细粒度信任链研究方法*)

石文昌[1,2,3] 单智勇[1,2] 梁 彬[1,2] 梁朝晖[1,2] 董 铭[1,2]
(中国人民大学 DEKE 重点实验室信息安全组 北京 100872)[1]
(中国人民大学信息学院 北京 100872)[2] (中国科学院研究生院 北京 100049)[3]

摘 要 分析信任链相关研究的当前发展水平,提出细粒度信任链和细粒度系统软件信任链的思想,阐明只有细粒度信任链才能描述现实应用的真实情况。根据问题空间的复杂性,提出细粒度信任链建模的问题分解方法。该方法通过逐步拓展的策略,首先建立细粒度系统软件信任链模型,然后在此基础上建立完全的细粒度信任链模型。

关键词 信任链,系统软件,可信计算,完整性,度量

Approach for Research on Fine-grained Chain of Trust

SHI Wen-chang[1,2,3] SHAN Zhi-yong[1,2] LIANG Bin[1,2] LIANG Zhao-hui[1,2] DONG Ming[1,2]
(Information Security Group, Key Lab of DEKE, Renmin University of China, Beijing 100872, China)[1]
(School of Information, Renmin University of China, Beijing 100872, China)[2]
(Graduate University, Chinese Academy of Sciences, Beijing 100049, China)[3]

Abstract The state of the art of relevant research on chain of trust is figured out. The concepts of fine-grained chain of trust and fine-grained chain of trust of system software are put forward. It is argued that only fine-grained chain of trust can describe the genuine scenario of real world application. According to complexity of the problem space in question, a two-step approach is proposed for modeling fine-grained chain of trust, in which the first step is to model the fine-grained chain of trust of system software, and the second step is to model the complete fine-grained chain of Trust.

Keywords Chain of trust, System software, Trusted computing, Integrity, Measurement

1 引言

随着电子商务、电子政务、跨域的资源共享等新的应用模式不断涌现,人们日益认识到在 Internet 环境下软件系统的可信性已经成为一个亟待解决的问题。Internet 环境下的软件可信问题源于 Internet 环境的资源行为不可控性和不确定性,而这与 Internet 资源本身的开放性、动态性与资源的成长性、自治性、多样性等自然特性有着密不可分的关系。同时,由于应用规模的不断扩展、所涉及资源的种类和范围的不断扩大、应用复杂度的提高以及计算模式的革新,都对 Internet 环境下软件系统的可信保障提出了更高的要求[1]。

可信与安全既密不可分又相互区别。传统安全技术的核心思想是保护信息资源的机密性、完整性和可用性,这种思想主要从信息资源拥有者的角度考虑问题,保护信息资源免遭破坏。在 Internet 环境中,还必须从信息资源使用者的角度考虑问题,保护使用者免遭有害信息资源的毒害,传统安全技术难以提供这方面的支持[2]。

传统安全技术在 Internet 环境中表现出严重的局限性,而信任(Trust)体系技术可以在其中发挥重要的作用[3]。不管是信息资源的拥有者,还是信息资源的使用者,都是计算环境中的实体(Entity)。提供服务的实体需要得到安全保护,接受服务的实体也需要得到安全保护。Internet 环境中有大量的各种实体存在,很多实体相互之间从来没有打过交道,建立信任体系有助于在 Internet 环境中降低实体面临的风险并保障他们进行正常的工作[4,5]。

建立实体间的信任关系,在电子商务、电子政务等现实应用中有非常重要的意义。以网上电子银行为例,客户实体与银行实体协作进行金融交易,银行实体需要确认客户实体的真实身份,而客户实体也需要确认银行实体的真实服务。只有建立客户实体与银行实体之间的信任关系,才能保障交易的安全进行,否则其风险是难以估量的。

虽然,迄今为止,关于信任的概念一直没有一个统一的定义,但是对信任体系的研究,国际上的研究者们开展了长期艰苦的工作[2-8]。尤其是最近几年来,随着可信计算组织(TCG: Trusted Computing Group)的成立,并推出以可信平台模块(TPM: Trusted Platform Module)[9]为核心的可信计算[10]规范,信任体系的研究在国际上进一步引起了研究者们的高度重视。

可见,信任体系的研究,尤其是 Internet 环境中的信任体系的研究,具有非常重要的现实意义,在电子商务、电子政务等现实应用中具有广阔的应用前景。本文研究的信任链模型属于信任体系模型的范畴。

2 信任链基本思想

本文研究的问题空间的现实应用场景是 Internet 环境中

*)国家 863 项目(2007AA01Z414),国家自然科学基金项目(60373054,60703102,60703103)资助。石文昌 博士,教授,博士生导师,CCF 高级会员,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;单智勇 博士,讲师,主要研究方向为信息安全与系统软件;梁 彬 博士,讲师,主要研究方向为信息安全与系统软件;梁朝晖 博士,讲师,主要研究方向为宽带无线网络与信息安全;董 铭 博士,副教授,主要研究方向为网络与通信、信息安全与管理信息系统。

的主流计算机系统之间协作完成应用任务。Internet 环境中的计算机系统 A 从启动到进入稳定的运行状态乃至运行过程中，需建立和维护一条信任链，以实现对系统完整性的度量；Internet 环境中的远程计算机系统 B 在需要时可以要求 A 提供系统完整性证明；此时，A 可以在 Internet 环境中向 B 证明“A 的完整性是有保障的”；B 在验证了 A 的完整性之后，确定是否进一步与 A 协作完成应用任务。

TCG 制定的可信计算规范通过硬件实现的信任根(Root of Trust)来支持信任链的建立工作。度量专用核心信任根(CRTM:Core Root of Trust for Measurement)启动信任链的建立过程，信任链借助信任边界的向外扩展逐步建立起来[10]。TCG 规范的信任链可以简单表示如下：

[CRTM]→[BIOS]→[MBR]→[OS_Loader]→[OS]→[Application]

这个表示中的“→”表达了系统组件间的完整性度量关系和运行控制权传递关系，“→”左侧的组件度量右侧的组件，然后左侧组件把运行控制权传递给右侧组件，右侧组件开始运行。

计算机主机平台上电时，CRTM 被启动运行，CRTM 首先度量 BIOS 的完整性，保存度量结果，然后再把控制权传给 BIOS；BIOS 执行初始化操作，度量主引导记录 MBR 的完整性，保存度量结果，装入 MBR，再把控制权传给 MBR；MBR 度量操作系统装载器 OS_Loader 的完整性，保存度量结果，装入 OS_Loader，再把控制权传给 OS_Loader；OS_Loader 度量操作系统 OS 的完整性，保存度量结果，装入 OS，再把控制权传给 OS；OS 度量应用软件 Application 的完整性，保存度量结果，装入 Application，再把控制权传给 Application；最后，应用软件 Application 进入运行。这个过程中的完整性度量结果可以由硬件 TPM 保存。

显然，上面表示中的“→”也表达了信任传递的方向，“→”左侧的组件度量右侧的组件的完整性，如果度量结果表明右侧组件是完整的，则信任边界从左侧向右侧扩展，形成包含右侧组件的新边界，信任从左侧组件传递到右侧组件。从整体上看，信任传递从 CRTM 开始，沿着“→”的方向，最后传递到 Application。这样，建立一条从 CRTM 经 BIOS、MBR、OS_Loader、OS 到 Application 的信任链。本文把这样的信任链传递方向称为原始维度方向。

3 国际发展现状与趋势

由于现实应用的迫切需要，在信任链相关的研究领域，国际上的研究活动非常活跃。然而，当前研究成果的水平还难以建立能够描述现实应用真实情况的信任链理论模型。

TCG 的可信计算技术的核心思想是通过硬件实现的信任根支持软件可信性的实现以及软件可信性对外证明的实现。TCG 是工业界的一个联盟，它的目标是建立可信计算的技术规范。TCG 的技术规范并没有相应的理论体系给以支持，学术界目前也还没有相应的理论体系为它提供支持。

结合 Internet 环境，信任体系研究的重要目标之一是建立 Internet 环境中需要协作的实体之间的信任关系模型，在实际运行于 Internet 环境的应用系统中实现相应的信任关系模型，并证明所建立的信任关系系统的正确性。缺乏理论模型的支持，系统的正确性将难以得到有效的证明。而信任的传递和信任链的建立乃信任体系的基础，因而信任链理论模型的建立是实现这样的信任体系研究目标的重要一环。

信任问题的重要思想是实体行为的可预测性和可控制性[10]。如果实体的完整性(Integrity)遭到破坏，实体行为的可预测性和可控制性就不可能得到保证。因此，实体的完整性是信任体系中的关键问题。在信任体系研究中，实体可定义为 Internet 环境中某可信平台上的软件的实例(Installation)，它的构成包含其赖以支撑的下层软件和硬件要素[11]。实体的信任问题与软件的可信性问题密切相关。

为确定实体的完整性，需要从硬件、固件、引导模块、系统软件、应用软件等多个层面对实体进行考察。从硬件层开始，朝着应用软件层的目标，信任边界逐层扩展，信任链逐步建立，实体的完整性逐渐得到度量。

目前，从硬件上电自检(POST：Power On Self Test)、基本输入输出系统(BIOS：Basic Input Output System)、引导模块(Boot)到作为单一组件的操作系统内核层的实体完整性度量和系统可信引导，已有比较成熟的技术成果[12]，比如 Carnegie Mellon 大学 J. D. Tygar 等人的 Dyad 系统(1991 年)[13]，Trusted Information Systems 公司 P. C. Clark 等人的 BITS 系统(1994 年)[14]，Pennsylvania 大学 W. A. Arbaugh 等人的 AEGIS 系统(1997 年)[15]，IBM T. J. Watson 研究中心 J. G. Dyer 等人的 4758 系统(2001 年)[16]等。把 TPM 应用到系统的可信引导之中，也有了较好的结果，比如 IBM 公司 H. Maruyama 等人的 TPod 系统(2004 年)[17]。这些研究成果重点解决实体组件运行前的完整性度量问题，这类度量属于静态度量。

把操作系统等作为单一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，把组件的粒度细化到与现实应用比较一致的程度，特别地对组件的代码与相应的数据进行有效的细化度量，还存在很多问题没有解决。为了解决这些问题，国际上的研究者开展了一系列经验型研究(指通过原型系统的设计与开发实验来验证设想的研究)的工作。比如，IBM T. J. Watson 研究中心 R. Sailer 等人的 IMA 系统(2004 年)[18]，Carnegie Mellon 大学 E. Shi 等人的 BIND 系统(2005 年)[19]，Pennsylvania 大学 T. Jaeger 等人的 PRIMA 系统(2006 年)[20]等，都是极具代表性的工作，也体现了最新的研究水平。PRIMA 研究在 IMA 研究成果的基础上引入 CW-Lite 信息流模型[21]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。动态度量考虑实体组件运行时的完整性问题。

在信任体系理论模型研究方面，国际上的现有成果多数是在应用软件层面上考虑解决不同实体之间信任关系的相关问题[3,22,6]。与本文的信任链模型研究较为接近的最新理论研究成果是 Dartmouth 学院 S. W. Smith 的 OA(Outbound Authentication)模型(2004 年)[11]。该理论模型通过 Maurer 式的演算技术，建立信任集和组件依赖函数等关键思想，确定实体组件信任传递的合理性(Soundness)和完备性(Completeness)。但 Smith 的 OA 理论模型有明显的局限性，其一，它针对的是 IBM 4758 安全协处理器的特定封闭环境，信任域(Trust Domain)的限定条件比较苛刻，不适宜描述主流计算机系统环境；其二，系统环境建立在安全引导的基础之上，仅凭借密钥和签名表示实体组件的可信性，不适合本文研究的情形；其三，只适合描述粗粒度实体组件。

显然，国际上在粗粒度实体组件完整性静态度量方面已有比较成熟的技术成果，在细粒度实体组件完整性静态度量和实体组件完整性动态度量方面只有初步的技术成果，在理

论成果方面，主要还局限于应用层面的信任问题或特定封闭环境的粗粒度实体组件的信任传递问题。国际相关研究解决信任链问题的总体趋势是：从最基础的硬件信任根向面向用户的应用层发展、从粗粒度实体组件向细粒度实体组件发展、从静态完整性度量向动态完整性度量发展、从经验型研究向理论体系构造发展。

4 国内研究状况

国内在可信计算相关领域的工作也非常活跃。2000 年，武汉瑞达公司和武汉大学合作，开始研制安全计算机，2004 年系统通过鉴定。2005 年，联想集团的 TPM 芯片和可信计算机研制成功，兆日公司的 TPM 芯片也研制成功[23]。在系统完整性静态度量[24]、TPM 安全性分析[25]等方面也逐步取得了一些成果。

总体上说，国内在硬件 TPM 芯片和相关系统的设计和制造方面技术比较先进，在可信计算相关理论的研究方面的工作比较滞后。目前，在能够描述现实应用真实情况的信任链理论模型方面还没有相应的研究成果。

我们对基于 TPM 的操作系统扩展可信路径机制[26]、基于 TPM 的可信文件系统[27,28]等进行了经验型研究并取得了相应的成果，这些工作及相应的成果为信任链理论模型的研究奠定了一定的基础。

5 信任链建模的可行途径

信任链理论模型研究是信任体系理论模型研究中的重要任务。

5.1 粒度细化的需求

TCG 所描述的信任链是粗粒度的，它把 OS 和 Application 等都只表示为一个组件，但事实上，仅就 OS 而言，它是一个复杂的系统，可能包含内核、可装载内核模块、动态库、实用例程、配制文件等多种成分[18]，它们都有可能影响系统的完整性。借助粗粒度的信任链描述显然无法反映实际系统的真实情况，因此，为了能够清晰地表达实际系统构成中的各种完整性相关成分，准确地描述实际系统中信任的传递和信任链的建立的真实情况，我们必须找到一种能够刻画实际系统构成成分的细粒度的信任链描述方法，所以我们需要研究细粒度的信任链理论模型。

5.2 问题分解方法

对一个灵活变化的系统环境进行完全的完整性验证是一项艰巨的工作。仍以 OS 的完整性度量为例，除了已提到的 OS 各种组成成分自身的完整性以外，这些成分在运行过程中接收的各种输入数据（包括结构化数据和非结构化数据），也有可能影响这些成分的完整性[19,20]，进而影响 OS 的完整性。Application 的情况则更复杂，除了存在与 OS 类似的情况外，应用软件的开发者还可以自行定义可执行脚本的格式和语义，开发自定义的脚本解析软件，局外人很难准确掌握此类成分的完整性语义[18]，因而很难为它们建立动态完整性度量方法。

考虑到细粒度应用软件完整性度量的现实复杂性和难度，我们认为一次性囊括所有问题将非常困难，不利于问题的解决，我们提出通过逐步扩展的问题分解策略解决细粒度信任链的建模问题，把问题的求解分解为两步进行，第一步建立细粒度系统软件信任链模型，第二步在第一步的基础上建立完全的细粒度信任链模型。

在第一步的工作中，暂时不必考虑细粒度应用软件的度量问题，只需集中考虑信任边界扩展到 Application 之前的信任链，即如下信任链：

$$[CRTM]\rightarrow[BIOS]\rightarrow[MBR]\rightarrow[OS\_Loader]\rightarrow^{@}[OS^{FG}]$$

这就是本文所称的系统软件信任链。需要强调的是，我们把 OS 改成了 OS^{FG}，表示细粒度的 OS。同时，把 OS_Loader 与 OS 之间的“→”改成了 OS_Loader 与 OS^{FG} 之间的“$\rightarrow^{@}$”，这表示从 OS_Loader 到 OS^{FG}，不是由简单的一步扩展所能完成的。这样表达的就是本文所称的细粒度系统软件信任链。

要准确地表达 OS^{FG}，必须对 OS 组件进行分解。这里所说的组件分解，并不是要人为地对 OS 组件进行切分，而是要把实际系统中构成 OS 组件的多种成分准确地表达出来。比如，一个实际运行的 Linux 类 OS 组件可能包含内核、可装载内核模块、动态库等多种成分，OS 组件分解的含义就是通过对这些成分的描述来表达相应的 OS 组件。

TCG 描述的信任链是一维的，它只能描述原始维度方向上的组件关系。但是，系统粒度细分之后，分解出的组件成分之间并非都能表示成原始维度方向上的关系。因此，为表示细粒度信任链，我们需要引入一个与原始维度不同的新的维度，本文把这个新引入的维度称为新增维度。

也就是说，要完成从 OS_Loader 到 OS^{FG} 的信任边界扩展，需要对信任链进行深度和维度的扩展。深度扩展是在原始维度方向上扩展 OS 组件的层次，如扩展出内核层和非内核层；维度扩展是在新增维度方向上对 OS 进行组件分解，如非内核层分解出可装载内核模块、动态库、实用例程等多种成分。

因此，本文提出的细粒度系统软件信任链研究，把一维信任链拓展为二维信任链，拓展的难度是实现 OS 深度和维度的扩展和完整性度量的扩展。

通过完成问题求解中的第一步研究任务，我们可以建立起细粒度系统软件信任链的理论模型，从而完成建立细粒度信任链理论模型的大部分工作。在此基础上，第二步的任务是研究建立以下信任链：

$$[CRTM]\rightarrow[BIOS]\rightarrow[MBR]\rightarrow[OS\_Loader]\rightarrow^{@}[OS^{FG}]\rightarrow^{@}[Application^{FG}]$$

这是覆盖细粒度应用软件的完全的细粒度信任链，为建立该信任链的理论模型，需重点解决应用软件的细粒度描述和完整性度量问题。细粒度系统软件信任链理论模型的建立已为完全的细粒度信任链理论模型的建立打下了重要的基础。

5.3 建模的理论手段

系统的完整性度量是信任链建立过程中的关键工作。历史上，通过系统的信息流对系统的完整性进行评估是一项典型的措施[29,30,20]。Biba 模型[29]和 Clark-Wilson 模型[30]是经典的信息流模型。无干扰模型[31]则是另一种类型的基于信息流思想的模型，其理论体系与信任链的建立和扩展的抽象表达和验证要求非常吻合。

原始的无干扰模型是由 J. A. Goguen 和 J. Meseguer 提出的[32]，随后得到了 J. Haigh 和 W. Young 的发展[33,34]。J. Rushby 在这些基础上发展得到的无干扰模型[31]更完善和更具有普遍意义，我们把它称为 Rushby 无干扰模型，该模型最符合对信任链的建立和扩展进行抽象和验证的需要。

通常，完整性是一个二值属性，对一个实体进行完整性度

量，结果只有两个：完整或不完整。同时，完整性也是一个相对属性，就系统B而言，系统A的完整性有时依赖于系统B对完整性的要求。系统B上的应用b要与系统A协作时，如果A上所有b依赖的组件都是完整的，则B可认为A就是完整的，尽管A上可能有其它不完整的组件存在。要处理这样的情形，可用Rushby无干扰模型来区分那些处在无干扰关系中的组件。

Rushby无干扰模型理论能为细粒度信任链理论模型研究提供强有力的理论支持，可以作为建立细粒度信任链理论模型的有效的理论手段。

结束语 本文考察了信任体系研究的现实意义，分析了信任链相关研究的当前发展水平，提出了细粒度信任链和细粒度系统软件信任链的思想，阐明了只有细粒度信任链才能描述现实应用的真实情况。根据问题空间的复杂性，提出了细粒度信任链建模的问题分解方法，该方法通过逐步拓展的策略，首先建立细粒度系统软件信任链模型，然后在此基础上建立完全的细粒度信任链模型。本文的工作为能够反映现实应用真实情况的信任链的理论建模确定了可行途径和理论手段，为信任链理论模型的研究建立了重要的基础，对实际应用系统信任链的建立和软件可信性的研究具有重要的现实意义。

参 考 文 献

[1] 王怀民，唐扬斌，尹刚，等. 互联网软件的可信机理. 中国科学，E辑，信息科学，2006，36(10)：1156-1169

[2] Josang A. Prospectives for Online Trust Management. IEEE Transactions on Knowledge and Data Engineering, 2007. http://sky. fit. qut. edu. au/~josang/papers/Jos2007-oltrustman. pdf

[3] Carbonel M, Nielsen M, Sassone V. A Formal Model for Trust in Dynamic Networks // Proceedings of International Conference on Software Engineering and Formal Methods (SEFM'03). IEEE Computer Society Press, 2003：54-61

[4] Seigneur J-M. Trust, Security and Privacy in Global Computing. Ph. D. Thesis. Ireland: University of Dublin, Trinity College, 2005

[5] Li H, Singhal M. Trust Management in Distributed System. IEEE Computer, 2007, 40(2)：45-53

[6] Grandison T, Sloman M. A Survey of Trust in Internet Applications. IEEE Communications Surveys, Fourth Quarter, 2000：2-16

[7] McKnight D H, Chervany N L. The Meanings of Trust. University of Minnesota, 1996. http://misrc. umn. edu/wpaper/WorkingPapers/9604. pdf

[8] Blaze M, Feigenbaum J, Lacy J. Decentralized Trust Management // Proceedings of the 1996 IEEE Symposium on Security and Privacy (SP'96). 1996：164-173

[9] Trusted Computing Group. TPM Main, Part 1 Design Principles, Specification Version 1. 2, Level 2 Revision 103. July 2007. https://www. trustedcomputinggroup. org/specs/TPM/mainP1DPrev103. zip

[10] Trusted Computing Group. TCG Specification Architecture Overview, Specification Revision 1. 4. 2nd Aug. 2007. https://www. trustedcomputinggroup. org/groups/TCG_1_4_Architecture_Overview. pdf

[11] Smith S W. Outbound Authentication for Programmable Secure Coprocessors. International Journal of Information Security, 2004, 3(1)：28-41

[12] Smith S. Magic Boxes and Boots: Security in Hardware. IEEE Computer, 2004, 37(10)：106-109

[13] Tygar J D, Yee B. Dyad: A System for Using Physically Secure Coprocessors. Technical Report. CMU-CS-91-140R. Carnegie Mellon University, May 1991

[14] Clark P C, Hoffman L J. BITS: A Smartcard Protected Operating System. Communications of the ACM, 1994, 37(11)：66-70, 94

[15] Arbaugh W A, Farber D J, Smith J M. A Secure and Reliable Bootstrap Architecture // Proceedings of the 1997 IEEE Symposium on Security and Privacy (S&P'97). 1997：65-71

[16] Dyer J G, Lindemann M, Perez R, et al. Building the IBM 4758 Secure Coprocessor, 2001, 34(10)：57-66

[17] Maruyama H, Seliger F, Nagaratnam N, et al. Trusted Platform on demand. Technical Report. RT0564. IBM, Feb. 2004

[18] Sailer R, Zhang X, Jaeger T, et al. Design and Implementation of a TCG-based Integrity Measurement Architecture // Proceedings of the 13th USENIX Security Symposium. San Diego, CA, USA, Aug. 2004：223-238

[19] Shi E, Perrig A, Doorn L V. BIND: A Fine-Grained Attestation Service for Secure Distributed Systems // Proceedings of the 2005 IEEE Symposium on Security and Privacy (S&P'05). 2005：154-168

[20] Jaeger T, Sailer R, Shankar U. PRIMA: Policy-reduced Integrity Measurement Architecture // Proceedings of the Eleventh ACM symposium on Access Control Models and Technologies. Lake Tahoe, California, USA, New York, NY, USA: ACM Press, 2006：19-28

[21] Shankar U, Jaeger T, Sailer R. Toward Automated Information-flow Integrity Verification for Security-critical Applications // 13th Annual Network and Distributed System Security Symposium. Internet Society, San Diego, California, Feb. 2006

[22] Weeks S. Understanding Trust Management Systems // Proceedings of the 2001 IEEE Symposium on Security and Privacy (S&P'01). 2001：94-105

[23] 张焕国，罗捷，金刚，等. 可信计算研究进展. 武汉大学学报(理学版)，2006，52(5)：513-518

[24] 徐娜. 基于可信赖计算平台的可信执行环境研究与实现. 硕士学位论文. 中国科学院研究生院(计算所)，2006

[25] 陈军. 可信平台模块安全性分析与应用. 博士学位论文(导师：侯紫峰). 中国科学院研究生院(计算所)，2006

[26] Shi Wenchang. Implementing Operating System Support for Extended Trusted Path in TPM-capable Environments. Wuhan University Journal of Natural Sciences, 2006, 11(6)：1493-1497

[27] 张伟伟，石文昌. 一个可信文件系统 Trusted FS 的设计与实现 //第五届中国信息和通信安全学术会议论文集(CCICS'2007). 长沙：科学出版社，2007：342-347

[28] 张伟伟. 面向可信计算平台的可信文件系统的研究与实现. 硕士学位论文. 中国科学院研究生院(软件所)，2007

[29] Biba K J. Integrity Consideration for Secure Computer Systems. Technical Report. ESD-TR-76-372. Electronic Systems Division, Air Force Systems Command, Hanscom Field, Bedford, MA, USA, 1977

[30] Clark D D, Wilson D R. A Comparison of Commercial and Military Computer Policies // Proc. 1987 Symposium on Security and Privacy. Oakland, CA, IEEE Computer Society, Apr. 1987：184-194

[31] Rushby J. Noninterference, Transitivity, and Channel-control Security Policies. Technical Report, CSL-92-02. Computer Science Laboratory, SRI international, Dec. 1992

[32] Goguen J A, Messeguer J. Security Policies and Security Models // Proc. 1982 Symposium on Security and Privacy. Oakland, CA, IEEE Computer Society, Apr. 1982：11-20

[33] Haigh J, Young W. Extending the Non-interference Model of MLS for SAT // Proc. 1986 Symposium on Security and Privacy. Oakland, CA, IEEE Computer Society, Apr. 1986：232-239

[34] Haigh J T, Young W D. Extending the Noninterference Version of MLS for SAT. IEEE Transactions on Software Engineering, 1987, SE-13(2)：141-150