Broadview

Windows 内核原理与实现

潘爱民 著

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

Windows内核原理与实现

潘爱民 著

電子工業出版社
Publishing House of Electronics Industry
北京 • BEIJING

内容简介

本书从操作系统原理的角度，详细解析了 Windows 如何实现现代操作系统的各个关键部件，包括进程、线程、物理内存和虚拟内存的管理，Windows 中的同步和并发性支持，以及 Windows 的 I/O 模型。在介绍这些关键部件时，本书直接以 Windows 的源代码（WRK, Windows Research Kernel）为参照，因而读者可以了解像 Windows 这样的复杂操作系统是如何在 x86 处理器上运行的。

在内容选取方面，本书侧重于 Windows 内核中最基本的系统部件，同时也兼顾到作为一个操作系统的完整性，所以，本书也介绍了像存储体系、网络、Windows 环境子系统等，这些虽然并不位于内核模块但却支撑整个 Windows 运行的重要部件。在本书最后，也介绍了 Windows Server 2003 以后的内核发展和变化。

虽然书中有大量关于 Windows 代码实现的描述，但是本书并没有罗列 WRK 中的代码，即使读者不对照 WRK 的源代码，也可以从这些章节的描述中理解 Windows 的实现机理。在每一个技术专题的介绍中，本书几乎都提供了一个框架图，并且有关键细节的实现分析，这样做的意图是让读者既能够对一项技术有总体上的把握，也通晓关键的实现细节。

Windows 操作系统已经有 20 年历史了，市面上有大量关于 Windows 技术的文档和书籍，但是，真正从源代码来诠释 Windows 底层机理的，本书还是第一次尝试。在本书覆盖的内容中，有相当一部分是第一次以文字形式披露出来的，期望这些内容能消除人们对于 Windows 的神秘感。

写作本书的目的是让对 Windows 有好奇心的人真正了解到 Windows 中的核心机理，让计算机专业的学生和老师，以及系统软件工程师可以快速地领略到 Windows 中先进的系统技术，以及在 Windows 上编写出更加高效的软件。本书也配备了一些小工具，通过这些小工具，读者可以查看内核中的静态或动态的信息，甚至观察系统的行为，可通过 Internet 下载这些工具。

图书在版编目（CIP）数据

Windows 内核原理与实现 / 潘爱民著. —北京：电子工业出版社，2010.5
ISBN 978-7-121-10528-9
Ⅰ. W… Ⅱ. 潘… Ⅲ. 窗口软件，Windows—程序设计… Ⅳ. TP316.7
中国版本图书馆 CIP 数据核字（2010）第 043898 号

策划编辑： 周　筠
技术编辑： 卢鸶翔　白　涛
责任编辑： 陈元玉
印　　刷： 北京智力达印刷有限公司
装　　订： 三河市皇庄路通装订厂
出版发行： 电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编　100036
开　　本： 787 × 980　1/16　　**印张：** 44.75　　**字数：** 850 千字
印　　次： 2010 年 5 月第 1 次印刷
印　　数： 6 000 册　　**定价：** 99.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。
质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。
服务热线：（010）88258888。

序 一
Foreword

The Windows Operating System has been evolving, along with the PC-based computer. Although systems based on UNIX have dominated in the universities since the 1970s, Windows has completely surpassed UNIX in commercial systems. Yet relatively little information about the internals of Windows – the architecture and code of the kernel – has been available outside of Microsoft.

With the support of Microsoft's executive management (particularly Rob Short, Jim Allchin, and Bill Gates), my colleagues Arkady Retik, Chris Fagan, and I set out to make the sources for the Windows kernel available to faculty, researchers, and students throughout the world. We wanted them to have the opportunity to examine the internals of the kernel and understand its implementation and thus be able to decide for themselves what is admirable about the Windows kernel – and what is not.

This project became the Windows Research Kernel (WRK), and is part of the Windows Academic Program (WAP www.microsoft.com/WindowsAcademic). The WAP also includes a large amount of teaching materials in the Curriculum Resource Kit (CRK). Effectively the WRK is the Windows kernel in source form, and the CRK is the Windows kernel in PowerPoint (with exercises, experiments, projects, and other learning aids).

All along we knew that the WRK and CRK by themselves would not make the Windows kernel as accessible as we wanted it to be. It would take books like *Understanding the Windows Kernel*[1] to draw together and organize the ideas in the kernel and explain the source code.

Our intention with the Windows Academic Program has never been to advocate that Windows supplant UNIX in the teaching of operating systems in universities. We believe that both the UNIX and the Windows kernel should be taught, so that it is

[1] *Understanding the Windows Kernel* 为本书英文书名。——编者注

possible to compare and contrast the architecture and design choices that are evident in both systems.

As you will see from this book, Windows has made some very different choices than UNIX. In large part this is because Windows was developed to target a very different system and user environment than UNIX did when it was created. Windows would run on multiprocessors, so threads are the unit of scheduling rather than processes. The systems would have at least 32 bits of pageable virtual address space, so the Windows kernel had the opportunity to build general purpose mechanisms such as the object manager and stackable I/O. Windows expected to run in a very dynamic environment, so the object name space resides outside the file system and is maintained by the kernel itself. Although not a microkernel, Windows did incorporate the Mach idea of multiple operating system personalities in user-mode "subsystem" processes, initially supporting OS/2, POSIX, and Win32 interfaces.

The internal architecture of the kernel reflects learning from UNIX, Mach and VMS, Digital's VAX operating system that was also designed by Dave Cutler, the senior architect of the Windows OS. There is a layer within the kernel which abstracts and schedules the CPUs (as threads, interrupts, traps, etc). This "kernel of the kernel" layer is about 5% of the code. The rest of the kernel is in the "executive" layer (including the device drivers, file systems, and network code which run outside of the kernel proper). The executive is implemented as fairly ordinary multithreaded code, which greatly simplifies the expertise required by kernel-mode developers.

For those who learned operating systems by reading and understanding only the ideas in UNIX, learning about Windows creates a great opportunity to evaluate their own assumptions about what an operating system should be. This is important because the nature of computer systems continues to rapidly evolve. UNIX was designed for 16-bit, low-memory, swapping systems. Windows for 32-bit systems with paged virtual addressing and megabytes of physical memory. But contemporary systems are 64-bit with gigabytes of memory, and many new features like virtualization hardware, solid-state disks, and integrated computational and graphics accelerators. Connectivity has evolved from the dial-up modems at UNIX's birth in 1969 to the 10 Mbps local Ethernet at Windows birth in 1989. And now we have multiple Gbps networks interconnecting computers around the world. Will a new operating system emerge which targets these new hardware environments? Or will existing systems just continue to evolve their capabilities? It is only by having a broader understanding than

either just UNIX or just Windows that software engineers will be able to judge for themselves.

Our colleague, Aimin Pan, is one of the bright stars in the new generation of computer scientists teaching and researching some of these important questions about the future of operating systems. His presentation of the Windows kernel will be a great asset for those who want to also participate in the evolving future of operating systems, as well as those who are just simply curious, or want to better understand the software architecture that they are using daily.

Dave Probert, Ph.D.
compsci@microsoft.com
Windows Kernel Architect – Microsoft
Developer of the Windows Research Kernel release
14 February 2010

序二

Windows 作为用户最多的主流操作系统，备受广大师生和工程技术人员的关注，大家希望能通过深入内核的实验和分析源代码的方式，来透彻地了解 Windows 操作系统。微软亚洲研究院认真地考虑了这一需求，并和微软 Windows 产品部的专家一起做了大量的努力，终于在 2005 年促成了微软公司面向教育界发布的 Windows 学术支持计划，其中就包括了为教育和科研提供 Windows 内核源代码的访问许可 WRK（Windows Research Kernel）。

我们围绕 WRK 进行了一系列的教师培训，大家认为 WRK 的发布是深入了解 Windows 操作系统内核的一个里程碑，但同时也反映出一个问题：WRK 包含大量的源代码，如何阅读和分析它们，并将这些源代码与教学中的知识结构对应起来，是进一步使用 WRK 的瓶颈。

潘爱民博士写的《Windows 内核原理与实现》这本书很好地解决了这一问题。潘博士采取将全局系统框图和关键实现分析相结合的方式，详细解析了 Windows 对现代操作系统的各个关键部件的实现。书中采用了 Windows 操作系统的真实数据结构，并将操作系统的实现机理与 WRK 中的源代码描述对应了起来。书中附带有很多分析工具，使读者可以方便地跟踪和观察操作系统内核的运行过程。

潘爱民博士是软件技术和操作系统方面的专家，他从 2004 年开始在微软亚洲研究院从事系统性能分析的研究工作，在 Windows 操作系统的内核优化方面有着丰富的知识和经验。潘博士承担了《Windows Internals, 4th Edition》一书中文版的翻译工作，他还参与了很多与操作系统有关的高校合作项目，通过教师培训和指导学生，对 Windows 操作系统在教学和科研中的实际需求有着深入的了解。

本书是潘爱民博士对 Windows 操作系统以及 WRK 深入研究的结晶，代表了微软亚洲研究院的科研人员对探究技术和帮助中国教育的极大热情。我很高兴看到本书的顺利出版，它是一本很好的操作系统教学参考书；对广大希望深入了解 Windows 操作系统的读者来说，它是一本很好的学习用书。

微软亚洲研究院 副院长 宋罗兰

2010 年 2 月 13 日

岁月如歌

—出版人感言—

3年前，潘老师在我的CSDN博客上留言：

我对翻译的热情越来越淡了，也希望出版界多做一些本土作家的精品，原文版的图书尽可能直接引进。这对国内工业界是有长远意义的。

当年，潘老师放弃了一本曾经由他翻译的经典C++著作的新版翻译，选择翻译吃力且未必讨好的《Windows Internals，第四版》，取舍之间，让我再次看到他的得失观和价值观，看到他于国内IT业界的长远眼光。

11年前，潘老师自己的原创作品《COM原理与应用》就得到了业内的诸多好评，若干年后还是不断有读者提起潘老师的这本处女作，并问潘老师怎么不写新的书。

侯捷老师在一篇文章里，提到翻译和原创技术图书，说：这如同歌手唱歌，翻唱别人的歌，总比写出自己的原唱歌曲要容易。

10年前，国内好的翻译图书都不多，潘老师和侯捷老师、裘宗燕老师这批人，以自己的勤奋认真，开创了一代有品质的技术图书的写译先河，为后来者树立起高标杆，善莫大焉，功莫大焉。但当时限于积累，原创图书还只能让位于翻译图书。

10年间，武汉博文团队在成立的7年里，一点点践行着潘老师的期望：出版界多做一些本土作家的精品。

去年，由潘老师策划组织的原创好书《程序员的自我修养——链接、装载与库》由武汉博文出版。上市后即受到许多技术高手的好评。该书繁体版2009年底在台湾上市后，连续三个月位居畅销榜第一名。侯捷老师认真读过这本书后，给予了高度赞誉。

今年，武汉博文再次得到潘老师的信任，获得了潘老师独立创作的《Windows内核原理与实现》的出版权。我和我的同事深感责任重大，不敢有丝毫懈怠。为了做好这本书，

我和同事们较早开始做各种准备。我付出的辛苦是最少的，担任责任编辑的陈元玉，担任技术编辑的卢鹄翔和白涛，担任设计任务的小勤和文佳，他们都比我劳累。

然而，出书的辛苦远不能和写书相比。多少个凌晨，我已从梦中醒来，潘老师却才入睡。他也一次次地和我商量着写书的细节，担心自己写的不够好。他对自己书稿的要求，远比编辑要高。我知道他身边还有着依恋爸爸的幼儿，但他牺牲了很多陪伴孩子和太太的时间。

孤灯一盏，孜孜不倦辛勤写作技术性极强而又极有价值的著作，希望弥补国内业界和出版界的空白。这种苦，不是什么人都能品尝的，没有足够的技术积累，没有内心深沉的热情，没有仔细的筹划，没有 Just do it 的执行力，难以坚持到最后一刻。惟其如此，这种种艰辛中的快乐和欣慰，也只有少数人有资格品味。

潘老师常常情不自禁地和我提起他的老师王选，这位以一己之力让一个行业发生了巨变的大侠，有着非凡的人格魅力。从潘老师身上，我看到：一个年轻人，在他初出茅庐之时，若得遇王选那样的智者指路，他的一生将会受到怎样深远的影响。

生命如歌，岁月如歌。

周 筠

2010 年春于武汉

致谢

首先，要感谢在微软工作的人，没有他们的鼎力支持和辛勤努力，这本书的内容和选题便不复存在：

- 感谢 Dave Probert 提供了一套可编译和运行的 Windows 源代码，将 Windows 的最新技术成果拿出来与社会分享，而且，每次到中国来，都会将自己在操作系统领域中的最新工作结果介绍给我。感谢 Arkady Retik 一直在 WAP（Windows Academic Program）项目上辛勤地工作，不遗余力地将 Windows 的教学资源在全球范围内推广。

- 感谢宋罗兰女士，她和她领导的高校关系团队长期以来支持我走到学生中间，让我始终能够贴近高校的氛围，并接触到中国最为优秀的学生。感谢张高和马歆，他们一直在努力让中国的大学生有更多的机会接近微软的技术，并有机会融入到微软的大家庭中。感谢历任 WinCore 工作组的负责人，邹静、李贝、Bill Luan、黄婷婷、景霓和 John Warren，他们的辛勤工作使国内更多教师和学生能获得 WRK 及相关教学资源。

- 感谢我的老板李世鹏博士，允许我在研究工作之余钻研 WRK，并支持我从事 Windows 系统性能方向的研究。感谢我的同事朱斌博士，作为最紧密的工作伙伴，对我所做的每一个项目都给予了最大力的支持。感谢微软亚洲研究院前院长沈向洋博士，他长期以来一直支持和鼓励我做任何感兴趣的研究题目，并把最优秀的学生交给我来指导。

- 感谢微软公司和微软亚洲研究院的众多同仁，让我在浓厚的技术氛围中感受着 Windows 无与伦比的魅力。我有机会经常跟一些同事讨论关于 Windows 核心技术细节，他们是：Stephen Hsiao、Jin Jia、Qiang Chen、Bin Zhao、党映农、刘未鹏。

其次，感谢过去几年中跟着我钻研 Windows 内核并从事 Windows 系统性能研究的实习生们，他们是：李子拓、黄晓晨、韩铮、郭津之、杨晨、武健、龙海、王嘉捷、张开敏、曹家鑫、潘震皓、齐鸣、高翔，其中高翔和曹家鑫编写了本书中用到的 5 个工具，潘震皓实现了附录 B 中介绍的 KInjectToolKit 项目。

成书的过程并不轻松，从立题一直到初稿完成，再到最后的审稿、修改和制作，都是在与出版社编辑人员的紧密合作下完成的。尤其在最后审稿和制作过程中，与几位编辑按流水线方式通力合作，既缩短了编审周期，也保证了终稿质量。

- 感谢周筠女士，她和她的团队是如此地让人信任，因而我一开始就找到她，期望她能出版这本书。事实证明，我的选择是正确的。谢谢周老师为这本书倾注的热情，以及对我一贯的信任。
- 感谢责任编辑陈元玉女士，从最初商定版式，一直到后期制作的所有事宜，她都事无巨细，承担了所有可能提供给我的编辑支持，让我可以安心地关注在本书内容上。
- 感谢技术编辑白涛和卢鹍翔，两位编辑夜以继日地检查书中的每一个细节，使书中的技术不精确程度降低到最小，也纠正了很多笔误。
- 感谢设计师杨小勤，这本书的封面以及书中的部分插图均出自他手。Windows 的内核就像一个交响乐队，复杂、深邃而又和谐，让上层应用程序可以安心和高效地运行。同时感谢为本书设计版式的设计师胡文佳，谢谢她的细心和耐心。

最后，感谢我的家人，我的妻子孙红芳女士和儿子安安在我写作本书的两年间对我的大力支持。这两年正好是安安从一岁到三岁的成长期，我在这本书上所花的时间有相当一部分本该是陪着他们的。

潘爱民

2010 年 3 月于北京西二旗

前言

《Windows 内核原理与实现》是一本讲述 Windows 内核机理，并配合 Windows 的源代码来解释其实现细节的学习用书。全书从操作系统原理的角度来组织内容。本书适合于已经有了操作系统基本概念的读者进一步理解 Windows 操作系统。通过学习本书的内容，读者不仅可以掌握 Windows 的核心机制，也可以理解像 Windows 这样的现代操作系统是如何构建起来的。因此，这是一本 Windows 操作系统内核学习用书，而不是一本指导在 Windows 平台上进行软件开发的书。

对于绝大多数 IT 从业人员及高校学生来说，Windows 已经是一个熟悉得不能再熟悉的操作系统了，而且也有相当多的人能够熟练地开发各种类型的 Windows 应用程序，然而，真正熟悉 Windows 内部机理的人却少而又少。究其根源，很多人将这归咎于 Windows 是一个闭源操作系统，也就是说，除了 Microsoft 内部的员工以外，人们接触不到 Windows 的源代码。这样的解释只能说是部分正确的，因为从历史发展来看，UNIX 类操作系统确实有广泛的群众基础，很多人得益于阅读各种 UNIX 版本的源代码（包括 Linux 和 FreeBSD）。但对于 Windows 操作系统，这似乎并非一个合理的解释，因为事实上，有相当多的人在过去三四年间已经获得了 Windows 的源代码，但是在代码阐释和核心深入理解方面却并未表现出拥有源代码而带来的优势。反倒是在过去几年间基于逆向工程而获得的 Windows 核心知识要深刻得多。

这么说并非指 Windows 源代码没有帮助，而是间接地揭示了一个事实：Windows 的源代码不是那么容易读的，Windows 的核心也不是那么好理解的。本书作者在过去几年间，与国内高校的操作系统课程的老师和学生有着广泛的接触和了解，真切地感受到，要想在操作系统课程中完全融入 Windows 的内容，迫切需要一本按照操作系统基本概念和理论来阐释 Windows 实现机理的书籍。基于这样的动机，本书力图让学习操作系统的读者能够理解 Windows 中的核心机制，并且可以近距离地观察到 Windows 内核中的实现细节。

本书针对的读者

对 Windows 内核有好奇心的读者，无论您是操作系统课程的学生，或是正在从事系统软件开发的软件工程师，或是有技术背景的 Windows 系统管理员，都可以从本书中找到很多问题的答案。虽然本书中描述的大多数细节是以 WRK（Windows Research Kernel）的代码为基础，但是，对于各种核心机制的介绍却是通用的，并不限于 WRK 内核，甚至适用于各种版本的 Windows，包括最新的 Windows Vista、Windows Server 2008 和 Windows 7。

对于有资格获得 WRK 的读者（包括高校老师和学生），本书可以是配套的学习参考用书。本书正文部分列出了 WRK 中一些关键的数据结构定义，而在介绍各种核心机制时，明确地指出了源代码中的函数调用关系、关键数据的含义和用途，甚至在有些地方指明了 WRK 中对应的代码片段。因此，配合 WRK 代码来阅读本书是最理想的阅读方式，就如同本书作者在写作过程中经常查看 WRK 代码一样。

然而，对于因某些原因而无法获得 WRK 的读者，本书仍然有指导意义。首先，本书在讲解 Windows 内核的核心机制时，几乎都以文字或图形的方式描述了这些机制的总体结构，并指出了关键的数据结构或函数。即使没有 WRK 代码的参照，读者也可以想象出这些函数的功能实现。其次，如果读者善用 windbg，并设置 Microsoft 提供的公共符号服务（参考本书附录 A），则可以在内核调试环境中查看到很多实现细节。书中描述的绝大多数核心机制不仅适用于 Windows Server 2003 SP1（即 WRK 内核适用的系统），也适用于其他的 Windows 版本，包括 Windows XP、Windows Vista、Windows Server 2008 和 Windows 7。

本书不是一本轻松的入门书籍，无论读者是否拥有 WRK，都最好在阅读以前先有操作系统的基本知识，以及较为熟练的编程能力，起码要能够轻松地阅读 C 代码。硬件系统结构、汇编语言和编译原理等知识虽然不是必要的，但是有了这些知识，可以让您更加深入地理解各种系统机制。

本书内容组织

本书的内容并没有覆盖 Windows 内核的方方面面，而是从现代操作系统的概念模型出发，重点讨论了进程、线程和内存管理的基本概念以及在 Windows 中的实现细节，也讨论了 Windows 中的并发性和同步处理机制。作为一个对各种硬件设备有广泛兼容性的桌面操作系统，Windows 的 I/O 模型是提供这种兼容能力的基础，因此，Windows 中的 I/O 处理也是本书重点讨论的内容之一。在此基础上，本书还讲述了 Windows 的存储体系结

构，它把 Windows 系统中众多的内核组件，包括缓存管理器、文件系统、磁盘管理驱动程序等，纳入到了一个统一的框架中。

这些内容构成了本书的主体。然而，仅仅这些还不足以形成一个完整的操作系统，Windows 内核中还有其他一些不可或缺的机制，比如内核中的对象管理、配置管理（即注册表服务）、系统内核日志、安全性管理、系统服务分发、LPC（Local Procedure Call，本地过程调用），以及网络和 Windows 子系统等，本书对此也给出了足够详尽的描述，以使读者能够透彻地理解 Windows 操作系统的全貌。另外，Windows 内核的初始化过程也在本书的覆盖范围之中。

在描述以上这些内核机制时，本书以 WRK 为主要的参照系统。WRK 支持 AMD64 和 Intel x86 两种体系结构，本书选择以 Intel x86 作为目标平台，而完全忽略在其他体系结构上的差异。Windows XP (x64)和 Windows Server 2003 SP1 (x86)分别代表了客户端操作系统和服务器端操作系统，它们使用了同一套内核源代码，所以，虽然本书讨论的内容以 Windows Server 2003 SP1 为基准系统，但同样适用于客户端操作系统。

本书讲述的内容显然超越了 WRK 中源代码的范围，对于在 WRK 中未提供源代码的话题，比如即插即用管理器、Windows 子系统和网络等，本书参考了 Microsoft 提供的各种文档，以及通过其他途径提供的源代码，例如 Windows DDK 文档和例子驱动程序。此外，Mark Russinovich 和 David Solomon 合著的《Windows Internals》（第 4、5 版，中文版书名为《深入解析 Windows 操作系统》）也是重要的参考材料。本书在讲解各个话题时注明了所引用资料的出处，对于 Microsoft 提供的参考资料，包括 Windows DDK 文档，本书在参考资料引用条目中注明了这些内容在 MSDN 网站上的链接。

本书内容简介

本书共包括 9 章正文内容和两个附录。内容介绍如下：

第 1 章介绍了操作系统的基础概念、Windows 操作系统的发展历史、Windows 内核的发展，以及学习操作系统的一些前提知识，最后还介绍了操作系统的研究进展。

第 2 章从总体上介绍了 Windows 操作系统。首先讨论现代操作系统的基本模型，以及一些重要部件的职责；然后讨论 Windows 的总体结构，以及有关 WRK 和相应源代码的一些说明。为了让读者对 Windows 内核有一个初步的认识，这一章还介绍了内核中的一些基本概念，包括进程、线程、内存管理、中断、异常、同步等，然后细致地讨论了 Windows 内核中的公共管理设施，包括对象管理器、配置管理器、事件追踪（ETW）和

安全性管理。之所以在这一章中介绍这些公共设施，是为了让读者在学习本书后面章节中的核心概念和机制时，可以方便地直接引用这些管理设施中的功能。这一章的最后还介绍了 Windows 引导过程，包括内核初始化和用户会话建立过程。

第 3 章讨论了 Windows 中的进程和线程管理。首先讨论了进程和线程的基本概念，然后描述了 Windows 中有关进程和线程的重要数据结构，列出了这些数据结构的定义并解释了其中域的含义。接下来介绍了 Windows 进程中的句柄表结构、进程和线程的创建过程和结束处理，以及系统的初始进程和线程。这一章后面还介绍了 Windows 中的线程调度，包括时限管理、环境切换、线程状态转移等内容。

第 4 章介绍了 Windows 中的内存管理。首先介绍了一般性的内存管理知识，包括页式内存管理和段式内存管理。然后介绍了 Windows 的系统内存管理，涉及系统地址空间初始化、换页内存池和非换页内存池的管理算法，以及系统 PTE 区域的管理算法。紧接着介绍了进程地址空间的内存管理，包括地址空间的创建和初始化、虚拟内存的管理数据结构，以及内存区对象（section object）。然后讨论了 Windows 的页面交换机制和物理内存管理。最后讨论了 Windows 的工作集（working set）管理。

第 5 章讨论了 Windows 中的并发和同步机制。这一章首先介绍了有关进程和线程同步的基础知识，然后介绍了 Windows 中的中断和异常处理机制，包括 IRQL（中断请求级别）、中断对象、DPC（延迟过程调用）、定时器管理、APC（异步过程调用），以及异常分发过程。接下来分别介绍了两种类型的同步机制：高 IRQL 时的同步处理，以及基于线程调度的同步机制。Windows 支持大量的同步语义，而且很多同步语义被导出成系统服务，可供应用程序使用。

- 第 6 章讲述了 Windows 的 I/O 模型。首先讨论了 Windows I/O 系统的三个重要部件：I/O 管理器、即插即用管理器和电源管理器，介绍了设备驱动程序的初始化，驱动程序对象、设备对象和文件对象的数据结构，设备的列举过程，以及 Windows 中电源 I/O 请求的处理过程。在此基础上，详细讨论了设备驱动程序，包括驱动程序分类、驱动程序结构模型，并通过一个例子（toaster）说明了驱动程序设备栈结构。最后，介绍了 I/O 处理，包括 IRP（I/O 请求包）的定义、I/O 请求的处理过程以及 I/O 完成处理。

第 7 章讨论了 Windows 的存储管理。这一章分别讨论了 Windows 的缓存管理器、卷的管理以及文件系统。缓存管理器将文件系统和内存管理器有机地结合起来；而文件系统和卷管理驱动程序以及磁盘管理驱动程序联合起来形成了一个存储栈，为 Windows 提供外部存储服务。还讨论了文件系统的过滤机制，以及三个文件系统的实现：RAW（由内核模块提供）、FAT 和 NTFS。

第 8 章讨论了 Windows 的系统服务。这一章首先描述了 Windows 用户模式代码调用内核模式系统服务的详细过程，然后介绍了 Windows 的系统服务分发机制。随后，还介绍了三个重要的跨进程通信系统服务：LPC（Local Procedure Call）、命名管道（Named Pipe）和邮件槽（Mailslot）。

第 9 章介绍了一些高级话题，重点讨论了 Windows 中的网络结构、Windows 子系统，以及 Windows 的内核日志。最后介绍了 Windows 最新版本（Windows Vista、Windows Server 2008 和 Windows 7）中的一些重要变化。

本书正文之后是两个附录。附录 A 讲述了如何编译 WRK，以及在虚拟机环境中运行和调试 WRK。附录 B 介绍了一个基于 WRK 的学生实习项目：KInjectToolKit，KInjectToolKit 允许把一段用户模式代码插入到内核模式下运行，从而提供了在响应内核模式事件时执行用户指定代码的能力。

本书内容与 Windows 系统的版本

本书中介绍的 Windows 核心机制的实现细节以 Windows Server 2003 SP1 为准。由于 Windows 体系结构在不同版本之间基本上没有变化，内核机制的变化也非常微小，所以，本书绝大多数内容适用于自 Windows 2000 以来的各个 Windows 版本，包括 Windows Vista、Windows Server 2008 和 Windows 7。此外，在本书讲解过程中，有些地方特别指出了 Windows Vista 及以后系统的内核变化。

本书的配套工具

为了配合本书中的内容讲解，本书也特别提供了一组工具，读者可以通过本书的主页下载到这些工具，它们不仅可以运行在 WRK 内核的系统中，也可以在其他版本的 Windows 系统上运行。下面是这组工具的简单介绍：

- ProcMon，进程和线程监视工具。它可以实时地显示当前每个处理器正在执行哪些进程和线程。这些信息还可以被转储到一个文本文件中供进一步分析和查看。
- MemMon，内存监视工具。它可以显示当前系统内存空间和每一个进程的用户空间的内存布局。在进程内存空间，MemMon 能够显示进程中的模块、堆、栈等信息。
- DPerfLite，一个可以揭示线程间同步关系的工具。它记录了当前系统中与同步有关的内核操作，允许用户在一个图形界面中直观地观察这些操作，并检查线程之间的各种同步关系。