

Gestion des Certificats

Access

Setup For Development

In order to complete the project, it would have been necessary to install Python 3 with Pip

```
pip install -r requirements.txt
```

Conception :

Créer un menu textuel :

```
#----- FUNCTIONS -----#
def certificate_chains():
    #We've set a Vault client to put,update or delete our keys
    client=utils.connect_to_vault(VAULT_ADDR,VAULT_TOKEN)
    while True:
        choice = utils.menu()
        utils.switch(choice)
#-----#

#----- LAUNCHING THE PROGRAMM -----#
try:
    certificate_chains()
except Exception as e:
    logging.error(e)
#-----#
```

On instancie la connection HTTPS à vault grâce un token admin et une adresse. : **
client=utils.connect_to_vault(VAULT_ADDR,VAULT_TOKEN) ** Puis on instancie les options.

```
#----- MENU -----#
def menu():
    logging.info('Press Enter button to continue...')
    input()
    os.system('cls')
    print('Selection Menu : please choose an option => ',
          '1. VERIFY matching(key/certificate)',
          '2. LIST all key(in folder)',
          '3. ADD Certification',
          '4. EXIT\n',
          sep='\n',
          )
```

```

    choice = input('Select => ')
    return choice

def switch(_value):
    cases = {
        '1': lambda: logging.info('Matching State ==>
    {} \n'.format(match_between_key_n_certificate()))),
        '2': lambda: list_Cert(),
        '3': lambda: insert_Cert(),
        '4': lambda: exit(),
    }
    cases.get(_value, lambda: print("This value is incorrect or doesn't
    exist...please try again !"))()
#-----#

```

Ensuite, on peut utiliser plusieurs option, comme inserer, modifier, lire les certificats.

Insérer

L'utilisateur choisit s'il veut un insérer dans fichier csv spécifique ou en créer un nouveau.

On appelle la commande **utils.list_CSV()**

```

def list_csv():
    PATH=(os.getcwd()+"/csv")
    FILES = [f for f in listdir(PATH) if.isfile(join(PATH, f))]
    CERT_LIST = []
    for f in FILES:
        CERT_LIST.append(f)
    return CERT_LIST

```

S'il choisit un nouveau csv on initialise le **bool** create à 0

On appelle la commande **utils.insert_Cert_Create(csv,create,private_key,cert)**

```

# - Option 3 - #
def insert_Cert_Create(csv_file,new_file,key_file,cert_file):

    myDateTime = datetime.datetime.today()
    DATE = ("{0}-{1}-{2}_{3}-{4}-{5}".format(myDateTime.year, myDateTime.month,
    myDateTime.day,myDateTime.hour, myDateTime.minute,myDateTime.second))

    C = {'KeyFile': [key_file],'CertFile': [cert_file],'Date': [DATE]}

    df = DataFrame(C, columns= ['KeyFile', 'CertFile', 'Date'])
    File_Dest=PATH+"/csv/"+csv_file
    if create:
        export_csv = df.to_csv(File_Dest,mode='a', header=True, encoding='utf-8',
        sep=';')
    else:

```

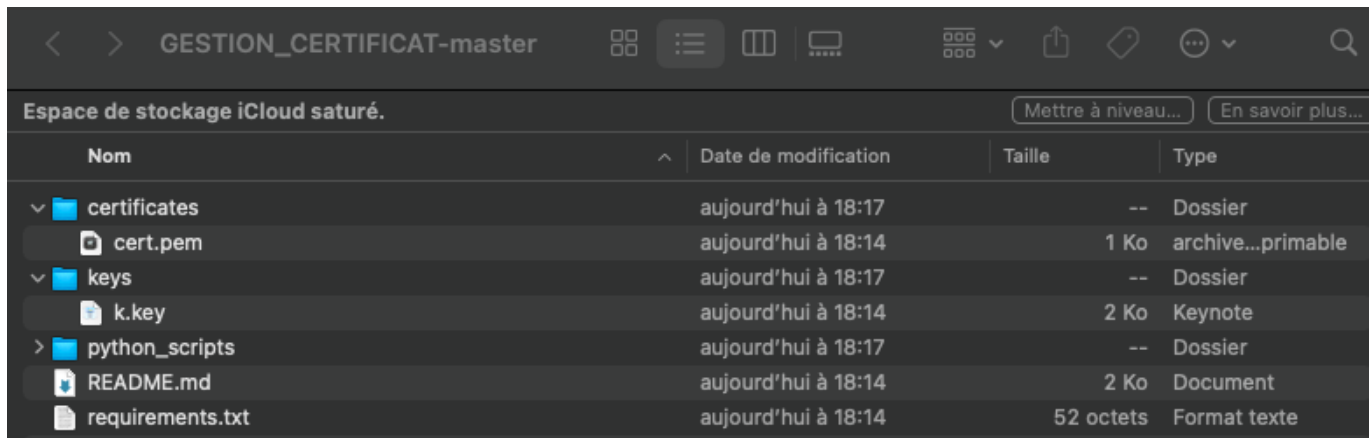
```
export_csv = df.to_csv(File_Dest,mode='a', header=False, encoding='utf-8',
sep=';')
```

Important

header=True rajoute les en-têtes pour un nouveau fichier du Dictionnaire **C**

Enfin on peut vérifier si les certificats ne sont pas encore obsolètes.

Tout d'abord on met un fichier .key dans le dossier KEYS et un fichier .pem ou ctr dans CERTIFICATS



The screenshot shows a file manager interface with a dark theme. At the top, there's a navigation bar with icons for back, forward, and search. Below it, a status bar indicates 'Espace de stockage iCloud saturé.' with buttons for 'Mettre à niveau...' and 'En savoir plus...'. The main area displays a list of files and folders:

Nom	Date de modification	Taille	Type
certificates	aujourd'hui à 18:17	--	Dossier
cert.pem	aujourd'hui à 18:14	1 Ko	archive...primable
keys	aujourd'hui à 18:17	--	Dossier
k.key	aujourd'hui à 18:14	2 Ko	Keynote
python_scripts	aujourd'hui à 18:17	--	Dossier
README.md	aujourd'hui à 18:14	2 Ko	Document
requirements.txt	aujourd'hui à 18:14	52 octets	Format texte

Ensuite avec le module openssl on va décrypter la clé et le certificat SHA1 et vérifier leur correspondance

```
try:
    cert_file = open(cert, 'r')
    cert_data = cert_file.read()
    certificate = crypto.load_certificate(crypto.FILETYPE_PEM, cert_data)
except OpenSSL.crypto.Error:
    raise Exception('certificate is not correct: %s' % cert_file)

context = OpenSSL.SSL.Context(OpenSSL.SSL.TLSv1_METHOD)
context.use_privatekey(private_key_obj)
context.use_certificate(certificate)
try:
    context.check_privatekey()
```

On Retourne le résultat de la correspondance à l'utilisateur.

Meta

Bourdon Maxime – Mbourdon.pro@gmail.com

<https://github.com/Mbourdon95/github-link>