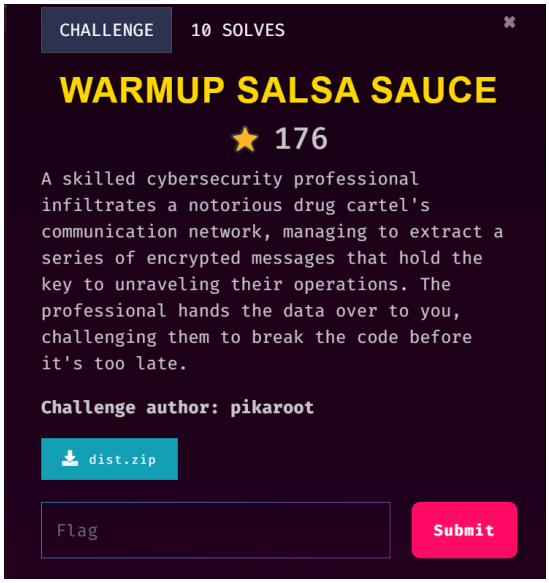
Girls in CTF 2024

Warmup Salsa Sauce

Warmup Salsa Sauce is in the CRYPTOGRAPHY category.



The challenge gave us 2 files, wss.py and out.txt.

wss.py

```python
from Crypto.Cipher import Salsa20
from secret import FLAG
from secrets import token_bytes as tb

def encText(text, key, nonce):
    cipher = Salsa20.new(key=key, nonce=nonce)
    ciphertext = cipher.nonce + cipher.encrypt(text)
    return ciphertext

if __name__ == "__main__":
    text = b"We covered the drugs with our favourite salsa sauce. The stupid cops will not find it."

    key, nonce = tb(32), tb(8)

    enc_text = encText(text, key, nonce)
    enc_flag = encText(FLAG, key, nonce)

    with open('out.txt','w') as f:
        f.write(f"Encrypted flag: {enc_flag.hex()}\n")
        f.write(f"Encrypted text: {enc_text.hex()}")
```

out.txt

```
└─$ cat out.txt
Encrypted flag: fa1c26b66ad926ab75cd51524f05ec08f7f3160c74bec57f8aec3f7cace6fbb7370923e8c540673f657dada9e9540101d7f4dc0b6627f147fc47627a244c88b2ea6c3340
Encrypted text: fa1c26b66ad926ab45cb05575b0ab90fcdf1060d72bfba6f90aa076cbcf2f3aa311a13fc800638382570bcbee142001290f1d41d3761e315b91730663c2cd8e1fe7a25482ce0cd69745028635ef5dae54282f162e448fec5f6b0e7d8ff85
```

By examining the source code, I determined that the encryption is done using Salsa20, which is also hinted at by the challenge name. I understand that Salsa20 employs XOR (exclusive OR) for both encryption and decryption.

In this case:

- Ciphertext = Plaintext XOR Key
- Key = Plaintext XOR Ciphertext

Since we already have the plaintext from the source code and the encrypted text from out.txt, we can use them to derive the key. This key can then be used to decrypt the flag.

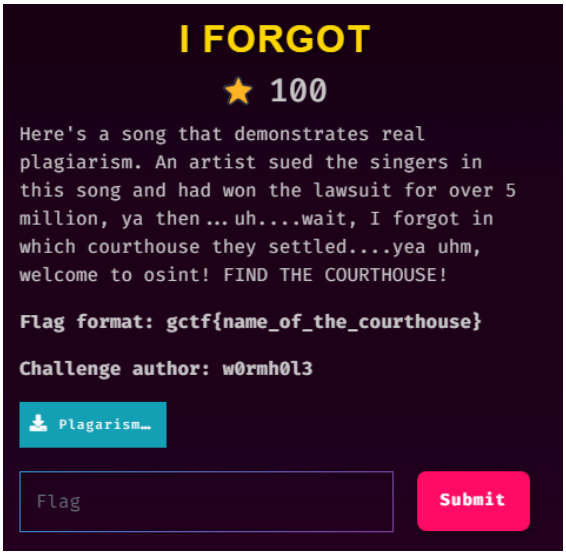
Here is the source code I used, **test.py**:

```python
from binascii import unhexlify

#Encrypted data from out.txt
encrypted_flag = unhexlify("fa1c26b66ad926ab75cd51524f05ec08f7f3160c74bec57f8aec3f7cace6fbb7370923e8c540673f657dada9e9540101d7f4dc0b6627f147fc47627a244
encrypted_text = unhexlify("fa1c26b66ad926ab45cb05575b0ab90fcdf1060d72bfba6f90aa076cbcf2f3aa311a13fc800638382570bcbee142001290f1d41d3761e315b91730663c2
plaintext = b"We covered the drugs with our favourite salsa sauce. The stupid cops will not find it."

#Since Salsa20 uses the first 8 bytes as the nonce, we want to skip that
nonce_len = 8
ciphertext_without_nonce = encrypted_text[nonce_len:]

#After nonce, XOR the encrypted text with the plaintext to get the keystream. We are using the keystream for decrypting flag later
keystream = bytes(a ^ b for a, b in zip(ciphertext_without_nonce, plaintext))

#Skipping the first 8 bytes as the nonce in encrypted flag as well
cipher_flag_without_nonce = encrypted_flag[nonce_len:]

#Now, XOR the keystream with the encrypted flag to recover the flag
flag = bytes(a ^ b for a, b in zip(cipher_flag_without_nonce, keystream))

#Printing out the decrypted flag
print(flag.decode())
```

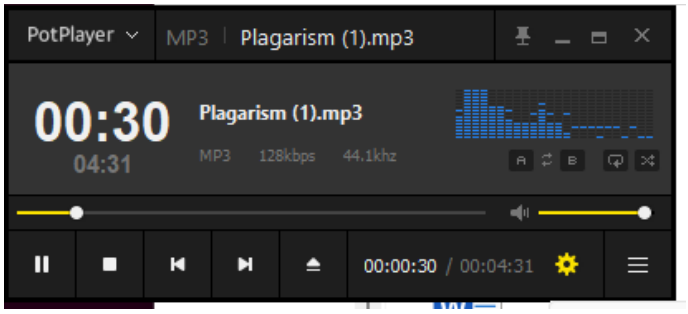
```
└─$ python3 ./test.py
gctf{y0u_f0und_th3_c0cain3_a7f9f6bdeabd34dde0fa3037284864eb}
```

I Forgot

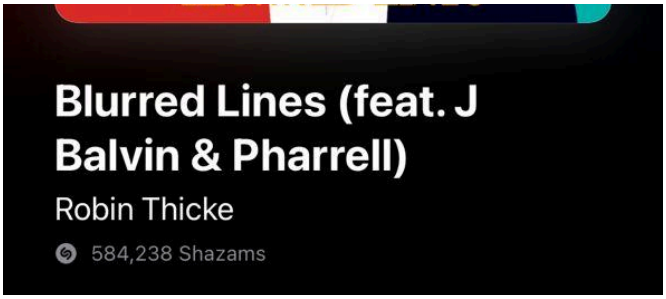
I Forgot is a challenge in the MISC category.



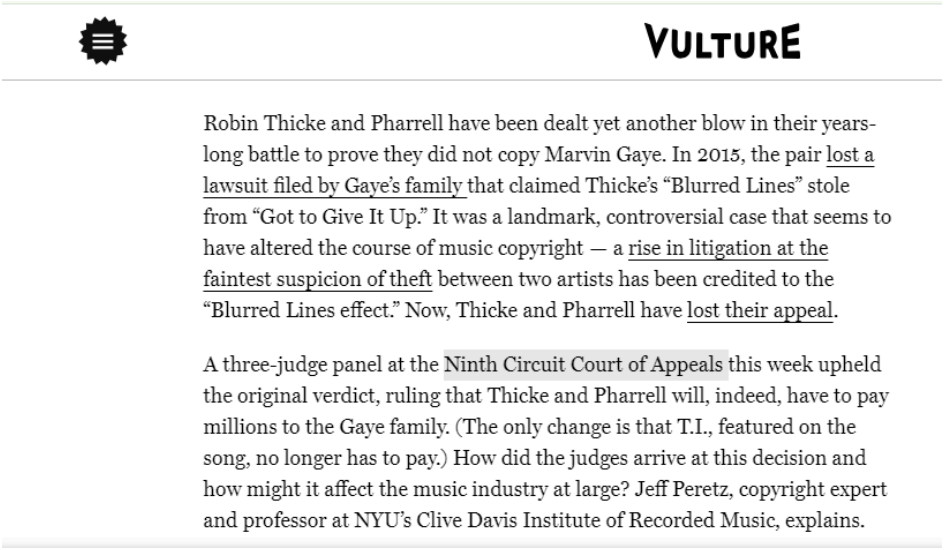
Based on the question I know I need to find the courthouse name. So, first I download the file given which is a mp3 file type. However, what song is in that audio file?



I applied Shazam, an app that allows you to search for music by name if you are unsure of the title, and voila—I am able to identify the song now.



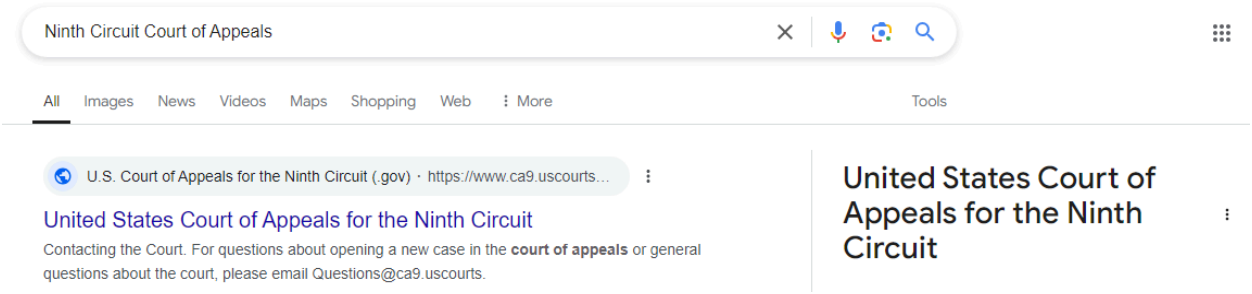
I read the challenge's description once more, and it turns out that all I need to do is some good old fashioned digging! My strategy is to locate the courtroom where the case is being heard after determining which song is at issue in the lawsuit. I have to do some experimenting because various news sources occasionally refer to the courthouse under different names. Eventually, after much trial and error, I came upon an article that gave the courthouse a unique name.



VULTURE

Robin Thicke and Pharrell have been dealt yet another blow in their years-long battle to prove they did not copy Marvin Gaye. In 2015, the pair lost a lawsuit filed by Gaye's family that claimed Thicke's "Blurred Lines" stole from "Got to Give It Up." It was a landmark, controversial case that seems to have altered the course of music copyright — a rise in litigation at the faintest suspicion of theft between two artists has been credited to the "Blurred Lines effect." Now, Thicke and Pharrell have lost their appeal.

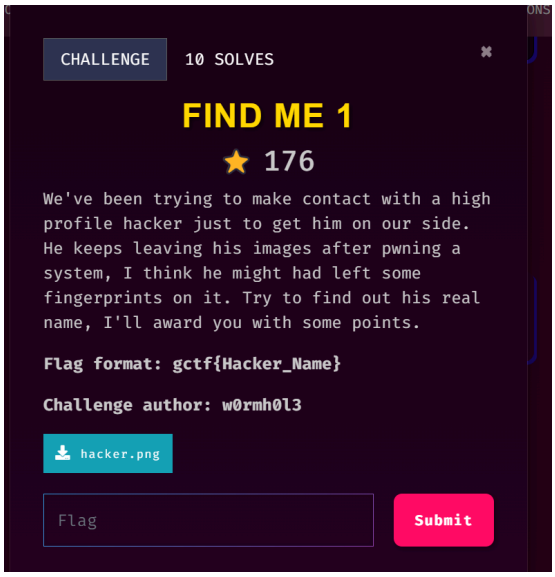
A three-judge panel at the Ninth Circuit Court of Appeals this week upheld the original verdict, ruling that Thicke and Pharrell will, indeed, have to pay millions to the Gaye family. (The only change is that T.I., featured on the song, no longer has to pay.) How did the judges arrive at this decision and how might it affect the music industry at large? Jeff Peretz, copyright expert and professor at NYU's Clive Davis Institute of Recorded Music, explains.



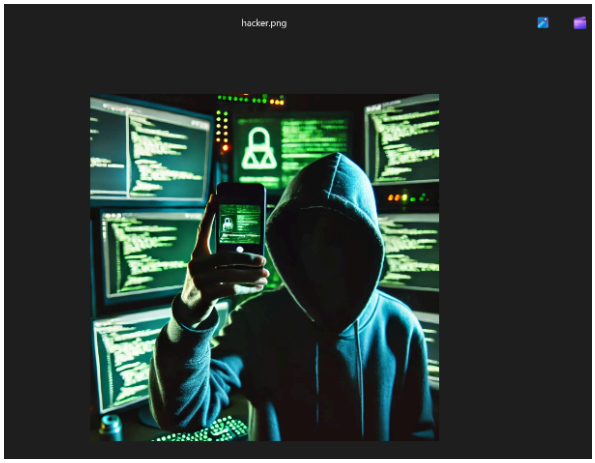
When I searched for the court's name, the entire name appeared! I tried it using the full name, and you know what happened? I got it right—I captured the flag!

Find Me 1

Find Me 1 is a challenge in the MISC category.



After reading the challenge description, I figured it is OSINT.



They provided us with this hacker picture. I tried zooming it and stuff, nothing extraordinary.

So I tried all the basic tools to use for an image (exiftool, binwalk, steghide) and while using exiftool, I found a certain information that caught my eyes, ‘ShadeRaider96’, near the bottom.

```
└─$ exiftool hacker.png
ExifTool Version Number         : 12.76
File Name                       : hacker.png
Directory                       : .
File Size                       : 785 kB
File Modification Date/Time     : 2024:10:11 23:56:56-04:00
File Access Date/Time           : 2024:10:11 23:58:15-04:00
File Inode Change Date/Time     : 2024:10:11 23:56:56-04:00
File Permissions                : -rw-rw-r--
File Type                       : PNG
File Type Extension             : png
MIME Type                       : image/png
Image Width                     : 677
Image Height                    : 673
Bit Depth                       : 8
Color Type                      : RGB with Alpha
Compression                     : Deflate/Inflate
Filter                          : Adaptive
Interlace                       : Noninterlaced
SRGB Rendering                  : Perceptual
Gamma                           : 2.2
Pixels Per Unit X               : 3779
Pixels Per Unit Y               : 3779
Pixel Units                     : meters
XMP Toolkit                     : Image::ExifTool 12.76
Rights                          : ShadeRaider96
Image Size                      : 677×673
Megapixels                      : 0.456
```

‘ShadeRaider96’ looked like a username to me so I went and search on X and there you go, the same username along with a real name.

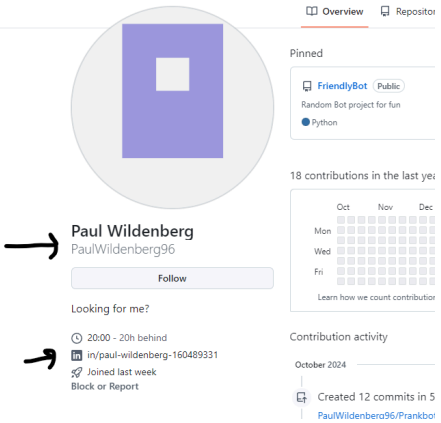
The picture posted in the account for more confirmation that it is the right person :D

Find me 2

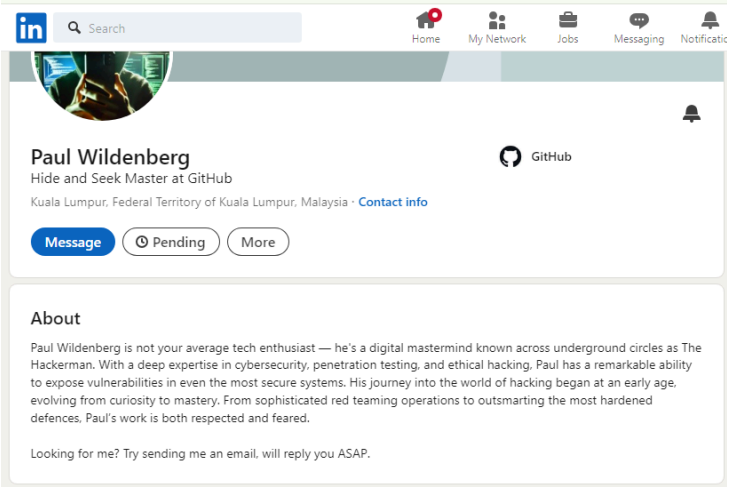
Find Me 2 is a challenge in the MISC category.

It’s a continuation of Find Me 1, this time I had to find a way to contact him.

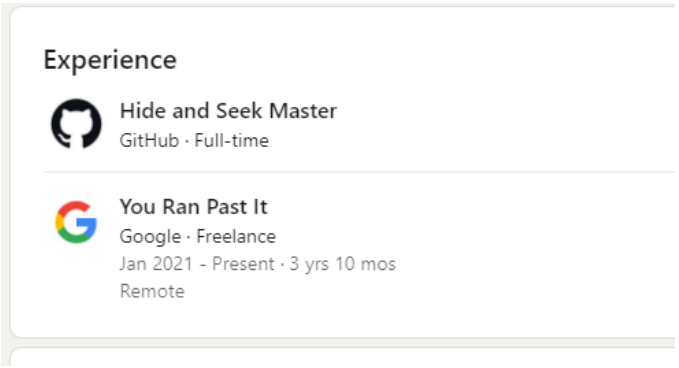
The description said he disliked replying through socials so we can’t message him on X. However on the X’s account bio, there is a link to a github account.



You can't send a personal message on GitHub (last I know), and after looking around there is nothing much on the account other than another link, this time to a linkedin account. Also note down the ‘PaulWildenberg96’ username.



We managed to get to his linkedin account. In the ‘About’ section, he mentioned sending him an email but after looking around, we could not find any email at all.



So in the ‘Experience’ section, I read these 2 things and after connecting some dots, I tried to link back to the ‘PaulWildenberg96’ from earlier and decide to see if ‘paulwildenberg96@gmail.com’ exists, and it does! We emailed him and achieved our flag :D