CSAPP 深入理解计算机系统 Buflab实验，缓冲区溢出攻击实验（1）

原创

shiyuqing1207 于 2015-06-01 19:18:53 发布 16464 收藏 60
分类专栏： web安全 汇编 CSAPP 深入理解计算机系统 linux 文章标签： csapp 安全 漏洞 栈

由于实验太长所以还是采用分开

其实感觉之前做过那个bomb实验以后，这个实验相对来说还是很容易的，主要是要搞懂缓冲区溢出原理，以及堆栈的过程，函数调用的实现过程，做完这个实验如果认认真真的做完，对于堆栈的过程，还有缓存溢出攻击的原理就会掌握的比较清楚的。个人感觉实验设计的还是非常的好的，非常值得做一做。做这个实验首先是要建立在bomb实验和对函数调用栈过程的基础上，如果你觉得你理解起来有困难，建议先补充这方面的知识，我的博客中也有专门的相关文章。

和之前做实验一样，首先把buflab-handout.tar.gz文件夹拖到我们的虚拟机的csapp文件夹下面。这里由于我之前做实验的时候已经解压过了，我们先删掉从头开始。

```
syq@syq-virtual-machine:~$ ls
core              Ubuntu One                          公共的  图片   音乐
csapp             VMwareTools-9.6.0-1294478.tar.gz    模板    文档   桌面
examples.desktop  vmware-tools-distrib                视频    下载
syq@syq-virtual-machine:~$ cd
syq@syq-virtual-machine:~$ cd csapp
syq@syq-virtual-machine:~/csapp$ ls
bits.c-solution        datalab-handout      mybuflab_solved.tar
buflab-handout         datalab-handout.tar  实验3
buflab-handout.tar.gz  mybuflab_solved      实验3.zip
syq@syq-virtual-machine:~/csapp$ rm -rf b
bits.c-solution        buflab-handout/        buflab-handout.tar.gz
syq@syq-virtual-machine:~/csapp$ rm -rf buflab-handout
syq@syq-virtual-machine:~/csapp$ ls
bits.c-solution        datalab-handout      mybuflab_solved      实验3
buflab-handout.tar.gz  datalab-handout.tar  mybuflab_solved.tar  实验3.zip
```


首先将文件进行解压，这里问价是.tar.gz格式的，先解压

```
syq@syq-virtual-machine:~/csapp$ tar zxvf buflab-handout.tar.gz
buflab-handout/bufbomb
buflab-handout/hex2raw
buflab-handout/makecookie
buflab-handout/
syq@syq-virtual-machine:~/csapp$ ls
bits.c-solution        datalab-handout      mybuflab_solved.tar
buflab-handout         datalab-handout.tar  实验3
buflab-handout.tar.gz  mybuflab_solved      实验3.zip
```

解压以后在查看一下里面有些什么：通过ll看到里面的文件都是可执行的

```
syq@syq-virtual-machine:~/csapp$ cd buflab-handout
syq@syq-virtual-machine:~/csapp/buflab-handout$ ls
bufbomb  hex2raw  makecookie
syq@syq-virtual-machine:~/csapp/buflab-handout$ ll
总用量 1076
drwxrwxr-x 2 syq syq    4096  5月 15 00:29 ./
drwxrwxr-x 6 syq syq    4096  5月 31 19:26 ../
-rwxrwxr-x 1 syq syq 1074674  5月 15 00:29 bufbomb*
-rwxrwxr-x 1 syq syq    7798  5月 15 00:29 hex2raw*
-rwxrwxr-x 1 syq syq    7384  5月 15 00:29 makecookie*
```

这里可以看到解压以后里面一共就三个文件，结合实验指导书，

这三个文件的作用是：

bufbomb: The buffer bomb program you will attack.

makecookie: Generates a “cookie” based on your userid.

hex2raw: A utility to help convert between string formats.

可以看到bufbomb就是一个有缓冲区溢出漏洞的程序，makecookie就是一个可以根据用户不同的userid生成的唯一的cookie，这个主要是说学生的userid不同，cookie不同，所要解决的方法就不同，所以接下来实验我都是用的我自己的userid：syq生成的cookie做的实验，你用你自己的userid时，答案跟我的是不同的。然后hex2raw的作用是使你编写的缓冲区利用代码的转换为一个字符串的格式，只有经过转换以后才可以输入到getbuf中，因为本身test函数就是需要你输入字符串格式的，如果你直接输入你的漏洞利用代码是不可以的。也就是说你每次编写的漏洞利用代码都要经过这个文件进行一下转换再输入到bufbomb中。

接下来首先先将bufbomb这个程序通过objdump进行反汇编，生成其汇编代码进行查看。

```
syq@syq-virtual-machine:~/csapp/buflab-handout$ objdump -d bufbomb >bufbomb.s
syq@syq-virtual-machine:~/csapp/buflab-handout$ ls
bufbomb  bufbomb.s  hex2raw  makecookie
syq@syq-virtual-machine:~/csapp/buflab-handout$
```

和前一个bomb实验一样，我们可以把bufbomb.s文件拖到windows中用notepad++以汇编语言的格式进行查看。

接下来根据自己的userid来生成一个cookie，因为每次运行bufbomb时要用到。

```
syq@syq-virtual-machine:~/csapp/buflab-handout$ ./makecookie syq
0x6087639b
syq@syq-virtual-machine:~/csapp/buflab-handout$
```

整个实验是针对getbuf具有的漏洞展开的，getbuf函数类似于gets函数，它是读入一段字符串，字符串以\n或者eof表示结束，并把存储起来，但是getbuf提供的缓冲区只有32个字符大小，但是getbuf本身又对输入的字符是否超过缓冲区大小进行安全检查，从而带来了缓冲区溢出漏洞。以下就是getbuf函数的代码：

```
1 /* Buffer size for getbuf */
2 #define NORMAL_BUFFER_SIZE 32
3
4 int getbuf()
5 {
6     char buf[NORMAL_BUFFER_SIZE];
7     Gets(buf);
8     return 1;
9 }
```


接下来是输入的字符串在31个字符之内和超出了31字符时（32个字符但是还有结束占一个字符）

```
unix> ./bufbomb -u bovik
Type string: I love 15-213.
Dud: getbuf returned 0x1
```

Typically an error occurs if we type a longer string:


```
unix> ./bufbomb -u bovik
Type string: It is easier to love this class when you are a TA.
Ouch!: You caused a segmentation fault!
```

可以看到超出的时候就会造成段错误，多出buf的部分就会导致程序的状态被重写，我们正好就可以利用这个漏洞，来编写我们的漏洞利用代码exploit string。

看一下bufbomb的不同的参数的含义。主要用到的就是-u，确保不同的userid用不同的cookie，然后就是-n是为了level4，栈基址随机化模式的时候使用的。-s是上传到服务器进行打分，这个部分是没有的。

-u *userid*: Operate the bomb for the indicated userid. You should always provide this argument for several reasons:

- It is required to submit your successful attacks to the grading server.
- BUFBOMB determines the cookie you will be using based on your userid, as does the program MAKECOOKIE.
- We have built features into BUFBOMB so that some of the key stack addresses you will need to use depend on your userid's cookie.

-h: Print list of possible command line arguments.

-n: Operate in "Nitro" mode, as is used in Level 4 below.

-s: Submit your solution exploit string to the grading server.

```
syq@syq-virtual-machine:~/csapp/buflab-handout$ ./bufbomb -h
Usage: ./bufbomb -u <userid> [-nsh]
  -u <userid> User ID
  -n          Nitro mode
  -s          Submit your solution to the grading server
  -h          Print help information
syq@syq-virtual-machine:~/csapp/buflab-handout$
```


然后我们可以把我们的exploit写到一个txt中，然后给出了几种将我们的txt执行的格式：

1. You can set up a series of pipes to pass the string through HEX2RAW.

```
unix> cat exploit.txt | ./hex2raw | ./bufbomb -u bovik
```

2. You can store the raw string in a file and use I/O redirection to supply it to BUFBOMB:

```
unix> ./hex2raw < exploit.txt > exploit-raw.txt
unix> ./bufbomb -u bovik < exploit-raw.txt
```


This approach can also be used when running BUFBOMB from within GDB:

```
unix> gdb bufbomb
(gdb) run -u bovik < exploit-raw.txt
```

Cat主要是多次提交我们的txt，主要是在level4中会用到，因为level4要求你连续输入5次你的string。

然后是一些比较重要的提示事项：

Important points:

- Your exploit string must not contain byte value 0x0A at any intermediate position, since this is the ASCII code for newline ('\n'). When Gets encounters this byte, it will assume you intended to terminate the string.

- HEX2RAW expects two-digit hex values separated by a whitespace. So if you want to create a byte with a hex value of 0, you need to specify 00. To create the word 0xDEADBEEF you should pass EF BE AD DE HEX2RAW.

一个是getbuf遇到换行就会认为输入停止，而换行\n的对应的ASCII值是0x0a，所以不可以输入的字符串中含有0x0a

另一个就是说，格式转换要注意每次输入的都是两个数字，比如你要输入0，就要输入00。

还有要注意小端格式的输入。

其实感觉实验提供的说明文档说的好清楚，再加上之前做过那个bomb实验，这个实验真的是比较简单了。但是挺有意思，你是不是跃跃欲试了，接下来就是开始吧。