文章编号:1009-3087(2014)06-0007-07

基于 KVM 的 Windows 客户机进程查杀技术

崔竞松[1,2],向 浩[1],郭 迟[3*],张雅娜[1],何 松[1]
(1. 武汉大学 计算机学院,湖北 武汉 430072;2. 武汉大学 空天信息安全与可信计算教育部重点实验室,湖北 武汉 430072;
3. 武汉大学 卫星定位导航技术研究中心,湖北 武汉 430079)

摘 要:传统反病毒架构不能有效利用虚拟化优势解决云平台上的 Windows 系统所面临的恶意软件威胁,并且传统反病毒软件自身面临安全威胁,针对此问题,提出一种基于 KVM 的无代理 Windows 客户机进程在线杀毒技术。通过在 KVM 内核模块中添加读写内存的函数,以及为进程处理模块提供在其中注册钩子的接口等方法,解析客户机当前进程信息。将进程在内存中的 PE(portable executable)镜像大致还原成运行前的磁盘文件后,调用开源杀毒引擎 ClamAV(Clam AntiVirus)进行扫描查毒。查毒结果返回给决策模块后,由进程处理内核模块对可疑进程进行相应处理,实现对客户机当前进程的无代理查杀。分析及测试结果表明,该技术利用虚拟化优势较好地解决了传统反病毒框架的资源耗费和自身安全性问题。

关键词:KVM 虚拟化;虚拟化安全;无代理方式;进程监控;PE 镜像还原;进程终止
中图分类号:TP302.1 **文献标志码**:A

Online Anti-virus Technology of Processes Running on Windows VM Based on KVM

CUI Jingsong[1,2], *XIANG Hao*[1], *GUO Chi*[3*], *ZHANG Yana*[1], *HE Song*[1]
(1. Computer School, Wuhan Univ., Wuhan 430072, China;
2. Key Lab. of Aerospace Info. and Trusted Computing, Wuhan Univ., Wuhan 430072, China;
3. Global Navigation Satellite System Research Center, Wuhan Univ., Wuhan 430079, China)

Abstract: Aiming at the problem that the traditional anti-virus structure cannot effectively solve malware threats on Windows OS on virtualization platform by using the benefits of virtualization, and traditional anti-virus softwares have to face their own security threats, an agentless online anti-virus technology of processes running on Windows VM based on KVM was proposed. By adding memory reading and writing functions in KVM kernel module and providing interfaces to register hooks in the kernel module of processes handling, the VM's processes' information could be resolved. After restoring process's PE image in memory into disk file before running, the open source antivirus engine ClamAV would be called to scan virus. When results returned to the decision-making module, process handling module would deal with suspicious processes accordingly, and the current process could be scanned and killed without any agent. Analysis and test results showed that the technique could solve the traditional anti-virus frameworks' resource consumption and security issues by taking advantage of virtualization's benefits.

Key words: KVM; virtualization security; agentless technique; process monitoring; PE image reduction; process killing

目前,虚拟化技术(virtualization)[1]在企业中得到了广泛的应用,并具有巨大的市场前景。然而,随着虚拟化平台大规模上线的使用,虚拟机也成为广大网络黑客攻击的对象。在虚拟化面临的安全问题[2]中,其中之一是以主机为基础的安全策略难以部署。目前基于虚拟机的安全软件都是基于物理机开发的,其防护方式无一例外都是借助传统方式,而且如果在每一台虚拟机上都安装安全软件,对物理

收稿日期:2014-06-23
基金项目:国家高技术研究发展计划资助项目(2013AA12A206);国家自然科学基金资助项目(41104010;91120002;61170026)
作者简介:崔竞松(1975—),男,副教授,博士. 研究方向:信息安全;云安全;网络验证码. E-mail:cuijs1@gmail.com
* 通信联系人 E-mail:guochi@whu.edu.cn
网络出版时间:2014-9-3 14:32:33 **网络出版地址**:http://www.cnki.net/kcms/detail/51.1596.T.20140903.1432.001.html

服务器的存储空间、内存资源占用较大。并且在虚拟机关闭期间,病毒代码是无法更新的,一旦开机,多个防病毒软件同时更新一个病毒码对网络带宽也有较大影响。网络安全设备目前也没有监测虚机之间通信流的能力,先进的虚拟平台搭配传统的防范策略,无疑影响了虚拟平台的使用,网络黑客和内部攻击可以利用这个时期大规模攻击虚拟机,并借助单台虚拟机攻击虚机群,业务系统随时崩溃。

针对上述问题,提出一种部署在主机上基于KVM虚拟化技术的无代理式[5] Windows客户机进程在线查杀系统KVMPST(KVM process scanner and terminator)。该系统以主机为单位进行保护管理,在虚拟机外部实现所有虚拟机进程的实时监控和查杀。值得注意的是,这种无客户端、基于虚拟设备的病毒防护方法也适用于其他虚拟机监控器(virtual machine monitor, VMM),并且与不具有虚拟化感知能力的实施方案(防病毒客户端处理于每台虚拟机中)相比,所消耗的CPU、内存和磁盘I/O资源大幅减少。

1 相关工作

就目前市场上的虚拟化安全应用看来,主要分为2大阵营:一是基于VMware为首的闭源虚拟化平台上的虚拟化安全应用,二是基于KVM和Xen等开源虚拟化平台上的虚拟化安全应用。凭借雄厚实力Vmware与如Symantec[6]、McAfee[7]、Trendmicro[8]、Kaspersky[9]等著名的安全公司进行合作,并且推出一系列用于Vmware平台上的虚拟化安全产品。在KVM和Xen的开源阵营,特别是国内,例如阿里云[10]、盛大云[11]、华为云[12]、腾讯云[13]等等,针对各自用户群的特点也提出了一些虚拟化安全服务产品。

虚拟化安全方面的应用跟具体虚拟化平台是否开源,呈现出的是不同的发展形势:以Vmware为主的闭源虚拟化平台大多与安全方面的合作伙伴共同研发,起步比开源的对手早,形成了基于各自平台的一套安全体系,允许合作的产品在之上运行,其安全应用的有效性和自身安全性这也就依赖于闭源虚拟化平台所提供的支持。以KVM和Xen为主的开源虚拟化平台发展势头迅猛,特别地国内云服务提供商基本都是基于这两种平台进行的开发,但是相应的安全应用主要集中在常规的服务器监控,没有针对虚拟化环境的安全需求,从而使得这些安全应用的有效性和自身安全都会受到影响。

这些各式各样的虚拟化安全服务,又可以按照有无代理插件来划分为2类:第1类是以Symantec和国内云服务提供商为主的在客户虚拟机中安装代理插件的传统方法。这些代理插件的功能包括对客户虚拟机状态和事件信息的采集,传送给安全分析组件(大多存在于客户虚拟机外部),以及接收安全分析组件传来的命令并执行一系列安全操作。运用Agent手段虽然对于传统的物理机和虚拟机都有较好的兼容性,在集中管理和配置方面比较出色,但是并没有利用虚拟化环境中的特权与隔离的优点,从而对于新兴的恶意程序与漏洞处理起来显得有些乏力。第2类是以McAfee和Kaspersky为主的无代理插件的方式。这类安全组件与需要保护的虚拟机之间的访问通信,都是通过在虚拟机底层通过驱动程序传递消息和控制命令。所有的虚拟化安全应用都能被分为有代理插件式的和无代理式。结合上述产品的分析,得出无代理插件是虚拟化环境中安全应用的发展趋势,因此本研究也从这条路线出发。

2 KVMPST架构设计

2.1 KVMPST系统架构

KVMPST的系统架构图如图1所示。

2.2 系统的工作流程

该系统部署在物理主机中,在内核空间中有2个模块:修改过的KVM内核模块(modified KVM-kmod)负责读写GuestOS内存,进程处理内核模块(process_handle module)负责分析和处理GuestOS当前进程。在用户空间有4个模块:进程监视模块(process monitoring)负责实时显示GuestOS当前进程列表,进程扫描模块(process scanning)负责检测GuestOS当前进程是否可疑,进程决策模块(process deciding)负责决策GuestOS当前进程终止与否,VNC消息推送模块(VNC server)负责与GusetOS用户通信。

当部署有KVMPST系统的主机里面客户虚拟机中的程序执行,访问操作系统的资源引起底层虚拟硬件的状态发生变化,触发了修改过的KVM模块中的钩子,控制流陷入KVM模块中,随即进入注册了相应事件回调函数的内核进程处理模块中进行处理。在内核进程处理模块中,首先对当前客户虚拟机中运行的进程进行分析,然后查表判断是否为新进程,如果在3张表(可疑表、终止表、安全表)中均无记录,则判断是新进程,通过netlink来进行内核空间与用户空间的信息交互,将所得信息送入用

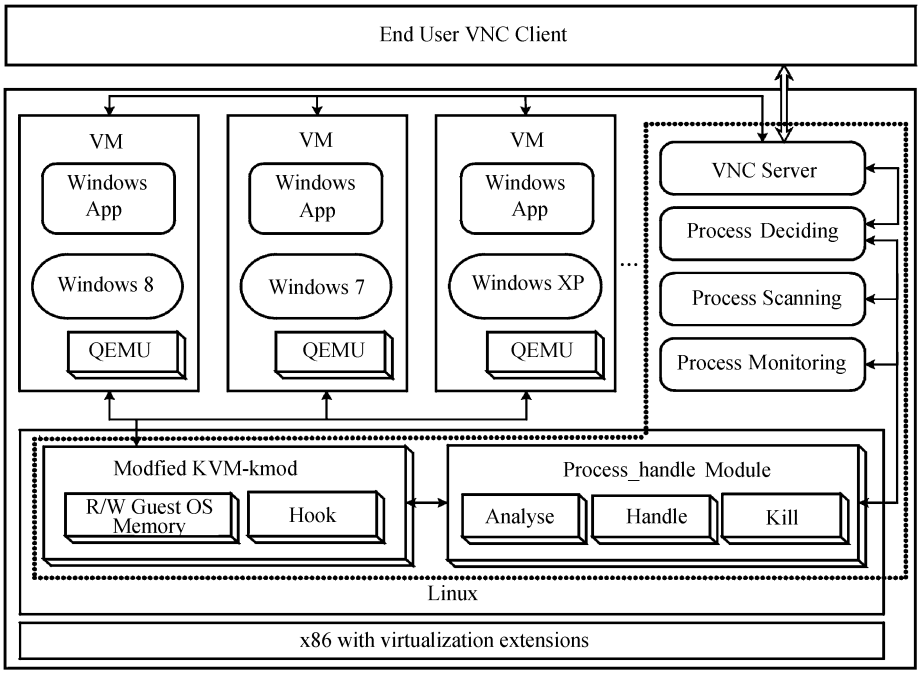

图 1　KVMPST 的系统架构

Fig. 1　Architecture of KVMPST system

户空间的安全扫描模块进行扫描,得出扫描结果返回给内核处理模块,将当前进程根据是否可疑加入安全表或可疑表,流程进入下一次循环。如果查表所得当前进程在可疑表中,则将进程信息发送到用户空间的进程决策模块,根据与用户交互反馈的结果,决定加入安全表或者终止表,流程进入下一次循环。如果查表所得当前进程在终止表中,内核处理模块将进行具体干预操作。若当前进程在安全表中,则不处理。从而达到对客户虚拟机中的进程的实时查杀。系统工作流程如图 2 所示。

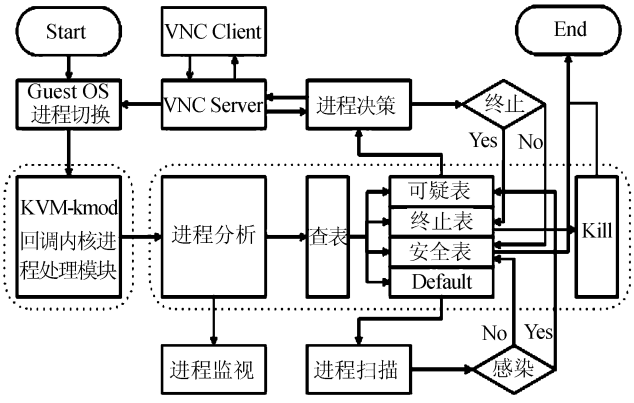

图 2　KVMPST 系统的工作流程

Fig. 2　Working flow of KVMPST system

3　KVMPST 架构实现

3.1　修改 KVM 内核模块

该模块运行在宿主机的 Ring0 层,在原 KVM 虚拟化管理器的基础上进行修改,导出了 KVM 内核模块中的 2 个原有的函数 kvm_read_guest_virt_system 和 kvm_write_guest_virt_system,用于在进程处理他内核模块中,读写虚拟机的虚拟内存;增加 2 个导出函数 kvm_vm_register_read 和 kvm_vm_register_write,用于读写虚拟机的寄存器。从而实现实现获取虚拟机系统信息、透视读写内存的功能。

另外,在 KVM 内核模块添加导出函数 kvm_register_vm_av_module,用于向 KVM 注册进程处理内核模块,其函数原型为:unsigned long kvm_register_vm_av_module(unsigned long vm_op, unsigned long longvm_event);其中,第 1 个参数 vm_op 是回调函数指针,允许进程处理内核模块在特定 VM EXIT 注册钩子,以实现进程信息的获取。回调函数原型为:int av_handle_vm_exit(struct kvm_vcpu * vcpu);回调参数为一个 kvm_vcpu 结构体指针 vcpu,代表发生 VM Exit 事件的虚拟 CPU。其中保存有 V-CPU 对应的虚拟机和发生 VM Exit 事件的原因。第 2 个参数 vm_event 是一个 64 位无符号整数,定义了回调函数的感兴趣事件。规定:第 n 位为 1 时,代表回调函数要处理 Intel 指导手册[14] 中对应的第 n 个 VM Exit 事件。少量未定义的位在 KVM 中并没有对应的处理函数,这些 VM Exit 事件并不需要被处理。注册的事件为控制寄存器 CR3 切换。

3.2　进程处理内核模块

该模块运行在宿主机的 Ring0 层,被 KVM 内核

模块里的 HOOK 函数回调后,通过调用 KVM 内核模块里的虚拟机内存读写函数接口,获得虚拟机的进程信息,实现进程分析,处理和终止功能:

1)进程分析。由于所采用的是无代理插件的方式,因此这些状态信息与事件动作通常是以一些原始数据的形式呈现,例如虚拟机的 CPU 状态信息,内存中的原始数据等。现代操作系统采用了复杂的数据结构,在内存中保存系统的关键数据。如果不能正确的解析内存数据代表的含义,即使获得内存数据,仍然无法正确的判断和安全的清除病毒。该系统没有在 Guest OS 内部安装任何辅助软件,而是在 VMM 层根据 Guest OS 的特性和 x86 架构的特点,重新构建 Guest OS 内核对象。

Windows 操作系统用户众多,稳定性好,因此我们选用 Windows 7 系统为例,说明如何在 VMM 层还原系统内核对象,寻找 Window 7 内核示意图如图 3 所示。

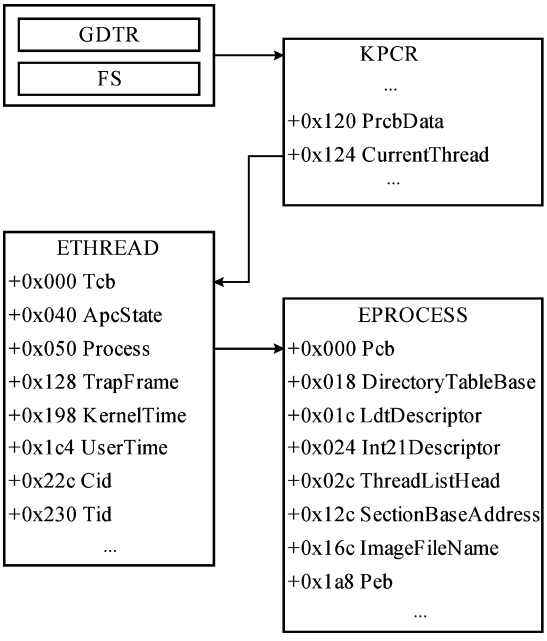

图 3 寻找 Window 7 内核示意图

Fig. 3 Sketch map of searching kernel of Windows 7

在 Windows GuestOS 中,当进程运行在内核级别(Ring0)时, FS 寄存器里的地址指向全局描述符表(global descriptor table, GDT)中某一段,该段存储的地址经过转换后指向系统的处理器控制区域(processor control region, KPCR),当前线程指针 CurrentThread 保存在 KPCR 偏移 0x124 的位置。在这个 KTHREAD 结构体偏移 0x50 的位置保存有当前线程所属进程的 KPROCESS 结构指针,此 KPROCESS 指针等于当前进程 EPROCESS 结构体指针。通过 EPROCESS 结构体可以获得当前进程的所有信息,主要包括:镜像基地址(SectionBaseAddress),所执行程序镜像的文件名(ImageFileName),用户空间进程环境块(Peb)和指向代表本进程用户空间内存分配的数据结构(VadRoot)。在 KPROCESS 结构体中保存有:本进程页表的物理地址(DirectoryTableBase),本进程的线程队列(ThreadListHead)。用户空间的 PEB 结构体中保存有:程序镜像起点(ImageBaseAddress),进程参数块(ProcessParameters,其中保存了当前 CURDIR CurrentDIrectory 当前文件目录)。结合镜像文件名,即可完整的获得进程所执行的程序在硬盘上的位置。由此即可以成功获得当前线程和进程的信息。

当 CPU 运行在用户态的时候,FS 寄存器指针指向当前运行的线程的 TEB 结构体,通过 TEB 即可获得该线程所属的进程的位于用户空间的 PEB 结构体。这样同样可以获得当前线程以及所属进程的所有信息。上面以获得进程信息为例,展示了通过直接内核对象的读取,可以直接获得 Windows Guest OS 的各种内核对象,而不会受到虚拟机内部运行的进程的干扰,保证了数据的正确性和完整性。同时做到了完全隐蔽,虚拟机内部的进程无法感知到这一过程的进行。

2)进程处理。在进程处理内核模块中维护着 3 张表:安全表(security table)存储经过进程扫描模块确认为安全进程的进程队列;可疑表(suspicion table)存储经过进程扫描模块确认为可疑进程的进程队列,这个列表大部分情况下经过后续处理后为空;终止表(terminate table)存储经过进程决策模块确认为需要终止的进程队列。在分析完 GuestOS 当前进程后,根据查询当前进程在这 3 张表中的结果,做出相应的处理操作。

3)进程终止。在某个进程在调度时被监测到,并且通过对其相关数据的分析得知这是一个恶意进程,本组件应该能对其采取控制措施。通常在终止操作系统中的某个进程的方法有多种,在不安装代理插件的情况下,通过 VMM 给目标进程发送终止的信号或者修改其指令流的方式可以实现。对于第一种发送信号的方式,只须在相应进程的进程描述符中的相关信号位置位,在该进程被调度时便会被终止。而另外一种简单有效的方法就是在虚拟机陷入 VMM 时,将虚拟寄存器 EIP 的值修改为一个非法值,或者将内存清零的方式[15]就能达到终止当前的虚拟机中的进程。但是这种方法过于粗暴,可能

导致整个虚拟机都无法继续运行。作者采用一种更为安全的方法,通过修改当前进程的程序段代码,改为一个预定的程序入口地址,后者指向的是一段自己调用终止函数的程序,例如_exit_thread 或者 ExitProcess 函数,从而在当前进程继续执行时便会终止掉自己。

3.3　进程扫描模块

ClamAV[16] 是 Unix 下基于 GPL Licence 的一款开源杀毒软件,被广泛应用于多种网络安全领域,尤其在防范网络蠕虫,电脑病毒和木马等攻击的网络入侵检测系统(NIDS)中被着重使用,是构成这些 NIDS 的主要部件之一。ClamAV 在运行中主要使用了入侵检测技术中的特征值检验技术。该工具包提供了包含灵活且可伸缩的监控程序、命令行扫描程序以及用于自动更新数据库的高级工具在内的大量实用程序,其核心在于其中可被用于各类场合的反病毒引擎共享库。

因为 ClamAV 只能扫描文件,要对内存中的进程数据进行查杀,还需要对其系统进行改进以增强系统对于进程的查杀能力。这里用到就是 PE[17] 镜像还原的技术。

在 Windows 系统下,当一个 PE 应用程序运行时,这个 PE 文件在磁盘中的数据结构布局和内存中的数据结构布局是一致的。系统在载入一个可执行程序时,首先是 Windows 装载器(又称 PE 装载器)把磁盘中的文件映射到进程的地址空间,它遍历 PE 文件并决定文件的哪一部分被映射。其方式是将文件较高的偏移位置映射到较高的内存地址中。磁盘文件一旦被装入内存中,其某项的偏移地址可能与原始的偏移地址有所不同,但所表现的是一种从磁盘文件偏移到内存偏移的转换。PE 文件中的数据按照磁盘数据标准存放,比如以 0x200 字节为基本单位进行组织,当 PE 文件装载到内存时,将按照内存数据标准存放,比如以 0x1000 字节为基本单位,所以文件偏移地址和相对虚拟内存会有细微的差别,这种差别称为节偏移。PE 文件从内存到磁盘映射关系如图 4 所示,PE 文件内部数据结构具体便宜如图 5 所示。

根据 PE 内存镜像还原算法,如算法 1 所示,经过转换之后得到进程大致的磁盘可运行文件,调用主机中的 ClamAV 杀毒引擎对文件进程查毒进而判断当前进程是否可疑。

算法 1　PE 内存镜像还原算法

Procedure PEM2F(varbuffer:charArray);

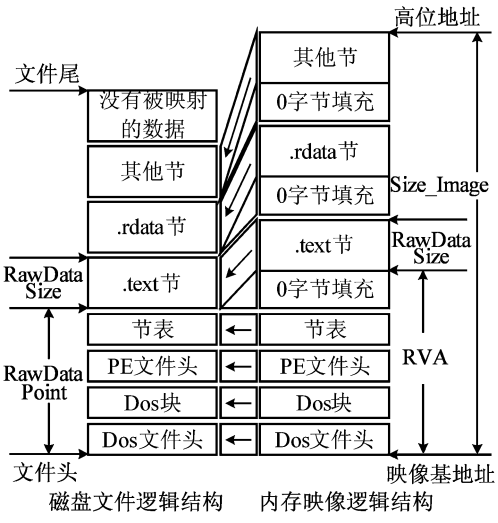

图 4　PE 文件从内存到磁盘映射关系

Fig. 4　Mapping of PE file from memory to disk

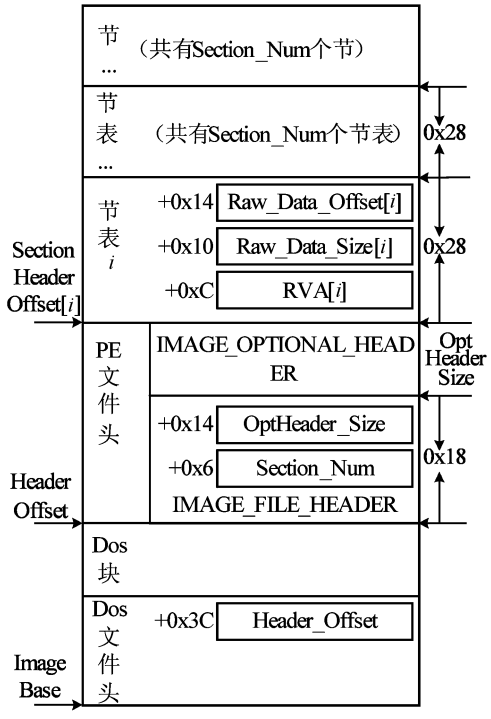

图 5　PE 文件内部数据结构具体偏移

Fig. 5　Migration of data structure in PE file

```
1 var Header_Offset, Image_Size: unsigned;
2 var Section_Num, OptHeader_Size: unsigned short;
3 var Section_Header_Offset, RVA, Raw_Data_Size,
    Raw_Data_Offset: unsignedArray;
4 Begin
5 Header_Offset = get_value(unsigned, buffer,0x3c);
6 Section_Num = get_value(unsigned short, buffer,
    Header_Offset + 0x6);
7 OptHeader_Size = get_value(unsigned short, buff-
    er, Header_Offset + 0x14);
```

```
8 for i = 1 to Section_Num do begin
9   Section_Header_Offset[i] = Header_Offset + 0x18 + OptHeader_Size + (i - 1) * 0x28;
10  RVA[i] = get_value(unsigned, buffer, Section_Header_Offset[i] + 0xc);
11  Raw_Data_Size[i] = get_value(unsigned, buffer, Section_Header_Offset[i] + 0x10);
12  Raw_Data_Offset[i] = get_value(unsigned, buffer, Section_Header_Offset[i] + 0x14);
13  for j = 0 to Raw_Data_Size[i] - 1 do buffer[Raw_Data_Offset[i] + j] = buffer[RVA[i] + j]
14 End for
15 End
```

3.4 进程监视、决策和 VNC 消息推送模块

这3个模块与前面的进程扫描模块一样运行在物理主机的用户空间层。

进程监视模块获取由内核处理模块分析得到的进程信息后,实时显示物理主机上所有虚拟机运行的当前进程信息,包括占用 CPU,内存等信息。在进程扫描模块扫描出 GuestOS 当前进程可疑时,由进程决策和 VNC 消息推送模块与 GuestOS 用户进行交流,由终端用户决定当前进程是否被列入终止表。本研究中 VNC 消息推送模块[18]同样是无代理的,通过修改 KVM 虚拟化平台中 Qemu-KVM 集成的 VNC Server 端源码,在源码中添加消息发送模块和反馈接收模块,将消息集成融入到虚拟机桌面图像中,并对 VNC Client 远程终端反馈的消息进行处理。最终建立一条对虚拟机自身系统透明的可在云平台和终端用户间双向交互的消息通道。进程决策模块负责传达通过 VNC 消息通道用户反馈的信息到内核模块,然后由内核模块进行相应处理。

4 结果分析与讨论

对 KVMPST 工作性能进行了测试,目的是分析本系统针对传统虚拟化反病毒架构的优势以及通过关键技术 PE 镜像还原提高病毒查杀率。

测试环境为:Intel Xeon E5530(支持 Intel VT-x 硬件虚拟化技术)处理器 / 16 G 内存 / Ubuntu 12.04 LTS 操作系统(Linux 内核版本 3.2.0-23-generic)/ Libvirt 版本:1.0.0/ 虚拟机管理器:Qemu-kvm 1.2.0/ 客户机数:5/ 客户虚拟机操作系统:Windows 7, CPU: 2core 内存:1 GB/ 杀毒引擎:ClamAV。

在传统虚拟化反病毒架构上每台虚拟机均部署单独的反病毒软件,测试结果表明,KVMPST 相比传统虚拟化反病毒架构,由于杀毒模块位于主机中,虚拟机中没有任何辅助杀毒模块,虚拟机中的恶意软件无法检测到主机中的杀毒模块,KVMPST 自身的安全性得到保障;由于只用部署一次反病毒程序来代替在多台虚拟机上分别部署,所消耗的 CPU、内存和磁盘 I/O 资源大幅减少,如表1所示。

表1 2种架构的反病毒程序占资源情况

Tab.1 Resource usage of two anti-virus structure

安全架构	CPU/%	内存/M	磁盘 I/O/(kB·s^{-1})
传统架构	19.8	205	32
KVMPST	5.1	42	8

在如今的开源杀毒引擎中,以 ClamAV 为例,在运行中主要使用了入侵检测技术中的特征值检验技术,PE 文件载入内存后,数据结构会发生偏移,直接用杀毒引擎扫描内存中的 PE 镜像,会使得有些病毒无法识别到。使用 PE 镜像还原技术后,开源文件型病毒扫描工具 ClamAV 在扫描常见的宏病毒、引导性病毒、脚本病毒、文件型病毒和特洛伊木马识别率上提高了23%,如表2所示。

表2 PE 镜像还原前后恶意进程识别率

Tab.2 Malicious processes recognition rate before and after PE image reduction

还原模式	常见病毒识别率/%
无镜像还原	62
有镜像还原	85

5 结束语

针对 KVM 虚拟化平台上的安全问题,提出一种部署在物理主机上的 Windows 客户机进程无代理在线杀毒技术,对里面所有虚拟机进程进行在线监控,以解决传统虚拟化反病毒架构的资源消耗和自身安全性问题,并通过 PE 镜像还原算法,提高现有开源杀毒引擎对进程内存信息的查杀率。在今后工作中,要结合新型扫描内存的杀毒引擎或者建立一个恶意进程内存信息特征库来辅助进行更好的恶意攻击防范工作。

参考文献:

[1] Rosenblum M, Garfinkel T. Virtual machine monitors: Current technology and future trends [J]. IEEE Computer, 2005, 38(5): 39 - 47.

[2] Feng Dengguo, Zhang Min, Zhang Yan, et al. Study on cloud computing security[J]. Journal of Software, 2011, 22(1): 71

-83. [冯登国,张敏,张妍,等. 云计算安全研究[J]. 软件学报,2011,22(1):71-83.]

[3]趋势科技. 虚拟化面临的安全问题[EB/OL]. [2014-04-20]. http://www.trendmicro.com.cn/cloudsecurity/12risks.html.

[4]Xiang Guofu, Jin Hai, Zou Deqing, et al. Virtualization-based security monitoring[J]. Journal of Software, 2012, 23(8): 2173-2187. [项国富,金海,邹德清,等. 基于虚拟化的安全监控[J]. Journal of Software, 2012, 23(8): 2173-2187.]

[5]Dunlap G W, King S T, Cinar S, et al. ReVirt: Enabling intrusion analysis through virtual-machine logging and replay [J]. ACM SIGOPS Operating Systems Review, 2002, 36(SI): 211-224.

[6]Symantec. Securing the promise of virtualization [EB/OL]. [2014-04-20]. http://www.symantec.com/content/en/us/enterprise/white_papers/b-WP_SecuringThePromiseOfVirtualization_WP_21229614.en-us.pdf.

[7]McAfee. Management for optimized virtual environments (MOVE) [EB/OL]. [2014-04-20]. http://www.mcafee.com/us/products/move-anti-virus.aspx.

[8]Trendmicro. Data center and cloud security [EB/OL]. [2014-04-20]. http://www.trendmicro.com/us/enterprise/cloud-solutions/index.html.

[9]Kaspersky. Kaspersky security for virtualization [EB/OL]. [2014-04-20]. http://www.kaspersky.com/products/business/applications/security-virtualization.

[10]阿里云. 阿里云监控服务[EB/OL]. [2014-04-20]. http://www.aliyun.com/product/jiankong/?spm=5176.383846.0.37.gcUi3w#detail.

[11]盛大云. 盛大云监控服务[EB/OL]. [2014-04-20]. http://www.grandcloud.cn/statics/docs/GrandCloud_CMS_Reference.pdf.

[12]华为云. 华为云安全体系[EB/OL]. [2014-04-20]. http://static.hwclouds.com/portal/product/html/ecc/ecc_security.html.

[13]腾讯云. 腾讯云平台安全防护体系[EB/OL]. [2014-04-20]. http://wiki.open.qq.com/wiki/腾讯云平台安全防护体系.

[14]Intel Corporation. Intel 64 and IA-32 architectures software developer's manual [EB/OL]. [2014-04-20]. http://www.intel.com/content/www/us/en/architecture-and-technology/64-ia-32-architectures-software-developer-manual-325462.html.

[15]Liu Jianfei. Research on technique of BIOS Trojan detection [D]. Zhengzhou: The PLA Information Engineering University, 2012. [刘建飞. 基于硬件虚拟化进程检测技术研究[D]. 郑州:解放军信息工程大学,2012.]

[16]Kojm T, Cathey M, Cordes C. Clamav anti-virus [J/OL]. [2008-04-14]. http://www.clamav.net.

[17]Li Zhuoyuan, Xian Ming. Analysis of PE file format. computer knowledge and technology [J]. 2009, 5(9): 2379-2381. [李卓远, 鲜明. PE 文件格式分析[J]. 电脑知识与技术,2009,5(9):2379-2381.]

[18]武汉大学. 云管理平台和虚拟机终端用户间 vnc 隐通道的建立方法:中国,CN103312814 A[P]. 2013-09-18.

(编辑　杨　蓓)