文章编号:1009-3087(2014)01-0029-06

一种高效的虚拟桌面可信保证机制

石 勇,郭 煜,韩 臻
(北京交通大学 计算机与信息技术学院,北京 100044)

摘 要:虚拟桌面系统与传统PC桌面系统结构的不同,导致其在保证安全机制自身可信的同时,也会带来"语义差别"和效率降低等问题。提出了一种安全虚拟机完整性监控机制SVMIM(security virtual machine integrity monitor)。SVMIM采用混杂模式的安全结构,基于可信计算技术对虚拟桌面系统的代码加载过程进行监视和控制,有效克服"语义差别"问题,并保证安全机制自身的可信;同时,SVMIM基于虚拟桌面网络引导机制,在网络存储端使用存储克隆技术,最大程度地降低安全机制对系统性能的影响。系统性能分析和基于SVMIM原型系统进行的实验表明,该技术是可行的,并且相对于传统的虚拟桌面安全保障方案具有较大的性能优势。

关键词:虚拟桌面;可信平台;完整性度量;存储克隆
中图分类号:TP393.08 **文献标志码**:A

An Efficient Trusted Virtual Desktop Assuring Mechanism

SHI Yong, GUO Yu, HAN Zhen
(Beijing Jiaotong Univ., Beijing 100044, China)

Abstract: The difference in structure between virtual desktop system and traditional PC desktop system causes "semantic differences" and efficiency issues, when improving the trust level of the security mechanisms of virtual desktop system. A security virtual machine integrity monitor (SVMIM) was build based on network bootstrap mechanism. SVMIM adopted a hybrid security structure, monitored and controlled the loading process of executable files in virtual desktop systems, which could effectively overcome the shortcomings of "semantic gap" and ensure the trustworthiness of security mechanism. In addition, SVMIM used storage clone technology on network storage to reduce the impact of security mechanism to the system performance. The performance testing on SVMIM prototype and the performance analysis showed its flexibility and advantage.

Key words: virtual desktop; trusted platform; integrity measurement; storage clone

随着技术的发展,通过物理服务器的硬件复用机制降低信息系统的建设成本和运维成本,以虚拟化技术[1-2]为支撑构建虚拟桌面系统,已经被包括IBM、CISCO、华为、中兴在内的许多国内外知名企业所关注。虚拟桌面系统在降低运维成本、提高管理效率的同时,也带来了新的安全问题和性能问题。对于虚拟化桌面的安全环境构建,目前通常采用内部系统加固和外部虚拟机保护的方式实现,需要对虚拟机内的操作系统进行大量的数据解析等串行操作。然而,由于虚拟桌面系统基于硬件复用实现,1台物理主机上运行多个虚拟化终端,任何针对该终端的串行操作都将影响到所有桌面系统的效率。在1台CPU为Xeon X5650、内存大小为20 GB、硬盘为500 GB的服务器上进行多台虚拟机同时杀毒实验,虚拟机使用镜像相同,杀毒扫描目录也相同。当测试到8台虚拟机同时杀毒时,杀毒平均耗时已经由181 s上升到了2 421 s。

在这样的条件下,如何在有效保证虚拟桌面系统安全性的同时,提高系统的效率,成为了国内外研究的热点问题之一。

收稿日期:2013-06-16
基金项目:国家"973"重点基础研究发展规划资助项目(2007CB307101);教育部高等学校博士学科点专项科研基金资助项目(20120009110007);教育部高校创新团队项目(IRT201206);2012年铁道部科技研究开发计划资助项目(2012X010-B)
作者简介:石 勇(1982—),男,博士. 研究方向:云计算;可信计算. E-mail:stonefly128@126.com

为此,提出了一种高效的虚拟桌面可信保证机制 SVMIM。一方面,SVMIM 采用“混杂模式”的安全检测结构,通过使用虚拟桌面操作系统中的 Hook 机制来主动监视虚拟桌面操作系统调用可执行代码事件,并获取所调用的可执行代码相关信息,在保证安全机制可信的同时避免出现 VMI 结构所固有的“语义差别”问题;另一方面,针对业务流程相对固定的生产系统,SVMIM 采用网络方式启动和运行虚拟桌面,并利用存储克隆机制分离出虚拟桌面的变化部分,使得系统可信验证机制仅对此部分进行验证,从而大幅减少可执行代码的验证范围和数量,最大程度的减小安全机制给系统带来的性能影响。

主要贡献如下:1)提出了一种高效虚拟桌面可信保证机制 SVMIM,在保证虚拟桌面系统安全性的同时有效提高系统的效率;2)对 SVMIM 安全性和性能进行了分析并实现了原型系统,基于原型系统的实验表明 SVMIM 应用于业务流程相对固定的生产系统时,具有较高的性能水平。

1 相关研究

虚拟桌面系统通常由虚拟机监控器(virtual machine monitor,VMM)和在 VMM 控制下的一系列虚拟桌面组成。VMM 运行在虚拟桌面操作系统和硬件之间,为每个虚拟桌面操作系统提供虚拟的专门硬件支持,并且实现虚拟桌面之间的隔离。针对 VMM 自身的安全和可信保证,Gebhardt[3] 和 Hohmuth[4] 等认为应该采取以下措施:通过去除不必要的 VMM 代码来减小 VMM 的规模,从而提高 VMM 的可信。除了上述措施外,另一类保护 VMM 安全可信的方法是采用可信计算技术[5] 等方法来验证和保护 VMM 的完整性,比如通过 vTPM 的机制实现硬件 TPM 的虚拟,在 VMM 之上为每个虚拟机提供一个独立的可信根[6] 等。

在可信 VMM 保证基础上,Garfinkel 等提出了一种基于“Out-of-the-Box”思想的虚拟机系统安全设计结构 VMI(virtual machine introspection[7]:将针对虚拟机的安全机制放置在 VMM 中,保证安全机制自身免遭虚拟桌面内部恶意代码的攻击。VMI 可以对虚拟机 CPU 和内存等资源的安全状态进行监视、分析和判断,并对虚拟机的行为进行干预。其他一些典型基于 VMI 结构实现的原型系统还有 XenAccess[8] 等。但由于这种方法从外部硬件层次对虚拟机进行分析,存在“语义差别”(semantic gap)问题[9]:在虚拟机外部难以解释虚拟机操作系统的底层物理信息和运行状态的相关语义,比如特定内存区域的数据含义,具有较高的技术难度和较大的局限性。

为了解决“语义差别”问题,Lares[10] 采用了一种混杂(Hybrid)模式的安全结构,将安全机制分置于 VMM 和虚拟操作系统中,其中大部分安全功能都像 VMI 那样置于 VMM 和可信虚拟机中,另一部分则置于被检测虚拟机的操作系统中。这种方式在一定程度上解决了语义差别问题,但是没有考虑虚拟桌面系统实际应用环节的特点及其对性能的影响。虚拟桌面系统中,一台物理服务器通常运行多台虚拟机,而 Lares 需要对系统硬件和内存进行大量分析,多个虚拟桌面对共享物理资源的占用往往会成为系统性能瓶颈,磁盘的 IO 问题尤其明显。

2 基于网络引导的虚拟桌面可信保证机制

2.1 基于网络引导机制的虚拟桌面系统

虚拟桌面基于虚拟技术实现,是虚拟技术的一个应用。利用 VMM 的资源分配和安全隔离机制,一台物理服务器上可以创建并运行高达几十个虚拟机实例,每个虚拟机实例都安装运行虚拟桌面操作系统,如 Windows XP 等。用户在远程平台(如 PC、瘦终端、手机等)通过远程桌面协议(如 remote desktop protocol,RDP)连接到这些虚拟机实例,远程桌面协议将虚拟桌面显示传送到用户所在的远程平台上,并将用户在远程平台上键盘鼠标信息传送到虚拟桌面中。

在操作流程相对固定的生产系统中,虚拟桌面系统可以根据实际应用需求对虚拟桌面进行分类。比如某类型的虚拟桌面专门用于办公作业,则该类型的虚拟桌面使用 Windows 操作系统,并安装办公所需套件如 Office 等就能满足用户需求;另一类型的虚拟桌面专门用于开发,则该类型虚拟桌面可以使用 Linux 操作系统,并安装 gcc 或其他开发工具套件就能满足用户需求。每一类型的虚拟桌面,用户只保留各自的私有数据和少量特性软件。因此,同一类型虚拟桌面所对应的虚拟机镜像文件存在着同构性。

针对同一类型虚拟桌面镜像文件同构特点,基于网络引导机制来构建可信虚拟桌面,并且为每一类虚拟桌面建立一个虚拟机镜像母本,使用存储克隆机制为该类型中所有虚拟桌面建立克隆镜像文件。

网络引导是指在虚拟桌面系统运行时,使用网

络存储系统提供的虚拟机镜像文件来启动和运行虚拟机,如图 1 所示。

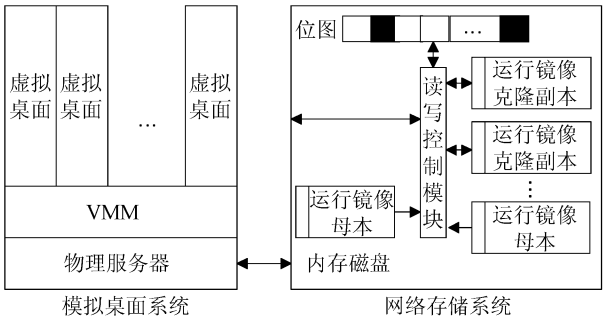

图 1　基于网络引导的虚拟桌面系统结构

Fig. 1　Architecture of virtual desktop system based on net-boot

使用网络存储系统作为网络存储服务端为虚拟桌面系统提供虚拟机镜像。在网络存储服务端上建立运行镜像母本,并且基于运行镜像母本为同一类型虚拟桌面建立独立的运行镜像克隆副本,母本和克隆副本之间通过 copy-on-write(CoW)[11] 位图实现关联。每个虚拟桌面在网络存储服务端都有一个对应其克隆副本物理存储块的位图集合。网络存储服务端的"系统读写控制模块"负责接收来自虚拟桌面系统的数据读写请求,并根据请求内容确定数据读写来源或目的地。当虚拟桌面进行写操作时,"读写控制模块"将数据写入克隆副本对应的物理块中,并将这些物理块在位图中对应的标识置位;当虚拟桌面在读取数据时,"读写控制模块"先检查读取数据块所对应的位图标识,如果位图标识被置位,"读写控制模块"从其克隆副本中读取数据,否则就从母本中读取数据。在非母本维护过程中,系统对母本只读不写,克隆虚拟桌面运行过程中任何操作都不会影响母本数据。

2.2　SVMIM 可信保证机制

为了建立系统的信任链,在虚拟桌面系统网络引导基础上,提出了 SVMIM 可信保证机制。SVMIM 采用"混杂模式"的安全结构,在虚拟桌面操作系统内核中实现可信度量模块,而安全验证模块在网络存储系统中实现。通过可信度量模块与安全验证模块交互,完成对虚拟桌面所要加载运行的代码进行可信验证和控制,如图 2 所示。

2.2.1　假设条件

SVMIM 可信保证机制能够正常工作的一个假设前提是 VMM 自身的可信以及硬件支持的内存保护能力。这一假设前提是合理的。首先,一般情况下 VMM 代码量都比较小[1],这为 VMM 的可信奠定了基础;其次,基于 DRTM(dynamic root of trust for measurement)机制[12] 的硬件体系结构可以进一步保证 VMM 的运行可信,Intel TXT 技术[13] 也为其提供了硬件保证;再次,目前大部分硬件处理器都支持内存页面的写保护和页面不可执行等安全功能。

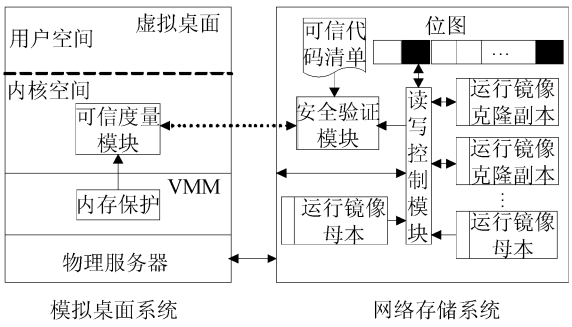

图 2　虚拟桌面可信构建方法结构流程

Fig. 2　Procedure of virtual desktop trust measurement

网络存储系统的安全可信是 SVMIM 正常工作的另一个假设前提。从图 2 可以看出,SVMIM 的大部分工作都是在网络存储系统上完成的,而且 SVMIM 也是虚拟桌面运行镜像文件的存放场地。由于其任务相对固定,网络存储系统安全可以通过物理安全、安全操作系统、可信计算平台技术等多种安全措施和机制来综合保证。

SVMIM 结构中,由于"运行镜像母本"在系统中的只读不写,在虚拟桌面系统启动时验证母本完整性一次就能保证其可信。因此,只要对母本的生成和维护进行适当控制,其完整性可以得到保证,SVMIM 中的"运行镜像母本"可以被认为是可信的。

2.2.2　可信度量

SVMIM 可信度量方法基于"可信度量模块"和"安全验证模块"来实现。"可信度量模块"运行在虚拟桌面操作系统的内核空间,通过拦截操作系统的模块加载,实现对虚拟桌面的代码调用过程监视。在代码被实际调用之前,"可信度量模块"将拦截的代码信息(文件路径名等)传送到网络存储系统中的"安全验证模块";"安全验证模块"基于系统预先制定的"可信代码清单"对调用代码进行验证并将验证结果传回"可信度量模块";"可信度量模块"根据代码可信验证结果控制系统是否实际加载代码运行,保证虚拟桌面系统中运行应用和服务的可信。

"安全验证模块"依赖于一个重要的数据库:可信代码清单。"可信代码清单"是系统预先根据虚拟桌面任务要求制定和生成的,它为每个虚拟桌面都保存一份可执行文件 hash 值列表,该列表涵盖了桌面所有被允许执行的代码文件的 hash 值,包括虚

拟桌面对应母本的可执行文件和每个桌面自身的特性软件。当虚拟桌面母本进行升级维护或者桌面特性软件进行更新时,必须对其相应的可信代码清单进行更新,保证更新代码通过可信验证。

“安全验证模块”接收“可信度量模块”传送过来的待验证代码文件信息后,完成以下操作:

1)通过“读写控制模块”判断相关文件是否被修改过。如果文件被修改过,那么它所包含的代码数据块在位图中对应的标识位一定有部分或全部为已设置状态。

2)如果代码文件没有被修改过,则该代码文件位于母本中,由于母本的“只读不写”性质,则表明该代码文件可信,“安全验证模块”将此验证结果返回给“可信度量模块”。

3)如果代码文件被修改过,“安全验证模块”就会通过“读写控制模块”从克隆副本中读取该文件的全部内容,并计算其 hash 值。

4)然后在“可信代码清单”中查询相关虚拟桌面的可执行文件 hash 列表,并检查待验证代码文件的 hash 值是否存在于其中;如果不在,该代码被认为不可信,否则,代码被认为是可信的;“安全验证模块”将验证结果返回给的“可信度量模块”。

2.2.3 安全性分析

SVMIM 采用了“混杂模式”的安全检测结构,该方法在结构和工作过程层面是安全的,表现在以下几个方面:首先,SVMIM 能主动监视虚拟桌面操作系统调用可执行代码事件,并获取所调用执行的可执行代码相关信息。由于 Hook 位于虚拟桌面操作系统内部,因此安全检测过程不存在 VMI 结构所固有的语义差别问题,Hook 的安全性可由 VMM 通过硬件的内存保护机制保证。其次,SVMIM 基于文件 hash 值来比对和验证系统所加载的代码,准确程度高。再次,安全结构简单,对虚拟桌面操作系统以及 VMM 的改动较少,从根本上保证了系统,尤其是 VMM 的可信程度。

但是在保证“可信度量模块”的自封闭性方面,SVMIM 依然存在风险。“可信度量模块”的 Hook 过程非常简单,而且不需要调用外部功能,但是它的另一个功能则相对复杂,即它和网络存储系统中的“安全验证模块”之间的通信功能。如何利用虚拟桌面操作系统现有的网络协议栈,实现通信功能,并且和“可信度量模块”一起形成一个自封闭代码集合,将是在进一步研究中主要探讨的问题。

3 原型系统和实验

3.1 原型实现

基于 SVMIM 设计并实现了原型系统。原型系统中,虚拟桌面系统物理服务器的配置为双 Xeon X5650 CPU,内存大小为 64 GB,硬盘为 240 GB 固态硬盘,千兆以太网接口,运行 40 个虚拟桌面;网络存储服务器为单 Xeon X5620 CPU,内存大小为 8 GB,硬盘为 6×240 GB 固态硬盘、千兆以太网接口。VMM 以 Xen 为基础研发,Xen 是一个开源的虚拟机监视器,针对 X86 系列计算机设计,能够支持多个虚拟机同时运行。网络存储系统采用 iSCSI Enterprise Target(IET)作为实现平台,该系统是一种基于 IP 协议的网络数据存储解决方案。在 iSCSI 开源基础上,通过修改 iSCSI 读写请求,实现了存储克隆机制和安全验证模块功能。安全验证模块中,采用 MD5 算法生成的 hash 值为可信代码清单。虚拟桌面操作系统使用 Windows XP,可信度量模块中通过 Hook 机制对 Windows XP 的 Createprocess( ) 和 Loadlibrary( )过程进行主动监视和控制。

3.2 性能分析

根据第 2 节,影响 SVMIM 性能的主要因素是安全验证模块在验证代码可信过程中的时间。这一时间包括 2 个主要部分:代码文件的 hash 值计算以及在可信代码清单中查询比对特定代码文件 hash 值的过程。实验表明,hash 值计算时间与文件大小成正比,而且平均每 250 kB 大小的文件计算 hash 值的时间在毫秒级,而采用二分法在有 1 000 000 项的 hash 列表中查询一个特定 hash 值所需要的时间在微秒级。因此,相对于代码文件 hash 值的计算过程,基于 hash 值的比对查询过程耗时可忽略不计。

为了使性能分析过程更加清晰,采用以下方式对 SVMIM 进行描述:假设 $V = \{v_1, v_2, \cdots, v_n\}$ 为1台物理服务器中的全部虚拟桌面集合,$v_1, v_2, \cdots, v_n$ 为虚拟桌面实例;M 表示虚拟桌面的初始运行环境,即系统运行镜像母本;C_i 表示虚拟桌面 v_i 的运行镜像克隆副本,其中,$1 \leq i \leq n$。

为了客观地评价 SVMIM 的性能水平,用非 SVMIM 模式作为对比性基础场景:虚拟桌面系统不使用 SVMIM,每个虚拟桌面都有独立的虚拟机镜像,系统在加载所有可执行代码之前都会进行可信验证,并且运行虚拟桌面数量和虚拟机镜像与 SVMIM 模式相同。

在上述对比性基础场景中,非 SVMIM 模式下,

使用$f_i(t)$表示v_i在运行过程中代码验证耗时,$F(t)$表示 V 中虚拟桌面同时运行时系统代码验证总耗时,则:

$$F(t) = \sum_{i=1}^{n} f_i(t) \tag{1}$$

在 SVMIM 模式下,假定对于任何一个虚拟桌面v_i,它调用C_i中代码的概率为p_i,调用 M 中代码的概率为$1 - p_i$,其中,$0 \le p_i \le 1, 1 \le i \le n$,使用$s_i(t)$表示$v_i$在运行过程中代码验证耗时,$M(t)$表示系统初始化时母本验证耗时,$S(t)$表示 V 中虚拟桌面实例同时运行时系统代码验证总耗时。根据第 2 节描述,如果虚拟桌面实例v_i从 M 中调用代码运行,SVMIM 认为这些代码是可信的,因此,不需要对其进行可信验证,而只需要对C_i部分的代码进行验证$(1 \le i \le n)$,因此有:

$$s_i(t) = f_i(t) \times p_i \tag{2}$$

$$S(t) = M(t) + \sum_{i=1}^{n} s_i(t) \tag{3}$$

用 F 表示 2 种模式的性能比,见式(4)。F 越小,表示 SVMIM 相对于非 SVMIM 模式的性能越高。

$$F = \frac{S(t)}{F(t)} \tag{4}$$

在生产系统中,由于操作模式和操作流程规范性要求,各虚拟桌面的运行环境基本相似,因此它们访问C_i的概率p_i可以被认为是相同的,假定$p_i = p$,$0 \le p \le 1, 1 \le i \le n$,由式(2)、(3)得出:

$$S(t) = M(t) + p \times \sum_{i=1}^{n} f_i(t) \tag{5}$$

由式(1)、(4)、(5),得出:

$$F = \frac{M(t)}{\sum_{i=1}^{n} f_i(t)} + p \tag{6}$$

由式(6)可以看出,F 值跟 $M(t)$、p、$f_i(t)$ 相关。实际生产系统中,系统运行的代码环境相对固定,一旦系统运行镜像母本生成后,桌面系统很少会发生代码级变动。因此,$M(t)$ 在特定环境下是固定的,而不同应用环境中的 p 值是不同的。根据在实验室模拟环境中的大量测试结果表明,母本大小为 16 GB 时,$M(t)$ 值为102 s,p一般小于0.3,而$f_i(t)$根据应用场景、虚拟桌面实例个数等因素发生变化。相同应用场景下,当虚拟桌面运行实例较少,$f_i(t)$ 总和小于 $M(t)$ 时,F 大于 1,此时,由于 $M(t)$ 值影响,SVMIM 性能相对较低。当虚拟桌面运行实例较多,$f_i(t)$ 总和大于 $M(t)$ 时,$M(t)$ 值对整体性能影响变小,F 相应变小。F 为 1 时,2 种模式性能相同,达到临界值。因此,在虚拟桌面实例越多的情况下,SVMIM 性能优势越明显。

3.3　验证实验

在原型系统中,制作的虚拟桌面运行镜像母本文件大小为 16 GB,模拟单一类型的虚拟桌面系统。另外,每个虚拟桌面的"可信代码清单"项数不超过 30 000 项。

为了验证 SVMIM 方法的有效性和高效性,分别对 SVMIM 模式和非 SVMIM 模式下的虚拟桌面启动总耗时进行度量。非 SVMIM 模式硬件环境和原型系统相同,同样使用 Xen 建立虚拟桌面,但是可信代码验证在虚拟桌面操作系统中实现,并且虚拟桌面使用本地磁盘镜像。

分别采集 2 种模式下启动虚拟桌面的耗时,多次采集后进行平均计算,实验结果如图 3 所示。之所以对虚拟桌面启动时间进行度量和比较,是因为虚拟桌面在启动时加载的可执行代码较多,并且顺序相对固定,上述 2 个环境下的度量值具有可比性。

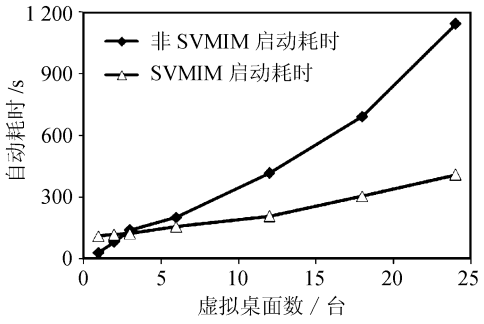

图 3　虚拟桌面启动效率比较结果

Fig. 3　Results of efficiency comparison on virtual desktop boot

根据式(4),虚拟桌面启动 F 值计算结果如表 1 所示。结合表 1、图 3 可以看出:当启动虚拟桌面小于 3 台时,F 值大于 1,SVMIM 性能较低;当启动虚拟桌面等于 3 台时,F 值小于 1,SVMIM 性能优势开始体现。实验结果符合 3.2 节性能分析结论,运行虚拟桌面实例越多,SVMIM 性能优势越明显。

表 1　虚拟桌面启动时的 F 值

Tab. 1　Values of F on virtual desktop boot

虚拟桌面数/台	非 SVMIM 启动耗时/s	SVMIM 启动耗时/s	F 值
1	32	112	3.50
2	82	118	1.43
3	140	122	0.85
6	203	157	0.77
12	416	207	0.49
18	692	305	0.44
24	1 143	409	0.36

4 结 论

提出了一种高效的虚拟桌面可信保证机制,基于该机制实现了原型系统并进行了性能分析和实验验证。SVMIM 从系统运行代码完整性方面保证虚拟桌面的运行可信,采用了一个基于混杂模式的虚拟桌面安全保证结构,既避免了 VMI 等"Out-of-the-Box"结构所固有的"语义差别"问题,又避免了传统的"In-the-Box"结构中安全机制处于一个不可信环境的困境;同时,SVMIM 也是一种高效的系统运行完整性保证机制,它通过可信虚拟机镜像母本形式避免了大部分的代码 hash 计算开销,适合虚拟机数量较大并且生产流程相对固定的应用场景;另外,SVMIM 还具有结构简单、对虚拟桌面操作系统和 VMM 改动小、系统运维简单等特点。

提出的 SVMIM 机制还存在一些不足,例如虚拟桌面中的"可信度量模块"的自封闭性要求等,将在以后的工作中进行深入研究。

参考文献:

[1] Barham P, Dragovic B, Fraser K, et al. Xen and the art ofvirtualization[J]. ACM SIGOPS Operating Systems Review, 2003, 37(5): 164 - 177.

[2] VMware, Inc. VMware[EB/OL]. [2013 - 04 - 15]. http://www.vmware.com/.

[3] Gebhardt C, Dalton C I, Brown R. Preventing hypervisor-based rootkits with trusted execution technology[J]. Network Security, 2008, 2008(11): 7 - 12.

[4] Hohmuth M, Peter M, Härtig H, et al. Reducing TCB size by using untrusted components: Small kernels versus virtual-machine monitors [C]//Proceedings of the 11th Workshop on ACM SIGOPS European Workshop. New York: ACM, 2004: 22.

[5] The Trusted Computing Group. TCG[EB/OL]. [2013 - 04 - 15]. https://www.trustedcomputinggroup.org/.

[6] Berger S, Cáceres R, Pendarakis D, et al. TVDc: Managing security in the trusted virtual data center[J]. ACM SIGOPS Operating Systems Review, 2008, 42(1): 40 - 47.

[7] Garfinkel T, Pfaff B, Chow J, et al. Terra: A virtual machine-based platform for trusted computing[C]//ACM SIGOPS Operating Systems Review. New York: ACM, 2003, 37(5): 193 - 206.

[8] Payne B D, de Carbone M D P, Lee W. Secure and flexible monitoring of virtual machines[C]//Proceedings of the 23rd Annual Computer Security Applications Conference. Miami Beach: IEEE, 2007: 385 - 397.

[9] Chen P M, Noble B D. When virtual is better than real [operating system relocation to virtual machines][C]// Proceedings of the Eighth Workshop on Hot Topics in Operating Systems, 2001. Piscataway: IEEE, 2001: 133 - 138.

[10] Payne B D, Carbone M, Sharif M, et al. Lares: An architecture for secure active monitoring using virtualization [C]//Proceedings of the 29th IEEE Symposium on Security and Privacy. Oakland: IEEE, 2008: 233 - 247.

[11] Wikimedia Foundation, Inc. Copy-on-write [EB/OL]. [2013 - 04 - 15]. http://en.wikipedia.org/wiki/Copy-on-write.

[12] Bugiel S. Using TCG/DRTM for application-specific credential storage and usage[D]. Lyngby, Denmark: Technical University of Denmark, DTU, 2010.

[13] Intel Corporation. Intel ® trusted execution technology [EB/OL]. [2013 - 04 - 15]. http://www.intel.com/technology/security/downloads/arch-overview.pdf.

(编辑 杨 蓓)