一种新的Capability实现机制

A Novel Realization Mechanism for Capability in Trusted Operating System

李 毅 周明天* 虞厥邦

(电子科技大学光电子技术系 计算机学院* 成都610054)

Abstract Generally the realization of the capability in trusted OS needs rewriting lots of the kernel code related to the syscalls. This paper introduces a new kind of realization mechanism for the trusted OS capability. Using the mechanism, not only the root user is removed, the root privileges are decomposed and can be issued to the common user, but also some user rights such as calling the syscall and taking use of shell commands related to root privilege, and the processes requesting the syscall, etc. can be controlled by system secure capability set. The new realization mechanism of the capability doesn't change lots of the kernel code and has been implemented successfully on Linux-based trusted OS.

Keywords Capability, Trusted OS, POSIX, DAC, MAC, LINUX Security

1 引言

信息安全,除CPU、编译器和网络安全外,最重要的组成部分就是操作系统的安全核心。POSIX.1e[1]和POSIX.2c[2]分别定义了操作系统安全核心中可选的(alternative)和附加的(additional)安全机制Capability(命令与系统功能调用控制),MAC(Mandatory Access Control,强制访问控制),Audi(Security Auditing,安全审计),ACL(Access Control Lists,访问控制表),IL(Information Labeling,信息标签)的C接口和shell命令接口,但是对各种安全机制的精确语义和完全实现机制未做出定义。此外,文[1,2]还允许对各种安全机制所包含的内容进行扩充。桔皮书[3](orange book, http://linux.kernel.org/pub/linux/libs/securi),通过给出DAC(Discretionary Access Control,任意访问控制),MAC,Audi,ACL的不同等级的具体功能要求,定义了D,C,B,A类可信操作系统中各种安全级的安全核心的标准(包括评价标准)。其中,Capability是重要成份之一。

目前,许多商业操作系统,如较高版本的SCOUNIX,Trusted FREE BSD等,都有符合文[1,2]的安全机制的实现。但是,出于商业或安全原因,这方面的技术是保密的。

惠普于1995年发布的据称是后来B2级的可信操作系统HP-UX CMW[4]早于文[1,2]。文[4]的Authorization&Privileges的作用(强化用户命令和syscall的权限管理),与文[1]的Capability功能类似,可能制定文[1]时参考了文[4]。文[4]只在管理手册中给出了相关的安全概念和安全机制的使用方法,同样没有提供安全机构的实现机制。

*GNU开放源码的LINUX核心(2.2.5-15以上的*版本)有符合文[1]的Capability机构,其源码和头文件在kernel,fs操作和exec.c等处可以见到。但是,其机制在运行意义上并未真正实现。

文[1]中推荐的Capability部分实现机制,若去除root用户,需要设计全新的权限检查及认证机制,也需要改写几乎每一系统功能处理函数的权限检查代码,工作量巨大。

本文提出一种新的Capability实现机制,其部分原理与用户进程陷入核心获取系统功能服务原理相似。要点是:

(1)核心中允许"root用户"进程运行(具有传统的root用户权限),但不允许root用户登录。当具有被分解的部分root权限的用户进程陷入核心时,它被仿真成root用户进程沿事先由系统指定的对应于该权限的路线推进,完成事先规定的对应于该权限的操作。

(2)通过对用户(进程)和程序的syscall权限进行控制,来实现对用户、程序及命令的权限作控制,进而提供对系统及用户资源的安全保护。

(3)少数无法用系统功能权限控制的root命令(如adduser),则实施关于该命令权限的单独控制。

本文提出的新机制避免了文[1]的缺陷,且能达到同样的安全控制效果。

2 安全模型

2.1 系统目标

设计本文的 Capability 新机制时，追求的目标是使它能做到：(1)强化 DAC，成为一种强制式的系统命令及系统调用权限的安全检查机制；(2)具有一定抗绕过、渗透和欺骗系统安全机构的能力。(3)开发和运行开销不宜过大。

2.2 安全策略

本文机制作用的安全策略是：

(1)从界面上剔除 root 用户，禁止用户以 root 身份登录系统，并且分解 root 特权。系统中只有普通用户，他可拥有部分 root 特权，但不能超越系统规定的用户最大特权组合。被分割的 root 权限是完全分立、独立和受限的。例如：Capability 的授权用户可给其它用户授 root 权限，但却不能执行这些 root 特权。

(2)系统中的每一个可运行程序都具有允许 Pf(Permitted)和可继承 If(Inheritable 亦即程序潜在要求的权限)系统功能权限集合。否则，使用系统规定的默认的程序文件权限。其中，Pf≤If，进程的系统功能请求不能超越 Pf。该权限检测优先级最高

(3)系统中的用户可能拥有被分解的 root 权限。当进程请求系统功能，且满足(2)时，此请求不能超越分配给该用户的 root 权限和系统规定的最大权限组合。进程满足 root 权限检测时，将以虚拟 root 身份执行规定的动作，完成任务后，恢复原普通用户身份。这样，只要在陷入入口处统一处理权限检测，无需修改原来的系统功能处理函数，便能实现 Capability 的功能(包括 root 权限分解和禁止用户以 root 身份登录)，并且可使本机制与 DAC 呈完全独立的叠加关系。

(4)当进程系统功能请求满足(2)、不满足(3)的权限检测时，若该请求不超越进程的有效权限 E(其计算见后)，它将以原本的普通用户身份得到核心的系统功能服务。此后，(C2级以上 OS 要求的)DAC 控制同原来一样有效。

(5)Capability 机制使用的关于用户(进程)和可运行程序的权限信息必须存放在系统中的安全位置，仅供具有相应 Capability 权限的用户和系统 Capability 控制机制操作，以避免攻击。此外，应该保证对该权限信息的访问是高效率的。为此，在本文机制的原型实现中使用了类似对换区的管理方法来存放这些信息，并应用索引和 hash 技术以减少磁盘访问次数，加快数据访问速度。本文 Capability 模型如图1。

2.3 软件体系结构

在本文机制的原型实现中，软件从功能上主要分为三块：

(1)核心的 Capability 控制机制，是2.2节中(2)、(3)和(4)的具体实现；

(2)作为增加的新系统功能的用户接口，供被授权的用户访问和管理 Capability 机制及权限数据。

(3)核心对用户(进程)和程序的 Capability 权限数据的管理和访问，供用户接口和核心 Capability 控制机制调用。其底层通过核心原本提供的内部函数接口完成对 Capability 权限数据的访问。

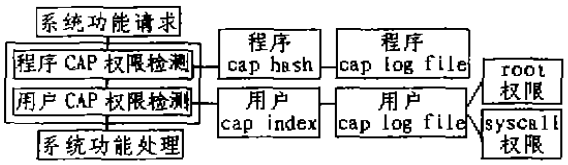

图1 Capability 模型

本文 Capability 机制原型实现的软件体系结构见图2。

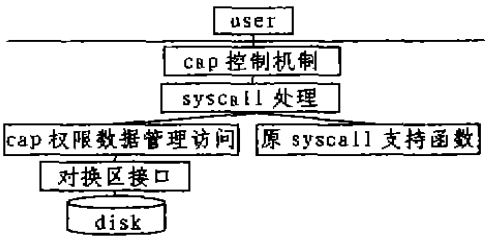

图2 Capability 原型软件体系结构

3 相关问题及其解决

3.1 权限计算

文[1]的 Capability 权限计算的主要目标是满足最小权限原理约束下的权限升降。其公式为：

I1＝I0

P1＝(Pf&&X)‖(If&&I0)

E1＝Ef&&P1

考虑到权限升降的实际意义并不大，且上述运算复杂，本机制在前提：(1)对进程 I＝P，归并 I 和 P，即取消 P；(2)对可执行程序 Pf≤If；(3)对可执行程序取消 Ef 条件下，简化的权限计算公式为：

I1＝I0

E1＝I1&&Pf

此外，由于用户 Capability 的某些 root 权限(可由单独系统功能描述，如 chmod、chown 等)，会与用户系统功能(syscall)权限(如普通用户的 chmod、chown 等)发生二义性冲突。因此，用户的 Capability 的 root 权限数据和普通用户的 syscall 权限数据是分开、存储处理的。

3.2 最小权限原理

最小权限原理是指赋给进程的权限不允许超越它完成任务实际所需要的最大权限集合。为满足该条件，核心为进程加载程序时装入管理员事先为程序分配的Pf和程序潜在要求的If，若前者为空，则装入系统默认的Pf。若后者为空，核心则自动扫描、分析程序可执行代码的符号表(symtbl)，生成它的If。如前所述，Pf≤If。需要指出的是，本机制在满足最小权限原理方面，如同文[1]一样，仅是必要的，并不充分。

3.3 root权限分解

root权限集合可分为二类，第一类的每个root权限对应于系统功能(如chown、chmod、read、write等)，可通过对系统功能的控制实施对它的控制。第二类root权限不能用单个系统功能描述，单个root权限映射为多个系统功能请求的组合(如adduser、passwd等)。需要以命令方式单独对其进行控制。但是，两类root权限的控制原理是相同的：若检测进程有该权限，则虚拟此进程为root进程，完成任务后恢复进程的原uid和gid。

3.4 核心Capability控制原理

本文机制原型实现的Capability控制原理如图3所示。进程的系统功能请求的权限检测满足后，通过syscall entry表获得系统功能服务。此时的DAC机制是正常运作的。获得root特权服务前后，需对进程进行root身份虚拟和恢复，以便在DAC机制下完成root权限的任务。从本质上讲，本文机制对root权限的控制原理与用户进程从3级运行态陷入核心改变为0级运行态，运行在核心，获得核心的系统功能服务后，进程恢复为3级运行态，返回到用户空间的原理是相同的。两者的受控进程都只能沿既定的路线向前推进。

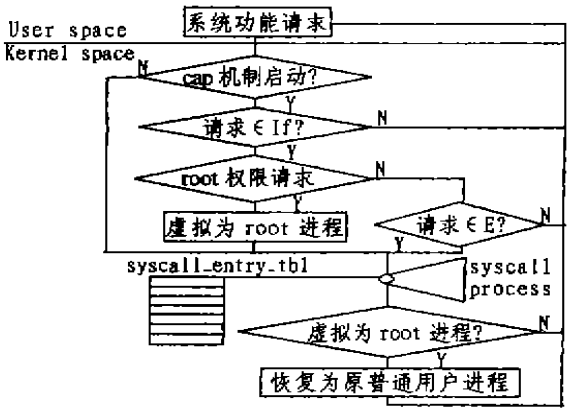

图3 Capability控制原理

本文原型是基于LINUX实现的。需对核心代码做如下修改：

(1)在entry.S增加Capability权限检测代码，实现图3的功能。

(2)在exec.c中增加代码，加载程序的Pf、If，并计算进程的E权限。

(3)在核外程序adduser、passwd等中调用新增的系统功能，开启进程的root命令检测标志；程序退出前，同样调用新增的系统功能，关闭进程的root命令检测标志。以便Capability机制能够对第二类root权限进行Capability的权限检测、控制。

3.5 Capability机制的启动

在login源代码中(执行bsh之前)增加Capability代码：

(1)拒绝root用户登录；

(2)读入进程(用户)的Capability权限信息I(root权限和普通用户系统功能syscall权限)，并装载到PCB(对于LINUX是task_struct)；

(3)开启该进程的Capability机制启动标志，供3.4节中的Capability控制机制检测使用。本文机制之所以设计成这样，是为了在login运行之前的系统初启、init程序、getty程序和系统后台进程(在rc中启动)等，在增加了Capability机制的情况下能够正常运行。

具有Capability机制的系统初建时，系统原始录入两个具特殊Capability权限的用户(CAP-ADDUSER增加用户、CAP_GRANTCAP给用户授Capability权限)的Capability权限信息。

3.6 系统启动后，后台进程的Capability控制

系统启动成功后，Capability授权用户(CAP_GRANTCAP)可通过控制进程的用户接口将后台进程的Capability权限缩减和规范，以防止这些进程的派生子进程以root身份对系统进行恶意的攻击。

结语 最后需要指出的是：

(1)本机制对用户、进程和程序可分别独立进行Capability权限控制，具极好的灵活性和可伸缩性；

(2)本机制可对系统功能、root用户使用的shell命令分别进行控制，可分解root特权，并禁止用户以root身份登录系统；

(3)本机制与C级操作系统的DAC机制呈叠加关系，完全不冲突，大大加强了系统的安全性。

(4)本机制的权限并不能由被授权的用户任意转让于它人，因此不具"DAC"特性；相反，一旦针对进程的Capability机制开启后，核心的Capability权限检测是强制的，因此具有"MAC"特性。

对已实现的基于Linux的原型系统的实际测试表明系统达到了预期目标，证明本文机制是可行的。尚需进行完善的工作有：

(1)文[1,2]仅给出Capability的接口规范。我们

路由器访问表技术研究

On Router Access Control List Technology

胡海璐　陈曙晖　苏金树

（国防科技大学计算机学院　长沙410073）

Abstract　In the paper, the principium of IP Acess Control List has been presented. Also the development trend of Access Control List has been discussed: Dynamic Access Control List, Time Based Accesslist, Reflexive Access Control list, Context Based Access Control.

Keywords　Acess conrtol list, Message filtering, Matching

1　引言

作为一个公共的网络，Internet是缺乏安全性的，因此对内部网而言，必须采取一定的措施，保证内部私有信息的安全。路由器工作在网络层，其主要工作是转发IP报文，它是内部网和Internet的衔接处，所有内部网络到Internet的报文都要流经路由器，尽管路由器非常灵活，并拥有高度的用户配置能力，但它们一般并不查看所传输的数据，也不提供认证远程用户，加密数据传输的流量，或执行代理的功能。因此，必须通过编写路由器的访问表来实现分组过滤，作为防止攻击的第一道防线。

本文将讨论访问表技术在路由器安全中的作用。

2　访问表概述

保证安全有许多手段，其中最为重要的是使用报文过滤来控制报文流进网络。通过检测报文头部的信息来防止特定的报文进出某一个网络的技术称为“报文过滤”。访问表的主要功能就是进行报文过滤。

访问表是根据匹配规则和报文来允许或拒绝报文流的排序表。访问表语句既可以允许报文的流动，又可以拒绝报文的流动，用于允许或拒绝报文的标准是基于报文自身所含的信息。通常这些信息只限于第三层和第四层报文头中所包含的信息。有一种称为基于上下文的访问控制(CBAC)，能够基于应用层信息过滤报文。

传统的访问表报文过滤策略是基于IP报文的源

胡海璐　硕士生，主要研究方向为计算机网络；陈曙晖　讲师，主要研究方向为计算机网络；苏金树　教授，主要研究方向为计算机网络。

有了可行的机制，但更需要好的“策略”，即对系统的Capability权限应做合理的全面设计和规划。

(2)本文机制的原型只实现了用户、进程和程序GET、SET的Capability接口函数，还需完成文[1,2]定义的其它接口函数。

(3)实现其它安全机构：ACL、MAC、Audi、IL，以及它们之间的交互。

(4)在尽量减少开销和复杂度的前提下，补充开发实际有用的Capability权限升降机制。

参考文献

1　Portable Applications Standards Committee of the IEEE Computer Society. Draft Standard for Information Technology-Portable Operating System Interface(POSIX)-Part 1: System Application Program Interface (API)-Amendment #: Protection, Audit and Control Interface [C Language]. New York: the Institute of Electrical and Electronics Engineers, Inc., 1997. 163～194

2　Technical Committee on Operating Systems and Application Environments of the IEEE Computer Society. Draft Standard for Information Technology-Portable Operating System Interface (POSIX)-Part 2: Shell and Utilities-Amendment #: Protection and Control Interface. New York: the Institute of Electrical and Electronics Engineers, Inc., 1997. 25～33

3　Morgan A G. DEPARTJIMENT OF DEFENSE STANDARD: DEPARTMENT OF DEFENSE TRUSTED COMPUTER SYSTEM EVALUATION CRITERIA (Aka. The Orange Book). DoD 5200. 28-STD; Supersedes; CSC-STD-001-83, dtd 15 Aug 83; Library No. S225,711, December 1985

4　Sutton S A. TST. The Hewlett-Packard Compartmented Mode Workstation HP-UX CMW Volume Ⅰ: Administration Tutorial. Urbara, Illinois: Trusted Systems Training, Inc., 1995, chapter 3, chapter 4