Lab 11

Chetan Sahrudhai Kimidi - ckimidi

***Abstract*—Learning about steganography and finding flags/secrets in images, using related tools.**

I. Introduction

In this lab, I learnt what steganography is, how I can use it to encode or decode messages/secrets/flags into media like images, here, PNG files.

II. 1.SECRET

First, I followed the instructions as given in the assignment, and learnt about Dominic's stego toolkit, [1]. I used **"docker pull dominicbreuker/stego-toolkit"** and obtained the image into my VM. Then, I downloaded both 1.secret and 2.secret from the files on Canvas, and loaded/mounted them into a container, while starting/initializing it, as seen in Fig. 1.
Now, as suggested in the assignment hints, I learnt about various tools in the toolkit, such as file, binwalk, strings, exiftool, zsteg, foremost, stegdetect etc. I played with all these tools, as seen in Fig. 2, Fig. 3, Fig. 4 and Fig. 5. The file command pretty much gave me what type of the file 1.secret was, which was a PNG file, its dimensions and the bit color profile. The exiftool command gave me the whole metadata of the file. Binwalk, which I noticed being popularly used in online resources and steganography challenges, surprisingly gave me the output of a PNG image and just a copyright string. I even used the additional parameter of -e, which is for extracting any hidden files, but there was nothing interesting in the output. Strings gave out bunch of characters, and there were no interesting/readable characters. Foremost command turned out a bit interesting, since it gave out a directory called "output". This contained an audit.txt and another directory called "png", which had an image named 00000000.png, so I was pursuing this further. I used diff between both files and there was no difference. I even used zsteg for both, but it gave out nothing. Then, I used identify command, but only to know that there was an incorrect sRGB profile, which I later learnt through the web, was a pretty common and negligible issue in PNG files.
But, then, I used the argument -a, in zsteg, which stands for all checks. Then, I got a particular output called **"b1,r,lsb,YX,prime .. text: "gc7W[*(C" "**. I searched regarding this online and learnt that this indicates there is some bit value manipulations with the red color layer. This can be seen in Fig. 6.
Later, I used a command called stegoveritas, which basically checks the PNG with many tools, bruteforces LSB checks, outputs various transformed images into a directory. As seen in Fig. 7, it gave out some lines like "Found something worth keeping!", and after the whole command finished running, it gave out a new directory called "results". In results, there were a bunch of various transformed images as seen in Fig. 8. I moved these images from the Docker container to my local VM, as seen in Fig. 9 and Fig. 10.
Out of the images, a few actually showed a flag on top of the image, as seen in Fig. 11.
Later, I also found some online tools, which were basically LSB type steganography decoders, which could decode the flag from a given image file. The results can be seen in Fig. 12 and Fig. 13.
The flag present in 1.secret is **w0ah 5uch 53cr3t m355ag3**.
So, I found out the presence of the flag from the 1.secret file, in two ways. One using the stegoveritas command and other by using online web-based steganography tools, both which showed the result i.e. presence of flag, visually in images.

III. 2.SECRET - BONUS

Similar to the 1.secret file, I performed binwalk on this file, and found multiple MySQL ISAM files presence. Putting that aside, I directly used another web-based tool to find if there was any flag hiding in the image, just like the previous task. And luckily, I found a flag, when I used the Inverse(RGB) option in the website.
The flag present in 2.secret is **w3_s3t_up_ambush_0n_l3dg3**.
There is a place circled Ledge, on the map. So, I understood the meaning of the flag in this way.
These can be seen in Fig. 14 and Fig. 15.

IV. Conclusion

To conclude this lab exercise, I learnt steganography, got familiar with Dominic's stego toolkit, found the hidden flags in the given files successfully.

References

[1] DominicBreuker's Stego Toolkit
[2] Docker container files mounting
[3] Copying files from Docker container to local machine
[4] Online tool 1 - Aperisolve
[5] Online tool 2 - StegOnline
[6] LSB Steganography

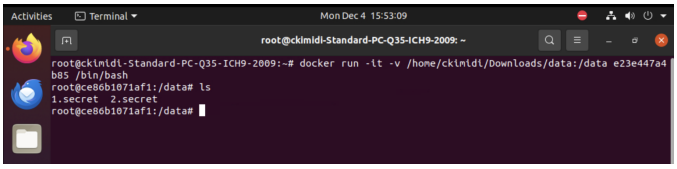

Fig. 1. Container started with data mounted

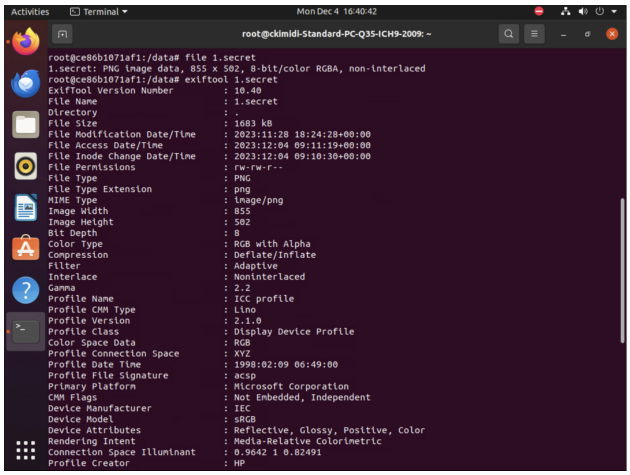

Fig. 2. File, Exiftool

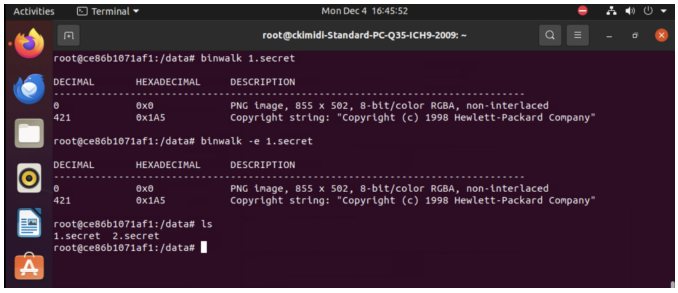

Fig. 3. Binwalk

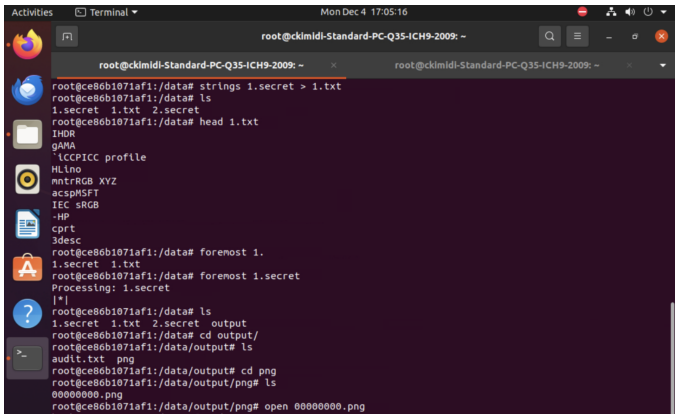

Fig. 4. Strings, Foremost

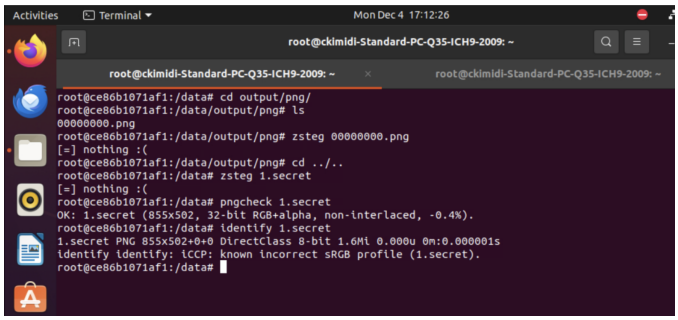

Fig. 5. Zsteg, PNGcheck, Identify

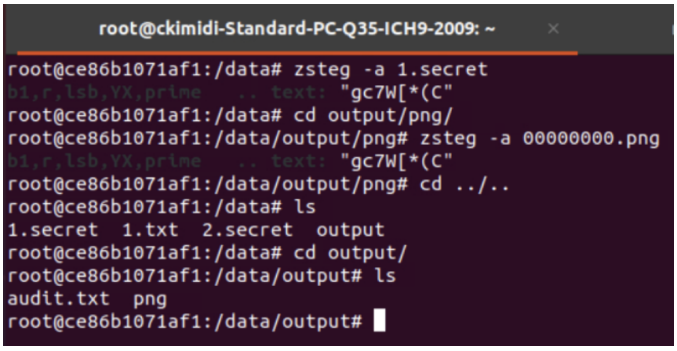

Fig. 6. zsteg -a output

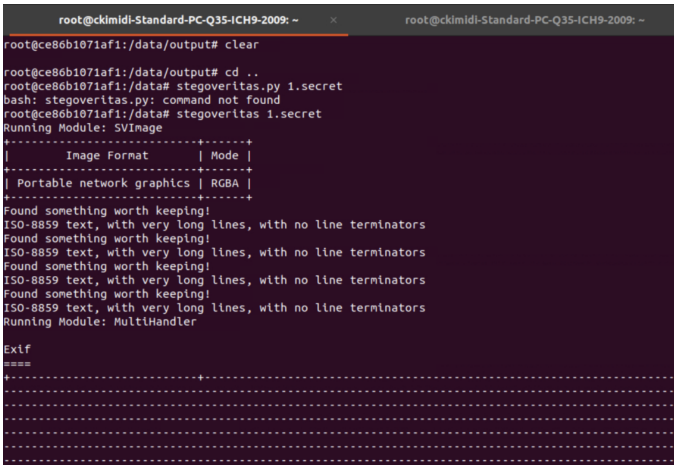

Fig. 7. Stegoveritas output

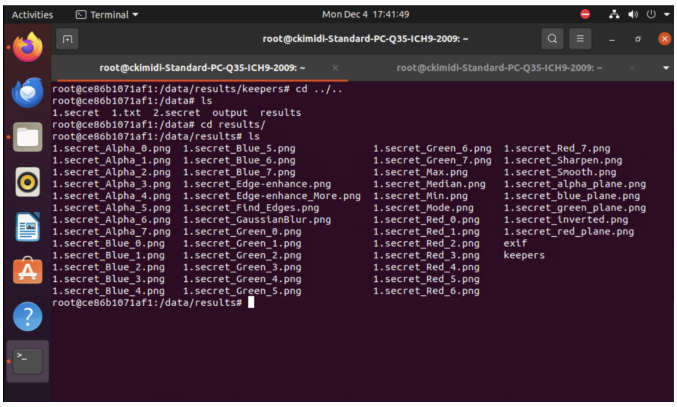

Fig. 8. Results directory

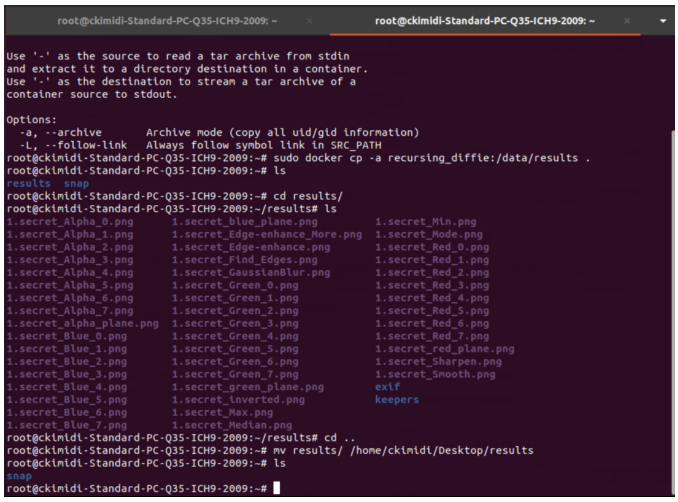

Fig. 9. Copying images from container to my VM

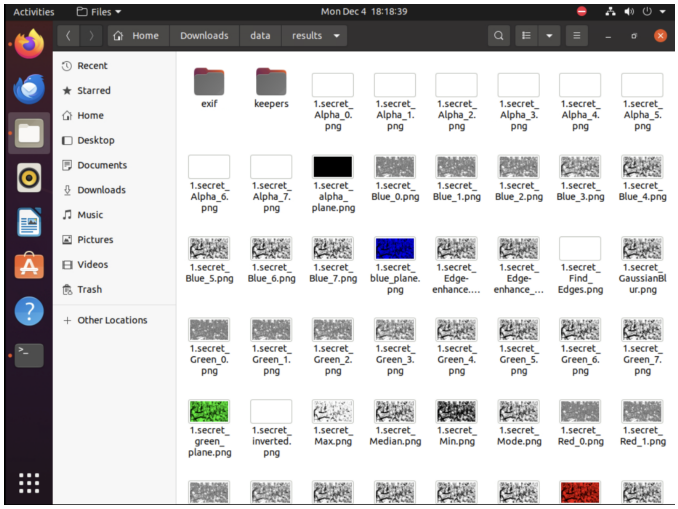

Fig. 10. List of images

Fig. 11. FLAG!!!

Fig. 12. Online tool - Aperisolve

Fig. 13. FLAG VISIBLE

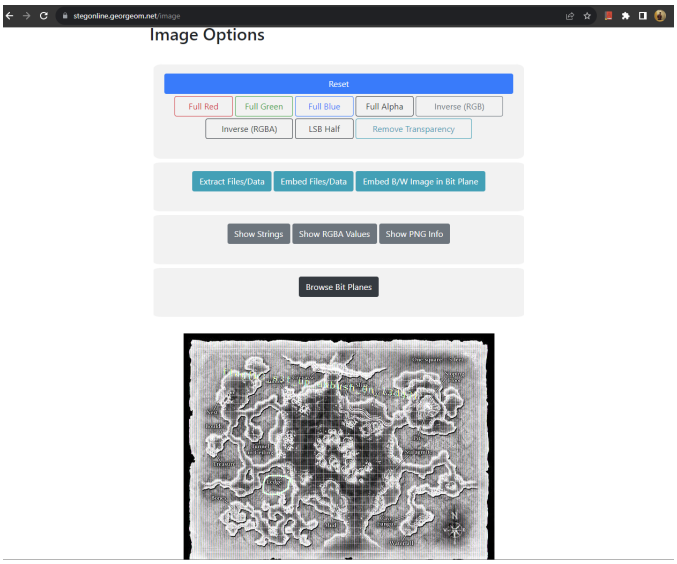

Fig. 14. Online tool - StegOnline by GeorgeOm

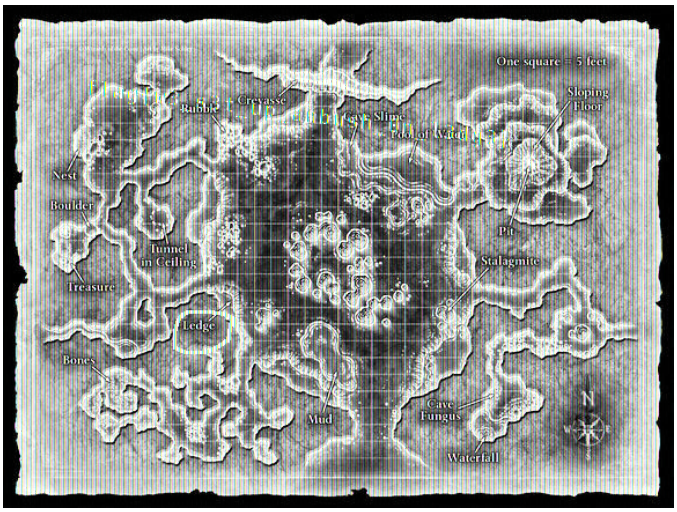

Fig. 15. FLAG VISIBLE