

Table 6 Time cost of PMA algorithm for static measure objects analysis

表 6 PMA 算法分析静态度量对象的时间开销

	SYSCALL	IDT	IDTFUNC	SYSCALLFUNC	KERNELCODE	总数
比特数	1 372	2 048	1 577 098	2 034 403	6 724 012	10 338 933
Time (ms)	0.008 21	0.012 53	9.4373	12.173 8	40.253 1	61.884 94

表 6 中的数据表明:PMA 算法分析静态内容的时间与内容的大小相关,随着内容的增多,分析时间越长,总体时间开销在 61.884 94ms 左右.图 8 给出了不同的进程数和模块数情况下 PMA 算法分析所需要的时间,分析时间与进程数呈多项式递增趋势,与模块数呈线性递增趋势.这是由于分析过程是通过遍历链表实现的,链表越复杂,时间越长.当进程数为 400 时,分析时间为 100ms,模块数为 55,分析时间为 10ms.

4.2.3 T_{he} 和 T_{im} 测试

Hash 和加密运算是完整性度量过程的重要步骤,图 9 给出了不同时刻对不同的度量内容进行运算所需要的时间开销 T_{he} ,其中,Hash 运算时间为黑色线,Hash 和加密运算总时间为灰色线.

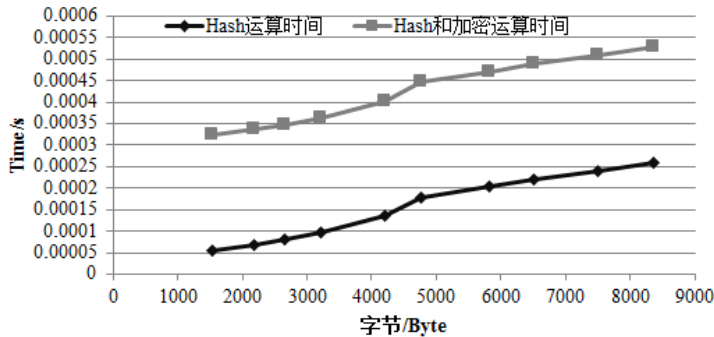


Fig.9 Time cost of Hash operation and cryptographic operation

图 9 Hash 运算和加密运算时间开销

从图 9 中可以看出,Hash 运算时间和总时间与内容大小呈线性正比关系.这与预期想法是一致的,Hash 和加密运算都是按字节进行运算的,由于 Hash 运算结果是固定大小的 160bit 值,加密所需要的时间是固定的,故总时间的变化关系只依赖于 Hash 运算时间.

完整性度量时间主要包括取基准值、解密和完整性匹配的时间,取基准值是从基准库中提取对应度量对象的基准值,解密是负责将经过加密的基准值解密得到 Hash 值,完整性匹配完成对运行过程中的度量对象的 Hash 值与基准值的匹配分析.表 7 中给出了各个部分所需要的时间和总时间,其中,取基准值所需要时间最长,最终完整性度量时间 T_{im} 约为 1.6ms.

Table 7 Time cost of part of measure process

表 7 度量过程各部分时间开销

	取基准值	解密	完整性匹配	完整性度量
Time (ms)	0.993 7	0.492	0.131	1.616 7

从上面分别测试可以得出:当系统进程数为 200、模块数为 50 时,公式(1)的值约为 349.636 6ms.其中,对度量效率影响最大的是 Hash 和加密运算阶段,其次是分析内存提取度量对象阶段,各环节占总时间的百分比如图 10 所示,总时间为 ms 数量级,对系统的造成的负担很小,并且由于 rootkit 活跃期和周期时间 T 一般都在 10s 数量级以上,所以只要度量点处于 rootkit 活跃期,均能度量发现系统中 rootkit 的存在.

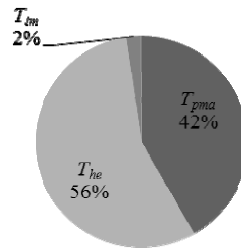


Fig.10 Breakdown of measure time

图 10 度量时间分解

Gibraltar 一次检测时间约为 20s, 因为其需要对整个内存进行扫描; Oस्क 针对某个 rootkit 的检测时间约为 0.5s, 即 500ms, 与基于内存取证的完整性度量方法时间相近, 但是基于内存取证的完整性度量方法所需要的时间并不是检测某一个 rootkit 的时间, 有若干个 rootkit 的情况其也可以在 ms 数量级的时间内实现, 并且 Oस्क 基于 KVM 带来了一定的系统负载; Copilot 中并没有明确指出一次度量所需要的时间, 只给出了其度量周期. 此外, 由于 KIMBMF 通过分析内存实现完整性度量, 故选择了 STREAM benchmark 测试系统负载. 经测试, KIMBMF 在 T 为 15s 的情况下对系统的负载仅为 0.27%, 相对于 Oस्क 的 2%~6%、SBCFI 的 3.6%~48.3% 而言, KIMBMF 对系统的负载可以忽略不计; 与 Gibraltar 的 0.49%、Copilot 的 0.84% 相比, KIMBMF 不仅负载较小, 而且不需要硬件辅助.

此外, 为了测试 KIMBMF 占用 CPU 对系统的性能影响, 我们通过 top 命令监控 KIMBMF 的 CPU 占用率. 经过 1 个小时的监控, KIMBMF 的 CPU 最高占用率仅为 6%, 平均占用率为 2%. 这是因为 KIMBMF 采用的随机化时间间隔运行机制, 只有在某给定时间内才会运行, 从而占用 CPU. 故, 可认为 KIMBMF 占用 CPU 对系统的性能影响可忽略不计.

因此, 通过与这些工具度量或检测所需要的时间和系统负载对比, 基于内存取证的完整性度量方法在时间开销和系统负载上具有一定的优势.

5 结论与下一步工作

本文针对现有的完整性度量方法中存在硬件开销、TOC-TOU 问题和 VMM 带来的性能损失, 提出了基于内存取证分析的内存完整性度量方法 KIMBMF. 它采用内存取证技术, 首先解决如何获取内存内容的问题, 通过 PMCE 算法实现对物理内存的访问和内容的提取; 然后解决了如何分析内存内容获取度量对象的问题. 通过关注 rootkit 攻击内核的方法, 总结 rootkit 攻击的内核对象, 以内核符号表和内核关键数据结构为入口点分析提取度量对象; 同时提出时间随机化算法弱化 TOC-TOU 问题, 提高度量的准确性; 在度量过程中, 将 hash 运算和加密运算相结合加强度量安全性, 避免攻击者伪造基准值躲避完整性度量. 通过实验研究与分析: KIMBMF 能够有效的度量内核的完整性, 时间开销保持在 ms 数量级, 对系统的负担影响很小, 且与周期性度量工具相比, 其度量能力及准确率更高.

实验虽然验证了时间随机化度量比周期性度量具有较高的度量准确率, 但该方法能够弱化而不是完全避免 TOC-TOU 攻击, 从实验数据可知, 该系统未能发现 9% 的 rootkit 攻击. 此外, 由于有些攻击方法并不对内核的任何地方进行修改实现提权、控制流劫持等, 进而破坏内核的完整性, 如 ROP 攻击^[24], 而现有的完整性度量方法 (包括本文的方法) 需要发现修改内核或者破坏内核的某些特性才能够发现内核的不完整性, 所以它们目前无法度量发现这类攻击, 这也涵盖在 9% 内. 下一步的工作将在现有工作的基础上着重研究访问控制、轻量级虚拟控制、内核级恶意软件数据特征等技术, 进一步提高完整性度量的有效性和准确率.

致谢 在此, 我们向对本研究工作提供帮助的老师和同学表示感谢. 同时, 我们也向对本文提出宝贵意见的评审

专家表示感谢.

References:

- [1] Wang YM, Beck D, Vo B, Roussev R, Verbowski C. Detecting stealth software with strider ghostbuster. In: Proc. of the Int'l Conf. on Dependable Systems and Networks. Washington: IEEE CS Press, 2005. 368–377. [doi: 10.1109/DSN.2005.39]
- [2] Joy J, John A. A host based kernel level rootkit detection mechanism using clustering technique. In: Proc. of the Int'l Conf. on Computer Science, Engineering and Information Technology. Berlin, Heidelberg: Springer-Verlag, 2011. 564–570. [doi: 10.1007/978-3-642-24043-0_57]
- [3] Liu ZW, Feng DG. TPM-Based dynamic integrity measurement architecture. Journal of Electronics & Information Technology, 2010,32(4):875–879 (in Chinese with English abstract). [doi: 10.3724/SP.J.1146.2009.00408]
- [4] Bratus S, D' Cunha N, Sparks E, Smith SW. TOCTOU, traps, and trusted computing. In: Proc. of the 1st Int'l Conf. on Trusted Computing and Trust in Information Technologies. Berlin, Heidelberg: Springer-Verlag, 2008. 14–32. [doi: 10.1007/978-3-540-68979-9_2]
- [5] Petroni NL, Hicks M. Automated detection of persistent kernel control-flow attacks. In: Proc. of the 14th ACM Conf. on Computer and Communications Security. New York: ACM Press, 2007. 103–115. [doi: 10.1145/1315245.1315260]
- [6] Hofmann OS, Dunn AM, Kim S, Roy I, Witchel E. Ensuring operating system kernel integrity with osck. In: Proc. of the 6th Int'l Conf. on Architectural Support for Programming Languages and Operating Systems. ACM Press, 2011. 279–290. [doi: 10.1145/1950365.1950398]
- [7] Peter AL, McGill KN. LKIM: Linux kernel integrity measurer. Johns Hopkins APL Technical Digest, 2013,32(2):509–516.
- [8] Rhee J, Riley R, Lin ZQ, Jiang XX, Xu DY. Data-Centric OS kernel malware characterization. IEEE Trans. on Information Forensics and Security, 2014,9(1):72–87. [doi: 10.1109/TIFS.2013.2291964]
- [9] Petroni NL, Fraser T, Molina J, Arbaugh WA. Copilot—A coprocessor-based kernel runtime integrity monitor. In: Proc. of the 13th Conf. on USENIX Security Symp. Berkeley: Usenix, 2004. 179–194.
- [10] Arati B, Vinod G, Liviu I. Detecting kernel-level rootkits using data structure invariants. IEEE Trans. on Dependable and Secure Computing, 2011,8(5):670–684. [doi: 10.1109/TDSC.2010.38]
- [11] Moon H, Lee H, Lee J, Kim K, Paek Y, Kang BB. Vigilare: Toward snoop-based kernel integrity monitor. In: Proc. of the 19th ACM Conf. on Computer and Communications Security. New York: ACM Press, 2012. 28–37. [doi: 10.1145/2382196.2382202]
- [12] Liu Z, Lee JH, Zeng J, Wen YF, Lin ZQ, Shi WD. CPU transparent protection of OS kernel and hypervisor integrity with programmable DRAM. In: Proc. of the 40th Annual Int'l Symp. on Computer Architecture. ACM Press, 2013. 392–403. [doi: 10.1145/2508148.2485956]
- [13] Stevens RM, Casey E. Extracting Windows command line details from physical memory. Digital Investigation, 2010,7(8):57–63. [doi: 10.1016/j.diin.2010.05.008]
- [14] James SO, Gilbert LP. Windows driver memory analysis: A reverse engineering methodology. Computers and Security, 2011,30(8):770–779. [doi: 10.1016/j.cose.2011.08.001]
- [15] Kollar I. Forensic RAM dump image analyzer [MS. Thesis]. Department of Software Engineering, Charles University of Prague, 2010.
- [16] Sylvea J, Caseb A, Marzialeb L, Richarda GG. Acquisition and analysis of volatile memory from android devices. Digital Investigation, 2012,8(3-4):175–184. [doi: 10.1016/j.diin.2011.10.003]
- [17] Carvey H. Windows Forensic Analysis. 2nd ed., Elsevier: Syngress, 2009. 59–63.
- [18] Vidstrom A. Memory dumping over firewire-uma issues. 2006. <http://ntsecurity.nu/onmymind/2006/2006-09-02.html>
- [19] Gladyshev P, Almansoori A. Reliable acquisition of RAM dumps from Intel-based Apple Mac computers over FireWire. In: Proc. of the 2nd Int'l Conf. on Digital Forensics and Cyber Crime. Berlin, Heidelberg: Springer-Verlag, 2010. 55–64. [doi: 10.1007/978-3-642-19513-6_5]
- [20] Wang LH. Model and methods research on computer live forensics based on physical memory analysis [Ph.D. Thesis]. Ji'nan: Shandong Univerisity, 2014 (in Chinese with English abstract).
- [21] LiME. 2012. <http://code.google.com/p/lime-forensics>

- [22] Petroni NL. Property based integrity monitoring of operating system kernels [Ph.D. Thesis]. University of Maryland, 2008.
- [23] Riley R. A framework for prototyping and testing data-only rootkit attacks. Computers and Security, 2013,37:62-71. [doi: 10.1016/j.cose.2013.04.006]
- [24] Pappas V, Polychronakis M, Keromytis AD. Transparent ROP exploit mitigation using indirect branch tracing. In: Proc. of the 22nd USENIX Security Symp. Washington: Usenix, 2013. 447-462.

附中文参考文献:

- [3] 刘孜文,冯登国.基于可信计算的动态完整性度量架构.电子与信息学报,2010,32(4):875-879.
- [20] 王连海.基于物理内存分析的在线取证模型与方法的研究[博士学位论文].济南:山东大学,2014.



陈志锋(1986-),男,福建漳州人,博士,讲师,主要研究领域为信息安全,可信计算.



张平(1969-),女,博士,副教授,主要研究领域为并行识别,并行编译,信息安全.



李清宝(1967-),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为信息安全,可信计算.



王炜(1975-),男,博士,副教授,CCF 会员,主要研究领域为计算机系统结构,信息安全.