计算机科学 1996 Vol. 23 No. 1

安全内核方法与实现考虑

林宣雄　李怀祖　张文修
（西安交通大学　西安 710049）

摘　要　Method of security kernel is based upon the conception of reference monitor, which reguires to define security policies explicitly and to observe a rigorous methodology in order to form a precise specification of behavior and coding in a high level language. By limiting the protection mechanism to a small portion of a system, it is rather easy to ensure the correctness and robustness of the part of security kernal, thereby escaping from the complexity of designing a whole operating system, so this method is a promising was in realization of security and privacy of computer.

关键词　Information protection, Security kernel, Reference monitor, Policy model.

一、引　言

现代信息系统的一个突出标志就是信息系统网络化，信息处理和存贮分布化。在这样复杂的网络环境下实现信息的安全保密，是一个复杂的系统工程。本文讨论的安全内核方法和技术是迄今为止，在信息安全保密方面研究的一个重要成果。安全内核方法为用有条理的设计过程代替智力游戏从而构筑安全的计算机系统，并进而为信息系统的开发建立安全的平台提供了理论基础，人们在研究可靠的内部控制方面进行了不懈的努力，但总归失败，究其原因：①操作系统通常十分庞大和复杂；②至今无人精确定义过由内部控制提供的安全的含义；③无人证明过已经实现的安全控制的正确性。安全内核方法把保护机制限制在系统的一个很小的部分内，避免了由于系统的庞大规模和复杂性所带来的问题。在安全内核方法下，要求明确定义安全策略，并遵循严格的一套方法（其中包括建立相应数学模型）来形成精确的行为规范以及高级语言编码，这样也就解决了②和③两个问题。

安全内核方法是基于存取监控器（Reference monitor）概念的一种安全保密方案。存取监控器源于 Lampson 提出的模型[1]，为保护思想的实现提供了基本的安全理论。在存取监控器（图 1）中，所有主体（Subject）必须根据系统存取权表来实现对客体（Object）的存取，对客体的每一次存取以及授权的改变都必须通过存取监控器。

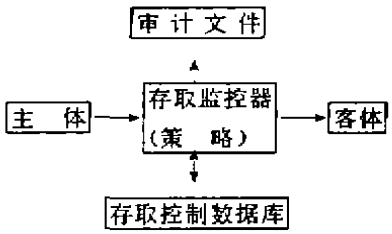

图 1　存取监控器示意图

安全内核由实现存取监控器的硬件和软件构成。安全内核概念由 Schell 在 1972 年首先提出，1974 年，Mitre 证实了构筑安全内核是可能的。第一个安全内核由不到 20 个原语构成，直接管理硬件资源并实施保护限制。整个安全内核由不到 1000 条、可编译的高级语言语句构成并运行在 DEC 的 PPP11/45 上。

为了证明这个安全内核的正确性，Mitre 构造了一个简单的实验性操作系统和应用程序。该操作系统由树型文件系统、支持受控信息共享的合作进程以及少量交互式终端的接口构成。操作系统和应用程序处在安全内核之上，它不影响由安全内核所

林宣雄　在职博士生，主要从事计算机安全，信息系统安全和计算机网络方面的研究。李怀祖　博士导师，主要从事决策理论、决策专家系统方面的研究。张文修　博士导师，主要从事模糊数学与集值随机过程等研究。

提供的信息保护。

自从这个初始安全内核原型问世以来，引发了一系列关于安全内核构造和验证的研究。今天，已经有一些基于安全内核的系统问世，Honeywell 的安全通信处理机(Scomp)就是一例[2]，文[3]对其它的基于安全内核的系统作了介绍。

二、安全内核方法的基本原理

开发基于安全内核系统的第一步是确定安全策略，在计算机系统中，一个好的安全策略应该包含主体对客体许可的所有存取方式。这种要求精确定义安全策略的方法是基于安全内核的系统与其它具有安全功能的操作系统的根本区别所在。后者趋向于实现通用的保护，而没有任何安全定义和安全标准，因而在那些系统中的保护机制基本上是给计算机提供了一些安全功能。相反，基于安全内核的方法要明确地解决策略和机制两个问题。不但要设计保护机制，而且要确定安全策略，按照安全策略实现保护。

1. 安全模型的形式定义

我们需要定义两类安全策略：指令型(nondiscretionary)安全策略和选择型(Discretionary)安全策略。前者包含有强制性的施加于所有用户的安全规则，而后者则由用户自己选择安全规则。

由安全内核实施的安全策略应体现在一组数学规则中，构成形式化的安全模型；选择型和指令型安全策略必须由这种模型规则来实施。

在评价一个安全模型是否有效时，需要考虑两个关键的问题：一是安全模型必须从总体上定义系统的信息保护行为；二是安全模型中必须符合安全定理，以保证模型下的行为与相关策略的安全要求相一致。

大多数安全内核实施的安全模型都是基于 Mitre 和 Case Western Reserve 大学早期的安全内核研究成果[3,5]，这种安全模型称为 Bell/Lapadula 模型，为防信息的非法存取提供了规则。通过把安全内核表示为有限状态机，这些规则允许系统从一个安全状态向下一个安全状态迁移。

在该模型中，存取监控器的每一个主体和客体都被给定一个存取类(由密级和类别构成)。在决定主体是否允许存取客体时，主体和客体的存取类被比较。通过把存取类以网格的形式来组织，就能支持更为广泛的安全策略。这种网格定义了存取类间的相互关系，可以决定一个存取类是大于等于还是小于另一个存取类，或者毫不相干。

对于模型的指令型规则，有两个是基本的。其一称作简单安全条件，规定只有主体的存取类大于或等于客体的存取类时，主体才能读取客体的内容，这个简单的安全条件阻止了用户直接读取其无权读取的数据的行为。其二称作星性质(star property)，该性质有助于防止所有主体对于客体的不正当读取，除非主体的存取类小于等于客体的存取类，否则主体不可以修改客体。第二个基本规则可以防止特洛伊木马的进攻。

模型的指令型规则并不提供在同一个存取类中区分不同用户的安全策略，选择型规则提供了这种类型的安全策略。Bell/Lapadula 模型的选择型规则允许授权用户和程序按用户名或其它信息任意地授予或收回信息存取权。既然选择型规则多多少少有点任意性，我们就不能对信息的流动作出许多绝对的说明。这样，在这种控制下，特洛伊木马进攻就比在指令型控制下难于避免，因此指令型控制总是被优先考虑。

除了信息不合适的泄露和修改两种威胁外，还有一种威胁叫做服务拒绝(即系统崩溃或系统不再响应用户请求)，这种威胁在绝大多数模型中没有提及，而基于安全内核的系统则可能可以承受这种威胁，至少不会比常规系统差，但有关服务拒绝的规则更难以在模型中形成。

2. 正确的实现

在确定内核所应提供的功能时，数学模型很有用，但是它并不规定应用程序与内核接口的具体设计。为了克服模型与实现间的差距，开发进程必须被分解成一系列工作步骤，一个普遍使用的技术是应用分层抽象规范到安全内核的设计中。对于每一步，十分重要的工作是证明其安全性，以使最终的系统的安全性有保障。

我们可以使用许多形式化、非形式化的方法来证明子系统的安全性，存取监控器机制并不强制任何一种方法。在内核方法形成的早期，人们认识到形式化规范和数学验证为实际证明提供了潜在可能性，这种实际证明是关于内核的实现是否正确地遵循了模型规则的证明。那些形式化方法在证明模型、层次规范和高级语言实现之间的一致性方面得

到程度不同的应用(图 2)。

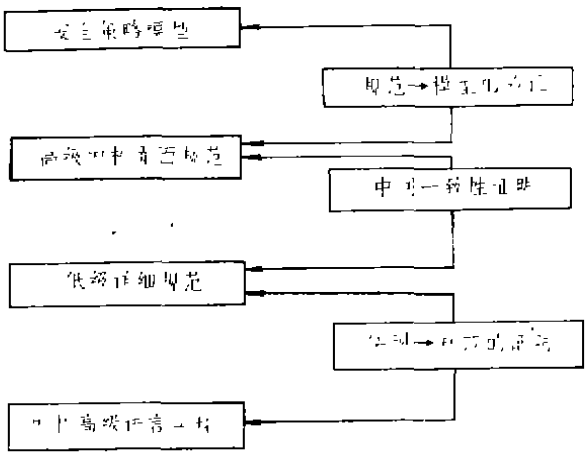

图 2 开发和验证的层次

正如任何操作系统设计工作一样,规范准备也是一项创造性的劳动。内核规范定义了所有内核的界面特点。我们能够使用高级内核规范来判断功能以及证明界面是否遵循了模型规则。一旦接口功能被确定,其安全性能就被精确地建立,通过逐渐引进更多的实现细节,就可以扩展功能,这不会影响已经建立的安全性能的有效性。

三类形式证明技术已经被应用到内核研制的不同阶段(图 2),每一类中又有若干个技术可以使用。第一类技术被用来检查和证明内核行为是否关于安全模型安全,安全流分析技术对于确定和分析规范中的信息流是一个相对简单的方法[5]。注意,仅仅接口规范的安全性必须证明,而不是证明更为困难的功能正确性。既然绝大多数功能是与安全不相关的,因而模型不需去考虑它们。

在第二类形式证明技术中,我们验证在层次结构中的任何中间规范和界面规范间的映象的一致性和正确性。第三类证明技术即最传统的证明正确性方法,用来证明内核实现与其规范的一致性。

三、安全内核的实现考虑

为了成功地实现一个基于安全内核的系统,我们必须考虑结构和工程两方面的问题。这在研制其它系统时可能不会碰到。虽然安全内核方法能被应用到所有类型的系统,但是这两方面的考虑在具有在线、交互式用户的通用操作系统的环境中能得到最好的体现。正如前面所述,安全内核提供了操作系统功能的相对小和简单的子集,内核原语是该子集与操作系统的其余部分(通常称为管理程序)的接口。同样,管理程序原语提供了由应用程序使用的操作系统功能。

1. 内核与管理程序间的折衷

一个操作系统通常被划分成一个个功能模块,如进程管理,文件系统管理和 I/O 控制等。在每一个功能模块内,一些功能是明显与安全有关的,因而必须放在内核中,而有一些则与安全无关。安全模型的规则有助于清晰地确定哪些功能是与安全有关的。

内核必须控制操作系统中管理多用户共享的(诸如内存、磁盘等)资源的那些部分。这些部分之所以放在内核中,是因为安全模型要求将那些资源虚拟化使得它们不能被不可信软件所直接访问。那些提供通用的、并不管理任何共享资源的实用程序以及那些处理服务拒绝的程序不在安全策略范围内,因而一般处在管理程序层内。

在实践中,我们通常能够应用模型的细节和特殊安全策略来决定什么必须包含在管理程序层中,然而工程实现经常迫使我们考虑把与安全无关的功能放在内核内。例如,把与安全无关的操作系统文件名解释机制从内核文件管理系统中区分开来就是一件十分麻烦的事情。在设计基于内核的系统时,经常需在性能、功能和复杂性间作出折衷,而当仿真一个功能不易被重新分配的已存操作系统时,折衷显得更加重要。

2. 可信主体

绝大多数系统要求安全策略迎合它们需要的基于满足基本安全模型的要求。这种特殊的策略一般为不频繁操作施加了限制,仅仅可能被应用在特殊环境下或应用到特殊用户上。实现这样一种扩展策略的内核通常提供一组接口。这组接口仅仅由确定的可信主体所调用(图 3)。这就是通过诸如权限指示器的一些内部标识号来识别软件。当一个运行程序具有这样权限时,它可能无需得到内核的存取检查就能完成相应的功能。

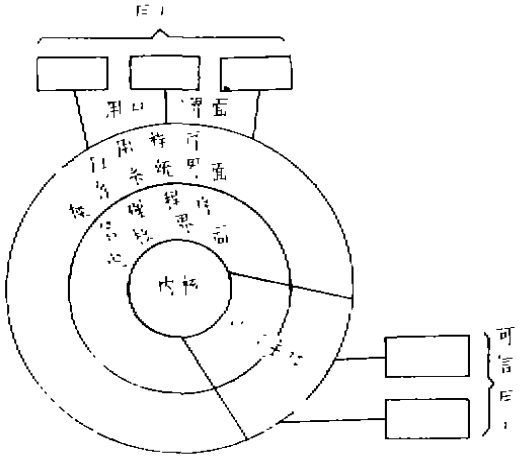

图 3　基于内核的操作系统的结构

可信主体通常完成系统维护以及控制实施于非信任主体之上的存取。例如，一个安全管理员必须被给予维护内核用来为用户指定存取类的权力，其所用的实现这一功能的软件就是一个可信主体，有时，一般的用户会调用确定的可信主体来完成某些安全敏感的功能。例如，安全模型不允许一个不可信主体访问其存取类的信息，而通过信任主体的中介就可以满足这种偶尔的要求。

可信主体经常作为异步进程来实现，这种进程也叫可信进程，或者作为内核的扩充叫做可信功能。

3. 软硬件要求

内核方法可能需要相当的硬件支持以获得足够的性能。这里有两个极端的情况：一是完全用软件实现在常规机器上的安全内核，使得内核作为一个单纯的解释器，在这种情况下，没有特殊的安全要求实现在硬件上。二是所有内核功能由硬件指令来实现，在这种情况下，硬件将担起全部安全工作，因为有了内核和管理程序间的折衷，因而一个具体的选择将受复杂性、规模和性能三方面的影响。

实现一个基于内核的操作系统所需的硬件和软件应是成熟的，从微型机到大型机，许多计算机结构都提供了实现基于内核的系统的硬件支持。几个已经实现的基于内核的系统由于缺乏足够的硬件支持导致了系统性能可观的下降。然而，只要有合适的硬件，就没有理由相信基于内核的操作系统的性能会比非基于内核的操作系统的性能差。R. Schell 给出了实现基于微处理器内核系统所需的硬件支持的例子[6]。

关于支持基于内核的通用操作系统所需的硬件支持和软件机制方面，有四个一般的结构性考虑，它们是：

●清晰的进程——多进程和进程间通信的有效支持；

●存贮器保护——大分段虚拟存储器，存储器的存取控制，以及清晰标识的客体；

●执行域——最少三个域（用户、管理员、内核）以及域间的有效控制转移。

●I/O 调解——I/O 设备、外部介质、存储器的控制存取。

如果能够满足上述四个结构要求，就能够为安全内核的构造打下良好的基础。

参考文献

[1] B. W. Lampson, Protection, Proc. Firth Princeton Symp. Information Sciences and Systems Mar. 1971

[2] Laster J. Fraim, Scomp: A Solution to the Multilevel Security Problem, Computer, Vol. 6, No. 7 1983

[3] O. E. Bell 等, Computer Security Model, Unified Exposition and Multics Interpretation Tech. Report ESD-TR-75-306, AD A023588, the Mitre Comporation, Beedford, Mass., June 1975

[4] K. G. Watter 等, Structured Specification of a Security Kernel, Proc. 1975 Int'l Conf, Reliable Software, IEEE Cat. No. 75 CH0940-7CSR, Los Angels, Calif., Apr., 1975

[5] J. K. Millen, Operating System Security Verification, Case Studies in Mathematical Modeling, W. E. Boyce, ed., Pitmann Publishing, Marshfield, Mass., 1981

[6] R. R Schell, The Structure of a Security Kernel for a Multiprocessor Microcomputer, Computer, Vol. 16 No. 7, 1983