维普资讯 http://www.cqvip.com

基于 Linux 系统的分权设计与实现*)

林志强 金 雷 茅 兵 谢 立
(南京大学软件新技术国家重点实验室 南京210093)
(南京大学计算机科学与技术系 南京210093)

The Design and Implementation of Privilege Divided Linux Based System

LIN Zhi-Qiang JIN Lei MAO Bing XIE Li
(State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing Jiangsu 210093)
(Department of Computer Science and Technology, Nanjing University, Nanjing Jiangsu 210093)

Abstract The secure operating system is the foundation of secure enhanced information systems. As an open source software, Linux has robust functions and good compatibilities with Unix. This paper analyzes the hidden troubles and faults in secure aspect of it, and fully discusses the design and implementation of privilege divided Linux-based system. It mainly focuses on the modifications of access control, super users' least privilege division and so on.

Keywords Privilege divided, Secure operating system, Linux, Information security

1 引言

操作系统是信息系统中最重要的基础软件,是构建其他软件安全运行的基础。对于整个计算机系统来说,一旦操作系统的防御被攻破,整个计算机系统的安全就受到了根本的威胁。而对于涉及到国家安全、军事机密和重要金融数据的计算机系统来说,其操作系统的安全性将是不言而喻的。

Linux 是被广泛使用的自由软件,与 POSIX 标准的兼容性和开放源码的特点,以及在系统结构、核心算法等方面与主流 Unix 系统的相似性使得它在短短的几年内取得了巨大的成功。

近年来,南京大学计算机系在 Linux 系统基础上,相继进行了一系列有关 Linux 安全增强技术的研究。在目前所开发出来的 SoftOS V1.1(基于 Linux 内核)已经达到信息安全国家标准第三级(B1级[1])。但 SoftOS V1.1所采用的系统管理模型仍然是 Linux 原有的管理模型,系统中存在着权力"至高无上"的超级用户。如果该超级用户的账号一旦被窃取,则入侵者就可以完全获取系统的控制权,对系统进行任意的篡改、破坏或窃取系统中的机密信息。

本文的主要工作就是在 SoftOS V1.1(Linux 内核2.4.2版本)基础上,进行有关系统分权的研究工作,为 SoftOS 向 B2级[1]升级提供理论与实践依据。

2 系统分析

传统 Unix 类操作系统中的超级用户已成为系统安全的重要隐患,其关键在于访问控制的处理粗糙而不灵活,访问控制机制无法满足"最小特权"原则[2]的要求,超级用户拥有日常管理、维护、审计等所有权限。所以在这样的访问控制机制之下,一旦系统的超级用户被攻破,整个系统将完全暴露在攻击者面前。同时这样也会使安全审计成为一种摆设,入侵者完全可以篡改审计数据,使它的入侵行为隐于无形。

另外在计算机系统中几乎所有带有危害性的行为可分为两类,一类是通过篡取某些用户帐号来获取特权,典型的例子有缓冲区溢出和特洛依木马攻击[3];另外一类是由于合法用户操作不当而造成对其拥有的权力的滥用,虽然这实际上是一种无意危害行为。而究其根源,我们就会发现这两种危害性行为都具有一个共同特征:由于操作系统对特权分配的粒度过粗而造成的危害。

所以解决上述问题的办法就是特权最小化。在目前我们所知道的特权划分方案中主要有两种:一是基于角色的特权最小化,利用基于角色的访问控制(Role Based Access Control)策略[4],根据用户角色的不同,限定用户对资源的访问,当然前提是要有不同角色的超级用户,此类系统主要有德国汉堡大学近年来在 Linux 上实现的安全增强系统 RSBAC (Rule Set Based Access Control for Linux)等;二是对系统进行一定的调整,以一组权力分离、相互制约的超级用户来取代原来的一位超级用户,访问控制策略仍然是自主访问控制(Discretionary Access Control)[6]和强制访问控制(Mandatory Access Control)[6]以及这二者的结合,此类系统如安全操作系统 UNIXWARE2.1/ES 的实现,系统中每个用户和进程只具有完成其任务的最小特权,系统可以不再有超级用户,而是由若干个系统管理员和操作员来共同管理系统,他们都只有部分特权且相互约束。

我们的解决办法同方案二类似,也借鉴了方案一的思想:细化超级用户的权限粒度,并按照权限操纵客体的不同将它们划分为若干类,同时创建若干个子超级用户,我们称之为子 Root,将集中的 Root 权限分配给各个子 Root,并且完全剔除原有系统中的单一超级用户,然后依据划分出的不同管理员的身份角色进入不同的受限的管理程序。这样,我们就既保留了原有系统中的操纵管理功能,又将安全隐患降至最低点。这就是我们所要实现的基于特权分化模型 PDM(Privilege-Divided Model)的分权系统或者说最小特权模型的基本思想。

*)本课题得到国家"863"高技术(NO:2001AA144010)经费资助。林志强 硕士生,研究方向:访问控制,安全操作系统。金 雷 硕士生,研究方向:信息与网络安全。茅 兵 教授,研究方向:系统安全。谢 立 教授,博导,研究方向:信息安全。

维普资讯 http://www.cqvip.com

3 管理员权力划分

在多用户操作系统中，每一个用户主体都对应着一个可信的客体操作集合。所以我们进行超级用户的权力划分，首先要对分离出的各个管理员主体的职责和他们对应的客体操作集合进行划分。

在尚未分权的 SoftOS V1.1中，超级用户(Root)的权限主要有：系统的日常维护，包括系统的设备管理，文件系统的管理，模块(module)的管理，系统常用服务(www，ftp，telnet等)的管理，系统核心的编译、更改，引导扇区的管理等；系统的审计管理，主要是审计文件的读写，审计进程的启停，审计属性(audit_ctrl、auidt_view，SoftOS V1.1实现)的设置，审计文件的检查和备份；系统的用户管理，添加删除用户，设置用户及组属性，修改用户口令，设置系统强制访问控制(MAC，SoftOS V1.1实现)开关，添加用户的安全标记，获得、改变原有用户的安全标记，覆盖强制访问控制检查等。

按照权力分离、互不干涉、互相制约和共同管理的原则，我们将超级用户(root)分成三类，分别是系统维护员(sysadm)、审计管理员(sysaudit)和安全管理员(syssec)。他们所要操作的客体集合主要包括：/etc 目录主要配置文件(configure file)，/bin、/usr/bin、/usr/sbin 目录有关管理命令及后台进程(executable & daemon)，相关系统调用(system call)，设备及/proc 文件系统(device & /proc)等。我们按照为完成基本管理所必需的客体集合为条件，进行客体集合划分。要达到某种安全性必然会有些牺牲，所以在划分管理员职能时我们也并未覆盖原有系统的所有管理需求，并且划分出的管理员职能也有交集，示意如图1。经过我们对系统的分析总结后划分出的各管理员的操作客体集合结果见表1所示(由于系统维护员 sysadm 涉及的客体集合太多，限于篇幅没有完全列出)。

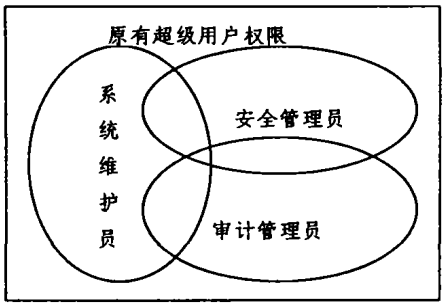

图1 权力划分示意图

表1

访问客体 / 管理员主体	管理命令	后台服务	配置文件审计
管理员 (sysaudit)	initlog logrotate loglevel audit_ctrl audit_view	klogd syslogd minilogd	syslog.conf initlog.conf logrotate.conf
安全 管理员(syssec)	useradd userdel groupadd guoupdel passwd usermod groupmod newuser mac vipw vigr pwck usernetctl	in.fingerd rlogin.d	login.defs passwd gshadow shadow default shells pam.d pwdb.conf security*.so
系统 维护员(sysadm)	mknod insmod lsmod rmmod depmod modinfo modprobe mkdosfs mke2fs mkfs mkfs.ext2 arp ifconfig ifdown ifup ifcfg ipmaddr iptunnel mount traceroute umount ….	rpc.lockd rpc.statd ypbind identd in.fingerd in.ftpd in.identd in.ntalkd routed snmped tcpd in.rexecd in.rlogind in.rshd in.talkd in.telnetd in.tftpd timed utmpd in.wuftpd inetd…	rc.conf rc*.conf lilo.conf grub.conf (most under etc directory files)…

4 设计及实现

在对系统的权力划分之后，我们主要从两个方面实现。首先是上层的管理程序，对不同的管理员我们启动不同的管理程序，由管理程序提供管理界面，在管理程序中控制对资源的配置访问(包括 Daemon 进程的启停等)。这实际上也是一种访问控制策略，即各个管理员用户只能在自己限定的客体集合上操作，不能越权访问不属于自己所要操作的客体。同时我们还要利用自主访问控制提供的保护措施，设定有关客体对象的属主及其访问权限，比如审计员要使用的/etc/syslog.conf 文件，它的属主(owner)就是审计员，其属性为 rw-------，这样通过自主访问控制就可以限定其他用户对其的操作。

然后在核心层我们还要根据 capability 机制和修改有关系统调用进行分权实现，因为管理程序一旦被攻破或者说如果我们划分的客体集合并不完备，我们就需要另外一层的保护，这就是修改 capability 和有关系统调用。具体的系统访问控制策略如图2所示。

4.1 上层管理程序

由于分权后每一个主体对应一个客体操作集合，因此当用户登录(login)时，依据各自的身份角色进入自己的管理程序(shell)。在用户 login 时根据其输入的 ID(审计员 sysaudit，安全员 syssec，系统员 sysadm，他们属于同组)跳转到不同的管理程序(shell)。

管理程序是我们根据具体的管理员的职责和相应的被控客体的集合开发的，我们假定它是可信的，同时在管理程序中我们有具体的安全机制来保障管理员的操作没有越权。因为我们把每个管理员所对应的客体划分成了互不相交的集合，

维普资讯 http://www.cqvip.com

在管理程序中，存在具体的映射机制，比如安全员就只能在安全员对应的客体集合中进行操作，而不能使用审计员有关的命令。对管理程序的简单理解就是它为用户提供了一种接口，这个接口的参数和可访问的集合我们是进行了限定的。

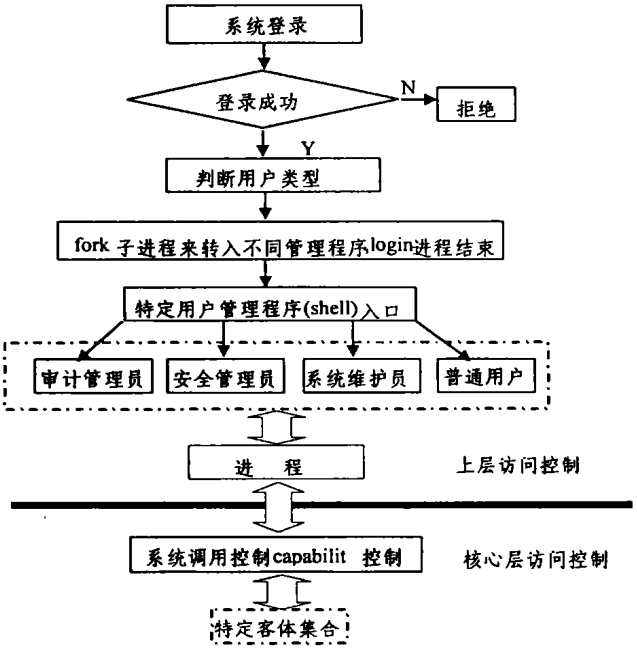

图2 系统访问控制策略图

系统中原来的管理命令许多是根据 Root 的 UID＝0来判断的，这是在自主访问控制策略下的要求。现在分权之后，各个管理员的 UID 不同(sysadm 0、syssec 498、sysaudit 499，他们属于同组用户，而普通用户的 UID 是从500开始的)，为此系统中还有大量的管理命令及后台服务进程需要修改，因为这些命令和服务本身是特权用户才能使用，现在将特权分离后，各个子特权用户应当能够不受影响地执行原有的特权用户的管理命令和服务。系统分权的大部分工作也都集中在这里。涉及的需要修改的管理命令和后台服务主要是在表1中所划分的客体集合。比如与安全员(syssec)有关的添加删除用户(useradd、userdel)命令，我们就要在它们的实现程序中进行修改，执行时判断用户的 UID 是否为498，并且还要限制他不能删掉 sysadm、sysaudit 用户(权力制约的要求)。

正如前文所述，我们还要利用自主访问控制设定有关文件的属性。这也即是根据在第三部分进行权力划分后所产生的客体集合，依据它们的属主设定每个文件的属性标记。然后利用核心提供的自主访问控制机制(C2级[1]系统都已具备)进行主客体的访问控制。

4.2 核心层

在核心层的实现策略是利用核心提供的 capability 机制；修改有关系统调用，建立系统访问控制规则表。

4.2.1 capability 机制 在 Linux 的内核中，有一种 capability 机制可以让我们来增强系统的安全性，因为可以通过设置用户的有关 capability 集合而绕过系统的自主访问控制。这一点使得假如我们在上层应用的分权不够完备的情况下，利用 capability 加以弥补。

在核心数据结构 task_struct 中定义了 kernel_cap_t 类型的32位无符号整数 cap_effective、cap_inheritable 和 cap_permitted 字段。每一个 capability 都由一个标志位来表示，目前一共定义了30种授权，具体的定义在/usr/src/linux/include/linux/capability.h 中。

“capability”的普遍含义是一个标识(token)，用于表示一个主体是否可以对一个客体实施某种动作。capability 通常体现在属主对某个客体的权限管理。在当今的各种操作系统中，都以某种形式表示了 capability 的存在。在 Linux 系统中，与 POSIX 1003.e[4]定义的 capability 的概念有些不同，它实际上是对超级用户的特权(Privilege)进行划分，形成若干个特权子集。基于 Linux 系统的 capability 分权的有代表性的是麻省理工学院开发的 Priv-Linux，可惜该操作系统并未最后完成。

一个进程有三类 capability 属性集：可继承 I Set(Inheritable)、允许 P Set(Permitted)和有效 E Set(Effective)。每一种 capability 都以位(bit)的形式记录下来。当一个进程试图请求特权操作时，系统内核都会检查该进程的相应 capability 位(调用 capable()函数)，而不是通常的检查 UID/EUID 是否为0(从 Linux 内核2.2版本以后就是如此)。例如一个进程准备设置系统时钟时，Linux 内核就会检查 CAP_SYS_TIME 位(目前在 cap_effective 的第25位)是否出现在 E Capability Set 中，而不再看他 UID 是否为0，只要他的 E Capability Set 中相应位为1即可。

在核心中，我们使用 capability 是针对不同的管理员进行不同的 capability 授权，调用 set_cap()函数，在系统启动的时候(init)进行赋值，然后在派生进程(fork、clone)的时候在 I Set 中继承。当涉及到访问控制的时候调用 capable()函数(定义于 kernel/sched.c)进行 capability 判断，仅当与访问控制策略相符时才允许相应的访问。

关于 capability 的初始值，我们根据目前的30位 capability 含义进行设定。比如对系统维护员(sysadm)，按照对系统权力的划分的需要，所设定的 I Set ＝ 0，P Set ＝ E Set ＝ CAP_CHOWN | CAP_DAC_OVERRIDE | CAP_SETPCAP | CAP_NET_ADMIN | CAP_SYS_MODULE | CAP_SYS_RAWIO | CAP_SYS_ADMIN | CAP_SYS_BOOT | CAP_SYS_TIME 权限位。按照权力不重叠的原则，这样系统安全员(syssec)的 capability 就没有 CAP_SYS_BOOT，相应的他就没有重启系统的权限，如果他调用 reboot 命令，核心就会显示 Opeation not permitted 信息。

4.2.2 系统调用的修改 通过对系统调用的分析发现，进程对配置文件的访问主要体现在 open()以及 chown()、chmod()调用族的调用上；管理命令的调用以及后台进程的启动主要体现在 exec()系统调用族的调用上；对于一般的系统调用，我们将按照系统原有的控制策略进行控制；而对设备的访问控制主要体现在系统调用 ioctl()的各种参数的控制，这种控制主要由相应的规则来描述；/proc 文件系统是由核心实现的一种“虚拟”文件系统，主要是用来使应用程序以访问文件的方式访问核心数据。这里修改系统调用，主要是修改该调用的入口，根据访问控制链表，判断该类用户在该调用下可以完成哪些工作。

然后我们根据不同的访问主客体建立访问控制规则表。由于系统涉及的控制主体较少(三类管理员和普通用户)，因此就以每类主体为对象，建立访问控制规则链表。当用户调用有关系统调用的时候，首先在访问控制表中依据被控客体进行遍历，如果找到，就按照分权的有关规范进行判断；否则就用系统缺省的强制访问控制(MAC，在 B1级中已经实现)或者自主访问控制(DAC)策略判断。具体的访问控制流程见图3(我们对该流程进行了推广，不止针对系统调用，凡是涉及到主体对客体的访问请求都可以利用该流程)，其中的主体 S 包

维普资讯 http://www.cqvip.com

括管理程序中的有关进程(管理命令、后台服务)和与分权有关的系统调用;客体O包括有关配置文件、设备文件、/proc目录下的文件。

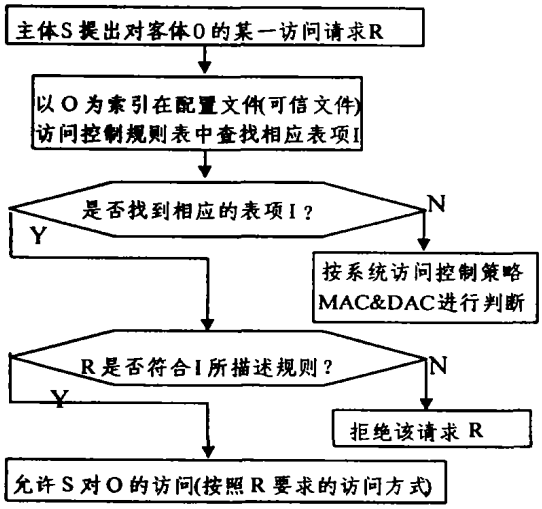

图3 核心级客体访问控制

5 相关比较

在传统的Unix类操作系统中,超级用户(root)可以访问任何文件,对其进行增、删、改;调用任何程序、管理命令;可以任意添加删除设备;任意添加删除修改用户信息;随意控制系统等。

而经过我们的分权工作后,所实现的系统相比,原有系统有如下特点:

1)特定的配置文件(/etc下的文件)或数据文件(如日志和审计数据文件)只能由特定的管理员来访问(读/写),并且该文件的属主即为该管理员。管理员对此的访问是通过特定的管理程序,即这些文件只能由相应的管理程序来访问,从而体现了管理员相应的特权。

2)特定的系统调用(或称为特权系统调用,这些系统调用在传统的UNIX系统中需要root权限)只能由特定的管理员来调用(其体现是通过相应的管理程序来实现的,即这些特权调用只能由相应的管理程序来执行),或者一些系统调用虽然它本身并非特权调用(如ioctl),但它的一些特殊参数,使得它需要管理权限,这时需要相应的管理员的权限。

3)特定的管理命令(/usr/sbin及/sbin目录下)只能由特定的管理员来执行,这些命令的文件属主也为该管理员。特定的设备特殊文件(虚/实两种设备)只能由特定的管理员来访问(通过特定的管理程序)。相应的后台进程(daemon)以相应的系统管理员的权限来运行。

以上的对比结果表明,我们所开发出的系统不仅可以防范缓冲区溢出攻击,而且即使入侵者篡取了系统的部分权力,它所产生的危害也只能限制在一个很小的范围内。

结束语 本文在现有Linux系统基础上,对超级用户的权力进行了划分并且予以实现,从而使权限粒度细化,将程序

(下转第164页)

优秀教材谱新篇
《人工智能及其应用》第三版问世

由蔡自兴教授和徐光祐教授编著的《人工智能及其应用》第三版本科生用书已由清华大学出版社出版并发行。该书第一版于1987年出版,是国内率先公开出版的人工智能教材。1992年在海外出版繁体字版。1996年了第二版。该书已先后印刷了10多次,共发行10万多册,居全国同类书籍之冠,已为国内数百所大学用作教材和教学参考书。由于该教材的特色和优势,它曾获1999年度国家教育部科技进步一等奖和2002年国际优秀论文(作品)奖,并与其它成果一起获2000年中国高校自然科学二等奖和2001年省部级优秀教学成果一等奖。

该书第三版本科生用书共11章。第一章叙述人工智能的概况,列举出人工智能的研究与应用领域。第二章和第三章研究传统人工智能的知识表示方法和搜索推理技术。第四章和第五章初步阐述了计算智能的基本知识,包含神经计算、模糊计算、进化计算和人工生命诸内容。第六章至第十章比较详细地讨论人工智能的主要应用,包括专家系统、机器学习、自动规划、艾真体(Agent)和自然语言理解等。第十一章评述近年来关于人工智能的争论,讨论人工智能对人类经济、社会和文化的影响,展望人工智能的发展。与第二版比,许多内容都是第一次出现的,如分布式人工智能与艾真体、计算智能与进化计算以及知识发现和数据挖掘等。其它章节也在第二版的基础上作了相应的修改、精简或补充。

该书可作为高等院校有关专业高年级学生的人工智能课程教材,也可供从事工智能研究与应用的科技工作者学习参考。研究生教材可使用该书的姊妹篇"研究生用书"。分别编著出版"本科生用书"和"研究生用书"是一种尝试,可望能够发挥更好的作用。

中国科学院院士、清华大学信息学院院长李衍达院士为该书第三版作序,对该书的特点和创新给予充分肯定。该书第三版还发表了中国工程院首任院长、原全国政协副主席宋健院士1988年2月任国务委员兼国家科委主任时给蔡自兴教授的亲笔信,对该书给予高度评价。两院士的序和信,进一步表明该书实属优秀著作和精品教材。

(魏世勇 孙国荣)