Forensics Challenge 2

Challenge | 59 Solves

Floppy offset

25

Forensics

When we seized an old laptop and floppy disks, we identified an odd one. The forensic image didn't match with the original. We can assume that there was some tampering. Also, the label on the floppy disk makes no sense to us. Can you recover the hidden information in the floppy disk?

Unlock Hint for 5 points

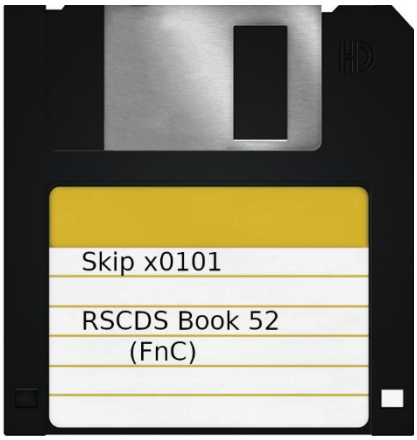
Unlock Hint for 10 points

Unlock Hint for 10 points

floppydisc.png | forensic.img | original.img

Flag | Submit

Answer: We are given 3 items, two .img files and a png. Below is the png.



Using Binwalk let's examine the .img files

```
kali@kali:~/Pictures$ binwalk original.img

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
17408         0x4400          JPEG image data, JFIF standard 1.01

kali@kali:~/Pictures$ binwalk forensic.img

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
17408         0x4400          JPEG image data, JFIF standard 1.01
```

Looks like they both have an image, let's extract and open it up.

```
kali@kali:~/Pictures$ binwalk -e forensic.img

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
17408         0x4400          JPEG image data, JFIF standard 1.01

kali@kali:~/Pictures$ binwalk -e original.img

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
17408         0x4400          JPEG image data, JFIF standard 1.01

kali@kali:~/Pictures$ cd _original.img.extracted/
kali@kali:~/Pictures/_original.img.extracted$ ls
4400
kali@kali:~/Pictures/_original.img.extracted$ cd ..
kali@kali:~/Pictures$ cd _forensic.img.extracted/
kali@kali:~/Pictures/_forensic.img.extracted$ ls
4400
kali@kali:~/Pictures/_forensic.img.extracted$ file *
4400: JPEG image data, JFIF standard 1.01, aspect ratio, density 1
kali@kali:~/Pictures/_forensic.img.extracted$ cd ..
kali@kali:~/Pictures$ cd _original.img.extracted/
kali@kali:~/Pictures/_original.img.extracted$ file *
4400: JPEG image data, JFIF standard 1.01, aspect ratio, density 1
```

So the jpeg is just a meme. Let's look deeper at the top files. Let grab the hex dump for each and compare.

```
kali@kali:~/Pictures$ xxd original.img > original
kali@kali:~/Pictures$ xxd forensic.img > forensic
kali@kali:~/Pictures$ diff original forensic
17,19c17,19
< 00000100: 0000 0000 0000 0000 0000 0000 0000 0000  ................
< 00000110: 0000 0000 0000 0000 0000 0000 0000 0000  ................
< 00000120: 0000 0000 0000 0000 0000 0000 0000 0000  ................
---
> 00000100: 0061 476c 7362 4852 7663 454e 5552 6e74  .aGlsbHRvcENURnt
> 00000110: 6f4d 3349 7a58 7a46 7a58 7a64 6f4d 3139  oM3IzXzFzXzdoM19
> 00000120: 6d62 4452 6e66 513d 3d00 0000 0000 0000  mbDRnfQ==.......
```

That looks like a base64 string. Converting it gives us, Flag: hilltopCTF{h3r3_1s_7h3_fl4g}

```
kali@kali:~/Pictures$ echo aGlsbHRvcENURntoM3IzXzFzXzdoM19mbDRnfQ== | base64 -d
hilltopCTF{h3r3_1s_7h3_fl4g}kali@kali:~/Pictures$
```