一种高效的面向虚拟桌面恶意代码检测机制

郭 煜,石 勇
(北京交通大学计算机与信息技术学院,北京 100044)

摘 要: 与传统的恶意代码检测方式相比,面向虚拟桌面的恶意代码检测方法面临着性能方面的挑战,同一物理服务器上多个虚拟桌面同时开展恶意代码检测使得磁盘等硬件成为严重的 IO 性能瓶颈.本文提出了一种高效的虚拟桌面恶意代码检测方案,基于母本克隆技术的虚拟桌面恶意代码检测机制(MCIDS),MCIDS 根据虚拟桌面系统的特点,通过系统映像网络存储克隆技术以及部署在网络存储系统中的恶意代码引擎减少虚拟桌面系统中的恶意代码检测范围,有效减少恶意代码检测所需的磁盘 IO 开销;同时 MCIDS 还克服了传统"Out-of-the-Box"安全检测机制存在的语义差别问题,改善了系统的安全性能.在原型系统上的实验显示该方法在技术上是可行的,与现有方法相比 MCIDS 具有较好的性能优势.

关键词: 恶意代码检测; 虚拟桌面; 网络存储; 存储克隆
中图分类号: TP393.08 **文献标识码:** A **文章编号:** 0372-2112 (2014)01-0119-06
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2014.01.019

An Effective Malicious Code Detection Mechanism for Virtual Desktop

GUO Yu, SHI Yong
(*School of Computer and Information, Beijing Jiaotong University, Beijing* 100044, *China*)

Abstract: Compared with traditional malicious codes detection mechanisms, detecting malicious codes in virtual desktop system faces serious performance challenge, multiple virtual desktops on a physical server conduct malicious code detection in parallel will face serious bottlenecks in the hardware devices with poor performance, such as hard disks. In this paper, we propose a malicious code detection mechanism for virtual desktop system, a malicious code detection mechanism based on mother-clone technology (MCIDS). Based on the characteristics of virtual desktops, MCIDS is designed to reduce the scope of mal-code detection by system image network storage clone technique and the mal-code detection engine deployed in network storage system, and this method effectively decreases the amount of disk IO operation caused by the detection of malicious codes, so that the performance of the system is improved greatly; In addition, MCIDS overcomes the semantic gap problem which often exists in traditional out-of-the-box security detection mechanisms. Experiments conducted on the prototype show that our method is technicallyfeasibleand the performance of MCIDS is better than other methods.

Key words: malicious code detection; virtual desktop; network storage; storage clone

1 引言

随着计算机硬件性能的提高及其成本的不断下降,虚拟化技术[1]已经开始进入成熟应用阶段,基于虚拟化和云计算的研究也广泛展开[2-6],虚拟桌面系统便是虚拟化技术的一种典型应用.虚拟桌面系统可以使一台硬件服务器能够支持几十台 Windows 虚拟桌面应用实例,不仅能有效提高了系统计算资源的利用率,而且还能帮助用户大幅降低 IT 系统方面的建设和运行成本.

由于虚拟桌面系统在结构上不同于传统的虚拟机应用,因此面向虚拟桌面的恶意代码检测机制在结构和方法上也不能简单的采用传统的虚拟机恶意代码检测机制.与一般的虚拟机应用相比,虚拟桌面系统有以下特点:首先是单台物理服务器支持的虚拟桌面数量多,一般都能达到几十个,因此恶意代码检测过程带来的磁盘性能瓶颈问题可能更为明显;其次是虚拟桌面的操

收稿日期:2013-03-07;修回日期:2013-10-29;责任编辑:郭游
基金项目:国家"973"重点基础研究发展计划 (No.2007CB307101);教育部高等学校博士学科点专项科研基金(No.20120009110007)

作系统比较单一,一般都是 Windows;其三是在生产系统的桌面应用配置比较统一.在这些特点下,如何避免磁盘性能瓶颈、减少恶意代码的检测范围和数量,是虚拟桌面系统中恶意代码检测面临的实际而又紧迫的问题.

为此,本文提出一个不同于 VMI 的虚拟桌面恶意代码检测方案,基于母本克隆技术的虚拟桌面恶意代码检测机制 Mother-clone malicious code detection system (MCIDS),该方案解决了面向虚拟桌面的恶意代码检测过程存在的磁盘 IO 性能瓶颈问题,方案利用系统映像网络存储克隆技术有效降低了恶意代码检测带来的磁盘 IO 开销,并通过将恶意代码检测引擎部署在网络存储系统中,减少了虚拟桌面系统中的恶意代码检测范围.MCIDS 有以下特征:其一是采用网络方式启动和运行虚拟桌面系统,减少磁盘 IO 数量,避免性能瓶颈;其二采用存储克隆机制支持统一桌面,有效减少恶意代码检测范围和数量,提高安全效率.与现有的面向虚拟机的恶意代码检测机制相比,MCIDS 能保证安全检测机制的可信、减少安全机制对系统性能的影响、提高虚拟桌面系统整体性能.

本文组织方式如下:第 2 节总结了国内外相关领域的研究工作进展;第 3 节描述了虚拟桌面系统的特征及基于这些特征的虚拟桌面系统网络引导结构,并对 MCIDS 的思想和结构进行了详细的描述;第 4 节简要介绍了 MCIDS 的一个原型实现方案,并完成了性能分析和实验评估;第 5 节对全文进行了总结.

2 相关工作

针对虚拟系统中的恶意软件检测问题,国内外学者进行了大量的研究,如将面向虚拟机的安全检测机制放在 VMM 中的 VMI 方法,Livewire[7]、XenAccess[8]、Lycosid[9]、Patagonix[10]和 Ether[11]等都是基于这种方法.在 VMM 中实现对虚拟机的安全检测有以下好处:首先,VMM 代码量很小,假定 VMM 可信一般是合理的;其次,VMM 的隔离机制可以保证被检测虚拟机中的恶意代码不会对 VMM 和其它虚拟机带来破坏;再次,在 VMM 中可以监视虚拟机的全部状态,包括 CPU、内存状态等,从而对虚拟机中的恶意代码进行分析判断;最后在 VMM 中可以对虚拟机的行为进行干预.但是 VMI 方法也有其局限性,即所谓的"语义差别"(Semantic Gap)问题[12].在虚拟机外部缺乏对虚拟机操作系统的底层信息的语义解释,比如内存结构及其信息含义、存储结构及其表示的数据含义,从而无法对其行为进行分析.针对"语义差别"问题,目前研究有两种主要方法:一种是语义差别转换(Semantic Gap Bridge)方法,它通过对被监视虚拟机操作系统的直接或间接知识,比如根据系统源代码或内核符号等,将在 VMM 里获取的原始底层信息进行重组和抽象,从而对被检测虚拟机的行为进行高层分析,X.Jiang 等人提出的 VMwatcher[13]采用了这一语义方法.另一种方法则不依赖于任何被检测虚拟机系统的语义信息,它通过间接或隐式的方法获取或判断被检测虚拟机的状态信息,Antfarm[14]是其中一个代表.

Lares[15]是一种混杂模式(Hybrid)的虚拟机安全检测方法,它将安全检测功能分为两个部分,其中大部分安全功能都像 VMI 方法那样置于 VMM 和受保护的虚拟机中,另一部分安全功能则以安全钩子(Hook)的形式放在被检测虚拟机的操作系统中.由于安全钩子位于操作系统内部,因此 Lares 不存在 VMI 方法的"语义差别"问题.SIM(Secure In-VM Monitoring)[16]则比 Lares 更进一步,它将虚拟机安全检测功能全部置于被检测虚拟机内部.但是,上述方案都没有充分考虑到恶意代码检测在虚拟机环境下的性能问题.

虚拟机技术是一种硬件资源(包括 CPU、内存、存储和其它设备)的共享和复用机制,同一台物理服务器中,如果某一个或某部分虚拟机对硬件资源大量占用,都会对其它虚拟机性能产生影响,尤其是数量比较稀缺或者比较容易形成性能瓶颈的那些硬件资源,比如硬盘.相对于内存和网络性能,硬盘的 IO 性能相对较低,而物理服务器中的最大硬盘数量又要受物理空间和成本限制,因此,在虚拟机环境下的恶意代码扫描过程中,系统的 IOPS(Input/Output Operations Per Second)[15]可能会成为性能瓶颈.

3 基于母本克隆技术的虚拟桌面恶意代码检测机制

3.1 基于网络引导模式的虚拟桌面系统

虚拟桌面是虚拟机技术的一种应用.虚拟桌面系统以硬件虚拟化为基础,在 VMM 的支持下,在同一物理服务器平台上安装运行多个个人桌面系统的虚拟机实例.用户在远程操作终端通过远程桌面协议连接这些虚拟机实例,如同操作本地操作系统一样.虚拟桌面可以提高物理硬件资源的利用率,但这种硬件资源共享的方式也有潜在的性能问题,尤其是磁盘的 IO 瓶颈问题.与一般的虚拟机应用相比,虚拟桌面系统有以下特点:首先,由于一般的桌面应用占用的硬件资源较少,因此单台物理服务器可能支持的虚拟桌面(或虚拟机)数量比较大;其次,这些虚拟桌面一般都采用相同的桌面操作系统,即操作系统同构性特征,如 Windows XP 或 Windows 7 等;其三,在生产系统中,同种工作性质的桌面系统所安装的应用都基本一致,即软件配置标准化特征.因此,基于这些特点,我们提出了采用网络引导和存储模式的虚拟桌面系统结构,如图 1 所示.

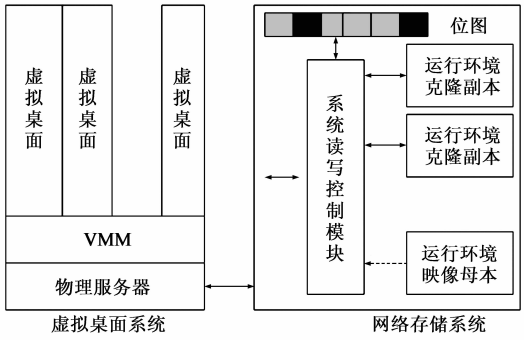

图1　基于网络引导模式的虚拟桌面系统结构

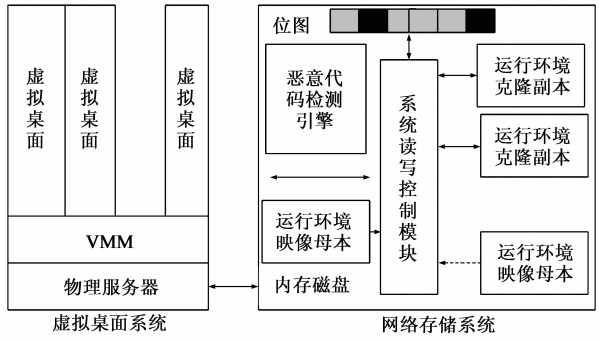

图2　面向虚拟桌面系统的恶意代码检测机制

在基于网络引导模式的虚拟桌面系统中,虚拟桌面的运行环境映像文件(包括操作系统和应用程序)都不再存放在虚拟桌面所在的物理服务器上,而是集中存放在网络存储设备(如 iSCSI 存储服务端)中,虚拟桌面通过网络远程引导这些系统映像并在本地运行.

由于生产环境中的虚拟桌面具有操作系统同构性特征和软件配置标准化特征,因此在网络存储系统服务端建立虚拟桌面运行环境映像母本,并基于这一母本为每个虚拟桌面建立独立的运行环境克隆副本.母本和克隆副本之间通过 CoW(Copy-on-Write)[17] 策略实现关联,即对每个虚拟桌面,只有当其环境发生变化时才把变化的部分写入到该虚拟桌面对应的克隆副本中,并在此刻以后,虚拟桌面都从其克隆副本中读取被修改过内容,否则虚拟桌面将从母本中读取内容.CoW 策略可以通过位图模式实现.网络存储系统为每个克隆副本建立一个位图结构,位图结构中的每一位对应克隆副本的一个数据块,系统读写控制模块通过位图控制系统的数据读写方式.如果某一个数据块被修改,那么系统读写控制模块将该数据块在位图中对应的位进行设置,并将被修改部分写入对应的克隆副本中;系统在读一个数据块时,系统读写控制模块首先检查该数据块在位图中的对应位,如果没有被设置,系统从母本读取对应的数据块,否则,系统从克隆副本中读取相应数据块.因此,在运行模式下系统对母本只读不写,除非通过专门方式升级或修改母本,母本不可能被外部篡改.因此只需确保母本的升级或修改过程安全可控,那么母本的完整性在系统运行中就不会被恶意破坏.

为了进一步提高系统的读写性能,我们还充分利用了虚拟服务器和网络存储具有的缓存写机制.由于系统写缓存机制的存在,虚拟桌面对母本或克隆副本的读写在很大概率上都是对虚拟服务器和网络存储上的相应缓存区域的操作,因此可以极大地减少虚拟桌面系统对磁盘的读写操作数量,从而避免磁盘 IO 成为系统瓶颈.

3.2　MCIDS 方法

基于图 1 的虚拟桌面系统结构,我们提出一个高效的虚拟桌面系统的恶意代码检测方法,基于母本克隆技术的虚拟桌面恶意代码检测机制(MCIDS).MCIDS 主要解决面向虚拟桌面的恶意代码检测过程可能存在的磁盘 IO 性能瓶颈问题,对于采用内存扫描的恶意代码检测过程不做考虑.

MCIDS 的核心思想是,网络存储系统作为虚拟桌面的运行存储环境,将恶意代码检测引擎放在网络存储系统中,并通过系统读写控制模块对系统读写内容进行检测,如图 2 所示.由于系统对母本只读不写,在母本被安全维护的条件下,其内容被认为是可信的,因此系统只需对克隆副本的读写内容进行检测,这样可以大幅减少系统的恶意代码检测范围.此外,为最大程度地降低 MCIDS 对系统磁盘 IO 性能的影响,MCIDS 在网络存储系统中通过 ramdisk 技术为母本建立专门的内存磁盘,系统启动后,MCIDS 将母本复制到内存磁盘中,此后系统对母本的读写全部是内存操作,不需要再访问物理磁盘.

网络存储系统中的系统读写控制模块对来自各虚拟桌面的读写请求进行解析,确定其读写方式及读写内容,并按照虚拟桌面对应的位图结构确定数据的实际来源或去向.如果从磁盘中读取数据,并且根据位图结构判定是未经修改过的数据块,那么系统读写控制模块将从母本中读取,并直接将数据返回虚拟桌面,而不会将读取内容提交到恶意代码测引擎,否则,系统将在虚拟桌面对应的克隆副本上进行实际读写操作.如果是从克隆副本读取数据,系统读写控制模块首先将读取的内容提交给恶意代码检测引擎进行恶意代码检测,确认没有恶意代码后再将数据返回给相应的虚拟桌面;如果是写数据,系统读写控制模块将要写的数据提交给恶意代码检测引擎进行恶意代码检测,确认没有恶意代码后再进行实际的写操作.

MCIDS 在系统启动后,将母本复制到为它建立的专门内存磁盘中,此后系统对母本的访问就完全基于该

内存磁盘中的母本映像.为母本建立专门的内存磁盘是合理的,在技术上也是可行的.首先,母本数量少,并且相对稳定;其次,母本具有只读不写的特性,它在整个系统运行期间内容和大小都不会发生变化,即使系统出现故障或意外关机,母本不存在缓存延迟写可能产生的问题.

与 VMI 方法和传统的"In-the-Box"方法相比,MCIDS 有如下特点:(1)不会被恶意代码规避或破坏.MCIDS 跟被检测的虚拟桌面不在同一个操作系统环境中,甚至不在一个物理服务器中,因此不会被恶意代码发现或破坏;另外,虚拟桌面所有的磁盘文件读写都经过 MCIDS 的"系统读写控制模块",所有非可信数据都需要经过检测引擎的分析;(2)不需要修改 VMM.恶意代码检测引擎运行在网络存储系统中,不涉及任何 VMM 层的修改;(3)恶意代码检测引擎独立于 VMM,可以支持不同类型的虚拟化软件;(4)高性能.

当前的 MCIDS 是为磁盘文件的恶意代码检测设计的,但是经过扩展,该方法也可支持内存恶意代码检测.对内存进行恶意代码检测需要 VMM 功能支持,比如 XenAccess 等,对虚拟桌面内存内容进行采集或内审,并报告给恶意代码扫描引擎;或采用 Ether[11] 等方法对虚拟桌面的系统调用执行、指令执行、内存写及环境切换等事件进行安全检测.

4 实验评估

4.1 原型实现

MCIDS 原型系统以开源系统为基础:虚拟桌面系统的 VMM 采用了 Xen,网络存储系统采用了 iSCSI Enterprise Target(IET),桌面操作系统采用了 Windows XP.原型系统中通过修改 IET 对数据的存取过程,实现了基于位图结构的存储母本克隆方式.在实验评估中,根据一般生产系统的工作需求,我们制作了大小为 16GB 和 32GB 的虚拟桌面母本映像文件.

由于我们的实验侧重 MCIDS 的功能验证和性能评估,而不是对恶意代码的实际检测效果和能力的验证,因此实验测试选用的恶意代码特征库数量小于 1000.原型系统中的恶意代码检测引擎采用了最为常见的恶意代码特征匹配方式.

在实验环境中,虚拟服务器的配置为双 Xeon X5650 CPU,内存大小为 64G,硬盘为 240G 固态硬盘,千兆以太网卡,做多同时运行 40 个虚拟桌面;网络存储服务器为单核 Xeon X5650 CPU,内存大小为 32G,硬盘为 6 * 240G 固态硬盘、千兆以太网卡.

4.2 性能分析

在一个生产系统的虚拟桌面中,系统文件和相同的应用软件占了较大比例,所以采用母本克隆技术后,大部分的内容都是存放在母本中的,克隆副本中存放的数据很少,MCIDS 可以减少大量的磁盘扫描检测量,从而提高效率.为了准确地评估 MCIDS 对系统性能的影响,我们对虚拟桌面恶意代码检测的典型场景进行了性能分析,系统运行过程中的实时动态检测,即系统在读写文件过程中的实时检测.

首先,我们定义相关概念和符号如下:

(1)假设 $V=\{v_1,v_2,\cdots,v_n\}$ 为一台物理服务器中 OS 同构的虚拟桌面集合,$v_1,v_2,\cdots,v_n$ 为实际的虚拟桌面;

(2)假设所有虚拟桌面都拥有相同的初始运行环境,这一初始运行环境就是系统运行环境映像母本,我们用 M 表示母本在内存磁盘中的存储块集合,$S(M)$ 表示母本大小;

(3)虚拟桌面运行一段时间后,其运行环境都可能发生变化,这些变化的部分都会被写到虚拟桌面对应的克隆副本中,我们用 C_i 表示虚拟桌面 v_i 的克隆副本在网络存储系统中的物理存储块集合,D_i 表示 C_i 在 M 中对应的内存磁盘块集合,这里 $1\leqslant i\leqslant n$;显然 D_i 和 C_i 大小相同,即 $S(D_i)=S(C_i)$;

(4)对于 V 中的任何一个虚拟机 v_i,$1\leqslant i\leqslant n$,其完整运行环境可以表示为 $E_i=C_i\cup(M-D_i)$;

(5)根据假设,$M\text{-}D_i$ 是虚拟桌面 v_i 初始运行环境未发生变化的部分,其中 $1\leqslant i\leqslant n$,MCIDS 认为这部分是安全可信的,因此不需要对其进行恶意代码检测;MCIDS 只需要对 C_i 部分进行检测.

4.2.1 实时动态检测

假定位于同一台物理服务器上的虚拟桌面 $v_1,v_2,\cdots,v_n$ 的磁盘访问操作分别服从强度为 $\lambda_1,\lambda_2,\cdots,\lambda_n$ 的泊松过程,且相互独立.如果对于每一个虚拟桌面 v_i,$1\leqslant i\leqslant n$,在它访问操作对象 E_i 的全部读写操作中,访问 C_i 的概率为 p_i,访问 $(M\text{-}D_i)$ 的概率为 $1-p_i$,其中 $0\leqslant p_i1\leqslant$,那么根据泊松过程的可分解性性质,v_i 读写 C_i 的过程是一个强度为 $p_i\lambda_i$ 的泊松过程,也就是说,MCIDS "系统读写控制模块"接收的所有读写请求中,对非母本的读写 $Z(t)$ 是一个服从强度为 $\sum_{i=1}^{n}p_i\lambda_i$ 的泊松过程.

所有假定每次读写请求的物理块数 B_j 是一列独立同分布的随机变量,且 $E(B_j)=\mu$,那么 $M(t)=\sum_{j=1}^{Z(t)}B_j$ 是一个复合泊松过程,且 $M(t)$ 的数学期望为 $E(M(t))=E\left(\sum_{j=1}^{Z(t)}B_j\right)=t\mu\sum_{i=1}^{n}p_i\lambda_i$.

如果在一个非 CIDS 环境中,系统的总体磁盘读写

$Y(t)$是一个服从强度为 $\sum_{i=1}^{n}\lambda_i$ 的泊松过程.并且,假定每次读写请求的物理块数与 CIDS 环境一样,也为 B_j,且 $E(B_j)=\mu$,那么 $N(t)=\sum_{j=1}^{Y(t)}B_j$ 也是一个复合泊松过程,且 $N(t)$的数学期望为 $E(N(t))=E\left(\sum_{j=1}^{Y(t)}B_j\right)=t\mu\sum_{i=1}^{n}\lambda_i$.

由于 $0\leqslant pi\leqslant 1$,所以 $E(M(t))\leqslant E(N(t))$,这一不等式表明了 CIDS 比传统的恶意代码检测方式需要更少的磁盘 IO 操作,我们用 $F_r=(E(M(t))/E(N(t)))\times 100\%$ 表示 CIDS 的实时运行磁盘 IO 操作评价指标.在同类应用系统中,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 $p_i=p$,$0\leqslant p\leqslant 1$,$1\leqslant i\leqslant n$,此时,我们可以计算出 $F_r=p\times 100\%$.

如图 3 所示,不同应用环境中的 p 值是不同的,但是根据我们在实验室环境中的大量测试结果表明,p 一般小于 0.2,因此可见,与传统的恶意代码检测方式相比,MCIDS 方法大大减少了检测过程所需的磁盘 IO 操作,从而提升了系统的整体性能.

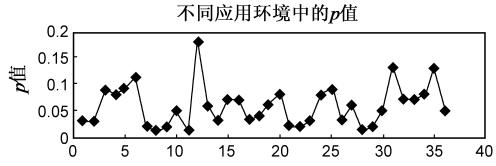

图3　不同应用环境中p的实测值

4.2.2　检测时间对比分析

为了能够更直观地对比采用传统的单机直接检测的方式和采用本文提出的 MCIDS 检测方式,在多个虚拟桌面系统同时运行的情况下开展恶意代码检测的效率,我们在实验系统上做了两种检测机制的时间对比实验.我们分别在 1、2、3、5、8 个虚拟桌面同时运行的情况下,利用 MCIDS 的母本-克隆机制,在磁盘存储系统中进行恶意代码检测,并记录了检测过程所需的平均时间.作为对比,在同样的实验条件下,我们采用分别在每个虚拟桌面中采用传统的恶意代码单机直接检测方式,统计了每个虚拟桌面完成检测所需的平均检测时间,结果如图 4 所示.

从图 4 可以看出,在多个虚拟桌面同时进行恶意代码检测时,由于每个虚拟桌面都要读取自身的磁盘映像文件,导致承载虚拟桌面的底层物理服务器需要大量的 IO 操作,这使得 IO 操作成为整个系统的性能瓶颈,对所有物理服务器上的虚拟桌面都造成了影响,导致性能大幅度下降,检测时间与同时开展检测的虚拟桌面数量呈指数增长的关系.而采用 MCIDS 检测方式,由于检测过程在存储系统上,并且母本克隆机制大幅减少了冗余的磁盘文件,对多个虚拟桌面同时杀毒性能有很大的提高,检测时间基本与同时开展检测的虚拟桌面数无关.

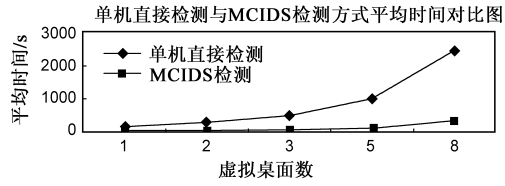

图4　单机直接检测与MCIDS检测方式平均时间对比图

5　总结

针对虚拟桌面系统缺乏高效的恶意代码检测技术的问题,我们提出了一个高效的虚拟桌面的恶意代码检测机制,基于母本克隆技术的虚拟桌面恶意代码检测机制(MCIDS),它适合生产系统中具有同构操作系统并且数量较大的虚拟桌面应用场景.MCIDS 结构简单,无需对桌面操作系统和 VMM 进行修改;MCIDS 避免了传统恶意代码检测机制在检测分析过程中对磁盘的大量读写操作,对系统的性能影响较小.

下一步,我们将对 MCIDS 方法进行扩展,使之能够支持对基于内存的恶意代码进行检测,提高 MCIDS 的可用性及其实用价值.

参考文献

[1] P Barham, B Dragovic, et al. Xen and the art of virtualization [A]. Proceedings of the Nineteenth ACM Symposium on Operationg Systems Principles [C]. NY, USA: ACM, 2003. 164 – 177.

[2] 俞能海,郝卓,徐甲甲,张卫明,张驰.云安全研究进展综述[J].电子学报,2013,41 (2):371 – 381.
YU Neng-hai, HAO Zhuo, XU Jia-jia, ZHANG Wei-ming, ZHANG Chi. Review of cloud computing security [J]. Acta Electronica Sinica, 2013, 41 (2): 371 – 381. (in Chinese)

[3] 李建敦,彭俊杰,张武.云存储中一种基于布局的虚拟磁盘节能调度方法[J].电子学报,2012,40 (11):2247 – 2254.
LI Jian-dun, PENG Jun-jie, ZHANG Wu. A layout-based energy-aware approach for virtual disk scheduling in cloud storage [J]. Acta Electronica Sinica, 2012, 40 (11): 2247 – 2254. (in Chinese)

[4] Alarifi S S, Wolthusen S D. Detecting anomalies in IaaS environments through virtual machine host system call analysis [A]. Internet Technology And Secured Transactions, International Conference for IEEE [C]. USA: IEEE, 2012. 211 – 218.

[5] Ibrahim A S, Hamlyn-Harris J, Grundy J, Almorsy M. DIGGER: Identifying OS Kernel Objects for Run-time Security Analysis [OL]. http://www.ict.swin.edu.au/personal/aibrahim/Pubs/NSS2012-61.pdf, 2012.

[6] Chiueh T, Conover M, Montague B. Surreptitious deployment and execution of kernel agents in windows guests[A]. Proceedings of the 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)[C]. USA: IEEE Computer Society, 2012.507-514.

[7] T Garfinkel, M Rosenblum. A virtual machine introspection based architecture for intrusion detection[A]. Proc of the 2003 Network and Distributed System Security Symposium[C]. USA, 2003.196-206.

[8] S T Jones, A CArpaci-Dusseau, R H Arpaci-Dusseau. Antfarm: Tracking processes in a virtual machine environment[A]. Proceedings of the USENIX Annual Technical Conference[C]. Boston, MA, USA, 2006.1-14.

[9] Stephen T. Jones, Andrea C Arpaci-Dusseau, Remzi H Arpaci-Dusseau. VMM-based hidden process detection and identification using lycosid[A]. Proceedings of the Fourth ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments[C]. Seattle, WA, USA, 2008.91-100.

[10] LLitty, H A Lagar-Cavilla, D Lie. Hypervisor support for identifying covertly executing binaries[A]. Proceedings of the 17th Conference on Security Symposium[C]. CA, USA, 2008.243-258.

[11] A Dinaburg, P Royal, M Sharif, W Lee. Ether: Malware analysis via hardware virtualization extensions[A]. Proceedings of the 15th ACM Conference on Computer and Communications Security (CCS)[C]. Alexandria, VA, 2008.51-62.

[12] Peter M Chen, Brian D Noble. When virtual is better than real [A]. In 8th Workshop on Hot Topics in Operating Systems, HotOS VIII[C]. SchossElmau, Germany, 2001.133-134.

[13] X Jiang, DXu, X Wang. Stealthy malware detection through VMM-based "out-of-the-box" semantic view reconstruction [A]. Proceedings of the ACM Conference on Computer and Communications Security[C]. Alexandria, VA, USA, 2007. 128-138.

[14] Bryan D Payne, Martim Carbone, Wenke Lee. Secure and flexible monitoring of virtual machines[A]. Proceedings of the Annual Computer Security Applications Conference[C]. Miami Beach, Florida, USA, 2007.385-397.

[15] BD Payne, M Carbone, M Sharif, W Lee. Lares: an architecturefor secure active monitoring using virtualization[A]. In SP '08: Proc of the 2008 IEEE Symposium on Security and Privacy[C]. Washington DC, USA, 2008.233-247.

[16] M Sharif, W Lee, W Cui, A Lanzi. Secure in-VMMonitoring using hardware virtualization[A]. Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS '09)[C]. Hyatt Regency Chicago, Chicago, IL, USA, 2009.477-487.

[17] IOPS[OL]. http://en.wikipedia.org/wiki/IOPS.

[18] Copy-on-write[OL]. http://en.wikipedia.org/wiki/Copy-on-write.

作者简介

郭 煜 男, 1982 年 5 月出生于陕西汉中. 2008 年获北京交通大学工学硕士学位, 现为在读博士生, 从事云计算、可信计算方面研究.
E-mail: gy-u@163.com

石 勇 男, 1982 年 12 月出生于湖南益阳. 2008 年获北京交通大学工学硕士学位, 现为在读博士生, 从事云计算、可信计算方面研究.
E-mail: stonefly@126.com