

**Rapport de Projet:**

# Gestionnaire de Mot de Passe

Projet réalisé par  
Jonny MATHANARUBAN

Dans le cadre du cours  
“Introduction à la sécurité”



# **Sommaire**

**I.        Présentation du projet**

**II.       Pré-requis**

**III.      Fonctionnement**

**IV.      Analyse du code**

**V.        Observation**

**VI.      Conclusion**



## **Contexte**

Pour mieux comprendre le cadre de notre projet, il est essentiel de rappeler que le cours que nous avons suivi se pense sur les principes fondamentaux de la sécurité informatique et de la cryptographie. L'objectif de ce cours est de nous sensibiliser aux enjeux liés à la sécurité, afin que l'on soit en mesure de poser des questions pertinentes lorsque nous serons amenés à concevoir des systèmes d'information à l'avenir, même si on a pas l'intention de se spécialiser dans le domaine de la sécurité.

La sécurité informatique est un domaine vaste dont l'objectif premier est de garantir la protection des systèmes informatiques contre les menaces et les vulnérabilités. Pour atteindre cet objectif, elle emploie des moyens techniques, juridiques et humains.

Mais pourquoi vouloir garantir la sécurité des systèmes informatiques ? Cette démarche repose sur deux objectifs principaux, la première repose sur la protection des personnes, et la seconde la protection d'intérêts, souvent d'ordres financiers.

La sécurité informatique est donc elle-même un moyen et pas une fin en soi. Dans ce projet nous allons nous intéresser aux moyens techniques que l'on peut mettre en place.

On peut les catégoriser par niveau qui correspondent à des disciplines plus ou moins distincts :

- La protection des la protection des systèmes et infrastructures avec la sécurité des systèmes d'information
- la protection des personnes avec la privacy.
- la protection des données brutes avec la cryptologie.

Dans le cadre de notre projet nous allons nous intéresser au dernier point de celui-ci qui est la protection des données brutes.



## **I.Présentation du projet**

Le gestionnaire de mot de passe est un programme dont l'objectif principal est d'utiliser des algorithmes de cryptage pour sécuriser et chiffrer des données brutes, notamment ici de chiffrer des mots de passe , et ainsi garantir leur confidentialité et leur protection contre tout accès non autorisé.

Notre Gestionnaire de Mots de Passe est un programme en Python qui permet de stocker de manière sécurisé les informations liées aux mot de passe associé notamment le site internet ou applications dans lequel on a un compte ainsi que le nom d'utilisateur (login) .

Le projet va utiliser la bibliothèque tkinter pour créer une interface graphique et la bibliothèque cryptography pour chiffrer et déchiffrer les données sensibles.

## **II.Pré-requis**

Pour compiler et exécuter ce projet nous allons devoir disposer d'une installation Python (de préférence python 3) sur notre système.

Pour installer Python sur Linux:

```
#Dans le terminal taper les commandes suivantes :  
user@name:~$ sudo apt-get update  
user@name:~$ sudo apt-get install python3  
  
#Vérifier que python est bien installé en tapant :  
user@name:~$ python --version
```

Ainsi que deux bibliothèque (tkinter et cryptography) pour cela on peut avoir plusieurs méthode soit en passant par apt-get ou bien utiliser le pip install  
Pour simplifier cette tâche j'ai créer un fichier setup.py



Pour utiliser le fichier `setup.py` faudra d'abord configurer un environnement virtuel à l'aide de la commande

```
user@name:~$ python3 -m venv /chemin/vers/le/répertoire/projet
```

Ensuite dans le dossier `venv` activer le bin python de celui-ci à l'aide de la commande

```
user@name:~$ source .bin/activate
#on aura un prompt comme celui-ci
(projet)user@name:~$
#qui sera affiché
```

Et lancer l'installation des dépendance dans le dossier `projet` où est situé `setup.py` à l'aide de la commande

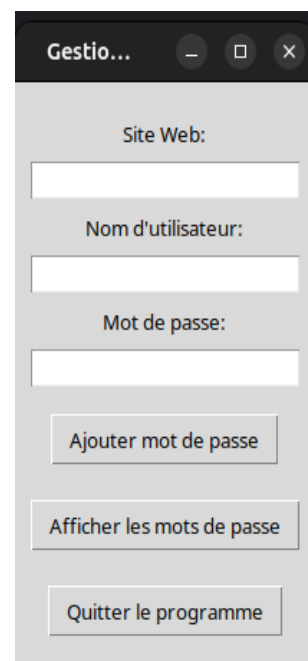
```
(projet)user@name:~$ pip install -e .
```

Ce qui va nous installer toutes les dépendance liés au projet de manière à ce que ça soit installé de manière isolée , pouvoir faciliter la gestion des packages et des versions , permettre le partage d'environnements entre développeurs et maintenir la propreté et l'indépendance de vos projets Python (les avantages de l'environnement virtuel) .

Pour exécuter le programme

```
(projet)user@name:~$ python3 gmpd.py
```

Ce qui va nous afficher une fenêtre comme celle ci:



### **III. Fonctionnement**

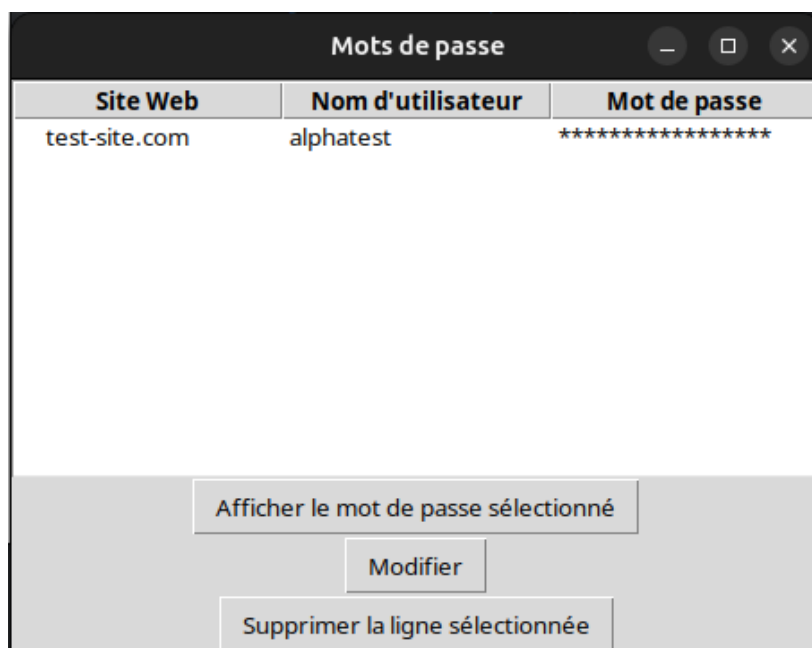
Lorsqu'on exécute le programme , la fenêtre du gestionnaire s'ouvre (voir [Document 1 : Fenêtre du gestionnaire](#)).

Le fichier "key.txt" sera alors créé , il nous sert ici à stocker la clé qui permet de voir nos informations stockées .

Il nous suffira alors d'ajouter les informations demandées dans les champs présents et ensuite cliquer sur le bouton "Ajouter mot de passe".

Ce bouton va nous permettre de créer un fichier "mots\_de\_passe.txt" qui va contenir toutes nos données liées au mot de passe.

Pour afficher nos données saisie et stocké il nous faudra alors cliquer sur le bouton "Afficher les mots de passe" ce qui va nous ouvrir une autre fenêtre intitulé "Mots de passe"



## **Document 2 : Fenêtre Mots de passe**

Le mot de passe est masqué par mesure de sécurité , on peut choisir de l’afficher en appuyant sur le bouton “Afficher le mot de passe sélectionné”

On peut également modifier les informations présentées ci-dessus (c’est à dire Site Web, Nom d’utilisateur et Mot de passe) à l’aide du bouton “Modifier” et finalement supprimer ces données là à l’aide du bouton “Supprimer la ligne sélectionnée”.

Pour quitter le programme on ferme la fenêtre [ci](#) et on clique sur le bouton “Quitter le programme”.

Ce programme permet de stocker, gérer et afficher des mots de passe en utilisant une clé de sécurité stockée dans le fichier “key.txt” .

Nous pouvons ajouter, afficher, modifier et supprimer des informations liées aux mots de passe à l’aide de l’interface utilisateur fournie

## **IV. Analyse du code**

La clé pour le gestionnaire de mot de passe est basée dans le fichier “key.txt”, la première instruction regarde si le fichier “key.txt” existe sinon elle nous affiche un message d’erreur qui nous prévient que le fichier n’existe pas elle est alors générée à l’aide de la fonction `Fernet.generate_key()` . Elle enregistre la clé générée pour crypter et décrypter

Pour cela nous générons une clé de cryptage de manière aléatoire à l’aide de la bibliothèque “cryptography” plus précisément en utilisant le module “Fernet” qui est un système de chiffrement symétrique AES de 128 bits

```
try:
    with open('key.txt', 'rb') as file:
        key = file.read()
except FileNotFoundError:
    key = Fernet.generate_key()
    with open('key.txt', 'wb') as file:
        file.write(key)
```



On crée alors la clé que l'on va enregistrer dans le fichier key

Elle sera générée comme ceci dans notre fichier

```
5wHnI_Fp9WQ1GCnPxhzcsYTqXaqn8XwC-wiENGK2sjk=
```

```
fernet = Fernet(key)
```

cette affectation là permet de créer une instance de la classe fernet en utilisant la clé chargée ou générée

On initialise ensuite 3 variable global

```
site_web_entry = None  
nom_utilisateur_entry = None  
mot_de_passe_entry = None
```

qui vont nous servir à saisir les valeurs dans la partie interface graphique

On définit nos fonctions de cryptage (chiffrer\_mot\_de\_passe) et décryptage (dechiffrer\_mot\_de\_passe) qui prennent une chaîne de caractères en paramètre et renvoient soit une chaîne cryptée ou décryptée à l'aide d'encode et decode avec 'utf-8' elle permet de convertir une chaîne de caractères en octet.

```
def afficher_gestionnaire():
```

Cette fonction est le corps de la fonction d'affichage, c'est ici que l'on définit l'affichage des champs de saisie ainsi que les boutons **Ajouter**, **Afficher** et **Quitter**, il nous faut l'appeler dans le code pour l'afficher.

Dans la bibliothèque "tkinter" le module bouton permet d'appeler une fonction que l'on va définir .

```
def ajouter_mot_de_passe():
```

On a besoin d'enregistrer de manière à crypter les informations saisies , pour cela on va utiliser la fonction chiffrer\_mot\_de\_passe qui va prendre la valeur saisie en paramètre et nous effectuer un cryptage de celui-ci.

Ensuite on va ouvrir ou créer le fichier "mots\_de\_passe.txt" qui lui va prendre toutes les valeurs cryptées et retirer les caractères inutiles qui vont s'ajouter dans la chaîne cryptée à l'aide de `.rstrip(b"b'").rstrip(b"'"')` en effet lorsque on essaie d'ajouter les données cryptées dans un fichier l'ajout de ces caractères





empêche le décryptage de celui-ci une fois fait on ajoute un délimiteur qui nous permet de séparer les informations que l'on a crypter

```
ligne =  
f"{site_web_chiffre.decode()}gestion{nom_utilisateur_chiffre.d  
ecode()}gestion{mot_de_passe_chiffre.decode()}\n"
```

voici un exemple de donnée cryptée à l'aide de cette instruction stockée dans le fichier mots\_de\_passe.txt

```
gAAAAABlRwIAAkWHlOGB_CzAhZt0hgwkGRagO64omo4KIu-UjuKeF3KWDwBjVh  
OwtQFRw15udV39mphm0uCr3RGLUJq_wHIkVw==gestiongAAAAABlRwIAHm-tA  
PscvGT99WDIPYUTfGsQLi38bItojst9yrW8tBvYTKoN1YHEDmcY-Zdc_4AC2fF  
AmMma86eX8ySULGGVEQ==gestiongAAAAABlRwIAtZrZ5IkRuV7VmyHp9gHuZv  
ppkdh1GGaHcBw_ae-uiVSthzuin7X9xHWEe3IGSaFkzFT4-EX-3kf91rxP5n  
Bg==
```

On peut voir que les données cryptées sont clairement délimitées par le mot "gestion".

Si les champs sont vides lors de la saisie le message

```
"Erreur", "Veuillez remplir tous les champs."
```

s'affiche, cela va permettre de remplir tous les champs pour le cryptage.

Une fois enregistré il nous faut afficher dans une sous fenêtre le site web, le nom d'utilisateur et le mot de passe de manière à ce qu'il soit masqué

```
def afficher_mots_de_passe():
```

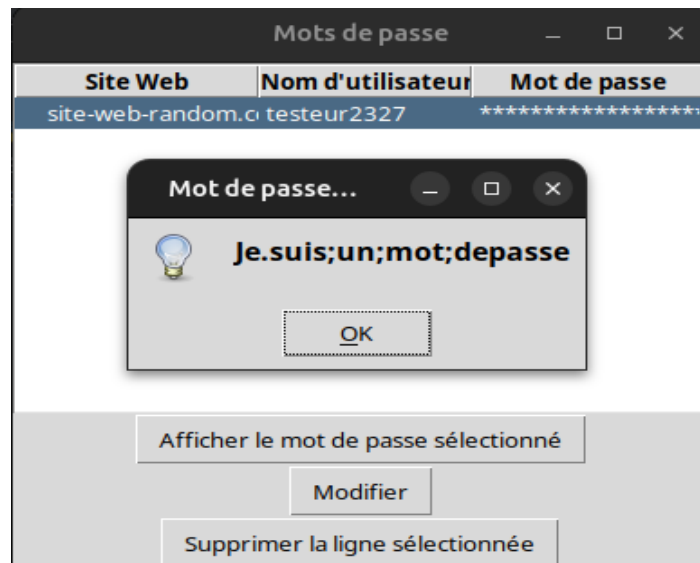
Cette fonction regroupe d'autres fonctions et ainsi que les paramètres du [tableau](#).

On définit d'abord dans un premier temps les colonnes de notre tableau

pour afficher notre mot de passe `def afficher_mot_de_passe_reel():`

masqué on crée une liste vide qui va nous permettre d'affecter l'index des mots de passe ajoutés lors du décryptage de ceux-ci grâce à l'instance `.selection()` cela va nous ouvrir une fenêtre avec le mot de passe (Document 3)





**Document 3 : Affichage du Mots de passe masqué**

Il va nous arriver de devoir modifier les informations que l'on a enregistré si par exemple si le nom du site change ou le besoin de modifier un mot de passe compromis `def modifier_mot_de_passe()` : nous permet d'effectuer cette modification la il va reprendre grâce l'instance de sélection toute la ligne que l'on souhaite modifier et que l'on va supprimer ensuite nous afficher une petite fenêtre qui va nous demander de saisir les informations à modifier.

```
new_site_web = simpdialog.askstring("Modifier le site
web",f"Site Web: {site_web}\nNom d'utilisateur:
{nom_utilisateur}\nAncien mot de passe:
{ancien_mot_de_passe}\nNouveau site web:")
if new_site_web is not None:
...
chiffrer_mot_de_passe(new_mot_de_passe)
```

Et à la fin crypter nos données pour les sauvegarder à nouveau dans le fichier "mots\_de\_passe.txt"

Pour avoir cette fenêtre la



**Document 4 : Affichage de la fenêtre de modification**

on va utiliser une fonction de la bibliothèque “tkinter” qui est `simpledialog.askstring` cette fonction va nous permettre de modifier les informations mis en paramètre

La dernière fonction présent dans la fonction `afficher_mots_de_passe` et la fonction `supprimé` qui supprime de manière définitive à l’aide de l’instance de sélection la ligne sélectionnée du fichier “mots\_de\_passe.txt”

```
def supprimer_ligne_selectionnee():
```

Dans le corps de cette fonction là on retrouve comment masquer le mot de passe , mais afficher le reste , le but premier d’afficher\_mots\_de\_passe et de décrypter les donnée présent dans mots\_de\_passe.txt

Le fait d’avoir mis un délimiteur permet de séparer les information crypté concaténer et de pouvoir ensuite les attribuer à des variables à l’aide de la fonction `.split(b'gestion')` qui permet de séparer quand dans le parcours de la boucle

```
for info in mots_de_passe_chiffres:
    try:
        site_web_chiffre, nom_utilisateur_chiffre,
        mot_de_passe_chiffre = info.split(b'gestion')
```

On ajoute ensuite les mots de passe décrypter dans une liste mot de passe pour les afficher de manière masqué si on n’arrive pas à les décrypter alors un



message d'erreur s'affiche (Document 5)



**Document 5 : Affichage de l'erreur**

On choisit la disposition des boutons et la largeur des colonnes du tableau.

et enfin la fonction `quitter()` qui permet de quitter le programme

## **V.Observation**

Durant le développement de ce programme , on s'est posé plusieurs question: comment générer la clé de cryptage, quel algorithme utiliser , comment rendre ce gestionnaire de mot de passe sûr...

Plusieur point que nous allons aborder ici, le choix de l'algorithme a été décisif il nous fallait un algorithme relativement sûr , nous étions partie sur l'algorithme [TEA](#) (tiny encryption algorithm) qui est un algorithme de chiffrement symétrique , on l'avait adopté pour chiffrer des chaînes de caractères.

Le principal problème rencontré dans l'implémentation de celui-ci était que cela fonctionnait mais la résistance de la clé était relativement faible car on pouvait décrypter de manière inexacte une chaîne cryptée avec une clé légèrement différentes ( mot cryptée "Hello World" → donnée "Hello Warld" par exemple donc pas fiable à 100 % donc on a opté pour l'approche AES situé dans la bibliothèque "cryptography" qui me génère une clé de manière aléatoire , on voulait utiliser un master password pour créer une fenêtre d'authentification et



vérifier si le login correspond à celui entré mais où peut on stocker cela , si on le stock en dur (dans le code) ça sera facilement exploitable pour une tierce personne.

La sécurité dans ce programme réside dans l'exploitation du fichier "key.txt" car si on chiffre un mot de passe on ne pourra plus le décrypter sans celui-ci , il faudra donc la déplacer après utilisation du programme (Sur une clé usb ou dans un dossier random dont on modifiera le nom du fichier "key.txt" par autre chose.txt).

On a choisi de ne pas utiliser d'extension .key car facilement remarquable si on cherche un fichier en particulier (`find *.key` cette commande permet de retrouver tout les extension .key sur la machine utilisé et donc testé une multitude de clé si l'utilisateur a oublié de le déplacer en dehors de la machine ou bien modifier le fichier ).

Le cryptage des information liées aux mots de passe est relativement sûr dans "mots\_de\_passe.txt" car on va directement lire les valeur crypté et c'est au programme avec l'aide de la clé qui va nous décrypter les informations cryptées et nous les afficher à l'aide de l'interface graphique

On peut remarquer aussi plusieurs petit soucis avec `modifier_mot_de_passe` et `afficher_mot_de_passe` on les modifie pas immédiatement il faut quitter la fenêtre ([Document 2](#)) pour que la modification soit prise en compte car le soucis c'est que le `mots_de_passe_afficher` après modification le mot de passe ne va pas être masqué mais il sera tout de même crypter dans le fichier "mots\_de\_passe.txt".

Le stockage des données cryptées pouvait se faire dans une base de données ce qui aurait été plus propre et mieux organisé .

Il aurait fallu définir des fonctions supplémentaires comme l'initialisation de la base de données, modifier ou filtrer les valeurs et drop par la même occasion les valeurs non souhaitées.

Comme idée d'amélioration , on aurait pu créer un générateur de mot de passe aléatoire qui respecte des règles que l'on pourrait sélectionné , ou même utiliser



des images combinant générateur de clé disponible par exemple dans la bibliothèque "cryptography.fernet" comme clé de cryptage (que l'on enregistrera dans un document .key)

## **VI. Conclusion**

En conclusion, notre gestionnaire de mot de passe, développé dans le cadre du cours "Introduction à la sécurité", représente un projet concret qui nous a initié aux concepts de sécurité informatique et de cryptographie. La sécurité informatique vise à protéger les systèmes contre les menaces et vulnérabilités, avec pour objectifs la préservation des personnes et d'intérêts financiers. Notre application, réalisée en Python, utilise les bibliothèques tkinter pour une interface conviviale et cryptography avec le module "Fernet" pour le chiffrement des données. La clé de chiffrement est générée de manière aléatoire et stockée dans le fichier "key.txt", garantissant ainsi la confidentialité des informations.

Pour utiliser notre gestionnaire de mot de passe, Python ainsi que les bibliothèques "tkinter" et "cryptography" sont nécessaires. Nous avons simplifié l'installation avec un fichier "setup.py".

L'application permet d'ajouter, afficher, modifier et supprimer des mots de passe de manière sécurisée. Les mots de passe sont stockés dans le fichier "mot\_de\_passe.txt" et peuvent être masqués ou affichés selon vos besoins.

Malgré son fonctionnement fiable, des améliorations restent possibles, notamment un générateur de mot de passe, l'utilisation d'une base de données et l'identification à l'aide d'une image

En résumé, notre gestionnaire de mot de passe offre une solution sécurisée pour gérer nos mots de passe. Toutefois, il est essentiel de prendre des mesures pour protéger la clé de chiffrement. Ce projet constitue une première étape vers une gestion plus avancée de la sécurité informatique.



# **Sources**

## **Documentation:**

Python. **Python 3 documentation.**

Disponible sur : <https://docs.python.org/3/>

Tkinter. **Tkinter Class API Reference**

Disponible sur : <https://tkdocs.com/pyref/index.html>

Cryptography. **Cryptography**

Disponible sur : <https://pypi.org/project/cryptography/>

Setup. **Python Packaging User Guide**

Disponible sur : <https://packaging.python.org/>

AES. **Advanced Encryption System**

Disponible sur : [https://fr.wikipedia.org/wiki/Advanced\\_Encryption\\_Standard](https://fr.wikipedia.org/wiki/Advanced_Encryption_Standard)

Fernet. **Fernet (symetric encryption)**

Disponible sur : <https://cryptography.io/en/latest/fernet/>

TEA. **Tiny Encryption System**

Disponible sur : [https://fr.wikipedia.org/wiki/Tiny\\_Encryption\\_Algorithm](https://fr.wikipedia.org/wiki/Tiny_Encryption_Algorithm)

Inspiration pour le gestionnaire de mot de passe:

Logiciel. **Keepass**

Disponible sur : <https://keepass.info/>

Inspiration pour le rapport de projet:

Rapport. **PROJET DOMOTIQUE DAIO**

Disponible sur : <http://perso-laris.univ-angers.fr>

