软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2015,26(5):1151−1172 [doi: 10.13328/j.cnki.jos.004821]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

内存取证研究与进展*

张 瑜[1], 刘庆中[2], 李 涛[3], 吴丽华[1], 石 春[1]

[1](海南师范大学 计算机科学系,海南 海口 571158)

[2](Department of Computer Science, Sam Houston State University, USA)

[3](四川大学 计算机学院,四川 成都 610065)

通讯作者: 张瑜, E-mail: bullzhangyu@126.com

摘 要: 网络攻击内存化和网络犯罪隐遁化,使部分关键数字证据只存在于物理内存或暂存于页面交换文件中,这使得传统的基于文件系统的计算机取证不能有效应对.内存取证作为传统文件系统取证的重要补充,是计算机取证科学的重要组成部分,通过全面获取内存数据、详尽分析内存数据,并在此基础上提取与网络攻击或网络犯罪相关的数字证据,近年来,内存取证已赢得安全社区的持续关注,获得了长足的发展与广泛应用,在网络应急响应和网络犯罪调查中发挥着不可替代的作用.首先回顾了内存取证研究的起源和发展演化过程;其次介绍了操作系统内存管理关键机制;然后探讨了内存取证的数据获取和分析方法,归纳总结目前内存取证研究的最新技术;最后讨论了内存取证存在的问题、发展趋势和进一步的研究方向.

关键词: 网络安全;内存取证;网络攻击;网络犯罪;应急响应

中图法分类号: TP309

中文引用格式: 张瑜,刘庆中,李涛,吴丽华,石春.内存取证研究与进展.软件学报,2015,26(5):1151−1172. http://www.jos.org.cn/1000-9825/4821.htm

英文引用格式: Zhang Y, Liu QZ, Li T, Wu LH, Shi C. Research and development of memory forensics. Ruan Jian Xue Bao/Journal of Software, 2015,26(5):1151−1172 (in Chinese). http://www.jos.org.cn/1000-9825/4821.htm

Research and Development of Memory Forensics

ZHANG Yu[1], LIU Qing-Zhong[2], LI Tao[3], Wu Li-Hua[1], SHI Chun[1]

[1](Department of Computer Science, Hainan Normal University, Haikou 571158, China)

[2](Department of Computer Science, Sam Houston State University, USA)

[3](College of Computer Science, Sichuan University, Chengdu 610065, China)

Abstract: Cyber attacks and cybercrimes that run stealthy in computer memory make the traditional file system-based computer forensics tasks difficult to be carried out effectively, and thus become a serious security threat. As an important branch of computer forensics, memory forensics is an effective way to combat evasive cyber attacks and cybercrimes. It extracts the evidence of cyber attacks and cybercrimes by comprehensively obtaining and analyzing memory data left by attackers. In recent years, the memory forensics which has drawn sustained attention of the security community, and undergone rapid development with wide range of applications, plays an irreplaceable role in the network incident response and cybercrime investigations. This paper first reviews the origin and evolution processes of memory forensics research, followed by introduction of the key mechanism of operating system memory management. It then explores the memory data acquisition and analysis methods, and summarizes the latest memory forensics technology. The paper concludes with a discussion of the existing problems of current memory forensics, and outlook of the trends and further research directions of memory forensics.

Key words: network security; memory forensic; cyber attack; cyber crime; incident response

* 基金项目: 国家自然科学基金(61462025, 61262077, 61173159, 61362016); 国家级大学生创新创业训练计划(201211658036); 海南省自然科学基金(613161, 613163)

收稿时间: 2014-05-15; 修改时间: 2014-11-17; 定稿时间: 2015-01-21

网络与信息技术的加速渗透和深度应用以及软件漏洞不断涌现,导致网络攻击和网络犯罪频发,造成了严重的网络安全威胁.计算机取证是打击计算机与网络犯罪的关键技术,其目的是将犯罪者留在计算机中的“攻击痕迹”提取出来,作为有效的诉讼证据提供给法庭,以便将犯罪嫌疑人绳之以法[1].

然而,由于反取证技术[2]、系统运行与内存交换机制等原因,使传统基于文件系统的取证技术难以有效提取相关数字证据,主要体现在 3 个方面:

(1) 反取证技术中的数据隐藏、数据转换、数据伪造等技术[3],更改了原始数字证据,导致取得伪证.

(2) 操作系统内核通常会及时释放内存页面,使暂存于内存或交换页面文件中的相关证据信息在关机后消失,造成事后无证可取.比如,Jason 等人[4]的研究表明,为了确保系统的正常运行,Windows 系统内核总是试图释放内存页面,多数应用程序内存页面的存留时间少于 5 分钟.

(3) 内存中通常存放有诸如解密密钥、应用程序口令、恶意代码、进程信息、注册表信息、网络连接、系统状态等敏感数字证据,传统取证方法难以获取,只有通过分析物理内存镜像和页面交换文件的二进制数据才能够获取.

以上事实提示:为了有效而全面地提取与网络攻击或网络犯罪相关的数字证据,需及时、规范地进行内存取证.

内存取证作为计算机取证科学的重要分支,是指从计算机物理内存和页面交换文件中查找、提取、分析易失性证据,是对传统基于文件系统取证的重要补充,是对抗网络攻击或网络犯罪的有力武器.当系统处于活动状态时,物理内存中保存着关于系统运行时状态的关键信息,比如解密密钥、口令、打开文件、进程信息、网络连接、系统状态信息等.进行内存取证,就是通过获取物理内存和页面交换文件的完整拷贝,并在另一台计算机中进行内存数据分析,重构出原先系统的状态信息.因此,内存取证技术研究对于完善计算机取证方法、规范取证流程(模型)、提取完整数字证据、洞悉网络攻击机理、快速进行应急响应、遏制打击网络犯罪等方面意义重大.

相对于传统基于文件系统的取证,内存取证研究起步较晚,始于 2005 年夏季著名的 DFRWS(Digital Forensic Research Workshop,数字取证研究工作组)发起的针对 Windows 系统的内存取证分析挑战赛[5],以鼓励内存取证分析研究和相关取证工具开发.DFRWS 于 2008 年又发起了针对 Linux 系统的内存取证分析挑战赛.之后,几乎每年的 DFRWS 大会都有内存取证分析的讨论议题.此外,著名黑客大会(Black Hat,Def Con, ShmooCon 等)从 2006 年也开始举办内存取证分析专题研讨会.

工业界和政府部门同样关注内存取证分析研究.美国联邦调查局 FBI 下属的 RCFL(Regional Computer Forensics Laboratory)自 2006 年开始了内存取证研究[6].2012 年,美国国防部高级研究计划署 DARPA[7]发起了对隐匿恶意软件的内存取证分析专题研讨.美国国土安全部 DHS[8]为应对网络犯罪而进行了恶意软件内存取证研究.与此同时,相关内存取证工具相继出现,比如 Guidance 公司的 WinEn、HBGary 公司的 FastDump、Mantech 开发的 MDD、Agile Consulting 开发的 Nigilant32 等.

目前,国内对内存取证的关注和研究相对较少.上海交通大学[9]、电子科技大学[10]、吉林大学[11]、重庆邮电大学[12]等高校已相继开展内存取证分析研究,相关内存取证软件工具主要有:上海盘石的“SafeImager 盘石计算机现场取证系统”[13]、厦门美亚柏科的“取证大师”[14]、山东省科学院计算中心的“计算机在线取证系统”[15,16]、重庆市公安局研制的“内存取证系统”[17]等.

作为一种实时提取数字证据、对抗网络攻击、打击网络犯罪的有力武器,内存取证已成为信息安全领域研究者所共同关注的热点,但目前国内尚未有详细而全面介绍内存取证机理与研究成果的综述论文.为深入理解内存取证机理和发展趋势,总体把握内存取证研究进展,并促进国内在该方向上的研究,综述内存取证研究进展工作非常必要.

本文第 1 节介绍内存取证的起源和发展演化过程.第 2 节概述 Windows 内存管理相关机制,是理解内存取证相关技术与方法的理论基础.第 3 节讨论内存数据获取方法.第 4 节探讨内存数据分析方法.第 5 节讨论目前内存取证存在的问题,并展望内存取证的未来发展趋势和进一步的研究方向.

1 内存取证的起源与发展

1.1 内存取证技术的起源与发展

内存取证概念最早出现在美国空军特别调查办公室的 Kornblum 发表在 2002 年 DFRWS 的主题报告《Preservation of Fragile Digital Evidence by First Responders》[18]中.为了处理网络应急响应所面临的问题,Kornblum 提出了需要调查易失性内存信息,以全面而准确地获取网络攻击和网络犯罪证据.真正意义上的内存取证研究,始于 2005 年夏季 DFRWS 发起的针对 Windows 系统的内存取证分析挑战赛.通过分析 DFRWS 给定的一个 Windows 2000 物理内存转储文件,要求参赛者提取该文件中所包含的隐匿进程及其隐匿方式、网络攻击者如何攻击以及何时、何处发起攻击等相关攻击时间轴信息.3 位参赛者脱颖而出,赢得了挑战赛的胜利:Betz[19]通过逆向分析 Windows 2000 内核,获取其重要的内核数据结构,并研发了内存取证工具 MemParser,详细地提取了该内存转储文件中所蕴含的信息;Garner 和 Mora 团队[20]通过分析虚拟机中 Windows 2000 崩溃转储文件的重要内核数据结构,开发了内存取证工具 KNTList,详尽地获取了该内存转储文件中的信息.DFRWS 于 2008 年又发起了针对 Linux 系统的内存取证分析挑战赛.

此后,内存取证研究获得计算机取证安全社区的极大关注并得以迅速发展,各种内存取证技术方法相继出现,呈百花齐放之势.回顾内存取证研究的发展历程,我们可将其划分为 3 个阶段:① 内存取证研究起步探索阶段(2002 年~2005 年);② 内存取证研究快速发展阶段(2006 年~2010 年);③ 内存取证研究准成熟阶段(2011 年~至今).内存取证研究起源与发展时间轴如图 1 所示.

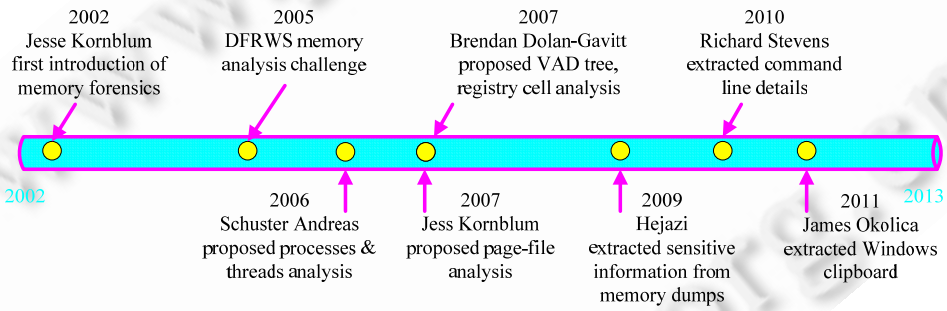


Fig.1 Timeline of memory forensics research

图 1 内存取证研究时间轴

在内存取证起步探索阶段,研究人员主要侧重于提出各种内存取证源和相关研究思路,为加快内存取证研究发展奠定了思想和理论基础.在此期间,Eoghan[21]提出了将网络连接信息作为一种取证源,Carvey[22]提出了 Windows 注册表可作为取证源,Andreas[23]提出了将系统内存池作为内存取证源,Dolan-Gavitt[24]提出了将 VAD(virtual address descriptor,虚拟地址描述符)作为内存取证源,Kornblum[25]提出了将页面交换文件数据作为内存取证源.同时,Kornblum[18]提出了在网络应急响应时需保存易失性内存数据,以提取尽可能多的攻击信息;Carrier 和 Spafford[26]提出了应将物理内存作为数字证据调查源,以全面获取相关网络攻击信息.

在内存取证快速发展阶段,研究者提出了各种内存数据获取和分析方法,迅速提升了内存取证研究水平及应用范围,使内存证据开始为相关执法者所采用.Andreas[27]详细分析了 Windows 内存转储文件中的进程和线程信息、阐述了 Windows 虚拟地址转译机制、重建了内存二进制文件、利用内存池分配机制提取了网络套接字连接信息.Dolan-Gavitt[24,28]提出了利用 VAD 树提取内存进程和线程信息的方法,详尽地分析了内存注册表数据结构,提出了内存注册表键值信息提取方法.Kornblum[18]提出了一种虚拟地址转换方案,即,使用无效 PTE 获取被换出至页面交换文件中的内存数据,以尽可能多地获取内存数据;还提出了重建内存可执行文件映像的方法.Arasteh 和 Debbabi[29]提出了从内存堆栈中提取进程执行历史信息的方法.

在内存取证研究准成熟阶段,研究者在掌握了各种内存取证方法的基础上,开始侧重于研究各种内存取证及评价系统框架与集成方案,以期实现内存取证研究的系统化、模块化、通用化.Petroni 等人[30]提出了一种模块化的、可扩展的 Volatility Framework,通过对低层内存数据获取的封装而向上提供数据接口,从而使取证研究者专注于高层的内存数据分析与挖掘.Volatility Framework 的模块化设计,使其能够轻松地支持诸如 Windows, Linux,Mac OS 及 Android 等系统,使研究者能针对不同取证需求而增加相应算法和取证插件,已成内存取证研究领域的一种事实上的标准系统框架.此外,针对各种内存取证工具软件的测评需求,Voemel 等人[31]提出了一种内存取证软件的评价平台模型.

1.2 内存取证工具软件的发展过程

内存取证工具软件通过捕获内存数据、分析内存数据,从而提取具有法律效力的数字证据,是内存取证研究的核心载体.随着网络攻击和网络犯罪技术的不断演化,研究者有针对性地开发出相应的内存取证工具软件,以不断适应对网络安全威胁的有效、全面、完整的取证要求.

早期的内存取证是作为磁盘取证的必要补充而提出的,因此,早期的内存取证工具主要侧重于搜索磁盘取证工具难以提取到的内存字符串(比如口令、用户名、IP 地址等).比如,德国 X-Ways 公司的 WinHex 是一款通用的 16 进制编辑器,可用于查找、修改内存数据.然而,随着网络攻击和网络犯罪技术的不断更新与演化,证据的内存化趋势已非常明显.这从客观上要求必须提出新的内存取证研究思路,研发新的内存取证工具软件.以 2005 年著名 DFRWS 的内存分析挑战赛为起点,计算机取证安全社区掀起了内存取证研究的热潮.此后,各种内存取证方法和相应的内存取证工具软件不断被提出和开发,已呈百花齐放之势.

此时,内存取证工具主要以操作系统内核数据结构为特征,通过搜索匹配内存转储文件,从而提取出其中蕴含的相关证据.比如,Betz 基于 Windows 2000 内核数据结构,研发了内存取证工具 MemParser[19];Garner 和 Mora 团队通过逆向分析 Windows 2000 崩溃转储文件的重要内核数据结构,开发了内存取证工具 KNTList[20]; Andreas 基于 Windows 内核进程和线程结构开发的内存进程与线程取证工具 PTFinder[32],基于 Windows 内存池分配机制研发了枚举内存进程与线程的 PoolTools[33];Dolan-Gavitt 通过分析 VAD 树结构开发的借助遍历 VAD 树来枚举内存进程与线程信息的取证工具 VADTools[34].此外,一些公司也开始推出内存取证工具软件,比如,Mantech 公司开发的 Memory DD(mdd)、Agile Consulting 公司开发的 Nigilant32、Mandiant 公司开发的 Memoryze.

随着内存取证技术的发展,相关内存取证工具软件已向系统化和集成化方向发展,比如,美国 Guidance Software 公司开发的 WinEn、美国 AccessData 公司的 FTK 取证工具、美国 HBGary 公司的 Responder Pro、德国 X-Ways software 公司的 X-Ways Forensics、韩国 FinalData 公司的 Final Forensics、澳大利亚 Nuix 公司的 FBI Forensic Desktop、Matthew Shannon 开发的 F-Response、MoonSols 公司的 Windows Memory Toolkit.这些内存取证工具软件可分析提取 Windows 休眠文件、完整内存转储文件、完整系统崩溃转储文件、虚拟机内存转储文件等多种文件,并将提取出的相关内存证据以图形化形式呈现,促进了内存取证技术的推广应用.

在系统化和集成化的同时,内存取证工具软件开发也在向模块化、框架化方向发展.比如,Volatile System 开发的 Volatility Framework[35]是一款用 Python 编写的内存取证工具,支持 32 位或 64 位的 Windows,Linux,Mac, Android 等平台.Volatility 以 Windows 符号和数据结构定义特征码,通过扫描内存查找这些特征码,提取出相关信息.Volatility 真正可称赞之处在于其可扩展框架,取证调查人员可利用此框架的核心功能编写自己的插件,以满足不同类型的内存取证研究需要.比如,对于内存恶意代码取证,开发了 MalFind,Kernel_Hook,Usermode_Hook 等相关插件;对于内存数据恢复,开发了 FileObjScan,DriverScan,ObjTypeScan,CryptoScan,ModDump 等插件;对于内存进程线程取证,开发了 PsTree,Suspicious 等插件.

从上述发展轨迹中我们可以看出,内存取证技术的发展紧随着网络攻击和网络犯罪等网络安全威胁的变化.作为内存取证技术载体的内存取证工具软件,也顺应这种网络安全威胁形势而不断发展更新自身的功能.在模块化、通用化、可扩展的内存取证框架软件(如 Volatility)基础上,针对各种网络安全威胁类型,研发相应的内存取证功能插件,以促成内存取证研究的可持续化发展.

2 内存管理机制

操作系统内存管理机制是内存取证研究的理论基础与发展基石.鉴于 Windows 操作系统的应用广泛性和代表性,本文将以 Windows 系统为例来阐述操作系统内存管理机制.Windows 内存管理可概括为三大机制[36]:① 虚拟地址空间管理;② 物理页面管理;③ 地址转译和页面交换.

2.1 虚拟地址空间管理机制

在早期的计算机系统中,程序员负责管理内存,后来,为了减轻程序员的负担,改由操作系统负责管理内存.这是程序设计发展史上的一次重要变革.在多进程运行环境中,为支持每个进程拥有逻辑上独立的地址空间,操作系统需使各进程地址空间相互隔离,互不干扰.

Windows 虚拟地址空间管理目的是实现各进程地址空间隔离,所以进程所见的是虚拟地址空间.虚拟地址也称为线性地址,是 Windows 在保护模式下所使用的逻辑地址,其空间大小取决于地址总线的宽度,比如,在 32 位系统上,虚拟地址空间大小为 2^{32} 字节,然而 CPU 实际访问的却是物理地址空间.所谓物理地址,是 RAM 中存储单元的索引,CPU 通过这组地址线与 RAM 相连,并通过在这些地址线加上电平信号来读写相应内存单元.换言之,在程序设计时使用虚拟地址,而程序执行时则使用物理地址.因此,Windows 需要对各进程虚拟地址进行管理,以便各进程互不干扰地运行于相同的物理内存中.

Windows 的进程虚拟地址空间是通过 VAD 来管理的.VAD 描述的是一段连续的地址空间范围,被设计为一棵平衡二叉树,以便 Windows 快速定位一段虚拟地址空间.随着进程的运行,不断地有地址范围被保留或提交,该进程的 VAD 树便逐渐形成.因此,借助于 VAD 不仅能够获取进程所使用的虚拟地址空间信息,还可以获取该进程的其他相关信息.

2.2 物理页面管理机制

Windows 既需管理虚拟地址,也需管理物理地址所在的物理内存页面.毕竟,Windows 的进程都是在物理内存中执行的.Windows 系统使用 PFN(page frame number database,页帧编号数据库)来描述物理内存各页面的状态.PFN 数据库中的每个 PFN 项对应于一个物理页面,记录了该页面的使用情况,包括其状态、对应页表项的地址等信息.此外,Windows 还维护着一组链表,分别将相同类型的页面链接起来,主要包括零化链表、空闲链表、备用链表、修改链表、坏页面链表等.Windows 的虚拟地址转译和页面交换机制是基于这些数据结构实现的.

2.3 地址转译和页面交换机制

(1) 地址转译

由于程序使用虚拟地址,而 CPU 则使用物理地址,因此,需借助于 CPU 芯片硬件和操作系统软件配合来实现从虚拟地址到物理地址的转换.这就是 Windows 系统的虚拟地址转译机制.转译机制是由 CPU 芯片硬件提供,而转译中所使用的数据结构则由 Windows 操作系统管理.

Windows 系统采用页式内存管理,虚拟地址空间是按页(page)来管理的,物理地址空间是按帧(frame)来管理的,页和帧的大小相等.因此,进程的虚拟地址页面可实现离散分配,即,虚拟地址空间中连续的页面对应于物理地址空间的页帧无需连续.

采用页式内存管理后,虚拟地址在结构上分为两部分:页索引和页内偏移.其中,页索引是指该虚拟地址在地址映射时的索引编号,页内偏移则指该地址在页面内部的具体位置.Intel x86 采用分级页表来管理地址映射.32 位虚拟地址中的页索引部分又被分成页目录索引(10 位)和页表索引(10 位)两部分.

因此,Intel x86 在解析一个虚拟地址时,首先根据最高 10 位在页目录中定位其页目录项 PDE(page directory entry),它指向一个页表;其次,根据接下来的 10 位在页表中定位其页表项 PTE(page table entry),此页表项指定了目标页面的物理地址;最后,在此物理地址基础上加上页内偏移量,得到最终的物理地址.Windows 虚拟地址转译机制如图 2 所示.

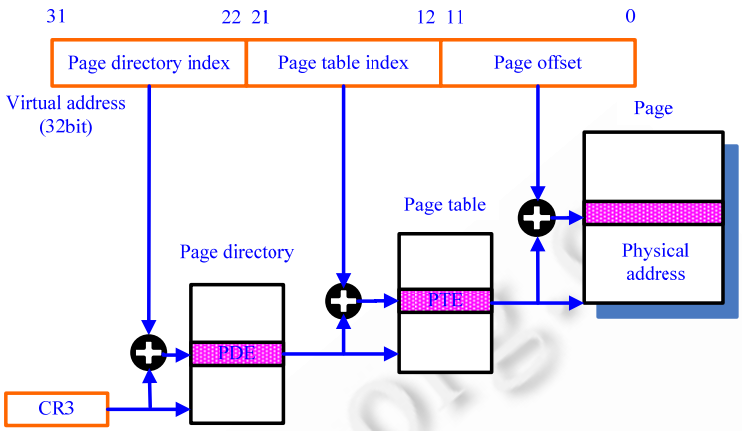


Fig.2 Virtual address translation mechanics

图 2 虚拟地址转译机制

如果 CPU 寄存器中的分页标志位被设置,那么在执行内存操作机器指令时,CPU 会自动根据页目录和页表中的信息,把虚拟地址转换成物理地址,完成该指令操作.因此,借由 Windows 的地址转移机制,可将进程中所使用的虚拟地址转换为物理内存中的物理地址,从而完成内存数据的定位,为内存数据获取提供支持.

(2) Windows 页面交换

当进程所需的内存数量大于计算机所安装的 RAM 时,Windows 采用页面交换机制予以处理.Windows 页面交换有两种情况:① 进程使用了某个尚未得到物理页面的虚拟地址;② 进程工作集限制其不能拥有更多物理页面.

第 1 种情况将导致页面换入,首先由 CPU 触发页面错误异常;然后,Windows 在页面错误处理例程中为其分配页面,并设置相关的页表项和页面之间的对应关系;最后,进程就可以透明地访问该地址.

第 2 种情况将导致页面换出,在内存紧缺时,Windows 将不紧急进程的部分代码或数据存放到硬盘文件 Pagefile.sys 中,腾出物理内存以供他用;当内存紧缺得以缓解时,再将硬盘上的进程代码或数据换入内存,使之继续运行.

由此可见,页面交换文件是物理内存的一种自然延伸.就其数据内容而言,页面交换文件与物理内存是同等的,只不过受页面调度影响而在不同时刻位于不同物理位置.所以,完整的内存数据应包括两部分:物理内存数据和页面交换文件数据.

(3) 进程和线程数据结构

在 Windows 中,进程是系统各种资源的容器,它定义了一个地址空间作为其基本执行环境;线程是指令执行序列,它直接访问所属进程中的资源.由于 Windows 内核采用层次结构,进程和线程在微内核层和执行体层上都有对应的数据结构.

Windows 内核使用 NtCreatProcess 函数创建一个进程的过程如下:首先,创建一个执行体层进程对象 EPROCESS 和内核层进程对象 KPROCESS;然后,创建一个初始的执行体层线程对象 ETHREAD 和内核层线程对象 KTHREAD;最后,设置好初始执行环境并参与进程(线程)调度.

在微内核层上,进程和线程的数据结构分别为 KPROCESS 和 KTHREAD,其中包含了系统资源管理和多控制流并发执行所涉及的基本信息.进程对象 KPROCESS 提供了线程的基本执行环境,包括进程地址空间和一组进程范围内公用的参数;线程对象 KTHREAD 提供了为参与线程调度而必需的各种信息及其维护控制流的状态.

在执行体层上,进程和线程的数据结构分别为 EPROCESS 和 ETHREAD.它们涉及操作系统各方面信息,不

仅内嵌了内核层上的进程和线程对象,而且还包括了更多涉及进程与线程管理的信息.比如,在 EPROCESS 中,包含进程的标识、映像文件、配额限制、锁、与进程的物理内存和虚拟内存相关的成员等;在 ETHREAD 中,包含线程的标识、启动地址、执行时间、安全属性等.Windows 内核中的进程与线程间相互关系如图 3 所示.

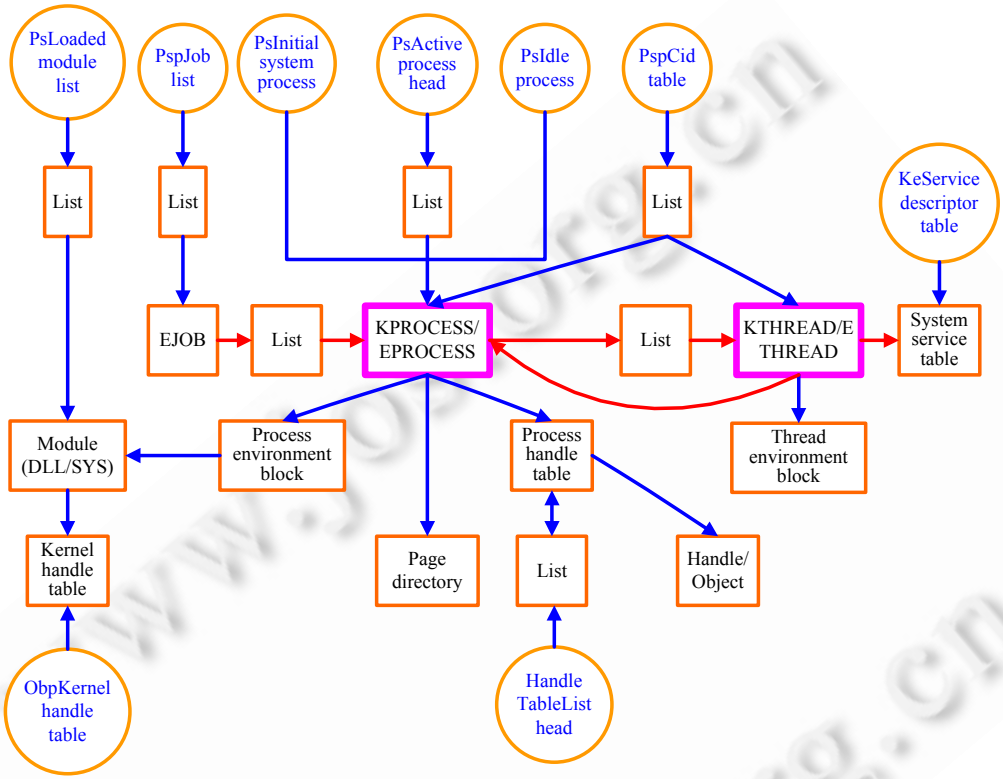


Fig.3 Relationship between process and thread

图 3 进程与线程的关系

由于上述内核数据结构都保存在内存中,这提示可通过内存取证获取相关进程或线程的信息.而系统中发生的任何攻击行为,总是进程(线程)执行的结果.由图 3 可知,选择从一个表头出发,采用顺藤摸瓜的方式,能够遍历所有 Lists,枚举出重要信息.因此,通过分析、提取此类信息,能够获取相关攻击行为证据数据.这也是内存取证的重要依据和理论基础.

3 内存获取方法

内存取证研究的首要问题是如何完整地获取内存数据.完整的内存数据包括两部分:物理内存数据和页面交换文件数据.物理内存通常是一个特殊的内核对象,比如,在 Windows 系统中,物理内存是内核内存区对象,即\\Device\\PhysicalMemory;在 Unix/Linux 系统中,物理内存为/dev/mem 和/dev/kmem.只要能读取该内核对象,就能获取物理内存数据.

目前,获取物理内存数据的方法很多[37,38],一般利用操作系统的相关机制和特性,通过不同方法获取物理内存数据.这些方法可概括为两大类:基于硬件的内存获取和基于软件的内存获取.

3.1 基于硬件的内存获取

根据所面向的硬件体系和操作系统的不同,基于硬件的内存获取方法可分为两类:(1) 面向 Intel 架构桌面操作系统(Windows,Linux,Mac OS 等)的内存获取;(2) 面向 ARM 架构智能终端操作系统(Android,iOS 等)的内

存获取.

(1) 面向 Intel 架构桌面操作系统的内存获取

此类基于硬件的内存获取方法主要通过插入硬件卡,利用 DMA(direct memory access,直接内存访问)指令去获取物理内存的拷贝.在拷贝物理内存时,目标系统的 CPU 将暂停,以避免因操作系统的执行而改变内存数据.目前有 4 种类型的硬件卡:① 基于 PCI 卡拷贝物理内存(hardware card based technique),比如 Carrier 等人提出的 Tribble[39]、Petroni 等人提出的 Copilot[40]、BBN Technologies 提出的 FRED[41]、Hulton 提出的 Cardbus[42]等物理内存提取卡;② 基于 IEEE 1394 火线接口拷贝物理内存(hardware bus based technique),比如,Ruff[43],Bock[44],Boileau[45],Piegdon 等人[46]提出了利用 IEEE 1394 火线接口进行物理内存拷贝;③ 基于固件的物理内存拷贝(firmware based technique),比如 Alessandro 等人[47]提出的 SMMDumper、Wang 等人[48]利用 PCI 网卡中的 SMM(system management mode,系统管理模式)固件获取物理内存和 CPU 寄存器内容;④ 基于网卡拷贝物理内存(network card based technique),比如,Štefan[49]利用网卡的 NDIS(network driver interface specification,网络驱动程序接口规范)来获取物理内存;⑤ 基于雷电接口 Thunderbolt 拷贝物理内存(Thunderbolt based technique),比如,美国的 TALINO[50]数字取证工作室已就 Thunderbolt 接口取证展开了相关研究.

此类方法的优点是:① 在基于 Intel 架构的操作系统(Windows,Linux,Mac OS 等)中,都可以利用该方法获取物理内存;② 采用 DMA 方式能够在不干扰操作系统和 CPU 的情况下,较为准确地获取物理内存拷贝.其缺点是:① 如攻击者对主板上的北桥芯片进行重新编程,则很容易欺骗此类方法;② 早期的 PCI 硬件卡须事先插入主板中,这对于遭受实时攻击的系统来说是不现实的;③ 新版 BIOS 对 SMM 模式进行锁定,限制了利用固件获取物理内存的应用;④ 如果攻击者对 NDIS 库进行劫持,就能篡改通过网卡传送的物理内存数据;⑤ 只能获取物理内存 RAM,不能获取页面交换文件.

(2) 面向 ARM 结构智能终端操作系统的内存获取

面向智能终端的硬件获取方法主要通过硬件与终端设备相连接,以获取智能终端设备的内存数据.目前,基于硬件的内存获取方法主要有两种[51]:① JTAG(joint test action group,联合测试行动小组)技术;② 芯片摘取(chip-off)技术.

- JTAG 技术是通过在印刷电路板上与特定的 JTAG 接头相连以连接智能终端设备中的 CPU,在向电路供电后,再借助 JTAG 软件转储整个内存数据.该方法的优点是无需获得智能终端设备的 Root 访问权限即可实现内存数据获取,但其缺点是,如果 JTAG 焊接点出错或供电电压错误,则将导致 JTAG 无法使用,且对智能终端设备造成严重损坏.
- 芯片摘取(chip-off)技术是将内存芯片从智能终端设备中取出,再通过特殊的硬件设备读取其中存储的数据.该方法的优点是可绕过智能终端设备的口令保护,还能修复已遭毁坏的智能终端设备中的数据;但其缺点是把内存芯片取出和重新装入印刷电路板,将可能损坏内存芯片.

目前,尽管基于硬件的物理内存获取方法仍易受到 Inception 等工具的攻击和存在损坏物理硬件的风险,但因其采用 DMA 方式,无需操作系统和 CPU 的干预,无需智能终端设备 Root 访问权限,减少了对内存数据的改变,从而确保所获内存数据的真实性与实时性.因此我们认为,基于硬件的物理内存获取方法将是未来内存取证的重要研究方向.

3.2 基于软件的内存获取

基于软件的内存获取方法主要利用软件去获取内核内存区对象,并借助于操作系统的内核数据结构和相关机制去解析重构内存数据[52].目前主要有 7 种方法[37]:(1) 基于用户模式程序的内存获取(user level applications);(2) 基于内核模式程序的内存获取(kernel level applications);(3) 基于系统崩溃转储的内存获取(crash dump technique);(4) 基于操作系统注入的内存获取(operating system injection);(5) 基于系统休眠文件的内存获取(hibernation file based technique);(6) 基于系统冷启动的内存获取(cold booting);(7) 基于虚拟机的内存获取(virtualization).

(1) 基于用户模式程序的内存获取方法

该方法通过用户模式的应用程序读取目标系统中内核内存区对象,即 Windows 系统中的\\.\Device\PhysicalMemory 对象、Linux/Unix 系统中的/Dev/mem 和/Dev/Kmem 对象、Android 系统中通过 DDMS(dalvik debug monitor service)模块,以此获取物理内存数据[46].比如,Garner[53]开发的 Data-Dumper、Klein[54]编写的 Process Dumper Utility 等都是此类工具软件.这种物理内存获取方法尽管实现简单,但存在如下缺陷:① 只能用于特定的操作系统和计算机体系结构中;② 该应用程序在内存中的运行会破坏部分内存信息;③ 该方法依赖于系统提供的系统服务,易被 Rootkit 劫持而导致所获取数据失真.

(2) 基于内核模式程序的内存获取方法

该方法通过内核模式驱动程序读取目标系统中内核内存区对象,比如 Windows 系统中的\\.\Device\PhysicalMemory 对象、Linux/Unix 系统中的/Dev/mem 和/Dev/Kmem 对象,以此来获取物理内存数据.相对于用户模式应用程序易受限于操作系统安全机制,内核模式驱动程序位于系统内核层,可轻松绕过系统安全机制而完整读取内存区对象.目前,使用此类方法获取物理内存数据的工具软件很多,比如 Mantech 公司的 Memory DD[55]、MoonSols 公司的 Windows Memory Toolkit[56]、Mandiant 公司的 Memoryze[57]、Guidance Software 公司的 WinEn[58]、AccessData 公司的 Forensic Toolkit[59]、GMG Systems 公司的 KnTDD[60]、HBGary 公司的 Fastdump Pro[61]等.但是,该方法也存在如下缺陷:① 程序一旦在内存中运行,就必定会破坏部分内存信息,因此我们认为,如果能将内存获取作为操作系统一个内核模块,通过按键操作来获取内存镜像,则将有助于内存取证分析;② 如果攻击者采用 DKOM(direct kernel object manipulation,直接内核对象操纵)技术直接修改内存中执行体内核及其所使用的内核对象,则将导致所获取的内存数据失真.

(3) 基于系统崩溃转储的内存获取方法

一般而言,当操作系统崩溃时,系统总是试图记录下有关当前系统的状态的信息,以帮助找出是哪个组件导致了这次系统崩溃[62].这给物理内存获取提供了线索.基于系统崩溃转储的内存获取方法就是利用操作系统的这种机制,通过获取系统崩溃时刻的系统内存纪录而获得物理内存数据.对于 Windows 系统而言,每次崩溃都会在\Windows\Minidump 目录中生成一个文件,只需用 Windbg 来打开该崩溃转储文件进行相关分析,就可解析出其中所蕴含的信息.其实,无需等到系统崩溃也能转储内存,通过 Windows 系统内置的 CrashOnCtrlScroll 功能即可获取崩溃转储内容[63].对于 Linux 系统来说,系统提供了多种内核崩溃内存转储机制,比如 LKCD(Linux kernel crash dump),Diskdump,Netdump,Kdump(kernel dump),MKdump(mini kernel dump)等,一旦系统崩溃开关被触发,则会在当前目录中生成一个 mydumpfile 内存转储文件,再利用 Crash 工具就能分析该内存转储文件.该方法缺点是:因崩溃而对目标系统可能会造成灾难性破坏,从而不利于开展后续相关的取证工作.

(4) 基于操作系统注入的内存获取方法

该方法通过向目标系统注入一个独立的操作系统,再冻结目标系统以获取当前系统快照.目前,采用这种方法实现内存数据获取的工具软件为数不多,只有 Bradley[64]提出了概念验证原型系统 BodySnatcher,用以获取 Windows 2000 下的内存镜像.尽管该方法能够较为精确地获取内存数据,但其缺陷也显而易见:在单处理器模式下,将花费很多时间进行写操作、消耗大量内存、且只支持在串口模式的 I/O 操作.

(5) 基于系统休眠文件的内存获取方法

休眠(hibernate)是指系统将内存中的数据全部转储到硬盘上一个休眠文件中,然后切断对所有设备的供电,并在下次开机时,系统会将硬盘上的休眠文件内容直接读入内存,并恢复到休眠之前的状态.由于休眠文件中保存的是休眠时刻系统的内存记录,因此,系统休眠文件也成了获取内存数据的一种方式.在 Windows 系统中,当系统进入节能休眠模式时,会自动生成休眠文件 Hiberfil.sys 且存放在 Windows 系统根目录下[65].在 Linux 系统中,休眠文件保存在 swap 分区中.在 Mac OS 系统中,休眠文件保存在/var/vm/sleepimage 中.基于系统休眠文件的内存获取方法,就是利用操作系统这种机制来获取内存数据.该方法尽管实现简单,但存在如下缺陷:① 由于系统休眠时保存的物理内存 RAM 中的数据,未包括页面交换文件数据,导致无法获取全部内存数据;② 需要与物理内存 RAM 大小相同的硬盘空间,以存储休眠文件;③ 不同操作系统的休眠文件格式未知,且压缩存放,这

给取证分析该文件带来了困难;④ 该方法除了需要 BIOS 支持以外,还需要外围硬件设备能够支持节电状态,且这些设备驱动必须能够接收电源管理指令.

(6) 基于系统冷启动的内存获取方法

该方法的原理如下:在系统关机后,物理内存 RAM 用于保存数据的电磁信号并没有即刻消失.通过利用液氮冷却相关物理内存条,则其中所保存信息的消磁时间将会延长.在冷启动后,利用专门定制的内核模块就能获取物理内存中残留的信息[66−68].该方法尽管能够获取部分物理内存数据用于事后取证分析,但存在如下缺陷:① 受环境温度影响大,易丢失信息;② 要求操作及时,实现较困难;③ 无法适用于实时内存取证.

(7) 基于虚拟机的内存获取方法

虚拟机技术已经在云计算、灾难恢复和系统安全等领域得到了广泛应用,因此,基于虚拟机的内存获取方法也随之得到迅速的关注与研究.目前,虚拟机的实现可概括为两类[69]:① 基于软件仿真,即,用软件来模拟计算机软硬件环境,通过共享宿主机的部分硬件以及宿主机 CPU 模拟的部分虚拟硬件,建立完整的系统运行环境;② 基于硬件支持,即,利用 Intel-VT 技术或 AMD-V 技术,在 CPU 和芯片组等硬件虚拟化技术支持下建立硬件辅助虚拟机.因此,对于基于虚拟机的内存获取也可分为两类:① 基于软件仿真虚拟机的内存获取;② 基于硬件辅助虚拟机的内存获取.

对于前者,由于在虚拟机中运行的程序,其在虚拟客户机内存中的代码与数据等内容都存储在宿主机的磁盘文件上,所以内存获取方法相对简单、快捷.比如,基于 VMWare 虚拟机在暂停或挂起时,内存状态信息会保存在以.Vmss 为扩展名的文件中;在建立系统快照时,包括内存在内的系统状态信息会保存在以.Vmsn 为扩展名的文件中;而虚拟客户机的所有内存信息会保存在以.Vmem 为扩展名的文件中[70].

对于后者,由于 VMM(virtual machine manager,虚拟机管理器)存在着两种模式:Root 模式和 Non-root 模式,这两种模式的转换须通过 VMCS(virtual machine control structure,虚拟机控制结构)和“客户虚拟地址 GVA→客户物理地址 GPA→宿主虚拟地址 HVA→宿主物理地址 HPA”来实现.因此,该类内存获取方法遵循的原理是:利用 VMCS 结构和相应地址转换机制来获取内存数据.比如,Mariano 等人[71]研究了基于 Intel-VT 技术虚拟机的内存取证框架,并实现了利用 VMCS 进行内存数据获取工具 Actaeon:可在宿主机内存中定位 Hypervisor、分析嵌套虚拟环境配置信息以及在嵌套虚拟环境中运行的各个客户虚拟机间的关系.

以上两类基于虚拟机的内存获取方法存在不足:① 尽管基于软件仿真的虚拟机内存获取方法容易获取内存数据,但由于.Vmem 或.Vmss 等文件格式未知,在进行相应文件格式转换时可能会造成数据丢失,这将给后续的内存数据取证分析带来困难;② 随着诸如 Intel-VT 等硬件辅助虚拟化技术的推出,从 CPU 虚拟化、芯片组虚拟化和网络虚拟化等方面提高了虚拟机的安全性和运行性能,降低了开发虚拟机的难度,然而也给全面完整的内存数据获取带来了困难,尤其在 Intel VT-d 技术推出后.Intel VT-d 技术是一种基于北桥芯片的硬件辅助虚拟化技术,通过在北桥中内置提供 DMA 虚拟化和 IRQ 虚拟化硬件,实现了新型的 I/O 虚拟化方式.Intel VT-d 能够在虚拟环境中极大地提升 I/O 的可靠性、灵活性与性能,但是 Intel VT-d 技术中的 DMA 重映射为设备访问内存提供硬件隔离:通过不同的 I/O 页表,使得每个硬件设备分配到特定的域;当设备尝试访问系统内存时,DMA 拦截访问,并且判断是否允许该访问.

随着云计算等虚拟机技术的迅猛发展,越来越多的应用将转移至虚拟计算环境中.因此我们预计,基于虚拟机的内存获取技术将成为未来内存取证研究的重要领域和发展方向之一.

3.3 内存获取方法的评估

尽管内存获取方法和相关工具较多,但对这些方法和工具的性能评估研究相对较少.为了客观而准确地测评相关内存获取方法和工具的性能,研究不同内存获取方法和工具的评估模型与方法具有重要的理论价值和实际意义.

目前,有关内存获取的评估研究工作已开展,研究者从不同角度提出了对于内存获取的测评理论与原则.从内存镜像的真实性与完整性方面,研究者利用不同方法尝试评估系统在进行内存镜像获取时状态变化的程度.比如,Wang 等人[72]提出了一个概率统计模型,以计算在内存镜像被加载至 RAM 中时内存区域的改变概率;

Walters 等人[73]通过建立快照基线,对比内存获取前后系统状态的变化,尝试去评估内存镜像在获取时的被破坏程度;Sutherland 等人[74]提出了实证内存快照建立时系统状态变化的方法;Inoue 等人[75]提出了一种能揭示内存镜像中系统错误的可视化技术.

此外,为了准确地评估相关内存获取方法的可靠性,研究者也提出了相应的测评原则,即,某种内存获取方法如满足某些测评原则,则认为该方法是可靠的.比如,Schatz[64]提出了 3 个通用的、理想的测评原则——精确性(precision)、可用性(availability)、可靠性(reliability);Bradley[76]提出了类似的 3 个测评原则——精确性(fidelity)、可靠性(reliability)、可用性(availability);Vömel 等人[31,37]提出了内存获取评估平台架构,并提出了 3 个测评原则——正确性(correctness)、原子性(atomicity)、完整性(integrity).

我们认为,无论哪个测评原则,都需要满足内存数据获取的真实性(authenticity)和完整性(integrity).真实性表征通过相关方法或工具获取的某时刻内存镜像与该时刻保存在内存区域中数据完全相同,即,该内存镜像真实地反映了该时刻内存状态.完整性表征所获取的内存镜像的全面性和未被修改性,这里的完整性有两个方面的涵义:① 获取的内存镜像全面完整,既包括物理内存 RAM,也包括页面交换文件;② 获取的内存镜像未被修改,即,通过某方法或工具获取的某时刻内存镜像未被更改.综合相关测评原则,上述内存获取方法在真实性和完整性两个方面的性能测评见表 1.

Table 1 Overview of memory accquisition techniques [37,64]

表 1 内存获取技术概览[37,64]

Technique	Authenticity	Integrity	Comments
Hardware card	High	Moderate	Require pre-installed before capturing memory data
Hardware bus	High	Low	Require pre-installed before capturing memory data
Virtualization	High	High	Have easily access to true and complete memory data
Crash dump	Low	Low	Not all memory is dumped, and can do harm to the operating system
User level application	Low	Low	Easily subverted, will modify memory when capturing it and will not have access to entire memory range
Kernel level application	Moderate	High	Easily subverted and will modify memory when capturing it
Operating system injection	High	Low	Reliance on hardware platforms and very slow
Hibernation	High	Low	Data analysis is very hard
Cold booting	High	Moderate	Need for additional equipment

根据上述内存获取方法的性能比较和相关方法的原理分析,目前多数内存数据获取方法存在数据完整性问题,主要体现在 3 个方面:① 易受 Rootkit 隐遁攻击等内存视图伪装技术的欺骗,致使获取的内存数据不完整;② 基于硬件的内存获取方法未能完整地获取内存数据,完整的内存数据既包括物理内存 RAM 中的数据,还包括页面交换文件中的数据;③ 基于软件的内存获取方法或多或少存在着干扰、破坏内存原始数据的问题,导致获取的内存镜像与获取时刻真实的内存区域数据存在一些细微的差异.因此,从严格意义上说,目前无论采取基于软件还是基于硬件的内存获取方法,都难以全面、完整地获取内存数据.我们预计,解决上述内存数据获取存在的问题,是未来内存取证研究的重要课题之一.

4 内存分析方法

在获取了内存数据之后,就需要对其进行深度分析,解析、重建出内存数据中所蕴含的网络攻击和网络犯罪证据信息.传统的内存数据分析主要采用字符串搜索方法,通过搜索内存中用户名、口令、IP 地址等文本字符串,以获取部分取证辅助信息.尽管该方法操作简单、使用方便,能够提取部分内存信息,但却不能有效分析与网络攻击和网络犯罪相关的进程、注册表、解密密钥、网络连接、可执行文件、系统状态等信息.

为了全面地进行内存数据分析,需依据操作系统内核数据结构和相关机制去解析与重建内存数据所蕴含的信息,进而提取相关网络攻击和网络犯罪证据.目前,内存数据分析研究还处于起步阶段,已有研究可分为 6 种[37]:① 进程信息分析;② 注册表信息分析;③ 密钥恢复分析;④ 网络连接分析;⑤ 可执行文件分析;⑥ 系统状态信息分析.

4.1 进程信息分析

任何网络攻击和网络犯罪行为,总是内存中进程执行的结果.通过内存中进程信息取证分析,有助于还原网络攻击和网络犯罪行为、理解其攻击机理与行为动机.内存进程取证分析主要通过定位内存进程中的相关内核数据结构,并依据进程 EPROCESS 结构和线程 ETHREAD 结构及其相互关系(如图 4 所示),从中提取相关进程信息.目前主要有 3 种方法:① 基于 EPROCESS 对象的取证分析[12,25,27];② 基于 KPCR(kernel processor control region,内核处理器控制域)的取证分析[16,76];③ 基于 VAD 的取证分析[24,27,77−79].

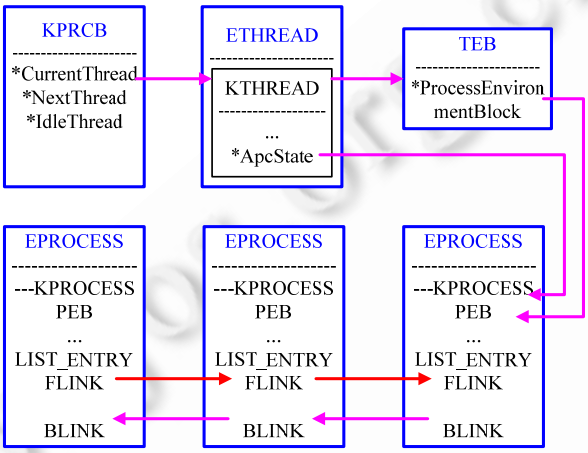


Fig.4 Relationship between EPROCESS and KPRCB

图 4 EPROCESS 与 KPRCB 的关系

基于 EPROCESS 对象的取证分析方法如下:在 Windows 内核中,每个运行的进程都有一个与之相对应的 EPROCESS 结构,以存储与该进程运行相关的各种资源信息.所有活动进程的 EPROCESS 都连接在一起,构成一个双向链表.通过表头 PsActiveProcessHead,就能枚举该链表中的所有进程.该方法的不足是:如果攻击者利用 DKOM 技术修改进程内核对象 EPROCESS 结构中希望隐藏进程的 FLINK 和 BLINK 指针,从而将其从列表中删除,则将无法查找到被隐藏的进程.

基于 KPCR 的内存取证分析是通过 KPCR→KPRCB→_KTHREAD→_KAPC_STATE→EPROCESS 的顺序找到当前执行进程的 EPROCESS 结构,再根据此结构中的 FLINK 和 BLINK 指针遍历 PsActiveProcessList 链表,找到所有的运行进程.此方法与基于 EPROCESS 方法的区别在于查证的起点不同,但殊途同归,因此其缺点也一样:无法查找到利用 DKOM 技术隐藏的进程.

基于 VAD 的内存取证分析方法如下:对于每个进程,Windows 内存管理器维护了一组 VAD,由它们来描述该进程地址空间的状态.Windows 使用平衡二叉树来管理 VAD 对象.进程 EPROCESS 结构中的 VadRoot 域指向此树的根.通过该树中的 MMVAD 类型节点,能够得到一个重要的指向控制区对象的指针 ControlArea.而控制区对象与文件对象、内存区对象都是互指的,因此通过该指针顺藤摸瓜,能够获取与当前执行进程相关的可执行文件信息.该方法的不足是:无法查证到利用 DKOM 技术隐藏的 VAD 树节点信息.

4.2 注册表信息分析

注册表是一个 Windows 系统中各类配置信息的数据库,保存有大量与系统和用户相关的信息,比如系统中运行的进程列表、用户和系统交互过程等.此外,注册表的内存镜像中还包含只在系统启动时创建的、系统关机后便不复存在的配置单元信息.因此,注册表的内存镜像也是取证网络攻击与网络犯罪证据的一座金矿[22].

Windows 注册表在逻辑上是以树形层次结构组织的,在物理上是由一组 HIVE 文件构成的.一个 HIVE 文件由多个巢箱(BIN)组成,HIVE 文件的首部有一个文件头(基本块 base block),用于描述这个 HIVE 文件的一些全

局信息.一个 BIN 由多个巢室(CELL)组成,用于存储不同的注册表数据(如图 5 所示).

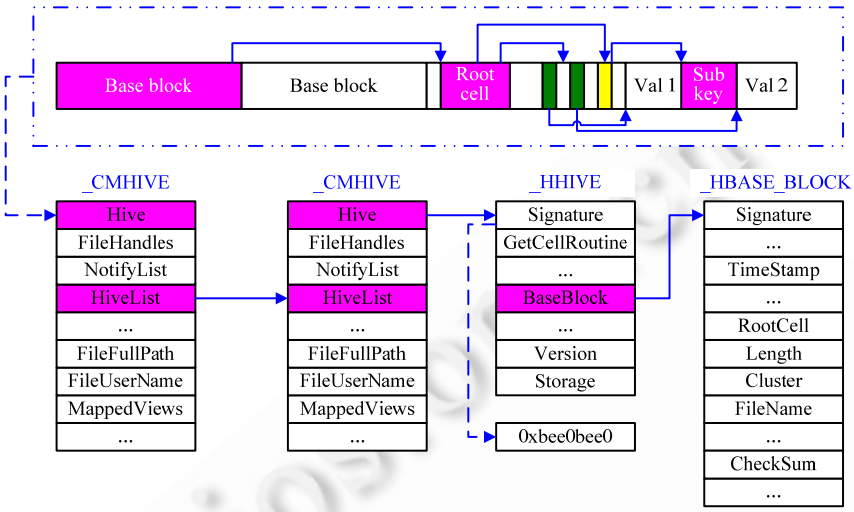


Fig.5 Structure of a registry Hive

图 5 注册表 Hive 的结构

注册表包括 5 种类型单元:健(key)单元、值(value)单元、子健列表(subkey list)单元、值列表(value list)单元、安全描述符(security descriptor)单元.其中,健单元的特征符是 kn,值单元的特征符是 kv,安全描述符单元的特征符是 ks.在内存取证分析中可利用这些特征符,在获取的内存映像和页面交换文件中找寻到有关注册表键值信息.Windows 配置管理器使用类似于 Intel x86 处理器的页表映射方式来解决 Cell 地址转译,如图 6 所示.

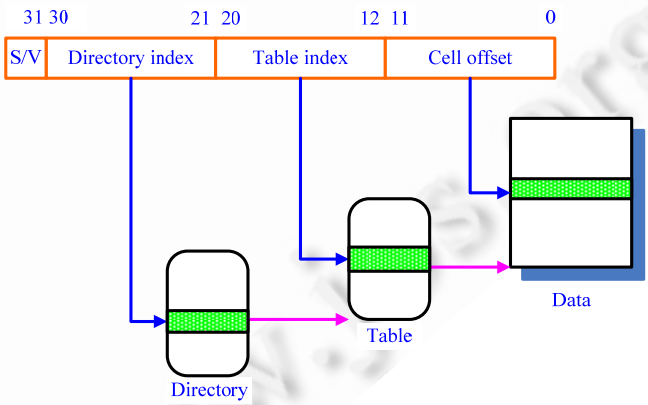


Fig.6 Structure of registry cell index

图 6 注册表巢室索引结构

内存注册表信息取证分析主要是通过相关特征符定位内存中的注册表数据结构,并从中提取有关注册表键值等信息.比如,Anand 利用 WinDbg 研究了将 Cell index 转换为虚拟地址的方法[80];Brendan 研究了 Windows 配置管理器对注册表的低层操作方法,对内存取证注册表信息作出了突出贡献[81−84];Stevens[85]从中提取出最近运行程序列表;Carvey[86,87]从中提取出最近添加到系统中的设备记录、无线网络连接信息.该方法的不足在于无法查证利用 DKOM 技术、通过修改 HiveList 指针而隐藏的 Hive 结构.

4.3 密钥恢复分析

密码技术的深度应用,确保了网络系统文件安全.对于密码技术,密码算法是公开的,但相关密钥却是保密的,甚至是加密的.对于已加密的密钥,通常会在程序运行时在内存中进行自动解密.因此,要想解密相关的应用系统,就需从内存中获取相关密钥信息[88].

内存取证密钥恢复分析主要通过解析相关密码算法的内存存储格式,从中还原出密钥信息.目前,关于密钥恢复分析已经开展,研究者从不同角度提出了不同内存密钥取证方法.比如,Hargreaves 和 Chivers[89]提出了利用线性内存扫描法获取密钥信息,该方法无需深入理解底层系统,简单易行,但对于分段存放的密钥却无能为力;Klein[90]利用私钥和证书通常以标准格式存储的特征,提出了从内存进程信息中恢复密钥的方法;Kaplan[91]基于密钥仅存于物理内存的实际情形,提出了从非换页内存中提取密钥的方法;Halderman,Muller 等人[66,92]提出了利用冷启动从内存提取密钥的方法;Tsow,Maartmann-Moe[93,94]分别提出了从内存提取密钥的算法.

4.4 网络连接分析

网络攻击和网络犯罪行为通常会产生网络连接信息,比如 IP 地址、端口号等.使用诸如 TCPView,Netstat 等工具,就能获取相关网络连接信息.然而,很多与网络连接相关的数据一般只存在于内存中,而且此类数据会随系统中进程运行而不断更替、消失.因此,进行内存网络连接信息取证[21],及时而有效地提取有关网络攻击和网络犯罪的辅助证据,将有助于深入了解网络攻击和网络犯罪真实场景.

在内存中进行网络连接信息分析主要通过分析 Windows 系统的网络驱动程序 Tcpip.sys,从中提取相关网络连接信息.为提取 TCP 和 UDP 网络连接信息,需解析存储在 Tcpip.sys 中的两个重要结构:_TCPT_OBJECT 和 _ADDRESS_OBJECT(如图 7 所示).

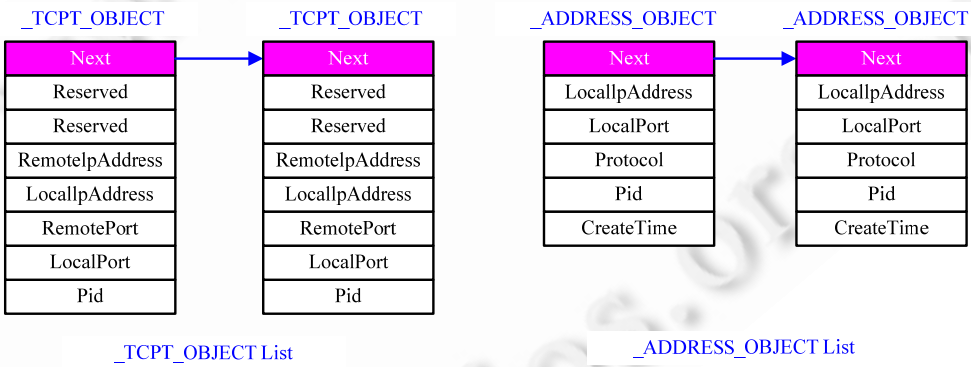


Fig.7 Structure of _TCPT_OBJECT and _ADDRESS_OBJECT list

图 7 _TCPT_OBJECT 和_ADDRESS_OBJECT 链表的结构

目前,取证研究者已提出了一些网络连接取证分析方法,比如,Schuster[23]基于非换页内存池的相关特征,通过扫描非换页内存池和逆向 Tcpip.sys 驱动程序,提出了一种提取监听套接字列表进而获取相关网络连接信息的算法;Ligh[95],Okolica 等人[96]提出了遍历 Tcpip.sys 中的_AddrObjTable 和_TCBTable 列表获取相关网络连接信息的方法(如图 8 所示).

然而在 Windows vista/Win 7/Win 8 系统中,网络驱动程序 Tcpip.sys 中却没提供_TCPT_OBJECT 和 _ADDRESS_OBJECT 这两个重要结构,因此需要通过其他方法进行网络连接的内存取证分析.Wang 等人[97]提出了基于 Tcpip.sys 中的数据结构 TcpEndpointPool 来提取网络连接信息的方法.

此类借助链表来获取网络连接信息方法的缺点是易被 DKOM 技术欺骗,即,通过修改_TCPT_OBJECT 和 _ADDRESS_OBJECT 结构中 Next 指针或 TcpEndpointPool 结构中的 FLINK 和 BLINK 指针来隐藏相关对象.

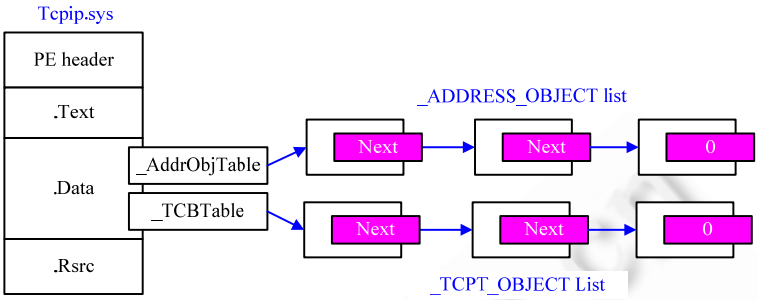


Fig.8 Enumerating of network socket object

图 8 网络套接字对象枚举

4.5 文件分析

网络攻击内存化已成网络犯罪的发展趋势,最主要的表现是无磁盘文件,目标系统关机后,攻击进程所对应的可执行文件映像自我删除.因此,重建并分析与网络攻击或网络犯罪进程相对应的可执行文件,是理解网络攻击或网络犯罪的关键.此外,当实施网络攻击或网络犯罪的进程运行时通常会加载其他模块、打开相关文件,而分析这些加载模块、打开文件,也成为理解网络攻击或网络犯罪的重要线索.因此,这里的文件分析主要是指与攻击进程相对应的可执行文件、进程加载的模块文件、进程读写的文件.

内存文件取证分析主要利用 Windows 系统的进程结构 EPROCESS、虚拟地址描述符 VAD 及其他内核结构所蕴含的信息及其相互关系信息(如图 9 所示),提取与该进程相关的文件信息.目前,内存文件取证分析可概括为两类方法:① 基于进程对象 EPROCESS 提取相关文件信息;② 基于进程虚拟地址描述符 VAD(virtual address descriptor)还原相关可执行文件信息.

- 基于进程对象 EPROCESS 可提取的文件信息主要包括读写的文件名及路径、加载的 DLL 模块及其基址等,比如 Schuster[98,99]提出的利用 EPROCESS 提取相关文件信息的方法.但此类方法易被 Rootkit 隐遁攻击所欺骗.
- 基于 VAD 可获取与当前执行进程相关的可执行文件信息,比如,Dolan-Gavitt[56]首先提出并实现了基于 VAD 的文件信息提取方法;Schuster[99],Kornblum[100],Van Baar 等人[101]相继提出了各种重建可执行文件的方法.该方法同样无法查证到利用 DKOM 技术隐藏的 VAD 树节点信息.

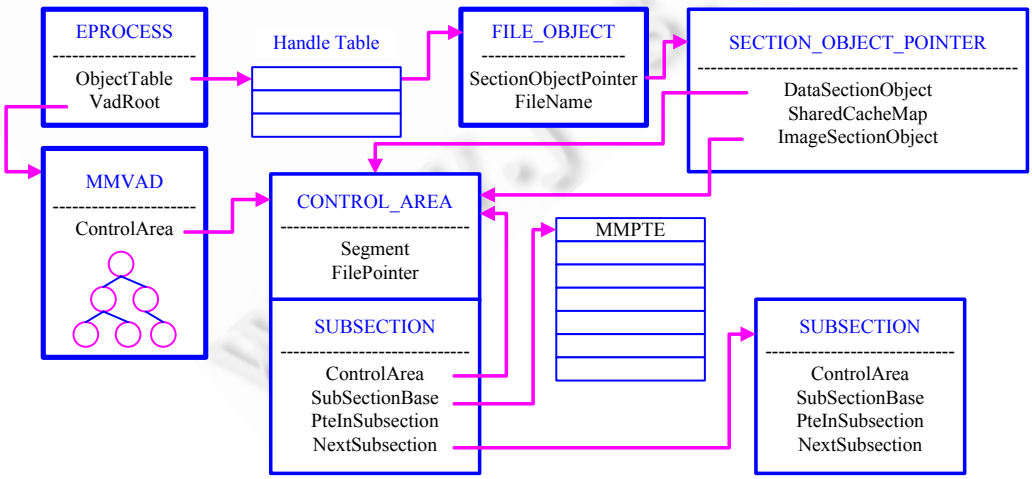


Fig.9 Relationship between EPROCESS, VAD and other objects

图 9 EPROCESS,VAD 与其他对象的关系

4.6 系统状态信息分析

系统状态信息是 Windows 系统在运行相关进程时必须保存的有关该进程运行状态的信息,包括 CPU 寄存器信息、进程自身信息、系统缓存中的信息等.此类信息可使取证调查者了解当前系统中软/硬件和网络的运行状态,对于理解网络攻击和网络犯罪有重要的参考价值与辅助意义.

由于系统状态信息涵盖较为广泛的信息范围,导致所采用的分析方法相对较多.比如,Schuster[78],Dolan-Gavitt[102]等人利用进程对象 EPROCESS 提取出有关进程启动退出时间、进程句柄表、进程在内存映像中的基地址、进程文件名、进程环境块 PEB、进程 VAD 等;Stevens 等人[103]提出了从内存中提取 Windows 命令行历史记录信息的方法;Simon[104],Gao[105],Markus[106],Drinkwater[107]等人相继研究了从内存中恢复相关应用软件数据的方法;Okolicia[108]等人提出了从内存中提取 Windows 剪贴板内容的方法;Zhao[88],Hejazi[109]等人从内存中取证各类敏感数据.

5 内存取证讨论

随着网络攻击和网络犯罪技术的发展,内存取证作为计算机取证研究的重要分支,对于获取、分析内存数据、提取网络攻击和网络犯罪证据、重构网络攻击和网络犯罪场景具有重要理论价值和实用价值.依据上述分析和掌握的最新资料,我们将总结目前内存取证存在的问题、展望内存取证发展趋势、探讨内存取证进一步研究方向.

5.1 内存取证现有问题

尽管内存取证研究已取得了长足发展与广泛应用,但现有内存取证技术目前仍存在如下不足与缺陷[31,37].

(1) 内存数据获取的完整性

“巧妇难为无米之炊”.内存数据获取始终是内存取证的首要问题,全面、完整的内存数据则是内存取证分析的关键.随着虚拟机技术的发展和云计算平台的兴起,虚拟机内存数据已成为内存取证和数字调查必不可少的证据源.但目前的内存取证工具与方法多侧重于原始主机的物理内存数据和页面交换文件数据的获取,而对搭载于原始主机之上的虚拟机内存数据的获取则未予以足够的重视.

(2) 内存数据分析的完备性

在完整获取内存数据之后,需依据相关环境与机制进一步深入解析、还原内存数据所蕴含的信息,以完备提取网络攻击和网络犯罪相关证据.目前的内存取证工具与方法在内存数据分析方面存在着完备性问题,具体表现在 3 个方面:① 根据前面的分析可知,内存数据获取方面存在完整性问题,致使内存分析所需数据不全面,从而导致内存数据分析同样不全面、不完备;② 未能从系统不同层次分析内存镜像数据,对于利用 DKOM 等内存视图伪装技术的隐遁网络攻击或网络犯罪,无法有效解析出相关进程信息;③ 对于无磁盘文件仅存在于内存中的隐遁网络攻击或网络犯罪,未能有效地重建与攻击进程相对应的可执行文件映像,导致攻击证据提取不全.

(3) 内存证据存储的通用性

目前,内存证据数据存储格式多且难以相互转换.尽管数字取证研究工作组 DFRWS 于 2006 年开展了通用证据存储格式的研究,但因缺乏相关的存储格式源支持,导致该工作组终止相关研究工作.

(4) 内存取证研究的滞后性

从攻防博弈的角度,内存取证属于防御技术的范畴.与攻击技术快速发展相比,内存取证针对新类型和新平台的网络攻击和网络犯罪的威胁而进行的研究与开发仍具有一定的滞后性.

5.2 内存取证发展趋势

内存取证技术随着网络攻击和网络犯罪技术的发展而不断发展与演化.

- 在内存数据获取方面,除了完整地获取内存数据(包括物理内存数据和页面交换文件数据)之外,还需确保所获内存数据的真实性和实时性.

- 在内存数据分析方面,如何防止 DKOM 等内存视图伪装技术欺骗、完备分析内存中的可知性文件映像以满足内存数据分析的完备性?此外,目前内存取证分析所得的证据数据存储格式不统一,造成证据数据相互转换、传输困难.这些仍是内存取证未来发展亟需解决的问题.
- 在内存取证技术开发方面,需采用模块化、可扩展的内存取证框架,以利于内存取证研究深入、快速发展.
- 在内存取证技术应用方面,针对新类型和新平台的网络安全威胁的内存取证技术,比如针对云计算平台和智能手机平台的内存取证技术,仍是计算机取证社区需要继续发展与解决的问题.

5.3 内存取证进一步研究方向

综合上述讨论与分析,我们认为,内存取证技术领域的进一步研究方向包括:

(1) 内存数据的完整获取

内存数据的完整获取是内存取证研究的首要问题.随着虚拟机技术和云计算平台的迅猛发展,虚拟机内存数据已成为内存取证和数字调查必不可少的证据源.因此,包括原始主机和虚拟机的内存数据完整获取,将是未来内存取证领域进一步研究的重要课题.

(2) 内存数据标准格式存储与整合的研究与开发

为了便于内存证据数据间相互转换、传输,未来需建立内存证据数据标准存储格式:存储证据数据及其相关联的元数据,以减少内存取证时间,增加内存取证可用的数据量,增加内存证据的可靠性.在这方面,Volatility 2.4 通过支持多种内存数据存储格式,比如 Windows 系统的 Crash Dump 崩溃转储文件、Hibernation 休眠文件,Mac 系统的 MachO 文件,VMware 的 Snapshot 快照文件,Linux 系统的 LiME 文件等,在同一框架中整合了这些内存数据存储格式.因此我们相信:随着内存数据获取途径的多样化,内存数据格式的整合与标准存储将是未来内存取证领域进一步的研究方向.

(3) 虚拟计算环境和智能手机平台的内存取证分析

随着云计算和智能手机的快速发展与广泛应用,很多应用业务将迁移至云虚拟计算环境和智能手机平台中,也导致其中会发生更多的网络攻击和网络犯罪会.这将从客观上促使针对虚拟计算环境和智能手机平台的内存取证分析技术发展.

(4) 模块化、图形化、可扩展的内存取证框架的研究与开发

采用模块化、可扩展的内存取证框架,将使研究者在标准的内存取证平台上针对不同应用研究开发不同取证插件,快速提取相应内存证据.而友好的图形界面使操作者易于理解原理、快速掌握内存取证方法,便于内存取证分析技术的推广应用.在这方面,Volatility Framework,FATKit 等已在内存取证的框架化、模块化方向进行了相关研究工作,但在内存取证的图形化方向的研究还不多.我们相信,内存取证的模块化、图形化、框架化将是未来内存取证研究的重要方向.

6 结束语

内存取证作为计算机取证科学的一个重要分支,在预防网络攻击、调查网络犯罪等方面有重要且不可替代的作用和应用前景,已成为信息安全研究者所关注的热点研究领域.本文从技术的角度探讨了内存取证原理、内存数据获取与分析方法,并在此基础上展望了内存取证的未来发展趋势和进一步的研究方向.从本质上分析,内存取证技术沿用的是攻防博弈的对抗性思维,因此,内存取证技术将随着网络安全威胁的演化而不断发展与更新.

致谢 我们向对本文工作给予支持和建议的同行,尤其是美国 Sam Houston State University 的 Lei Chen 博士、Bing Zhou 博士、Lijen Shannon 博士表示感谢.

References:

[1] Kessler GC. Advancing the science of digital forensics. Computer, 2012,45(12):25−27. [doi: 10.1109/MC.2012.399]

[2] Stuttgen J, Cohen M. Anti-Forensic resilient memory acquisition. Digital Investigation, 2013,10:S105−S115. [doi: 10.1016/j.diin.2013.06.012]

[3] Blunden B. The Rootkit Arsenal: Escape and Evasion in the Dark Corners of the System. Burlington: Jones & Bartlett Learning, 2013.

[4] Solomon J, Huebner E, Bem D, Szeynska M. User data persistence in physical memory. Digital Investigation, 2007,4(2):68−72. [doi: 10.1016/j.diin.2007.03.002]

[5] DFRWS 2005 memory analysis challenge. 2014. http://www.dfrws.org/2005/challenge/index.shtml

[6] Regional computer forensics laboratory. 2014. http://www.rcfl.gov/

[7] New DARPA program seeks to reveal backdoors and other hidden malicious functionality in commercial IT devices. 2014. http://www.darpa.mil/NewsEvents/Releases/2012/11/30.aspx

[8] Report cyber incidents. 2014. http://www.dhs.gov/how-do-i/report-cyber-incidents

[9] Sang T. Research and development of memory forensics tools [MS. Thesis]. Shanghai: Shanghai Jiaotong University, 2013 (in Chinese with English abstract).

[10] Liu Y. Research on Windows-based memory data acquisition and forensics [MS. Thesis]. Chengdu: University of Electronic Science and Technology of China, 2012 (in Chinese with English abstract).

[11] Wang F. Research on memory volatile data forensics analysis based on Windows [MS. Thesis]. Changchun: Jilin University, 2012 (in Chinese with English abstract).

[12] Chen L, Jing K, Dong ZX, Tian QY. Searching physical memory method based on EPROCESS characteristics. Journal of Chongqing University of Posts and Telecommunications, 2013,25(1):122−125 (in Chinese with English abstract).

[13] Panshi computer live forensics system. 2014. http://www.pansafe.com/

[14] Yamei paike forensics master. 2014. http://300188.cn/subjectview/viewSubjectCategory?cid=2&sortsid=126

[15] Computer crimes online forensics system. 2014. http://news.xinhuanet.com/legal/2011-01/12/c_13687726.htm

[16] Guo M, Wang LH. Windows physical memory analysis method based on KPCR structure. Computer Engineering and Applications, 2009,45(18):74−77 (in Chinese with English abstract).

[17] Memory forensics system. 2014 (in Chinese with English abstract). http://www.most.gov.cn/dfkj/cq/zxdt/201309/t20130905_109157.htm

[18] Kornblum J. Preservation of fragile digital evidence by first responders. In: Proc. of the 2002 Digital Forensic Research Workshop. 2002.

[19] MemParser. 2014. http://www.dfrws.org/2005/challenge/betzReport.shtml

[20] KNTList. 2014. http://www.dfrws.org/2005/challenge/rossettoecioccolato-DFRWSChallengeOverview.pdf

[21] Eoghan C. Network traffic as a source of evidence: Tool strengths, weaknesses, and future needs. Digital Investigation, 2004,1(1):28−43. [doi: 10.1016/j.diin.2003.12.002]

[22] Carvey H. The Windows registry as a forensic resource. Digital Investigation, 2005,2(3):201−205. [doi: 10.1016/j.diin.2005.07.003]

[23] Andreas S. Pool allocations as an information source in Windows memory forensics. In: Proc. of the Int'l Conf. on IT-Incident Management & IT-Forensics. 2006.

[24] Dolan-Gavitt B. The VAD tree: A process-eye view of physical memory. Int'l Journal of Digital Forensics and Incident Response, 2007,4(1):62−64. [doi: 10.1016/j.diin.2007.06.008]

[25] Kornblum J. Using every part of the buffalo in Windows memory analysis. Journal of Digital Investigation, 2007,4:24−29. [doi: 10.1016/j.diin.2006.12.002]

[26] Carrier B, Spafford EH. Getting physical with the digital investigation process. Int'l Journal of Digital Evidence, 2003,2(2):1−20.

[27] Andreas S. Searching for processes and threads in Microsoft Windows memory dumps. Journal of Digital Investigation, 2006,3S:S6−S10. [doi: 10.1016/j.diin.2006.06.010]

[28] Dolan-Gavitt B. Forensic analysis of the Windows registry in memory. Digital Investigation, 2008,5:S26−S32. [doi: 10.1016/j.diin.2008.05.003]

[29] Arasteh AR, Debbabi M. Forensic memory analysis: From stack and code to execution history. Digital Investigation, 2007,4S:S114−S125. [doi: 10.1016/j.diin.2007.06.010]

[30] Petroni N, Walters A, Fraser T, Arbaugh W. FATKit: A framework for the extraction and analysis of digital forensic data from volatile system memory. Digital Investigation, 2006,3(4):197−210. [doi: 10.1016/j.diin.2006.10.001]

[31] Voemel S, Stuettgen J. An evaluation platform for forensic memory acquisition software. Digital Investigation, 2013,10:S30−S40. [doi: 10.1016/j.diin.2013.06.004]

[32] PTFinder. 2014. http://computer.forensikblog.de/en/2007/11/ptfinder-version-0305.html

[33] PoolTools. 2014. http://computer.forensikblog.de/en/2007/11/pooltools-version-130.html

[34] Virtual address descriptor tools. 2014. http://vadtools.sourceforge.net/

[35] The volatility framework. 2014. http://www.volatilityfoundation.org/

[36] Pan AM. The Principle and Implementation of Windows Kernel. Beijing: Publishing House of Electronics Industry, 2010 (in Chinese).

[37] Vömel S, Freiling FC. A survey of main memory acquisition and analysis techniques for the Windows operating system. Digital Investigation, 2011,8:3−22. [doi: 10.1016/j.diin.2011.06.002]

[38] Osborne G. Memory Forensics: Review of Acquisition and Analysis Techniques. Defense Science and Technology Organization, 2014.

[39] Carrier BD, Grand J. A hardware-based memory acquisition procedure for digital investigations. Journal of Digital Investigation, 2004,1(1):50−60. [doi: 10.1016/j.diin.2003.12.001]

[40] Petroni NL, Fraser T, Molina J, Arbaugh WA. Copilot—A coprocessor-based kernel runtime integrity monitor. In: Proc. of the 13th USENIX Security Symp. 2004.

[41] BBN Technologies. Fred: Forensic RAM extraction device. 2006. http://www.ir.bbn.com/vkawadia/

[42] David H. Cardbus bus-mastering: Owning the Laptop. In: Proc. of the ShmooCon 2006. Washington, 2006.

[43] Ruff N. Windows memory forensics. Journal in Computer Virology, 2008,4(2):83−100. [doi: 10.1007/s11416-007-0070-0]

[44] Bock B. Firewire-Based physical security attacks on Windows 7, EFS and BitLocker. 2009. http://www.securityresearch.at/publications/windows7_firewire_physical_attacks.pdf

[45] Boileau A. Hit by a bus: Physical access attacks with firewire. In: Proc. of the Ruxcon. 2006.

[46] Ruff N. Windows memory forensics. Journal in Computer Virology, 2008,4(2):83−100. [doi: 10.1007/s11416-007-0070-0]

[47] Reina A, Fattori A, Pagani F, Cavallaro L, Bruschi D. When hardware meets software: A bulletproof solution to forensic memory acquisition. In: Proc. of the ACSAC 2012. 2012. [doi: 10.1145/2420950.2420962]

[48] Wang J, Zhang FW, Sun K, Stavrou A. Firmware-Assisted memory acquisition and analysis tools for digital forensics. In: Proc. of the SADFE. 2011. 1−5. [doi: 10.1109/SADFE.2011.7]

[49] Balogh Š. Memory acquisition by using network card. Journal of Cyber Security, 2014,3(1):65−76. [doi: 10.13052/jcsm2245-1439.314]

[50] TALINO forensic workstations. 2014. http://sumuri.com/product-category/talino-forensic-workstations/

[51] Hoog A. Android Forensics: Investigation, Analysis and Mobile Security for Google Android. Syngress Publisher, 2011.

[52] Microsoft Corporation. Device physical memory object. 2014. https://msdn.microsoft.com/en-us/library/ms804008.aspx

[53] Garner GM. Forensic acquisition utilities. 2014. http://gmgsystemsinc.com/fau/

[54] Klein T. Process dumper. 2006. http://www.trapkit.de/research/forensic/pd/index.html

[55] Memory dd. ManTech CSI. 2014. http://sourceforge.net/projects/mdd/files/

[56] MoonSols. Windows memory toolkit. 2010. http://moonsols.com/product

[57] Mandiant. Memoryze. 2010. http://www.mandiant.com/products/free_software/memoryze/

[58] Guidance Software. Encase enterprise platform. 2010. http://www.guidancesoftware.com/computer-forensics-fraudinvestigation-software.htm

[59] AccessData. Forensic toolkit. 2014. http://accessdata.com/product-download/digital-forensics/forensic-toolkit-ftk-version-5.6

[60] Knttools with Kntlist. GMG systems. 2007. http://gmgsystemsinc.com/knttools/

[61] HBGary. Fastdump—A memory acquisition tool. 2009. https://www.hbgary.com/products-services/fastdump/

[62] Microsoft Corporation. Windows feature lets you generate a memory dump file by using the keyboard. 2014. http://support.microsoft.com/kb/244139

[63] Microsoft Corporation. Overview of memory DUMP file options for Windows server 2008. 2014. http://support.microsoft.com/kb/969028

[64] Schatz B. Bodysnatcher: Towards reliable volatile memory acquisition by software. Digital Investigation, 2007,4S:S126−S134. [doi: 10.1016/j.diin.2007.06.009]

[65] Russinovich ME, Solomon DA, Ionescu A. Microsoft Windows Internals. 5th ed., Microsoft Press, 2009.

[66] Halderman JA, Schoen SD, Heninger N, Clarkson W, Paul W, Calandrino JA. Lest we remember: Cold-Boot attacks on encryption keys. Communications of the ACM, 2009,52(5):91−98. [doi: 10.1145/1506409.1506429]

[67] Vidas T. Volatile memory acquisition via warm boot memory survivability. In: Proc. of the 43rd Hawaii Int'l Conf. on System Sciences. 2010. [doi: 10.1109/HICSS.2010.439]

[68] Chan EM, Carlyle JC, David FM, Farivar R, Campbell RH. Bootjacker: Compromising computers using forced restarts. In: Proc. of the 15th ACM Conf. on Computer and Communications Security. 2008. [doi: 10.1145/1455770.1455840]

[69] Smith JE, Nair R. The architecture of virtual machines. Journal of Computer, 2005,38(5):32−38. [doi: 10.1109/MC.2005.173]

[70] VMware Incorporation. What files make up a virtual machine? 2014. https://www.vmware.com/support/ws5/doc/ws_learning_files_in_a_vm.html

[71] Graziano M, Lanzi A, Balzarotti D. Hypervisor memory forensics. Lecture Notes in Computer Science, 2013,8145:21−40. [doi: 10.1007/978-3-642-41284-4_2]

[72] Wang LH, Zhang RC, Zhang SH. A model of computer live forensics based on physical memory analysis. In: Proc. of the Int'l Conf. on Information Science and Engineering (ICISE). 2009. 4647−4649. [doi: 10.1109/ICISE.2009.69]

[73] Walters A, Petroni NL. Valatools: Integrating volatile memory forensics into the digital investigation process. In: Proc. of the Black Hat DC. 2007.

[74] Sutherland I, Evans J, Tryfonas T, Blyth A. Acquiring volatile operating system data tools and techniques. ACM SIGOPS Operating Systems Review, 2008,42(3):65−73. [doi: 10.1145/1368506.1368516]

[75] Inoue H, Adelstein F, Joyce RA. Visualization in testing a volatile memory forensic tool. Digital Investigation, 2011,8(1):S42−S51. [doi: 10.1016/j.diin.2011.05.006]

[76] Schatz B. Finding object roots in vista (KPCR). 2010. http://blog.schatzforensic.com.au/2010/07/finding-object-roots-in-vista-kpcr/

[77] Dolan-Gavitt B, Srivastava A, Traynor P, Giffin J. Robust signatures for kernel data structures. In: Proc. of the 16th ACM Conf. on Computer and Communications Security. 2009. [doi: 10.1145/1653662.1653730]

[78] Schuster A. Pool allocations as an information source in windows memory forensics. In: Proc. of the IT-incident Management & IT-forensics. 2006. 15−22. [doi: 10.1016/j.diin.2008.05.007]

[79] Schuster A. The impact of Microsoft Windows pool allocation strategies on memory forensics. Digital Investigation, 2008,5(S1):S58−S64. [doi: 10.1016/j.diin.2008.08.002]

[80] Anand G. Internal structures of the windows registry. 2008. http://blogs.technet.com/ganand/archive/2008/01/05/internalstructures-of-the-windows-registry.aspx

[81] Dolan-Gavitt B. Cell index translation. 2008. http://moyix.blogspot.com/2008/02/cell-index-translation.html

[82] Dolan-Gavitt B. Enumerating registry hives. 2008. http://moyix.blogspot.com/2008/02/enumerating-registry-hives.html

[83] Dolan-Gavitt B. Reading open keys. 2008. http://moyix.blogspot.com/2008/02/reading-open-keys.html

[84] Dolan-Gavitt B. Forensic analysis of the Windows registry in memory. Digital Investigation, 2008,5(1):S26−S32.

[85] Stevens D. UserAssist. 2006. http://blog.didierstevens.com/programs/userassist/

[86] Carvey H. Registry mining. 2005. http://windowsir.blogspot.com/2005/01/registry-mining.html

[87] Carvey H. Windows Forensic Analysis. Norwell: Syngress Press, 2007.

[88] Zhao Q, Cao T. Collecting sensitive information from Windows physical memory. Journal of Computers, 2009,4(1):3−10. [doi: 10.4304/jcp.4.1.3-10]

[89] Hargreaves C, Chivers H. Recovery of encryption keys from memory using a linear scan. In: Proc. of the 2008 3rd Int'l Conf. on Availability, Reliability and Security. 2008.

[90] Klein T. All your private keys are belong to use extracting RSA private keys and certificates from process memory. 2006. http://trapkit.de/research/sslkeyfinder/keyfinder_v1.0_20060205.pdf

[91] Kaplan B. Ram is key extracting disk encryption keys from volatile memory [MS. Thesis]. Carnegie Mellon University, 2007.

[92] Muller T, Dewald A, Freiling FC. Aesse: A cold-boot resistant implementation of AES. In: Proc. of the 3rd European Workshop on System Security. 2010. [doi: 10.1145/1752046.1752053]

[93] Tsow A. An improved recovery algorithm for decayed as key schedule images. Lecture Notes in Computer Science, 2009,5867: 215−230. [doi: 10.1007/978-3-642-05445-7_14]

[94] Maartmann-Moe C, Thorkildsen SE, Arnes A. The persistence of memory: Forensic identification and extraction of cryptographic keys. Digital Investigation, 2009,6(S1):S132−S40. [doi: 10.1016/j.diin.2009.06.002]

[95] Ligh M, Adair S, Hartstein B, Richard M. Malware Analyst's Cookbook and DVD: Tools and Techniques for Fighting Malicious Code. Wiley Press, 2010.

[96] Okolica J, Peterson GL. Windows operating systems agnostic memory analysis. Digital Investigation, 2010,7(1):S48−S56. [doi: 10.1016/j.diin.2010.05.007]

[97] Wang LH, Xu LJ, Zhang SH. A method on extracting network connection information from 64-bit Windows 7 memory images. China Communications, 2010,6:44−51.

[98] Schuster A. Dump file structure. 2006. http://computer.forensikblog.de/en/2006/03/dmp_file_structure.html

[99] Schuster A. Reconstructing a binary. 2006. http://computer.forensikblog.de/en/2006/04/reconstructing_a_binary.html

[100] Kornblum J. Recovering executables with Windows memory analysis. 2007. http://www.jessekornblum.com/presentations/ dodcc07.pdf

[101] Van Baar R, Alink W, Van Ballegooij A. Forensic memory analysis: Files mapped in memory. In: Proc. of the Digital Forensic Research Workshop (DFRWS). 2008. [doi: 10.1016/j.diin.2008.05.014]

[102] Dolan-Gavitt B. Linking processes to users. 2008. http://moyix.blogspot.com/2008/08/linking-processes-to-users.html

[103] Stevens RM, Casey E. Extracting Windows command line details from physical memory. Digital Investigation, 2010,7(1):S57−S63. [doi: 10.1016/j.diin.2010.05.008]

[104] Simon M, Slay J. Recovery of skype application activity data from physical memory. In: Proc. of the Int'l Conf. on Availability, Reliability and Security (ARES). 2010. [doi: 10.1109/ARES.2010.73]

[105] Gao Y, Cao T. Memory forensics for QQ from a live system. Journal of Computers, 2010,5(4):541−548. [doi: 10.4304/jcp.5.4.541-548]

[106] Huber M, Mulazzani M, Leithner M. Social snapshots: Digital forensics for online social networks. In: Proc. of the Annual Conf. on Computer Security Applications. 2011. [doi: 10.1145/2076732.2076748]

[107] Drinkwater R. Facebook chat forensics. 2009. http://forensicsfromthesausagefactory.blogspot.com/2009/03/facebook-chat-forensics.html

[108] Okolicia J, Peterson GL. Extracting the Windows clipboard from physical memory. Digital Investigation, 2011,8:S118−S124. [doi: 10.1016/j.diin.2011.05.014]

[109] Hejazi SM, Debbabi M, Talhi C. Automated Windows memory file extraction for cyber forensics investigation. Journal of Digital Forensic Practice, 2008,2(3):117−131. [doi: 10.1080/15567280802552829]

附中文参考文献:

[9] 桑厅.内存取证工具的研究与实现[硕士学位论文].上海:上海交通大学,2013.

[10] 刘洋.基于 Windows 平台的内存数据获取和取证技术研究[硕士学位论文].成都:电子科技大学,2012..

[11] 王峰.基于 Windows 的易失性内存数据取证分析方法研究[硕士学位论文].长春:吉林大学,2012.

[12] 陈龙,敬凯,董振兴,田庆宜.基于 EPROCESS 特征的物理内存查找方法.重庆邮电大学学报,2013,25(1):122−125.

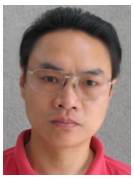
[13] 盘石计算机现场取证系统. 2014. http://www.pansafe.com/

[14] 美亚柏科取证大师.2014. http://300188.cn/subjectview/viewSubjectCategory?cid=2&sortsid=126

[15] 计算机犯罪在线取证系统.2014. http://news.xinhuanet.com/legal/2011-01/12/c_13687726.htm
[16] 郭牧,王连海.基于 KPCR 结构的 Windows 物理内存分析方法.计算机工程与应用,2009,45(18):74–77.
[17] 内存取证系统.2014. http://www.most.gov.cn/dfkj/cq/zxdt/201309/t20130905_109157.htm
[36] 潘爱民.Windows 内核原理与实现.北京:电子工业出版社,2010.

张瑜(1975－),男,湖南辰溪人,博士,副教授,CCF 会员,主要研究领域为网络安全,恶意代码取证,免疫计算.

刘庆中(1971－),男,博士,副教授,主要研究领域为网络安全,计算机取证,人工智能.



李涛(1965－),男,博士,教授,博士生导师,CCF 会员,主要研究领域为网络安全,智能计算及应用.

吴丽华(1963－),女,教授,主要研究领域为人工智能.

石春(1977－),男,博士,副教授,CCF 会员,主要研究领域为无线网路安全,人工智能.