## Mot de passe & Attaque « *bruteforce* »

**4 –** Un système utilise des mots de passe sur 5 lettres uniquement.

a. Proposez une méthode *brute force* pour casser les mots de passe utilisés par ce système.

b. Programmez cette méthode en Python et essayez là sur des mots de passe choisis entre vous.

```
#!/usr/bin/python3

import subprocess
import string
import sys

# utilisation d'unicode
alphabet = string.ascii_lowercase + 'éèàùç#@!'
alphabet += alphabet.upper()
print(alphabet)
subprocess.run("stty -echo", shell=True)
mdp_à_découvrir = input("Entrez le mot de passe:")
subprocess.run("stty echo", shell=True)

def construction_proposition(taille,p):
    global mdp_à_découvrir
    if (taille != 0):
        for c in alphabet:
            construction_proposition(taille-1,p+c)
    else:
        if (p == mdp_à_découvrir):
            print("Trouvé ! votre mot de passe est :", p)
            sys.exit(1)

# version récursive extensible
#construction_proposition(5,'')

# version itérative
for c1 in alphabet:
    for c2 in alphabet:
        for c3 in alphabet:
            for c4 in alphabet:
                for c5 in alphabet:
                    proposition = c1+c2+c3+c4+c5
                    if (proposition == mdp_à_découvrir):
                        print("Trouvé ! votre mot de passe est :", proposition)
                        sys.exit(1)
```

**6 –** Vous avez récupéré une archive ZIP protégée par un mot de passe.
Essayez les mots du dictionnaire comme mot de passe pour retrouver le contenu de l'archive.

Créer un programme en Python pour automatiser la procédure.

```
$ unzip -P password archive.zip
```

```
#!/usr/bin/python3

import subprocess,sys,re

nom_fichier_dico = "Corpus_des_mots_francais_propre-UTF8.txt"
nom_fichier_zip = "archive.zip"

re_erreurs = re.compile(r"incorrect|error")

try:
    dico = open(nom_fichier_dico,"r")
except Exception as e:
        print(e.args)
        sys.exit(0)

while(True):
    mot = dico.readline().rstrip('\n')
    if not mot:
        break
    commande = subprocess.Popen("unzip -P %s -o archive.zip 2>/dev/stdout"%mot,
shell=True,stdout=subprocess.PIPE)
    (resultat, ignorer) = commande.communicate()
    correspondance = re_erreurs.search(str(resultat,encoding='utf8'))
    if not correspondance:
        print(resultat)
        print("Trouvé ! le mot de passe est ",mot)
        break
dico.close()
```

## Intégration Empreinte

**11 –** Faire un programme en Python permettant de stocker l'empreinte d'un fichier dans un autre fichier ayant le même nom suffixé par le nom de la fonction de hachage (par exemple "fichier.SHA256" ).

```
#!/usr/bin/python3

import subprocess, re, sys

fonction_hachage = "sha256"

nom_fichier = sys.argv[1]
nom_fichier_empreinte = nom_fichier + "." + fonction_hachage
empreinte = subprocess.run("openssl dgst -%s < %s"%(fonction_hachage,nom_fichier),
shell=True,stdout=subprocess.PIPE)
try:
    fichier = open(nom_fichier_empreinte,"bw")
except Exception as e:
    print(e.args)
    sys.exit(1)
fichier.write(empreinte.stdout)
fichier.close()
```

**12 –** Faire un programme en Python permettant d'intégrer un fichier et ses fichiers empreintes (MD5 et SHA256) associées dans une seule archive.

```
#!/usr/bin/python3

import subprocess, re, sys

liste_fonction_hachages = ("sha256", "md5")

nom_fichier = sys.argv[1]
liste_fichiers_à_archiver = nom_fichier

for fonction_hachage in liste_fonction_hachages:
    nom_fichier_empreinte = nom_fichier + "." + fonction_hachage
    liste_fichiers_à_archiver += " " + nom_fichier_empreinte
    empreinte = subprocess.run("openssl dgst -%s < %s"%(fonction_hachage,nom_fi
chier), shell=True,stdout=subprocess.PIPE)
    try:
        fichier = open(nom_fichier_empreinte,"bw")
    except Exception as e:
        print(e.args)
        sys.exit(1)
    fichier.write(empreinte.stdout)
    fichier.close()

subprocess.run("zip mon_archive_empreintes.zip %s"%liste_fichiers_à_archiver,
shell=True)
```

**13 –** Faire un programme en Python permettant de faire l'inverse : calcul et vérification automatique des empreintes lors de l'extraction du fichier de l'archive.

```
#!/usr/bin/python3

import subprocess, re, sys

liste_fonction_hachages = (b"sha256", b"md5")
chemin_temp = b"temp"

nom_fichier = bytes(sys.argv[1],encoding='utf8')
subprocess.run(b"mkdir %s;cd %s;unzip ../%s"%(chemin_temp,chemin_temp,nom_fichier),shell=True)

liste_fichiers_repertoire_temp = subprocess.run(b"ls %s"%chemin_temp,shell=True,st
dout=subprocess.PIPE).stdout

for nom_fichier in liste_fichiers_repertoire_temp.splitlines():
    for fonction_hachage in liste_fonction_hachages:
        if (fonction_hachage in nom_fichier):
            try:
                fichier_empreinte = open(nom_fichier,"rb")
            except Exception as e:
                continue
            empreinte_sauvegardée = fichier_empreinte.readline()
            fichier_empreinte.close()
            re_nom_fichier_sans_extension = re.compile(rb"(.*)\."+fonction_hachage)
            nom_fichier_sans_extension = re_nom_fichier_sans_extension.search(nom_fi
chier).group(1)
            empreinte = subprocess.run(b"openssl dgst -%s < %s"%(fonction_hachage,nom_fi
chier_sans_extension), shell=True,stdout=subprocess.PIPE).stdout
            if (empreinte_sauvegardée == empreinte):
                print(nom_fichier_sans_extension, "-> OK pour ", fonction_hachage)
```