

Cryptographie

```
# Générer une clef aléatoire  
key = Fernet.generate_key()  
# Obtention de l'objet Fernet  
fernet = Fernet ( key )  
# Chiffre le contenu d'un fichier  
with open ( filepath, "rb" ) as f :  
    encrypted = fernet.encrypt ( f.read() )  
# Déchiffre le contenu d'un fichier  
with open ( filepath, "rb" ) as f :  
    decrypted = fernet.decrypt ( f.read() )
```

Winreg

```
# Créer une clef de registre dans HKCU  
winreg.CreateKey ( winreg.HKEY_CURRENT_USER, REG_PATH )  
# Ouvrir une clef de registre en écriture  
# dans HKCU  
reg_key = winreg.OpenKey ( winreg.HKEY_CURRENT_USER,  
    REG_PATH, 0, winreg.KEY_WRITE )  
# Assigner une valeur à une clef de registre  
winreg.SetValueEx ( registry_key, name, 0,  
    winreg.REG_SZ, value )  
# Fermer une clé de registre  
winreg.CloseKey ( reg_key )
```

Requests

```
# Simple requête GET  
r = requests.get ("https://esdacademy.eu")  
# Requête POST avec envoi de données  
r = requests.post  
("https://esdacademy.eu/login", data = {  
    username="esdmaster"  
    password="2secret4u"  
})  
# Obtention du code de retour HTTP, des  
en-têtes de réponse et de la page web  
status = r.status_code  
headers = r.headers  
html = r.text
```

ESD academy

Site : esdacademy.eu
Twitter : @esd_academy
Instagram : esd_academy
Github : ESD-academy



Aide mémoire



esdacademy.eu

SubProcess

```
# Demarrer un sous-processus
subprocess.call ("calc.exe")

# Demarer un sous-processus dans un shell
subprocess.call ([ "dir"], shell = True)

# Renvoie la sortie d'un sous-processus
subprocess.check_output ([ "netsh", "Wlan",
                           "show", "profiles"], stderr = subprocess.PIPE)

# Recuperation de la sortie erreur
p = subprocess.Popen ([ "unknow-command"], stdout = subprocess.PIPE)

# Creation d'un socket TCP
s = socket.socket (socket.AF_INET,
                   socket.SOCK_STREAM)

# Creation d'un socket UDP
s = socket.socket (socket.AF_INET,
                   socket.SOCK_DGRAM)

# Se connecte a un socket
s.connect ((host, port))

# Reception de donnees
s.recv (1024)

# Envoi de donnees
s.send (data)

# Resolution de nom
socket.gethostname (hostName)

# Recuperation du nom d'host
socket.gethostbyaddr (hostIp)
```

Scapy

```
# Creation d'un paquet TCP/IP
packet = IP (dst = "192.168.1.24",
             src = "192.168.1.17")
/ TCP (dport = 443, flags = "S")

# Envoi du paquet et attente de la reponse
sr1 (packet)

# Inspection d'un paquet
ls (packet)

# Creation d'une requete ARP
packet.show ()

# Sniff le traffic TCP sur le port 80 et
# applique une callback sur chaque paquets
captures = sniff (filter = "tcp port 80",
                  prn = callback)

# Sniff le traffic sur le port 80 et
# capture les captures
sniff (filter = "tcp port 80", prn = callback)
```

```
# Capture d'écran instantanée
screen = pyautogui.screenshot()

# Enregistrement de la capture
screen.save (filename)
```

ScreenShot

```
# Win32Clipboard
data = win32clipboard.OpenClipboard ()
win32clipboard.GetClipboardData ()
win32clipboard.CloseClipboard ()

# Recuperer le contenu du presse-papier
# Remplace le contenu du presse-papier
# par le contenu du clipboard
data = win32clipboard.OpenClipboard ()
win32clipboard.EmptyClipboard ()
win32clipboard.SetClipboardText (string)
win32clipboard.CloseClipboard ()
```

Win32Clipboard