An incident responder who's seeking opportunities to work in technology company!
Operator in Cookie Han Hoan
Admin in Cyber Mely

OSCTF 2024 - Forensic

OSCTF Writeup Command and Control Powershell Blue Team

Published on 13 Jul 2024

All solved forensic challenges_

Hi guys, this time I joined HITCON CTF with my team: World Wide Union, but because of no forensic challenges, I had to go here and try to solve some challenges. Now it's my writeup for them, let's go!

The Lost Image Mystery

They gave us a corrupted image and we need to recover it. I used **xxd** to check hex values inside:

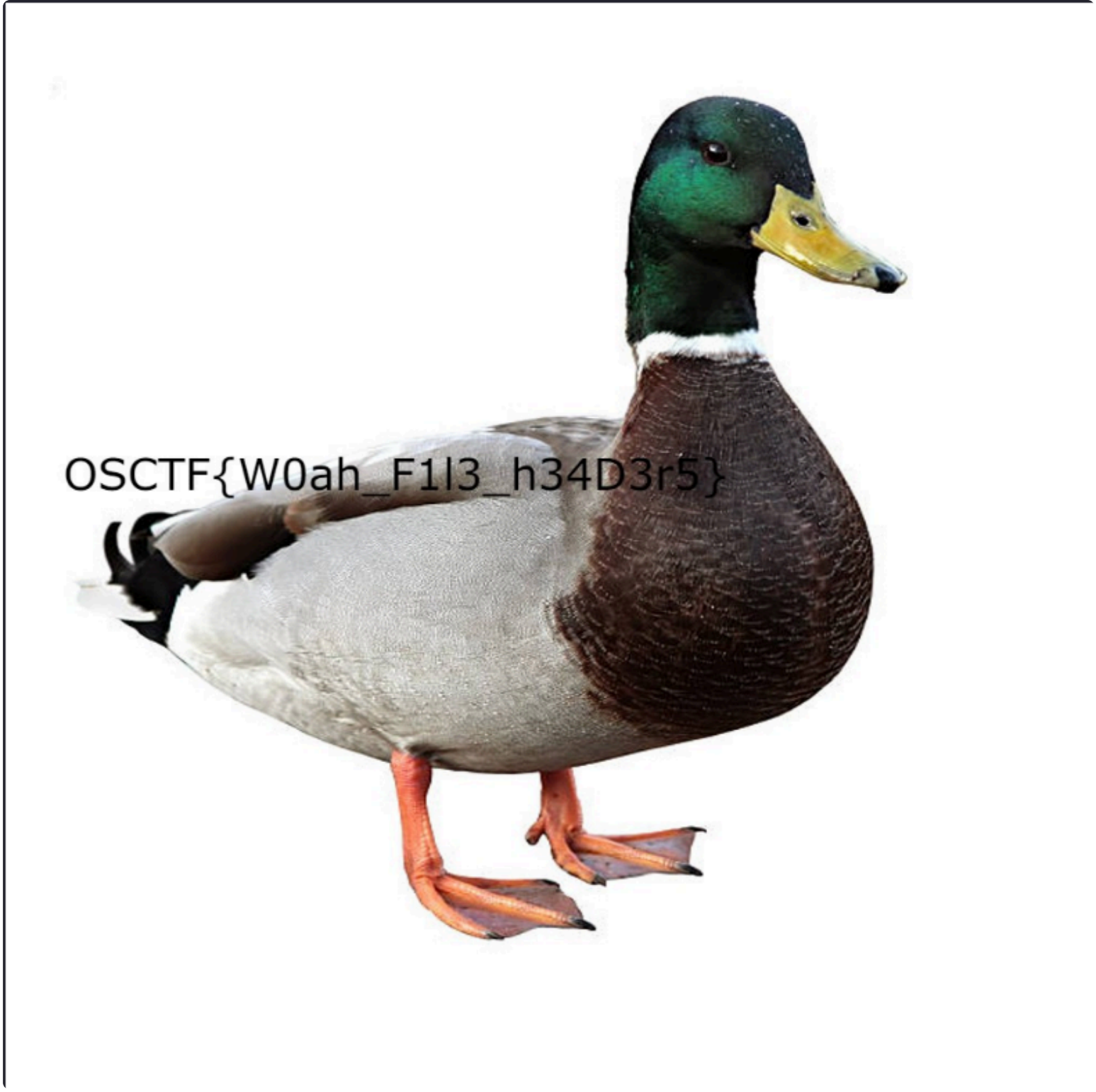
```
┌──(odin㉿DFIR)-[~/Downloads]
└─$ xxd image.png | head -n 10
00000000: d8e0 0049 4600 0101 0000 0100 0100 00ff   ...IF...........
00000010: e201 d849 4343 5f50 524f 4649 4c45 0001   ...ICC_PROFILE..
00000020: 0100 0001 c800 0000 0004 3000 006d 6e74   ..........0..mnt
00000030: 7252 4742 2058 595a 2007 e000 0100 0100  rRGB XYZ .......
00000040: 0000 0000 0061 6373 7000 0000 0000 0000   .....acsp.......
00000050: 0000 0000 0000 0000 0000 0000 0000 0000   ................
00000060: 0000 0000 0100 00f6 d600 0100 0000 00d3   ................
00000070: 2d00 0000 0000 0000 0000 0000 0000 0000   -...............
00000080: 0000 0000 0000 0000 0000 0000 0000 0000   ................
00000090: 0000 0000 0000 0000 0000 0000 0000 0000   ................

┌──(odin㉿DFIR)-[~/Downloads]
└─$
```

You can guess easily it must be JPG or JPEG file because of ...IF. From here you can use this list to check the signature for the file:

FF D8 FF DB	ÿØÿÛ	0	jpg jpeg	JPEG raw or in the JFIF or Exif file format[16]
FF D8 FF E0 00 10 4A 46 49 46 00 01	ÿØÿà NULDLE JFIF NULSOH			
FF D8 FF EE	ÿØÿî			
FF D8 FF E1 ?? ?? 45 78 69 66 00 00	ÿØÿá??Exif NULNUL			

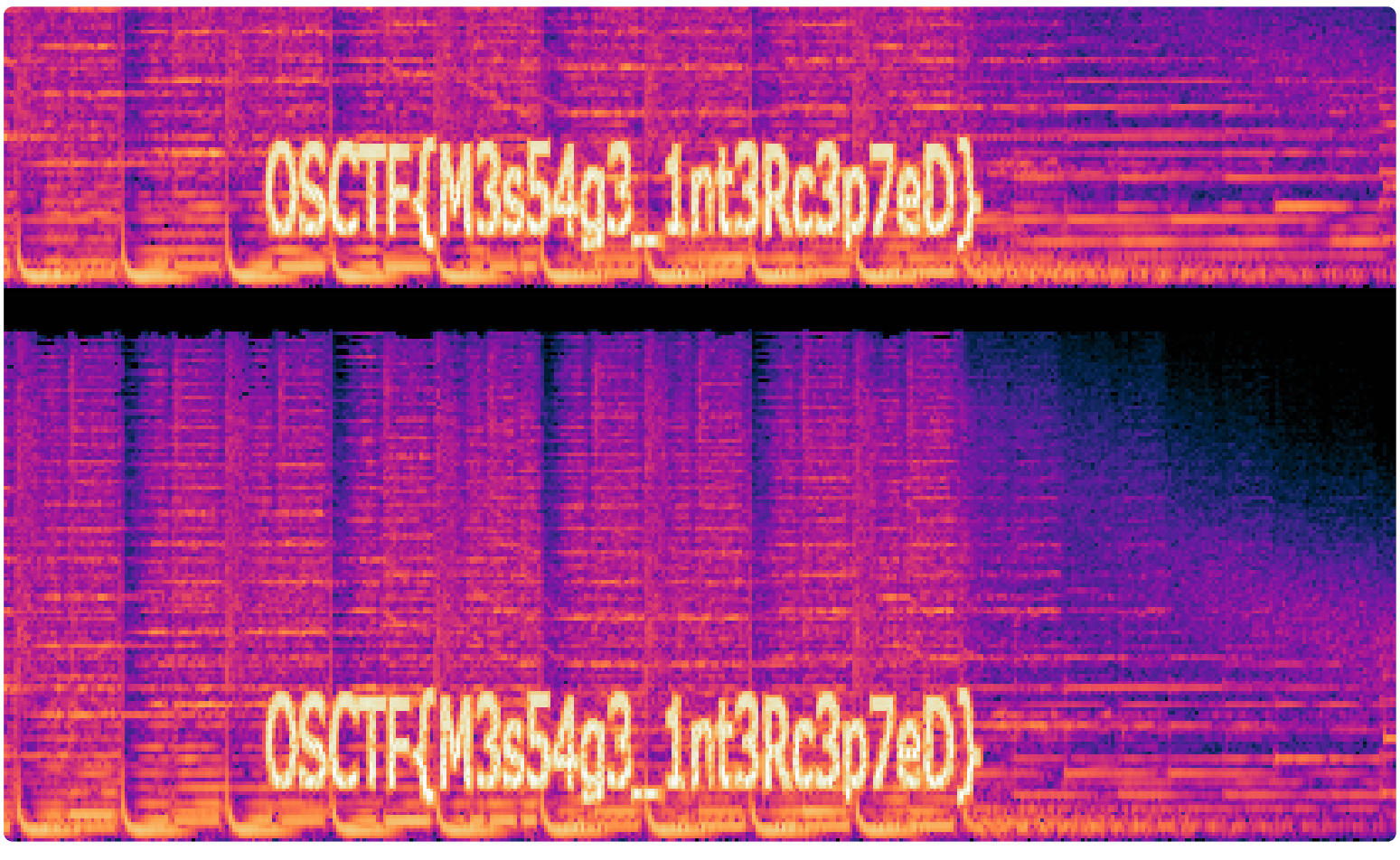
Use **hexedit** to edit hex value, open the file again and enjoy your result:



Flag: OSCTF{W0ah_F1l3_h34D3r5}

The Hidden Soundwave

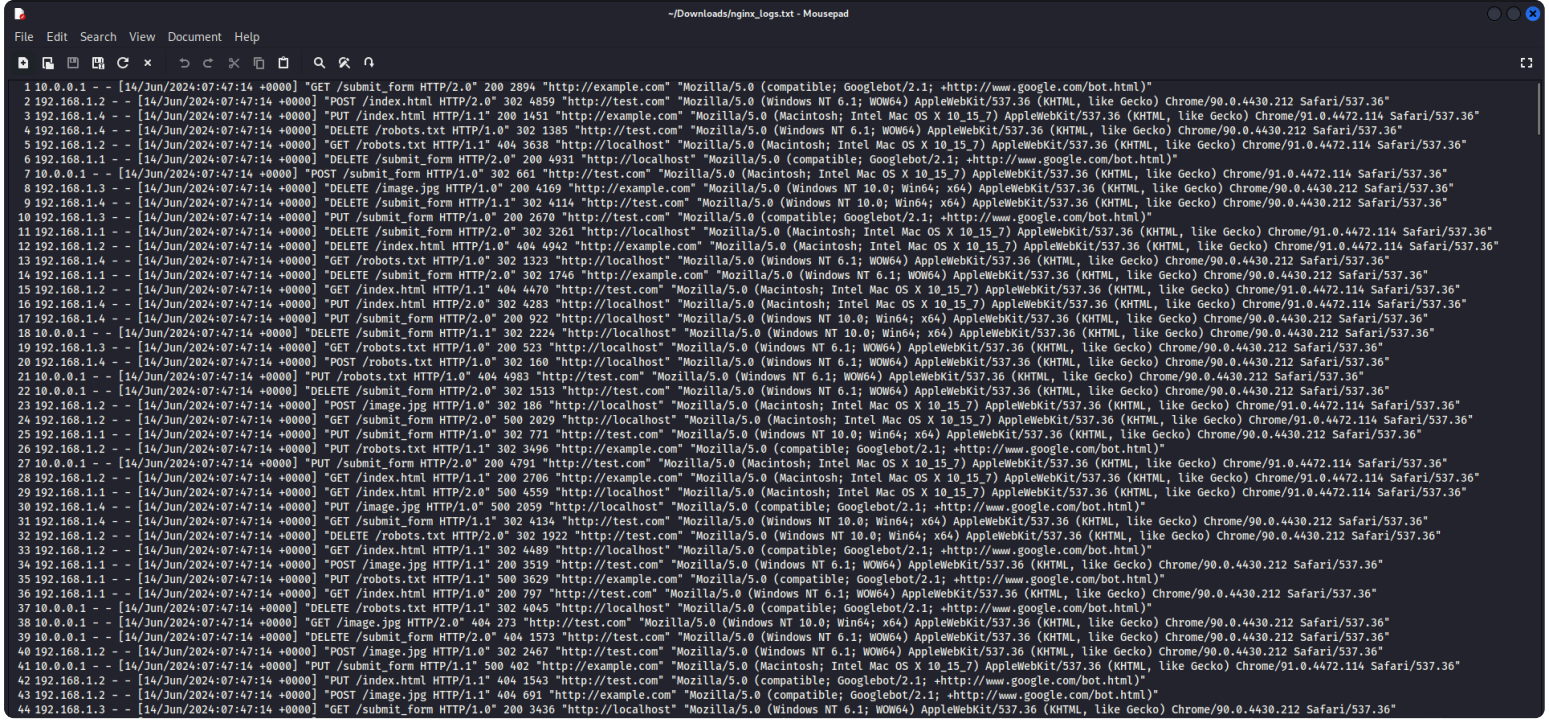
We got an audio file, and as the title, you need to find hidden information inside the audio file. Very basic, I always check spectrogram because it appeared in many CTFs 😂 😂 😂. From here I used audacity to open audio file, change to spectrogram view and I got the flag:



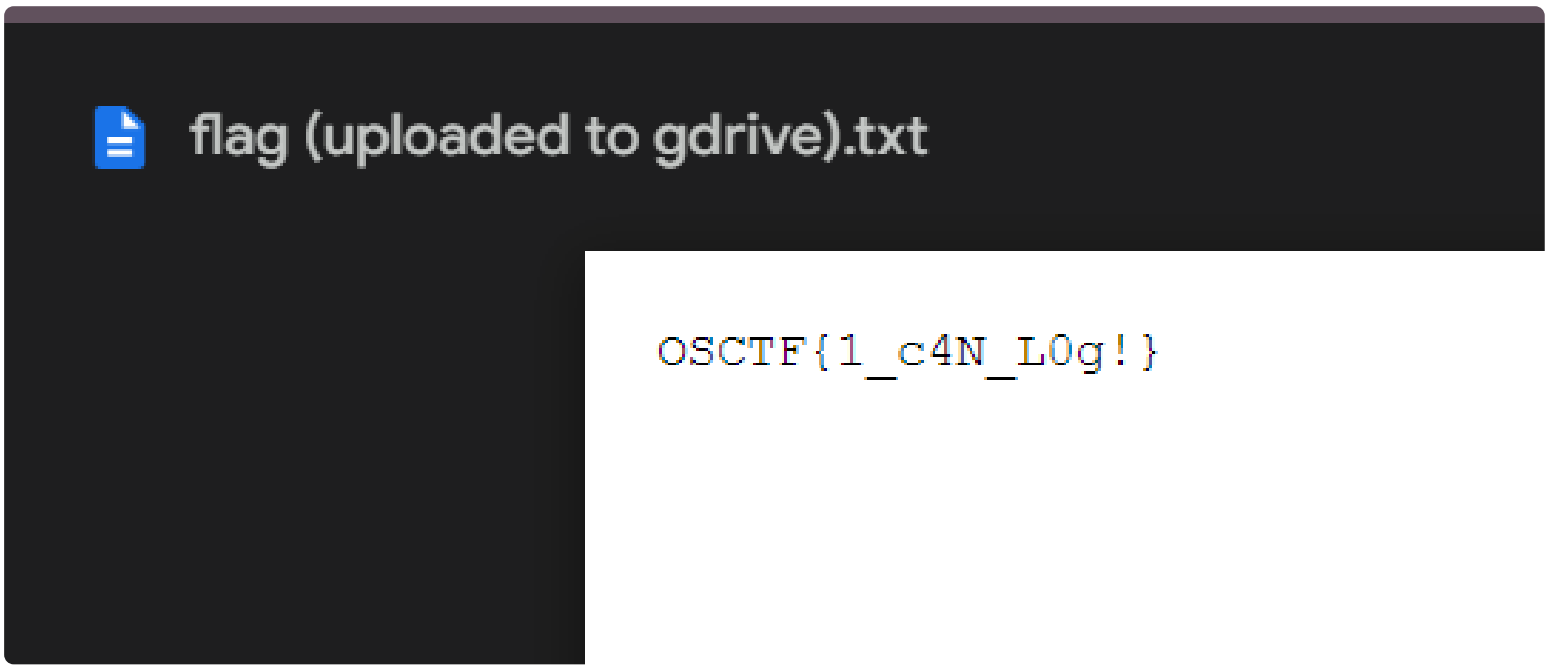
Flag: OSCTF{M3s54g3_1nt3Rc3p7eD}

Mysterious Website Incident

Now we had a nginx log, and very simple, we just open in text editor and analyse it:



After searching, I found a GG drive link, open it and I got the flag:



Flag: OSCTF{1_c4N_L0g!}

Phantom Script Intrusion

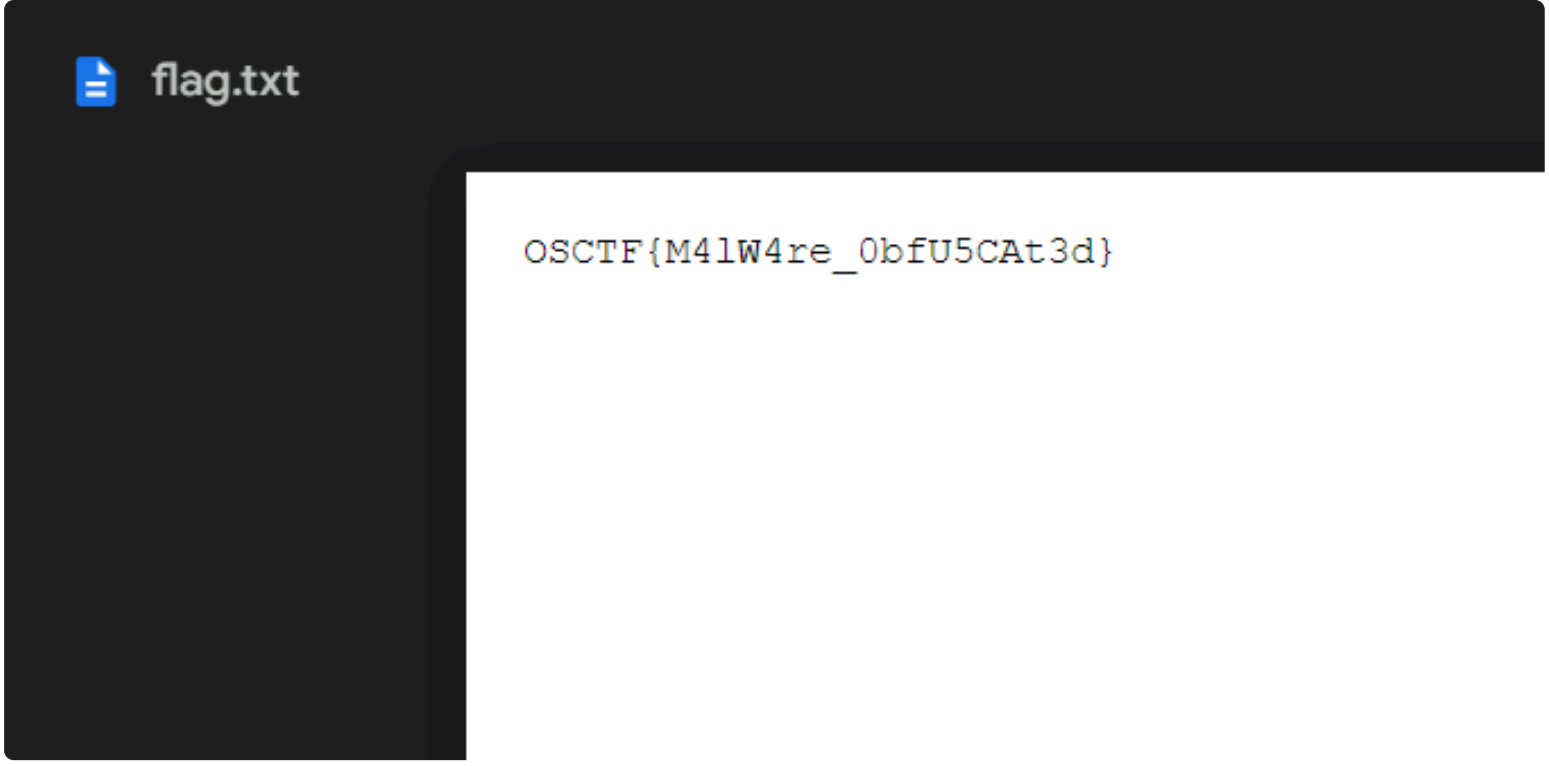
For this challenge, they gave us a PHP code, and it was obfucated:

```
<?php
 goto Ls6vZ; apeWK: ${"\x76\141\x72\61"} = str_rot13("\x24\x7b\x22\134\x78\x34\x37\134\x78\x34\143\x5c\x78\64\x66\x5c\170\x34\x32\134\x78\64\61\x5c\170\x34\x63\134\x78\x35\x33\x42\x7d"); goto G9fZX; Ls6vZ: ${"\x47\x4c\x4f\x42\101\114\123"} = "\150\x58\x58\x70\x73\72\x2f\57\163\150\x30\162\164\x75\x72\x6c\56\x61\164\x2f\x73\x31\146\x57\62"; goto apeWK; XT2kv: if (strlen(${"\x76\141\x72\x32"}) >= 0) { ${"\166\x61\x72\x33"} = ${"\x76\x61\x72\x32"}; } else { ${"\166\141\x72\63"} = ''; } goto ZYamk; V2P3O: foreach (str_split(${"\166\141\x72\x33"}) as ${"\166\x61\x72\x35"}) { ${"\166\141\162\x34"} .= chr(ord(${"\166\141\162\65"}) - 1); } goto Ly_yq; G9fZX: ${"\x76\141\162\x32"} = base64_decode(${${"\166\x61\162\x31"}}); goto XT2kv; Ly_yq: eval(${${"\x76\x61\x72\x34"}}); goto IFMxz; ZYamk: ${"\166\141\162\64"} = ''; goto V2P3O; IFMxz: ?>
```

To make it easier to follow, I deobfucated it and this is my final script:

```
${"GLOBALS"} = "hXXps://sh0rturl.at/s1fW2";
${"var1"} = str_rot13("${\"\\x47\\x4c\\x4f\\x42\\x41\\x4c\\x53\"}");
${"var2"} = base64_decode(${${"var1"}});
if (strlen(${"var2"}) > 0) {
    ${"var3"} = ${"var2"};
} else {
    ${"var3"} = "";
}
${"var4"} = "";
foreach (str_split(${"var3"}) as ${"var5"}) {
    ${"var4"} .= chr(ord(${"var5"}) - 1);
}
eval(${${"var4"}});
```

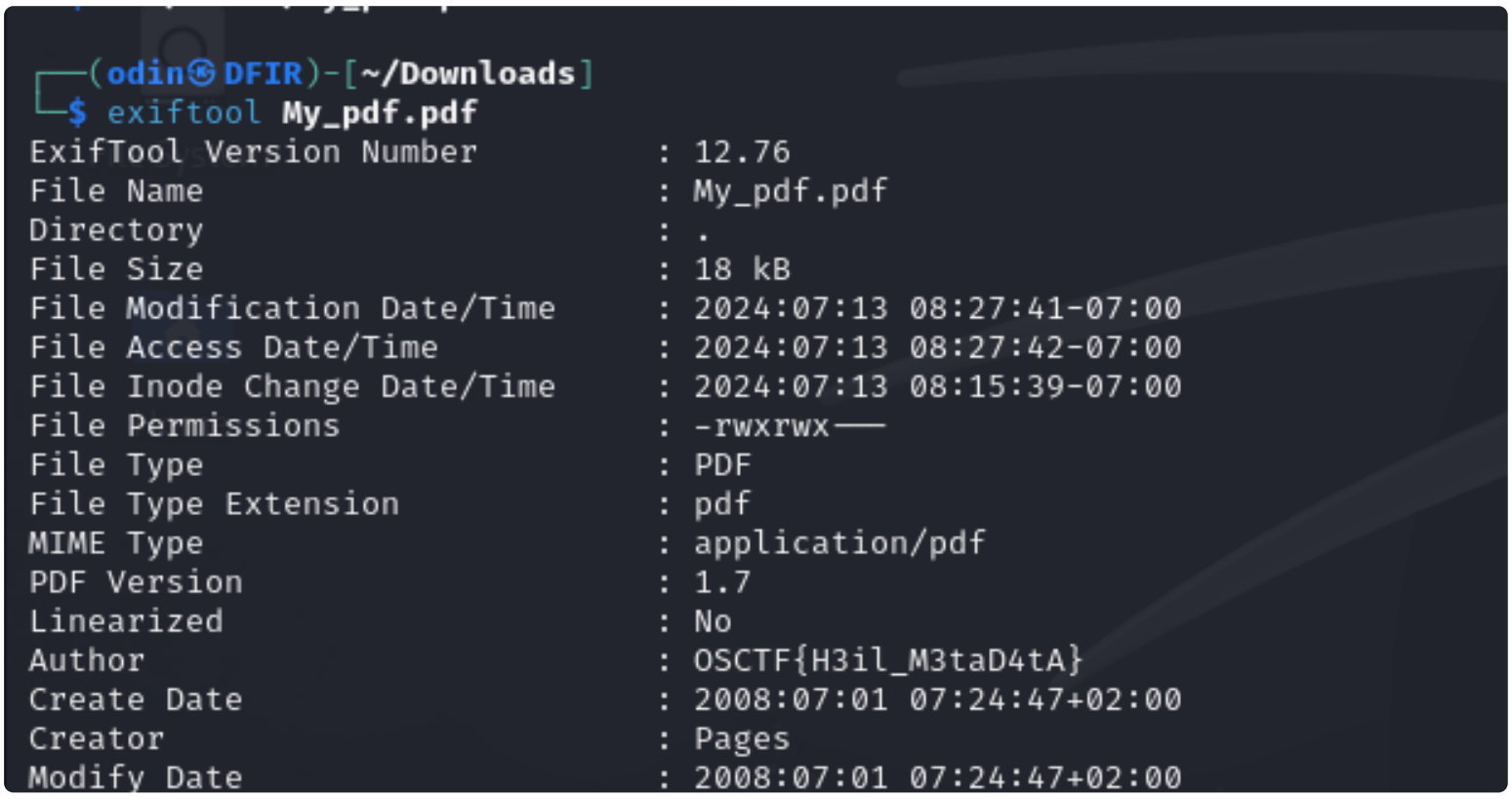
There's a shorturl link, access it and got the flag:



Flag: OSCTF{M4lW4re_0bfU5CAt3d}

PDF Puzzle

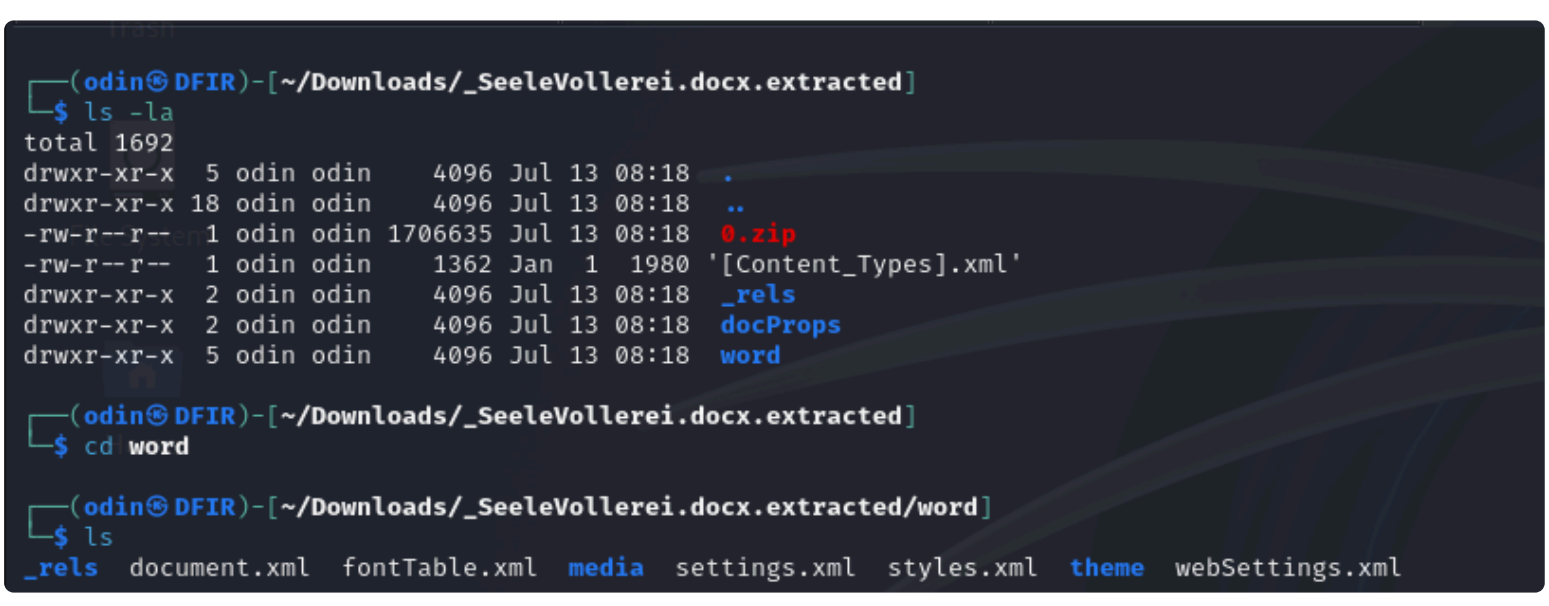
Just check the metadata of the file => get the flag:



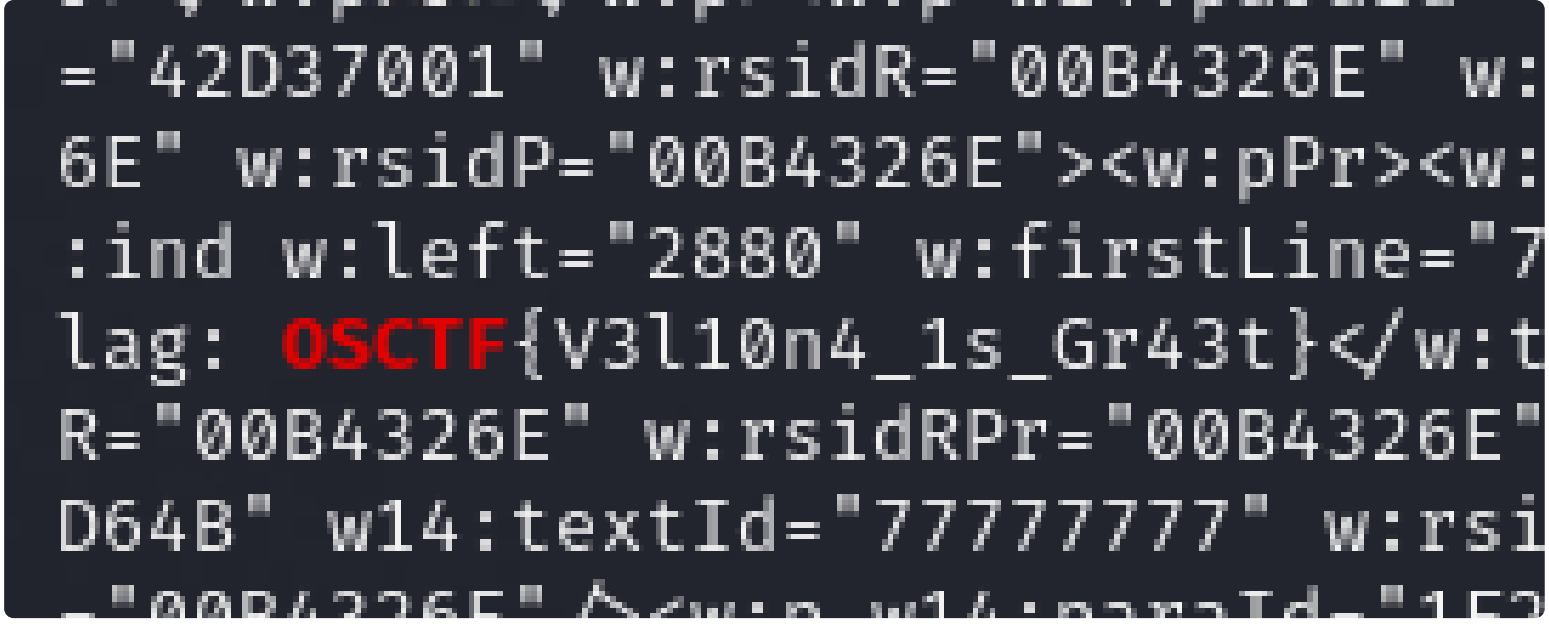
Flag: OSCTF{H3il_M3taD4tA}

Seele Vellorei

In this challenge we had a docx file. At first I tried to find out VBA code inside, but there's nothing, so I think maybe flag was hidden somewhere inside the file. Because word structure is same with zip file, you can use **binwalk** to extract all files inside:



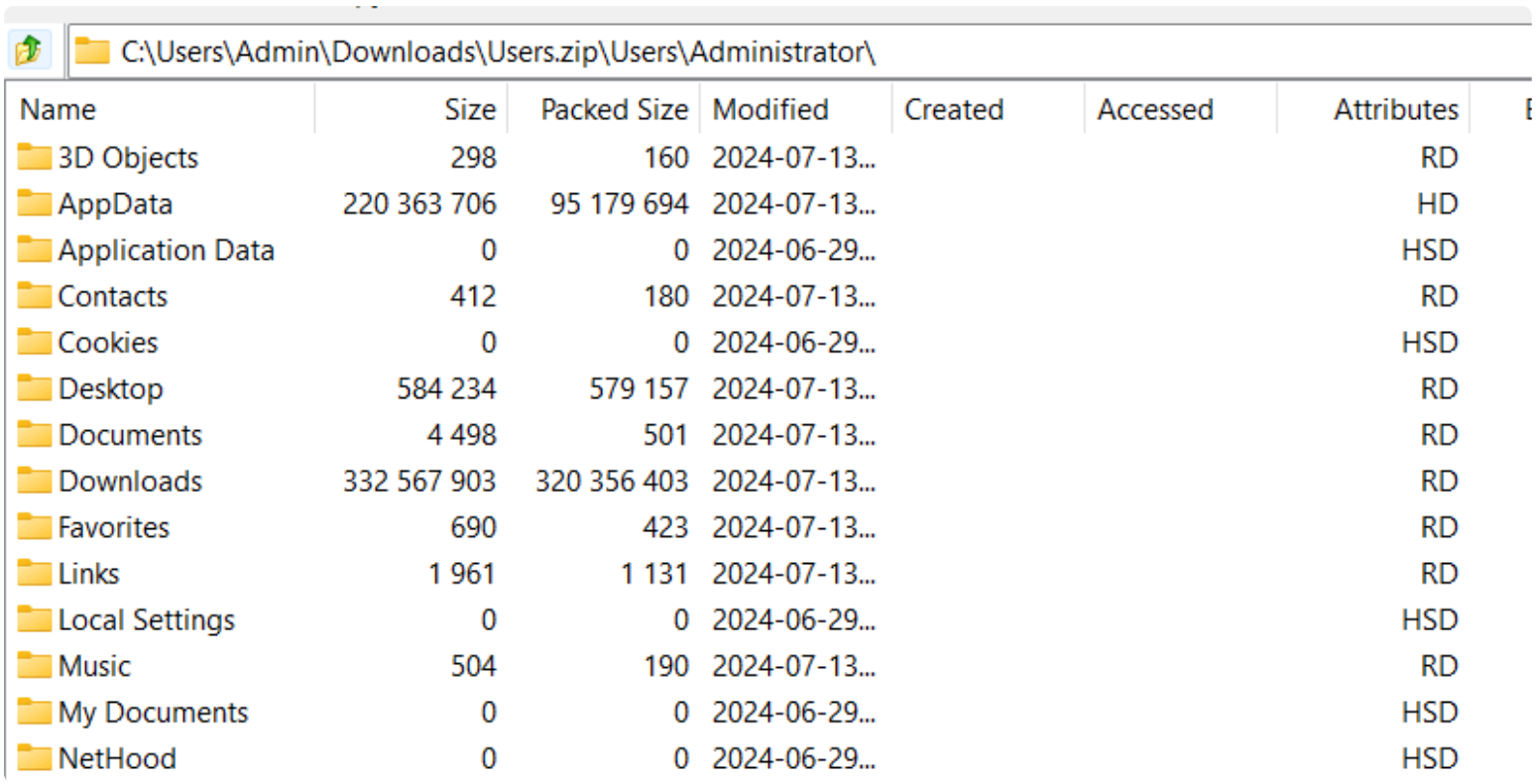
Navigate to **document.xml** where content of file was stored, use **grep** and I found the flag:



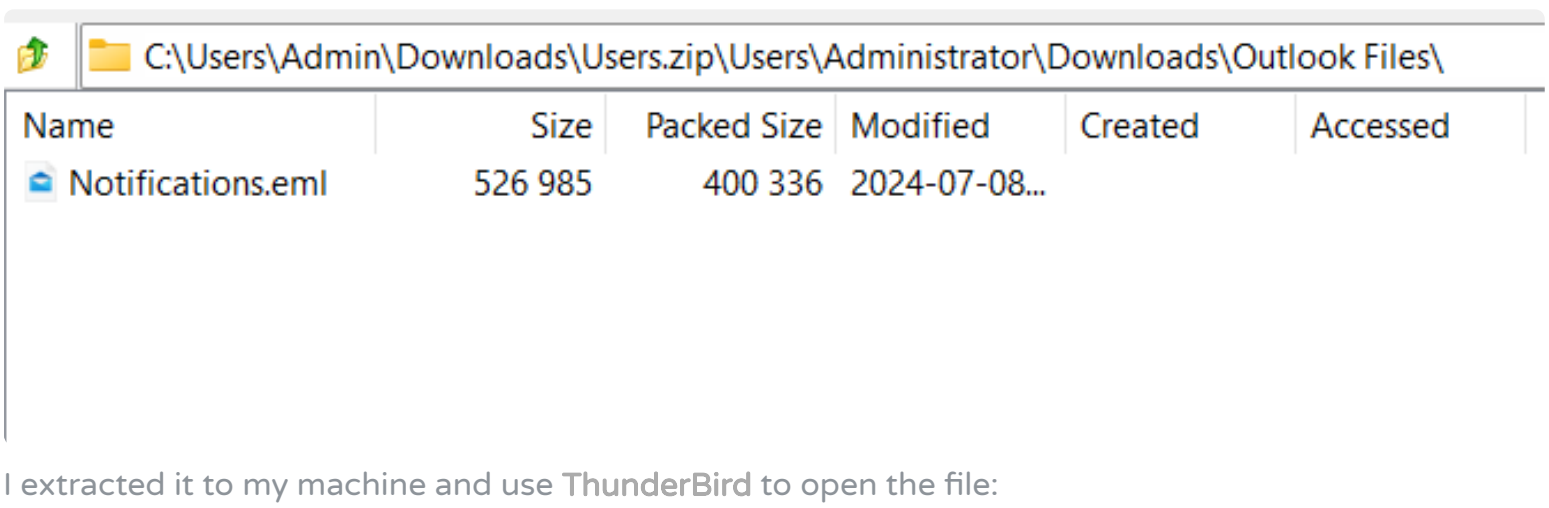
Flag: OSCTF{V3l10n4_1s_Gr43t}

FOR101

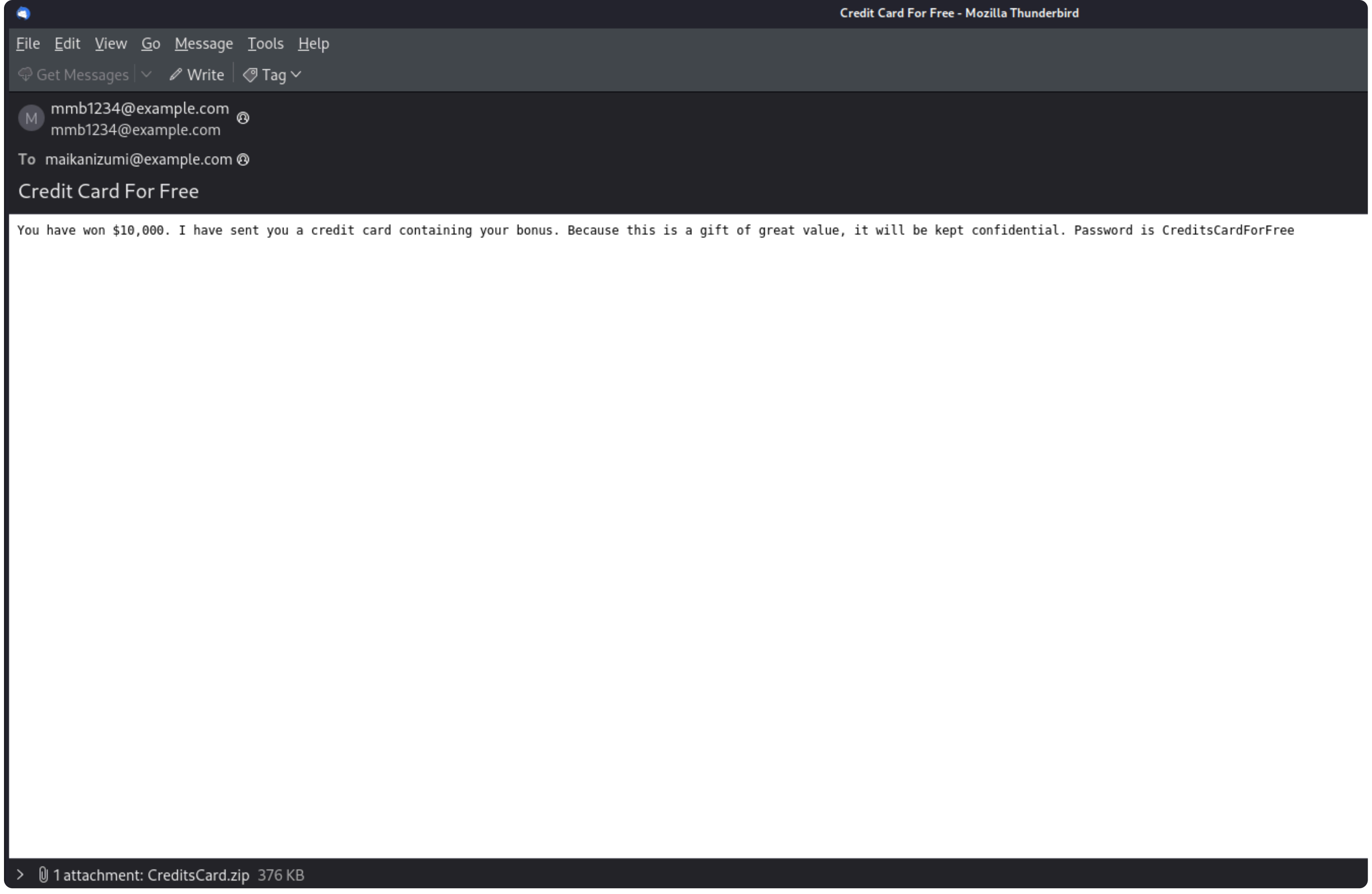
I love this challenge most, so I will explain it carefully. In this challenge we had a zip file contains datas inside an User directory. I opened it by 7z:



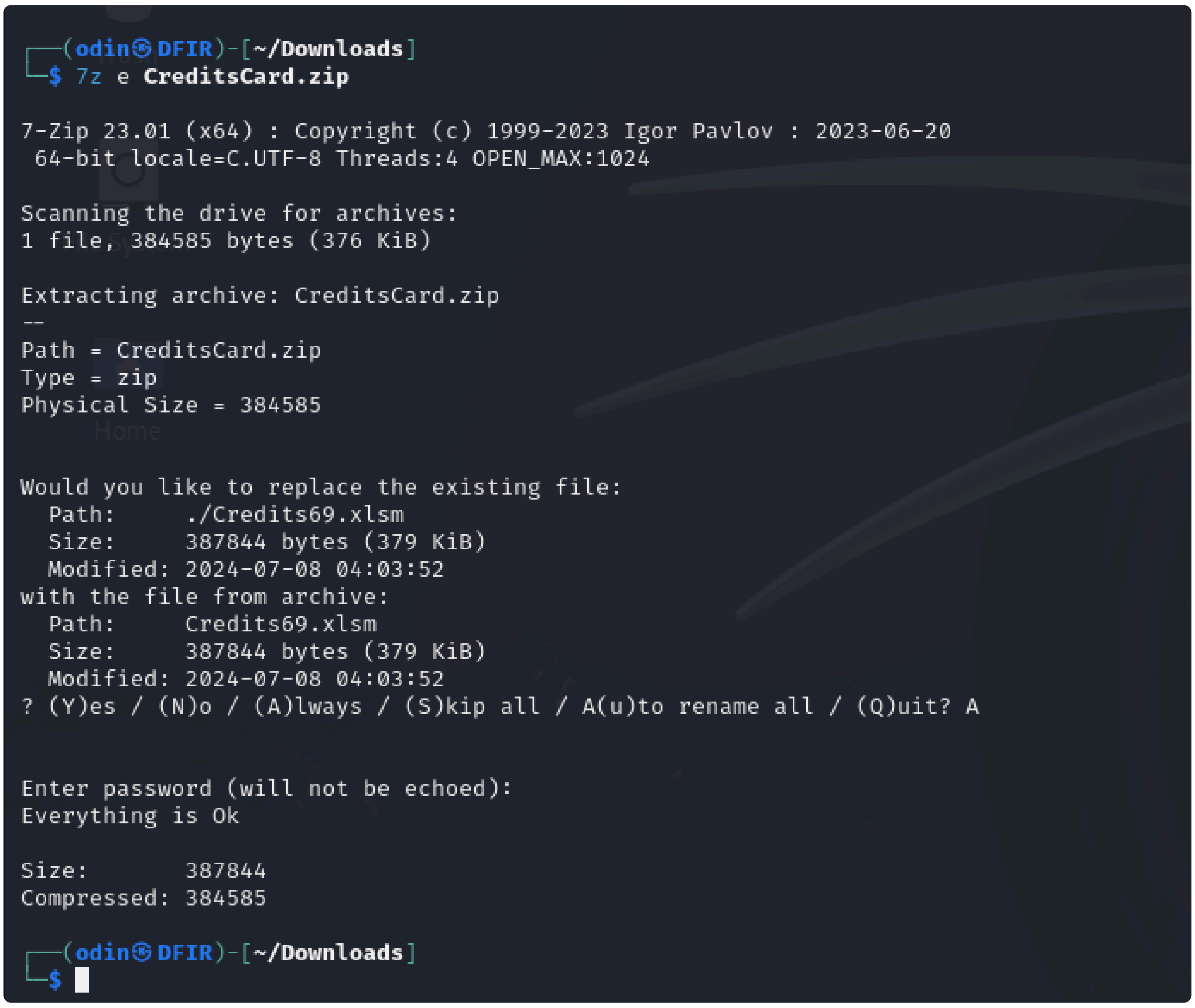
After searching, I found an .eml file at \Users\Administrator\Downloads\Outlook Files named Notifications.eml:



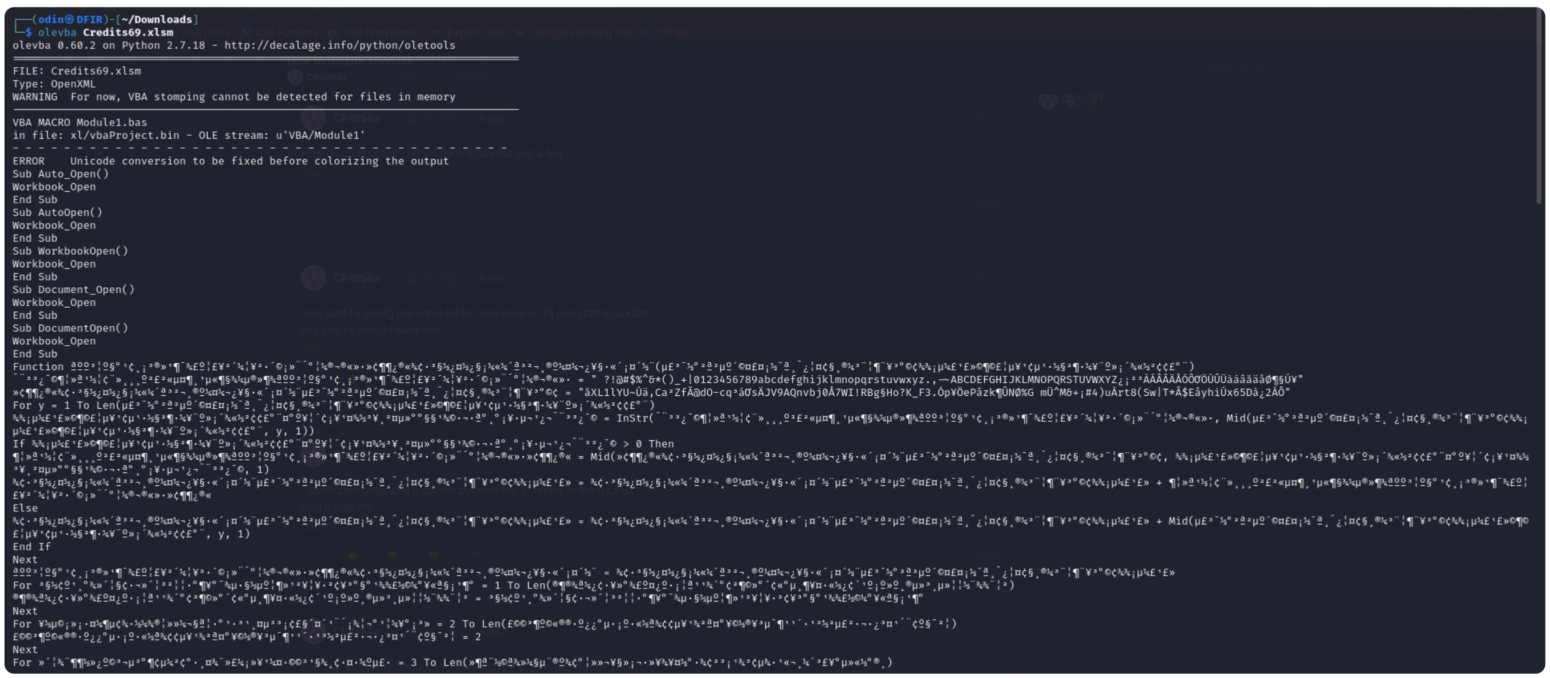
I extracted it to my machine and use ThunderBird to open the file:



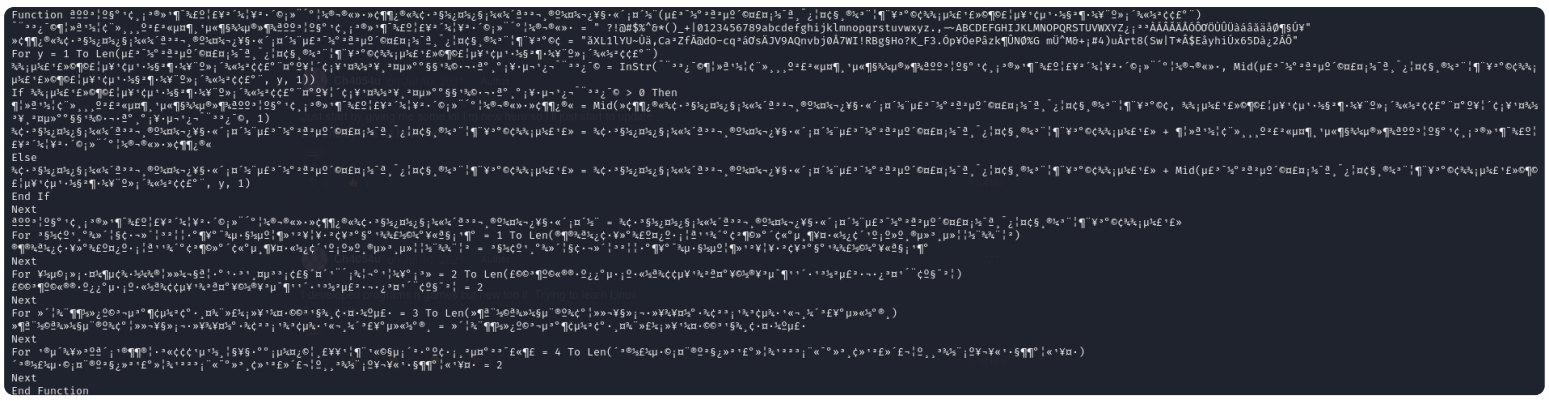
You can see that there's a zip file and the password is CreditsCardForFree. Now let's open this file and see what inside:



There's a xlsm file, and as usual, I always check VBA code inside by using olevba:

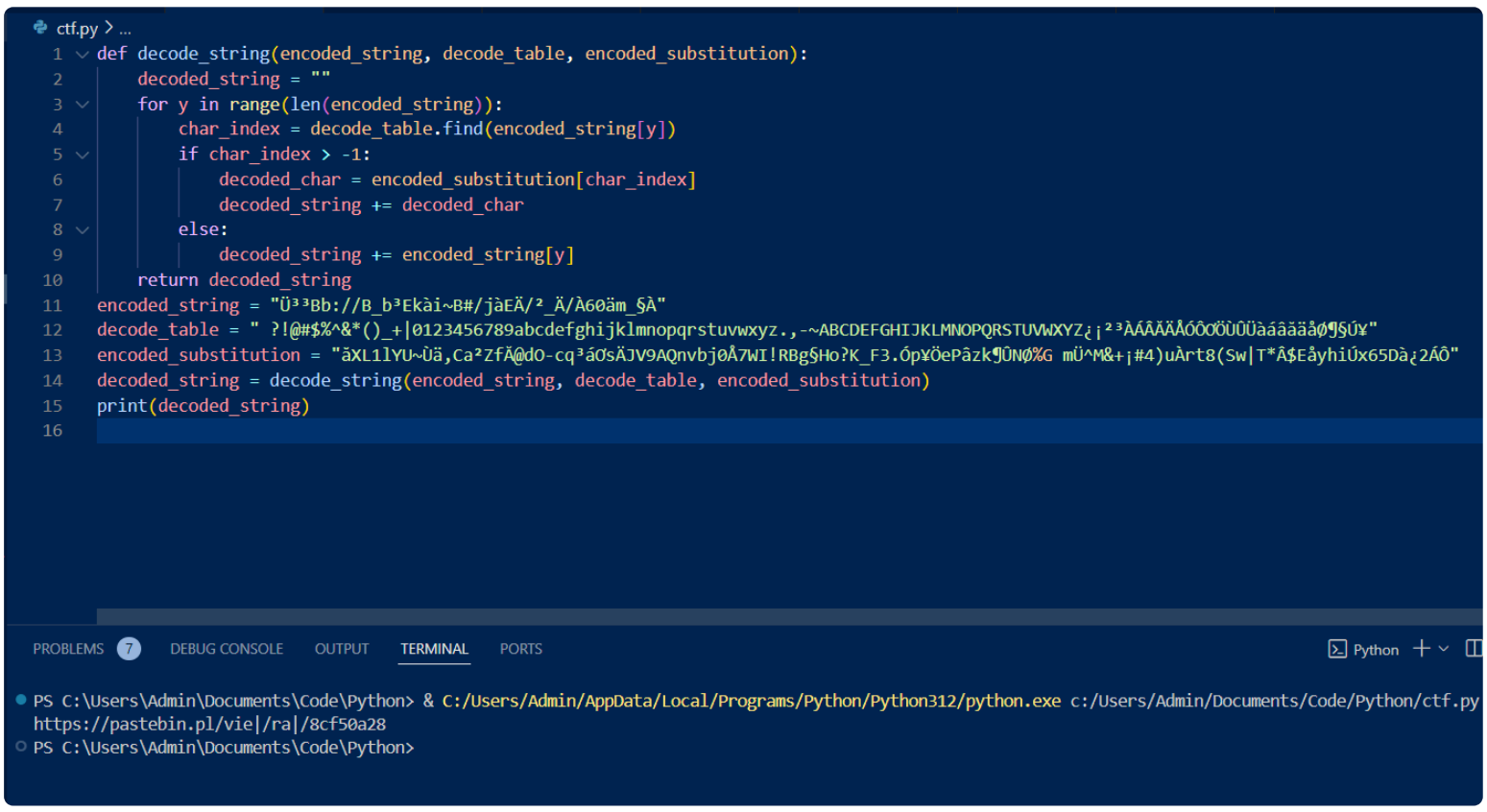


You can see that there's a VBA code and it's obfucated, and we don't any choice except deobfucate it by your hand or you can read code by Ctrl+F+the_name_of_func. After this I found that function will process a string looks like URL:



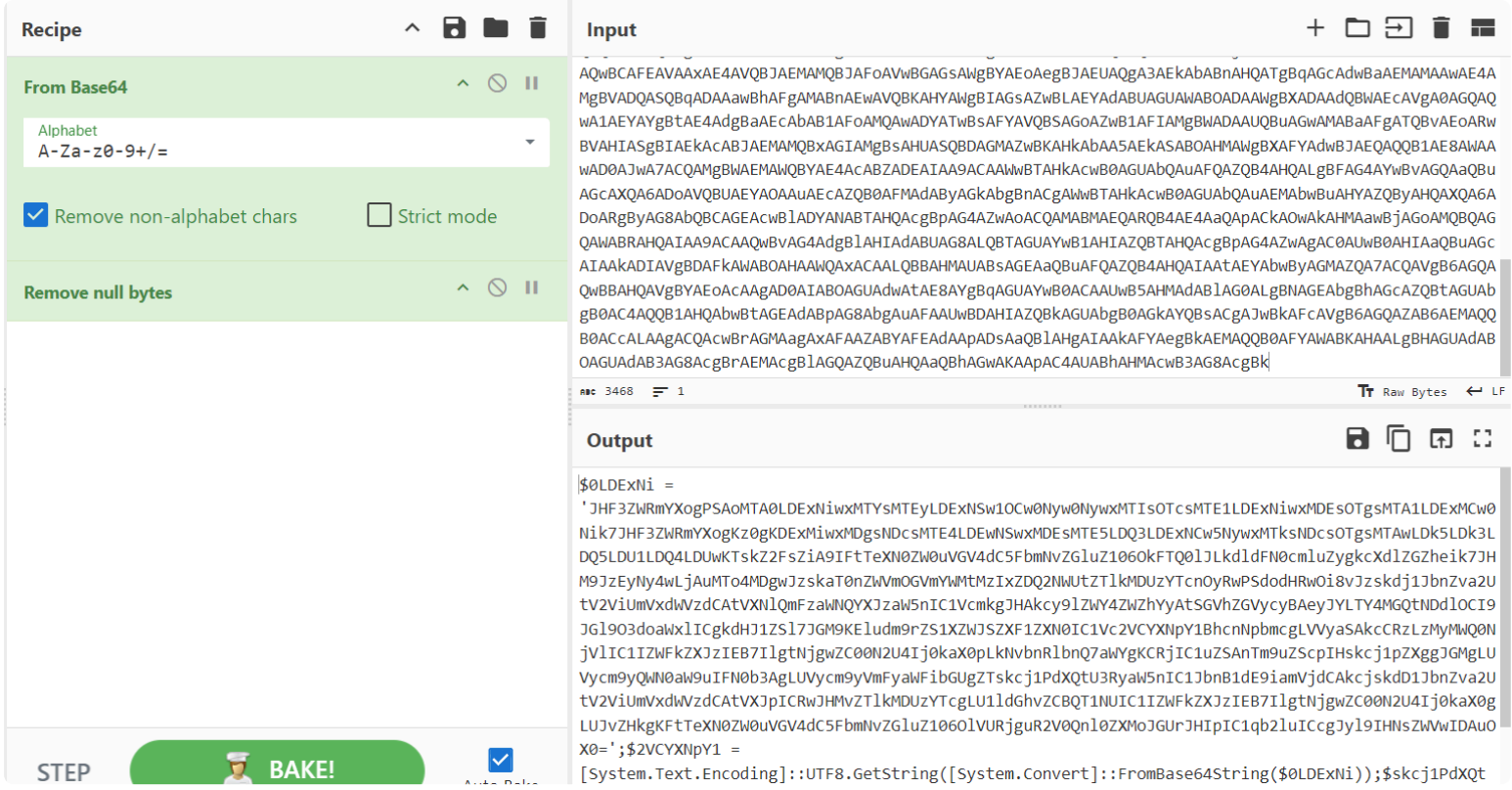
From here I can realise that our function are trying to decode that string. Based on their function, I rewrote a Python script for automatic decoding:

```
def decode_string(encoded_string, decode_table, encoded_substitution):
    decoded_string = ""
    for y in range(len(encoded_string)):
        char_index = decode_table.find(encoded_string[y])
        if char_index > -1:
            decoded_char = encoded_substitution[char_index]
            decoded_string += decoded_char
        else:
            decoded_string += encoded_string[y]
    return decoded_string
encoded_string = "Ü³³Bb://B_b³Ekài~B#/jàEÄ/²_Ä/À60äm_§À"
decode_table = " ?!@#$%^&*()_+|0123456789abcdefghijklmnopqrstuvwxyz.,-~ABCDEFGHIJKLMNOPQRSTU
encoded_substitution = "ăXL1lYU~Ùä,Ca²ZfĂ@dO-cq³áƠsÄJV9AQnvbj0Å7WI!RBg§Ho?K_F3.Óp¥ÖePâzk¶ÛNØ
decoded_string = decode_string(encoded_string, decode_table, encoded_substitution)
print(decoded_string)
```

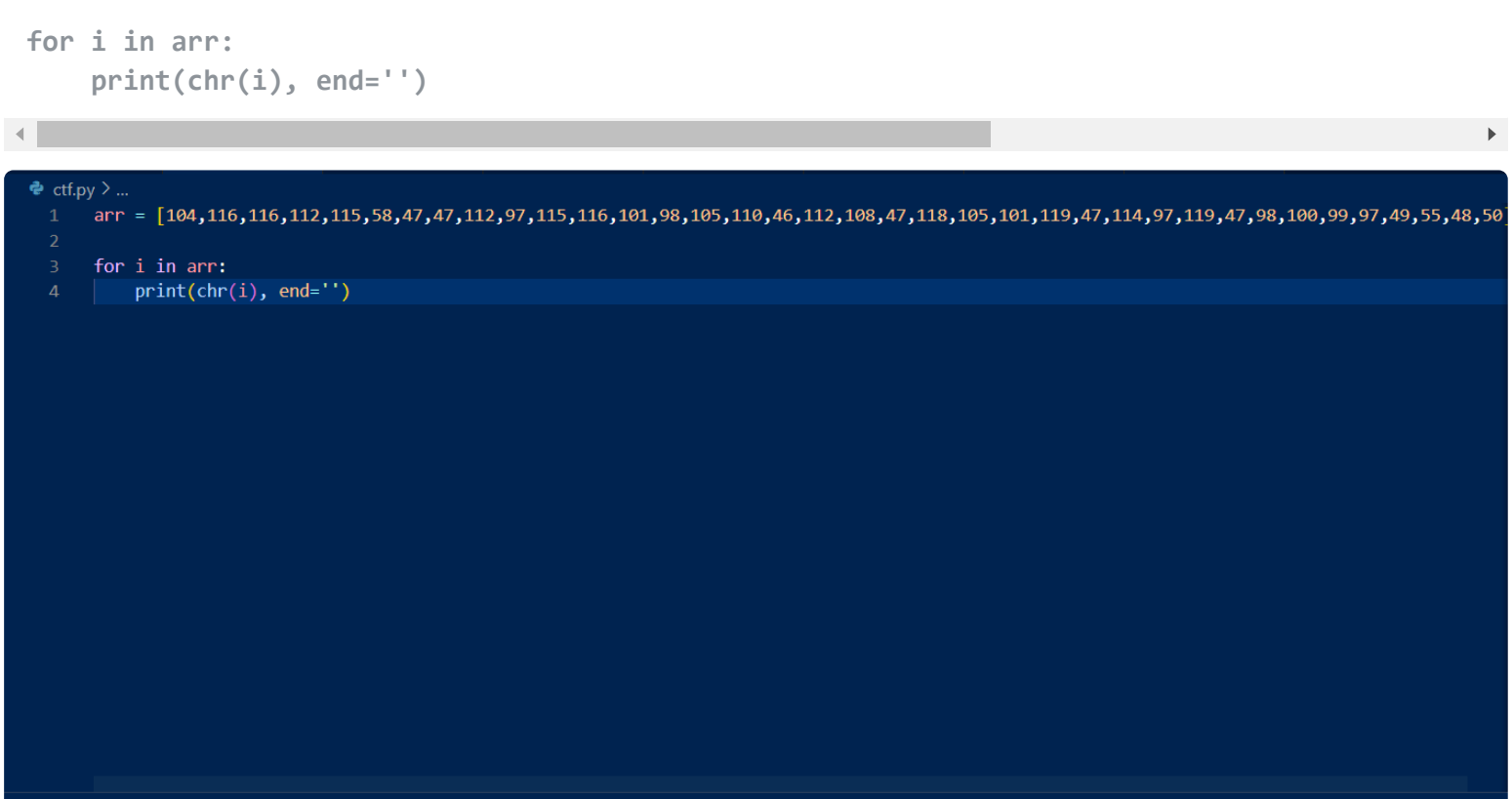


I got a link, now let's open it and see what inside:

You can see that there's a Powershell script and it will execute a command that was encoded by base64. Now we continue to decode base64 string:



There's a base64 string again. I decoded it and got one more script:

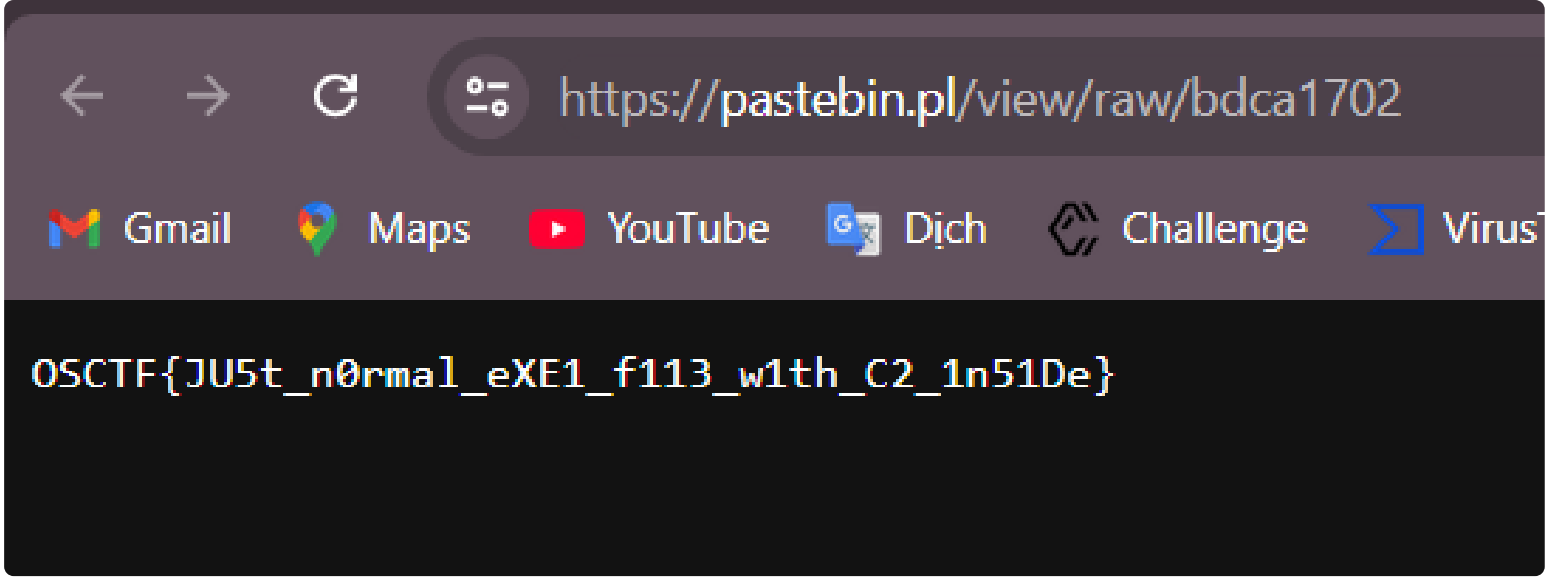


In $galf variable, it will take each elements in $qwedfaz and decode it to ascii character. From here I wrote a Python script again:

```
arr = [104,116,116,112,115,58,47,47,112,97,115,116,101,98,105,110,46,112,108,47,118,105,101,

for i in arr:
    print(chr(i), end='')
```


I got a link again, opened it and I got the flag:



Flag: OSCTF{JU5t_n0rmal_eXE1_f113_w1th_C2_1n51De}

Thank you for watching, hope you enjoy this. I solved other challenges but I still love forensic so I just wrote writeup for it 😂😂😂. See you in other CTFs, bye!!!

related posts

- DUCTF 2024 - Forensic
- UIUCTF 2024 - SoMeSINT writeup
- WaniCTF 2024 - Forensic

all tags

AES Decrypt AKASEC BITSCTF BYUCTF Blue Team CTFtime Command and Control DES3 decrypt DFIR DUCTF Email forensic Forensic Git log HackTheBox JavaScript KCSC Macros Memory Forensic MireaCTF Network Forensic OSCTF OSINT Powershell Real case Redis TheCyberCoopCTF TrueCrypt UIUCTF Virustotal Volatility WaniCTF WannaGame Wireshark Word Writeup