**实 验 报 告**



**课程名称 《网络攻击与防御技术》**

**学 院 计算机科学技术学院**

**专 业 信息安全**

**姓 名 黄 力**

**学 号 15307130275**

**开 课 时 间 2018 至 2019 学年第 1 学期**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | 实验项目  名 称 | Linux 本地栈溢出 | 成绩 |  |   **一、实验目的**  （1）了解C语言程序中函数调用时程序栈的变化情况和栈溢出的基本原理  （2）通过实验掌握如何使用栈溢出获取带有root权限的shell  （3）熟悉一些基本的linux命令，了解linux下的编程和调试基本工具。  **二、实验内容**  （1）在非root用户下利用tryof.c编译得到的tryof程序中的栈溢出漏洞和其具备suid标志位的属性获取具有root权限的shell  （2）分析实验成功或失败的原因  **三、实验环境**  （1）PC机操作系统：macOS Mojave 10.14  （2）虚拟机操作系统（Parallels Desktop 13.1.1）：32位redhat3  **四、实验原理**  通过阅读tryof程序的源代码tryof.c可以发现该程序的主要功能是将用户在命令行参数中给出的字符串使用strcpy函数拷贝到大小为300字节的buf中。在本实验中，程序开启了栈上内容可执行并关闭了栈地址随机化，结合tryof具备suid标志位的属性。为获得具有root权限的shell，可以利用该栈溢出漏洞构造shellcode填充到程序栈中，并修改strcpy函数执行完后的返回地址至shellcode的起始地址使得程序从strcpy函数返回后继续执行构造的shellcode从而达到攻击。  **五、实验步骤及结果**  （1）从云复旦http://cloud.fudan.edu.cn/shareFolder/466220002/UHWpvrr中下载redhat3.rar，解压并利用其中的虚拟硬盘在Parallels Desktop安装redhat操作系统获得实验环境，使用hacker作为登入帐号（无密）登入，登入目录为/home/hacker，在此目录中已有编译好的具备suid标志位的tryof程序。可使用ls -l tryof命令查看，结果如下图：  （2）阅读tryof.c源码文件发现buf的大小为300字节，但由于程序中还定义了ret等其余变量，所以buf的溢出大小比300应该要大一点，但不会大太多。这里可以利用两种方法来得到产生segment fault的溢出大小值：第一种方法是使用objdump -d tryof命令查看tryof的汇编代码找到strcpy函数调用前给%eax寄存器赋值的大小（%eax的值代表buf的地址），如下左图红圈所示，0x14c也就是十进制的332即是溢出大小；第二种方法是使用二分法在尝试不同长度的字符串输入是否产生segment fault来判断，如下右图当使用331个a作为输入时不产生segment fault，而使用332个a作为输入时则会产生segment fault。    （3）使用C语言编写exploit程序tryof\_exploit.c对tryof程序进行栈溢出攻击。tryof\_exploit.c程序的源代码如下左图所示：  此程序的逻辑是：  1、构造shellcode数组，此处我使用了老师给的stackexploilt2.c给出的shellcode模版，然后在main函数中大小为888的buf用于构造输入的字符串，此处需要注意的是buf的大小必须是4的倍数（内存对齐原则，下同）。  2、使用p获得buf的起始地址，此值加上456（这里得456只要大于第4步中选取的344且是4的倍数即可）即为修改后返回执行的首条指令的地址，将这个值用ret保存起来，在第4步中需要用到。  3、调用memset函数将buf的值全部初始化为0x90，即nop指令（程序执行此指令时什么也不做，并继续接下来的指令）。  4、使用一个for循环将buf前344个字节赋值为返回地址（ret），此步需要注意的是覆盖的范围应该大于332（（2）中分析得到），代码中选择了344，也可以选择338、342等，还需要注意的就是这个值也需要是4的倍数。  5、调用memcpy函数将shellcode拷贝到buf中从444+344=788往后的strlen（shellcode）个字节中（这里的444是可供选择的调节shellcode与返回地址的值，必须也是4的倍数），此时shellcode到返回地址的距离就是444+（344-332）=456，到buf[0]的距离为788。这样buf中的内容就构造完毕。buf中的内容和布局如上右图所示。  6、使用execl函数将buf作为命令行输入运行tryof程序。tryof在调用strcpy函数后执行返回地址指向的指令：位于p+456处，该指令为nop，之后继续执行往后的nop指令直到shellcode所在位置：p+788，执行完shellcode后就得到了具有root 权限的shell。  （4）编译运行tryof\_exploit.c获取root权限的shell并验证：使用gcc -o tryof\_exploit tryof\_exploit.c命令编译得到攻击程序tryof\_exploit，然后使用./ tryof\_exploit命令运行攻击程序，结果如下图，获得shell后可以分别使用whoami和id命令验证结果，从下图中红圈部分可得知本次实验已经成功。    **六、实验总结**  通过本次实验，我理解了栈溢出的原理，并成功编写exploit程序利用该漏洞获得了具有root权限的shell。本次实验的难点主要有两点：一是shellcode的编写，解决方法是借用了老师提供的shellcode。二是332这个溢出值的寻找，解决方法是参考使用了参考资料1中的方法。  **七、主要参考资料**  1、<https://blog.csdn.net/tyskfs2/article/details/42318531>  2、<https://github.com/tinyclub/open-c-book/blob/master/zh/chapters/02-chapter5.markdown>  3、<https://blog.csdn.net/azloong/article/details/6158401>  4、<https://blog.csdn.net/raintungli/article/details/43865041> |