操作系统安全性研究的演进

石文昌 李伟楠 李翰超 贾春福

从20世纪40年代第一台计算机的诞生，到今天信息化逐渐成为时代潮流。与之相应，信息安全问题也成为世界面临的重要挑战。作为计算机系统中最基础的软件，操作系统在信息安全中扮演着至关重要的角色。权威研究报告显示，超过50%的信息安全问题源自软件工程中的安全缺陷，而其中很多问题都可以追溯到操作系统的安全脆弱性。本文将从历史角度探寻操作系统安全性研究的演进历程。

小荷初露尖尖角

早在20世纪60年代，计算机资源共享系统中的安全控制问题就引起了美国国防部的高度重视。1967年，美国国防科学部组建了计算机安全特别机构，拉开了操作系统安全性研究的序幕。在之后10年里，随着各方面研究工作的开展，操作系统安全性的基本理论和安全系统设计的主要思想逐步建立起来。

1969年，魏茨曼（C. Weissman）发表了历史上第一个安全操作系统Adept-50的研究成果。该系统运行于IBM/360硬件平台，以形式化的高水标（High-water-mark[1]）安全模型为基础，实现了美国的一个军事安全系统模型。与现实世界中将信息划分为绝密、机密、秘密和非秘等敏感级别相呼应，该系统为诸如文件等被访问的客体标识了敏感级别。系统实现的基本安全原则是：对于读操作，不允许信息的敏感级别高于用户的安全级别；对于写操作，在授权情况下，允许信息从高敏感级别移向低敏感级别。

同年，借助形式化表示方法，兰普森（B.W. Lampson）第一次运用主体、客体和访问矩阵的思想对访问控制问题进行了抽象。主体是访问操作中的主动实体，客体是访问操作中被动实体，即主体对客体进行访问。访问矩阵是以主体为行索引、以客体为列索引的矩阵，矩阵中的每一个元素表示一组访问方式，是若干访问方式的集合。矩阵中第i行第j列的元素M_{ij}记录着第i个主体S_i可以执行的对第j个O_j客体的访问方式，比如$M_{ij}=(read,write)$表示S_i可以对O_j进行读和写访问。

1970年，在一份研究报告中，维尔（W.H. Ware）研究了多渠道访问的、共享资源的计算机系统引起的安全问题，结合实际国防信息安全等级的划分，将共享系统中的敏感信息划分等级，为实现多级安全系统提供了依据。报告指出，安全级别和需知权限是多级安全问题的重要成分，而基本的多级安全问题就是要确定具有特定安全级别和需知权限的个体是否能够访问在给定物理环境中的某个范围内的敏感信息。它强调计算机安全系统必须与现实的安全等级划分结构一致。

1972年，在资源受控共享的背景下，

[1] 这里water-mark指数值大小界限

作为美国空军一项计算机安全规划的研究成果，安德森（J.P. Anderson）在一份研究报告中提出了引用监控机、引用验证机制、安全核和安全建模等重要思想。他提出将授权机制与控制进程运行的系统环境结合，以支持资源的受控共享。授权机制负责确定进程对资源的访问许可权，而控制进程运行的机制负责把用户进程对资源的访问限制在授权的范围之内。引用监控机则是进程运行控制的具体应用，提供了对共享资源访问权限的验证。

引用监控机的实现称为引用验证机制。它要实现引用监控机的硬件和软件的组合，其设计要满足以下3个原则：

1. 必须具有自我保护能力；

2. 必须总是处于活跃状态；

3.必须设计得足够小，以利于分析和测试，从而能够证明它的实现是正确的。

安全核是系统中与安全性实现相关的部分，包括了引用验证机制、访问控制机制和授权管理机制等成分。

1973年，在研究对程序的限制问题时，兰普森提出了隐蔽信道的概念。隐蔽信道，是指那些按照人们的常规认识不会被用来传送信息但实际上却被利用于泄漏信息的信息传送渠道。隐蔽信道的存在意味着可能会出现隐性的信息泄露问题，威胁到系统的安全。

同年贝尔（D.E. Bell）和拉帕杜拉（L.J. Lapadula）提出了第一个可证明的安全系统的数学模型，即Bell & Lapadula模型，简称BLP模型。该模型通过主体和客体的安全标记来实现访问控制，是一个基于状态机的系统模型。

BLP模型定义的状态是一个四元组（b,M,f,H）。其中，b是当前访问的集合，当前访问由三元组（主体，客体，访问方式）表示，是当前状态下允许的访问；M是访问控制矩阵；f是安全级别函数，用于确定任意主体和客体的安全级别；H是客体间的层次关系。抽象出的访问方式有4种，分别是只可读r、只可写a、可读写w和不可读写（可执行）e。主体的安全级别包括最大安全级别和当前安全级别，最大安全级别通常简称为安全级别。简单安全特性、星号安全特性、自主安全特性、基本安全定理构成了BLP模型的核心内容。

简单安全特性（ss–特性）：

如果（主体，客体，可读）是当前访问，那么一定有：

$level(\text{主体}) \geq level(\text{客体})$

其中，$level$表示安全级别。

星号安全特性（*–特性）：

在任意状态，如果是当前访问，那么一定有：

1. 若方式是a，则：

$level(\text{客体}) \geq current\text{-}level(\text{主体})$

2. 若方式是w，则：

$level(\text{客体}) = current\text{-}level(\text{主体})$

3. 若方式是r，则：

$level(\text{客体}) \leq current\text{-}level(\text{主体})$

其中，$current\text{-}level$表示当前安全级别。

自主安全特性（ds–特性）：

如果（主体-i，客体-j，访问方式-x）是当前访问，那么，访问方式-x一定在访问控制矩阵M的元素M_{ij}中。

与ds-特性处理自主访问控制相对应，ss-特性和*-特性处理的是强制访问控制。自主访问控制的权限由客体的属主自主确定，强制访问控制的权限由特定的安全管理员确定，由系统强制实施。

基本安全定理：如果系统状态的每一次变化都能满足ss-特性、*-特性和ds-特性的要求，那么，在系统的整个状态变化过程中，系统的安全性是不会被破坏的。

与BLP模型的思想相似的还有同一时期由比巴（K.J. Biba）提出的比巴模型。比巴模型的目的是对系统的完整性加以保护，防止对信息的非授权更改。所谓完整性，是指信息不会被用户非法修改，系统不会被破坏，能够完成其正常的工作。比巴模型中每个客体都有一个完整性标签，标记其完整级别，而信息的流向只能从高完整级别流向低完整级别，即人们只相信完整级别较高的信息。

1975年，邵泽（J.H. Saltzer）和施罗德（M.D. Schroeder）总结了保护机制的体系

结构并提出了安全机制设计的八大原则。他们深入分析了访问控制机制的两种典型的实现模式：基于权能（Capability）的模式和基于访问控制表（ACL）的模式。权能模式是一种面向门票的模式，对应访问控制矩阵中的行结构，以主体为中心，确定每个主体能够访问的所有客体。访问控制表模式是一种面向名单的模式，与访问控制矩阵中的列结构相对应，以客体为中心，确定能够访问该客体的所有主体的名单集合。由他们提出的安全保护机制设计的八大原则是：经济性原则、失败-保险原则、完全仲裁原则、开放设计原则、特权分离原则、最小特权原则、最少公共机制原则和心理可接受性原则。为讨论信息保护问题，从概念上，可以为每一个需保护的客体建立一个不可攻破的保护墙，保护墙上留有一个门，门前有一个卫兵，所有对客体的访问都首先在门前接受卫兵的检查。在整个系统中，有很多客体，因而有很多保护墙和卫兵。对客体的访问控制机制的实现结构可分为两种类型：面向门票（ticket-oriented）的实现和面向名单（list-oriented）的实现。在面向门票的实现中，卫兵手中持有一份对一个客体的描述，在访问活动中，主体携带一张门票，门票上有一个客体的标识和可访问的方式，卫兵把主体所持门票中的客体标识与自己手中的客体标识进行对比，以确定是否允许访问；在整个系统中一个主体可能持有多张门票。在面向名单的实现中，卫兵手中持有一份所有授权主体的名单及相应的访问方式，在访问活动中，主体出示自己的身份标识，卫兵从名单中进行查找，检查主体是否记录在名单上，以确定是否允许访问。

1976年，哈里森（Harrison）、鲁泽（Ruzzo）和乌尔曼（Ullman）等提出了操作系统保护体系的第一个基本理论，简称HRU理论。HRU理论的核心内容由操作系统保护体系的形式化模型和相关的三个定理组成。该理论定义的保护系统∑由一个通用权限集***R***和一个命令集***C***组成。保护系统∑的配置（或者说状态）***Q***由一个三元组（***S,O,P***）表示，其中，***S***为主体集，***O***为客体集，***P***为主体对客体拥有的权限集。HRU理论中的访问控制机制是基于命令实现的。对于给定一系列的参数，命令函数通过检测其中的权限是否合法来确定下一步要执行的操作，从而使系统从一个状态转换到另一个状态。相关的三个定理是：

定理HRU1：存在一个算法，确定一个给定的单一操作的保护系统和初始配置对一个给定的普通权限***r***是否是不安全的。

该定理说明一个单一操作的保护系统是否“不安全”是可判定的。

定理HRU2：一个给定的保护系统的一个给定的配置对一个给定的普通权限是否是安全的，这是不可确定的。

该定理说明，一个系统是否安全是难以判断的。

定理HRU3：没有 create（创建）命令的保护系统的safety（安全性）问题在多项式空间中是完全的。

HRU理论证明了这些定理的正确性。通过这些定理，HRU理论给出了操作系统中有关访问权限判定的一系列具有重要意义的结论。

1979年，在描述一个基于安全核的计算机安全系统的设计方法时，尼巴尔都（G.H. Nibaldo）建立了可信计算基（Trusted Computing Base，TCB）的思想。他把计算机系统中安全相关的部分与系统的其他部分区别开来，这个安全相关的部分就是可信计算基。

从20世纪60年代末至20世纪80年代初，操作系统安全性的研究经历了从无到有的探索过程，安全Multics、Mitre安全核、UCLA（加州大学洛杉矶分校）数据安全UNIX、KSOS和PSOS等一系列安全操作系统相继诞生，操作系统安全性的基本思想、理论、技术和方法逐步建立，为以后操作系统安全性的研究奠定了重要的基础。

七色彩虹引领方向

随着安全操作系统陆续从实验室中诞生，人们需要一些有效的评价体系来衡量它们所能提供的安全支持能力，以便能够比较不同系统的安全性，从而为实际应用在系统

选择方面提供依据。

1983年，美国国防部颁布了世界上第一个计算机安全评价标准，即可信计算机系统评价标准，简称TCSEC[2]，称之为橘皮书。1985年，美国国防部颁布了TCSEC的修订版。

TCSEC颁布之后，与之相配套的一系列规范不断问世，比如，关于数据库的规范、关于网络的规范、关于隐蔽信道的规范以及关于形式化的规范，等等。每一种规范文本的封面都呈现不同的颜色，因此人们把TCSEC标准及其相配套的这一系列规范统称为彩虹系列。

当时，安全操作系统的用户主要是美国军方和政府。美国政府规定，只有通过TCSEC标准评价的操作系统才有资格销售到政府和军事部门。所以，彩虹系列一度左右着安全操作系统等安全系统的研究与开发。

TCSEC是在安全核思想的指导下设计出来的，其中的核心概念可信计算基正是安全核的体现。TCSEC定义了D、C1、C2、B1、B2、B3、A1等七个由低到高的系统安全性评价等级。在政府机构和国防部的主导下，业界出现了一批以TCSEC标准作为衡量尺度的系统实现。

1984年，AXIOM技术公司的克莱默（S. Kramer）发表了LINUS IV系统的设计与开发成果。LINUS IV是以4.1BSD UNIX为原型、结合TCSEC标准的要求开发的实验型安全操作系统。它支持基于访问控制表结构的自主访问控制，在强制访问控制中实现了BLP模型的思想，采用隐藏子目录法实现了对 /tmp 等共享目录的支持，实现了基于事件严重程度的安全审计，实现了超级用户的特权分离。

1986和1987年，IBM公司的格里格（V.D. Gligor）等发表了安全XENIX系统的设计与开发成果。安全XENIX系统的目标是实现TCSEC标准B2-A1级的安全要求。系统的开发采用了改造/增强法。它实现了基于访问控制表结构的自主访问控制和基于BLP模型的强制访问控制，实现了以安全注意键（SAK，Secure Attention Key）为基础的可信路径机制，通过可信系统程序员、系统安全管理员、系统帐户管理员和系统安全审计员等特权用户实现超级用户的特权分割。

1988年，贝尔实验室的弗林克（C.W. Flink II）和维斯（J.D. Weiss）发表了System V/MLS系统的设计与开发成果。System V/MLS是以AT&T（美国电话电报公司）的UNIX System V为原型的多级安全操作系统，以TCSEC标准的安全等级B为设计目标。为保持系统兼容性，System V/MLS利用系统中的GID（Group Identification）域表示强制访问控制的安全等级标记，通过在内核中加入多级安全性（MLS，Multi-Level Security）模块实现对多级安全性的支持。多级安全性模块负责实现强制访问控制判定，在UNIX原始系统内核函数中的访问判定点插入调用多级安全性模块的命令，从而实现原始内核机制与多级安全性机制的连接。

1989年，加拿大多伦多大学的格雷尼尔（G.L. Grenier）、霍尔特（R. Holt）和方肯豪泽尔（M. Funkenhauser）发表了安全TUNIS系统的设计与开发成果。TUNIS是一个与UNIX兼容的操作系统，是UNIX内核的一个新的实现，用强类型的Turing Plus高级语言编写，具有较好的模块化结构，系统的设计目标是TCSEC标准的B3-A1等级。安全TUNIS系统实现了策略与机制分离的思想，它把内核分割成两个部分，一部分实现安全策略，另一部分实现安全机制。

1990年，TRW公司的沃尔德哈特（N.A. Waldhart）和维特（B.L. Di Vito）等发表了ASOS系统的设计与开发成果。ASOS是为美军开发的军用安全操作系统，由两类系统组成，其中一类是多级安全操作系统，设计目标是TCSEC标准的A1级，另一类是专用安全操作系统，设计目标是TCSEC标准的C2级。系统用Ada语言实现，同时支持用Ada语言编写的实时战术应用程序。ASOS的多级安全操作系统类型既实现了基于BLP模型的机密性强制访问控制支持，也实现了基于受限比巴模型的完

[2] Trusted Computer System Evaluation Criteria。

整性强制访问控制支持。在该系统中，既可以利用独立的机密性安全标记或完整性安全标记进行访问控制，也可以利用机密性与完整性相组合的安全标记进行访问控制。

在TCSEC标准颁布后不到十年的时间里，以TCSEC标准为核心的彩虹系列在安全操作系统领域产生了巨大影响。大量的研究与开发工作纷纷把TCSEC标准的安全评价等级设定为追求的目标，一系列B级以上的安全操作系统应运而生，最高安全评价等级（A1级）的安全操作系统的开发也付诸实现，这为操作系统安全性的研究留下了宝贵的财富。

安全策略繁花似锦

美丽的彩虹是迷人的，但终归会消失在蓝天之中。虽然彩虹系列标准意义重大，但其固有的局限性在岁月的冲刷下日益显露出来。TCSEC标准的特征是安全策略的单一性，明显存在BLP模型的痕迹。随着20世纪90年代初互联网影响的迅速扩大，分布式应用的迅速普及，在单一安全策略框架与需求繁多的现实世界之间的矛盾越来越大。

1992年，美国推出联邦标准草案，欲取代TCSEC标准，以消除TCSEC的局限性。1993年，美国国防部在TAFIM[3]计划中推出了称为DGSA[4]的新的安全体系结构，其显著特点之一是要求为多种安全策略提供支持。这给操作系统安全性的研究提出了新的挑战。

美国国防部要求在其所辖范围内强制实施TAFIM计划制定的体系结构框架。DGSA是该体系结构框架的组成部分。它的安全需求包括：为多种信息安全策略提供支持，采用开放式系统，提供充分的安全保护，以及实现共同的安全管理。

为了支持用户共享信息对象，同时充分保护共享环境中的信息对象，DGSA定义了信息域的概念。一个信息域由一些信息对象、一些用户和一个信息安全策略构成。支持多种安全策略的系统将拥有与安全策略数量一样多的信息域，这样的系统有能力处理数量非常多的信息域。

应用的需要促使人们设计出各式各样的安全策略。在一个分布式系统中实现基于应用的多种安全策略集成是计算机安全中的一项重要挑战。虽然传统的引用监控机的体系结构思想在确定的范围内的作用是成功的，但在支持基于多种应用的安全策略方面的能力有限。

为解决在分布式多策略环境中支持用户定义安全策略的问题，1992年，泰默尔（M.M. Theimer）等提出了访问控制程序（Access Control Programs， ACP）思想；1993年，哈蒂格（H. Hartig）等提出了看守员（custodian）思想。访问控制程序思想和看守员思想都采用了基于算法的安全策略解决方案。

在基于客户/服务器的分布式环境中，有时需要利用中间实体来代表客户端执行特定的任务，访问控制程序可用于定义面向应用的安全策略，对中间实体进行细粒度的访问授权。一个访问控制程序是一个程序，当客户端需要委派某个中间实体到服务器端执行任务时，客户端总是把一个访问控制程序和一个请求绑定在一起传送给服务器。一个访问控制程序描述客户端针对每一个请求给中间实体所授予的权限。访问控制程序附带有数字签名，以防止中间实体对它进行篡改。在服务器端，作为权限检查工作的一部分，服务器运行随请求一起收到的访问控制程序，如果访问控制程序允许，则把访问权限授给中间实体。

看守员思想要实现的是一个对象模型。对象模型支持用户在分布式系统中定义所需要的安全策略。在对象模型中，安全策略由算法和数据组成。首先，看守员可以成为安全策略的保护外壳，能够增强安全策略的抗篡改性。其次，看守员可以成为策略中立的引用监控机的附加组件，在保持引用监控

[3] The Technical Architectural Framework for Information Management，信息管理技术体系结构框架。

[4] DoD Goal Security Architecture，美国国防目标安全体系结构。

机的抗篡改性、完全仲裁性和可验证性的同时，该附加组件为引用监控机接纳用户定义的安全策略提供了条件。在支持多安全策略的分布式环境中，可以采用过程调用的通信方式来为不同策略间的协作建立良好的策略接口。

1997年，美国安全计算公司（SCC）和国家安全局（NSA）推出了DTOS[5]安全操作系统。这是一个支持多安全策略，基于微内核的安全操作系统，是在Mach操作系统的基础上开发实现的。

DTOS操作系统突出的设计目标之一就是支持安全策略的灵活性，即内核必须能够支持一系列的安全策略，当然包括国防部所要求的基于多级安全性的强制访问控制策略。同时，系统必须具有较强的策略变换能力，这要求策略变化所导致的系统变化可以局限在某个单一的系统组件之内。

DTOS安全体系结构的基础是一个由管理器和安全服务器构成的通用系统框架，通过安全判定与安全实施的分离支持安全策略的灵活性。安全判定由安全服务器履行。安全判定的结果由管理器实施。

DTOS原型系统的管理器是卡内基梅隆大学的Mach 3.0微内核的一个安全增强版本。它通过与安全服务器的协作，实施由安全服务器提供的安全判定，能够支持多种多样的安全策略，包括军用策略和民用策略，如多级安全性策略、基于标识的访问控制（Identity Based Access Control，IBAC）策略和类型裁决（Type Enforcement，TE）策略等。开发完成的安全服务器样例实现对多级安全性和类型裁决策略的支持。DTOS的Lite UNIX仿真环境提供安全UNIX的职能。

随需应变的梦想

从支持单一安全策略，到支持多种安全策略，操作系统安全性研究迈出了向实际应用环境靠近的可喜一步。然而，从支持多种安全策略，到支持安全策略的多样性，即提供安全策略的随需应变能力，还有相当一段距离。人们需要为之付出进一步努力。1999年诞生的Flask系统是这方面的努力的初步收获。

Flask是以Fluke操作系统为基础开发的安全操作系统原型。Fluke是一个基于微内核的操作系统，它提供一个基于递归虚拟机思想、利用权能系统的基本机制实现的体系结构。

Flask项目属于DTOS项目的延伸，实际上，Flask系统的安全体系结构是从DTOS原型系统的安全体系结构衍生而来的。虽然DTOS的安全体系结构是独立于特定安全策略的，但它仍无法支持动态安全策略。Flask的安全体系结构则克服了DTOS体系结构的这一不足，实现了动态安全策略，支持策略灵活性。

Flask安全体系结构包含客体管理器和安全服务器两类子系统，定义了两类子系统之间的相互作用，以及子系统组件应满足的要求：客体管理器子系统实施安全策略的判定结果，安全服务器子系统做出安全策略的判定。该体系结构的主要目标是不管安全策略判定是如何做出的，也不管它们是否随时间的推移而发生变化，都确保这些子系统总是有一个一致的安全策略判定视图，以便为安全策略的确定和定义提供灵活性支持。

Flask安全体系结构为客体管理器提供了三个基本要素。首先，它提供查询安全服务器中的访问判定、标记判定和多例化判定的接口。访问判定描述两个实体间（通常是是一个主体与一个客体间）的一个特定权限是否得到批准。标记判定描述分配给一个客体的安全属性。多例化判定描述一个特定的请求应该访问多例化资源中的哪一个成员。其次，系统提供一个访问向量缓存（Access Vector Cache， AVC）模块，该模块允许客体管理器缓存访问判定，以便减少性能开销。最后，它为客体管理器提供接收安全策略变

[5] Distributed Trusted Operating System。

化通知的能力。

Flask原型系统的安全服务器实现了由四个子策略组成的安全策略。这几个子策略是多级安全性策略、类型裁决策略、基于标识的访问控制策略和基于角色的访问控制（Role Based Access Control，RBAC）策略。安全服务器提供的访问判定必须满足每个子策略的要求。

由Flask的安全服务器封装的安全策略通过两种方式定义，一种方式是通过程序代码，另一种是通过策略数据库。能够由Flask原型的策略数据库语言表示的安全策略可以简单地通过修改策略数据库来实现。对于其它的安全策略，需要修改程序代码或完全重写安全服务器，以改变安全服务器的内部策略框架，从而获得支持。值得注意的是，不管是否需要修改安全服务器的程序代码，都无需修改客体管理器。

就原型系统所实现的几个安全策略而言，除了标记本身以外，多级安全性策略的策略逻辑大部分是通过安全服务器的程序代码来定义的，其它子策略的策略逻辑主要通过策略数据库语言来定义。Flask体系结构或Flask系统的实现并非只能支持以上四个安全策略，选择在安全服务器原型中实现这几个安全策略只是为了验证Flask体系结构的主要特性。

2001年，洛斯可可（P. Loscocco）等发布了SE-Linux的研究成果。这是一个以Linux操作系统为基础的基于Flask安全体系结构的安全操作系统。

Flask安全体系结构是基于微内核操作系统而设计的，而Linux却是非微内核的操作系统。Flask项目完成后，作为Flask系统的主要开发者，美国NSA启动了把Flask的安全体系结构嫁接到Linux操作系统中的项目，目的正是要检验该体系结构在非微内核操作系统中的生命力。美国网络伙伴公司（NAI）的实验室、SCC和MITRE公司等协助NSA完成嫁接工作。

在SE-Linux实现中，安全服务器和访问向量缓存是在Linux操作系统中增加的两个新组件。安全服务器是Linux内核中的一个子系统，内核中的其它子系统属于客体管理器。SE-Linux实现的安全服务器定义了一个由策略、基于角色的访问控制策略和多级安全性策略组合成的安全策略，其中类型裁决和基于角色的访问控制策略总是系统实现的安全策略的有机组成，而多级安全性策略是可选的策略。

SE-Linux项目的新贡献是提供了一个与安全服务器相配套的安全策略配置语言，用于对安全服务器中的安全策略的配置进行描述，以支持安全策略的灵活定义。构造安全策略时，系统生成一个由策略配置语言表示的配置文件。由配置语言表示的配置文件可编译成二进制形式，安全服务器在引导时读取二进制形式的配置数据，配置其中的安全策略。

不管是Flask系统，或是SE-Linux系统，还是其他别的相关系统，都只是操作系统安全策略多样性支持道路上的有益尝试，所取得的只是通往安全策略多样性目标的阶段性成果，安全策略随需应变的梦想还没有变成现实。

完整性度量之路

安全与可信是孪生姊妹。如果说访问控制能够在安全性中唱主角的话，那么，完整性度量必定是可信性中的主旋律。从20世纪60年代开始，访问控制社区已经热闹非凡。与此略有不同，在早期的岁月里，完整性度量家园似乎悄无声息。然而，于无声处有惊雷，完整性度量的世界同样精彩。

早在20世纪70年代末，尼巴尔第（G.H. Nibaldi）就对可信计算的概念进行了探讨，建立了可信计算基的思想。该思想为TCSEC标准的制订奠定了重要的基础。可信计算基思想的重要启示之一是通过硬件、固件和软件的合作来构筑系统平台的安全性和可信性。

与硬件结合是实现完整性度量的重要途径，以可信平台模块（Trusted Platform Module， TPM）为基础的可信计算技术是这方面的代表性成果之一。1999年，可信计算

平台联盟（TCPA[6]）的创立，为这样的技术注入了加速生长素。2003年，可信计算平台联盟演变为可信计算组织（TCG[7]），把可信计算向实用化推进了一大步。

可信计算要研究的根本问题是信任问题。信任问题的本质是实体行为的可预测性和可控制性，即实体的完整性。因此，如何度量和维护实体的完整性自然是可信计算的关键使命。可信计算平台只能算是其中的典型代表，但无论如何也算不上可信计算的始祖，因为在它之前还有很多“长老”。

1991年，卡内基梅隆大学的泰格（J.D. Tygar）等人提出了基于安全协处理器的Dyad系统模型。该模型的安全协处理器硬件为系统提供私密性和完整性支持。它通过数字签名检验操作系统和其他系统软件的完整性，为进入运行状态的操作系统提供完整性验证和加解密等服务，并支持操作系统验证其他组件的完整性、实现信息的加解密、建立与远程系统的加密连接。

1994年，美国可信信息系统公司的克拉克（P.C. Clark）和乔治华盛顿大学的霍夫曼（L.J. Hoffman）给出了基于智能卡的引导完整性令牌系统（Boot Integrity Token System，BITS）模型。该模型把系统的主引导程序存放在智能卡中，以保护主引导程序的完整性。存放在智能卡中的还有其他引导文件的哈希值以及用户口令和主机标识。系统启动时，首先验证用户使用智能卡的合法性和智能卡与主机的匹配关系，然后从智能卡中取出主引导程序，开始引导过程。主引导程序从主机中读取其他引导文件，完成引导过程。存放在主机中的文件的完整性借助智能卡中的哈希值进行验证。

1997年，宾夕法尼亚大学的阿宝（W.A. Arbaugh）等人提出了AEGIS安全引导体系结构模型。该模型修改了主机系统的BIOS[8]，并增加一个AEGIS ROM，以实现对可执行代码的完整性检查，如果完整性检查失败，则提供系统恢复支持。该模型把引导过程涉及的系统组件划分为6层，第0层是基础BIOS和AEGIS ROM，包含验证代码、公钥证书和系统恢复代码；第1层是其余BIOS；第2层是扩充的只读存储；第3层是操作系统的引导块；第4层是操作系统；第5层是应用软件。第0层中的软件是可信软件，用作完整性检查链的根。其余各层的可执行代码在执行之前，由低层进行完整性检查。完整性检查通过哈希值和数字签名实现。

2001年，IBM华生研究中心的代尔（J.G. Dyer）、达特茅斯学院的史密斯（S.W. Smith）和美国Cryptographic Appliances公司的魏因加特（S. Weingart）等人研制了IBM 4758安全协处理器。它在一个物理装置中封装了三组成分：硬件、固件和软件。硬件通过PCI[9]接口与主机系统连接。固件包含POST[10]和微引导程序。软件包含操作系统装载程序、操作系统和应用软件。4758是一个独立的缩微计算机系统，支持应用软件在其内部运行，安全应用软件可以部署在其内部，并通过安全协议与主机系统中的软件通讯，构成大的应用系统。4758内部把组件划分为若干层，借助数字签名实现内部系统的安全引导。

2004年，IBM公司的丸山（H. Maruyama）等人在他们设计实现的TPod体系结构中利用可信计算平台实现了系统的可信引导。在TPod实现的可信引导中，基础BIOS作为信任根首先执行并度量其余BIOS的完整性，其余BIOS执行并度量操作系统装载程序GRUB[11]的完整性，GRUB度量操作系统（这里包括SE-Linux内核和/etc/init脚本等）的完整性。

在完整性度量中，把操作系统等作为简

[6] Trusted Computing Platform Alliance。

[7] Trusted Computing Group。

[8] Basic Input/Output System，基本输入/输出系统。

[9] Peripheral Component Interconnect，外围设备互连。

[10] Power On Self Test，上电自检。

[11] GRand Unified Bootloader。

单组件对待属于粗粒度的问题处理方法。从操作系统层开始，直到应用软件层，如把度量对象的粒度细化到与实际应用系统比较一致的程度，则需要解决更多的问题。细粒度的完整性度量是当前的热点研究问题。

2004年，IBM华生研究中心的塞勒（R. Sailer）等人提出了完整性度量的IMA（Integrity Measurement Architecture）体系结构，设计实现了Linux系统的组件细化完整性度量原型系统。IMA原型以可信计算平台为可信硬件，给出了系统BIOS、操作系统装载程序GRUB、Linux内核、Linux模块、应用软件等各层组件执行前的完整性度量方法，重点给出了Linux内核、Linux可装载内核模块、动态可装载库、结构化数据和可执行用户程序等的度量方法。

2005年，卡内基梅隆大学的史（E. Shi）和IBM华生研究中心的范·多恩（L. Van Doorn）等人提出了为分布式系统建立可信环境的BIND（Binding Instructions aNd Data）框架。BIND把代码的完整性证明细化为关键代码段的完整性证明，并为关键代码段产生的每一组数据生成一个认证器。认证器附着到相应数据上，从而实现关键代码段的完整性证明与其所产生的输出数据的绑定。因此，BIND可以通过关键代码段及其输入数据的完整性证明来达到系统完整性证明的目的。

2006年，宾夕法尼亚州立大学的耶格（T. Jaeger）、IBM华生研究中心的塞勒和加州大学伯克利分校的山克（U. Shankar）提出了基于信息流的 （下转30页）

石文昌

中国计算机学会高级会员，信息保密、系统软件、开放系统等专委会委员。博士，中国人民大学教授。主要研究方向为信息安全、可信计算、系统软件与虚拟机技术。

李伟楠

中国人民大学信息学院硕士研究生。主要研究方向为系统与信息安全。

李翰超

中国人民大学信息学院硕士研究生。主要研究方向为系统与信息安全。

贾春福

博士，南开大学教授。主要研究方向为信息安全与可信计算，恶意软件的发现与防治技术。

参考文献

[1] Virgil D. Gligor. 20 Years of Operating Systems Security. Proceedings of the 1999 IEEE Symposium on Security and Privacy, Oakland, California, May 1999

[2] 石文昌, 孙玉芳. 安全操作系统研究的发展(上). 计算机科学, 29(6), 2002

[3] 石文昌, 孙玉芳. 安全操作系统研究的发展(下). 计算机科学, 29(7), 2002

[4] 石文昌, 单智勇, 梁彬, 梁朝晖, 董铭. 细粒度信任链研究方法. 计算机科学，35(9), 2008

[5] Andrew S. Tanenbaum, Jorrit N. Herder, Herbert Bos. Can We Make Operating Systems Reliable and Secure. IEEE Computer, 39(5), 2006

[6] 石文昌, 孙玉芳, 梁洪亮, 张相锋, 赵庆松, 单智勇. 安全Linux内核安全功能的设计与实现. 计算机研究与发展, 38(10), 2001:1255~1261

（上接39页）

PRIMA（Policy-Reduced Integrity Measurement Architecture）完整性度量体系结构，并研究了以SE-Linux为基础的原型系统。PRIMA项目的研究工作在IMA研究成果的基础上引入CW-Lite信息流模型来处理组件依赖关系，为基于信息流的系统完整性动态度量进行了卓有成效的尝试。

在开放式、分布式的全球计算环境中，建立信任关系是现实应用的迫切需求。可信计算受到了工业界和学术界的广泛关注，很多成果已经进入我们的视野。然而，在完整性度量的道路上，依然“路漫漫其修远兮”，尚需“上下而求索”。

结语

纵观操作系统安全性研究的演进过程，从最初操作系统安全相关基础理论的形成到之后安全标准的推行，从系统对单一安全策略的支持，到后来多种策略并存以及对策略多样性的灵活支持，无论是早期主要面向军事需求的机密性支持，还是后来面向商业需求及众多领域的完整性支持，操作系统安全性的研究经历了四十多年的锤炼，积累了很多宝贵的研究成果。在信息化社会中，攻与防的对抗将永不停息，因此操作系统安全性的研究需要不断应对新的挑战，为信息安全生态系统的建立和维护提供强有力的基础支撑。