Linux 审计功能的分析和扩展

须文波 ， 王 斌 ， 冯 斌

（江南大学信息工程学院，无锡 214036）

摘 要：安全审计是实现安全的操作系统的一个重要组成部分。本文在分析当前审计系统所存在问题的基础上，给出了一个利用可加载内核模块的方法实现内核级审计的例子。

关键词：审计；系统调用；可加载内核模块；设备驱动

引 言

随着信息技术的发展与普及，特别是随着互连网络的迅速扩张，使得信息的交流和共享成为现代科技和发展的重要前提，但随之而来的网络入侵事件的数量也成倍增长。安全审计是评价计算机系统安全性的重要标准之一，获得准确和完整的审计信息对于安全检测和入侵检测有重大的作用。

1 Linux 系统安全审计系统分析

Linux 系统现行的安全审计机制是在应用程序级实现的，它是通过独立于操作系统的审计程序 syslogd 来记录用户登陆和相关操作的信息的，对用户正常或异常的操作所产生的警告和提示等信息以统一的格式记录下来，其功能结构如图 1 所示。其 shell 命令被记录在用户工作目录下的 .bash_history 文件中，但没有记录 shell 命令的时间。该种记录方式没有全局所有用户的审计信息，因而不便于浏览和管理。由于系统实现安全审计功能的是应用程序，因而入侵者取得一定的权限后有可能绕过 syslogd，而使其入侵操作不被记录，甚至抹掉所有的审计信息和入侵记录。

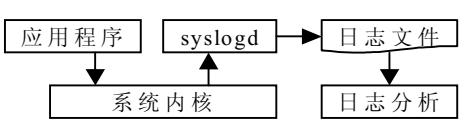

图 1 现有 Linux 系统审计逻辑

为了保证审计信息的有效准确和实时性，可利用可加载内核模块（Loadable Kernel Module）方式进行系统调用，劫持在内核检测用户执行的命令以实现审计信息的获取。

2 Linux 内核级安全审计系统设计

Linux 系统安全审计机制应具备以下特征：

（1）审计的进行应独立于应用程序，能够获取足够的信息，获取审计信息的过程不会被绕过且不会被非法用户破坏。

（2）审计数据的安全保护和访问控制。审计数据能够得到很好的保护，防止非法用户访问审计数据。

因此只有在系统内核级实现。利用 LKM 技术进行系统调用的劫持以实现审计数据的获取，这样不论是任何用户都无法绕过审计信息的获取。当用户请求访问审计日志时审计模块对用户身份进行验证以确定用户是否有权访问日志，保证了审计数据的安全，防止审计数据被非法访问。

3 可加载内核模块

（1）简介

Linux 的可加载内核模块（Loadable Kernel Modules）是操作系统内核为了扩展其功能提供的一种机制。LKM 的主要优点是动态加载，无须重新编译整个内核。基于这一特性，LKM 常被用于特殊设备的驱动程序。当需要该模块的功能时超级用

户可通过 insmod 命令加载，当不在需要时可通过 rmmod 命令将它从内核中卸载。这样保证了内核的紧凑性，又保证了 Linux 本身固有的单一体系结构的特点。

(2)基本结构

一旦 LKM 被载入核心后，它就成为核心代码的一部分，它与其他核心代码的地位相同。LKM 可以访问符号表指定的核心资源，可以修改内核变量，重载内核函数，也就为我们提供了修改系统基本行为的方法。

每个 LKM 至少要包含两个函数：

```
int init_module(void) /* 在 LKM 被载入内核时调用 */
void cleanup_module(void)
/* 在 LKM 从内核中卸载时调用 */
```

在 init_module( )里，我们将系统调用替换为我们自己的系统调用，修改某些内核参数。从而，在整个 LKM 生命周期我们在内核级控制，监视系统的行为。在 cleanup_module( )里恢复系统的设置。

(3)系统调用的劫持

系统调用是应用程序和操作系统内核之间的功能接口，其主要目的是使用户可以使用操作系统提供的有关设备管理、输入/输出系统、文件系统和进程控制、通信以及存储管理等方面的功能，而不必了解系统程序的内部结构和有关硬件细节，从而减轻用户负担和保护系统以及提高资源利用率。

在 Linux 系统中，系统调用是作为一种异常类型实现的，它将执行相应的机器代码指令来产生异常信号。产生中断或异常的重要效果是系统自动将用户态切换到核心态来对它进行处理。系统调用号是内核结构 sys_call_talbe[ ]的下标，若要对系统调用进行劫持需要修改 sys_call_table[ ]的相应入口地址。

4 实 现

LKM 与用户空间的进程间通信可使用 proc 文件系统和使用设备文件，此处笔者使用了设备文件方式。实现是使用 mknod 命令创建一个字符设备/dev/shelllog，使用 LKM 实现此设备的驱动程序，利用 LKM 劫持系统的 execve( )系统调用，取得用户的 shell 命令信息生成用户 shell 记录的链表，shell 命令信息可包括 shell 命令、用户 ID、有效 ID、组 ID、进程 ID、系统当前时间等，如图 2。

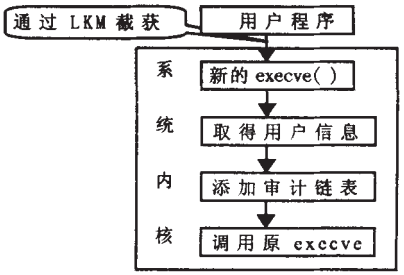

图 2 通过系统调用劫持获取审计信息

当有用户进程要读取/dev/shelllog 设备时，首先判断该用户进程是否为超级用户，如果不是则拒绝读操作，否则将审计链表的信息从内核空间通过 copy_to_user( )内核系统，调用拷贝到用户空间供用户进程使用。

(1)初始化

在 init_module( )函数中首先调用 devfs_register( )注册一个字符设备，在注册设备时须定义一个设备驱动程序接口，即 file_operations {}，file_operations {}结构中需要指定打开、读、释放操作的处理函数。当有进程对该设备进行操作时就会调用相应的处理函数，然后取原来的系统调用 execve( )的入口地址将其保存，再将我们自己的新 execve( )函数地址写入 sys_call_table[ ]中。

```
int init_module( )
{
 mydev_handle=devfs_register(NULL,"shelllog",DEVFS_
FL_DEFAULT,MAJOR_NO,0,S_IFCHR,&shelldrv_ops,NULL);
 orig_execve=sys_call_table[SYS_execve];
 sys_call_table[SYS_execve]=(void *)new_execve;
 return 0;
}
void cleanup_module( )
{
sys_call_table[SYS_execve]=orig_execve;
devfs_unregister(mydev_handle);
}
```

(2)设备驱动程序

该设备驱动程序只须定义打开、读和关闭操作。当执行读操作时首先判断该用户是否为超级用户，如果不是则拒绝读操作，否则将审计链表的信息从内核空间通过 copy_to_user( )内核系统调用拷贝到用户空间。

```
size_t mydev_read  (struct file *filp,char *buf,size_t count,
```

```
loff_t *f_ops)
    {
     char ctmp[MAXNAME+128];
     if(! suser( )) return -1;
     memset(ctmp,'\0',sizeof(ctmp));
     if(count>MAXNAME+127) count=MAXNAME+127;
     if(lcur! =NULL) /* lcur 指向日志链表首 */
    {
      sprintf(ctmp,"%s:%d:%d:%d:%d:%d\n",lcur->shell,lcur->
uid,lcur->euid,lcur->gid,lcur->pid,lcur->time);
      copy_to_user(buf,ctmp,count);
     /* 将内核空间数据拷贝到用户空间 */
      lcur=lcur->next;
     }
      else
        count=0;
     *f_pos+=count;
     return count;
     }
```

结　语

通过 LKM 可以实现在内核级的审计功能，任何用户都无法绕过，可以实时获得用户经 exec 调用的 shell 命令，且可以获得用户身份、执行时间等充分的审计信息。该方式取得的审计信息不会影响用户的操作，且保证了只有超级用户能够使用审计信息，具有较好的隐蔽性和安全性。

参考文献

[1]贾春福，徐伟，郑辉. Linux 系统内核级安全审计方法研究. 计算机工程与应用，2002(6)

[2]Alessandro Rubin. Linux 设备驱动程序. 北京：中国电力出版社，2000

[3]李善平，刘文峰，李程远等. Linux 内核 2.4 版源代码分析大全. 北京：机械工业出版社，2002

[4] Pragmatic. Linux 可装载模块完全指南. http://www.the-hackerschoice.com/papers/LKM_HACKING.html.2001

（收稿日期：2003-05-07）

Analyzing and Extension the Function of Auditing in Linux with LKM

XU Wen-bo , WANG Bin , FENG Bin

(School of Information Technology, Southern Yangtze University, Wuxi 214036 China)

Abstract：Auditing is a critical part for realizing safety Operating System. The disadvantages of current Linux audit system are analyzed in this paper, and then an example of realizing the function of kernel-level audit system based on LKM is given.

Key words：Auditing；System Call；LKM；Device Driver