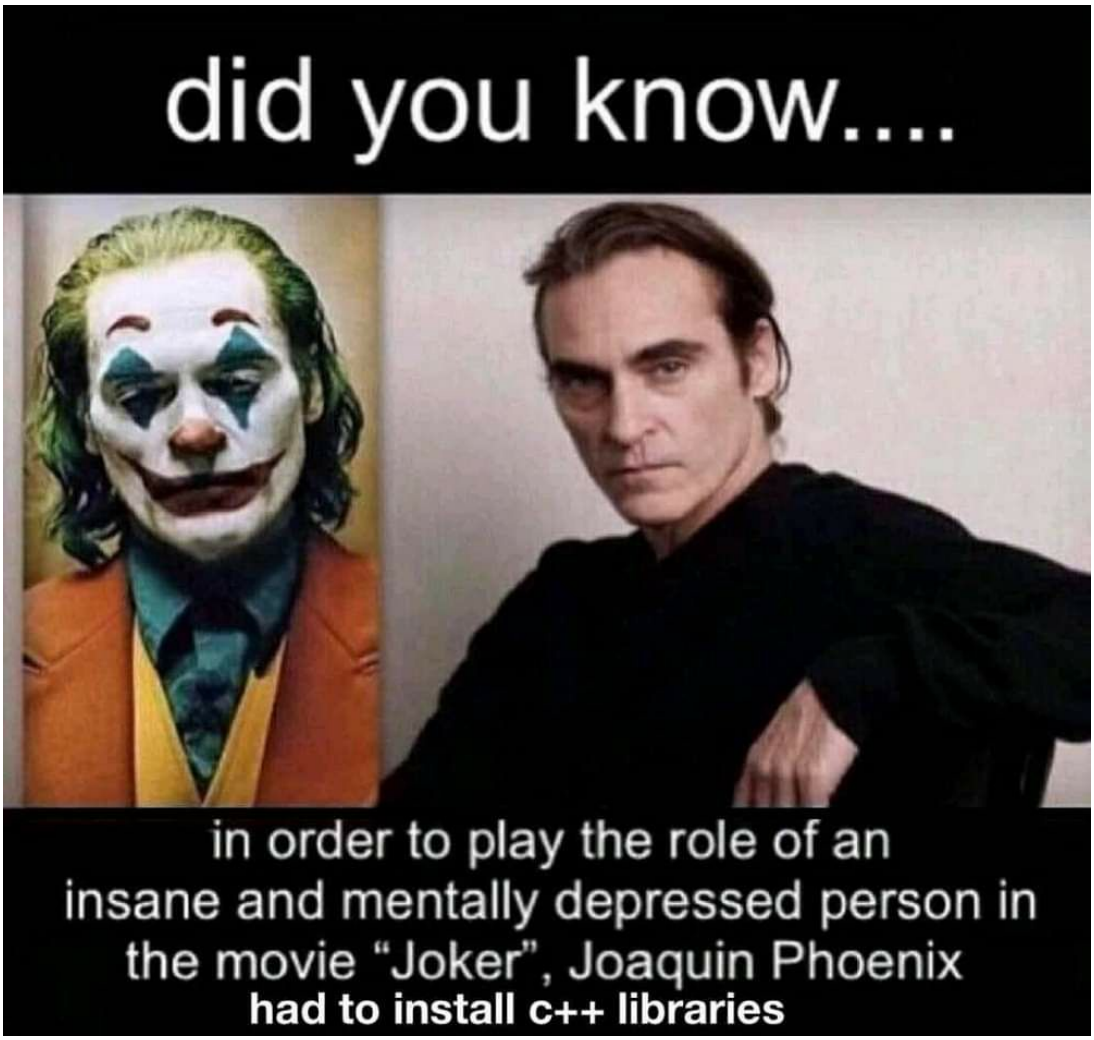
WriteUp COMPFEST13

PPKM LEVEL 1337



mbeerrr

ChaO

AnehMan

Cryptography

1. Snab? Yes, Snab

a. Executive Summary

Snab likes to give you a challenge. It is a simple challenge of RSA encrypted messages, and you only have to find out what those messages really are! Be careful though, Snab has some trick up his sleeve.

Author: houseoforion

b. Technical Report

Diberikan file .py dan output.txt. Berikut penampakannya

```
from Cryptodome.Util.number import*

e = 0x10001
s = pow(p + q, 2)
n = p*q
a = pow(s, 3, r)
b = (s - q*(2*p + q))*r

m_list = [findme]

c_list = []
for i in range(len(m_list)):
   m = bytes_to_long(m_list[i])
   c = pow(m*r, e, n)

   c_list.append(c)

output = open("output.txt", "w")
output.writelines([str(i) + "\n" for i in [e, s, n, a, b,
c_list]])
output.close()
```

Intinya, kita perlu mencari s, p, q, r. Mencari s cukup sederhana, hanya tinggal akarkan saja

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

```
import gmpy2
import codecs

gmpy2.get_context().precision = 1000

f = open("output.txt").read().strip().split("\n")
e = int(f[0])
s = int(f[1])
n = int(f[2])
a = int(f[3])
b = int(f[4])
c_arr = eval(f[5])

s = int(gmpy2.sqrt(s))
```

Hasilnya adalah
227710668211235386344951923287073134619953664501798990625359508856921854534439819561334870 10

Mencari p dan q, hanya mengikuti rumus diskrit ini.

$$p = \frac{s \pm \sqrt{s^2 - 4n}}{2}$$

```
q = (s + gmpy2.sqrt(s**2 - 4*n)) // 2
p = s-q
```

Setelah itu melakukan decrypt satu per satu

```
phi = (p-1)*(q-1)

d = int(inverse(e,phi))

m = []

for c in c_arr:
   m.append(pow(c,d,n))
```

Kami sempat mencoba untuk mencari r dengan cara membalik persamaan b, tetapi gagal. Akhirnya kami lakukan bruteforce untuk mencari nilai r. Kami menemukan string aneh ketika r = 19578

```
# brute r, ditemukan string mencurigakan ketika r -> 19578
for r in range(a, 20000):
   print(long_to_bytes(m[-1]//r), r)
```

```
b'  \x8b\xab\x8a\xfds\xc6\xd6\x95\xcf' 19577
b'    r += 1\n' 19578
b' \x1f\xb4\x98)6[\x1e9\xd1?' 19579
b' \x1fI\x12\xb0#\xc61\xb1<\xcd' 19580
```

Kemungkinan hasil yang di-encrypt adalah script, bukan flag. Jadi kami lanjut membagi hasil decrypt tadi dengan nilai r yang sudah didapat

```
script = b""
for c in m:
   script += long_to_bytes(c//r)

print(script.decode())
```

Hasil:

```
#Snab says good job! But you're not done yet
flag = findme
halfa = ''.join([flag[i] for i in range (0, len(flag), 2)])
halfb = ''.join([flag[i] for i in range (1, len(flag), 2)]
p = bytes_to_long(bytes(halfa, encoding = 'utf-8'))
q = bytes_to_long(bytes(halfb, encoding = 'utf-8'))
r = 0
while (not(isPrime(p) and isPrime(q))):
    p += 1
    q += 1
    r += 1
```

Intinya p dan q adalah flag, lalu p, q dan r ditambah 1 sampai nilai p dan q adalah bilangan prima. Karena nilai awal r adalah 0 dan nilai akhir r adalah 19577, jadi kita hanya perlu mengurangi p dan q dengan r. Flag didapat.

Full script:

```
from Crypto.Util.number import *
import gmpy2
import codecs

gmpy2.get_context().precision = 1000

f = open("output.txt").read().strip().split("\n")
```

```python
e = int(f[0])
s = int(f[1])
n = int(f[2])
a = int(f[3])
b = int(f[4])
c_arr = eval(f[5])

s = int(gmpy2.sqrt(s))

q = (s + gmpy2.sqrt(s**2 - 4*n)) // 2
p = s-q

phi = (p-1)*(q-1)

d = int(inverse(e,phi))

m = []

for c in c_arr:
   m.append(pow(c,d,n))

# brute r, ditemukan string mencurigakan ketika r -> 19578
# for r in range(a, 20000):
#     print(long_to_bytes(m[-1]//r), r)

r = 19578

script = b""
for c in m:
   script += long_to_bytes(c//r)

# print(script.decode())

pbytes = long_to_bytes(p-r)
qbytes = long_to_bytes(q-r)

f = ""
slam = 0
for i in range(len(pbytes) + len(qbytes)):
   if i%2 == 0:
```

```
        f += chr(pbytes[slam])
    else:
        f += chr(qbytes[slam])

    if len(f) % 2 == 0 and len(f) != 0:
        slam += 1

print(f)
```

Hasil:

```
anehman@ubuntu:~/ctf/compfest/2021/quals/crypto/snab/public$ python3 solve.py
Cool! You did it! {y0U_d1DnT_3xpEcT_t0_FinD_pQ_4s_a_fl4g_DiD_y0u_7e1877a801}
```

c. Flag

Flag:

COMPFEST13{y0U_d1DnT_3xpEcT_t0_FinD_pQ_4s_a_fl4g_DiD_y0u_7e1877a801}

2. You AES Me Up

a. Executive Summary

So I can stand on scoreboard~

nc 103.152.242.242 5592

Author: prajnapras19

b. Technical Report

Diberikan file .py, berikut penampakannya

```
#!/usr/bin/env python3
import sys
import os
import random
import binascii
from Crypto.Cipher import AES
from Crypto.Util.number import long_to_bytes, bytes_to_long
from secret import FLAG

IV = os.urandom(AES.block_size)
KEY = os.urandom(AES.block_size)

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)

def pad(msg):
```

```
    return msg + (chr(16 - len(msg) % 16) * (16 - len(msg) %
16)).encode()

def get_flag():
    flag = pad(FLAG)
    cipher = AES.new(IV, AES.MODE_ECB)
    flag = cipher.encrypt(flag)

    enc = b''
    flag = pad(flag)
    iv = IV
    for i in range(0, len(flag), 16):
        cipher = AES.new(KEY, AES.MODE_CBC, iv)
        enc += cipher.encrypt(flag[i:i+16])
        iv    =    long_to_bytes(bytes_to_long(enc[i:i+16])    ^
bytes_to_long(flag[i:i+16]))
    print('flag (in hex) =', binascii.hexlify(enc).decode())

def encrypt():
    msg = input('msg (in hex) = ')
    if (len(msg) % 2 != 0):
        print('Invalid input!')
        return
    msg = binascii.unhexlify(msg.encode())
    cipher = AES.new(KEY, AES.MODE_CBC, IV)
    enc = cipher.encrypt(pad(msg))
    print('enc (in hex) =', binascii.hexlify(enc).decode())

def decrypt():
    enc = input('enc (in hex) = ')
    if (len(enc) % 32 != 0):
        print('Invalid input!')
        return
    enc = binascii.unhexlify(enc.encode())
    cipher = AES.new(KEY, AES.MODE_CBC, IV)
    msg = cipher.decrypt(enc)
    print('msg (in hex) =', binascii.hexlify(msg).decode())

def menu():
    print('1. Get encrypted flag')
```

```
    print('2. Encrypt a message')
    print('3. Decrypt a message')
    print('4. Exit')

if __name__ == '__main__':
    while True:
        try:
            menu()
            choice = input('> ')
            if choice == '1':
                get_flag()
            elif (choice == '2'):
                encrypt()
            elif (choice == '3'):
                decrypt()
            elif (choice == '4'):
                print('Bye.')
                break
            else:
                print('Invalid input!')
        except:
            print('Something went wrong.')
            break
```

Intinya,

1. Flag di-encrypt 2 kali, yaitu dengan mode ECB dan mode PCBC
2. Key pada mode ECB == IV pada mode PCBC
3. Menu encrypt dan decrypt menggunakan algoritma AES mode CBC

Yang harus dilakukan adalah

1. Membalikan mode PCBC melalui menu decrypt
2. Cari IV
3. Decrypt lagi dengan mode ECB

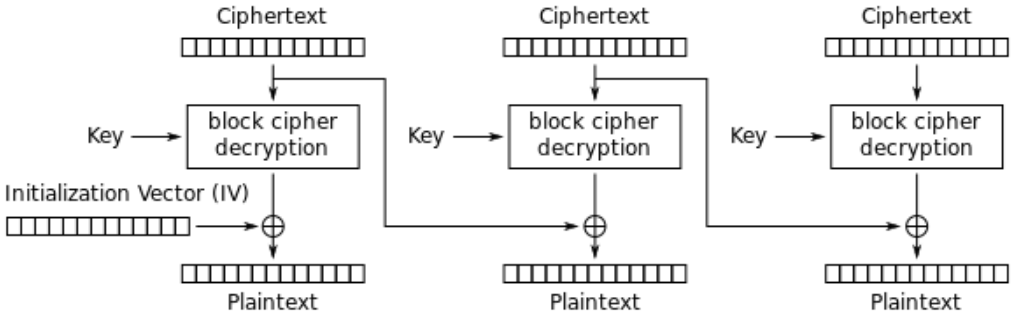
Membalikkan mode PCBC melalui menu decrypt cukup mudah

1. get_flag()
2. decrypt(get_flag())
3. Block ke-(n-1) dari hasil decrypt di-XOR dengan block ke-n dari hasil decrypt

Contoh script ada dibawah

```
from Crypto.Util.number import *
from Crypto.Cipher import AES
from binascii import unhexlify
from pwn import *

# p = process("./chall.py")
p = remote("103.152.242.242",5592)

p.sendline("1")
p.recvuntil("flag (in hex) = ")
flag_enc = p.recvline().strip()

p.sendline("3")
p.sendline(flag_enc)
p.recvuntil("msg (in hex) = ")
flag_dec = p.recvline().strip()

flag_enc = [flag_enc[i:i+32] for i in range(0,len(flag_enc),32)]
flag_dec = [flag_dec[i:i+32] for i in range(0,len(flag_dec),32)]

flag_p1 = [unhexlify(flag_dec[0])]

# PCBC -> CBC
for i in range(1,len(flag_enc)):
   flag_p1.append(xor(flag_p1[i-1], unhexlify(flag_dec[i])))
```

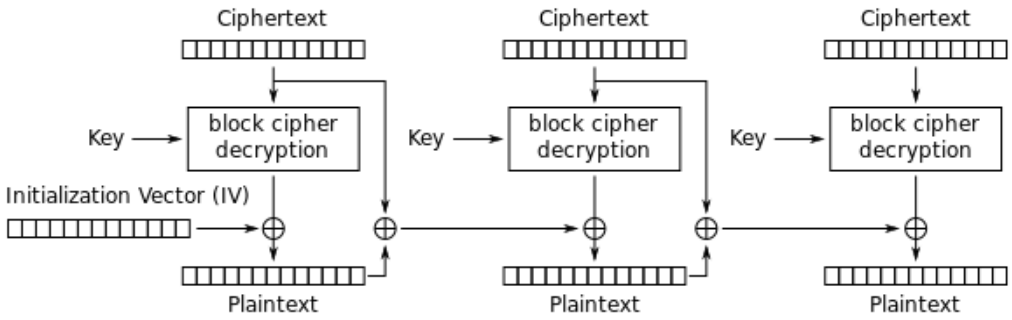
Selanjutnya adalah mencari IV. Gambar dibawah adalah proses dekripsi mode CBC



Pada blok pertama, jika hasil dekripsi AES pada blok pertama sebelum dilakukan XOR diberi nama variabel `ct_d` dan blok plaintext pertama diberi nama `pt1`, maka kurang lebih persamaannya sebagai berikut:

```
pt1 = ct_d ^ IV
```

Gambar dibawah adalah proses dekripsi dari mode PCBC



Pada blok kedua, jika hasil dekripsi AES pada blok kedua sebelum dilakukan XOR diberi nama variabel `ct_d` dan plaintext blok kedua diberi nama `pt2`, maka kurang lebih persamaannya sebagai berikut:

```
pt2 = ct_d ^ ct_pcbc[0] ^ pt_pcbc[0]

Ket:
ct_pcbc[0] -> blok ciphertext ke-0
pt_pcbc[0] -> blok plaintext ke-0
```

Sekarang jika pt1 dan pt2 di-XOR, maka persamaannya akan seperti berikut:

```
pp = pt1 ^ pt2
pp = ct_d ^ IV ^ ct_d ^ ct_pcbc[0] ^ pt_pcbc[0]
pp = IV ^ ct_pcbc[0] ^ pt_pcbc[0]
```

Karena `ct_pcbc[0] ^ pt_pcbc[0]` sudah diketahui, maka IV bisa didapat dengan melakukan xor `pp` dengan `ct_pcbc[0] ^ pt_pcbc[0]`. Jadi yang harus dilakukan adalah:

1. decrypt(blok flag_enc ke-2)
2. Implementasi persamaan diatas

Script:

```
# find IV
p.sendline("3")
p.sendline(flag_enc[1])
p.recvuntil("msg (in hex) = ")
```

```
one_block = unhexlify(p.recvline().strip())

ctpt = xor(unhexlify(flag_enc[0]), unhexlify(flag_dec[0]))
pp = xor(one_block, flag_p1[1])
iv = xor(ctpt, pp)
```

Sekarang hanya tinggal menggunakan IV tadi sebagai KEY untuk decrypt flag. Perlu diperhatikan, karena flag di-padding 2 kali, jadi blok terakhir tidak dipakai.

Full script:

```
from Crypto.Util.number import *
from Crypto.Cipher import AES
from binascii import unhexlify
from pwn import *

# p = process("./chall.py")
p = remote("103.152.242.242",5592)

p.sendline("1")
p.recvuntil("flag (in hex) = ")
flag_enc = p.recvline().strip()

p.sendline("3")
p.sendline(flag_enc)
p.recvuntil("msg (in hex) = ")
flag_dec = p.recvline().strip()

flag_enc = [flag_enc[i:i+32] for i in range(0,len(flag_enc),32)]
flag_dec = [flag_dec[i:i+32] for i in range(0,len(flag_dec),32)]

flag_p1 = [unhexlify(flag_dec[0])]

# PCBC -> CBC
for i in range(1,len(flag_enc)):
   flag_p1.append(xor(flag_p1[i-1], unhexlify(flag_dec[i])))


# find IV
p.sendline("3")
```

```python
p.sendline(flag_enc[1])
p.recvuntil("msg (in hex) = ")
one_block = unhexlify(p.recvline().strip())

ctpt = xor(unhexlify(flag_enc[0]), unhexlify(flag_dec[0]))
pp = xor(one_block, flag_p1[1])
iv = xor(ctpt, pp)

# decrypt using AES mode ECB
flag_pcbc = b"".join(flag_p1)
cipher = AES.new(iv, AES.MODE_ECB)
flag = cipher.decrypt(flag_pcbc)
print(flag)
```

Hasil:

```
anehman@ubuntu:~/ctf/compfest/2021/quals/crypto/you_AES/public$ python3 solve.py
[+] Opening connection to 103.152.242.242 on port 5592: Done
b'COMPFEST13{Y0u_aes_me_Uppppppp_____t0_c0dE_on_st0rmy_Seaaaas____e0212d1a34}\x05
\x05\x05\x05\x05\xc3\x86o\xb3cFH\x15 \xa0P\xaf\x17\x9c2\x80'
[*] Closed connection to 103.152.242.242 port 5592
anehman@ubuntu:~/ctf/compfest/2021/quals/crypto/you_AES/public$
```

c. Flag

Flag:

COMPFEST13{Y0u_aes_me_Uppppppp_____t0_c0dE_on_st0rmy_Seaaaas____e0212d1a34}

Forensic

1. VidCap

a. Executive Summary

Found this pcap of my ex's network traffic. I knew they're streaming video but I can't extract it. Can you help me ?

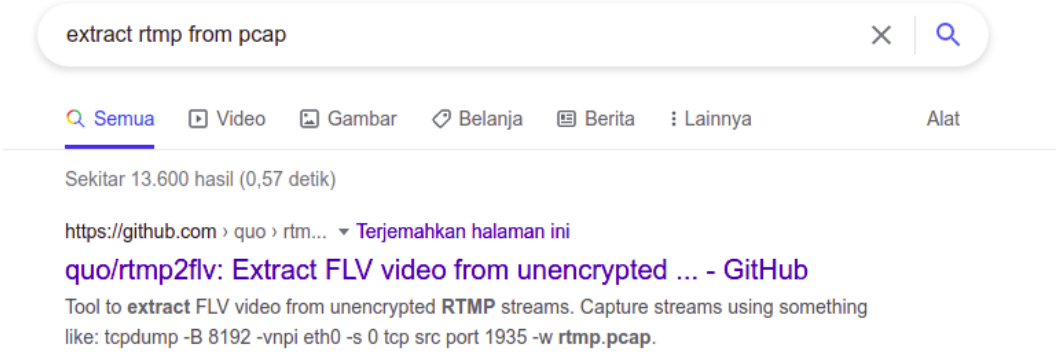
Author: xMaximusKl

b. Technical Report

Diberikan file .pcap, berikut penampakannya

```
RTMP          161 onStatus('NetStream.Publish.Start')
TCP            44 55015 → 1935 [ACK] Seq=3438 Ack=3483 Win=62013 Len=0
RTMP          455 @setDataFrame()
TCP            44 1935 → 55015 [ACK] Seq=3483 Ack=3849 Win=61647 Len=0
RTMP           63 Audio Data
TCP            44 1935 → 55015 [ACK] Seq=3483 Ack=3868 Win=61628 Len=0
RTMP          113 Video Data
TCP            44 1935 → 55015 [ACK] Seq=3483 Ack=3937 Win=61559 Len=0
TCP          4152 55015 → 1935 [PSH, ACK] Seq=3937 Ack=3483 Win=62013 Le
TCP            44 1935 → 55015 [ACK] Seq=3483 Ack=8045 Win=57451 Len=0
RTMP           78 Video Data
```

Pada protokol RTMP, terdapat video data dan audio data. Karena kami ingin mencoba mengekstrak semua data yang ada di RTMP tapi tidak tau caranya, jadi kami mencoba search dengan keyword “extract rtmp from pcap”. Untungnya, kami langsung menemukan tools yang diinginkan.



Jadi tinggal ikuti dari instruksi yang ada di github. Instruksi pertama adalah cara membuat file .pcap. Karena kita sudah ada file .pcap, jadi instruksi tersebut bisa diabaikan. Lalu ketik

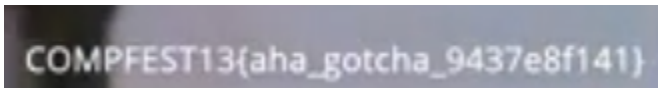
```
tcpflow -T %T_%A%C%c.rtmp -r capture.pcapng
```

Untuk meng-extract semua RTMP stream. Setelah itu, kita bisa convert RTMP ke FLV dengan cara

```
./rtmp2flv.py *.rtmp
```

Berikut hasilnya

```
2021-04-11T11:13:56Z_192.168.018.010c1.rtmp
2021-04-11T11:13:56Z_192.168.018.010.rtmp
2021-04-11T11:13:56Z_192.168.018.010.rtmp.1.flv
2021-04-11T11:14:22Z_127.000.000.001c1.rtmp
2021-04-11T11:14:22Z_127.000.000.001.rtmp
capture.pcapng
report.xml
rtmp2flv.py
```



*tak apa kena rickroll, yang penting flag *hiks*

c. Flag

Flag: **COMPFEST13{aha_gotcha_9437e8f141}**

Misc

1. Sanity Check

a. Executive Summary

COMPFEST13{Welcome_to_CTF_COMPFEST_13}

b. Technical Report

Tinggal submit saja h3h3

c. Flag

Flag: **COMPFEST13{Welcome_to_CTF_COMPFEST_13}**

2. Promotional Video

a. Executive Summary

Marketing Committee: Can you show this video to your participants?

CTF committee: Ok, no problem.

Marketing Committee: Are all your participants use English as their first language?

CTF committee: No, but we can fix that easily. Don't worry!

https://youtu.be/047T5AZpOiI

Author: prajnapras19

b. Technical Report

Cara dapet flag:

1. Extract subtitle di https://9xbuddy.com/
2. Download
3. ???
4. PROFIT!!

```
Don't forget to follow our social media and visit our website (link in description)

C

O

M

P

F

E

S

T

1
```

c. Flag

Flag: **COMPFEST13{c4ptUr3_Th3_Fl4g_cb1217bccd}**

3. Baby JS

a. Executive Summary

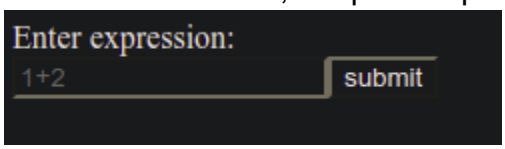
We create JS console emulator. Could you read our platform code?

http://103.152.242.243:5535/

Author: Bonceng

b. Technical Report

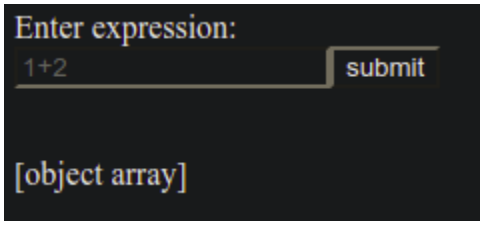
Diberikan URL web, tampilan seperti dibawah



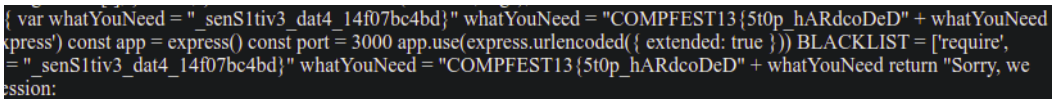
Kami langsung menebak kalau ini soal eval(). Jadi langsung saja coba `system('ls')` muncul error seperti berikut

```
ReferenceError: system is not defined
    at eval (eval at <anonymous> (/usr/src/app/index.js:87:19), <anonymous>:1:1)
    at /usr/src/app/index.js:87:19
    at Layer.handle [as handle_request] (/usr/src/app/node_modules/express/lib/router/layer.js:95:5)
    at next (/usr/src/app/node_modules/express/lib/router/route.js:137:13)
    at Route.dispatch (/usr/src/app/node_modules/express/lib/router/route.js:112:3)
    at Layer.handle [as handle_request] (/usr/src/app/node_modules/express/lib/router/layer.js:95:5)
    at /usr/src/app/node_modules/express/lib/router/index.js:281:22
    at Function.process_params (/usr/src/app/node_modules/express/lib/router/index.js:335:12)
    at next (/usr/src/app/node_modules/express/lib/router/index.js:275:10)
    at /usr/src/app/node_modules/body-parser/lib/read.js:130:5
```

Ternyata web menggunakan javascript. Jadi langsung coba `require("child_process")`. Eh difilter :(
Jadi kami memutuskan untuk dump semua object yang ada dengan cara `Object.entries(this)`. Hasilnya adalah sebagai berikut



Tinggal ubah ke string dengan `Object.entries(this).toString()`. Flag keluar



c. Flag

Flag: **COMPFEST13{5t0p_hARdcoDeD_senS1tiv3_dat4_14f07bc4bd}**