

Recovering secret information from images

Objective

The goal of this inquiry is to retrieve private credentials from two files: Monalisa.jpg and canvas.txt. This task deciphers a hidden account number and password using metadata analysis, binary file inspection, reverse engineering, and steganography detection.

Tools Used

- **ExifTool** – Extract metadata from image files.
- **Binwalk** – Identify embedded files.
- **Stegbreak + jpseek** – Detect and extract JPHide embedded data.
- **Python** – Run a script to reverse binary content.
- **zsteg** – Analyze LSB steganography in PNG images.
- **Wine** – Execute Windows-based steganography tools.

Investigation Process: Commands and Summary

Step 1: Metadata and Hidden File Discovery

We started by inspecting Monalisa.jpg with ExifTool and Binwalk. ExifTool validated the image's structure, and Binwalk discovered a hidden JPEG file.

Commands used: **cd /media/sf_guri**
exiftool Monalisa.jpg binwalk
Monalisa.jpg

```

Ubuntu-18.04---SIT282 1 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal
user@Ubuntu1804: /media/sf_gurl
File Edit View Search Terminal Help
user@Ubuntu1804:~$ cd /media/sf_gurl
user@Ubuntu1804:/media/sf_gurl$ exiftool Monalisa.jpg
ExifTool Version Number      : 10.80
File Name                    : Monalisa.jpg
Directory                   : .
File Size                    : 834 kB
File Modification Date/Time   : 2025:05:08 11:29:19+10:00
File Access Date/Time        : 2025:05:08 14:19:37+10:00
File Inode Change Date/Time   : 2025:05:08 13:03:26+10:00
File Permissions             : rwxrwx---
File Type                    : JPEG
File Type Extension          : jpg
MIME Type                    : image/jpeg
JFIF Version                 : 1.01
Resolution Unit              : None
X Resolution                  : 1
Y Resolution                  : 1
Image Width                  : 1200
Image Height                  : 1788
Encoding Process              : Baseline DCT, Huffman coding
Bits Per Sample              : 8
Color Components              : 3
Y Cb Cr Sub Sampling         : YCbCr4:2:0 (2 2)
Image Size                   : 1200x1788
Megapixels                   : 2.1
user@Ubuntu1804:/media/sf_gurl$ binwalk Monaliksa.jpg
General Error: Cannot open file : [Errno 2] No such file or directory: 'Monaliksa.jpg'
user@Ubuntu1804:/media/sf_gurl$ binwalk Monalisa.jpg
DECIMAL      HEXADECIMAL    DESCRIPTION
-----
0            0x00      JPEG image data, JFIF standard 1.01
user@Ubuntu1804:/media/sf_gurl$

```

Summary: This step revealed embedded content within the original image, indicating the first layer of steganography.

Step 2: Password Brute-force and Extraction using Wine

We used stegbreak with a dictionary to crack the stego image, successfully identifying it was created using JPHide. With the correct wordlist, jpseek was used to extract the hidden content to a file named hidden.txt.

Commands used: **wine ~/Desktop/win-tools/jphide\ and\ Stegbreak/stegdetect/stegbreak.exe -r rules.ini -f words Monalisa.jpg**

wine ~/Desktop/win-tools/jphide\ and\ Stegbreak/jphswin/jpseek.exe Monalisa.jpg hidden.txt

```

user@Ubuntu1804:/media/sf_gurl$ wine ~/Desktop/win-tools/jphide\ and\ Stegbreak/stegdetect/stegbreak.exe -r rules.ini -f words Monalisa.jpg
001c:err:module:import_dll Library MSVCR100_CLR0400.dll (which is needed by L"C:\windows\Microsoft.NET\Framework64\v4.0.30319\mscorlib.exe") not found
001c:err:module:import_dll Library mscorcore.dll (which is needed by L"C:\windows\Microsoft.NET\Framework64\v4.0.30319\mscorlib.exe") not found
001c:err:module:attach_dlls Importing dlls for L"C:\windows\Microsoft.NET\Framework64\v4.0.30319\mscorlib.exe" failed, status c0000135
000f:err:service:process_send_command service protocol error - failed to write pipe!
Corrupt JPEG data: bad Huffman code loaded 1 files...Monalisa.jpg : jphide[v5](tool)Processed 1 files, found 1 embeddings.Time: 6 seconds: Cracks: 52011, 8668.5 c
/suser@Ubuntu1804:/media/sf_gurl$ wine ~/Desktop/win-tools/jphide\ and\ Stegbreak/jphswin/jpseek.exe Monalisa.jpg hidden.txt
wine: cannot find '/home/user/Desktop/win-tools/jphide\ and\ Stegbreak/jphswin/jpseek.exe'
user@Ubuntu1804:/media/sf_gurl$ wine ~/Desktop/win-tools/jphide\ and\ Stegbreak/jphswin/jpseek.exe Monalisa.jpg hidden.txt
wine: cannot find '/home/user/Desktop/win-tools/jphide\ and\ Stegbreak/jphswin/jpseek.exe'
user@Ubuntu1804:/media/sf_gurl$ wine ~/Desktop/win-tools/jphide\ and\ Stegbreak/jpseek.exe Monalisa.jpg hidden.txt
001b:err:module:import_dll Library MSVCR100_CLR0400.dll (which is needed by L"C:\windows\Microsoft.NET\Framework64\v4.0.30319\mscorlib.exe") not found
001b:err:module:attach_dlls Importing dlls for L"C:\windows\Microsoft.NET\Framework64\v4.0.30319\mscorlib.exe" failed, status c0000135
000f:err:service:process_send_command service protocol error - failed to write pipe!

Welcome to jpseek Rev 0.51
(c) 1998 Allan Latham <alathan@flexsys-group.com>
This program is freeware.
No charge is made for its use.
Use at your own risk. No liability accepted whatever happens.
Contains cryptography which may be subject to local laws.

Passphrase:
user@Ubuntu1804:/media/sf_gurl$ ls
canvas.txt hidden.txt Monalisa.jpg rules.ini words

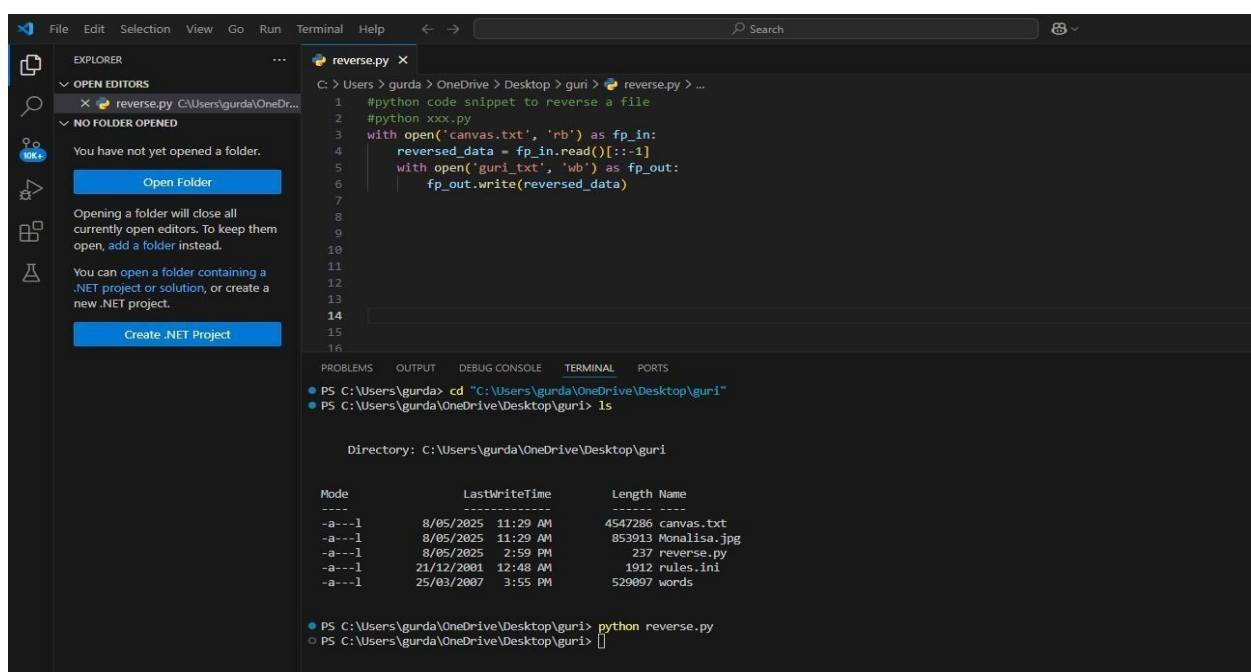
```

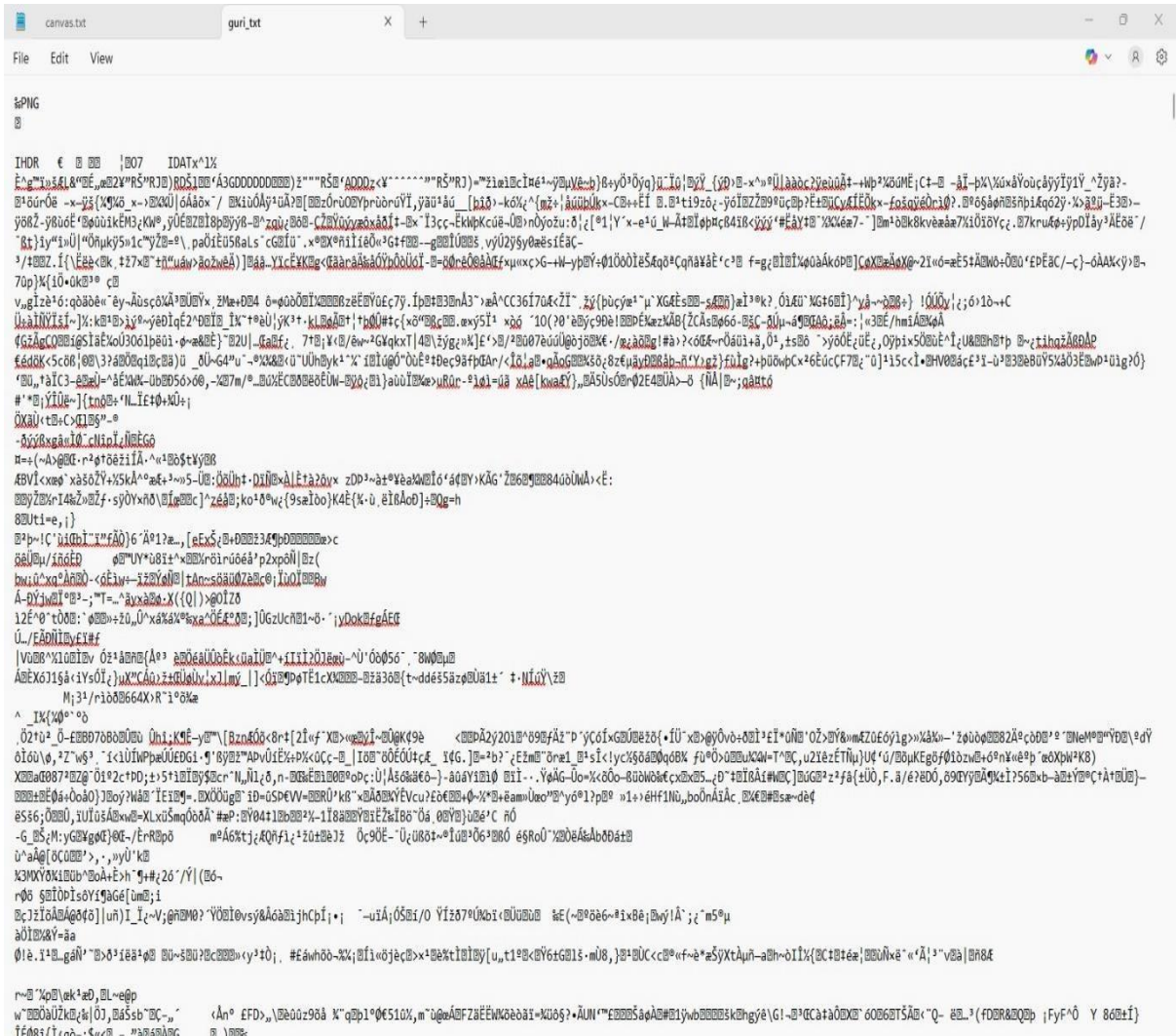


Summary: This phase successfully extracted a hidden file after determining the correct embedding tool and brute-forcing the passphrase.

Step 3: Reversing a File With Python

Opening hidden.txt yielded a Python script. This code was used to reverse the content of canvas.txt, resulting in an output entitled guri_txt, which was subsequently renamed to guri.png once it was verified to be an image file.





Summary: This phase revealed that canvas.txt was a binary-reversed PNG file, and reversing it revealed the original stego image, guri.txt.

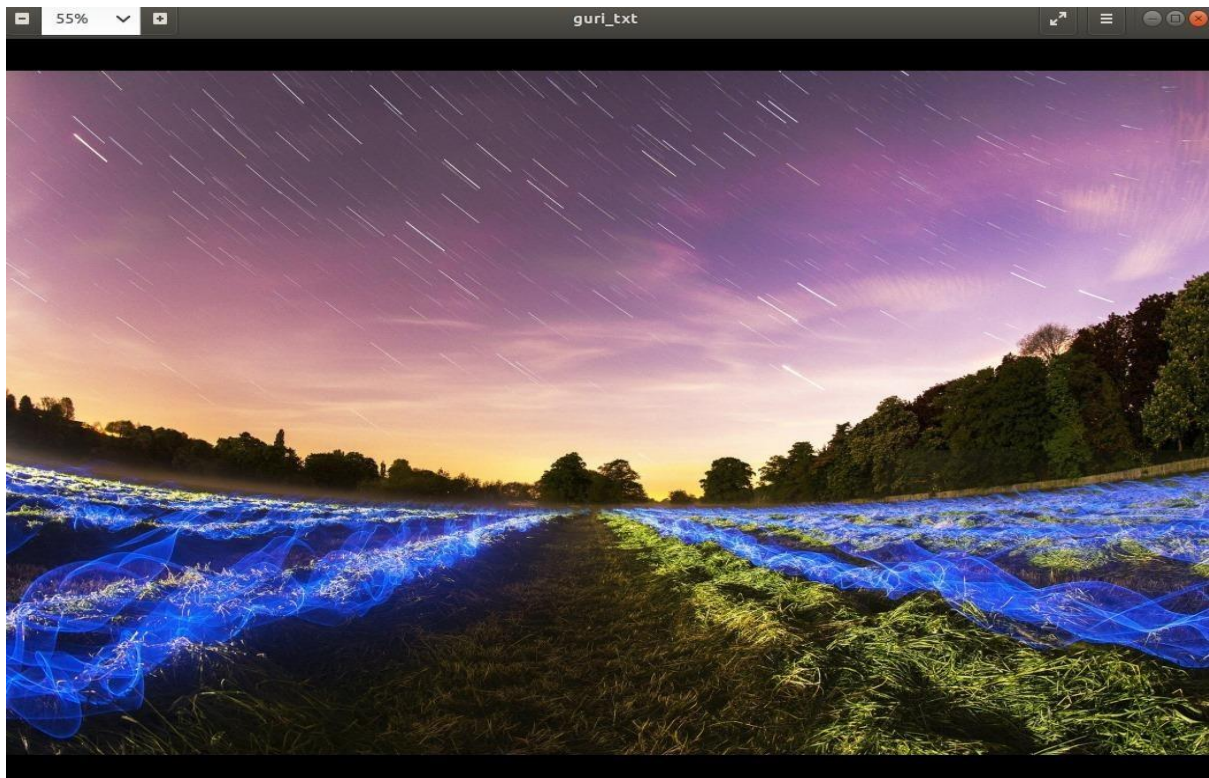
Step 4: LSB extraction using zsteg.

Since guri.txt was actually a png file that was converted to guri.png and as zsteg requires Ruby 2.7.8, we first installed and setup the appropriate Ruby environment. After installing zsteg, we performed the analysis on the PNG file. Commands used: **rbenv install**

2.7.8. rbenv global version

2.7.8 rbenv rehash gem

install zsteg zsteg guri.png



The concealed data was successfully recovered from the red channel's LSB, revealing the complete credentials.

Account:01112345678

Password:DeakinSIT282/703

```

user@Ubuntu1804: /media/sf_guri$ ./configure --prefix=$HOME/.rbenv/versions/2.7.8 --enable-shared --with-ext=openssl,psych,+
user@Ubuntu1804: /media/sf_guri$ make -j 1
user@Ubuntu1804: /media/sf_guri$ make install
==> Installed ruby-2.7.8 to /home/user/.rbenv/versions/2.7.8

NOTE: to activate this Ruby version as the new default, run: rbenv global 2.7.8
user@Ubuntu1804: /media/sf_guri$ ruby -v
ruby 2.5.1p57 (2018-03-29 revision 63029) [x86_64-linux-gnu]
user@Ubuntu1804: /media/sf_guri$ rbenv rehash
user@Ubuntu1804: /media/sf_guri$ ruby -v
ruby 2.5.1p57 (2018-03-29 revision 63029) [x86_64-linux-gnu]
user@Ubuntu1804: /media/sf_guri$ rbenv global 2.7.8
user@Ubuntu1804: /media/sf_guri$ rbenv rehash
user@Ubuntu1804: /media/sf_guri$ ruby -v
ruby 2.7.8p225 (2023-03-30 revision 1f4d455849) [x86_64-linux]
user@Ubuntu1804: /media/sf_guri$ gem install zsteg
Fetching rainbow-3.1.1.gem
Fetching zsteg-0.2.13.gem
Fetching png-0.4.5.gem
Fetching iostream-0.5.0.gem
Successfully installed iostream-0.5.0
Successfully installed rainbow-3.1.1
Successfully installed png-0.4.5
Successfully installed zsteg-0.2.13
Parsing documentation for iostream-0.5.0
Installing ri documentation for iostream-0.5.0
Parsing documentation for rainbow-3.1.1
Installing ri documentation for rainbow-3.1.1
Parsing documentation for png-0.4.5
Installing ri documentation for png-0.4.5
Parsing documentation for zsteg-0.2.13
Installing ri documentation for zsteg-0.2.13
Done installing documentation for iostream, rainbow, png, zsteg after 2 seconds
4 gems installed
user@Ubuntu1804: /media/sf_guri$ zsteg guri.png
b1,r,lsb,xy .. text: "7JFcf2w3s"
b1,rgb,lsb,xy .. text: "Account:0112345678\nPassword:DeakinSIT282/703"
b2,bgr,lsb,xy .. text: "r3wc3_Q1wl\n"
b2,bgr,msb,xy .. text: "UQ@EDTTP"
b4,r,msb,xy .. text: "5FDPAWkq"
b4,g,msb,xy .. text: "twU\`2IDP`GBgSu"
b4,b,msb,xy .. text: "GqD`UQ!6%"
b4,rgb,msb,xy .. text: "Gu4GVguSA"
b4,bgr,msb,xy .. text: "eWEWAp14A"
user@Ubuntu1804: /media/sf_guri$

```

Summary: *The final step proved that sensitive credentials were concealed in the least significant sections of the PNG picture using steganography.*

Final Recovered Credentials

- **Account Number:** 01112345678
- **Password:** DeakinSIT282/703

Conclusion

This forensic inquiry highlighted the complexities of multi-layered steganographic procedures. Data was hidden via JPEG embedding, a reversible Python-obfuscated file, and then LSB-encoded information within a PNG image. The concealed credentials were successfully recovered using the appropriate tools and a systematic approach. The investigation shows how file inspection, reverse engineering, and steganography detection techniques can be combined to recover digital evidence.

References

- <https://github.com/zed-0xff/zsteg>
- <https://steghide.sourceforge.net/>
- <https://exiftool.org/>
- <https://manpages.ubuntu.com/binwalk>
- <https://github.com/rbenv/rbenv>