Key Software Security Techniques for Virtualization Environments

面向虚拟化环境的软件安全关键技术

田东海 著

北京理工大学出版社
BEIJING INSTITUTE OF TECHNOLOGY PRESS

Key Software Security Techniques for Virtualization Environments

面向虚拟化环境的软件安全关键技术

田东海 著

北京理工大学出版社
BEIJING INSTITUTE OF TECHNOLOGY PRESS

内 容 简 介

本书根据作者多年的虚拟化软件安全研究成果，对近年来虚拟化软件安全技术进行了梳理和总结。全书深入浅出地介绍了虚拟化技术在应用软件安全加固和系统软件安全加固方面的应用，具体包括：①结合离线分析和在线保护检测应用程序的缓冲区溢出攻击；②利用虚拟化技术保护操作系统内核数据的完整性；③利用虚拟化内存隔离技术和程序分析技术保护操作系统内核模块安全；④利用虚拟化技术和并行算法高效检测操作系统内核堆溢出攻击；⑤利用虚拟化技术和符号执行技术检测内核模块中的安全漏洞。本书不仅介绍了相关方法、技术和实验结果，还分析了国内外相关的研究工作，可供读者学习和参考，理解相关方法与技术的原理和应用。

本书可供计算机、软件工程、网络空间安全等专业的教师及软件安全相关领域的技术开发人员参考，也可作为相关专业的高年级本科生和研究生的教材。

图书在版编目（CIP）数据

面向虚拟化环境的软件安全关键技术/田东海著．—北京：北京理工大学出版社，2020.1

ISBN 978-7-5682-8107-2

Ⅰ.①面…　Ⅱ.①田…　Ⅲ.①软件开发-安全技术-研究　Ⅳ.①TP311.522

中国版本图书馆 CIP 数据核字（2020）第 021399 号

出版发行／北京理工大学出版社有限责任公司
社　　址／北京市海淀区中关村南大街 5 号
邮　　编／100081
电　　话／（010）68914775（总编室）
　　（010）82562903（教材售后服务热线）
　　（010）68948351（其他图书服务热线）
网　　址／http：//www.bitpress.com.cn
经　　销／全国各地新华书店
印　　刷／保定市中画美凯印刷有限公司
开　　本／710 毫米×1000 毫米　1/16
印　　张／9.5
字　　数／140 千字
版　　次／2020 年 1 月第 1 版　2020 年 1 月第 1 次印刷
定　　价／46.00 元

责任编辑／王玲玲
文案编辑／王玲玲
责任校对／刘亚男
责任印制／王美丽

前　言

随着信息化的推进，软件的应用已渗透到社会生产和生活的各个方面。软件作为信息化的核心，发挥了越来越多的作用，随之而来的软件安全问题变得日益重要。由于软件的代码规模不断增大和软件的逻辑变得日趋复杂，导致发布的软件产品可能含有安全漏洞。一旦这些软件漏洞被攻击者所利用，将对信息系统安全产生严重的影响。近年来，由软件安全问题引发了大量的网络攻击事件，给社会带来了较大的经济损失。

基于操作系统的软件安全技术只能抵御用户级的软件攻击，对内核级的软件攻击无能为力，并且难以解决与现有软硬件结构兼容的问题。虚拟化技术的出现为增强软件安全性提供了新的技术途径。另外，越来越多的软件被部署到虚拟化环境中，给软件安全技术应用提出了新的需求。

本书包括6章内容。第1章为绪论，论述了基于虚拟化技术研究软件安全的背景、研究意义和相关研究工作。第2章介绍了离线分析与在线保护相结合的缓冲区溢出防护技术，该技术能有效提高含有潜在缓冲区溢出漏洞的应用软件安全。第3章介绍了如何利用虚拟化技术保护内核数据的完整性，从而提高操作系统内核数据的安全。第4章介绍了如何使用虚拟机监控器来实现对脆弱内核模块的安全加固，从而保护操作系统内核不受脆弱内核模块的影响。第5章介绍了如何利用虚拟化技术在不影响系统性能的同时实现对内核堆上缓冲区的并行监控，从而保护操作系统内核堆区的安全。第6章介绍了利用虚拟化技术对内核模块进行安全测试的方法，能有效发现内核模块中存在的安全漏洞。

本书大部分内容取自作者自身及指导的研究生漆定军的科研成果，感谢马锐老师参与了第6章的撰写工作。由于作者水平有限，本书中难免存在不足之处，恳请广大读者批评指正。

本书可供计算机、软件工程、网络空间安全等专业的教师及相关领域的技术开发人员参考，也可作为相关专业的高年级本科生和研究生的教材。

最后，感谢北京理工大学计算机学院网络攻防对抗技术研究所为本书的出版创造了宝贵的条件，感谢北京理工大学计算机学院常务副院长胡昌振教授对本书的大力支持，感谢美国宾夕法尼亚州立大学刘鹏教授和吴丁浩教授为本书相关研究工作给予的指导和帮助。感谢作者的家人在作者从事研究和撰写本书过程中给予的关心和帮助。感谢科技部和国家自然科学基金委的大力支持。本书的相关研究得到科技部国家重点研发计划（2016QY07X1404）、国家自然科学青年基金（61602035）、中国科学院信息安全国家重点实验室开放课题、中国科学院网络测评技术重点实验室开放课题等项目的资助，在此表示衷心感谢。

田东海
Email：donghaitad@ gmail. com
北京理工大学计算机学院
网络攻防对抗技术研究所
北京市软件安全工程技术重点实验室
2019 年 8 月

目　录

图索引

表索引

第1章 绪 论

1.1 引言

随着计算机软件的功能和结构日趋复杂，软件安全漏洞形成概率不断增加，并且暴露的频率也不断增大。根据美国国家漏洞数据库的统计[①]：2009 年总计报告了 5 733 个软件漏洞，其中高危险性漏洞的比例超过了 1/3。如果这些高危漏洞被黑客非法利用，将会给计算机系统带来严重的危害，如篡改用户数据甚至控制整个计算机系统。此外，许多病毒程序和恶意代码也利用软件漏洞，并通过网络进行传播，如著名的红色代码病毒[②]和蓝宝石蠕虫病毒[③]。这些病毒程序的出现严重阻塞了全球的网络，给全世界造成了重大的经济损失。

为了提高应用软件的安全，传统的解决方法通常是在操作系统内核中插入安全模块，以增强对上层应用程序的管理和控制，如可以在 Linux 系统中增加 SELinux[④] 和 AppArmor[⑤] 安全模块。操作系统拥有庞大的代码，自身存在安全漏洞的可能性相当大，几乎所有通用操作系统都有安全漏洞[⑥⑦]，在操作系统内核层实现的安全机制并不能完全保证上层应用程序的安全。此外，传统的安全机制也无法对操作系统内核本身进行保护。

---

① National Vulnerability Database. http://nvd. nist. gov/.

② CERT Advisory CA - 2001 - 19 http://www. msen. com/cert - ca - 2001 - 19. html.

③ SQL Slammer https://en. wikipedia. org/wiki/SQL_Slammer.

④ Nsa. security enhanced linux. http://www. nsa. gov/selinux/.

⑤ Apparmor. http://www. novell. com/linux/security/apparmor/.

⑥ sqrkkyu and twzi. Attacking the core: Kernel exploiting notes. http://phrack. org/issues. html, 2007.

⑦ C. S. Technologies. Openbsd ipv6 mbuf remote kernel buffer overflow. http://www. securityfocus. com/archive/1/462728/30/0/threaded, 2007.

虚拟化技术的出现为提高软件安全提供了新的技术途径。虚拟化概念最初由 IBM 公司于 20 世纪 60 年代提出。当时人们研究虚拟化技术的目的是充分利用相对昂贵的硬件资源。由于硬件成本不断降低，使得 20 世纪 80 年代和 90 年代对虚拟化技术的研究趋于沉寂。近年来，随着硬件性能的不断提升，在普通机器（如 PC 机）上同时运行多个互不相干的操作系统已经成为可能，虚拟化技术又重新受到学术界和工业界的关注。

实现系统虚拟化的常用方法是利用虚拟机监控器（Virtual Machine Monitor，VMM）为上层操作系统提供一组物理硬件抽象[1]。由于虚拟机监控器在最底层运行，对上层客户机系统拥有完全的控制权，使得在虚拟机监控器内部实现软件安全保护机制成为可能。与通用操作系统相比，虚拟机监控器的代码量要小很多，并且对外界提供的接口也很简单，因此比较适合成为可信计算平台。此外，在虚拟机监控器内部实现软件安全保护机制不会给上层操作系统和应用程序带来兼容性问题。目前，国内外大量的科研机构都开始研究如何利用虚拟化技术来解决传统的软件安全问题。近年来，以虚拟化技术为基础的云计算得到迅猛发展，越来越多的应用服务被部署到云中，如何保证这些云中软件的安全已成为云计算发展所面临的重大挑战①。

1.2 虚拟化技术背景

虚拟机是一台由软件实现的机器，但能像真实的物理机一样运行各种程序。根据虚拟机实现的层次不同，可将其分为三类：应用层虚拟机、操作系统层虚拟机和硬件层虚拟机。应用层虚拟机是用户程序和操作系统之间的一层软件，为上层应用程序提供一个对底层软件和硬件资源进行抽象的跨平台可编程环境。Java 虚拟机（Java Virtual Machine，JVM）是目前使用较为广泛的应用层虚拟机。Java 程序被编译成 Java 字节码（Java Bytes Code），由 JVM 进行解释执行。类似的应用层虚拟机还有微软 .NET 架构

① Cloud computing: A new era of IT opportunity and challenges. http://www.zdnet.com/blog/hinchcliffe/cloud-computing-a-new-era-of-it-opportunity-and-challenges/.

下的 CLR（Common Language Runtime）运行环境。操作系统层虚拟机可看作系统级的隔间（Compartment），其中包含独立的文件系统、进程、用户账户和根账户等。Jail 是 FreeBSD 上的操作系统层虚拟机。该虚拟机允许系统管理员将 FreeBSD① 系统分成若干独立的微型系统，称之为 Jail。每个 Jail 都拥有独立的系统资源以及相关的配置，从而用户可在不同的 Jail 中运行不同配置的应用程序。OpenVZ② 是基于 Linux 的操作系统层虚拟机，它允许在一台物理机上同时运行多个系统实例，并且每个实例可单独管理系统的资源。与 Jail 和 OpenVZ 技术类似，Solaris 操作系统提供了区域（Zone）技术[2,3]，以支持操作系统层虚拟化。硬件层虚拟机允许不同的客户虚拟机共享底层硬件资源，并且不同的客户虚拟机能运行不同的操作系统。此外，硬件层虚拟机技术通过虚拟机监控器（Virtual Machine Monitor，VMM）管理上层操作系统对底层硬件资源的访问。VMware③ 和 Xen④ 是目前业界比较流行的硬件层虚拟机。应用层虚拟机和操作系统层虚拟机都无法抵御针对操作系统内核的攻击，因此硬件层虚拟机技术成为当前研究的重点。

1.2.1 硬件层虚拟机技术

根据虚拟机监控器在整个计算机系统中的实现位置和方法不同，Popek 和 Goldberg 定义了两类虚拟机监控器模型[1]，即 Type Ⅰ VMM 和 Type Ⅱ VMM。如图 1.1 所示，Type Ⅰ VMM 运行在物理硬件和客户机操作系统之间，可看成是一种特殊的操作系统；Type Ⅱ VMM 运行在现有操作系统之上，通过利用操作系统中现有的各种服务管理和维护上层虚拟机的运行。

为了实现硬件层虚拟化，Popek 和 Goldberg 提出了三条基本准则：同质、高效和资源受控。同质是指程序在虚拟机中运行的行为应与在真正物理机中运行的行为相同。高效是指程序在虚拟机中运行的性能应接近在物

① Free BSD jails，http://www.freebsd.org/.

② OpenVZ project，http://wiki.openvz.org/Main_Page.

③ VMware，http://www.vmware.com/.

④ Xen，http://www.xen.org/.

图 1.1 Type Ⅰ VMM 和 Type Ⅱ VMM

理机中运行的性能。为了达到这个目的，必须保证绝大多数机器指令能直接在物理硬件上执行。资源受控是指 VMM 必须完全控制虚拟机所拥有的资源。

Popek 和 Goldberg 进一步提出将指令集架构（Instruction Set Architecture，ISA）中的指令分为两类：特权指令和敏感指令。特权指令是指处理器在特权模式下执行的指令。如果这些指令在非特权模式下运行，将会产生异常，并导致处理器切换到特权模式。敏感指令又可分为控制敏感指令和行为敏感指令。控制敏感指令是指那些试图重新配置系统资源的指令，如更新虚拟内存到物理内存映射的指令和修改特权寄存器的指令。行为敏感指令是指那些根据不同硬件资源的配置，表现出不同行为的指令，如加载内存指令。为了使某个体系结构可虚拟化，Popek 和 Goldberg 指出其条件是所有的敏感指令都必须是特权指令。经典的可虚拟化体系结构包括 IBM System 360/370 和 Motorola MC6820。然而，业界普遍采用的 x86 体系结构却不能完全支持可虚拟化，其原因在于 x86 指令集架构中有 17 条敏感指令不是特权指令[4]。

为了在 x86 体系结构上实现硬件层虚拟化，有三种不同的解决方法：基于二进制代码翻译（Binary Translation）的完全虚拟化（Full - virtualization）技术、半虚拟化（Para - virtualization）技术和硬件辅助的虚拟化（Hardware Assisted Virtualization）技术。

1. 完全虚拟化技术

以 VMware 为代表的虚拟化平台采用了基于二进制代码翻译的方法实现了系统的完全虚拟化。该技术允许虚拟机中的应用程序直接在 CPU 上运行，而虚拟机中操作系统内核代码则在 VMM 中指令转换引擎的控制下执行。该引擎通过动态扫描内核代码中的敏感指令，将其转换成等价的翻译代码，以确保 x86 体系结构下所有敏感指令均能产生自陷，从而被虚拟机监控器截获。完全虚拟化技术不要求修改操作系统内核，因此具有良好的兼容性。

2. 半虚拟化技术

与完全虚拟化技术不同，半虚拟化技术通过直接修改操作系统使其内核代码中所有敏感指令均能产生自陷，从而也同样实现了系统的虚拟化。典型的半虚拟化系统包括英国剑桥大学开发的 Xen[5] 和美国华盛顿大学开发的 Denali[6]。半虚拟化技术的最大特点是使虚拟机尽可能直接访问硬件资源，以降低由系统虚拟化带来的性能开销。

3. 硬件辅助的虚拟化技术

硬件辅助的虚拟化技术通过增加 x86 处理器的执行模式和相关指令使硬件本身能直接支持系统虚拟化。为了使该项技术商业化，Intel 和 AMD 公司分别推出了 VT[7-9]（Virtualization Technology）技术和 pacifica[10] 技术。硬件辅助的虚拟化技术的出现降低了虚拟机监控器实现的难度，并且可以避免对客户机操作系统内核进行修改。Xen 从 3.0 版本开始支持 Intel VT 和 AMD Pacifica 技术，从而使各种未经修改的操作系统（如 Windows）能直接在 Xen 上运行。

虚拟化技术最早出现在 20 世纪 60 年代。随着近年来多核系统、集群、网格和云计算的普及，虚拟化技术在商业上的应用受到业界越来越多的关注。有超过 80% 的全球百强企业采用了 VMware 的虚拟化产品[11]，而基于 Xen 平台的 Amazon EC2① 系统也被广大的中小企业甚至个人用户所使用，

① Amazon Elastic Compute Cloud, http://aws.amazon.com/ec2/.