

# Open-Source Passwort Manager

Leon F. Fischer

Frankfurt University of Applied Sciences

SS 2021

## Zusammenfassung

Während andere Passwort-Manager Lösungen mehr und mehr die ‚always online‘ oder sogar die ‚online only‘ Route wählen und oftmals nur in Browsern funktionieren, zeigt dieses Dokument die Konzeptionierung und Ausarbeitung einer offline Lösung, welche aber mit Hilfe von third-party Cloud Software vergleichbare Konnektivität erreichen kann. Es folgt eine Erklärung des Menü-Systems, der Farbwahl, des Passwort Generations Algorithmus, der Datenverschlüsselung und des Löschverfahrens.

Das übergreifende Konzept dieses Projektes ist, dass ein Passwort-Manager vor allem, sogar vor hoher Sicherheit, eine gute Benutzererfahrung anbieten muss. Falls das Speichern und Abrufen der Passwörter zu aufwendig erscheinen, kann es schnell vorkommen, dass der Benutzer aufhört den Manager zu benutzen und stattdessen z.B. wieder anfängt das gleiche, einfach zu merkende Passwort für alles zu verwenden.

## Konzeptionierung

Nach dem Lesen der Anforderungen war schnell klar, dass ein konsolenbasierter Passwort-Manager zu unintuitiv für die meisten - wenn nicht alle - Nutzer sein würde. Daher wurde die Entscheidung getroffen, das Beste aus dem limitierenden Konsoleninterface zu machen. Daher ist das Hauptmenü der Grundstein des Programms, da es die Funktionen einfach darstellt und auch für Konsolenlaien einfach zu bedienen und sicher zu beenden ist. Sicheres schließen des Programms ist gerade durch die gewählte Verschlüsselungsvariante ein wichtiger Teil der Operation. Weitere Menüs an mehreren Stellen des Programms sorgen für eine einfache und saubere Bedienung, welche trotzdem schnell zu benutzen ist. Durch die internationale Natur von Open-Source Projekten wurde Englisch als Sprache innerhalb des Programmes gewählt.

## Menüführung

Nach dem das Programm gestartet wird, wird automatisch das Hauptmenü (siehe Abbildung 1.) angezeigt, zusammen mit einer

Abbildung 1. Hauptmenü

```
Welcome to Passwordmanager.  
What would you like to do?  
  
Choose an option via arrow keys:  
Recall    <-  
New  
Options  
Support Us  
<-Exit  
Press Enter to continue...
```

Willkommensnachricht und einer kurzen Erklärung, wie das Menü bedient wird. Dieses und jedes andere Menü wird mit den Pfeiltasten und der Eingabetaste benutzt. ‚Exit‘ ist hier die einzige Option, die zu keinem Unterpunkt führt, da das Programm damit beendet wird. Falls der Benutzer

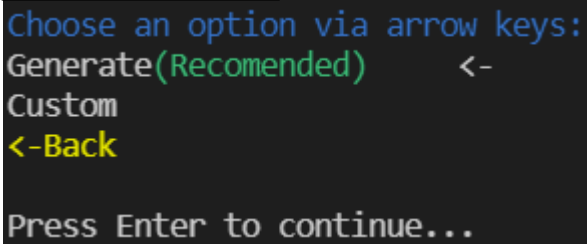
eine der anderen Optionen auswählt, wird das Menü verschwinden und von der gewählten Option ersetzt.

Menüs wie diese bestehen aus einer einfachen Schleife (Code-Beispiel 1.1), die Teile eines Arrays (Code-Beispiel 1.2) druckt und je nach Stand einer Variablen den kleine Cursorpfeil hoch oder runter setzt. Zur Veränderung dieser Variablen namens ‚selected‘ werden zwei Funktionen namens ‚up()‘ und ‚down()‘ (Code-Beispiel 1.3) benutzt. Ein Aufruf dieser Funktionen verändert die Variable und bewirkt, dass sich bei der nächsten Ausgabe des Menüs, welche auch durch die Funktionen ausgelöst wird, sich der Cursorpfeil hinter einem anderen Menüpunkt befindet. Dieser kann jetzt mit der Eingabe Taste ausgewählt werden.

## Farben

In einigen der Menüs wurde mit etwas Farbe gearbeitet, um entweder die Separation der Optionen und des Cursors hervorzuheben (siehe Abbildung 1.) oder um einen Menüpunkt

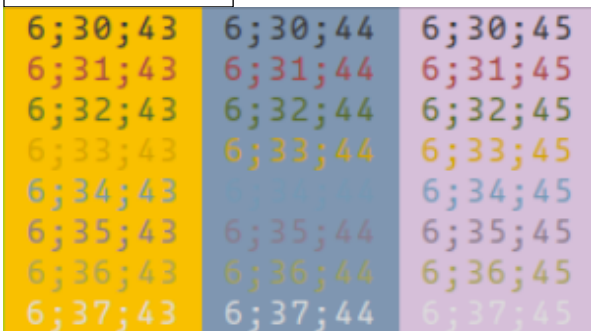
Abbildung 2. Empfohlener Punkt



```
Choose an option via arrow keys:
Generate(Recomended)    <-
Custom
<-Back
Press Enter to continue...
```

visuell zu unterstreichen. Abbildung 2 zeigt das Menü, in dem der Benutzer zwischen einem generierten Passwort und der Eingabe seines eigenen Passwortes auswählen kann. Das in grün hervorgehobene „Recomended“ macht klar, dass das Programm die Verwendung dieser Option vorzieht. Passwörter, die von diesem Programm generiert werden, haben meist eine wesentlich höhere Sicherheitsstufe.

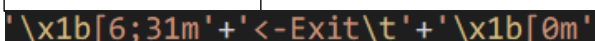
Abbildung 3. Farboptionen  
<https://stackoverflow.com/a/21786287>



Das Eingeben eigener Passwörter ist dennoch eine wichtige Option, da in seltenen Fällen Passwörter von z.B. Firmen - Logins vorgegeben werden. Auch das Speichern existierender

Passwörter, bzw. das Migrieren von einem anderen Passwort-Manager-Service ist damit möglich. In den Menüpunkten werden Farben mit Hilfe von Farbcodes dargestellt. Die hier

Abbildung 4. Farbbeispiel -Rot



```
'\x1b[6;31m'+ '<-Exit\t'+ '\x1b[0m'
```

verwendeten Farbcodes können Variationen von 16 Farben darstellen. In Abbildung 4 wird mit ‚\x1b‘ der Beginn der Farbänderung festgelegt. ‚[6;31m‘ setzt folgenden Text in die Farbe: Rot. (Siehe Abbildung 3. &

Abbildung 1). Danach begrenzt ‚\x1b[0m‘ den gefärbten Text und es geht in normaler Farbe weiter.

## Passwort Generation

Wenn ein Benutzer ein neues Passwort anlegen möchte, wählt er den Punkt ‚New‘ aus dem Hauptmenü (Abb. 1) aus. Daraufhin muss er einen Titel für sein Passwort eingeben. Dieser Titel sorgt für das einfache wiederfinden des Passwortes und verhindert das versehentliche Erstellen eines bereits unter dem Titel existierenden Passwortes. Nach dieser Eingabe wird ein Menü aufgerufen (Abb. 2), welches -wie oben bereits beschrieben-, den Benutzer auffordert, entweder sein eigenes Passwort einzugeben oder es bestenfalls von dem Programm erstellen zu lassen. In diesem Fall wird eine Eingabe des Benutzernamen nötig, da dieses Feld bei den meisten Webseiten vor der Eingabe des Passwortes erscheint. Nachdem der Benutzername eingegeben und bestätigt wurde, ist zu beachten, dass der Benutzername nicht leer sein darf. Dann wird um die Länge des Passwortes gebeten und das Programm empfiehlt, dass das Passwort mindestens 20 oder mehr Charaktere lang ist. Die Länge ist nicht innerhalb des Programms limitiert, da einige Anwendung sehr lange und damit sichere Passwörter unterstützen. Nach Eingabe und Bestätigung der Länge wird ein neues Menü angezeigt, in dem der Benutzer gebeten wird, Bestandteile seines Passwortes an oder auszuschalten. (Abb. 5)

Abbildung 5. Passwort Options Menü

```
Choose an option via arrow keys:  
Lowercase=True <-  
Uppercase=True  
Digits=True  
Punctuation=True  
Confirm?  
  
Press Enter to change selected option...  
Press Enter on 'Confirm?' to confirm...
```

Es sind bei dem Start des Programmes automatisch alle Optionen angeschalten, was zu einem sicheren Passwort führt. Da manche Dienste jedoch nicht alle gegebenen Optionen

unterstützen, ist es wichtig, sie an und ausschalten zu können. Gerade die Option ‚Punctuation‘ -also ‚Satzzeichen‘- kann für Probleme sorgen. Um Optionen zu ändern, wird auch in diesem Menü der Cursor mit den Pfeiltasten bewegt und dann die gewählte Option durch einen Druck auf die Eingabetaste geändert. Dies verändert eine Boolean (Code-Beispiel 2.1), welche entweder Wahr oder Falsch sind, somit ist die Option jeweils entweder an oder aus. Danach wird das Menü neu dargestellt, was aber so schnell geschieht, dass es für den Endbenutzer aussieht, als ob sich nur ein Wort verändert hätte. Nachdem alle Optionen richtig eingestellt sind, muss der Benutzer dies durch das Auswählen und Bestätigen von ‚confirm?‘ dem Programm mitteilen. Danach wird in einem Bruchteil einer Sekunde ein Passwort aus den vorgegebenen Parametern generiert. Dieses wird direkt in die Zwischenablage kopiert und kann innerhalb der nächsten 30 Sekunden in - z. B. das entsprechende Feld einer Webseite - eingefügt werden.

Innerhalb des Programms erfolgt dieses Erstellen wie folgt:

Nachdem die Auswahl der Optionen bestätigt ist, wird für jede aktive Option ein Character - Satz an einen string Namens

‚alphabet‘ angefügt und ausgeschaltete Optionen werden übersprungen. So werden z.B. kleine Buchstaben an den string ‚alphabet‘ angehängt, wenn dies ausgewählt wurde, aber große Buchstaben können übersprungen werden. (Code-Beispiel 2.2). Anschließend wird dieser string an die Funktion ‚newpassword()‘ zurückgegeben. Dieser Funktion kümmert sich um die Erstellung oder Eingabe neuer Passwörter.

Jetzt, nachdem die Parameter des Passwortes genau definiert sind, wird es endlich generiert. Dazu wird ein Teil der ‚secrets‘ Bibliothek verwendet, um mit Hilfe einer Schleife je nach der eingegebenen Länge des Passwortes einen zufälligen Charakter aus dem zuvor besprochenen string ‚alphabet‘ auszuwählen und an einen neuen string anzufügen.

```
newword = ''.join(secrets.choice(alphabet) for i in range(int(length)))
```

Code-Beispiel 3. Passwort Generierung

Dieser string enthält nach Durchlauf der Schleife das neue Passwort. (Siehe Code-Beispiel 3). Nachdem das neue Passwort auf unerwünschte Character - Folgen untersucht wurde, wird es gespeichert und danach für 30 Sekunden in die Zwischenablage kopiert. Nach dem Ablauf dieser Zeit wird die Zwischenablage aus Sicherheitsgründen geleert. Damit soll vermieden werden, dass der Benutzer seine Passwörter aus Versehen in unangebrachte Plätze hineinkopiert.

## Speichern

Wie auf der letzten Seite angesprochen wird das fertige Passwort gespeichert, damit es in der Zukunft auch wieder einfach abgerufen werden kann. Dies geschieht mit Hilfe der ‚save(...)‘ Funktion. (Siehe Code-Beispiel 4.).

```
def save(title,passw,uname): # puts provided strings into the file
    saveline='->'+title+'=>'+passw+'_>'+uname
    f=open('verysecure.txt','a+')
    f.write(saveline+"\n")
    f.close
    return True
```

Code-Beispiel 4. Speicher Funktion

Die Speicherfunktion erhält den zuvor eingegebenen Titel, das generierte Passwort und den eingegebenen Benutzernamen. Diese werden zu einen neuen string zusammengesetzt, jedoch wird vor jedem Teil der Eingabe eine Type-definierende Zeichenfolge angefügt, mit der die einzelnen Teile wieder auseinandergeschnitten werden können, wenn das Passwort wieder gebraucht wird. ‚->‘ steht vor dem Titel, ‚=>‘ steht vor dem Passwort und ‚\_>‘ steht vor dem Benutzernamen. (Siehe Abbildung 6.). Danach öffnet das Programm das Textdokument, in dem alles abgespeichert wird. In dem es mit ‚a+‘ geöffnet wird,

Abbildung 6. Gespeicherte Einträge

```
->facebook=>passwort123_>test@gmail.com
->amazon=>chicken123_>test@gmail.com
->Netflix=>dabbies_>test@gmail.com
```

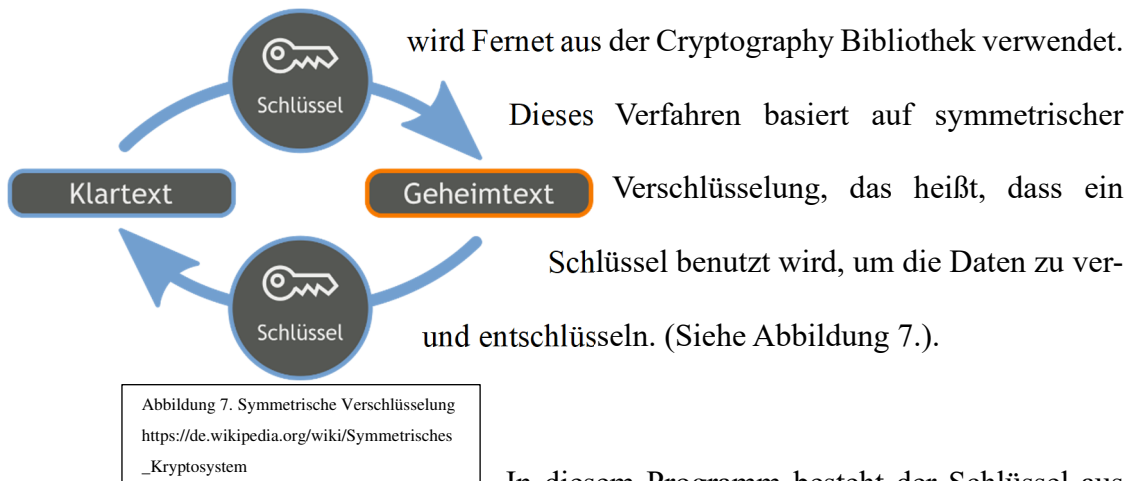
kann das Programm einfach neue Einträge an das Ende des Dokuments anhängen. Nachdem die neue Zeile mit

einem Zeilenumbruch beendet wurde, wird das Dokument wieder geschlossen.



## Datenverschlüsselung

Natürlich ist es nicht zu empfehlen Passwörter einfach als simplen Text zu speichern. Daher verwendet das Programm ein Verschlüsselungssystem, um die gespeicherten Passwörter zu schützen, wenn das Programm nicht in Verwendung ist. Dazu



In diesem Programm besteht der Schlüssel aus

einer speziellen, separaten Datei: einer key.key Datei. Ein Schlüssel sieht dann z.B. so aus: „Y9w\_2OY7SYuMO0OQZsovvUckEaq4JOO29kKhFuTh5Sk= „, und besteht, wie die generierten Passwörter, aus einer Kombination von sowohl kleinen als auch großen Buchstaben, Zahlen und Satzzeichen. Bei Start des Programmes wird der Schlüssel aus der Datei in das Programm

geladen und die Speicherdatei wird entschlüsselt. (Siehe Code-

Abbildung 8. Verschlüsselter Text

```
gAAAAABg12FDSK4q9KdZXWBvIeGXQxfay
LbkAvmEGjx-jGbt25XDyBU8vTWJeylrSb
HOIQsc7FMM5amFaIJZwNqRO2jUMXMfCUU
-Zet0PdyMo7CoMXYWHxau_oiv2_EurA_I
u0bgbM0BVTE5k-SSZlG5GIDd3_NTJsxTQ
ga8L2CarRdCHYZMDGH3bI3LKUGTiYntfE
_3I_d05ICiDDy6Ug0PEzCSv0I904Ljam9
55_nBVuuIt7CrmRr9IYQN1DfJ4ZybrZn7
Xr6qYX_oUWouM4ZQqVhOfIm09Q==
```

Beispiel 5.). Bei Beendigung des Programms wird der geladene Schlüssel wieder genutzt, um die Datei zu verschlüsseln. (Siehe Code-

Beispiel 6.). Die in Abbildung 6. gezeigte Datei ist nach der Verschlüsselung stark verändert. (Siehe Abbildung 8.).

## Löschen gespeicherter Passwörter

Auch bei Verwendung sicherer Passwörter wird empfohlen diese regelmäßig zu ändern. Einige Webseiten fordern ihre Benutzer dazu auf, doch die meisten machen es in der Regel nicht.

Abbildung 9. Options Menü

```
Choose an option via arrow keys:
Delete Password <-
Change Master-Password
Change Password Interval
Request Plaintext Table
<-Back
Press Enter to continue...
```

Um ein Passwort zu löschen, muss der Benutzer einfach von dem Hauptmenü in das Untermenü: Optionen wechseln. (Siehe Abbildung 9.).

Nachdem ‚Delete Password‘ ausgewählt wurde, wird der Benutzer aufgefordert den Titel des zu löschenden Passwortes einzugeben. Daraufhin durchsucht das Programm die Speicherdatei nach dem ähnlichsten Titel. Sicherheitshalber wird der gefundene Titel angezeigt und der Benutzer muss das

Abbildung 10. Löschen Bestätigen

```
You are about to delete:"AMAZON"
Choose an option via arrow keys:
Confirm <-
<-Back
Press Enter to continue...
```

Löschen nochmals bestätigen. Es gibt auch die Option das Löschen abubrechen und zum Optionsmenü zurückzukehren. (Siehe. Abbildung 10.).

Nach der Bestätigung wird der Benutzer aufgefordert das Masterpasswort einzugeben. Das Suchen, Finden und Löschen des Passwortes geschieht mit Code, der auf der Passwort abrufenden Funktion basiert. (Siehe Code-Beispiel 7.). Jedoch wurde der Lösch-Code modifiziert, sodass er statt dem Passwort den Titel ausgibt, damit das Löschen bestätigt werden kann. (Siehe Code-Beispiel 8.).

## Quellenangaben

- <https://stackoverflow.com/questions/56723852/console-select-menu-in-python><sup>1</sup>
- <https://stackoverflow.com/questions/26002497/how-to-run-a-background-timer-in-python><sup>2</sup>
- <https://stackoverflow.com/questions/4940032/how-to-search-for-a-string-in-text-files><sup>3</sup>
- <https://stackoverflow.com/questions/287871/how-to-print-colored-text-to-the-terminal><sup>4</sup>
- <https://www.thepythoncode.com/article/encrypt-decrypt-files-symmetric-python><sup>5</sup>
- <https://cryptography.io/en/latest/fernet/><sup>6</sup>
- <https://stackoverflow.com/questions/3854692/generate-password-in-python><sup>7</sup>
- <https://docs.python.org/3/library/secrets.html><sup>8</sup>
- <https://datatofish.com/executable-pyinstaller/><sup>9</sup>

---

1 Basis des Menü-Systems

2 Basis des Multithreaded Timer

3 Basis der Such-Funktionen

4 Basis der Farbelemente

5 Implementierungs-Anleitung der Verschlüsselung

6 Details zur verwendeten Verschlüsselung

7 Beispiel der ‚secrets‘ Funktion

8 Erklärung der ‚secrets‘ Funktion

9 Erstellung einer .exe Datei

## Code-Beispiele

```
def show_menu(): #displays main menu
    global move
    move=True
    global selected
    global length
    length =6
    global Where
    Where='menu'
    print("\n" * 2)
    print('\x1b[6;32m'+ 'Choose an option via arrow keys:' + '\x1b[0m')
    for i in range(1, length):
        print(menu[i-1], "{1}".format(i, "\x1b[6;32m" + "<-
"+ "\x1b[0m" if selected == i else " "))
        #prints menu and moves < selector thingy when triggered
    print("\nPress Enter to continue...")
```

Code-Beispiel 1.1. Hauptmenü Code

```
menu=['Recall\t', 'New\t', 'Options\t', 'Support Us ', '\x1b[6;31m'+ '<-Exit\t'+ '\x1b[0m']
```

Code-Beispiel 1.2. Hauptmenü Array

```
def down(): # moves cursor down
    global selected
    global move
    if(move==True):
        if selected == length-1:
            return
        selected += 1
        clear()
        if(Where=='menu'):
            show_menu()
        elif(Where=='options'):
            show_options()
        elif(Where=='newpass'):
            show_newpass()
        elif(Where=='delete'):
            show_delete()
        elif(Where=='alpha'):
            show_alphabet()
```

Code-Beispiel 1.3. down() - Funktion

```
def getalphabet(): # switches the password generation options variables on or
off,
    # to be displayed when the menu is printed
    clear() # it mostly just swaps True for False and False for True
    global lcase, ucase, dcase, pcase
    alphabet=''
    #selected = 1
    show_alphabet()
    input()
    if(selected==1): #lower
        if(lcase==True):
            lcase=False
        elif(lcase==False):
            lcase=True
        getalphabet()
    if(selected==2):
        if(ucase==True):
            ucase=False
        elif(ucase==False):
            ucase=True
        getalphabet()
    if(selected==3):
        if(dcase==True):
            dcase=False
        elif(dcase==False):
            dcase=True
        getalphabet()
    if(selected==4):
        if(pcase==True):
            pcase=False
        elif(pcase==False):
            pcase=True
        getalphabet()
```

Code-Beispiel 2.1. getalphabet()-Funktion Teil1 - Menü

```

if(selected==5):
    if lcase+ucase+dcase+pcase == False: #Checks if at least one option is
        selected
        print('At lest one option needs to be active!')
        input()
        getalphabet()
    else: #adds selected character sets to a string to have a password ge
nerated from them
        if lcase == True:
            alphabet=alphabet+string.ascii_lowercase
        if ucase == True:
            alphabet=alphabet+string.ascii_uppercase
        if dcase == True:
            alphabet=alphabet+string.digits
        if pcase == True:
            alphabet=alphabet+string.punctuation

        return alphabet

```

Code-Beispiel 2.2. getalphabet()-Funktion Teil 2 - String

```

def load_key(): # Loads key from key.key file

    return open("key.key", "rb").read()

```

Code-Beispiel 5. load\_key()-Funktion

```

def encrypt(filename, key): # Used key to encrypt the data file

    f = Fernet(key)

    with open(filename, "rb") as file:
        # read all file data
        file_data = file.read()
    # encrypt data
    encrypted_data = f.encrypt(file_data)

    # write the encrypted file
    with open(filename, "wb") as file:
        file.write(encrypted_data)

def decrypt(filename, key): # Used key to decrypt the data file

    f = Fernet(key)
    with open(filename, "rb") as file:
        # read the encrypted data
        encrypted_data = file.read()
    # decrypt data
    decrypted_data = f.decrypt(encrypted_data)
    # write the original file
    with open(filename, "wb") as file:
        file.write(decrypted_data)

```

Code-Beispiel 6. Ver- und Entschlüsselungs-Funktionen

```
def recallpassword(): # finds, cuts, and presents your selected passwords
    # if masterpassword is correct
    global inclip
    exists=False
    clear()
    lookup1=input("Recall password for which service? :")
    lookup2='->'+lookup1.lower()
    with open('verysecure.txt','rt') as file:
        for line in file:
            if lookup2 in line.lower():
                nline=line.split("=>")
                word=nline
                if(lookup2 in nline[0].lower()):
                    nline=nline[0].split("->")

                    exists=True
                    break
    ...
```

Code-Beispiel 7. Passwortaufrufende Funktion

```
lookup1=input("Delete password for which service? :")
lookup2='->'+lookup1.lower()
with open('verysecure.txt','rt') as file:
    for line in file:
        if lookup2 in line.lower():
            nline=line.split("=>")
            if(lookup2 in nline[0].lower()):
                nline=nline[0].split("->")
                todel=nline[1].upper()
                exists=True
                delete=line
                file.close
                break
    ...
```

Code-Beispiel 8. Such-Teil der Passwort-Lösch-Funktion