> TRICKY COMMUNICATIONS

BY: ES

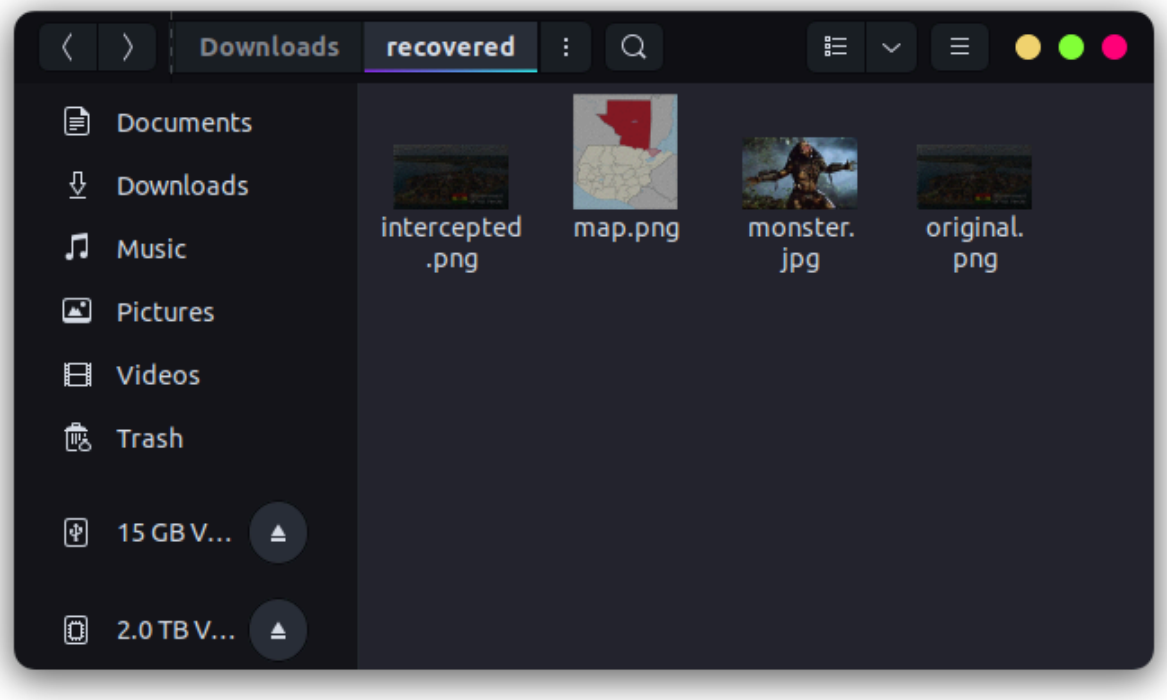
DIFFICULTY: EASY
CATEGORY: STEGO
SOLVES: 0 (SORRY)

Description: Some images were intercepted in communication between potential bad actors. Figure out what they were talking about. Flag format is a btc address.

Artifact: recovered.zip

SOLUTION:

Decompress the archive and we see 4 images:

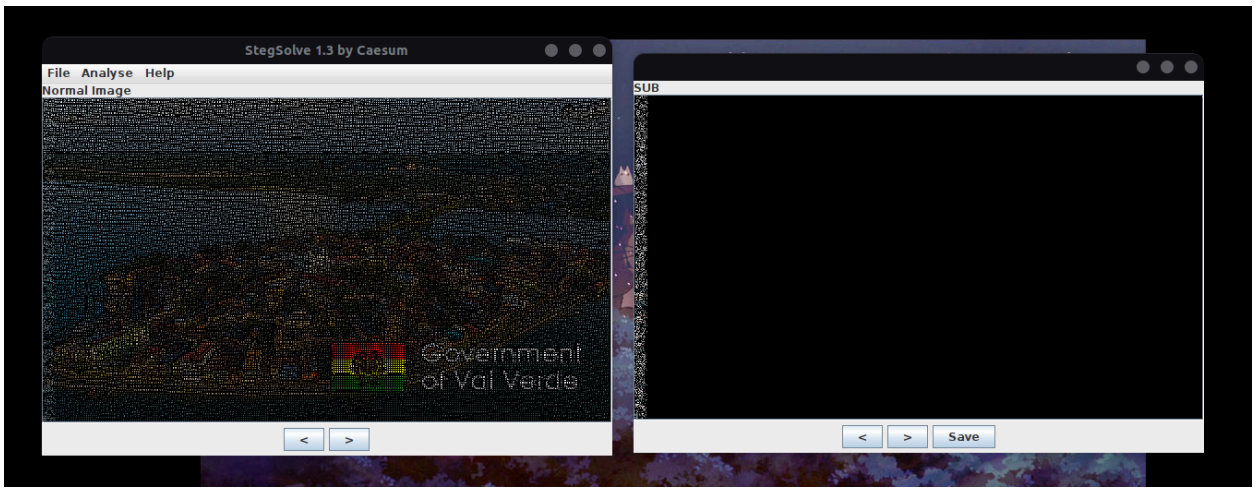


What I would typically do is try and run tools like foremost, steghide, binwalk, and other stego tools on the images in the archive, and on not finding anything delve a little bit deeper.

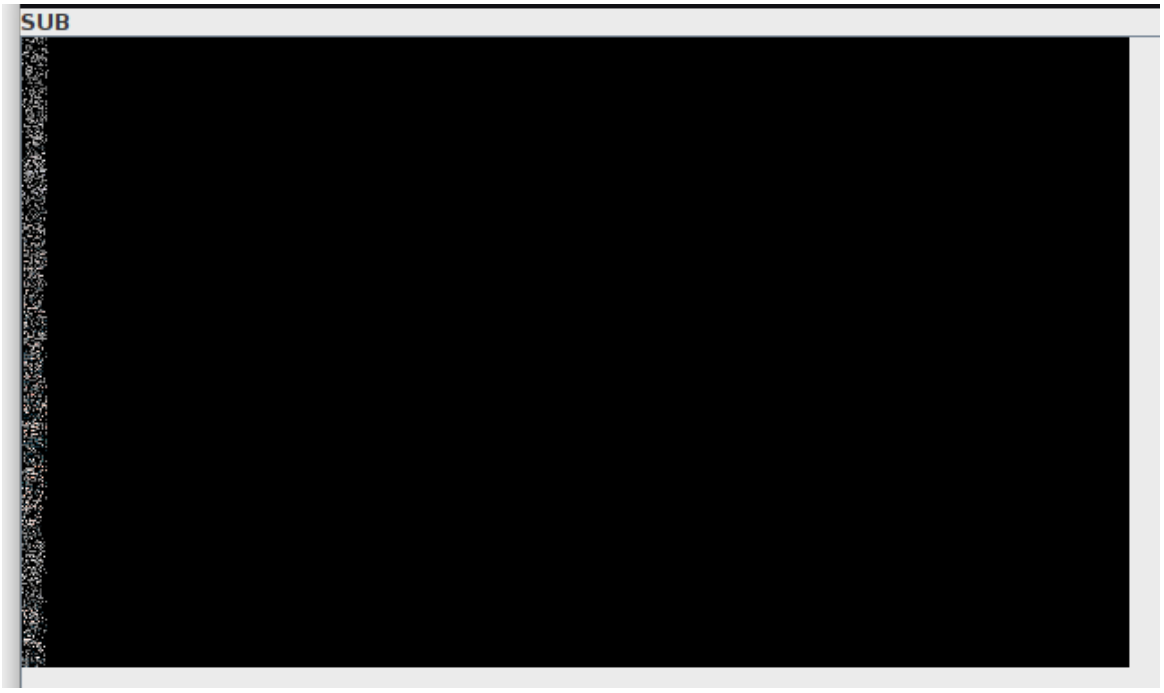
You may notice that in this archive there are two images: `intercepted.png` and `original.png` that are the same image but have different noise on the left.

A method of hiding data is in the difference between two images. This is the thread you are supposed to pull on, the other two images in the archive are red herrings.

If we use stegsolve with the Image Combiner tool we can get the difference (SUB) between the 2 images:



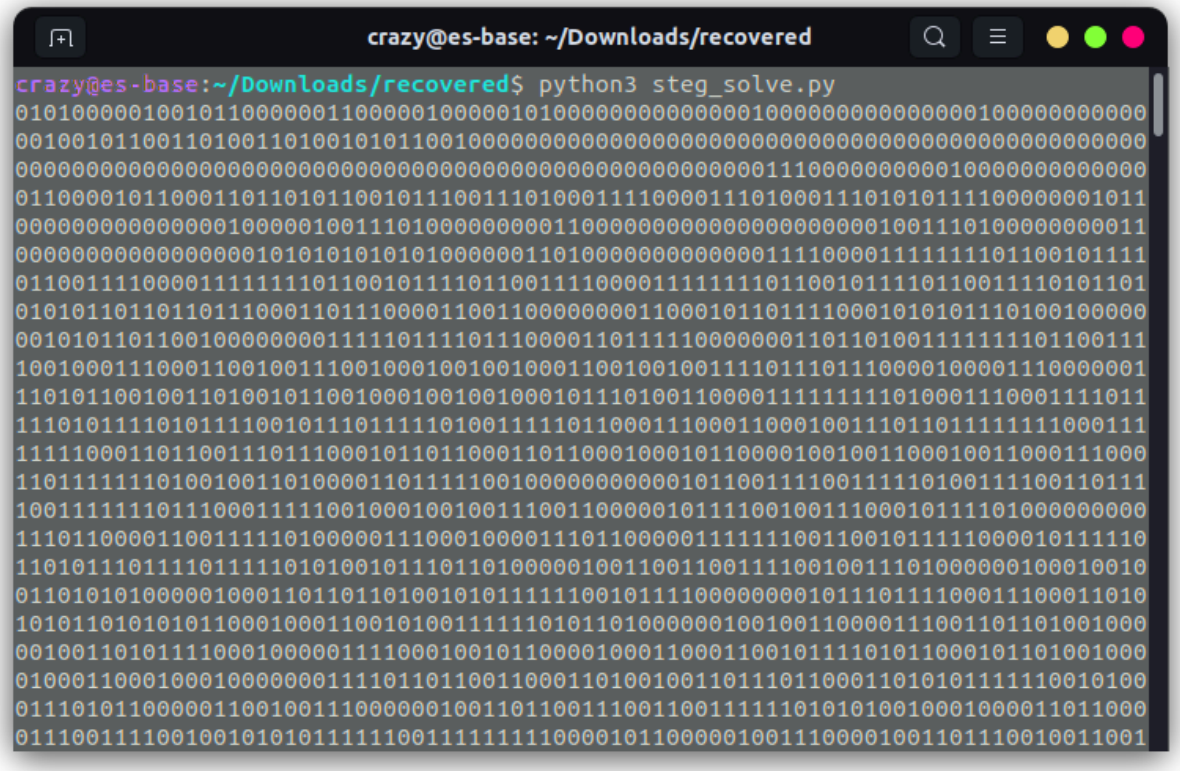
Shown more closely:



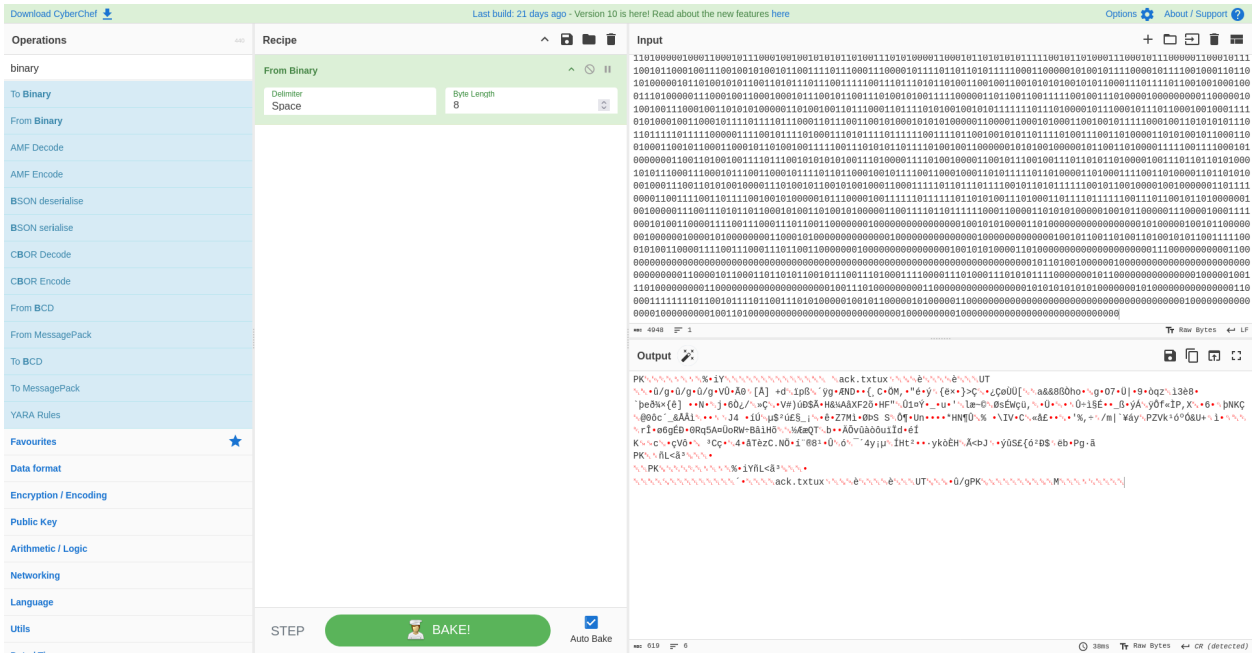
We can see that the noise along the size of the image is binary data.
We can write a script to extract it, converting black pixels to 0s and white pixels to 1s:

```
from PIL import Image

img_orig = Image.open("solved.bmp")
rgb_im = img_orig.convert('RGB')


width = 615
height = 350

pix = img_orig.load()
output = ""
for j in range(width):
   for i in range(height):
       bin = pix[j,i]
       if str(bin) == "(0, 0, 0)":
           output += "0"
       else:
           output += "1"

print(output)
```

We get the following:

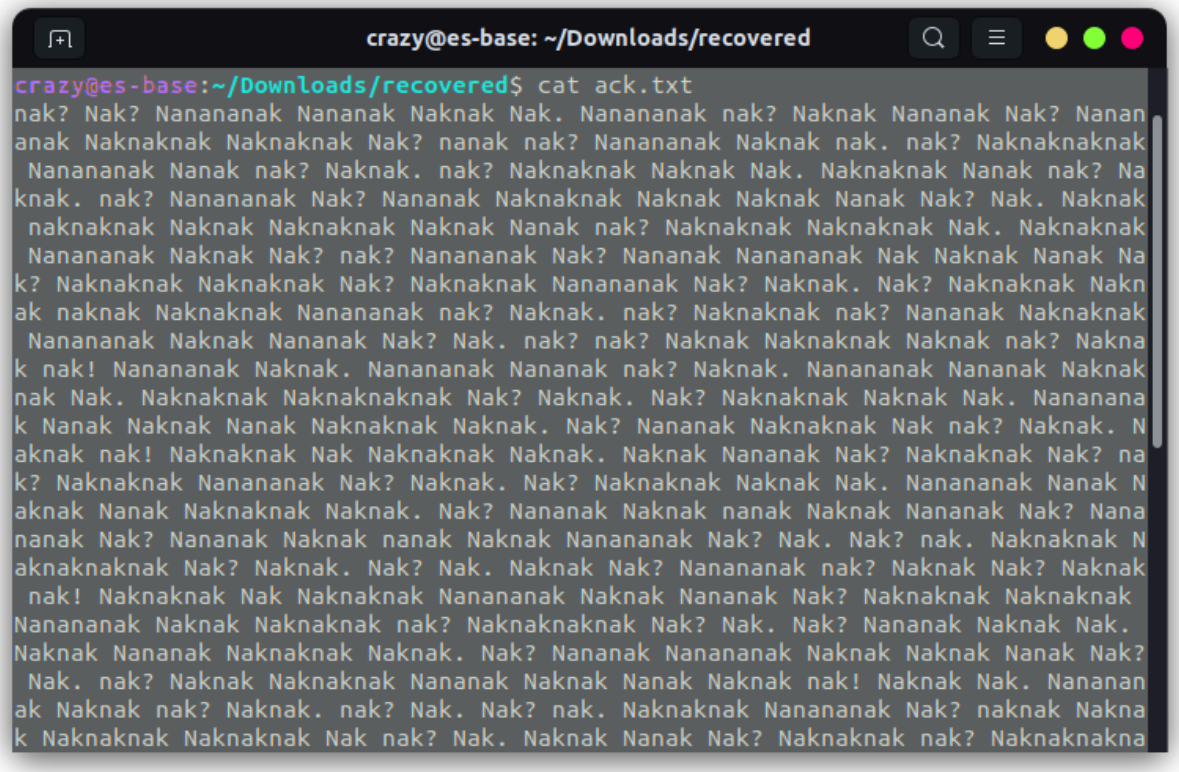


Lets drop it into good olde cyberchef:

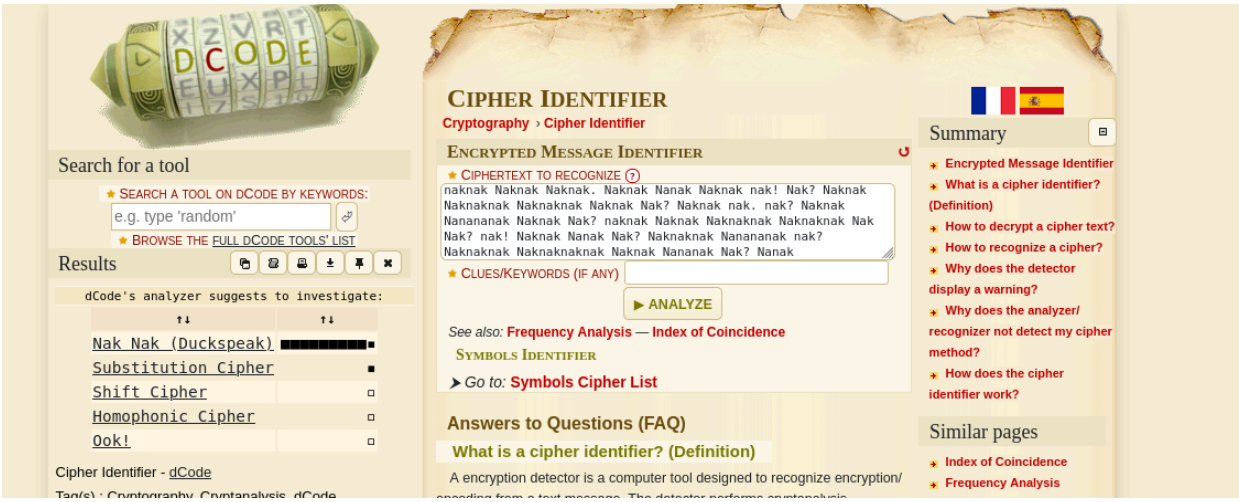


We can see that this is a zip file, but when we try to open it after downloading it, it will be corrupted. We must have the correct number of trailing null-bytes in order to get a zip file. This can be easily done with a script, adding 0's after the last 1 in the output of the previous script (steg_solve.py) until you get a valid zip file.
The total size of the zip file should be `4952` bytes.
Extract the contents of the archive.

When we extract the contents of the zip file we get a text file ack.txt:



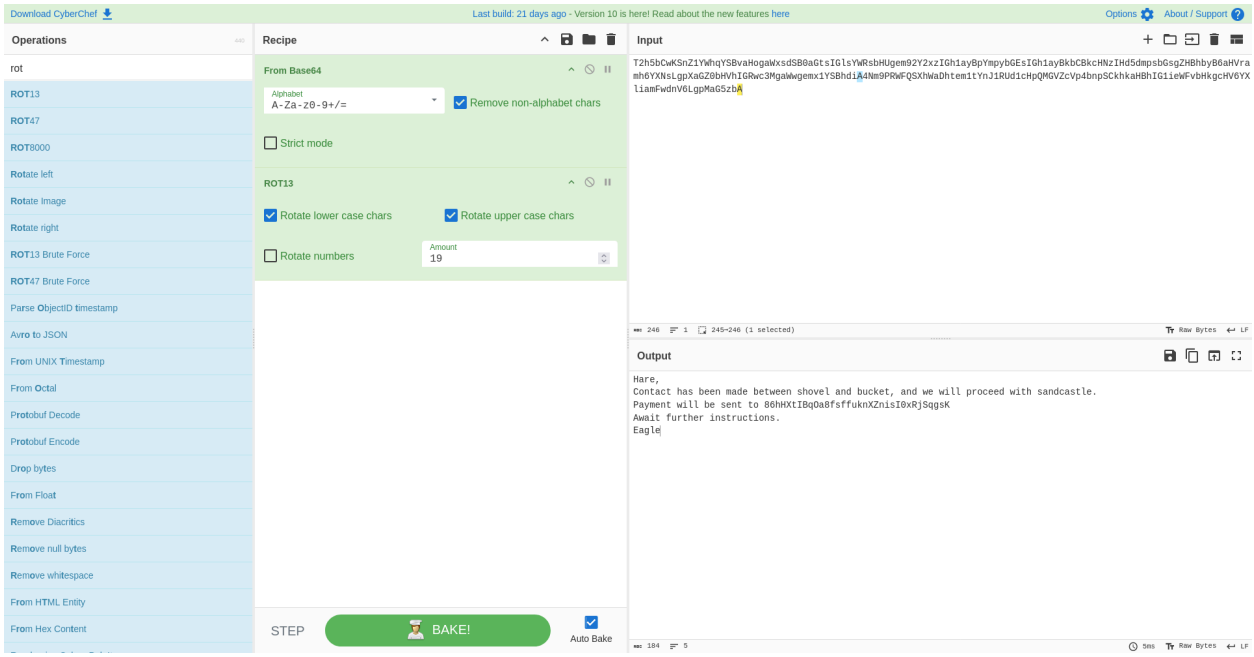
We visit my favorite site: dcode.fr
And use their cipher identifier to figure out what in the tarnation this is:
https://www.dcode.fr/cipher-identifier



Its Duckspeak, quack

Use the Nak Nak (Duckspeak) tool to decode it:

Use cyberchef to finish decoding and we can see the BTC address:



Flag: 86hHXtIBqOa8fsffuknXZnisI0xRjSqgsK

DONE. THATS A WRAP :)

FEEL FREE TO DM ME WITH FEEDBACK ON THIS CHALLENGE :)