学到了 fini段和init段的作用

也了解到了心理博弈

https://bbs.pediy.com/thread-254172.htm

ida拖入

```
🦜 IDA - attachment.elf C:\Users\wsxk\Desktop\ctf\BUUCTF\2019红帽杯_easyRE\attachment.el
File Edit Jump Search View Debugger Lumina Options Windows Help
 Library function Regular function Instruction Data Unexplored External symbol Lumina function
                                              □ ♂ x 🗓 IDA V··· ② 🗓 Pseudoc··· ② 🗓 Pseudoc··· ② 🗓 Pseudoc··· ② 🗓 Pseudoc··· ② 💆 Strings v··· ② 🔘 Hex V··· ② 🖪 Struc·· ② 🗒 Emms ③ 📆 Imp··· ② 🗗 Exp··· ③
📝 Functions window
                                                                16 char v13[4]; // [rsp+6Dh] [rbp-B3h] BYRE
Function name
                                                                      char v13[4]; // [rsp+60h] [rbp-85h] BYREF
char v14[19]; // [rsp+71h] [rbp-AFh] BYREF
char input[32]; // [rsp+96h] [rbp-96h] BYREF
int v16; // [rsp+86h] [rbp-76h]
char v17; // [rsp+84h] [rbp-60h] BYREF
unsigned __int64 v19; // [rsp+108h] [rbp-18h]
                                                                                    _readfsqword(0x28u);
                                                                     v19 = __readfsqword(0x28u);
qmemcpy(v12, "Iodl>Qnb(ocy", 12);
v12[12] = 127;
qmemcpy(v13, "y.i", 3);
v13[3] = '\x7F';
qmemcpy(v14, "d'3w}wek9{iy=~yL@EC", sizeof(v14));
memset(input, 0, sizeof(input));
v16 = 0;
v17 = 0;
scanf(0, input, 0x25ull);
                                                                       scanf(0, input, 0x25uLL);
Line 1 of 1
                                                                       if ( ((_int64 (_fastcall *)(char *))length)(input) == 36 )
A Graph overview
                                                                           for ( i = 0; i < (unsigned __int64)((__int64 (__fastcall *)(char *))length)(input); ++i )</pre>
                                                                                                          _int8)(input[i] ^ i) != v12[i] )// 输入与输入的索引异或必须等于上面的字符
                                                                    000009C6 sub_4009C6:16 (4009C6)
0utput window
                                                                                                                                                                                                                                                                                               □ ₽
| Output vindow | 443860: using guessed type __int64 __fastcall sub_443860(_QNORD); |
6C62820: using guessed type void *off_6C6282e; |
6CF29C: using guessed type int dword_6CF29C; |
138, 89, 65, 49, 59, 116, 67, 34, 113, 67, 69, 9, 4, 80, 70, 51, 46, 0, 69, 9, 20, 80, 83, 34] |
b'aYA1;tC'aCEtt\x04P51.\x06Ett\x14P5" |
b'aYA1;tC'aCEtt\x04P51.\x06Ett\x14P5" |
b'flag(Active_DefenSe_Test)' |
caching 'Functions window'... ok |
Caching 'Functions window'... ok |
AU: idle Down Disk: 91GB
```

顺着这个思路一直下去

得到两个结果

```
b'https://bbs.pediy.com/thread-254172.htm'
b'Info:The first four chars are `flag`'
```

很明显,自己被搞了

这题应该是考了fini_array的知识点

(看了wp才知道)

fini_array

fini段的作用:

此节区包含了可执行的指令,是进程终止代码的一部分。程序正常退出时,系统将安排执行这里的代码。

```
nit_array:000000000006CBEC8 _init_array
                                        segment qword public 'DATA' use64
nit_array:00000000006CBEC8
                                        assume cs:_init_array
nit_array:00000000006CBEC8
                                        ;org 6CBEC8h
nit array:00000000006CBEC8 funcs 4020B8
                                        dq offset sub 400970
                                                             ; DATA XREF: sub 402080+21o
nit array:00000000006CBEC8
                                                              ; sub 402080+Aîo ...
nit array:00000000006CBEC8
                                        dq offset sub 4009AE
nit array:00000000006CBED8 funcs 402130
                                        dq offset sub 4005F0
                                                            ; DATA XREF: sub 402110:loc 4021301r
nit_array:00000000006CBED8 _init_array
                                        ends
nit_array:00000000006CBED8
ini_array:00000000006CBEE0
ini_array:00000000006CBEE0 ; Segment type: Pure data
ini_array:00000000006CBEE0 ; Segment permissions: Read/Write
                                        segment qword public 'DATA' use64
ini_array:000000000006CBEE0 _fini_array
ini_array:00000000006CBEE0
                                        assume cs:_fini_array
ini_array:00000000006CBEE0
                                       ;org 6CBEE0h
ini_array:00000000006CBEE0 off_6CBEE0
                                       dq offset sub_400940
                                                              ; DATA XREF: sub_402080:loc_4020C81o
ini_array:00000000006CBEE0
                                                              ; sub_402110+6<sup>o</sup>
ini array:00000000006CBEE8
                                       dq offset sub 400D35
ini array:0000000006CBEF0
                                        dq offset sub 4005C0
ini array:00000000006CBEF0 fini array
                                        ends
```

可以看到在结束程序输入后,运行了3个函数

其中400d35函数有判断字符输入(其他两个其实都是正常的检查函数)

```
unsigned __int64 v5; // [rsp+28h] [rbp-8h]
10
     v5 = __readfsqword(0x28u);
11
     v1 = sub 43FD20(0LL) - qword 6CEE38;
12
    for (i = 0; i \le 1233; ++i)
 13
       sub_40F790(v1);
14
       sub_40FE60();
15
16
       sub 40FE60();
       v1 = sub_{40}FE60() ^ 0x98765432;
17
 18
19
     if ( ((unsigned __int8)v1 ^ byte_6CC0A0[0]) == 'f' && (HIBYTE(v4) ^ (unsigned __int8)byte_6CC0A3) == 'g' )
20
 21
       for (j = 0; j \le 24; ++j)
22
23
         sub_410E90((unsigned __int8)(byte_6CC0A0[j] ^ *((_BYTE *)&v4 + j % 4)));
 24
25
       result = __readfsqword(0x28u) ^ v5;
26
     if ( result )
27
       sub 444020():
28
     return result;
29 }
    000000025 aub 400025.6 (400025)
```

可以看到v1和首四个字节异或应该得到flag 那么v1的每个字节为 flag和bcc0a0的前四个字节的异或 写脚本,就能得到flag