

WreckIT 4.0 Writeup

By: Nekat Banetz



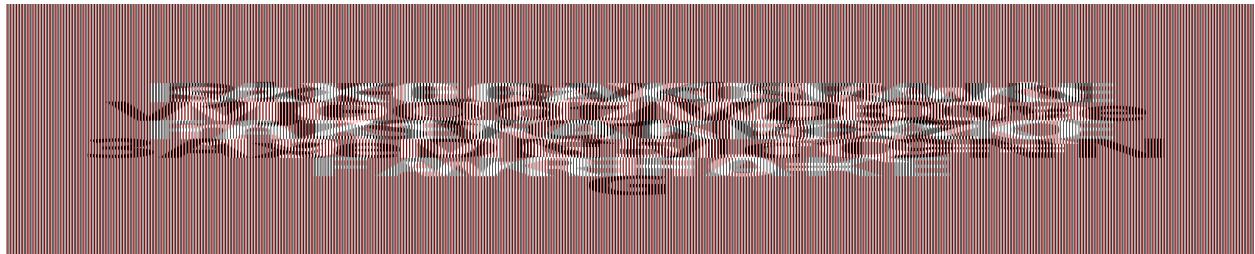
I. Forensic

Problem 1: Mixxedup

File c.jpg merupakan file jpg dan zip sekaligus. Dengan menjalankan tool unzip, Ada 2 file hasil extract, yaitu dobleh.txt dan flag.png.

dobleh.txt berisi: "saya aslinya 400, sekarang 2000"

flag.png berisi:



Berdasarkan hint dari dobleh.txt, sebenarnya width yang asli adalah 400px, sedangkan yang kita dapatkan adalah 2000px. Maka dari itu, image di resize dengan GIMP (atau photo editor lainnya) dimana width menjadi 400, tanpa menggunakan interpolation. Hasilnya:



Waduh, ternyata ada salah. Jika dilihat image original nya, terdapat suatu efek "interlacing" vertikal. Pada saat kita langsung resize, lompatan pixel yang dipakai adalah lompatan pertama. Ternyata, kuncinya adalah meng-offset pixel dari image nya untuk mengelabui GIMP untuk memakai lompatan pixel yang kita inginkan.

Dari image original, setelah di offset sebanyak 4 pixel ke kiri, lalu mengulang resize, maka akan dapat hasil:



Tulisan ini menyerupai base64, namun kode ini kurang lengkap.

Dari image original, setelah di offset sebanyak 8 pixel ke kiri, lalu mengulang resize, maka akan dapat hasil:

**V1JFQ0tJVDQwe
3AxeDNMc19NN
G**

Setelah tulisan ini digabung dengan tulisan yang sebelumnya, maka akan menjadi:

V1JFQ0tJVDQwe3AxeDNMc19NNszX00zX0MwbmZ1NTNkXzQwRH0=

Dan tentu saja, hasil decode string ini menjadi flag:

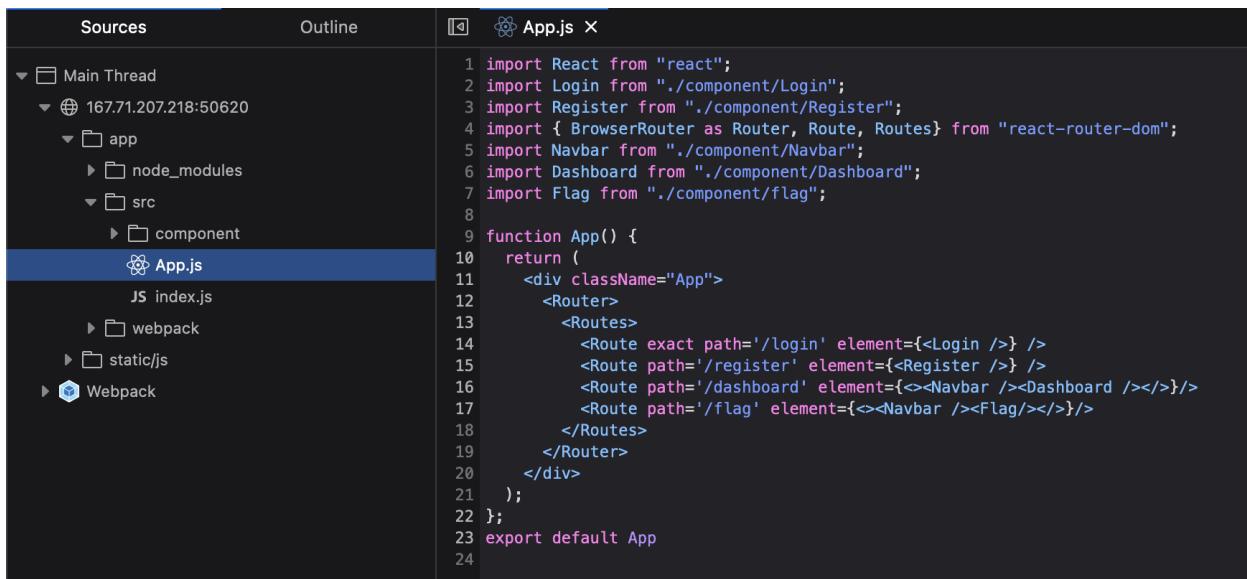
WRECKIT40{p1x3Ls_M4k3_M3_C0nfu53d_40D}

II. Web

Problem 1: jwttt

Situs <http://167.71.207.218:50620/> mengarah ke aplikasi berbasis React.

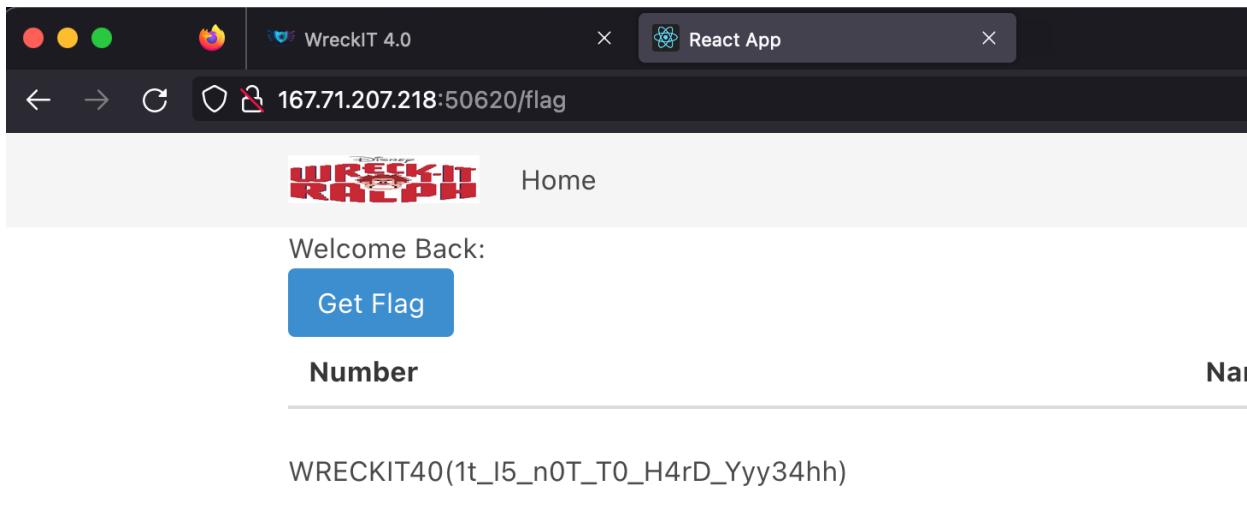
Menggunakan DevTools Firefox, terdapat source file bernama App.js.



The screenshot shows the Firefox DevTools Sources tab. The left sidebar lists the project structure: Main Thread (167.71.207.218:50620), app (node_modules, src, component, App.js, index.js, webpack, static/js), and Webpack. The App.js file is selected. The right pane displays the code for the App.js file:

```
1 import React from "react";
2 import Login from "./component/Login";
3 import Register from "./component/Register";
4 import { BrowserRouter as Router, Route, Routes } from "react-router-dom";
5 import Navbar from "./component/Navbar";
6 import Dashboard from "./component/Dashboard";
7 import Flag from "./component/flag";
8
9 function App() {
10   return (
11     <div className="App">
12       <Router>
13         <Routes>
14           <Route exact path='/login' element={<Login />} />
15           <Route path='/register' element={<Register />} />
16           <Route path='/dashboard' element={<><Navbar /><Dashboard /></></>} />
17           <Route path='/flag' element={<><Navbar /><Flag /></></>} />
18         </Routes>
19       </Router>
20     </div>
21   );
22 }
23 export default App;
24
```

Ternyata, ada route /flag, dan ketika di pakai...



The flag is there all along! Kami pikir ini merupakan unintended solution, karena solusi kami tidak menggunakan jwt sama sekali.

WRECKIT40(1t_I5_n0T_T0_H4rD_Yyy34hh)

III. Crypto

Problem 1: CRYPTO Free Flag

Diberikan sebuah file dengan isi seperti berikut

```
0011010001100001001101000110000100110100001100110011010100110110001101  
000011010100110101001101000011001100110101001110010011010001101000110  
0010001101010110000100110100001101000011010100110100001101000011100100  
110100011001000011010000110100001100110011001100110011000110001000110100  
001100100011010000110011001001101000110011001100110011000110001000110100  
0100110011001100100011010000111000001101000011100100110100011000010011  
01010011000000110100110111001101000110011000110101001110010011010100  
11010000110101011000010011010001100100011000011000010011010100110001  
00110100111000001101000011100100110011001101001101001101000011010001101  
00001100100011010001100011001100110011001100110000110100001101000011  
1000001101010011000100110011001100100011010000110011001100110011011100  
110100011001100011010000110110001101001000110100001101000011011100110100  
00110011001100110011010001100110011001000110010000110010001100100001101  
01001101000011010100110100011011100011010000110110001100110011001100011  
00110011011000110100001101000011010000110010001100100011001000011010100  
110110001101010000110100001101000011010000110100001100110011001100110101  
0011010000110011001000011010000110100001100110011001000110000100110000100  
1100110101001101000011011100011001100110010000110100001101000011011000110  
0011001101000011010000110100001101000011010000110100001101000011010000110  
11010000110111001101000011001100110011001000011001100110011001100110101  
0011011000110100011001100011010000110010001100100011001000110010000110100  
0000110100001101010011001100110100011001100110011001100100011001100110011  
010100110000001101010011010001100110011001000011001100110011001000011001100  
100100001100110110010000110011011001000011001100110011001000011001101000100
```

Berdasarkan dseripsi soal yang diberikan “BiHB32R13” bisa kita ketahui urutan encoding nya yaitu Biner -> Hex -> Base32 -> ROT13.

Berikut adalah script python yang digunakan untuk solve soal diatas

```
from binascii import unhexlify  
from base64 import b32decode  
import codecs
```

```
enc = open('./soal.secret', 'r').read().strip()

def bin2str(s):
    return ''.join(chr(int(s[i*8:i*8+8], 2)) for i in range(len(s)//8))

decode_bin = bin2str(enc) # Decode bin to str
decode_hex = unhexlify(decode_bin) # Decode Hex to str
decode_b32 = b32decode(decode_hex) # Decode base32
flag = codecs.decode(decode_b32.decode('latin1'),
'rot_13') # Decode rot-13

print(flag)
```

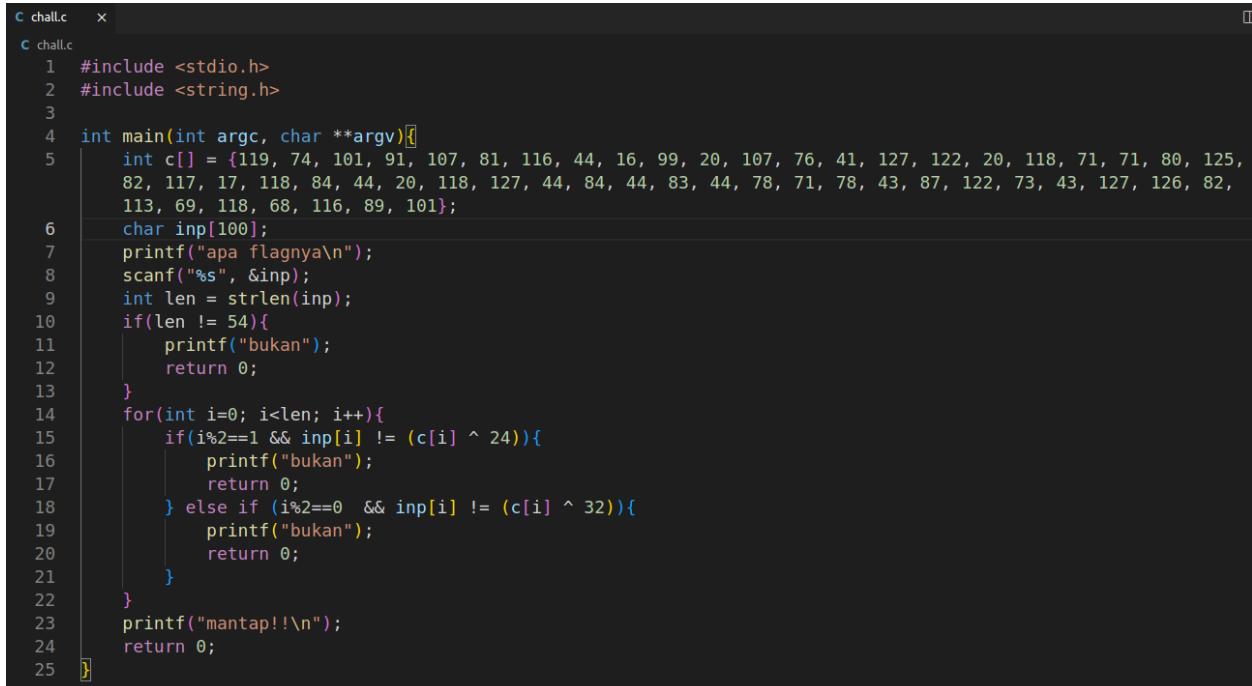
Secara garis besar kita decode dulu dari biner ke string ternyata setelah di decode kita mendapatkan hex string, selanjutnya kita decode juga dari hex string ke ascii setelah dilakukan kita mendapatkan base32, setelah dilakukan decode dengan base32 didapatkan enc dari rot13. Langkah terakhir kita tinggal decode dari rot13 ke flag yang ada.

FLAG : WRECKIT40{CRYPTO_tolongin_aku_dong!!,_kurangPemanasan_hehehe}

IV. Rev

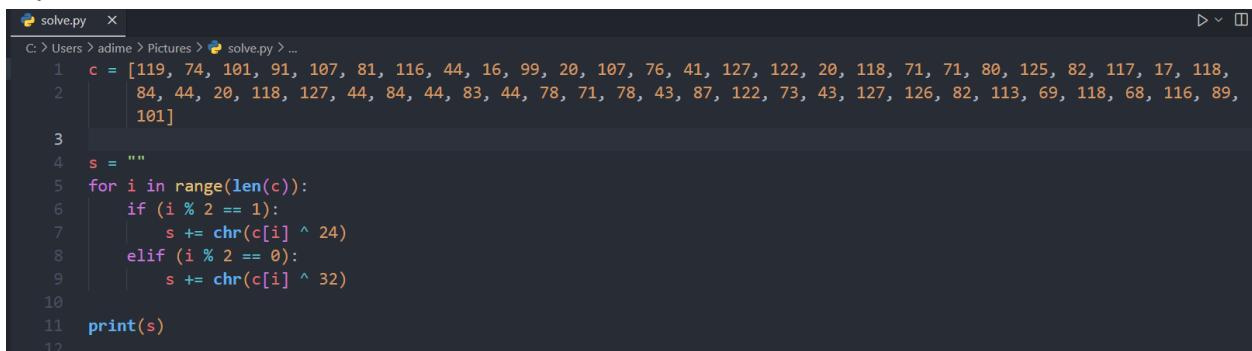
Problem 1: REV Free Flag

Nah disini dikasi soal yang terdapat file `chall.py` dan kita buka dengan menggunakan Visual Studio Code dan hasilnya sungguh mencengangkan



```
c chall.c
c chall.c
1 #include <stdio.h>
2 #include <string.h>
3
4 int main(int argc, char **argv){
5     int c[] = {119, 74, 101, 91, 107, 81, 116, 44, 16, 99, 20, 107, 76, 41, 127, 122, 20, 118, 71, 71, 80, 125,
6     82, 117, 17, 118, 84, 44, 20, 118, 127, 44, 84, 44, 83, 44, 78, 71, 78, 43, 87, 122, 73, 43, 127, 126, 82,
7     113, 69, 118, 68, 116, 89, 101};
8     char inp[100];
9     printf("apa flagnya\n");
10    scanf("%s", &inp);
11    int len = strlen(inp);
12    if(len != 54){
13        printf("bukan");
14        return 0;
15    }
16    for(int i=0; i<len; i++){
17        if(i%2==1 && inp[i] != (c[i] ^ 24)){
18            printf("bukan");
19            return 0;
20        } else if (i%2==0 && inp[i] != (c[i] ^ 32)){
21            printf("bukan");
22            return 0;
23        }
24    }
25 }
```

Nah cuss buatkan file Python untuk solve yang dimana variabel `c` bertipe data array dan berisi angka-angka nah lalu dibuatkan looping untuk panjang data nah lalu dikasi tipe kondisi-kondisi jika angka di modulus dua hasilnya 1 maka xor 24 dan kondisi kedua jika angka di modulus dua hasilnya 0 maka xor 32 lalu masukan ke dalam variabel baru seperti kode di bawah ini



```
解决.py
解决.py
1 c = [119, 74, 101, 91, 107, 81, 116, 44, 16, 99, 20, 107, 76, 41, 127, 122, 20, 118, 71, 71, 80, 125, 82, 117, 17, 118,
2 84, 44, 20, 118, 127, 44, 84, 44, 83, 44, 78, 71, 78, 43, 87, 122, 73, 43, 127, 126, 82, 113, 69, 118, 68, 116, 89,
3 101]
4 s = ""
5 for i in range(len(c)):
6     if (i % 2 == 1):
7         s += chr(c[i] ^ 24)
8     elif (i % 2 == 0):
9         s += chr(c[i] ^ 32)
10
11 print(s)
12
```

Nah kita mendapat flagnya

FLAG : WRECKIT40{4s11_b4ng_perm1nt44n_4t4s4n_n3wbi3_friendly}

V. Misc

Problem 1: Welcome

Just a greeting from WreckIT 4.0 welcoming all participants. Flag is also attached, thanks!

WRECKIT40{J4NG4N_lupa_Absen_YGYGY}

Problem 2: Rabithole

Terdapat hint yaitu Matryoshka Doll, sebuah boneka Rusia dimana didalamnya terdapat boneka lebih kecil, dan boneka tersebut terdapat boneka lebih kecil lagi, dan seterusnya sampai lebaran monyet. Prinsip yang sama terdapat dalam file 1000.zip ini. 1000.zip terdapat file pw999.txt yang berisi password, dan 999_password.zip yang menggunakan password dari text file itu, dimana isinya adalah zip file berikutnya.

1000.zip/

```
|   pw999.txt  
|   999_password.zip/  
|       999.zip/  
|       ...
```

Tentu saja kita tidak meng-extract nya satu-satu, jadi saya menggunakan script ini:

```
doll.py > ...  
1  import zipfile  
2  
3  master_zip = zipfile.ZipFile("1000.zip")  
4  master_zip.extractall()  
5  
6  for i in range(999, 0, -1):  
7      pw = open(f'pw{i}.txt', "r").read()  
8      print(pw)  
9      z = zipfile.ZipFile(f'{i}_password.zip')  
10     z.extractall(pwd=pw.encode())  
11     z = zipfile.ZipFile(f'{i}.zip')  
12     z.extractall()  
13
```

Pada akhirnya, flag.txt muncul yang berisi:

575245434b495434307b215f483070335f755f6431646e27375f64305f69375f
6d344e75343131795f3430447d

Ketika seri bilangan hexadecimal ini di-konversi menjadi ASCII, akan menjadi flag:

WRECKIT40{!_H0p3_u_d1dn'7_d0_i7_m4Nu411y_40D}

Problem 3: Hide and Seek on Zero Day

CHALLENGE 39 SOLVES X

Hide and Seek on Zero Day

100

Dawn bermain sebuah permainan yang sangat menyenangkan sekali bersama dengan teman terbaiknya di abad ini yaitu Snork. Permainan yang kita lakukan adalah petak umpet luar biasa dimana Snork akan bersembunyi di planet Bumi sedangkan tugas Dawn adalah mencari dimanakah Snork saat ini. Karena Dawn sangat pintar dia sebelumnya sudah menempelkan kamera dan alat pendekripsi koordinat di baju Snork. Kemudian sesaat sebelum kedua alat tersebut kehabisan data, alat itu meninggalkan beberapa data yang sangat berguna yang telah dia sembunyikan dalam secret note.

Sebelum itu, karena mereka berdua berteman baik, Dawn juga tahu bahwa Snork sangat suka meninggalkan pesan pada suatu tempat saat dia sedang bermain permainan. Dapatkah kamu mengetahui pesan yang ditinggalkan oleh Snork?

author: wondPing

[Download park.jpg](#) [Download secret.txt](#)

Diberikan dua buah file yaitu park.jpg dan secret.jpg. Pada park.jpg diberikan foto sebuah taman antah berantah. Pada secret.txt diberikan file base16. Base 16 tersebut kita decode dan mendapat text yang merupakan base64. Setelah di decode beberapa kali menggunakan base16 dan base64 secara bergantian maka didapatkan

From To

Hexadecimal Text

Paste hex numbers or drop file

```
657930324c6a67784e4455314d7a63304e6a6b314f5445354d7977674d5445  
774c6a67314d4451334e6a6b334d4455774e444d3566513d3d
```

Character encoding

ASCII

```
ey02LjgxNDU1Mzc0Njk1OTE5MywgMTEwLjg1MDQ3Njk3MDUwNDM5fQ==
```

Decode from Base64 format

Simply enter your data then push the decode button.

```
ey02LjgxNDU1Mzc0Njk1OTE5MywgMTEwLjg1MDQ3Njk3MDUwNDM5FQ==
```

For encoded binaries (like images, documents, etc.) use the file upload form a little further down on this page.

UTF-8 Source character set.

Decode each line separately (useful for when you have multiple entries).

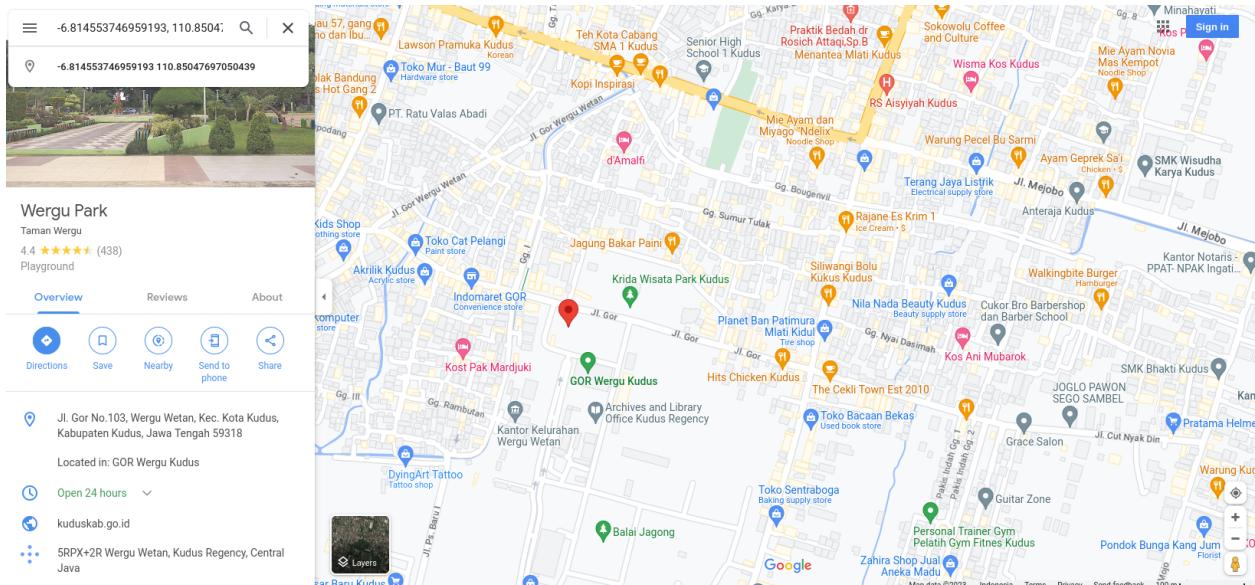
Live mode OFF Decodes in real-time as you type or paste (supports only the UTF-8 character set).

DECODE Decodes your data into the area below.

```
{-6.814553746959193, 110.85047697050439}
```

koordinat {-6.814553746959193, 110.85047697050439}.

Lalu saya mencari koordinat tersebut di maps dan menemukan lokasi ini screenshootan



Lalu saya mengecek ulasan pada tempat tersebut, dan menemukan ulasan dari user

@DAW?DAW?DWA?DWA?DWA? ??SNORK?? Dengan komentar :

bendera:

Tinggal tambahkan format flagnya menjadi :

Problem 4: Iwanttocry

Terdapat instruksi untuk login ke ssh dengan command:

```
ssh 167.71.207.218 -p 35022
```

Terdapat file creds.txt yang berisi credentials untuk login ke ssh. File tersebut berisi:

```
++++++[>+>++>++++++>++++++<<<-]>>>+++++++.-----  
---.+++++++.-----.+++++.<-----.-.----.>-----  
-----.<---.+++.>+++++++.+++.<---.+++.-.----.>  
+++++++.<.
```

Terlihat seperti kode Brainfuck, ketika dijalankan menggunakan Brainfuck interpreter, memunculkan hasil:

```
malbi:77U37dg261yyo1
```

Dengan menggunakan identitas ini, berhasil login ke remote machine. Hint menunjukkan bahwa binary crying adalah ransomware nya, dari nama tersebut mengingatkan kepada ransomware bernama WanaCry dimana “killswitch” nya adalah membuat request kepada domain aneh yang dimana sukses, malware nya akan gagal jalan.

Mengetahui ini, kami menemukan binary nya di /bin/crying

```
[malbi@2282c344fb50:~$ ls -la /bin/crying  
-rwxr-xr-x 1 root root 6976656 Apr 8 04:26 /bin/crying
```

Ketika dijalankan, terdapat delay dikit, lalu keluar hasil mengecewakan... Dijalankan pakai sudo juga engga bisa.

```
[malbi@2282c344fb50:~$ ./bin/crying  
Sob...  
[malbi@2282c344fb50:~$ sudo ./bin/crying  
sudo: unable to resolve host 2282c344fb50: Name or service not known  
Sob...  
[malbi@2282c344fb50:~$ ]
```

Tapi... Ketika di Ctrl+C di tengah eksekusi, muncul seperti ini:

```
[malbi@2282c344fb50:~$ sudo /bin/crying
sudo: unable to resolve host 2282c344fb50: Name or service not known
^CTraceback (most recent call last):
  File "crying.py", line 1, in <module>
  File "<frozen importlib._bootstrap>", line 991, in _find_and_load
  File "<frozen importlib._bootstrap>", line 975, in _find_and_load_unlocked
  File "<frozen importlib._bootstrap>", line 671, in _load_unlocked
  File "PyInstaller/loader/pyimod02_importers.py", line 352, in exec_module
  File "requests/__init__.py", line 118, in <module>
  File "<frozen importlib._bootstrap>", line 991, in _find_and_load
  File "<frozen importlib._bootstrap>", line 975, in _find_and_load_unlocked
  File "<frozen importlib._bootstrap>", line 671, in _load_unlocked
  File "PyInstaller/loader/pyimod02_importers.py", line 352, in exec_module
  File "requests/utils.py", line 25, in <module>
  File "<frozen importlib._bootstrap>", line 991, in _find_and_load
  File "<frozen importlib._bootstrap>", line 975, in _find_and_load_unlocked
  File "<frozen importlib._bootstrap>", line 671, in _load_unlocked
  File "PyInstaller/loader/pyimod02_importers.py", line 352, in exec_module
  File "requests/certs.py", line 15, in <module>
  File "<frozen importlib._bootstrap>", line 991, in _find_and_load
  File "<frozen importlib._bootstrap>", line 975, in _find_and_load_unlocked
  File "<frozen importlib._bootstrap>", line 671, in _load_unlocked
  File "PyInstaller/loader/pyimod02_importers.py", line 352, in exec_module
```

Python? PyInstaller? Oke, jadi executable ini merupakan hasil dari PyInstaller menjadikan script crying.py menjadi satu executable.

Setelah itu, saya transfer executable ini menggunakan scp ke mesin saya untuk di analisa.

Dengan menggunakan command objcopy --dump-section pydata=pydata.dump crying, kami membuat dump dari file-file yang terdapat di section pydata di executable nya.

Menggunakan tool <https://github.com/extremecoders-re/pyinstxtractor>, file dump berhasil di extract, dimana isinya sebagai berikut.

> base_library	Today 13.24	--	Folder
> certifi	Today 13.13	--	Folder
> cryptography	Today 13.13	--	Folder
> cryptography-3.4.8.egg-info	Today 13.13	--	Folder
> lib-dynload	Today 13.13	--	Folder
> PYZ-00.pyz_extracted	Today 13.13	--	Folder
base_library.zip	Today 13.13	1 MB	ZIP archive
crying.pyc	Today 13.13	641 bytes	Document
libbz2.so.1.0	Today 13.13	75 KB	Document
libcrypto.so.3	Today 13.13	4,5 MB	Document
libexpat.so.1	Today 13.13	195 KB	Document
liblzma.so.5	Today 13.13	170 KB	Document
libpython3.8.so.1.0	Today 13.13	5,4 MB	Document
libssl.so.3	Today 13.13	668 KB	Document
libz.so.1	Today 13.13	109 KB	Document
pyi_rth_inspect.pyc	Today 13.13	684 bytes	Document
pyiboot01_bootstrap.pyc	Today 13.13	859 bytes	Document
pyimod01_archive.pyc	Today 13.13	9 KB	Document
pyimod02_importers.pyc	Today 13.13	15 KB	Document
pyimod03_ctypes.pyc	Today 13.13	4 KB	Document
PYZ-00.pyz	Today 13.13	1,3 MB	Document
struct.pyc	Today 13.13	311 bytes	Document

Tentu saja entry pointnya adalah `crying.pyc`, namun ini dalam bentuk bytecode. Dengan menggunakan module `uncompyle6`, kami berhasil decompile bytecode tersebut. Isinya:

```
1 # uncompyle6 version 3.5.0
2 # Python bytecode 3.8 (3413)
3 # Decompiled from: Python 2.7.5 (default, Nov 16 2020, 22:23:17)
4 # [GCC 4.8.5 20150623 (Red Hat 4.8.5-44)]
5 # Embedded file name: crying.py
6 import requests, os
7
8 def check_domain--- This code section failed: ---
9
10      5      0 SETUP_FINALLY          28  'to 28'
11
12      6      2 LOAD_GLOBAL           requests
13          4 LOAD_ATTR              head
14          6 LOAD_FAST               'domain'
15          8 LOAD_CONST              5
16          10 LOAD_CONST             ('timeout',)
17          12 CALL_FUNCTION_kw_2     2  ''
18          14 STORE_FAST             'response'
19
20      7      16 LOAD_FAST              'response'
21          18 LOAD_ATTR              status_code
22          20 LOAD_CONST              200
23          22 COMPARE_OP             ==
24          24 POP_BLOCK
25          26 RETURN_VALUE
26      28_0 COME_FROM_FINALLY     0  '0'
27
28      8      28 DUP_TOP
29          30 LOAD_GLOBAL           requests
30          32 LOAD_ATTR              exceptions
31          34 LOAD_ATTR              RequestException
32          36 COMPARE_OP             exception-match
33          38 POP_JUMP_IF_FALSE     52  'to 52'
34          40 POP_TOP
35          42 POP_TOP
36          44 POP_TOP
37
38      9      46 POP_EXCEPT
39          48 LOAD_CONST              False
40          50 RETURN_VALUE
41      52_0 COME_FROM             38  '38'
42      52 END_FINALLY
43
44 Parse error at or near `POP_BLOCK` instruction at offset 24
45
46
47 domain = 'http://yiewvciwyefiowuteyrt63257486gdfewytifuywewhfg.co.ph'
48 if check_domain(domain):
49     os.system('echo "I\'m no longer crying. Here\'s your flag:"')
50     os.system('cat /opt/flag.txt')
51     os.system('cp /etc/hosts.bak /etc/hosts')
52 else:
53     print('Sob...')
```

Ternyata mirip seperti ransomware WanaCry. Script mengecek apakah domain yang dituju mereturn status code 200, jika iya, flag akan di print. Script juga memberikan hint untuk memainkan /etc/hosts.

Cara yang ampuh menurut kami:

- Buat 2 koneksi SSH
 - 1 koneksi menjalankan server Python sederhana
 - 1 koneksi mengeksekusi script untuk memodifikasi /etc/hosts dan menjalankan /bin/crying

Perlu beberapa percobaan agar berjalan dengan baik, dan pada akhirnya flag nya ketemu juga.

WRECKIT40{R341_c453_0f_w4NN4cRY}

Problem 5: Survey

<https://form.korpstar-poltekssn.org/index.php/199124> berisi form seputaran WreckIT 4.0.

Setelah diselesaikan, muncul flag berikut. Thanks for yet another free flag!

WRECKIT40{M4KAS1H_UDAH_I51_SURV3Y_SEM0G4_F1N4L}