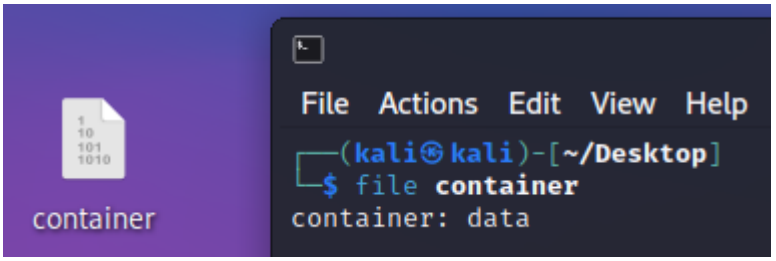
Write Up : **Zipception**

Step 1

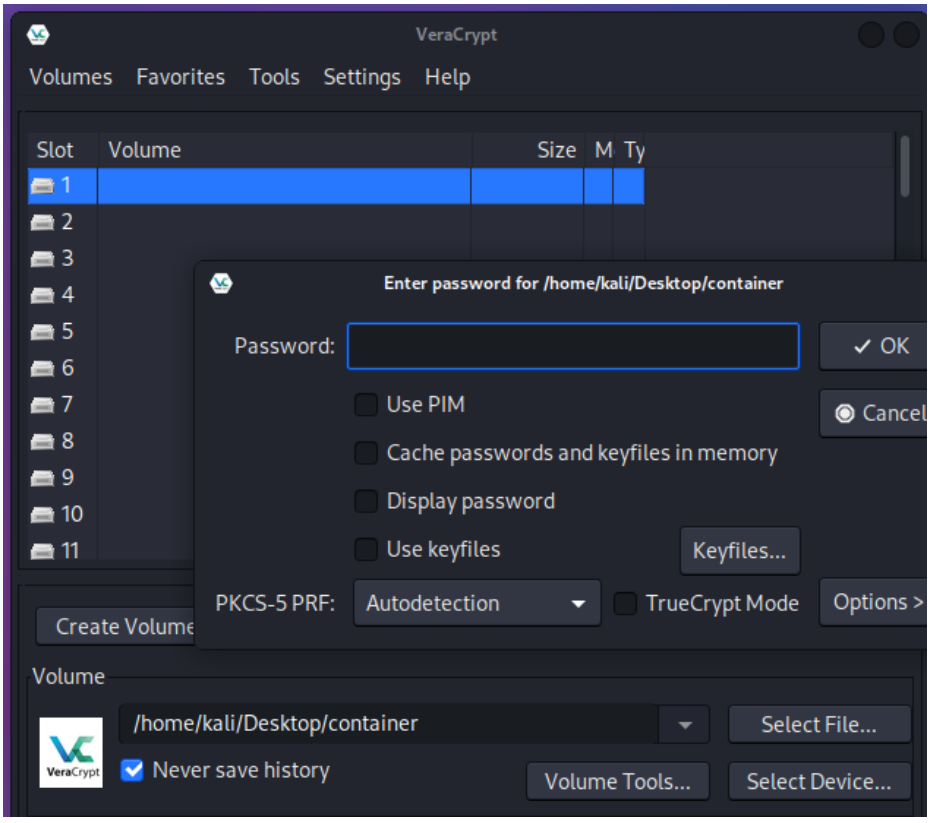
Le fichier initial fourni pour le challenge est nommé “**container**”



Nous n’avons pas d’information sur le format du fichier mais étant donné le nom et la description du chall, on peut supposer qu’il s’agit d’un container VeraCrypt (ou TrueCrypt).

On installe VeraCrypt à partir du lien https://www.veracrypt.fr/en/Downloads.html (tarball)

Le format est bien confirmé lorsque l’on tente de le monter, ça demande un mot de passe:



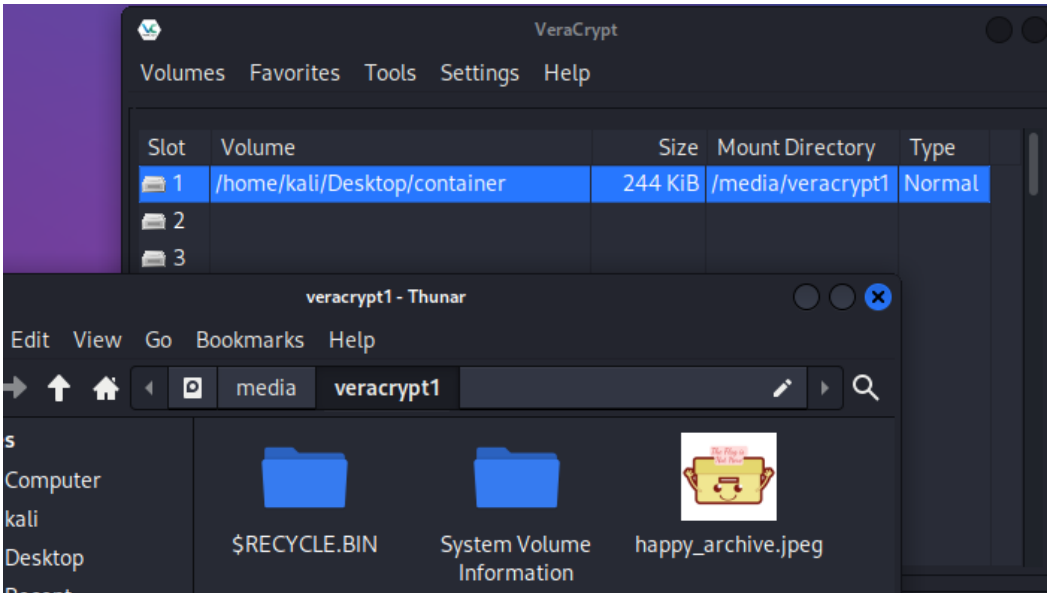
Sans information supplémentaire, on va donc tenter un bruteforce (spécifique VeraCrypt) https://codeonby.com/2022/01/19/brute-force-veracrypt-encryption/

```
┌──(kali㉿kali)-[~/Desktop/test]
└─$ hashcat -w 1 -m 13721 container /usr/share/wordlists/rockyou.txt
hashcat (v6.2.6) starting
```

Le mot de passe (qui est en début de la wordlist rockyou) est trouvé très rapidement :

```
container:iloveu

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 13721 (VeraCrypt SHA512 + XTS 512 bit (legacy))
Hash.Target......: container
Time.Started.....: Tue May  2 15:43:21 2023 (6 secs)
Time.Estimated...: Tue May  2 15:43:27 2023 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:       22 H/s (0.61ms) @ Accel:128 Loops:125 Thr:1 Vec:4
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 128/14344385 (0.00%)
Rejected.........: 0/128 (0.00%)
Restore.Point....: 0/14344385 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:499875-499999
Candidate.Engine.: Device Generator
Candidates.#1....: 123456 → diamond
Hardware.Mon.#1..: Util: 76%

Started: Tue May  2 15:43:14 2023
Stopped: Tue May  2 15:43:28 2023
```

On monte alors le container sur un disque avec le mot de passe trouvé :



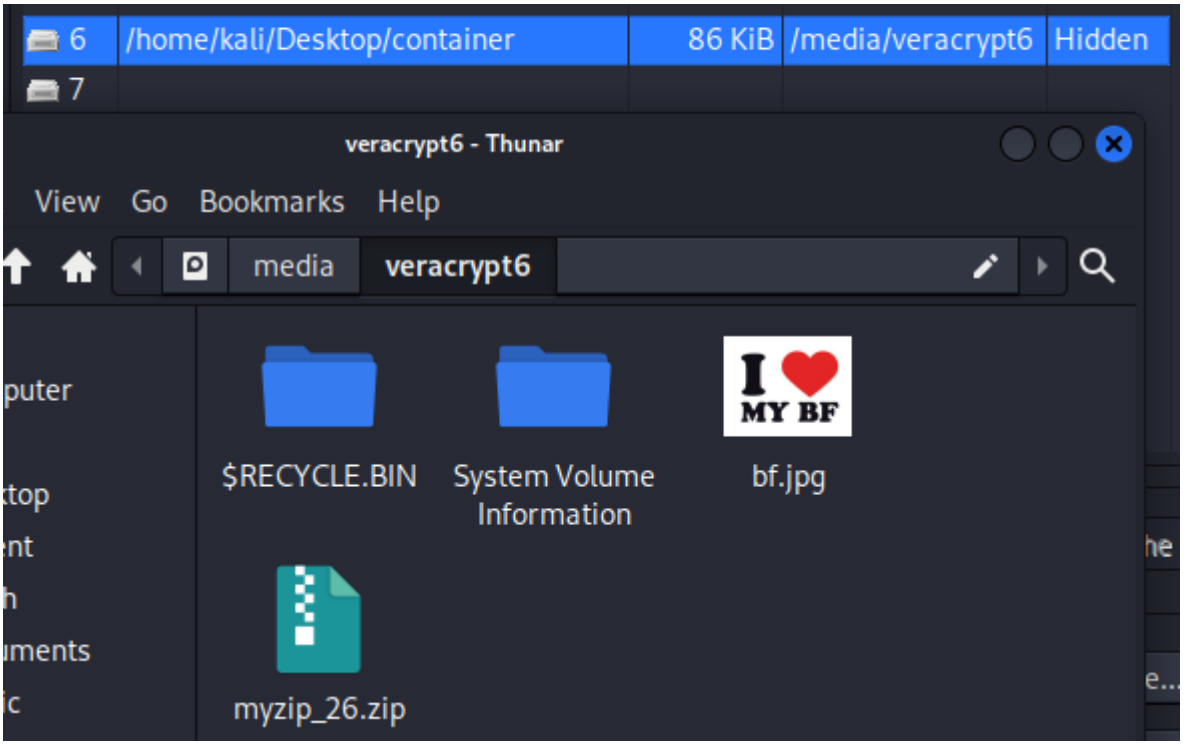
Une image semble nous indiquer qu'on fait fausse route…

Mais en regardant le répertoire monté d'un peu plus près, il existe un fichier caché :

```
┌──(kali㉿kali)-[/media/veracrypt1]
└─$ ls -la
total 140
drwx------ 4 kali kali  16384 Dec 31  1969  .
drwxr-xr-x 3 root root   4096 May  2 15:48  ..
drwx------ 2 kali kali    512 May  2 13:12 '$RECYCLE.BIN'
-rwx------ 1 kali kali 120758 May  2 12:27  happy_archive.jpeg
-rwx------ 1 kali kali     10 May  2 14:52  .password
drwx------ 2 kali kali    512 May  2 14:26 'System Volume Information'

┌──(kali㉿kali)-[/media/veracrypt1]
└─$ cat .password
spongebob
```

VeraCrypt offre la possibilité de créer des "**hidden folders**" au sein d'un container, https://arcanecode.com/2021/05/31/creating-and-using-hidden-containers-in-veracrypt/

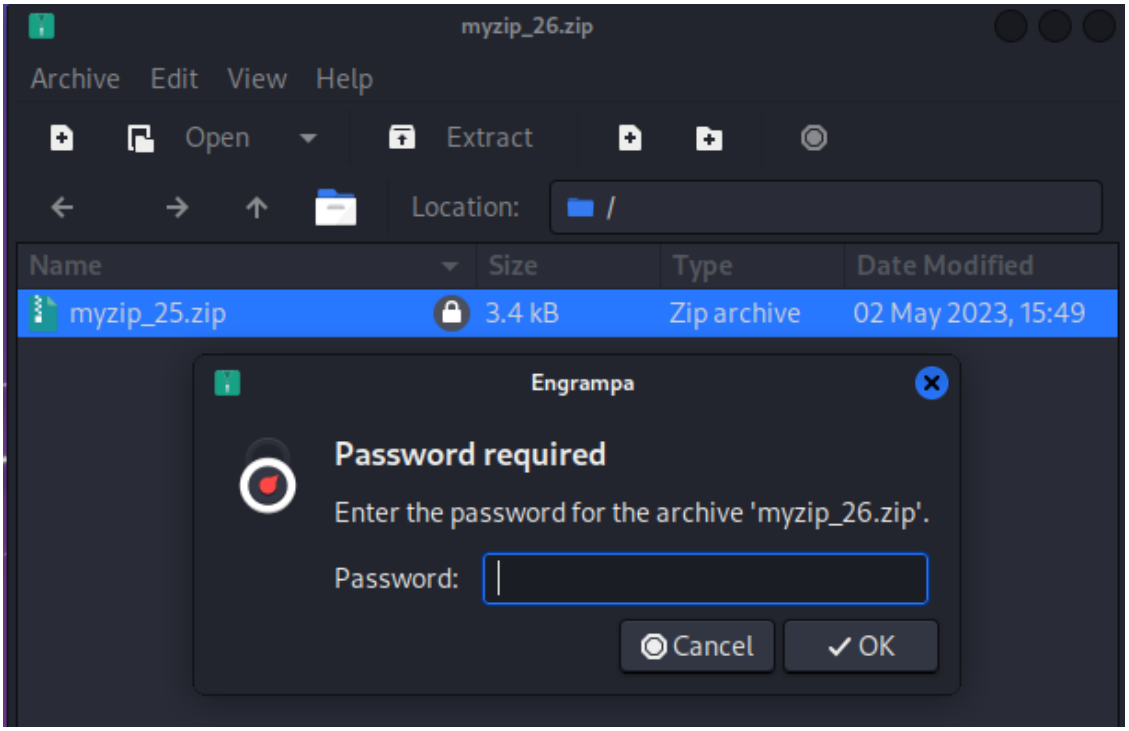
On va donc tenter de monter le container avec le mot de passe "spongebob" cette fois-ci.



On récupère alors une archive zip ! (et une image)

Step 2

On voit que l'archive contient elle-même une archive et qu'un mot de passe est demandé



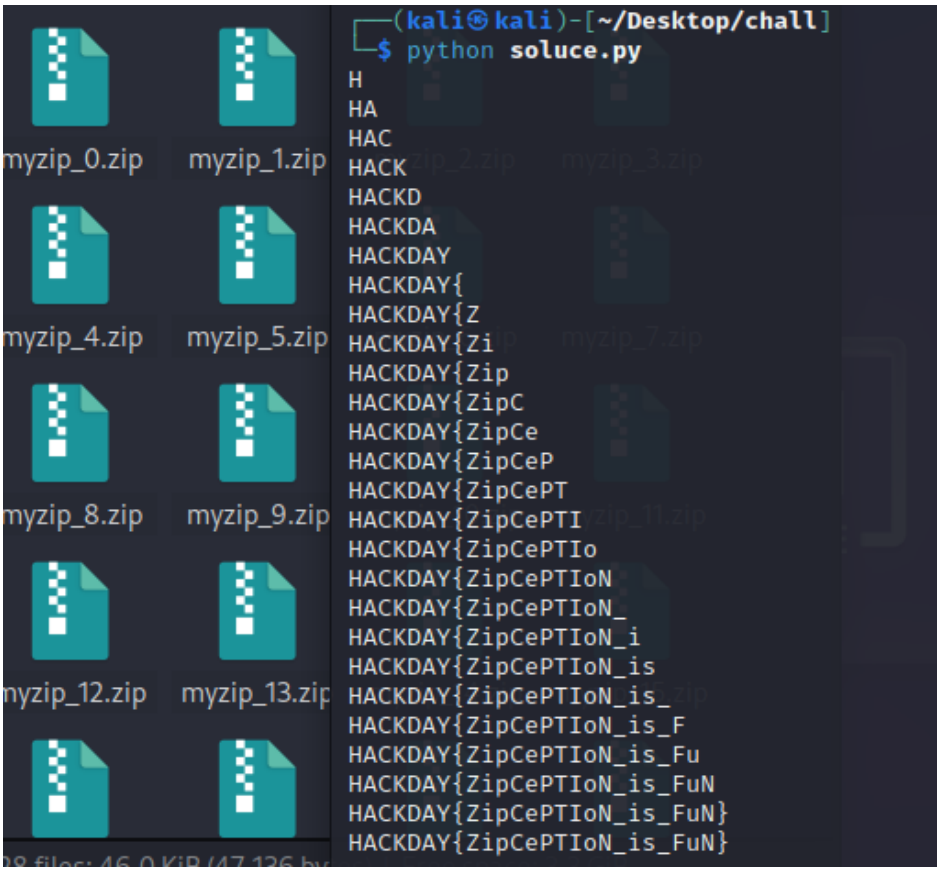
D’après le nom du challenge et de la numérotation des archives, la suite apparaît assez évidente, il va falloir automatiser avec un script la décompression des différentes archives.

Par ailleurs l’image est un hint qui semble indiquer qu’il s’agit d’un **bruteforce.**

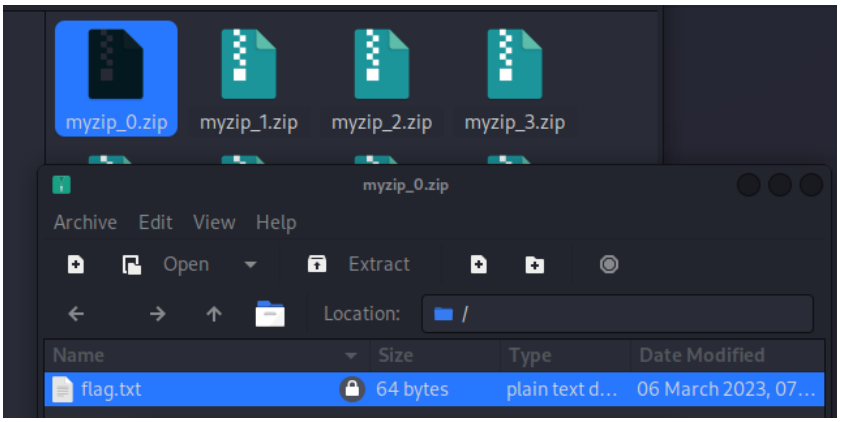
On écrit donc le script python suivant (exemple) :

```
#soluce.py
import zipfile

flag = ""
zip_number = 26

for i in reversed(range(zip_number+1)):
    file_name = 'myzip_%d.zip' % i
    for c in range(30,127):
        try:
            with zipfile.ZipFile(file_name) as file:
                file.extractall(pwd = bytes(chr(c), 'utf-8'))
            flag += chr(c)
            break
        except:
            pass

    print(flag)
```

Et l'on obtient le flag :)



PS : la dernière archive qui est nommée "**myzip_0.zip**" n'a pas été décompressée, son mot de passe est "*password*" et elle contient un fichier flag.txt



Ce dernier laissera une chance supplémentaire, avec un indice pour les personnes ayant malheureusement oublié de "print" les caractères au fur et à mesure du bruteforce.