STEGANOGRAPHY

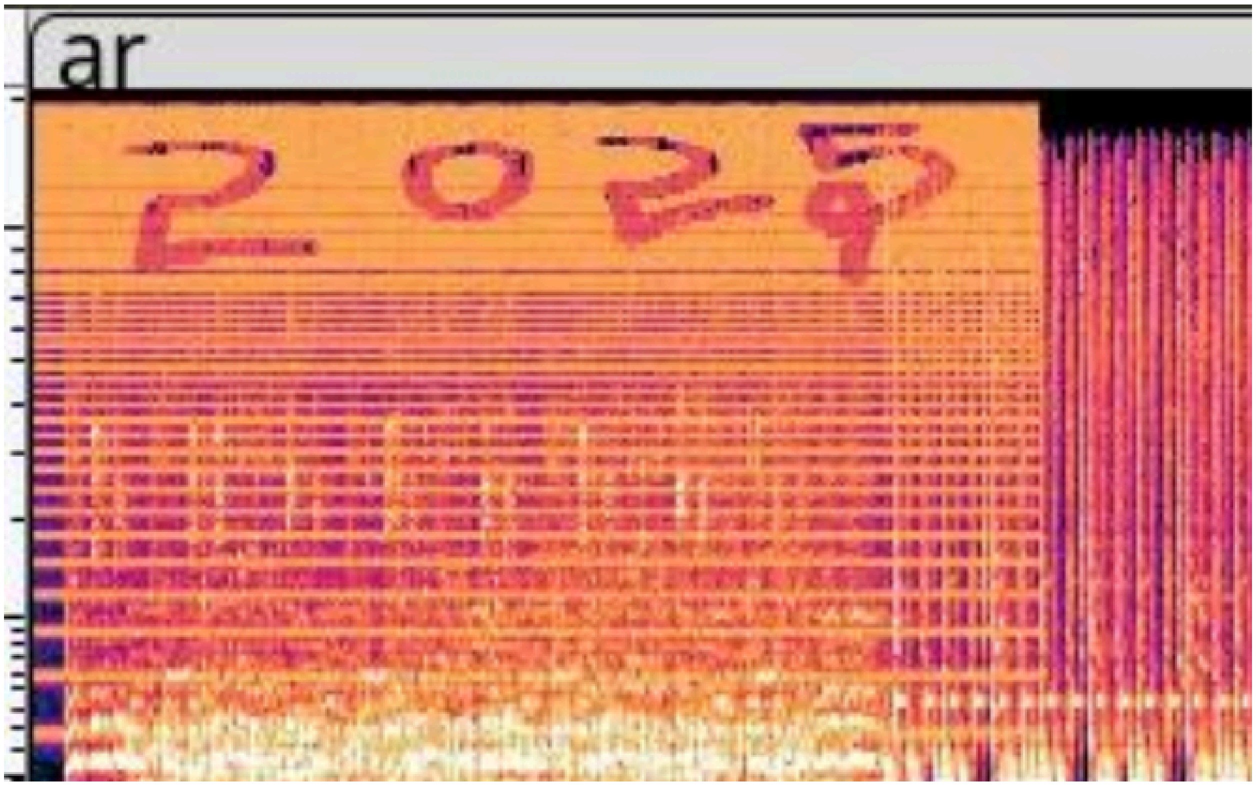
ECHO OF TIME

Description:

An audio file named ab within this audio lies a crucial piece of information: a year that marks a significant event.

Approach:

- The audio file was downloaded and loaded into wavacity website(alternative to audacity tool)
- Then,changed the mode to steganography mode and edited the gain, range and frequencies in a way so that i get a visual representation of the audio file
- Got the hidden year 2025 which was embedded in the file
- And that was the flag needed



Flag: r00t@localhost{2025}

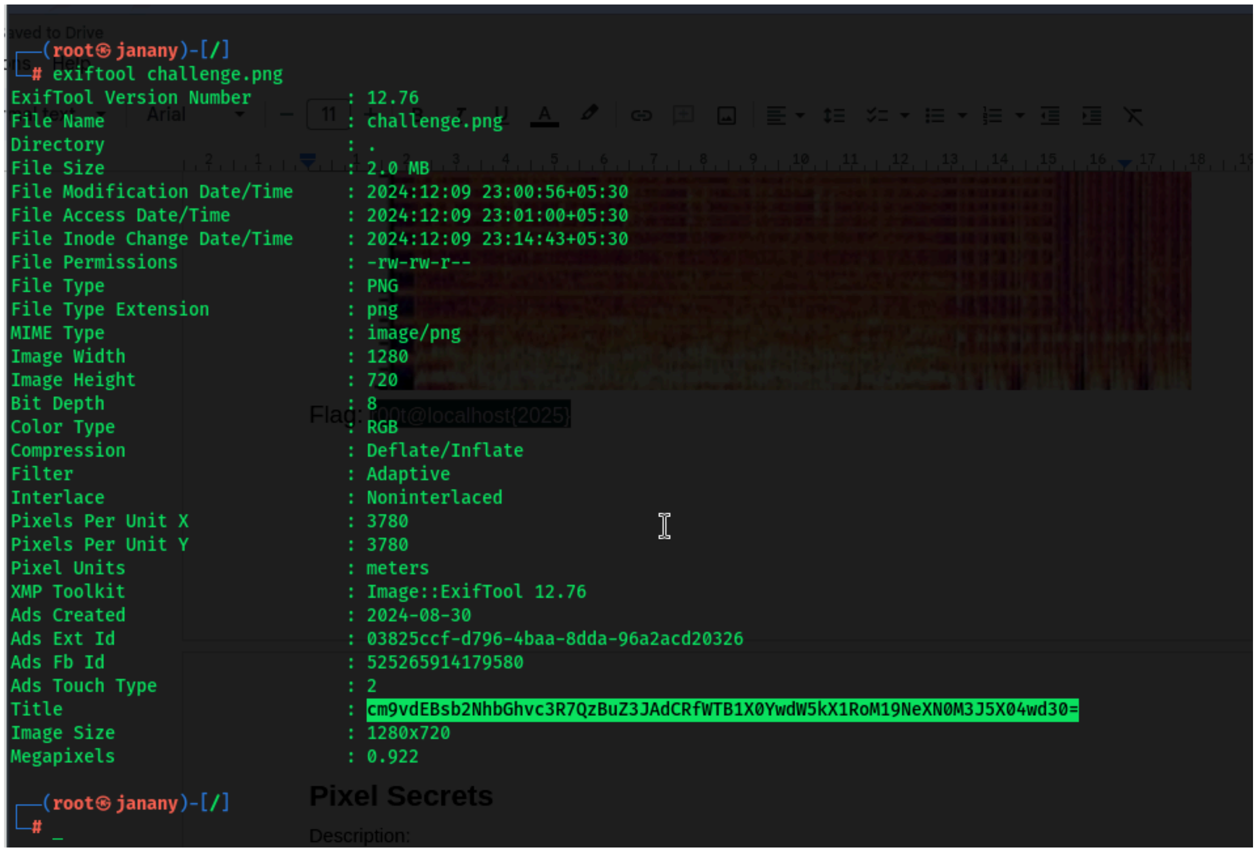
Description:

Hidden Truth

A hidden message lies concealed within a jumble of characters and numbers. Can you crack the code and reveal the secret? The mystery is waiting for you to uncover it.

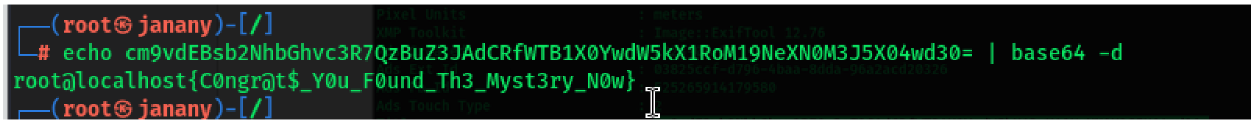
Solution:

- Its a pretty straight forward challenge, just a **binwalk** solved the problem.



```
┌──(root㉿janany)-[/]
└─# exiftool challenge.png
ExifTool Version Number         : 12.76
File Name                       : challenge.png
Directory                       : .
File Size                       : 2.0 MB
File Modification Date/Time     : 2024:12:09 23:00:56+05:30
File Access Date/Time           : 2024:12:09 23:01:00+05:30
File Inode Change Date/Time     : 2024:12:09 23:14:43+05:30
File Permissions                : -rw-rw-r--
File Type                       : PNG
File Type Extension             : png
MIME Type                       : image/png
Image Width                     : 1280
Image Height                    : 720
Bit Depth                       : 8
Color Type                      : RGB
Compression                     : Deflate/Inflate
Filter                          : Adaptive
Interlace                       : Noninterlaced
Pixels Per Unit X               : 3780
Pixels Per Unit Y               : 3780
Pixel Units                     : meters
XMP Toolkit                     : Image::ExifTool 12.76
Ads Created                     : 2024-08-30
Ads Ext Id                      : 03825ccf-d796-4baa-8dda-96a2acd20326
Ads Fb Id                       : 525265914179580
Ads Touch Type                  : 2
Title                           : cm9vdEBsb2NhbGhvc3R7QzBuZ3JAdCRfWTB1X0YwdW5kX1RoM19NeXN0M3J5X04wd30=
Image Size                      : 1280x720
Megapixels                      : 0.922

┌──(root㉿janany)-[/]
└─# _
```

- Decoding the base64 string gives us the flag.



```
┌──(root㉿janany)-[/]
└─# echo cm9vdEBsb2NhbGhvc3R7QzBuZ3JAdCRfWTB1X0YwdW5kX1RoM19NeXN0M3J5X04wd30= | base64 -d
root@localhost{C0ngr@t$_Y0u_F0und_Th3_Myst3ry_N0w}
┌──(root㉿janany)-[/]
```

Pixel Secrets

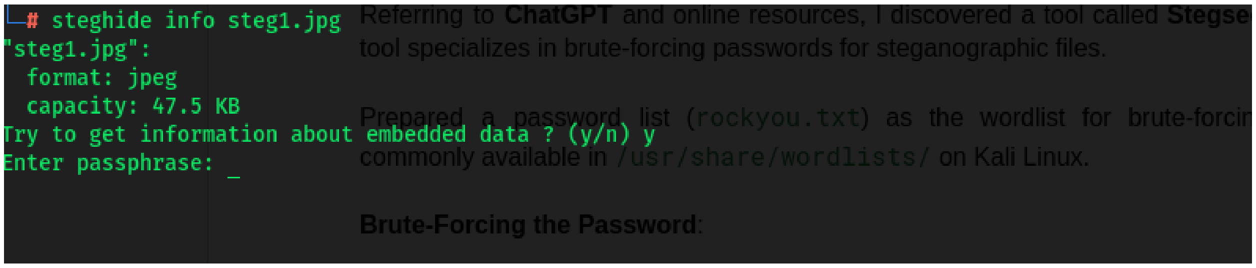
Description:

Decode the hidden message embedded in this image. Use steganographic techniques to uncover the flag that lies beneath the pixels!

Solution:

Initial Analysis with Steghide:

- First, I attempted to analyze the file using `steghide`

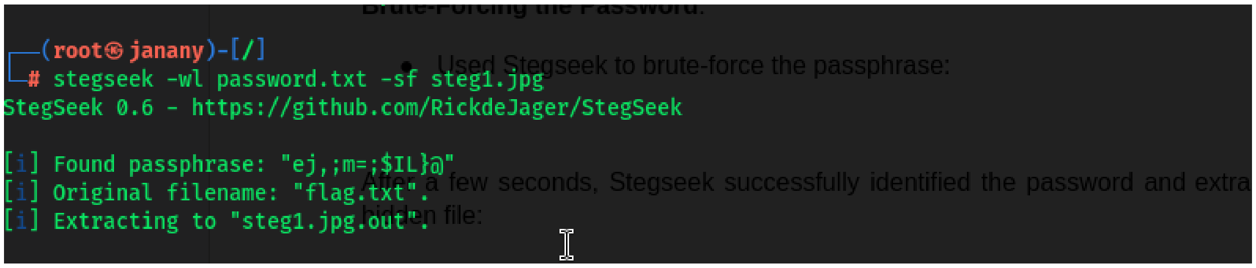


```
└─# steghide info steg1.jpg
"steg1.jpg":
  format: jpeg
  capacity: 47.5 KB
Try to get information about embedded data ? (y/n) y
Enter passphrase: _
```

-
- The output confirmed the presence of embedded data, but extracting it required a passphrase:
-
- Referring to **ChatGPT** and online resources, I discovered a tool called **Stegseek**. This tool specializes in brute-forcing passwords for steganographic files.
-
- Prepared a password list (`password.txt`) as the wordlist for brute-forcing.(given in the challenge)

Brute-Forcing the Password:

- Used Stegseek to brute-force the passphrase:

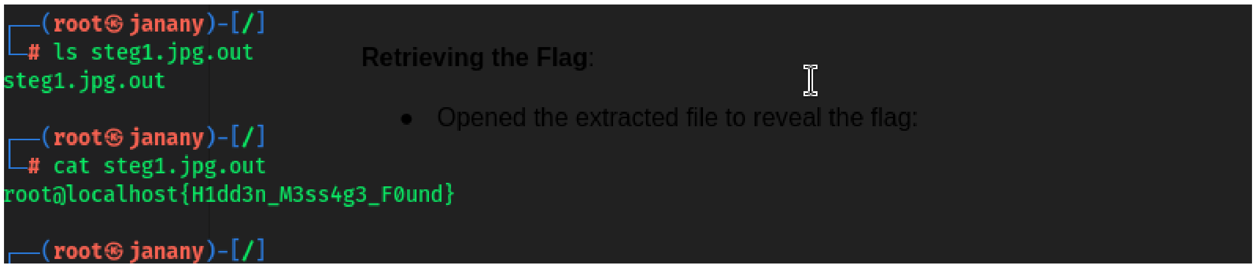


```
┌──(root㉿janany)-[/]
└─# stegseek -wl password.txt -sf steg1.jpg
StegSeek 0.6 - https://github.com/RickdeJager/StegSeek

[i] Found passphrase: "ej,;m=;$IL}@"
[i] Original filename: "flag.txt".
[i] Extracting to "steg1.jpg.out".
```

After a few seconds, Stegseek successfully identified the password and extracted the hidden file:

Retrieving the Flag:

- Opened the extracted file to reveal the flag:



```
┌──(root㉿janany)-[/]
└─# ls steg1.jpg.out
steg1.jpg.out

┌──(root㉿janany)-[/]
└─# cat steg1.jpg.out
root@localhost{H1dd3n_M3ss4g3_F0und}

┌──(root㉿janany)-[/]
```

Retrieving the Flag:

- Opened the extracted file to reveal the flag:

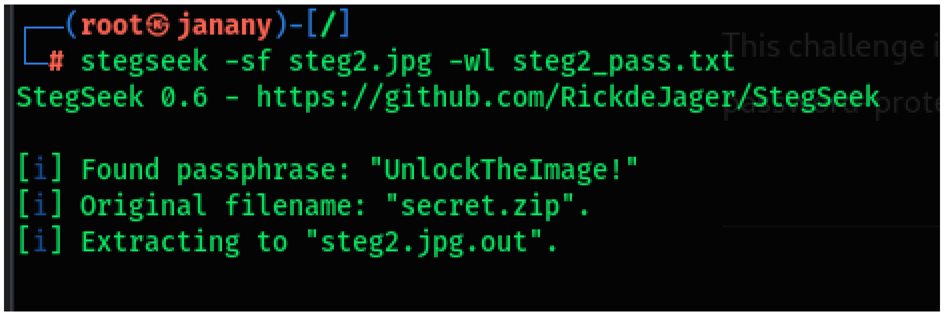
Secret Stash

Description:
n a charming old bookstore, an artist's illustration graces the cover of a vintage volume. The artwork seems like a beautiful enigma, with intricate details and hidden symbols. Among the various elements, one particular design element holds a clue that leads to a hidden archive within the book. The true prize, a coveted flag, rests safely inside a concealed digital treasure. To uncover the secret, examine the image closely and uncover the secret passage to the zip file within.

Solution:
Brute-Forcing the Image File:

- **Analyzing the File**: Used `steghide` to check for embedded data:

Brute-Forcing the Password: Leveraged **Stegseek** with the `given` password list to crack the password and extract the hidden ZIP file:

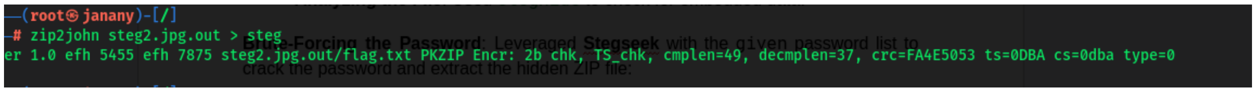


```
┌──(root㉿janany)-[/]
└─# stegseek -sf steg2.jpg -wl steg2_pass.txt
StegSeek 0.6 - https://github.com/RickdeJager/StegSeek

[i] Found passphrase: "UnlockTheImage!"
[i] Original filename: "secret.zip".
[i] Extracting to "steg2.jpg.out".
```

- **Output**: The password was successfully cracked, and a secret ZIP file was extracted.

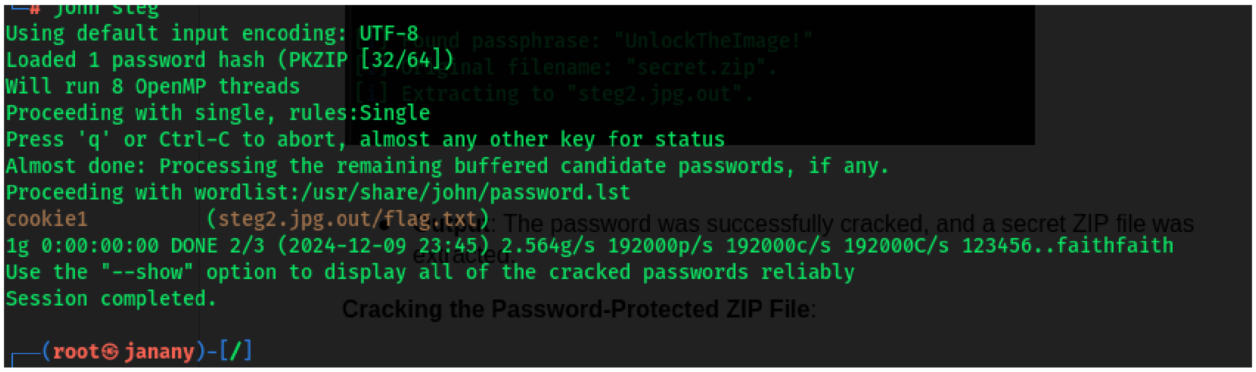
Cracking the Password-Protected ZIP File:

- **Preparing for Cracking**: Used `zip2john` to generate a hash of the ZIP file for cracking with **John the Ripper**:



```
──(root㉿janany)-[/]
└─# zip2john steg2.jpg.out > steg
er 1.0 efh 5455 efh 7875 steg2.jpg.out/flag.txt PKZIP Encr: 2b chk, TS_chk, cmplen=49, decmplen=37, crc=FA4E5053 ts=0DBA cs=0dba type=0
```

Brute-Forcing the ZIP Password: Used John the Ripper to crack the ZIP file's password:



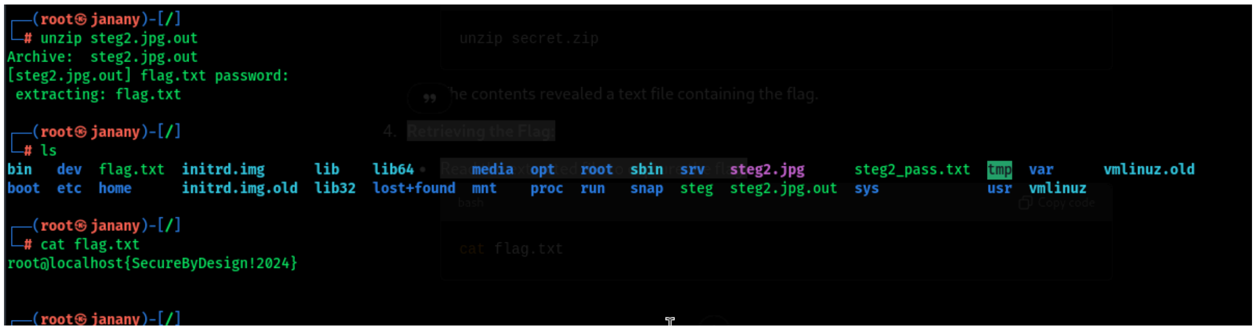
```
└─# john steg
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
Will run 8 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
cookie1          (steg2.jpg.out/flag.txt)
1g 0:00:00:00 DONE 2/3 (2024-12-09 23:45) 2.564g/s 192000p/s 192000c/s 192000C/s 123456..faithfaith
Use the "--show" option to display all of the cracked passwords reliably
Session completed.

┌──(root㉿janany)-[/]
```

Extracting the ZIP Contents:

- Unzipped the file using the cracked password : cookie1

Retrieving the Flag:

- Read the extracted file to capture the flag:



```
┌──(root㉿janany)-[/]
└─# unzip steg2.jpg.out
Archive:  steg2.jpg.out
[steg2.jpg.out] flag.txt password:
 extracting: flag.txt

┌──(root㉿janany)-[/]
└─# ls
bin   dev  flag.txt  initrd.img      lib    lib64       media  opt   root  sbin  srv   steg2.jpg      steg2_pass.txt  tmp  var      vmlinuz.old
boot  etc  home      initrd.img.old  lib32  lost+found  mnt    proc  run   snap  steg  steg2.jpg.out  sys             usr  vmlinuz

┌──(root㉿janany)-[/]
└─# cat flag.txt
root@localhost{SecureByDesign!2024}


┌──(root㉿janany)-[/]
```