计算机科学2002Vol.29№.11

支持动态策略的安全核(Security Kernel)机制的研究*)

Research of Security Kernel Mechanism Supporting Dynamical Policies

吴新勇　熊光泽

(电子科技大学计算机科学与工程学院　成都610054)

Abstract Security of information system requires a secure operation system. Security kernel meets the requirement and provides a bedrock to security of operation system. This paper extracts the deficiency of traditional security kernel, presents a security kernel mechanism supporting policy flexibility, simplified secure interface. It optimizes the performance by reused policy cache, provids a method to revoke granted permissions and assures the atomicity of revocation permissions and granting new permissions. As a result, all refinements help security kernel to improve its flexibility, extensibility and portability.

Keywords Security kernel, TCB, Policy flexibility, Permission revocation

1. 引言

随着网络信息技术的普及,网络设备和信息服务器、终端在政府、国防、金融、商务等各行各业广泛应用,信息安全问题日益突出。操作系统是应用的基石,所以保障信息安全的需求对操作系统本身的安全提出了挑战,需要找到既满足功能、性能要求,又具备足够的安全可信度的操作系统,而研究适用的安全核(Security Kernel)机制是开发安全操作系统中的首要工作。TESEC[1](美国国防部可信计算机评估标准)等标准要求在系统中实现 DAC(自主访问控制)和 MAC(强制访问控制)技术来保证系统的安全,使信息系统具有:保密性(confidentiality)、完整性(integrity)和可获得性(availability),因此,安全核的功能就是进行访问控制。

几十年来,国内外就此课题做了相关的研究:由 AT&T 和 MIT(麻省理工学院)联合开发的 Multics 迈出了安全操作系统设计的第一步,为后来的安全操作系统研制积累了大量经验;美国 Trusted Information Systems 公司以 Mach 操作系统为基础开发了 B3级的 Tmach(Trusted Mach)[2]操作系统,采用了改进型的 BLP 模型;美国 Key Logic 公司开发的以 keykos 为基础的 KeySAFE[3]模型,主要实现了多层安全(MLS);美国尤他大学研究的 DTOS(Distribute trusted operation system)[4]模型,对 KeyKOS、Tmach 等模型系统做了改进,强调了安全策略的灵活性;美国国家安全局参照 FLASK 安全核结构改造的安全 Linux—slinux[5],对支持安全策略的灵活性提出了一些新的思想。我国的安全操作系统起步较晚,有中软 B1级的安全增强型 Linux[6]等等。

以往的研究中多数安全核支持有限的策略灵活性,仅是针对具体的安全策略提出的方案,安全模型缺乏策略通用性,并对权限撤销等重要问题没有提出有效的解决方案,本文通过对传统的安全核(Security kernel)和其他安全模型的分析改进,提出了一种支持多策略的通用的安全核机制。

2. 传统的安全核及其不足

根据 TCB(Trusted Computing Base)的思想,操作系统的元素可分为两大类:主体和客体,主体是可以访问和控制客体的主动实体,包括导致信息在系统中流动或改变系统状态的用户或为用户服务的操作过程(如进程、任务等),客体是系统中被主体使用的能包含或接受信息的被动实体,如文件、内存、消息、网络包,及 I/O 设备等,TCB 实现主体对安全敏感的客体的访问控制机制。安全核是 TCB 的软实现,其功能是对主体和客体实施隔离,把系统的安全策略(Securtiy policy)封装在系统的某组件(安全核)中,只要主体访问了安全敏感信息,安全核都必须根据安全策略对它进行控制和处理,以免被非授权用户(一般以系统任务的形式出现)侵入,造成信息的泄漏、数据的非法修改和误导对系统合法访问的拒绝。传统的安全核不需要系统其他组件的合作,独立承担维护系统安全的重任[7]。安全操作系统中安全核的实现一般有几种方法:虚拟机法、改进/增强法及仿真法等,其原理[8]如图1所示。需要说明的一点是安全核不是从属于操作系统内核或其他任何组件的,它是一种安全保障机制的非功能模块,按 TCB 的思想,一个全面的安全核应是分散式的结构。

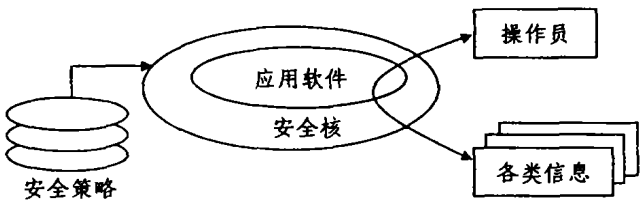

图1　安全核原理

传统的安全核其实仅是一个单纯的请求拦截器(Request interceptor)或引用监视器,在所有应用程序或特定的应用程序和操作系统之间设置一拦截层(如 KeySAFE),它可以位于操作系统内部或外部某组件内,将所有截获的请求重定向到安全核进行处理。这种思想的好处在于:安全代码本地化,很容易生成和修改,甚至可以用基于接口定义的语言编译器自动生成安全代码;因为控制了系统调用接口,所以可以修改请求或请求产生的返回值;对现有操作系统作较小的修改就可加入安全核。

*)本文获国防预研项目基金资助。吴新勇　博士研究生,主要研究方向:系统安全及可靠性。熊光泽　教授,博士生导师,主要研究方向:实时计算机系统及软件开发支持。

但是传统的安全核的策略实现机制存在严重不足，主要表现在：

·安全核没有提供相应的映射机制，所以函数接口必须提供安全策略要控制的所有抽象属性和信息流，并将所有相关的内部状态直接以参数的形式传送到拦截层，往往在决策的时候内部状态已改变，所以无法保证唯一性和原子性。

·另外，直接用参数传递的抽象属性可能是不适用的，例如在基于名字的调用中遇到别名或多控件查询时的映射问题。

·安全层只能够在请求通过或返回时影响系统的操作，系统对安全策略的后续变化不可能作出响应，比如在策略改变后对迁移权限的撤销问题上。

·因为要求提供统一的调用接口，所以增添策略需要重新进行安全编码，策略配置的灵活性差。

对传统安全核的局限性的分析促使其他机制的引入以弥补不足，本文采用了 SID/Sattr（安全标记/安全属性）映射的机制解决原有接口暴露所有抽象属性和信息流的问题，接口通过〈主体 SID，客体 SID，权限〉的三元组或三元集组形式的参数传递信息，简化了接口；在各类组件服务器（如文件服务器、网络服务器）中放置子系统管理器，除完成策略实施外，保证了权限的授予和撤销的原子性，策略的配置以验证器模块数据库的方式实现，减少编码。具体机制将在第4节详述。

3. 安全策略模型

安全策略是一套规则或约束集，它用来管理一个组织如何管理、保护和发布敏感信息。在计算机系统里，我们关心的是访问控制，所有的安全策略都可以看作一个访问控制矩阵（access control matrix）或其中一部分，矩阵的每一行是一系列的〈客体，权限集〉二元组，称为执行环境（execute environment）；每一列是一系列的〈责任者，权限集〉二元组，称为 ACL（Access Control List）[9]。大多数的访问控制策略都是服务于保密性和完整性的目的，以下列举了最常用的几个：

（1）MLS（MultiLevel Secure）：BLP（Bell and LaPadula）[10]安全模型是经典的 MLS 策略，用分层点阵定义了主体和客体的安全等级，其特征是：只有当主体的安全级包含了客体的安全级，才允许主体读客体；只有当主体的安全级被客体的安全级所包含，才允许主体写客体。以上两个特征保证了信息流的单向流动，即只能向高安全层方向流动。

（2）TE（Type Enforcement）模型[11]：TE 将主体和客体分别归并成类（Types）和域（Domains），它们之间是否有访问权由 DDT（Domain Definition Table）决定，DDT 由安全管理员负责管理和维护。

（3）RBAC（Role-Based Access Control）[12]：RBAC 在用户和访问权限间引入了角色（role）的概念，用户按需要与特定的一个或多个角色相联系，角色与一个或多个访问许可权相联系，角色可根据实际需要生成或取消，用户通过会话（session）激活角色，绑定权限，角色其实是 ACL 的载体。RBAC 的优点是策略无关性，角色的概念贴近于现实世界的信息管理系统。

其他的安全策略还有保证信息完整性的 Clark and Wilson[13]策略模型、TE 的发展版本 DTE，保护商业机密性的 Chinese Wall[14]模型。策略的集成性要求所有的决策都要满足它们的约束需求，即各策略决策的交集，安全策略的实现一般以安全代码和策略数据库完成。

4. 安全核体系结构

本安全核体系结构是按决策和实施分离的原则，采用客户/服务器模式，以扩展基于微内核的操作系统为例构建的。

4.1 体系结构及组成

安全决策首先要求拦截对客体的访问请求，获取主客体实体的引用和访问类型，解决的途径就是设置拦截层，传统的 Security 核一般设一个拦截层，拦截所有与安全相关的系统功能调用，但是不能深入获取各子系统内部状态，比如对文件的访问权限保存在打开的文件描述符中，如果权限发生迁移，跟踪和撤消是不可能的，因为撤消要牵涉到内存页管理器，为了保证一致性和原子性，还要调度管理器协调，因此拦截功能被嵌入到各子系统（如内存管理器、文件子系统、网络子系统等）中，整个系统的体系结构如图2所示。

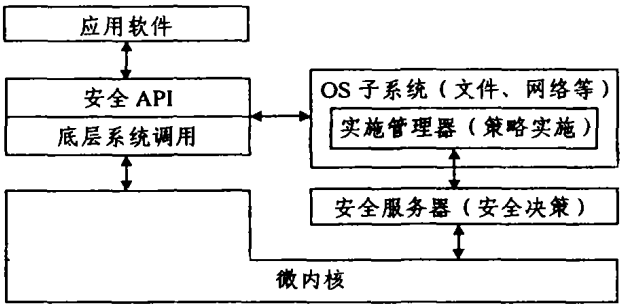

图2 Security 核体系结构

访问控制机制主要由两个组件实现：安全服务器（Security Server）和实施管理器（Enforcement Manager）。

·安全服务器 负责加载和更改安全策略，进行访问仲裁（或决策），负责维护安全标示符/安全属性（SID/Security Attribute）表，SID 和 Sattr 间的映射，给新建的主客体分配安全标示符，管理可重用策略缓存，安全服务器的组成原理如图3所示，安全服务器得到来自实施管理器的请求后，检索与主客体的 SID 相匹配的安全属性，根据安全属性调用不同的策略验证器进行权限验证，安全服务器收集验证结果，并返回所有策略授权的访问模式和请求权限的交集。安全服务器的子组件策略验证器负责获取实体对应的属性，并进行策略逻辑计算，每种策略对应一验证器。当修改策略时，只需要增添或删除相应的策略验证器和策略逻辑即可，保证了对动态策略的支持。

·实施管理器 是监控安全相关实体，接受其请求，并实施策略的组件，在具体实现上，实施管理器被嵌入到进程管理器、网络服务器、文件服务器等子系统中。

除了安全服务器和实施管理器外，安全核还提供了其他的组成部分，包括安全标识符（SIDs）、安全属性（Sattr）、策略解释器、可重用策略缓存：

·安全标识符（SIDs）和安全属性（Sattr） 所有与安全相关的实体都根据特定安全策略做了标记以表示其安全属性，对象根据权限分类，而 SID 是其类别的数字表示，SID 的特点是临时的和本地化的；安全属性是一种包含了绑定到安全相关实体的安全信息的数据结构，它对于除了安全服务器和微内核外所有的组件都是不透明的，比如一个权限集，而不同的 SID 对应了不同的安全属性，SID/Sattr 是映射关系，由策略服务器把实体的 SID 映射为相应的安全属性。使用 SID/Sattr 映射是支持安全标记的举措，避免了原来的安全接口暴露大

量的信息流的问题。

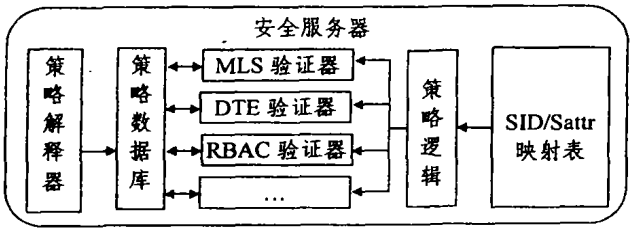

图3 安全服务器组成原理图

·策略解释器 是隶属于安全服务器的组件，负责加载安全策略，创建各策略验证器数据库中的数据元素。

·可重用策略缓存 如果对安全实体的每一次请求都交由策略服务器仲裁的话，系统开销是不可忽略的，可重用策略缓存是优化性能的举措，将仲裁成功的访问请求以〈SID，SID，access mode list〉的形式存储在缓存中，当以后对象管理器需要安全服务器的决策时，它先去查询缓存，获取匹配，而不是直接把主客体双方的 SID 提交给安全服务器，省去了重复决策的代价，提高了系统速度。

4.2 工作原理

在安全核中，实施管理器是 client，安全服务器是 server，而可重用策略缓存是中间载体，同实施管理器一样属于子系统本地化组件，该组件提供管理接口给安全服务器，供安全服务器通知策略发生改变，更新缓存，同时提供检索接口给实施管理器，在权限匹配时进行快速查询。

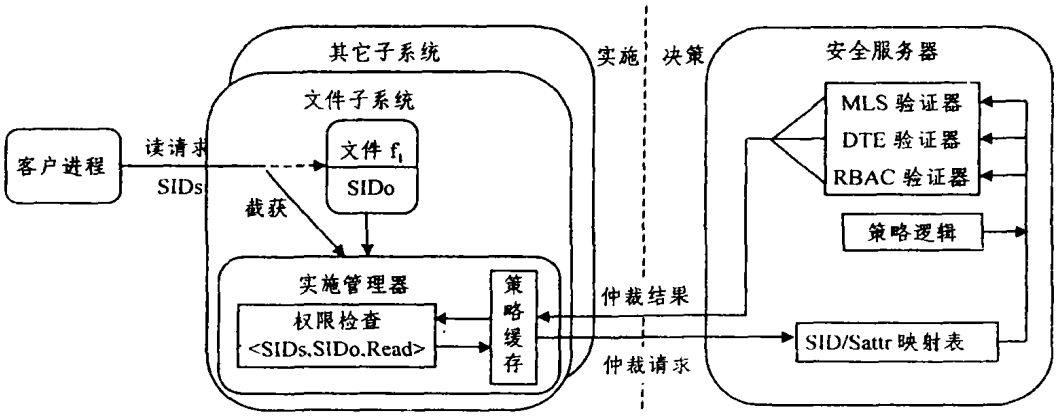

图4 访问控制原理图

整个安全决策和实施的过程是：实施管理器捕获主体的访问请求（该请求由相应子系统传递来的，如进程管理器、文件子系统等），实施管理器先把主客体 SID 和请求的权限组成的〈SID，SID，Permissions〉三元组交给可重用策略缓存的查询接口，在缓存中查找匹配的决策，如有则返回结果，否则，交由安全服务器仲裁，安全服务器在 SID/Sattr 匹配表中查询到主客体的安全属性，再根据策略逻辑调用各策略验证器，策略验证器按访问规则计算，返回结果，安全服务器取各验证器仲裁结果与请求权限的交集，得到最终的访问模式，返回给实施管理器，如交集非空还要在可重用策略缓存中保留一个备份，最后实施管理器根据仲裁结果允许或拒绝访问请求。图4以进程请求读文件为例表述了这一流程。

4.3 权限撤销机制

权限撤销是安全核设计的难点，不但要安全体系结构提供措施，还要求操作系统提供内存管理和调度等支持。只有实现了权限撤销，才能实现动态策略。当安全策略改变时，必须要先撤销过时的策略，才能进行新的授权，并且是一个原子性的过程。对需要撤销权限的进程（或任务）可以采用三种措施：

·立即终止进程运行，并返回访问错误。

·暂停进程，重新进行仲裁，按需要授权或拒绝访问。

·只是等待该进程结束。

很明显，第一种方法太过武断，第三种方法太过消极，都会造成不可预料的后果，而第二种方法可把不可预料性减至最小，避免应用失败，所以在我们的结构中选择支持第二种方法。

具体的机制是在访问请求成功后，在保存结果到可重用缓存的同时，请求者在缓存中同时注册一回调函数，该回调函数实现权限撤销时进程进行的相关处理，比如恢复客体状态。安全服务器在安全策略改变时，调用管理函数升级可重用策略缓存的相关条目，策略缓存再调用注册在该条目的回调函数，暂停进程的执行，检查该进程的内部状态和内存状态，根据更新后的权限继续或结束进程运行，整个过程要求内存管理器和进程调度的支持，以保证安全服务器和可重用策略缓存所需的 CPU 资源，满足撤销的实时性要求，避免撤销过程的死锁。暂停进程操作不能受进程再激活或来自其他进程的激活影响，涉及较多的系统调用，比如对 IPC 权限的撤销可能需要暂停多个进程，所以是安全核中系统开销最大的操作，但是，通常的策略改变是受外部驱动的，非频繁，非周期性的过程，所以不会出现频繁策略改变带来的系统性能的损失。

结束语 Security Kernel 是操作系统安全的核心，给安全应用层提供了基石。本文在分析传统安全核和应用拦截层的作用和不足的基础上，对支持策略灵活性、安全接口简化、权限撤销问题进行了研究，提出了一种新的安全核机制，该机制支持动态策略，简化了接口，可重用策略缓存优化了性能，并给出了一种权限撤销的解决方案。现在，我国的安全操作系统的发展趋势是对现有的自主操作系统和开放源码系统（如 Linux）进行安全增强，因此我们下一步的研究将在国产自主操作系统上实现安全核的原型，并继续对运行时权限撤销问题进行研究。

参考文献

1 NCSC. Trusted Computer System Evaluation Criteria. Department of Defence U. S. A. 1985. DoD 5200. 28-STD

2 Trusted Information Systems, Inc. Trusted Mach System Architecture. Oct. 1995

（下转第140页）

同时接收 $n-1$ 条消息，这在 Ad-Hoc 网络中难以实现。BD 协议在某些环境，如广播 LANs 中具有实际用途。

GDH 系列协议中，GDH.1 效率低，没有实际用途。GDH.2/3需要 $M_n(M_{n-1}、M_n)$ 具有广播功能。其中，GDH.2的消息数和交换数达到理论中的下界，但轮数 n 却大于 $\lceil \log_2 n \rceil$，另外，M_i 的幂运算数随 i 的增大而增大，总的幂运算数为 $O(n^2)$，计算量较大；GDH.3 的消息数和交换数为 GDH.2的2倍，但计算量却小。因此 GDH.3适合于较大的情况。

就安全性来说，GDH 协议可抵抗被动攻击(基于 DDH 问题难的事实)，但不能抵抗主动攻击，而且广播消息的成员更易受到攻击。Steiner et.al. 提出了带部分认证的 A-GDH.2协议[5,13]，以及完全认证的 SA-GDH.2协议，并证明了协议在主动攻击下的安全性。另外，Asokan 和 Ginzboorg[3]引入盲因子，提出了带密码认证的 GDH.3协议，这些协议适用于一些特定的 Ad-Hoc 环境。

超立方体协议的轮数 $d=\lceil \log_2 n \rceil$ 达到理论中的下界，但消息数($n\log_2 n$)和交换数($(n\log_2 n)/2$)却大于最优值，总的幂运算数为 $O(n\log_2 n)$，优于 GDH.2。超立方体协议最大的缺点在于对网络拓扑结构的要求，这对动态的、连接不可靠的 Ad-Hoc 网络而言难以实现。但在某些成员数较少的 Ad-Hoc 环境中，Asokan 和 Ginzboorg[3]提出了具有容错功能、带密码认证的超立方体协议，既获得了较高的效率，又保证了协议的安全性。

基本的 Octopus 协议中心为4个节点($d=2$)，其交换数为 $2n-4$，在没有广播功能的情况下达到理论中的下界值，消息数 $3n-4$ 高于 $2n-2$，总的幂运算数为 $4n+12$。由于 Octopus 协议用超立方体作为中心，因此该协议与超立方体协议有相似的特点，轮数少，但难以维持网络的拓扑结构。

综上所述，以上协议均不能使每项效率参数都达到最优值。因此，上述协议只在一些特定的 Ad-Hoc 环境中才适用。

结束语 本文结合 Ad-Hoc 网络的特点，分析比较了各种密钥协商协议的效率性、安全性。它们有的需要节点具有广播能力，有的需要维持某种特定的拓扑结构，这些在 Ad-Hoc 网络中都很难实现，因此应根据实际的情况加以选择应用。而如何实现一个满足任意拓扑结构、高效的密钥协商协议仍然是一个公开的问题。

参考文献

1 Zhou Lidong, Haas Z J. Securing Ad-Hoc Networks. IEEE Network Magazine, 1999, 13(6)

2 Karpijoki V. Security in Ad-Hoc Networks. Tik-110.501, seminar on network secutity, Helsinki University of Technology Telecommunications Software and Multimedia Lab., Nov. 2000. Available at: http://citeseer.nj.nec.com/hietalahti01key.html

3 Asokan N, Ginzboorg P. Key agreement in ad hoc networks. Computer Communications, 2000, 23: 1627～1637

4 Hietalahti M. Key Establishment in Ad-Hoc Networks: [technical report of Helsinki University of Technology Lab. for Theoretical Computer Science]. 2001. Available at: http://citeseer.nj.nec.com/hietalahti01key.html

5 Ateniese G, Steiner M, Tsudik G. New Multiparty Authentication Services and Key Agreement Protocols. IEEE Journal of Selected Areas in Communications, 2000, 18(4)

6 Stajano F, Anderson R. The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks. In: Proc. of the Seventh Intl. Workshop on Security Protocols, April 2000. Available at: http://citeseer.nj.nec.com/stajanoresurrecting.html

7 Diffie W, Hellman M E. New Directions in Cryptography. IEEE Transactions on Information Theory, 1976, IT-22(6): 644～654

8 Maurer U M, Wolf S. The Diffie-Hellman Protocol. supported by SNF, No. 20-42105.94, July 1999. Available at: http://citeseer.nj.nec.com/maurer99diffiehellman.html

9 Steiner M, Tsudik G, Waidner M. Diffie-Hellman Key Distribution Extended to Group Communication. 3rd ACM conf. on Computer and Communications Security, March 1996, New Delhi, India. Available at: http://citeseer.nj.nec.com/steiner96diffiehellman.html

10 Becker K, Wille U. Communication complexity of group key distribution. Fifth ACM Conference on Computer and Communications Security, Nov. 1998. Available at: http://www.zurich.ibm.com/security/publications/1998/BecWil98.ps.gz

11 Ingemarsson I, et al. A conference key distribution system. IEEE Transactions on Information Theory, 1982, 28(5): 714～720

12 Burmester M, Desmedt Y. A secure and efficient conference key distribution system. Advances in Cryptology. In: Proc. of EUROCRYPT'94, Springer-Verlag, 1995

13 Ateniese G, Steiner M, Tsudik G. Authenticated Group Key Agreement and Frinds. In: Proc. of the 5th ACM Conf. on Computer and Communication Security, Nov. 1998, San Francisco, CA. Available at: http://citeseer.nj.nec.com/ateniese98authenticated.html

(上接第156页)

3 Key Logic, Inc. Introduction to KeySAFE. Key Logic Document SEC009

4 Secure Computing Corporation. DTOS Lessons Learned Report. DTOS CDRL A008, June 1997

5 Loscocco P, Smalley S. Integrating Flexible Support for Security Policies into the Linux Operating, NSA Labs, Jan. 2001

6 中软安全增强 Linux. http://linux.cosix.com.cn

7 黎忠文，熊光泽. 安全(Safety)内核机制的研究与实现. 计算机科学，2001，28：87～90

8 King R. Safety kernel enforcement of software safety policies: [Doctor Thesis]. USA: University of Virginia, 1995

9 Graham G S, Denning P J. Protection - principles and practice. In: Proc. AFIPS 1972 SJCC, AFIPS Press, 1972, 40: 417～429

10 Bell D E, La Padula L J. Secure computer systems: Mathematical foundations and model: [Technical Report M74-244]. The MITRE Corporation, May 1973

11 O'Brien R C, Rogers C. Developing applications on LOCK. In: Proc. 14th National Computer Security Conf. Washington, DC, Oct. 1991. 147～156

12 Ferraiolo D F, Cugini J A, Kuhn D R. Role-Based Access Control (RBAC): Features and motivations. In: Proc. of the Eleventh Annual Computer Security Applications Conf. Dec. 1995

13 Clark D D, Wilson D R. A comparison of commercial and military computer security policies. In: IEEE Symposium on Security and Privacy, Oakland, CA, April 1987. 187～194

14 Sandhu R S. A lattice interpretation of the Chinese Wall policy. In: Proc. of the 15th NIST-NCSC National Computer Security Conf. Baltimore, MD, Oct. 1992. 221