文章编号:1001 - 9081(2008)07 - 1769 - 03

# 基于本地虚拟化技术的隐藏进程检测

温 研[1],赵金晶[2],王怀民[1]
(1. 国防科学技术大学 计算机学院, 长沙 410073; 2. 北京系统工程研究所,北京 100101)
(celestialwy@126.com)

**摘 要**:自隐藏恶意代码已成为 PC 平台下急需解决的安全问题,进程隐藏则是这类恶意代码最常用和最基本的规避检测的自隐藏技术。针对这个问题,提出了一种新的基于本地虚拟化技术的隐藏进程检测技术——Gemini。基于该本地虚拟化技术,Gemini 在本地化启动的虚拟机中(Local-Booted Virtual Machine)完整重现了宿主操作系统的运行环境,结合隐式的真实进程列表(TVPL)获取技术,Gemini 实现了在虚拟机监视器(VMM)内检测宿主操作系统内隐藏进程的能力。测试结果证明了宿主计算环境重现的有效性与隐藏进程检测的完整性。

**关键词**:虚拟机监视器; 虚拟机; 自隐藏恶意代码;本地虚拟化技术;进程隐藏
**中图分类号**: TP309.5 **文献标志码**:A

# Detecting hidden process with local virtualization technology

WEN Yan[1], ZHAO Jin-jing[2], WANG Huai-min[1]
(1. *School of Computer, National University of Defense Technology, Changsha Hunan* 410073, *China*;
2. *Beijing Institute of System Engineering, Beijing* 100101, *China*)

**Abstract:** Currently stealth malware is becoming a major threat to the PC computers. Process hiding is the technique commonly used by stealth malware to evade detection by anti-malware scanners. In this paper, we presented a new VM-based approach called Gemini that accurately reproduced the software environment of the underlying preinstalled OS within the Gemini VM. With our new local-booting technology, Gemini VM just booted from the underlying host OS but not a newly installed OS image. In addition, Gemini adopted a unique technique to implicitly construct the Trusted View of Process List (TVPL) from within the virtualized hardware layer. Thus, Gemini provided a way to detect the existing process-hiding stealth malware in the host OS. Our evaluation results with real-world hiding-process rootkits, which are widely used by stealth malware, demonstrate its practicality and effectiveness.

**Key words:** Virtual Machine Monitor (VMM); Virtual Machine (VM); stealth malware; local virtualization; process hiding

## 0 引言

随着互联网技术的发展与普及,从互联网上下载各类应用程序已经成为 PC 用户常用的软件应用模式,同时这些下载的非可信程序中可能携带的恶意代码也对计算机系统的安全构成了很大威胁。其中,自隐藏恶意代码的危害尤为突出。所谓的自隐藏代码就是指在操作系统启动时运行,并利用各种相关技术使得系统工具无法检测到代码所在进程存在的恶意程序,而进程隐藏则是这类恶意代码最常用也是最基本的规避检测的自隐藏技术。按照微软公司发布的恶意代码清除工具(Malicious Software Removal Tool)所做的统计[1],该工具所检测到的恶意软件中超过 90% 使用了进程隐藏技术。因此,检测隐藏进程的能力将为抵御自隐藏恶意代码提供有效支持。

检测隐藏进程的有效机制是所谓的多视图验证技术(cross-view validation)[2],即分别从可信视图和非可信视图获取进程列表,存在于可信视图但未出现在非可信视图中的进程即为隐藏的进程。已有的可信视图获取技术主要分为两类,一类基于操作系统内核,一类基于虚拟机监视器。前者无法抵御内核态恶意代码的攻击;而后者依赖的操作系统信息依然可被内核态恶意代码篡改。此外,现有的虚拟机监视器的技术都是针对在虚拟机中重新安装部署的操作系统,这些操作系统并不能反映实际需要保护的宿主计算机系统的真实状态。

针对这些问题,本文提出了一种新的基于本地虚拟化技术的隐藏进程检测技术——Gemini。除了能够有效抵御内核态恶意代码的攻击外,与已有的基于虚拟机技术的解决方案相比,Gemini 具有两大优势:一是基于本地虚拟化技术的宿主操作系统重现,即 Gemini VM 中加载的操作系统不是重新安装部署的新的操作系统,而是在 Gemini VM 启动时创建的宿主操作系统的副本,从而完整再现了宿主操作系统的运行环境;另一个是不依赖于操作系统的隐式进程自省技术,Gemini 利用处理器管理进程地址空间的特征构造了不依赖于操作系统的进程枚举技术,使得所有的检测操作均在虚拟机监视器(即 Gemini VMM)的层次完成,有效保护检测机制的安全性。

## 1 体系结构

如前所述,现有基于虚拟机监视器的检测技术实际检测是在虚拟中安装运行的操作系统中潜在的恶意代码,而不是

**收稿日期**:2008 - 01 - 03;**修回日期**:2008 - 04 - 04。
**基金项目**:国家 973 计划项目(2005CB321801);国家杰出青年科学基金资助项目(60625203)。
**作者简介**:温研(1979 - ),男,博士研究生,主要研究方向:信息安全、虚拟化; 赵金晶(1981 - ),女,博士,助理研究员,主要研究方向:信息安全、计算机网络; 王怀民(1962 - ),教授,博士生导师,主要研究方向:分布计算、网络、信息安全。

PC 用户真正关心的宿主操作系统。与这些基于虚拟机监视器的检测技术不同,Gemini 在 Gemini VM 中完整重现了宿主操作系统的运行环境,保证了检测的真实性。

实现本地化虚拟机启动技术的关键是如何与宿主操作系统共享已安装在系统卷上的操作系统映像。因为 Gemini VM 需要访问系统卷以启动 Local-Booted 操作系统,而此时宿主操作系统也在修改系统卷,前者对数据的修改并没有使用宿主操作系统提供的访问接口,后者的修改信息也无法及时被 Gemini VM 内的文件系统获取,这就造成了文件系统数据与磁盘数据的冲突,操作系统关键文件的不一致将会导致系统的崩溃。

如图 1 所示,为了解决这个问题,Gemini 引入了基于卷快照的虚拟简单磁盘(Virtual Simple Disk)技术。卷快照是在创建时刻其对应的原始卷的一致性副本,它提供了与文件系统标准卷相同的访问接口。而虚拟简单磁盘则是卷快照(必须包括系统卷的快照)与虚拟分区表的集合,它是将卷快照以存储设备的形式导出到 Gemini VM 的实现方式。

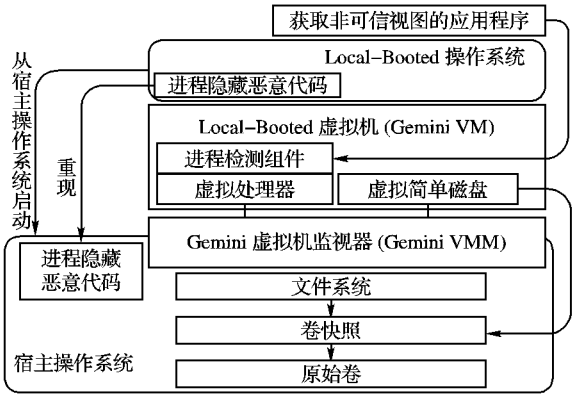


图 1 Gemini 体系结构

进程检测组件的核心功能即为获取 TVPL,进程检测组件利用了 Intel x86 处理器管理进程地址空间(Process Address Space)的通用机制实现了 TVPL 的获取。对于所有 Intel x86 体系结构的处理器,每个进程都有一个独立的进程地址空间,而处理器的 CR3 寄存器则指向了一个进程的地址空间的页目录表,因此,实际上 CR3 寄存器与进程形成了一一映射。利用动态指令转换技术,在创建了 Gemini VMM 后,所有修改 CR3 寄存器的操作都将被 Gemini VMM 捕获,进程检测组件正是利用这一特性通过监视 CR3 寄存器的变化来获取 TVPL。

同时,本文扩展了交叉视图验证,并定义可信视图为集合 $T$,而非可信视图为 $U_n$,其中 $n$ 表示非可信视图获取的层次,$H$ 表示隐藏资源的集合,则可得:

$$H = \bigcup_{n=1}^{K} (T - U_n) \tag{1}$$

由式(1)可知,如果 $U_n$ 不等于 $U_{n+1}$,则拦截机制是在第 $n$ 层与第 $n+1$ 之间实现的,从而实现了定位恶意代码的功能。

## 2 实现

前面描述的 Gemini 的体系结构显然是与操作系统无关的,但考虑到目前 PC 平台最常见的配置是 Windows 操作系统加 Intel x86 处理器的,所以 Gemini 的原型首先在配置了 Intel 处理器的 Windows 操作系统上实现。本地虚拟化技术的详细介绍见文献[4]。

对于所有 Intel x86 体系结构的处理器,每个进程都有一个独立的进程地址空间,进程基于“逻辑地址”访问内存,启动了段式内存管理的 x86 处理器首先将此逻辑地址转换为“线性地址”,接着处理器通过页式内存管理机制将由段式管理形成的线性地址映射为“物理地址”。x86 处理器的页式内存管理将进程地址空间分为固定大小的页面,每一个页面被映射到物理内存中的一页。

如图 2 所示,进行线性地址到物理地址的转换时,x86 处理器的页式内存管理器利用页目录表/页表/页描述项这三层结构维护地址空间中的线性地址到物理内存页的映射,线性地址被划分为三个部分:页目录表下标、页表下标、页内地址偏移。x86 处理器中负责存放指向当前页目录表指针的是控制寄存器 CR3。寻址时先从 CR3 取得页目录表的基地址,然后根据页目录索引,取得相应页表的基地址,接着根据页表索引和之前得到的基地址,从页表中取得相应的页描述项,最后将页描述项中的页面基地址和线性地址中的页内地址偏移相加,得到物理地址。

如前所述,CR3 寄存器的值指向了一个进程的地址空间的页目录表,因此,实际上 CR3 寄存器与进程形成了一一映射。利用动态指令转换技术,Gemini 虚拟机内所有修改 CR3 寄存器的操作都将被 Gemini VMM 捕获,进程检测组件 Gemini 正是利用这一特性,通过监视 CR3 寄存器的变化来获取 TVPL。

Gemini 利用红黑树组织 TVPL 信息,每个节点包含一个进程的信息,节点由其表示的进程所对应的 CR3 寄存器的值索引。进程检测操作在 Gemini VMM 捕获到修改 CR3 操作时被调用,所以 Gemini 的输入实际就是新的 CR3 的值。

进程创建的检测。如果新的 CR3 的值未在现有 TVPL 中出现,则表示检测到新创建的进程。接着 Gemini 将读取新进程相关信息(包括进程映像的路径,命令行等)并与 CR3 值绑定后创建新的进程节点后插入 TVPL 树。

进程退出的检测。Gemini 检测进程退出的技术利用了操作系统的进程地址销毁机制。在 Windows 操作系统或 Linux 操作系统下,一个进程退出后,操作系统将会移除所有用户态内存页所对应的页描述项,而由于所有进程共享了内核态的地址空间,因此内核态内存页所对应的页描述项将不会被清除,但是非分页(non-paged)内存所占用的页描述项中的 un-present 比特位将被置 1。

所以检测进程退出的关键就是这个特殊的非分页内存页的选择,而此内存页的地址必须满足以下四个要求:

1)该信息是隐式信息,即该地址不能被修改或者修改后操作系统不能正常运行。

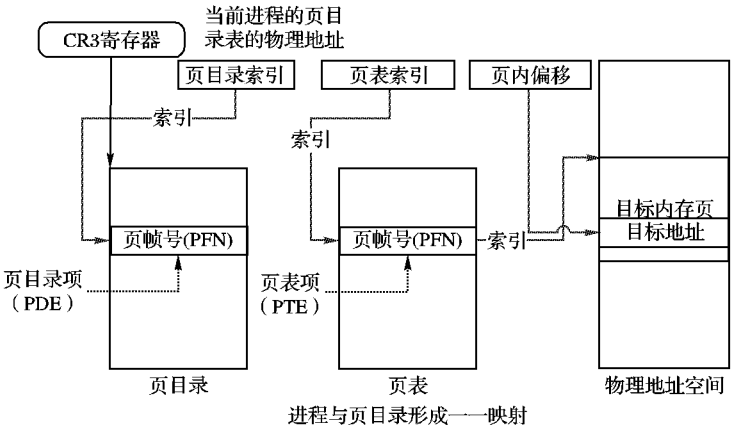


图 2 x86 体系结构处理器的页式地址转换机制

2)在进程创建时即被初始化,这是因为 Gemini 在检测到新进程(或者是新创建的,或者是 Gemini VMM 启动后第一次被调度运行的)时就需要获取此内存页的地址。

3)在进程的整个生命期内固定不变,如果这个内存页地址改变了则会导致 Gemini 无法正确获取该页的页描述表项,进而产生误判。

4)在进程的整个生命期内均可访问,这一点保证这个地址直到进程刚退出时都有效。

在 Windows 平台下,存储进程信息的内核数据结构进程执行体信息块(Executive Process Block,EPROCESS)所在的内存页就满足以上四个要求,而且该数据结构内存储了与进程相关的重要信息。一个进程的 EPROCESS 的地址存储在线程执行体信息块内(Executive Thread Block,ETHREAD),而 ETHREAD 在所有 NT 系列的 32 位 Windows 版本下地址均为 0xFFDFF124。因此,Gemini 可以在检测到新进程时通过直接内存读取获取 EPROCESS 的地址及其包含的进程信息。从安全的角度考虑,若内核态恶意代码为了攻击 Gemini 而修改 ETHREAD 中 EPROCESS 地址,则会破坏线程与进程的关联关系,进而导致 Windows 内核崩溃,对恶意代码而言这显然会破坏其本身的正常运行。此外,利用 EPROCESS 中的进程信息,Gemini 还获取了进程映像文件的路径与命令行信息。

基于对进程地址空间销毁原理的分析,Gemini 为每个进程均选择了一个非分页的内存页的地址。当 CR3 改变时,Gemini 将会遍历 TVPL 中的每个进程,取得该进程对应的非分页内存页的地址,通过这个地址获取该内存页的页描述项,如果该项的 un-present 比特位已被设置,则说明这个进程的地址空间已被销毁,进程已经退出,进而 Gemini 将把该进程对应的节点从 TVPL 树中删除。如果现有 TVPL 树中没有新的 CR3 对应的进程,则说明有新的进程创建,则创建新的进程节点,并获取该进程的 EPROCESS 的地址,最后将绑定了 CR3 值与相应进程信息的新进程节点插入 TVPL 树。

至此,Gemini 准确获取了 TVPL。而为了检测隐藏进程,Gemini 还需要获取进程的非可信视图。为此,Gemini 通过调用函数 ZwSetSystemInformation 来获取非可信的进程列表。

## 3　实验

### 3.1　TVPL 检测的完整性

为了测试 Gemini 检测进程的完整性,测试计算机重新安装了操作系统以确保系统未被恶意代码感染。从测试完整性的角度考虑,进程检测的不完整只包括两种情况:一是误报,即检测实际上系统中不存在的进程;另一个是漏报,即未检测到系统中实际运行中的进程。

表 1 比较了实际进程的创建/退出的次数,地址空间的创建/销毁的次数与 Gemini 检测的次数,测试使用 Windows 标准的 CreateProcess API 创建进程,测试时每分钟创建 20 个进程,每个进程运行 30 s 后退出。如表 1 中数据所示,Gemini 的 TVPL 检测机制没有产生任何漏报和误报。

### 3.2　隐藏进程检测

为了验证 Gemini 检测隐藏进程的有效性,测试使用了五种基于不同进程隐藏技术的常用恶意代码,包括 Aphex[5]、Hacker Defender[6]、FU[7]、FUTo[8] 和 phide_ex[9]。测试同时比较了已有 Anti Rootkit Group[10] 推荐的检测工具,包括 F-Secure BlackLight 2.2, DarkSpy 1.0.5, IceSword 1.20, RkUnhooker 3.0, UnHackMe 2.5.5, GMER 1.0, KProcCheck 0.2, bitdefender _ antirootkit-BETA1, Process Hunter 1.1 和 TaskInfo 6.2。

测试结果显示所有被测试的恶意代码检测工具都可以检测到 Aphex, Hacker Defender 和 FU, BlackLight、Icesword、UnHackMe 和 TaskInfo 不能检测到 FUTo,而所有的这些检测工具均不能检测到 phide_ex,只有 Gemini 能检测到这五种方式隐藏的进程。

**表 1　TVPL 检测完整性测试**

| 操作系统 | 进程创建次数 | 地址空间创建次数 | Gemini 检测到次数 |
|---|---|---|---|
| Windows 2000 SP4 | 3 200 | 3 200 | 3 200 |
| Windows XP SP2 | 3 200 | 3 200 | 3 200 |
| Windows 2003 SP1 | 3 200 | 3 200 | 3 200 |
| 操作系统 | 进程创建次数 | 地址空间创建次数 | Gemini 检测到次数 |
| Windows 2000 SP4 | 3 200 | 3 200 | 3 200 |
| Windows XP SP2 | 3 200 | 3 200 | 3 200 |
| Windows 2003 SP1 | 3 200 | 3 200 | 3 200 |

## 4　结语

针对 PC 平台,为了有效检测宿主操作系统的进程隐藏恶意代码,本文基于本地虚拟化技术构造了支持操作系统迁移的虚拟机监视器 Gemini VMM,并在 Gemini VMM 中实现了不依赖于操作系统的隐式的隐藏进程检测技术,为自隐藏恶意代码的检测技术提供了有效支持。

本文的测试结果证明了 Gemini 能够准确获取真实的进程列表(TVPL),充分说明了本文提出的解决方案的有效性。

**参考文献:**


[1] Microsoft. Windows Malicious Software Removal Tool [EB/OL]. [2007-12-20]. http://www.microsoft.com/security/malwareremove/.

[2] WANG Y M, BECK D, VO B, *et al*. Detecting stealth software with strider GhostBuster[C]// Proceedings of the International Conference on Dependable Systems and Networks. Washington, DC, IEEE Press, 2005:368-377.

[3] UHLIG R, NEIGER G, RODGERS D, *et al*. Intel virtualization technology [J]. IEEE Computer, 2005, 38(5): 48-56.

[4] WEN Yan, WANG Huai-min. A secure virtual execution environment for untrusted code [C]// 10th International Conference on Information Security and Cryptology, Berlin: Springer, 2007: 156-167.

[5] Aphex: AFX Windows Rootkit 2003 [EB/OL]. [2007-11-21]. http://www.iamaphex.cjb.net.

[6] HackerDefender[EB/OL]. [2007-11-25]. http://hxdef.org/.

[7] Fuzen_op, FU Rootkit [EB/OL]. [2007-11-23]. http://www.rootkit.com/project.php?id=12.

[8] SILBERMAN P. FUTo: Bypassing Blacklight and IceSword [EB/OL]. [2007-11-26]. https://www.rootkit.com/newsread.php?newsid=433.

[9] PE386. phide_ex - untimate process hiding example [EB/OL]. [2007-11-24]. http://forum.sysinternals.com/printer_friendly_posts.asp?TID=8527.

[10] Anti Rootkit Group [EB/OL]. [2007-11-28]. http://www.antirootkit.com/blog/.