

FOFANA Abdel-malik

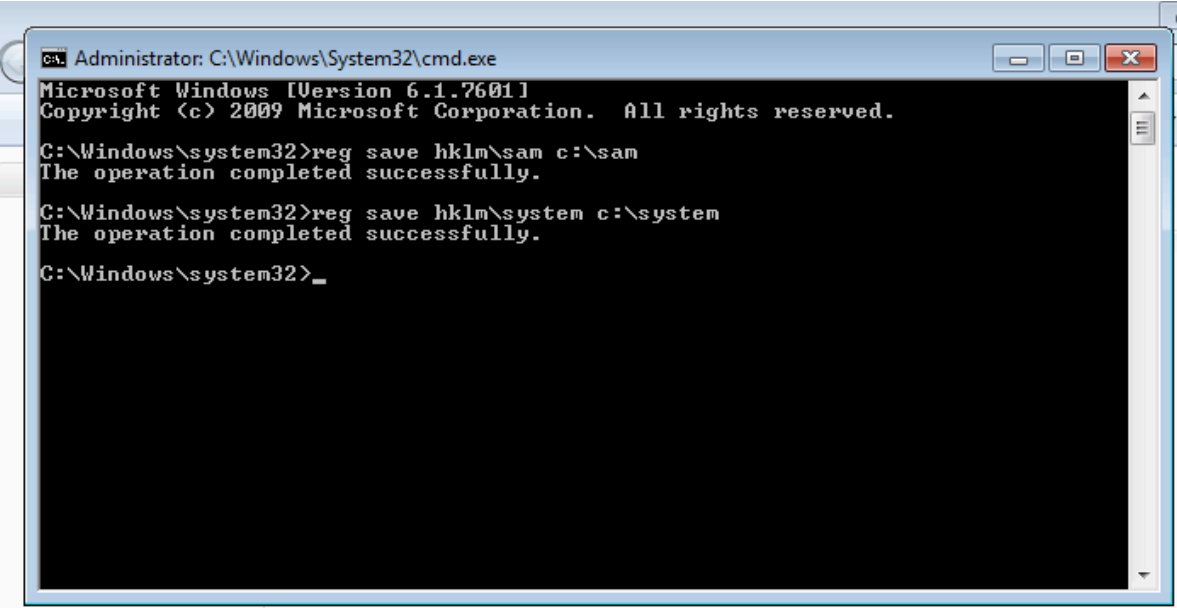
FOURMONT Thibault

Lab Cracking-Steganographie

Partie 1 : cracking mot de passe

Pour cette partie du TP, j'ai décidé d'utiliser une machine Windows 7 ainsi qu'une machine Kali Linux afin de cracker les hashes extraits de la machine Windows.

J'ai utilisé les commandes ci-dessous afin d'extraire le fichier SAM et System de la machine Windows.

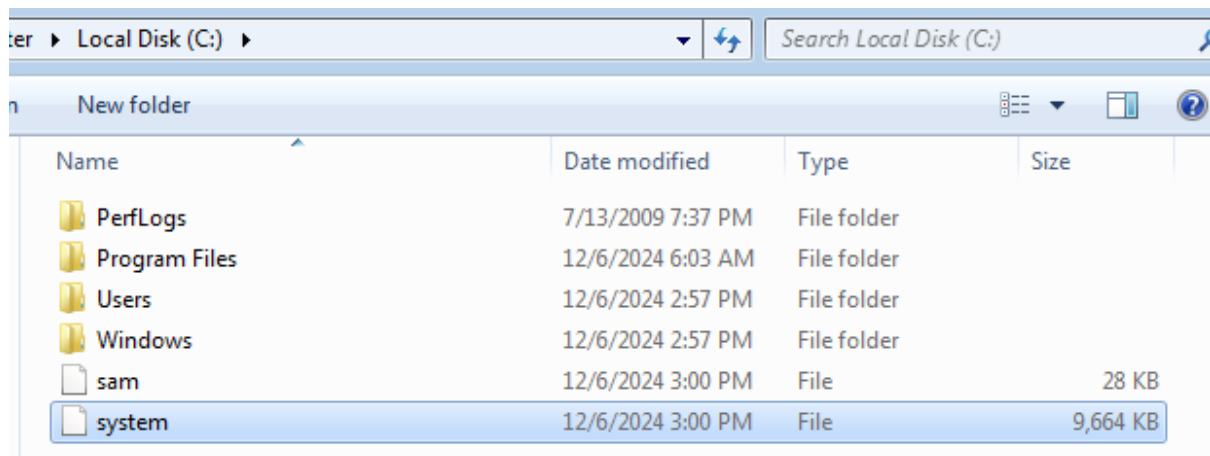


```
C:\Windows\system32>reg save hklm\sam c:\sam
The operation completed successfully.

C:\Windows\system32>reg save hklm\system c:\system
The operation completed successfully.

C:\Windows\system32>
```

On peut voir comme ci-dessous que l'extraction a bien fonctionnée.



Grâce à un dossier partagé, j'ai pu placer les deux fichiers sur ma machine Kali Linux.

Puis j'ai utilisé la commande "samdump" afin de pouvoir extraire les hashes du fichier SAM.

On peut voir qu'il y a bien l'utilisateur "**testuser**" avec son mdp hashé grâce à NTLM.

```
(kali@kali)-[~/Downloads]
$ samdump2 system sam > hashes.txt

(kali@kali)-[~/Downloads]
$ ls
code_1.95.3-1731513102_amd64.deb  hashes.txt  sam  system

(kali@kali)-[~/Downloads]
$ cat hashes.txt
*disabled* Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
*disabled* Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
testuser:1001:aad3b435b51404eeaad3b435b51404ee:a9fdfa038c4b75ebc76dc855dd74f0da:::
HomeGroupUser$:1002:aad3b435b51404eeaad3b435b51404ee:b3101ab404350bf6b5a1f0d5d261a93b:::

(kali@kali)-[~/Downloads]
$
```

Pour cracker le mot de passe, j'ai décidé d'utiliser John The Ripper avec la wordlist rockyou.txt.

En quelques secondes, le mot de passe est cracké : **password123**

```
(kali㉿kali)-[~/Downloads]
$ john --format=NT --wordlist=/usr/share/wordlists/rockyou.txt hashes.txt

Using default input encoding: UTF-8
Loaded 3 password hashes with no different salts (NT [MD4 128/128 SSE2 4x3])
Warning: no OpenMP support for this hash type, consider --fork=2
Press 'q' or Ctrl-C to abort, almost any other key for status
password123      (testuser)
(*disabled* Administrator)
2g 0:00:00:02 DONE (2024-12-06 09:16) 0.9009g/s 6461Kp/s 6461Kc/s 6463KC/s      markinho..*7;Vamos!
Warning: passwords printed above might not be all those cracked
Use the "--show --format=NT" options to display all of the cracked passwords reliably
Session completed.

(kali㉿kali)-[~/Downloads]
$
```

Partie 2 : Stéganographie:

1) Préparation et Introduction

Stéganographie définition: La **stéganographie** est une méthode pour cacher des informations dans des fichiers (images, sons, vidéos) de manière invisible. Elle utilise des espaces inutilisés, comme les bits moins significatifs, pour insérer les données sans altérer le fichier.

Immersion rapide:

Essayons de trouver le message secret "IR4 - COURS CEH Oct Nov 2020"

On peut voir que la commande **zsteg statue.EncodeWith-stylesuxxWeb.png** nous donne le message secret on comprend que dans des images qui semble anodine on peut trouver des secrets cachés dedans

```
maliki@maliki-club:~/Téléchargements/images
Installing ri documentation for iostream-0.2.0
Parsing documentation for zsteg-0.2.13
Installing ri documentation for zsteg-0.2.13
Done installing documentation for rainbow, zpng, iostream, zsteg after 1 seconds
4 gems installed
(maliki@maliki-club) - [~/Téléchargements/images]
$ zsteg statue.EncodeWith-stylesuxxWeb.png

imagedata      .. file: MIPSEB Ucode
1,b,msb,xy     .. file: PDP-11 overlaid pure executable not stripped
1,rgb,lsb,xy   .. text: "IR4 - Cours CEH Oct Nov 2020 "
2,r,msb,xy     .. text: "ADPQEDUQAQP"
2,g,msb,xy     .. text: "TE@@D@PU@"
2,b,msb,xy     .. file: OpenPGP Secret Key
3,rgba,lsb,xy  .. text: "Iy'Iy'Sp"
4,r,msb,xy     .. text: "v0bc6PB!R$p!%TGW"
4,g,msb,xy     .. text: "E&55B\"7`q7GWCAG4W$"
4,b,msb,xy     .. text: "g5*\"bA$7D3`10@ eRpatP"
4,rgb,msb,xy   .. text: " $\"QrddABU4#U4\#$"
4,bgr,msb,xy   .. text: "\"$)RataDBT%3T%3'4"
4,rgba,lsb,xy  .. text: "\r/!o!o2_"
4,abgr,msb,xy  .. text: "FOB0U/30U/3"

(maliki@maliki-club) - [~/Téléchargements/images]
$
```

2) Chaînage de fichiers

Objectif

Cacher une vidéo dans une image et explorer le chaînage pour intégrer plusieurs fichiers dans un seul hôte.

Chaînage

Le chaînage consiste à cacher plusieurs fichiers, comme une vidéo et un texte, dans une même image, avec des mots de passe distincts pour une extraction indépendante.

Pour cacher la vidéo et le texte :

Nous avons utilisé Steghide pour intégrer un fichier compressé contenant la vidéo (**video.mp4**) et un fichier texte (**texte.txt**) dans une image (**image.jpg**). Chaque fichier est protégé par un mot de passe distinct.

La vidéo a été cachée avec la commande :

```
steghide embed -cf image.jpg -ef video.mp4 -p azertyvideo123
```

```
(maliki@maliki-club) - [~/Téléchargements/images]
$ cat texte.txt
message secret
```

```
(maliki@maliki-club) - [~/Téléchargements/images]
$ steghide embed -cf image.jpg -ef video.mp4 -p azertyvideo123
camouflage des données de "video.mp4" dans "image.jpg". terminé.
```

Le texte a ensuite été caché dans la même image avec :

```
steghide embed -cf image.jpg -ef texte.txt -p motdepasse1
```

```
(maliki@maliki-club) - [~/Téléchargements/images]
$ steghide embed -cf image.jpg -ef texte.txt -p motdepasse1
camouflage des données de "texte.txt" dans "image.jpg". terminé.
```

Pour extraire la vidéo et le texte :

À partir de l'image modifiée (**image.jpg**), nous avons récupéré les fichiers cachés en utilisant les mots de passe correspondants.

Pour extraire la vidéo :

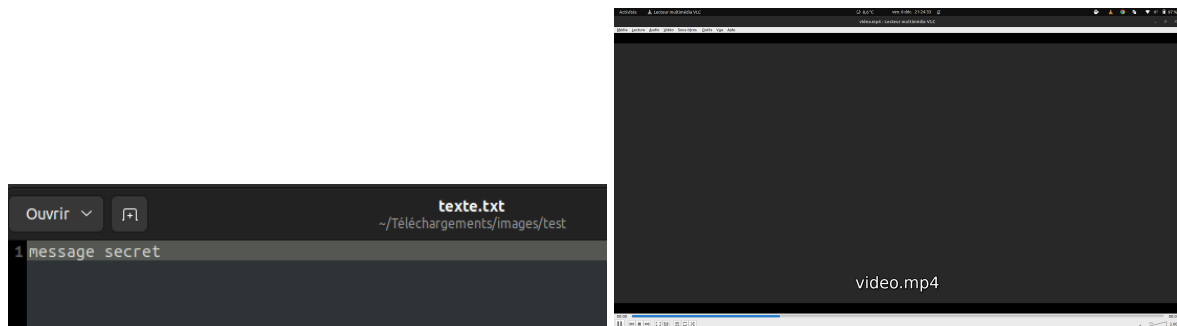
```
steghide extract -sf image.jpg -p azertyvideo123
```

Pour extraire le texte :

```
steghide extract -sf image.jpg -p motdepasse1
```

Les fichiers extraits ont été vérifiés pour garantir leur intégrité.

```
maliki@maliki-club:~/Téléchargements/Images/test
(maliki@maliki-club) - [~/Téléchargements/images/test]
$ ls
image.jpg
(maliki@maliki-club) - [~/Téléchargements/images/test]
$ steghide extract -sf image.jpg -p motdepasse1
écriture des données extraites dans "texte.txt".
(maliki@maliki-club) - [~/Téléchargements/images/test]
$ steghide extract -sf image.jpg -p azertyvideo123
écriture des données extraites dans "video.mp4".
(maliki@maliki-club) - [~/Téléchargements/images/test]
$ ls
image.jpg  texte.txt  video.mp4
```



3) Gestion de Multi-Flux (Alternate Data Streams)

Objectif

Simuler et manipuler des flux multiples sur Linux en attachant des données cachées à un fichier principal.

on va créer un fichier `secret.txt` que l'on va fractionner en 3 parties que l'on va cacher dans les 3 `photo.png` séparément ,

```
(maliki@maliki-club) - [~/Téléchargements/images/test/test]
$ echo "secret" >> secret.txt
(maliki@maliki-club) - [~/Téléchargements/images/test/test]
$ ls
photo1.png photo2.png photo.png secret.txt
(maliki@maliki-club) - [~/Téléchargements/images/test/test]
$ cat secret.txt
secret
(maliki@maliki-club) - [~/Téléchargements/images/test/test]
$
```

On divise le fichier texte en trois parties :

```
split -n 3 -d secret.txt part
```

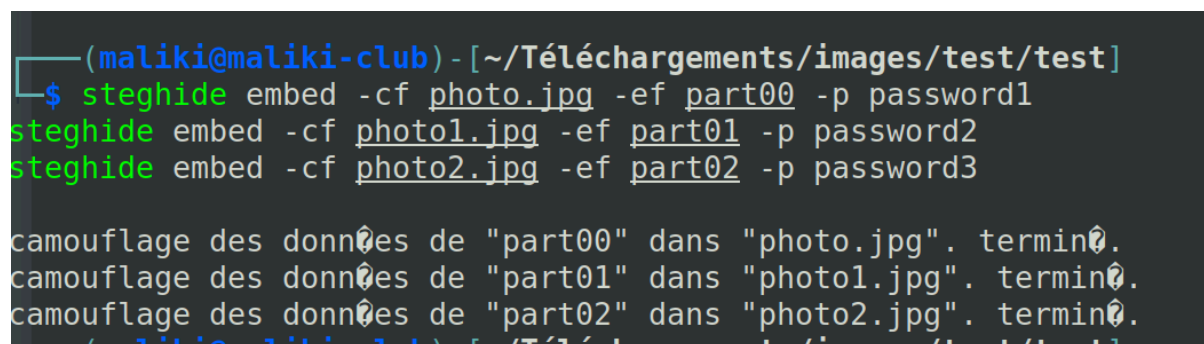
```
(maliki@maliki-club) - [~/Téléchargements/images/test/test]
$ split -n 3 -d secret.txt part
(maliki@maliki-club) - [~/Téléchargements/images/test/test]
$ ls
part00 part01 part02 photo1.png
(maliki@maliki-club) - [~/Téléchargements/images/test/test]
```

On utilise **steghide** pour intégrer chaque partie dans une image :

```
steghide embed -cf photo.jpg -ef part00 -p password1
```

```
steghide embed -cf photo1.jpg -ef part01 -p password2
```

```
steghide embed -cf photo2.jpg -ef part02 -p password3
```



```
(maliki@maliki-club)-[~/Téléchargements/images/test/test]
$ steghide embed -cf photo.jpg -ef part00 -p password1
steghide embed -cf photo1.jpg -ef part01 -p password2
steghide embed -cf photo2.jpg -ef part02 -p password3

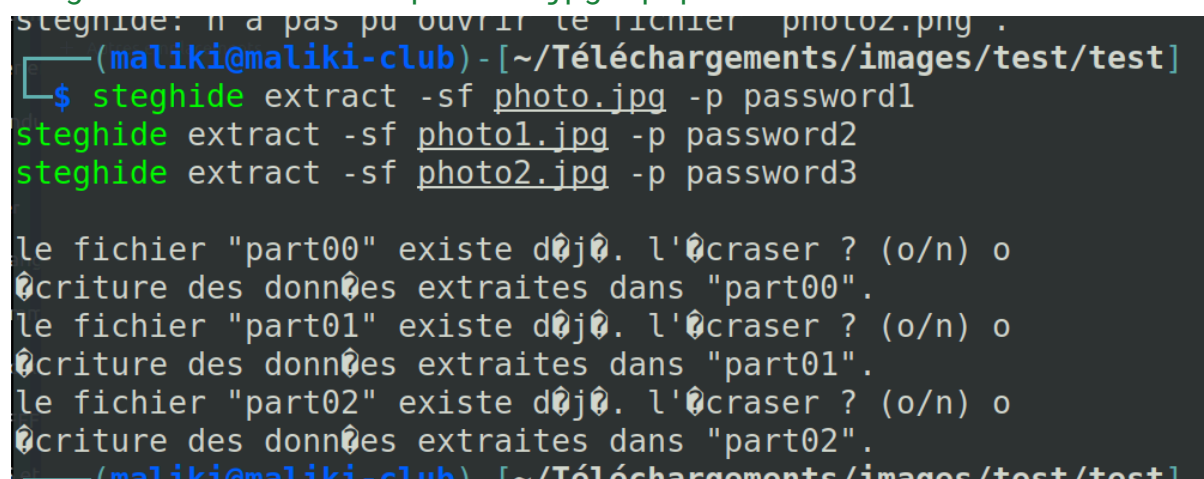
camouflage des données de "part00" dans "photo.jpg". terminé.
camouflage des données de "part01" dans "photo1.jpg". terminé.
camouflage des données de "part02" dans "photo2.jpg". terminé.
(maliki@maliki-club)-[~/Téléchargements/images/test/test]
```

Pour extraire les parties cachées :

```
steghide extract -sf photo.jpg -p password1
```

```
steghide extract -sf photo1.jpg -p password2
```

```
steghide extract -sf photo2.jpg -p password3
```



```
steghide: n'a pas pu ouvrir le fichier "photo2.png".
(maliki@maliki-club)-[~/Téléchargements/images/test/test]
$ steghide extract -sf photo.jpg -p password1
steghide extract -sf photo1.jpg -p password2
steghide extract -sf photo2.jpg -p password3

le fichier "part00" existe déjà. l'écraser ? (o/n) o
écriture des données extraites dans "part00".
le fichier "part01" existe déjà. l'écraser ? (o/n) o
écriture des données extraites dans "part01".
le fichier "part02" existe déjà. l'écraser ? (o/n) o
écriture des données extraites dans "part02".
(maliki@maliki-club)-[~/Téléchargements/images/test/test]
```

Enfin on reconstitue le fichier **secret.txt** à partir des trois parties :

```
(maliki@maliki-club) - [~/Téléchargements/images/test/test]
$ ls
part00 part01 part02 photo1.jpg photo2.jpg photo.jpg secret.txt
(maliki@maliki-club) - [~/Téléchargements/images/test/test]
$ cat part00 part01 part02 > secret_reconstruit.txt
(maliki@maliki-club) - [~/Téléchargements/images/test/test]
$ cat secret_reconstruit.txt
secret
```

Le message a bien été reconstruit à parties de 3 parties
cachés dans 3 images différentes

Conclusion

Dans ce lab on a exploré deux techniques clés en
stéganographie : le **chaînage de fichiers** et la gestion de **flux
multiples (multiflux)**.

- **Chaînage de fichiers** : Nous avons intégré une vidéo et un
texte dans une image unique, avec des mots de passe
distincts pour une extraction sécurisée.
- **Multiflux** : En fragmentant un fichier texte et en
dissimulant les parties dans plusieurs images, nous avons
démontré une méthode renforçant la sécurité en
répartissant les données.

Ces approches montrent la puissance de la stéganographie pour
dissimuler et protéger des informations sensibles tout en
garantissant leur récupération.