操作系统信任基建模的方法学研究

石文昌
(中国人民大学数据工程与知识工程教育部重点实验室 北京 100872)
(中国人民大学信息学院 北京 100872)

摘 要 倡导操作系统在确立应用系统的可信性中具有不可或缺的作用之理念,以Web应用为聚焦点,诠释操作系统信任基(TBOS)的思想。以如何确保TBOS的可信性为主线,讨论TBOS建模的研究方法。提出由信任监控核心引擎、内核信任监控器和核外信任监控器三大部分组成的TBOS体系结构,提出发挥硬件效能与缩小软件尺寸相结合的研究方针,阐述TBOS建模中的关键问题和关键技术,从模型构造方法、信任监控方法、域间协同方法、隔离保护方法、硬件特性抽象方法和软件尺寸缩减方法等方面建立TBOS建模的方法学基础。

关键词 操作系统,信任基,建模,方法学,模型,硬件
中图法分类号 TP309 **文献标识码** A

On Methodology of Modeling the Trust Base in Operating Systems

SHI Wen-chang
(Key Lab of Data Engineering and Knowledge Engineering of Ministry of Education, Renmin University of China, Beijing 100872, China)
(School of Information, Renmin University of China, Beijing 100872, China)

Abstract Advocates the philosophy that operating systems are of indispensable significance to make applications trusted. Elaborates the concept of Trust Base in Operating System (TBOS) with a focus on Web applications. Discusses research methods for modeling the TBOS with a longing for ensuring that the TBOS is trusted. Proposes the TBOS architecture that consists of three main parts, which are the Trust Monitoring Core Engine, the In-Kernel Trust Monitor and the Out-of-Kernel Trust Monitor. Proposes a research guideline that is to exploit the potential of hardware and reduce the size of software. States key issues and key techniques in modeling the TBOS. Establishes the methodology foundation of modeling the TBOS from the aspects of model construction method, trust monitoring method, inter-domain collaboration method, protection-by-isolation method, hardware features abstraction method and software size minimization method.

Keywords Operating system, Trust base, Modeling, Methodology, Model, Hardware

信任是现实社会和信息空间中的热点话题[1-3]。与现实社会中的情况类似,在信息空间中,技术越进步,人们的信任意识就越强。计算模式的每一次变革,往往会唤起人们对信息空间中的应用系统可信性更强烈的渴望[4],比如,近几年的云计算模式的兴起,就再一次使人们对计算机系统的可信性提出了新的要求。

可以阐明[5,6]:在计算机系统中,操作系统对于确立应用系统的可信性具有不可或缺的作用。为了解决操作系统对应用系统可信性的有效支撑问题,我们提出了操作系统信任基(TBOS, Trust Base in Operating System)的思想[7]。

本文进一步深化TBOS的思想,选择当今乃至将来具有广泛影响的Web应用作为我们聚焦的应用类型,结合现代CPU等硬件的新特性,研究构造TBOS模型的方法,为实现可信的TBOS软件,从而为支持确立应用系统的可信性,建立有效的方法学基础。

1 操作系统信任基研究思想概貌

1.1 操作系统信任基的含意及其聚焦对象

操作系统信任基研究的基本想法就是给操作系统装备一个监控器,用来监测操作系统是否可信,以及应用系统是否可信。我们把此监控器称为操作系统信任基(TBOS),并以定义1进一步明确其含意。

定义1(操作系统信任基) 操作系统中为系统(S)的可信性提供支持的相关机制的整体统称为操作系统信任基(TBOS),其中,S既包括操作系统自己也包括操作系统之上的应用系统,可信性支持主要指TBOS对S中的安全敏感部分(P)进行监控,防止P遭受非法篡改或不按预期工作。

作为操作系统层次的机制,TBOS具有相当大的应用独立性,但是,为了提高可信性支持的有效性,希望TBOS同时应该能够为应用提供一定的个性化支持。

本文受国家863计划课题(2007AA01Z414),国家自然科学基金项目(60873213,60703103),北京市自然科学基金项目(4082018),上海市智能信息处理重点实验室开放课题(IIPL-09-006)资助。
石文昌(1964—),男,博士,教授,博士生导师,主要研究方向为信息安全、可信计算、系统软件,E-mail:wenchangshi@pku.org.cn。

当前，Web 应用是一种主流的应用类型，我们相信，在今后相当长的时间里，也将依然如此，因而，它将产生广泛而深远的影响，无论在电子政务、电子商务还是寻常百姓的日常消遣中，都不可或缺。故此，Web 应用系统的可信性意义重大。本文选择 Web 应用作为聚焦的应用类型，并将其作为 TBOS 的主要服务对象。

1.2 研究思想的基本出发点

TBOS 的最终目的是通过监测确保应用系统的可信性，为了使得 TBOS 得出的监测结果能够令人信服，TBOS 本身必须可信。我们考虑采取两个措施来把 TBOS 做得可信，一是尽可能地把它做到最小，二是发挥硬件的效能。

根据软件工程的研究结果，软件的尺寸（或称大小，即代码量）和复杂性与软件中存在的 Bug 的数量有很大的关系，软件尺寸越大，复杂性越高，软件中存在的 Bug 就越多。软件中的 Bug 越多，软件的缺陷就越多，软件中存在安全漏洞的可能性就越高，从而软件的可信性就会相应地降低。TBOS 的最小化，就是尽可能地缩小 TBOS 的尺寸，降低其复杂性，这有利于提高 TBOS 的可信程度，因此研究 TBOS 的最小化具有重要的现实意义。

在计算机技术的发展历程中，软件滞后于硬件，是一种普遍的现象，甚至形成了妨碍应用充分利用硬件能力的软件墙。硬件中的某些能力，具有软件能力所不及的优势，把这样的硬件能力挖掘出来，可以为构造 TBOS 带来靠纯软件手段难以获得的效果。

我们不主张把 TBOS 建立在专有特殊硬件的基础之上，但是，不应该让已经唾手可得的硬件能力白白地浪费掉。发挥硬件的效能既可以增强 TBOS 的功能，也可以缩小 TBOS 的尺寸，比如，在基于指纹法的代码完整性检查中，运用硬件的哈希运算特性，取代软件的相应函数功能，既可以增强完整性检查的可信性，也可以缩小监控器的尺寸。

本文定位的硬件效能是支持 Intel 的 TXT 技术[8] 的 CPU 效能和支持 TCG 的 TPM 规范[9] 的可信芯片的效能，因为，这些类型的硬件目前已经可以广泛用于装备普通计算机。发挥 Intel TXT 兼容 CPU 和 TCG TPM 兼容芯片的效能，对于增强 TBOS 对可信性的支持能力，对于缩短软件相对于硬件的滞后，具有重要的有意义。

1.3 建模方法的指导原则

本文的研究任务是对 TBOS 建模，根据以上的讨论，结合实际应用的发展状况，我们确定建模方法的以下 3 个指导原则。

原则 1（兼容性原则） TBOS 的设计与实现要尽可能地与现有系统兼容，为了使用 TBOS 的可信性支持功能，不应该对 Web 应用软件进行修改，为了开发 TBOS，不能要求对 Web 应用运行环境或操作系统内核进行颠覆性的改造。这是为了保持应用发展的继承性和确保信息资产的可利用性。

原则 2（硬件潜力原则） TBOS 的设计与实现不能要求为之配备专门的特殊硬件，以免削弱其实际应用的可接受性，但是，TBOS 的设计与实现应该充分发挥现代硬件的先进特性（重点是兼容 Intel 的 TXT 技术的 CPU 特性和兼容 TCG 的 TPM 规范的可信模块特性等），以改善软件技术滞后于硬件技术的局面，解决单靠软件手段难以解决或不好解决的问题。

原则 3（最小化原则） 尺寸大和复杂性高潜在地形成了解决软件系统安全性、可信性问题的障碍，研究 TBOS 时从一开始就要把规避这种障碍作为立足点，设计 TBOS 模型时要把模型的最小化作为重要的设计目标，使得以该模型为基础开发 TBOS 软件时能缩小程序的尺寸，降低程序的复杂度。

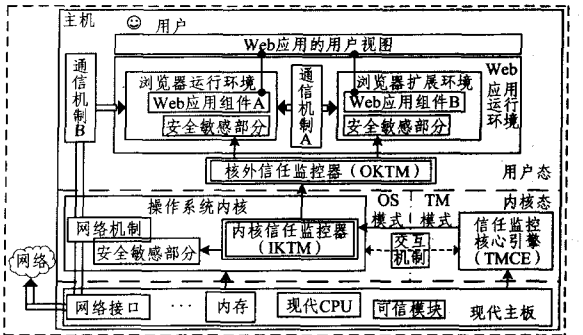
原则 1 表明不要求对已有的 Web 应用作改动，Web 应用的运行环境要有相应的配合，但不需要太大的修改，对操作系统的修改也不能导致对原系统的大规模推倒重来。原则 2 规定 TBOS 的应用不能夹带配备特殊硬件的附加条件，我们要做的是挖掘已有硬件的潜能，力求改变一方面软件能力不足而另一方面硬件能力浪费的状况。原则 3 强调“欲以 TBOS 为可信性提供支持，必须排除 TBOS 自身固有的潜在不可信因素”的理念，软件的尺寸和复杂性普遍被认为与潜在的不可信因素有关，因此，我们通过 TBOS 的最小化来增强其自身的可信性。

2 操作系统信任基的模型研究

2.1 TBOS 的体系结构设计

根据我们前期研究的积累和国际上的研究成果[10-12]，Web 应用程序的运行环境可以概括为图 1 的形式。Web 应用程序主要由浏览器运行，浏览器提供的运行环境运行 Web 专用的应用程序，为浏览器扩展的运行环境可运行非 Web 专用的应用程序，包括传统的代码（Legacy Code 或 Native Code）。无论浏览器运行环境还是浏览器扩展环境，都有用于支持安全性的安全敏感部分，那是 Web 应用环境开发者需要考虑的问题，本文关注的是如何确保它们能够可信地工作。

如图 1 所示，我们把 TBOS 的体系结构设计为信任监控核心引擎（TMCE，Trust Monitoring Core Engine）、内核信任监控器（IKTM，In-Kernel Trust Monitor）和核外信任监控器（OKTM，Out-of-Kernel Trust Monitor）三大部分。其中，TMCE 独立于操作系统内核，IKTM 处于操作系统内核之中，OKTM 处于应用空间之中。这样规划的原因是既要降低被监控对象对 TBOS 的潜在影响，也要便于 TBOS 有效地获取被监控对象的可信性相关语义。文献[7]中定义的基本信任基存在于 TMCE 之中。



注：操作系统信任基被设计为包含TMCE、IKTM和OKTM三大部分，现代CPU指兼容TXT的CPU，TXT指Intel的可信执行技术，可信模块指兼容TPM的可信芯片，TPM指可信计算组织（TCG）的可信平台模块，OS指操作系统，TM指信任监控，系统空间被分为用户态和内核态两种形态，内核态被分成OS模式和TM模式两种模式，现代主板指支持现代CPU和可信模块的计算机主板，实线箭头表示可信性支持关系。

图 1 操作系统信任基的体系结构及工作环境示意图

TMCE 是 TBOS 中最关键的部分，它实现核心的可信性支持功能，具有很高的独立性，对 TBOS 的可信运转以及整个系统的可信监控具有决定性的作用，它负责监视操作系统内核的行为，并能排除操作系统内核对它的干扰。IKTM 对操

作系统内核中的安全敏感部分的可信性进行监控，它的可信性由 TMCE 给予确保。OKTM 对用户态中的安全敏感部分的可信性进行监控，特别地，它负责对 Web 应用运行环境中的安全敏感部分的可信性进行监控，它的可信性由 IKTM 给予确保。

2.2 TBOS 模型研究的基本思想

把 TBOS 建模方法的指导原则与 TBOS 的体系结构结合起来，可以形成 TBOS 模型研究的基本思想，它可以由以下 4 个要点描述。

要点 1　分析在 Web 应用运行环境中和操作系统内核中提取安全敏感部分的方法（这方面国际上有很多可利用的成果），研究在此基础上构造信任监控器 OKTM 和 IKTM 的方法，研究分别面向 Web 应用运行环境和面向操作系统内核的可信性监控策略（Policy），以及对 OKTM 和 IKTM 进行保护的方法。

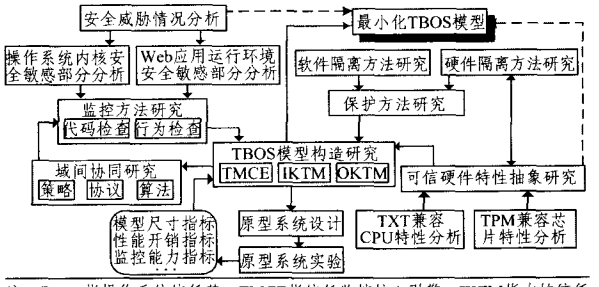
要点 2　以 Intel 的可信执行技术（TXT）和 TCG 的可信计算技术（TCT）为代表，研究在软件中发挥现代硬件潜能的方法，以 TBOS 的需要为驱动，建立能反映 TXT 与 TCT 综合应用特点的可信硬件特性的抽象模型（THFAM），用于支持 TBOS 建模。

要点 3　以 THFAM 为支撑，研究在内核态中构造操作系统模式和信任监控模式两种彼此独立的运行模式的方法，以及以这两种运行模式为基础的 TMCE 的构造方法，使 TMCE 能脱离操作系统内核独自运行，具有监视操作系统内核行为的能力和排除操作系统内核及其它软件的影响和破坏的能力。

要点 4　借助 THFAM，研究通过可信硬件特性的支持能力延伸，对 IKTM 模型和 OKTM 模型进行精简的方法，以及把精简化的 IKTM 模型和 OKTM 模型与 TMCE 模型集成起来构造最小化 TBOS 模型的方法。

3　操作系统信任基的建模方法

TBOS 的建模方法可以用图 2 描述。从宏观上说，TBOS 的建模方法就是从 Web 应用系统和操作系统内核面临的安全威胁出发，以发挥 Intel 的 TXT 兼容 CPU 和 TCG 的 TPM 兼容芯片所拥有的可信性支持特性为支撑，研究构建 TBOS 的最小化模型。具体地说，TBOS 的建模方法可以表示为分析、研究和实验等 3 个主要环节。分析环节是基础，研究环节是核心，实验环节是保障。建模工作的实施从分析环节开始，接着是研究环节，随后是实验环节。一旦启动之后，3 个环节交替往复，相互依赖，相互促进。



注：TBOS指操作系统信任基，TMCE指信任监控核心引擎，IKTM指内核信任监控器，OKTM指核外信任监控器，TXT指Intel的可信执行技术，TPM指可信计算组织（TCG）的可信平台模块，虚线箭头表示逻辑因果关系，实线箭头表示工作结果流向，粗线箭头表示最终结果的形成方向。

图 2　操作系统信任基建模方法的思想体系示意图

3.1 TBOS 建模的分析环节

这个环节要解决承前启后的问题，即利用已有的研究成果，建立 TBOS 建模的前提条件，以便以同领域当前国际水平作起点，实施 TBOS 建模的工作。这个环节实行以下措施：

措施 1-1　对 Web 应用系统和操作系统内核相关的安全威胁进行归纳和总结，进一步明确 TBOS 主要针对的安全威胁。分析从软件系统中提取安全敏感部分的方法，这方面国际上有不少可以借鉴的工作。安全敏感代码执行时往往需要提升的特权，我们要分析面向特权分离的安全敏感程序切分方法。

措施 1-2　结合浏览器的体系结构、工作原理和 Web 应用程序的执行方法，分析用于运行 Web 应用的浏览器运行环境和浏览器扩展环境的安全敏感部分的构成和工作方法。结合操作系统的体系结构、工作原理和操作系统安全机制的特点，分析操作系统内核的安全敏感部分的构成和工作方法。分析破坏 Web 应用运行环境和操作系统内核的安全敏感部分的代码完整性和控制流完整性的可能方法。

措施 1-3　结合软件系统可信运行和保护方面的需要，分析支持 Intel TXT 技术的 CPU 特性，重点是硬件虚拟化、动态信任根（Late Launch）、段粒度访问控制、页粒度访问控制、特权环等方面的特性。分析与 TCG TPM 规范兼容的可信芯片的特性，主要是代码完整性度量、存储、报告和密码运算方面的特性。

3.2 TBOS 建模的研究环节

这个环节解决构造 TBOS 模型所涉及的主要方法问题，关键是 Web 应用运行环境和操作系统内核中的安全敏感部分的可信性监控以及 TBOS 的保护两个方面。这个环节实行以下措施：

措施 2-1　研究 Web 应用运行环境和操作系统内核中的安全敏感部分的可信性的确定方法。安全敏感部分的可信性包括代码可信和行为可信两个方面，所以，需要确定代码可信性的检查方法和行为可信性的检查方法。代码可信指代码没有受到非法篡改，行为可信指代码按预期工作。代码的可信性检查需要确定检查方法和检查时机，我们采用代码指纹法判断代码的完整性。与时机相关，有代码装载时检查和代码运行时检查。代码装载时检查已经有了不少比较成熟的方法，可以直接借鉴。在代码运行时检查方面，可以考虑的角度很多，我们重点定位于针对控制流完整性的检查，实施运行时的监控。

措施 2-2　研究确定由 TMCE，IKTM 和 OKTM 3 部分构成的 TBOS 的体系架构设计。根据浏览器运行环境和浏览器扩展环境中的安全敏感部分的特点，确定 OKTM 的构造方法，确定由 OKTM 实施的监控策略，OKTM 按照该策略监控 Web 应用运行环境的安全敏感部分的可信性。根据操作系统内核的安全敏感部分的特点，确定 IKTM 的构造方法，确定由 IKTM 实施的监控策略，IKTM 按照该策略监控操作系统内核的安全敏感部分的可信性。根据陷入恶意用户控制之中的操作系统内核可能对 IKTM 产生的危害，确定 TMCE 的构造方法，确定 TMCE 实施的监控策略，TMCE 按照该策略监控操作系统内核相关行为的可信性。

措施 2-3　根据 OKTM，IKTM，TMCE 以及操作系统内核之间在 TBOS 实施可信性监控时可能存在的依赖关系，研

究它们协同执行可信性监控任务的方法，构造协同策略、协同协议和协同算法，以此来确保协同监控任务的有效执行。协同策略确定协作个体在协同环境中执行监控任务的规则，协同协议确定协作个体在执行监控任务的过程中通过交互进行配合的方式，协同算法确定协作个体在协同执行监控任务过程中的各个阶段的具体行为。

措施 2-4　针对 OKTM，IKTM 和 TMCE 所处环境存在差异的情况，研究对 TBOS 进行保护的有针对性的方法。保护的前提基础是通过约束对内存区域的访问来建立隔离域。OKTM 在用户态运行时，应考虑在进程的层次为它建立隔离域。IKTM 运行在内核态并与操作系统内核共存时，应考虑在操作系统内核的层次为它建立隔离域。TMCE 需要拥有监控操作系统内核有关行为的能力，需要在内核态运行，并且能独立于操作系统内核。我们通过在内核态定义两种模式来处理这种需求，这两种模式是操作系统(OS)模式和信任监控(TM)模式，操作系统内核运行在 OS 模式，TMCE 运行在 TM 模式，TM 模式具有比 OS 模式更高的主动性。据此研究 OS 模式和 TM 模式的构造方法。内核态的 TM 模式形成 TMCE 的隔离域。OKTM 和 IKTM 主要通过软件隔离方法保护，TMCE 通过硬件隔离方法保护。研究确定隔离域间的可信性监控方法、控制转移方法和干扰排除方法。

措施 2-5　对措施 1-3 中涉及的 TXT 兼容 CPU 的特性和 TPM 兼容芯片的特性进行抽象，对有利于支撑 TBOS 的特性进行提炼，屏蔽产品相关的细节，将其表示成易用的特性形式，建立可信硬件特性抽象模型，供 TBOS 建模使用。基于可信硬件抽象模型，研究可信硬件特性在硬件隔离中的运用方法，确定可信硬件特性在内核态的 TM 模式构造中的运用方法，以及在 TMCE 构造中的运用方法。研究可信硬件特性在系统可信性监控中的运用方法，以及运用于缩小监控器尺寸的方法。比如，在基于指纹法的代码完整性检查中，运用硬件的哈希运算特性，取代软件的相应函数功能，既可以增强完整性检查的可信性，也可以缩小监控器的尺寸。

3.3　TBOS 建模的实验环节

这个环节解决 TBOS 模型的工程可实现性问题和指标落实问题，其中涉及模型的最小化问题。这个环节实行以下措施：

措施 3-1　以 TBOS 模型构造工作取得的阶段性结果为基础，研究原型系统的构造方法。此处所说的原型系统并不一定是整个 TBOS 模型的原型系统，可以是该模型任何局部的原型系统。根据需要构造相应的原型系统，设计并实施原型系统实验，实验类型包括：用于检验模型在监控系统的可信性方面的能力的监控能力实验、用于检验模型受保护程度的保护程度实验、用于检验硬件特性对模型的支持能力的硬件效能实验、用于测试模型的实现所产生的性能开销的性能开销实验等。通过实验获取关于模型尺寸(即代码量)、监控能力、性能开销等方面指标的结果。

措施 3-2　对实验结果进行分析，评估当前的 TBOS 模型构造方案在模型尺寸、监控能力、保护程度、硬件效能发挥和性能开销等方面的效果，研究模型的构造方法对各项相关指标所产生的可能影响，研究调整模型构造方案的方法，特别是通过充分发挥硬件效能，精简 IKTM 和 OKTM，缩小 TBOS 模型尺寸，逐步达到 TBOS 最小化的方法。

4　操作系统信任基建模中的关键问题

在 TBOS 建模中，必须解决最小化 TBOS 的保护和多域协同的系统可信性监控等两个关键问题。

4.1　最小化 TBOS 的保护

在信息安全领域，所有与监控相关的主题都涉及到监控器的保护问题。

一个与监控相关的系统(S)可以简要地表达为监控方(M)对被监控方(O)进行监控的系统，这样的系统的本质思想就是由 M 确定 O 是否符合安全要求，是否值得信任，M 就是通常所称的监控器。

显然，关于一个系统是否安全或可信的结论(C)，是借助于 M 的工作来做出的。因此，对 M 的保护非常重要，因为，如果 M 受到了干扰或破坏，那么，结论 C 就难以令人信服。

长期以来，监控器的保护始终是困扰着人们的一个棘手问题，国内外有大量的研究工作和成果，但是，这个问题在现实中迄今仍没有令人满意的解决方法。原因是多方面的，其中非常关键的一点是：对监控器 M 的保护是与系统 S 的本质密切相关的，S 的发展也会给 M 的保护提出新的挑战。

最小化 TBOS 的保护属于监控器的保护问题，其中，监控器 M 对应的是 TBOS，系统 S 对应的是 Web 应用系统，包含操作系统和 Web 应用运行环境等成分，被监控方 O 对应的是操作系统中和 Web 应用运行环境中的安全敏感部分。

该问题的特点是 TBOS 跨越了内核态和用户态，包含 TMCE，IKTM 和 OKTM 3 个部分，保护问题涉及到 3 个层面。我们设定的威胁模型假定操作系统内核也可能受到攻击，要求即使操作系统内核受到了黑客控制也不能致使 TBOS 做出错误的结论。因此，我们无法寄希望于由操作系统内核来提供对 TBOS 进行保护的根基。

问题的难点之一是：现实中的很多主流操作系统的设计给操作系统内核赋予了最高的访问特权，它有权对各个层次的软件进行访问。如何防止受黑客控制的操作系统内核影响 TBOS 的工作成了问题的关键。我们不可能要求修改操作系统内核来改变这种局面，因为这违反了建模方法需遵循的指导原则 1。

近年来国内外热议的基于虚拟机监控器(VMM)的保护方法也不适宜解决这个问题，因为实现 VMM 所需的代码量本身就不小，不符合建模方法需遵循的原则 3。再者，这种方法也不利于获取被监控对象中一些重要的可信性相关语义。

4.2　多域协同的系统可信性监控

多域协同的系统可信性监控问题是由 TBOS 的应用目标导向的性质所催生的。

TBOS 的应用目标是帮助确定 Web 应用的可信性，Web 应用程序的运行特点要求对 Web 应用运行环境中的安全敏感部分的可信性进行监控，而这又有赖于操作系统内核的可信性给予支撑，相应地，后者又有赖于操作系统内核以外的可信性的支持，由此，TBOS 需要包含 OKTM，IKTM 和 TMCE 3 个部分，它们分别落在 3 个彼此独立的工作域(或称信任域)中。

存在 OKTM 所在的信任域(简称 OKTM 域)的必要性是为了能够有效地获取 Web 应用运行环境中的可信性相关语义。相应地，IKTM 域之所以存在，是有效地获取操作系统内

核的可信性相关语义的需要。而 TMCE 域，则是为了建立 TBOS 自身的可信性所必需的。

为了确保能够有效地监控 Web 应用运行环境的可信性，并且，确保可信性监控行为和结果的可信性，单靠 OKTM 是难以做到的，同样，IKTM 或 TMCE 都难以独自担此重任，只有通过 OKTM，IKTM 和 TMCE 三者的有效协作，才有可能完成该项任务。

由于 OKTM，IKTM 和 TMCE 处在彼此独立的信任域中，它们之间的协作属于跨信任域的实体之间的协同工作，因此，对 Web 应用系统的可信性进行监控的问题，属于一种多域协同的系统可信性监控问题。

在多域协同的系统可信性监控中，需要结合域间的可信性支持关系，从监控任务的功能需求和可信性需求两个方面把一项监控任务分解为必要的子任务，分派给相关的协作实体，由它们合作完成。TBOS 中的协作实体就是 OKTM，IKTM 和 TMCE。这里，需要构造合理的协同策略与交互协议，同时，需要排除域间的潜在干扰。

另外，除了 OKTM，IKTM 和 TMCE 之间的配合工作之外，多域协同的系统可信性监控还涉及到 TBOS 与操作系统的其它部分的协作，尤其是 IKTM 与操作系统内核其它部分的协作。在这类协作中，排除域间潜在干扰的难度更大。这里，主要是要排除可信性受到损害的操作系统其它部分对 TBOS 的干扰。

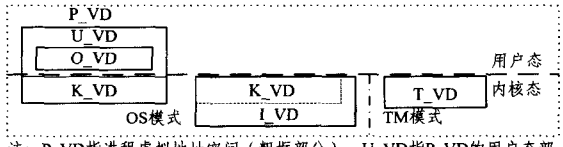
5 操作系统信任基建模中的关键技术

根据前面的讨论，参见图 1，TBOS 建模的本质就是构造由 TMCE，IKTM 和 OKTM 组成的 TBOS，关键任务是使 TMCE，IKTM 和 OKTM 正常工作并得到保护，从最基础的层面说，就是使它们在内存中的程序正常执行并得到保护。围绕着这样的关键任务，必须着重处理面向 TBOS 的内存空间管理和 TBOS 相关的控制转移等两项关键技术。下面以 Linux 操作系统为原型进行讨论。

5.1 面向 TBOS 的内存空间管理技术

这项技术为 TBOS 分配符合要求的内存空间，并防止内存空间中的内容被非法访问。这涉及到内存虚拟地址空间的分配、物理地址空间的分配以及虚拟地址到物理地址的翻译。

参见图 3 中的概念式示意。通常，操作系统为每个进程分配一个独立的虚拟地址空间(P_VD)，操作系统内核(OSK)的虚拟地址空间(K_VD)在每个进程的虚拟地址空间中都有相同的映射，P_VD 由用户态部分(U_VD)和 K_VD 两部分构成，进程的程序映射在 U_VD。



注：P_VD指进程虚拟地址空间（粗框部分），U_VD指P_VD的用户态部分，K_VD指P_VD的内核态部分，对应操作系统内核的虚拟地址空间，T_VD指信任监控核心引擎（TMCE）的虚拟地址空间，I_VD指内核信任监控器（IKTM）的虚拟地址空间，O_VD指核外信任监控器（OKTM）的虚拟地址空间，OS指操作系统，TM指信任监控。

图 3 操作系统信任基的相关虚拟地址空间的概念式示意图

TMCE，IKTM 和 OKTM 各自对内存空间有不同的要求。TMCE 独立于 OSK，需要有独立的虚拟地址空间(T_VD)。IKTM 既要方便地获得 OSK 中安全敏感部分的可信性相关语义，又要排除 OSK 可能产生的影响，可以为 IKTM 分配独立的虚拟地址空间(I_VD)，并把 K_VD 映射到其中，使得 IKTM 可以看到 OSK，但 OSK 看不到 IKTM。OKTM 在用户态运行，它的虚拟地址空间(O_VD)落在进程的 U_VD 中，可以通过设定特定区域的方式加以保护。

T_VD，K_VD 和 I_VD 都是处于内核态的虚拟地址空间，但是它们互不相同。T_VD 处于 TM 模式，K_VD 和 I_VD 处于 OS 模式，TM 模式比 OS 模式具有更高的主动性，T_VD 中的程序可以捕获 K_VD 和 I_VD 中的事件，对相应的行为进行监控。

这些虚拟地址空间分配方案能否有效地付诸实现，地址翻译策略的设计是关键。

5.2 TBOS 相关的控制转移技术

这项技术确定 TBOS 相关的不同虚拟地址空间之间的控制转移方式，这里的控制转移，指的是 CPU 执行了某个虚拟地址空间中的一条指令之后，转向执行另一个虚拟地址空间中的指令。

通常，进程执行系统调用时，控制由 U_VD 向 K_VD 转移，系统调用返回时，控制向反方向转移，这是常见的控制转移。

TMCE 可以通过捕获 OSK 和 IKTM 中的特定事件对 OSK 或 IKTM 进行监控，通过事件触发，控制可以从 K_VD 和 I_VD 向 T_VD 转移。可以采取在 OSK 中插入钩子(Hook)的方法，实行 IKTM 对 OSK 的监控，可以为 I_VD 设计专门的入口和出口，当 K_VD 中的代码执行到某钩子的相应处时，将启动一个程序调用，通过专门的入口，控制由 K_VD 向 I_VD 转移，执行对应的钩子处理程序，处理程序执行完毕，通过专门的出口，控制向反方向转移，回到 OSK 中执行。

从合法的起点，通过合法的方式，到达合法的终点的控制转移，称为合法的控制转移。本项技术的关键是确定 TBOS 相关的所有合法的控制转移。为说明的简单起见，此处说对 OSK 进行监控，实际上主要是对 OSK 的安全敏感部分进行监控。

6 相关工作

本文工作的贡献反映在综合实施 TBOS 的最小化、硬件协助的可信性支持和 Web 应用系统的可信性等 3 个方面的内容，主要体现于通过“硬件—操作系统—应用”3 个层面的措施的融合，建立有效、可信的 TBOS，并服务于 Web 应用的可信性监控。国际上的不少研究与本文的研究相关，不过，还没有一项研究能全部实现本文综合考虑的全部内容。

在国际上，有采用信任基(Trust Base)这个概念的研究工作[13]，但并不多。大多数的研究一般采用可信计算基(TCB，Trusted Computing Base)[14,15]这个概念。TCB 这个概念涵盖的范围相对比较广，很多研究工作虽然使用 TCB 这个概念，但常常仅涉及 TCB 的某个局部。我们的工作采用 TBOS 概念，使目标定位更加有针对性和具体化。

有关尽量缩小软件安全机制的尺寸的研究具有悠久的历史。1972 年 J. P. Anderson 讨论引用监控器(RM，Reference Monitor)时[16]，就指出 RM 的实现应该尽可能地小。1975 年 J. H. Saltzer 和 M. D. Schroeder 给出安全机制设计的八大原

则时把经济性原则列在首位[17]。遗憾的是，总体上说，由于软件系统变得越来越庞大，实际系统中的 TCB 总是很大。

依靠硬件来增强系统安全性或可信性的想法也是由来已久，实际上，TCB 的定义中就包含有硬件的内容[15]。有大量的研究工作致力于基于硬件增强安全性和可信性，不过，早期的很多工作都定位在专用特殊硬件上[21]，这削弱了系统的用户可接受性，很少得到广泛的采用。

佐治亚理工学院[12]从应用软件和系统软件两个层面研究降低 TCB 复杂性的问题，提出了 Nizza/AppCore 方法。它从系统软件和应用软件中提取出安全敏感部分，分别构成可信的 Nizza 体系结构和可信的 AppCore，由这两者组成 TCB。Nizza 体系结构建立在 L4 微内核之上。该方法借助 TCG TPM 实现可信引导。AppCore 的尺寸是万行代码的数量级，Nizza 的尺寸是十万行代码的数量级。

卡内基-梅隆大学[18]以缩小尺寸、减低复杂性为目标，借助 AMD 的 SVM 技术，提出了一个操作系统内核代码完整性的监控模型，设计实现了一个称为 SecVisor 的小型监控器。它利用 CPU 提供的内存虚拟化特性，实现监控器与操作系统内核的隔离，确保监控器的可信性，实现对操作系统内核代码完整性的有效监控。SecVisor 的尺寸是千行代码的数量级。

卡内基-梅隆大学[19]以 TCB 最小化为目标，通过综合运用 AMD 的 SVM 技术和 TCG 的 TPM 技术，提出一个称为 Flicker 的可信执行体系结构，构造一个独立于操作系统的隔离执行环境，用于执行应用系统的安全敏感代码，确保应用系统的可信性。Flicker 的一个非常突出之处是完全抛弃操作系统对可信性的一切支持，仅依靠硬件特性提供的可信性支持来建立应用系统的可信性。Flicker 的尺寸是百行代码的数量级。

佐治亚理工学院[20]研究虚拟机环境中的操作系统内核的安全监控问题，提出了一种监控器与被监控的操作系统内核处于同一个虚拟机中的监控架构，称为 SIM。研究的重点是防止被监控的操作系统内核对监控器进行干扰或破坏。SIM 利用 Intel 的支持 VT 技术的 CPU 的硬件虚拟化特性和内存保护特性，依靠虚拟机监控器(VMM)实现操作系统内核安全监控器与操作系统内核的隔离，确保安全监控器的有效性和可信性。

谷歌公司[10]研究在 Web 应用系统中安全地使用本机代码(Native Code)的问题，它提出了一个称为 NaCl 的系统体系结构，为本机代码建立受控的执行环境，以充分发挥本机代码的效能，防止本机代码损害 Web 应用系统的可信性。NaCl 利用 Intel x86 CPU 的段保护特性，设立内部沙箱(Sandbox)，在进程中划出安全子域，创建隔离的本机代码执行环境，约束本机代码对内存的访问，消除损害 Web 应用系统可信性的隐患。

除了借助 TCG 的 TPM 进行引导时的代码度量之外，文献[12]没有考虑利用其它硬件特性或对 TCB 进行有效保护的问题，另外，它得到的 TCB 还是太大。文献[18]只考虑操作系统内核可信性的监控，没有考虑应用系统可信性的监控，而且，它只监控代码的完整性，不涉及控制流的完整性，就是说，没有考虑操作系统内核行为方面的可信性。文献[19]构造了只有百行代码数量级的 TCB，这很惊人，但是，它完全抛开了操作系统对可信性的支持作用，却又依赖操作系统内核启动隔离执行环境的运行，致使隔离执行环境的可信性存在隐患。另外，其性能开销过大，当前的硬件能力难以支撑。文献[20]的最大不足是依赖虚拟机监控器(VMM)的支持，VMM 的代码量不小，达不到最小化的要求。另外，它也只考虑对操作系统内核的监控，没有考虑对应用的监控。文献[10]为充分发挥本机代码的效能并确保 Web 应用系统的可信性提供了一个很好的方法，不过，相对于本文而言，它只是考虑了应用空间的可信性问题，并不涉及操作系统的可信性问题。

我们的前期工作[7]提出了 TBOS 的思想，文献[7]侧重于从功能视角对 TBOS 进行描述。本文的工作一方面从实现技术的视角对 TBOS 的思想进行深化，另一方面在这种深化的基础上，着重研究 TBOS 模型构造的方法学问题。

结束语 本文从可信的 Web 应用对操作系统支持能力的依赖性出发，以为 Web 应用的可信性提供有效的支持为应用导向，以充分发挥现代硬件的先进特性为切入点，研究建立操作系统信任基(TBOS)的最小化模型的方法，为开发可信性有保障、性能开销可接受的 TBOS 建立方法学基础。本文具有如下几个方面的创新性：(1)提出以监控 Web 应用的可信性为导向，发挥 Intel TXT 兼容 CPU 和 TCG TPM 兼容可信芯片的能力，构造操作系统信任基最小化模型的方法；(2)提出以硬件保护与软件保护相结合的方式，保护最小化的操作系统信任基的方法，增强操作系统信任基的防干扰和防破坏能力；(3)提出通过多信任域间实体的协同，对系统的可信性进行监控的方法，包括协同策略、协同协议和协同算法。

参 考 文 献

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

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

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

[4] 石文昌，单智勇，梁彬，等. 细粒度信任链研究方法[J]. 计算机科学，2008，35(9)：1-4

[5] Shi Wenchang. On Design of Trusted Software Base with Support of TPCM[C]// The 2009 International Conference on Trusted Systems (INTRUST 2009), LNCS. Springer-Verlag, 2010

[6] Loscocco P A, Smalley S D, Muckelbauer P A, et al. The Inevitability of Failure: The Flawed Assumption of Security in Modern Computing Environments[C]// Proceedings of the 21st National Information Systems Security Conference. Oct. 1998: 303-314

[7] 石文昌. 操作系统信任基的设计研究[J]. 武汉大学学报：信息科学版，2010，35(5)

[8] Intel Corporation. Intel Trusted Execution Technology-Software Development Guide-Measured Launched Environment Developer's guide[R]. 315168-005. June 2008

[9] Trusted Computing Group. TPM Main-Part 1 Design Principles-Specification Version 1.2[S]. July 2007

(下转第 45 页)

器并且读写器平均通信时间长度为 10 的情况下($M=18$, $N=5$, $d=10$),在给定负载的基础上,FHR 协议系统输出 S 与 PULSE 算法和 Colorwave 算法的对比如图 7 所示。可以看出,开始三者的系统输出差不多,但是随着读写器负载的增大,FHR 协议的系统输出 S 要比 PULSE 算法和 Colorwave 算法高,这是因为 FHR 协议采用了多频率,避免了读写器冲突。

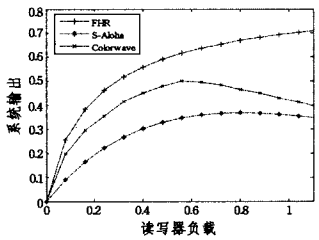


图 7　三种算法的对比

需要指出的是,在数据分析中假设读写器已经进入了同步状态,没考虑同步时间消耗,并且忽略了传输延迟和数据的处理时间,因此理论结果较理想。在实际中,性能会比此理论值低。

结束语　本文提出了一种基于跳频预约的 RFID 读写器网络 MAC 协议 FHR,并对此协议进行了分析。FHR 协议通过竞争预约时隙及对应的通信频率,并通知邻居读写器,避免了读写器-标签冲突;采用读写器和标签,并分别使用不同的频率通信机制及多频率跳频机制,避免了读写器频率冲突。对 FHR 协议进行的分析表明,此协议在读写器负载较大及读写器平均通信时间较长时,单位时间内读写器与标签通信成功概率高。FHR 协议不会产生读写器-标签冲突,并且频率利用率较高。

参 考 文 献

[1] 中华人民共和国科学技术部等. 中国射频识别(RFID)技术政策白皮书[OL]. http://www.rfidinfo.com.cn/,2006

[2] Engels D W, Sarma S E. The reader collision problem[C]// Proceedings of IEEE International Conference on Systems, Man and Cybernetics. Hammamet, Tunisia, October 2002:6-9

[3] Leong K S, Ng M L, Cole P H. The Reader Collision Problem in RFID Systems[C]// 2005 IEEE International Symposium on Microwave, Antenna, Propagation and EMC Technologies for Wireless Communications Proceedings. Beijing, China, 2005: 658-661

[4] EPCGlobal. EPC radio-frequency identity protocols class-1 generation-2 UHF RFID protocol for communications at 860MHz-960 MHz version 1.0.9 [OL]. http://www.epcglobalinc.org/Standards,2005

[5] RFID CHINA. 第二代 RFID 技术标签标准[J]. 中国防伪报道, 2006(5):39-44

[6] ETSI. ETSI EN 302 208-1 V1.1.1[OL]. http://www.esti.org,2004

[7] Waldrop J, Engels D W, Sarma S E. Colorwave: An Anti-collision Algorithm for the Reader Collision Problem [C]// IEEE Wireless Communications and Networking Conference (WCNC). New Orleans, Louisiana, USA, 2003:1701-1704

[8] Birari S M, Sridhar I. PULSE: A MAC Protocol for RFID Networks[C] //1st International Workshop on RFID and Ubiquitous Sensor Networks (USN). Nagasaki, Japan, 2005: 1036-1046

[9] Tang Z, Garcia-Luna-Aceves J J. Hop Reservation Multiple Access (HRMA) for Multichannel Packet Radio Networks[C]// Proc. IEEE IC3E'98, the Seventh International Conference on Computer Communications and Networks. Lafayette, Louisiana, US, 1998:388-395

[10] Tang Zhenyu, Garcia-Luna-Aceves J J. Hop Reservation Multiple Access (HRMA) for Ad-Hoc Net works[C]// Proc. of IEEE INFOCOM 1999, the Conference on Computer Communications, Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. New York, USA, 1999: 194-201

(上接第 6 页)

[10] Yee B, Sehr D, Dardyk G, et al. Native Client: A Sandbox for Portable, Untrusted x86 Native Code[J]. Communications of the ACM, 2010, 53(1):91-99

[11] Douceur J R, Elson J, Howell J, et al. Leveraging Legacy Code to Deploy Desktop[C]// Applications on the Web. 8th USENIX Symposium on Operating Systems Design and Implementation (OSDI 2008). USENIX Association, 2008

[12] Singaravelu L, Pu C, Härtig H, et al. Reducing TCB Complexity for Security-Sensitive Applications: Three Case Studies[C]// Proceedings of the 1st ACM SIGOPS/EuroSys European Conference on Computer Systems (Eurosys 2006). Belgium, Apr. 2006

[13] Guttman J, Herzog A, Millen J, et al. Attestation: Evidence and Trust[R]. MTR080072. Mar. 2008

[14] Nibaldi G C. Specification of a Trusted Computing Base[R]. M79-228. MITRE Corporation, Bedford, MA, USA, 1979

[15] DoD 5200.28-STD, Department of Defense Standard. Department of Defense Trusted Computer System Evaluation Criteria [S]. National Computer Security Center, Ft. Meade, MD, USA, Dec. 1985

[16] Anderson J P. Computer Security Technology Planning Study Volume II[R]. ESD-TR-73-51. Vol. II, Electronic Systems Division, Air Force Systems Command, Hanscom Field, Bedford, MA, USA, Oct. 1972

[17] Saltzer J H, Schroeder M D. The Protection of Information in Computer Systems[J]. Proceedings of the IEEE, 1975, 63(9): 1278-1308

[18] Seshadri A, Luk M, Qu N, et al. SecVisor: A Tiny Hypervisor to Provide Lifetime Kernel Code Integrity for Commodity OSes[C]// The 21st ACM Symposium on Operating Systems Principles (SOSP 2007). Washington, USA, Oct. 2007:335-350

[19] McCune J M, Parno B, Perrig A, et al. Flicker: An Execution Infrastructure for TCB Minimization[C] // Proceedings of the ACM European Conference on Computer Systems (EuroSys). Glasgow, Scotland, Mar. -Apr. 2008

[20] Sharif M, Lee W, Cui Weidong, et al. Secure In-VM Monitoring Using Hardware Virtualization[C]// The 16th ACM Conference on Computer and Communications Security (CCS 2009). Chicago, IL, USA, Nov. 2009:477-487

[21] 石文昌, 梁朝晖. 信息系统安全概论[M]. 北京: 电子工业出版社, 2009:341-361