman_dome.py
Enter Master Password: 1234567890
Enter website: yahoo
Account found
Retrieving details...
Website: yahoo
Password: &uf6M50Q+W

C:\Users\Abiola\Desktop>python pwdman_dome.py
Enter Master Password: wrong_password
Enter website: yahoo
Wrong Master password, Try again!

C:\Users\Abiola\Desktop>
```

Password retrieval
function

wrong Master Password test

Now, to test a website which is not already in the database, let us input the website 'google' and also select '1' at the prompt so that we input the password to be saved manually this time. It should be noticed that when the user entered '1' as an option, the program prompted the user to enter the password. This password has to be a minimum of 10 characters, otherwise the user is asked to try again with longer password length.

```
C:\WINDOWS\system32\cmd.exe

Enter Master Password: 1234567890
Enter website: yahoo
Account found
Retrieving details...
Website: yahoo
Password: &uf6M50Q+W

C:\Users\Abiola\Desktop>python pwdman_dome.py
Enter Master Password: wrong_password
Enter website: yahoo
Wrong Master password, Try again!

C:\Users\Abiola\Desktop>python pwdman_dome.py
Enter Master Password: 1234567890
Enter website: google
Searching...
Account not in db
Please Enter (1) to Input password manually or (2) to generate password for account
Enter Selection: 1
Enter pwd: 9876543?@J
Stored!

C:\Users\Abiola\Desktop>
```

To test retrieving the password manually entered, run the program and supply the correct master password, then enter the website 'Google' which the password was manually inputted. The screenshot below shows the program successfully retrieved the password and also was able to still retrieve the previously auto-generated password for the website 'yahoo'.

```
C:\WINDOWS\system32\cmd.exe

Please Enter <1> to Input password manually or <2> to generate password for account
Enter Selection: 1
Enter pwd: 9876543?@J
Stored!

C:\Users\Abiola\Desktop>python pwdman_dome.py
Enter Master Password: 1234567890
Enter website: google
Account found
Retrieving details...
Website: google
Password: 9876543?@J

C:\Users\Abiola\Desktop>python pwdman_dome.py
Enter Master Password: 1234567890
Enter website: yahoo
Account found
Retrieving details...
Website: yahoo
Password: &uf6M50Q+W
```

To reset, we simply remove or delete the encrypted password file using the command 'rm PasswordFile.txt.enc'.