

Windows操作系统的发展

潘爱民 戴夫·普罗伯特

引言

Windows是微软公司在操作系统领域中的品牌和产品系列。微软公司在操作系统领域中的发展最早开始于MS-DOS操作系统，并在20世纪80年代后期分为2个分支：一个分支是基于MS-DOS的Windows开发平台，最后发展成为Windows 95/98/Me等一系列操作系统；另一个分支则发展成以Windows NT为代表的操作系统系列，经历了Windows NT、Windows 2000、Windows XP/Server 2003，直到Windows Vista/Server 2008。其中，第2个分支是按照20世纪80年代后期的硬件水平进行全新设计的操作系统，包含了现代操作系统的各个要素，也奠定了Windows作为现代操作系统典范的基础。毫无疑问，Windows是一个成功的商业操作系统。在个人计算机产业中，无论是台式机，还是服务器，Windows都是主流的预装操作系统，占据了相当大的市场份额。

本文首先从Windows操作系统的各个发行版本来介绍Windows NT内核的发展史，其次讲述Windows NT内核的结构以及它的一些重要特征，再次讲述在Windows Vista/Server

2008和下一版本中内核的一些最新进展，最后介绍Windows操作系统的发展对软件产业的影响。

Windows操作系统的历史

Windows NT内核是在1988年秋季开始设计的。当时的设计目标是提供一个真正的32位、抢占式调度和采用虚拟内存的网络操作系统。设计的初衷是让这个操作系统可以在多种硬件体系结构和平台上运行。但随着计算机硬件市场的逐步发展，Windows NT最终只能支持英特尔处理器兼容的体系结构。另一个值得一提的设计目标是兼容已有的操作系统，包括对于OS/2¹、POSIX²和Windows³的兼容性支持。但其间由于Windows 3.0在商业上取得巨大成功，Windows NT的设计也作了相应调整，改为以支持Windows应用程序的兼容性为主，而把OS/2兼容性移到一个子系统中。

Windows NT内核在设计之初就有清晰的目标，并且微软还集中了一批以戴维·卡特勒⁴（David Cutler）为首的当时最有内核设计经验的专家来担当此任。在此背景下，不但使Windows NT内核有一个非常整齐、先进的体系结构，也使其综合考虑了安全性、扩展性

¹ 当时IBM公司主导发展的一个兼容主机的操作系统

² 针对UNIX类操作系统的一个接口标准

³ 基于MS-DOS版本的窗口系统

⁴ VAX/VMS、RSX-11M和VAXELN等操作系统的设计者

等要素。该结构在接下来的20年发展历程中基本保持不变，即使近几年硬件有了飞速发展，Windows XP/Vista、Server 2003/2008仍然沿袭了NT内核的基本结构，后文将对其进行讨论。

基于Windows NT内核的第一个操作系统发布于1993年，为Windows NT 3.1。该版本支持Intel i386、Intel i486、MIPS R4000以及Digital Alpha处理器。此后，1994年秋天发布了Windows NT 3.5。1995年5月发布了Windows NT 3.51，并在这个版本中加入了对IBM PowerPC处理器的支持。这些版本均实现了Windows NT的基本设计目标。但是，从NT 3.1到NT 3.51版本变化的主要目的是要让系统运行得更加可靠、速度更快，并且让系统自身的规模变得更加合理、精巧。

Windows NT 3.1 ~ 3.51版本采用了Windows 3.1操作系统的窗口界面风格，并且保持了对MS-DOS和Windows 3.1应用程序的兼容性。之后，Windows NT 4.0于1996年7月发布。此版本完全融合了Windows 95的界面风格，并且对Windows子系统作了适当调整，特别是将GDI⁵转移到内核中，从而显著地提高了系统性能。Windows NT 4.0是一个相对比较成熟和稳定的网络服务器操作系统，并且内置了微软（Microsoft）IIS⁶作为基本的网络（Web）服务器软件。在随后的3年多时间里，微软又为NT 4.0版本发布了一系列补丁，更进一步地完善了NT 4.0作为企业服务器操作系统的能力。

Windows 2000是其后的下一个版本，综合了Windows NT 4.0及其补丁中的各种组件，如活动目录和COM+⁷等。而且，Windows 2000终结了Windows 9x这一分支的发展，也就是说，自Windows 2000以后，微软的客户端操作系统和服务端操作系统合并到一起。Windows 2000包含4个版本，既有客户端版本，也有服务器端版本。这些版本共享

同样的内核文件。除了功能上的综合以外，Windows 2000在安全性方面也有显著提高。它充分发挥了NT内核中安全框架的作用，为企业网络环境提供全方位的安全服务。

微软于2001年8月发布了客户端操作系统Windows XP，并于2003年3月发布了服务器端操作系统Windows Server 2003。Windows XP采用了全新的用户界面风格，并且在稳定性和运行效率方面比Windows 2000有了较大改进。Windows XP首次采用了软件激活机制来应对软件盗版问题，实践证明这是一种有效的软件保护手段。Windows Server 2003在稳定性、安全性和易管理方面都超过了其前身Windows 2000服务器版本，是微软公司在服务器操作系统领域发展的一个重要里程碑，并且也是微软在倡导可信计算（Trustworthy Computing）以后发布的第一个操作系统。顺便提及，从Windows XP和Windows Server 2003开始，Windows操作系统除了发行32位版本，还发行64位版本，标志着Windows操作系统进入了64位计算的领域。

Windows Vista和Windows Server 2008分别是Windows XP和Windows Server 2003的替代版本。Windows Vista于2007年1月发布，除了全面更新用户界面风格之外，还集成和增强了桌面搜索能力。Windows Vista比过去任何一个Windows操作系统更加关注安全性。它避免了让用户直接在管理员权限下运行软件，从而防止了因滥用管理员权限而导致的入侵事件发生。Windows Server 2008集成了Vista的内核更新，并且在内存和处理器等硬件资源的热插拔方面提供了更好的支持，从而减少了服务器系统因需要重新引导而停机的次数和时间。另外，Windows Server 2008对硬件失败和进程崩溃提供了更加全面的报告机制。Windows Server 2008所有的版本都支持多处理器，不再区分单处理器和多处理器版本。值得一提的是，Windows

⁵ Graphics Device Interface，图形设备接口

⁶ Internet Information Server

⁷ 一种基于Windows操作系统的开放的构件标准

Server 2008新引入了对虚拟化的支持，此特性被称为Hyper-V。另外，Windows Vista和Server 2008改进了Windows NT文件系统NTFS（New Technology File System），文件可以在不重新启动的情况下做到自我修复。而在此之前，操作系统必须在重新引导之后才能修复数据损坏。由于采用了更新的SMB2⁸协议，Windows Vista客户与Server 2008服务器之间的数据传输效率也有较大的提高。

Windows操作系统的核心结构——NT内核经过将近20年的发展，已经非常成熟，其扩展性和安全性等重要设计目标在这段发展历程中体现得淋漓尽致。Windows操作系统的每一个重要版本都将这一框架推向顶峰。在微软即将推出的下一个版本（NT内部版本7.0）中，这一框架依然会沿袭下来。

Windows NT内核结构

下面我们将介绍Windows操作系统的基本框架。Windows NT内核结构本身在Windows操作系统的发展过程中并没有重大变化，所以这些内容适用于自Windows 2000以后的各个版本。

图1显示了Windows NT内核的基本结构图。NT内核采用了双模式（dual mode）结构来保护操作系统本身，以避免应用程序的错误波及到操作系统。操作系统核心运行在内核模式（kernel mode）下，应用程序的代码运行在用户模式（user mode）下。每次当应用程序需要用到系统内核或内核的扩展模块（内核驱动程序）所提供的服务时，应用程序都会通过硬件指令从用户模式切换到内核模式，当内核完成了所请求的服务以后，控制权又回到用户模式代码。这样，用户代码和内核代码有各自的运行空间，而且它们可

以访问的内存空间也并不相同，在32位系统中，内核代码可以访问当前进程的整个4GB虚拟地址空间，而用户代码只能访问底端的2GB虚拟地址（或者3GB，如果打开了内核启动开关/3GB的话）。对于每个进程而言，用户空间是独享的，而内核空间则是共享的，所以内核空间有时候也称为系统空间。

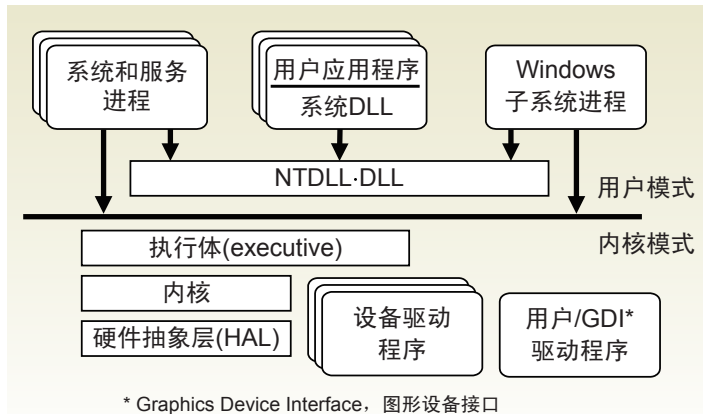


图1 Windows NT内核结构

从结构上看，NT内核分为三层，与硬件直接打交道的这一层称为硬件抽象层（Hardware Abstraction Layer，HAL）。这一层的用意是把所有与硬件相关联的代码逻辑隔离到一个专门的模块中，从而使上面的层次尽可能做到与硬件平台独立。HAL之上是内核层，包含了基本的操作系统原语和功能，如线程和进程、线程调度、中断和异常的处理、同步对象和各种同步机制以及内存管理等。在内核层之上则是执行体（executive）层。这一层的目的是提供一些供上层应用程序或内核驱动程序可以直接调用的功能和语义，包括进程和线程管理、虚拟内存的管理、安全引用监视器（Security Reference Monitor）、I/O管理、即插即用设备管理、电源管理、缓存管理和一组辅助运行函数。在执行体中也包含一个对象管理器，用于统一管理执行体中的对象。内核层和执行体层有分工。内核层实现操作系统的基本机制，把所有的策略决定留给了执行体。执

⁸ Server Message Block。

行体中的对象绝大多数封装了一个或者多个内核对象，并且通过某种方式（比如对象句柄）暴露给应用程序。这种设计体现了机制与策略分离的思想。

在内核中除了执行体、内核和HAL以外，其他的模块都以设备驱动程序的形式存在。Windows操作系统中的设备驱动程序并不一定要对应于物理设备。它既可以创建虚拟设备，也可以完全与设备无关。可以认为驱动程序是Windows内核的一种扩展机制。系统通过它们可以支持新的物理设备或者扩展功能。特别值得一提的是，内核模式下有一个特殊的驱动程序，称为Win32k.sys。它负责管理Windows的窗口和图形设备接口（GDI）。该驱动程序和用户空间中的Windows子系统进程（在Windows中为csrss.exe进程）以及一组系统动态链接库（Dynamic Link Libraries, DLL）合起来构成了Windows子系统。其中Windows子系统进程主要负责控制台窗口以及创建或删除进程和线程等。在Windows NT的早期版本中，Windows除了支持Windows子系统，还支持其他的环境子系统，包括POSIX和OS/2子系统。在Windows XP以后的发行版本中，微软只支持Windows子系统。图1中只显示了Windows子系统。

Windows应用程序通过一组系统动态链接库与操作系统进行通讯。最基本的系统动态链接库包括Kernel32.dll、User32.dll、Advapi32.dll和GDI32.dll。应用程序也可以直接调用NTDLL.DLL中的函数与操作系统进行通讯。Windows操作系统对于应用程序的接口面是大量的API（Application Programming Interfaces, 应用程序接口）函数，这些函数的具体实现都在系统动态链接库中。在必要的时候，系统动态链接库也通过NTDLL.DLL与操作系统进行通讯。所以，Windows应用程序开发人员在调用Windows API函数的时候，实际上调用了系统动态链接库中的函数。NTDLL.DLL负责将用户模式的请求转译为内核模式的服务，所以，此过程会涉及到模式

切换（mode switch），在英特尔 x86硬件体系结构上通过sysenter/sysexit来完成。

总之，Windows应用程序与操作系统打交道有以下四种方式：

1. 应用程序调用Windows API函数，直接访问操作系统内核提供的功能，比如创建一个互斥体（mutex）对象。
2. 应用程序调用GDI32或User32的API函数以及调用Win32k.sys中的功能。
3. 应用程序通过本地过程调用（Local Procedure Calls, LPC）访问Windows子系统进程中的功能，比如创建进程。
4. 应用程序通过DeviceIoControl函数来调用设备驱动程序的功能，这是用户程序访问设备驱动程序的一种标准方法。用户程序也可以通过WriteFile/ReadFile来访问驱动程序的标准数据传输方法。

除了Windows子系统进程csrss.exe以外，另外还有一些系统进程对于Windows系统来说是必不可少的，它们是：

系统空闲进程（Idle）：其标识号（ID）为0，其中每个处理器或核对应一个线程。

系统（System）进程：其标识号为4。它包含了内核模式系统线程。

会话管理器（Session manager, smss.exe）：Windows利用会话来隔离每个登录用户的运行空间。

登录进程（Winlogon.exe）：负责处理交互用户的登录和注销。

本地安全认证服务进程（lsass.exe）：负责执行与认证和安全检查相关的工作。

Shell⁹进程（Explorer.exe）：这是Windows的默认Shell。

服务控制管理器（Services.exe）：负责管理Windows的系统服务。

Windows操作系统特征

上一部分介绍了Windows操作系统的基本框架，包括内核模式的系统组件和用户模式

⁹ Shell就像一个壳层，介于用户和操作系统之间，负责将用户的命令解释为操作系统可以接收的低级语言，并将操作系统响应的信息以用户可以理解的方式来显示。

的系统进程。接下来，我们将从现代操作系统基本功能的角度来介绍Windows操作系统的特征。

进程和线程管理

在Windows操作系统中，进程是系统分配资源的一个基本载体。它定义了一个独立的用户地址空间，所有在这个地址空间中特有的信息都由进程来管理，比如，进程的虚拟内存分配。在Windows中，进程本身并不是系统调度的对象，所以，它不参与CPU资源的分配，但是，每个进程至少有一个线程。操作系统按照线程来分配CPU资源。虽然Windows把空间资源的管理与计算资源的管理分开了，但是两者又有相互关联。下面分别介绍进程和线程。

应用程序可通过进程句柄来访问一个进程。进程句柄对应于执行体中的一个EPROCESS¹⁰对象。在EPROCESS中又封装了一个KPROCESS¹¹，对应于内核层的一个对象。EPROCESS对象中包含的重要信息有：进程ID、各种配额限制（比如可使用多少非换页内存）、虚拟地址描述符、工作集信息（参见下文关于内存管理的介绍）、处理异常和调试的端口信息（允许其他进程通过这些端口接收异常或调试消息）、访问令牌（用于安全认证）、句柄表、进程环境信息（可执行文件的版本、基地址、堆（heap）等）以及Windows子系统所需要的一个信息块。KPROCESS对象是EPROCESS对象的一个组成部分，有时候也称为进程控制块（PCB，Process Control Block），其中包含了涉及线程调度所需要的一些信息，例如内核模式时间和用户模式时间计数、进程的基本优先级、进程亲和性（affinity），更重要的是一个指向进程页目录的指针，代表了该进程的整个地址空间。

在Windows中，线程是真正的调度单元。与之类似，线程的数据结构也分为ETHREAD和KTHREAD，分别对应于执行体层和内核

层所需要的管理信息。ETHREAD中包含的信息有：创建和退出的时间信息、线程ID、线程的启动地址、身份信息、定时器信息，以及正在处理的I/O请求，并且还内嵌一个KTHREAD结构。KTHREAD中包含的信息有：用户模式和内核模式时间、内核栈信息、指向系统服务表的指针、线程调度信息、同步信息、待处理的APC（Asynchronous Procedure Call，异步过程调用）列表、定时器信息和等待对象的列表等等。

值得一提的是，Windows应用程序通过系统API来创建进程和线程，不仅需要执行体层和内核层建立起相应的内部对象，还需要Windows子系统参与并维护相应的信息，因为这些进程和线程也是属于Windows子系统的。详细的过程请参考文献[6]。

Windows以线程为单位进行调度，实现了一个基于优先级的抢占式调度算法。每个线程有一个基本的优先级和一个动态的优先级，系统在选择线程的时候按照动态优先级来决定顺序。在某些情况下线程的优先级会得到提升，比如当I/O完成以后（目的是让那些正在等待I/O完成的线程有更多的机会被立即运行）、前台线程完成了一个等待以后或者图形用户界面（Graphics User Interface，GUI）线程醒来的时候（目的是增加交互式应用程序的响应能力）。系统为每个获得执行权的线程分配一个时限（quantum）值，这是一个大于1的整数，表明了该线程接下来可以运行这么多时间滴答（tick）。这里时间滴答的长度是由时钟分辨率来决定的，比如说在Windows XP系统环境下为15ms。如果一个线程在运行过程中被抢占或者进入等待，那么当它再次获得执行权时，它的时限值或者被重置，或者它继续执行完原来剩下的时间，系统会根据适当的情形做出调度决策。Windows的调度算法考虑到了各种细微之处，因此支持32个优先级别，其中级别16~31用于实时线程的调度，普通用户线程的优先级为8。另外，Windows

¹⁰ 执行体进程。每个Windows进程都有一个EPROCESS结构，里面保存着进程的各种信息和相关结构的指针。

¹¹ 内核进程，包含在执行体进程内。

的调度算法支持多处理器和NUMA（non-uniform memory access，非一致性存储）结构，允许指定进程和线程的亲中性（即限定只在某些CPU上可以运行）。

内存管理

在32位Windows系统中，在默认情况下，一个进程的虚拟空间为2GB，系统空间也是2GB。一方面，操作系统为每个进程维护了相应的页目录和页表，以保证进程中引用的虚拟地址可以被正确地转译成对应的物理地址。另一方面，当物理内存不足时，系统将某些内容转移到磁盘上，当需要这些内容时再从磁盘上读回来，这称为页面换出和换入。

在Windows中，内存管理器是Windows执行体的一部分，提供了一组系统服务，例如分配、释放和管理虚拟内容，供上层应用程序使用。内存管理器也负责处理各种内存异常，例如页面错误（page fault）和访问错误（access fault）。

虚拟地址的转译过程是由硬件体系结构决定的。在x86系统中，每个虚拟地址都映射到一个页表项（PTE，Page Table Entry）。Windows默认使用2级页表结构实现这一映射，每个32位虚拟地址可解释成3个单独的部分：10位页目录索引、10位页表索引和12位字节索引。其中，页目录索引用于定位到该虚拟地址的页表项所在的页表。页表索引用于定位到正确的页表项，而页表项包含了一个虚拟页面所映射的物理地址。字节索引用于找到该物理页面中的正确地址。所以，CPU在翻译虚拟地址的时候，先定位到页目录（切换进程的时候已经放到一个特殊的寄存器CR3中），根据页目录索引定位到正确的页目录项，然后由页目录项得到页表的地址，再根据页表索引定位到正确的页表项，由页表项得到物理页面的地址，最后加上字节索引，得到最终的物理地址。这个过程由CPU自动完成，内存管理器只需小心地维护好这些数据结构。在支持PAE（物理地址扩展）的x86系统中，地址转译过程需要用到三级页表方案。在x64系统中，64位Windows用到了4级页表方案。

Windows使用工作集的概念来动态地管理每一个进程所需要的物理内存。一方面它控制每个进程使用的物理内存数量，确保真正需要内存的进程能够得到资源；另一方面从全局角度调配系统的物理内存，将长时间不访问的页面交换到外存中，从而确保有一定数量的物理页面可用于再分配。Windows使用一个页面帧编号（Page Frame Number，PFN）数据库来管理所有的物理内存。页面帧编号数据库是一个线性阵列，其中的每个页面帧编号（代表了一个物理页面）与对应进程中的页表项之间有双向指针进行连接，因而，从页表项可以快速定位到物理页面，从页面帧编号也可以快速地定位到使用该物理页面的页表项。除了页面帧编号数据库本身，Windows系统也维护了几个列表：备用页面列表（standby page list）、修改页面列表（modified page list）、空闲页面列表（free page list）、坏页面列表（bad page list）以及已修改但不写出的页面列表（modified-no-write page list）。每个物理页面要么在某个进程的工作集中，要么在这些页面列表中移动。

I/O管理

Windows的I/O系统（也称为I/O管理器）为各种设备提供了统一的抽象表示，既为内核扩展提供了统一的标准，也为应用程序使用硬件设备和扩展模块提供了标准接口。驱动程序可以为实际的设备提供硬件访问能力，也可以对内核已有的功能进行扩展。Windows提供的驱动程序模型允许驱动程序层叠起来，串行地处理同一个I/O请求，所以同一个I/O请求可以被多个驱动程序处理。I/O系统负责将I/O请求传递给适当的驱动程序，并且当I/O请求完成时向客户代码返回必要的信息。驱动程序在加载时向I/O系统登记必要的信息，然后接收I/O管理器传递过来的I/O请求，并执行I/O请求中指定的操作，再返回给I/O管理器。I/O请求在内核空间传递过程中是由一个称为IRP（I/O Request Package，I/O请求包）的数据结构来表达的。

一个I/O请求的处理流程是这样的：客户代码通过系统动态链接库发送一个I/O请求时

(比如客户调用ReadFile或WriteFile), 执行体层接到这个I/O请求后, 把此请求交给I/O管理器进行处理。I/O管理器根据I/O请求中的信息, 定位到适当的设备驱动程序, 并且构建一个IRP对象, 然后调用驱动程序中适当的方法, 并把IRP对象传递给此方法。当驱动程序完成了对I/O请求的处理时, 它可以把返回信息存放在IRP对象中, I/O管理器会继续将IRP传递给下一个适当的驱动程序以便进一步处理此I/O请求。此过程一直继续到所有该参与此I/O请求处理的驱动程序都完成其处理工作。最后, I/O管理器从IRP对象中提取出结果信息, 返回客户代码。在IRP数据结构中, 有一个称作栈单元 (Stack Location) 的域用来记录在多个驱动程序之间传递时的状态信息。

Windows的I/O系统在本质上是完全异步进行的, 但是, 往上层有可能被封装成同步的方式, 即发起I/O请求的线程在把I/O请求交给I/O系统以后, 它在一个同步对象上等待该I/O请求的完成通知。所以, 对于上层应用程序或者内核中的其他模块, 它们仍然以同步或异步的方式执行I/O操作。另外一个值得一提的特性是I/O完成端口。这一机制利用多线程模型来实现I/O的完成通知, 它控制执行完成通知的线程的数量, 以避免当大量I/O请求被完成时频繁的线程切换。这对于处理大量客户请求的服务器系统 (像数据库系统和网络服务器) 有重要的意义, 可以显著提升它们的吞吐量。

安全性

安全性是Windows NT的一个设计目标, 所以, 安全性融入在NT内核结构的各个方面。1999年11月, Windows NT 4.0 (with Service Pack 6a) 获得了TCSEC (“橘皮书”) 的C2安全等级的认证。

Windows NT从设计之初就将目标定为多用户环境使用, 所以, 用户身份的认证是一个基本的环节。Windows NT利用会话管理器 (session manager) 来隔离各个用户的运行环境, 每个会话都需要得到用户的身份 (称作令牌)。以后, 在这个会话中的每次资源访问, 都需要经过系统内核中的安全引用监

视器 (Security Reference Monitor, SRM) 检查用户是否有权访问目标资源。每个线程在获得一个对象句柄的时候, 必须要指定它期望在对象上执行哪些类型的动作。对象管理器在得到对象后调用安全引用监视器执行安全访问检查, 然后才返回给线程。安全引用监视器安全模型的本质是一个包含三项输入的等式, 这三项输入是: 一个线程的安全身份、该线程想要获得的对一个对象的访问权以及该对象的安全设置。输出是“是”或“否”, 表明了此安全模型是否授予该线程它所期望的访问权。所以, 从本质上讲, Windows通过安全引用监视器实现了用户对于资源的访问控制。

在Windows中, 除了安全引用监视器作为一个核心的安全组件以外, 还有其他一些重要的安全组件:

1. 本地安全权威子系统 (Lsass.exe进程)。它负责本地系统安全策略 (比如允许哪些用户登录到本地机器上、口令策略、授予用户和用户组的特权, 以及系统安全审计设置)、用户认证, 以及发送安全审计消息到事件日志 (Event Log) 中。

2. Lsass策略数据库。它是包含本地系统安全策略设置的数据库, 例如: 哪些域是可信任的, 从而可以认证用户的登录请求; 谁允许访问系统, 以及如何访问 (交互式登录、网络登录, 或者服务登录); 分配给谁哪些特权; 执行哪一种安全审计等等。

3. 安全账户管理器 (Security Account Manager, SAM) 服务。这是一组子例程。它们负责管理SAM数据库。SAM服务是在\Windows\System32\Samsrv.dll中实现的, 它运行在Lsass进程中。在非域控制器的系统上, SAM数据库包含了已定义的本地用户和用户组, 连同它们的口令和其他属性。在域控制器上, 它保存了该系统的管理员恢复账户的定义及其口令。

4. 登录进程 (Winlogon)。这是一个用户模式进程。它负责响应SAS (secure attention sequence) 和管理交互式登录会话。例如, 当用户登录的时候, Winlogon创建一个用户的Shell进程。

5. 网络登录服务（Netlogon）。这是一个Windows系统服务。它建立起与域控制器之间的安全通道。

6. 内核安全设备驱动程序（KSecDD）。这是一个内核模式函数库。它实现了本地过程调用（LPC）接口。其他的内核模式安全组件，包括加密文件系统（Encrypting File System, EFS），利用该接口与用户模式下的Lsass进行通讯。

Windows的安全模型为Windows提供了全方位的保护，同时也允许插入扩展的安全组件，比如非Windows标准的认证方法（利用指纹、智能卡或虹膜等）。同时，对象粒度的安全控制能力也使得应用系统有较大的灵活性来管理各种资源。

其他特征

以上介绍的是Windows NT结构中最突出显著的功能特征。限于篇幅，其他一些重要的特征本文仅作简单描述。

窗口系统

Windows的窗口系统是在Windows子系统中实现的。它不仅提供了窗口管理和分发消息的能力，同时也规定了应用程序的开发模型。Windows平台上的客户软件无一不建立在Windows窗口系统的基础之上。在Windows的发展历程中，其窗口系统超越其他同类操作系统的优势为它赢得了更多、更广泛的用户。微软在设计和改进窗口系统的过程中积累起来的经验也使得Windows的窗口系统越来越人性化，越来越得到用户的认可，同时也推动了人机交互技术的发展和普及。

注册表

在Windows系统中，注册表是一个重要的管理机制。它既是系统全局配置的存储仓库，也是每个用户的配置信息的存储仓库。注册表并非像很多人想象的那样，是磁盘上的静态数据（一个或一组文件），它也是Windows执行体和内核所维护的各种内存中数据结构的一个窗口。所以，应用程序通过注册表接口（一组APIs）可以访问到系统的一些动态信息。此外，注册表中的信息像文件系统一样，也受到Windows安全模型的保护。

COM/COM+

COM/COM+是Windows操作系统的中间件基础，为应用程序之间的通讯提供了全面支持，允许同一进程内或不同进程中的组件之间很方便地相互通讯。Windows应用程序用到的许多技术，例如，Microsoft Office、Internet Explorer的插件规范，都建立在COM/COM+基础之上。

NTFS文件系统

NTFS文件系统是Windows操作系统的原生文件系统格式。NTFS的空间限制是：卷的大小不超过256TB，每个卷上可存放232-1个文件，每个文件的大小不超过16TB。除了有效地管理磁盘空间以外，NTFS还具有安全访问控制、压缩、可自我恢复、配额限制和加密等高级特性。为了支持恢复能力，NTFS以事务方式记录元数据的改变情况，所以，即使系统被意外停止了，文件系统结构仍可以被修复到一致的状态，文件或目录结构信息不会丢失。（然而，文件数据可能会丢失。）

Windows资源管理器和控制面板

Windows资源管理器（Windows Explorer）是Windows默认的Shell程序。Windows用户通过它来管理和访问各种资源，包括文件系统和网络共享资源。Windows资源管理器的可视化界面降低了人们使用计算机的门槛，这对普及个人计算机的使用起到了重要的作用。控制面板（Control Panel）则是用户管理和配置计算机的主要手段，熟练的计算机用户利用控制面板可以解决一些常见的故障或配置一些资源的特殊用法。

Windows NT内核的最新进展

前面提到，Windows NT内核的结构一直被沿袭下来了，它的扩展能力使得后续版本的Windows操作系统总是能够适应软硬件的发展和变化。尽管Windows内核的结构保持相对稳定，但是内核本身却一直在发展和变化。接下来简单地介绍在Windows操作系统新版本中的一些变化，这里所说的新版本是指Windows Vista、Windows Server 2008以及下一个发布版本（微软内部称为Windows7）。

1. 在线程调度方面, 新的NT内核对线程的运行时间做了更加精细的计量, 以便每个线程有更加公平的机会得到执行, 比如, 从被中断线程的时间计量中剔除中断占用的时间。

2. I/O请求引入了优先级机制, 因此, 高优先级的I/O先得到处理, 低优先级的I/O后处理。像内存管理器需要的页面换入换出I/O可以得到优先处理, 而为文件系统建立索引这样的后台任务可以延迟到I/O空闲的时候再处理。

3. 动态系统地址空间 (Dynamic System Address Space), 内核的地址空间不再像以前那样在系统启动的时候按固定的布局划分好, 而是允许在运行过程中动态分配。

4. 在内存管理方面, 当物理内存被回收的时候, 也考虑了进程和页面的优先级, 这样, 当需要物理内存的时候, 低优先级的内存先被回收。对NUMA系统的内存管理也做了改进, 在内核的许多地方基于NUMA结构的特点来分配和管理内存。

5. 虚拟系统支持Hyper-V (目前仅限于Windows Server 2008版本)。在操作系统底层直接支持虚拟机监视器 (Virtual Machine Monitor), Hyper-V控制对于系统资源的访问和管理, 客户操作系统 (Guest OS) 通过一种称为虚拟总线 (VMBus, Virtual Bus) 的机制与Hyper-V进行通讯。

6. 物理内存管理中的PreFetch和SuperFetch机制。PreFetch是在Windows XP系统中引入的, 其目的是加快应用程序和系统启动过程中数据和代码被读入内存的速度。SuperFetch则更进一步, 它利用物理页面被使用的历史知识, 主动地预读代码或数据到空闲的物理内存中。

7. 内核实现了一个事务管理器, 允许客户代码在访问资源时具有原子性, 即对资源的修改要么成功, 要么完全恢复原样, 不会破坏资源的状态。这种能力可以避免由于内核资源的状态破坏而造成系统不稳定。

8. 进程崩溃后的错误处理。WER (Windows Error Reporting) 服务代替以前的Dr. Watson机制。

9. 出于安全性考虑, 可执行模块 (包括内核模块本身和系统动态链接库) 被加载到内存中时, 系统会选择地址空间中的256个随机点。这一机制被称为地址空间加载随机化 (ASLR, Address Space Load Randomization)。

以上列出的并非是所有的内核改进, 有关更进一步的信息, 请参考文献[1-5]。

Windows操作系统与软件产业共同发展

Windows操作系统自诞生以来, 每一次发布都影响着包括硬件工业和软件工业的整个计算机工业界。由于每一个Windows操作系统新版本都顾及到了软件和硬件的兼容性, 所以, 工业界总可以或快或慢地跟随Windows的步伐, 为企业用户或家庭用户带来丰富多彩的应用软件。

Windows操作系统一面世即衍生了一种新型的软件开发模式。它可以让应用程序有漂亮丰富的界面, 尤其是微软自己开发的桌面应用软件证明了这种开发模式可以吸引更多的计算机用户, 办公软件、家庭财务管理软件、游戏软件、辅助教学软件等都可以建立在Windows操作系统平台上。基于Windows的软件开发行业很快发展起来, 各种开发工具和用于快速开发的软件包出现在市场上。从软件技术人员的教育和培训, 到软件开发实践, 再到应用软件的教育和推广, 这一过程本质上构成了一个产业链, 使20世纪90年代初的软件产业得到了迅猛发展。

Windows的图形用户界面 (GUI) 开发模式奠定了桌面应用软件开发的基础。这一开发模式也在不断发展和演进, 从早期的基于Windows API的开发方式, 到20世纪90年代中后期基于COM/ActiveX技术的开发模型, 再到21世纪初建立起来的基于.NET和CLR (Common Language Runtime, 通用语言运行时) 的开发模式, 一直到最近的基于WPF (Windows Presentation Foundation) 的应用开发模式。Windows提供的应用开发模型和工具也在不断完善。计算机科学领域中的研究成

果被转化到软件开发实践中，极大地提高了软件开发效率，软件产品的开发周期不断缩短。在此过程中，软件产业持续发展，其规模持续扩大。对软件开发人员的大量需求，导致教育领域也进行改革和发展。国内高校为软件产业输送了大量技术人才。

20世纪90年代，互联网迅速发展和普及，Windows操作系统也迎头赶上。系统内置的浏览器Internet Explorer不仅提供了友好的图形界面让计算机用户方便地获得互联网上的各种信息，而且所提供的扩展支持（ActiveX技术）也使得互联网上的网络页面能够动起来，而不只是一些静态的数据。这极大地推动了网络应用的发展，让网络应用有了更强的交互能力。同时，Windows操作系统服务器版本内置了IIS¹²作为网络服务器软件，使得无论在企业网络内部还是在互联网上，部署网络服务器都非常方便，这又从另一个角度推动了网络应用的发展。Windows操作系统对于

互联网的支持在后来的版本中又不断加强。

作为一个在商业领域获得巨大成功的操作系统，Windows已经深入人心。但是，由于它始终由微软独家控制，系统软件开发人员难以掌握Windows的核心开发技术，人们总是质疑它的安全性。比如，硬件厂商或者系统安全软件（像防病毒软件）厂商必须掌握Windows系统底层的实现机制才能开发出更加有效的驱动程序或者安全软件。在这种形势下，微软于2006年上半年开放了一份内核源代码供研究和教学使用，这份代码被称为WRK（Windows Research Kernel）^[7]。它是根据当时最新的版本（Windows Server 2003 SP1）从Windows产品代码中摘录出来的，其中包括Windows内核中绝大多数源代码。学校的学生和科研人员通过对这份代码及相关资料的学习，可以很快掌握Windows的核心技术。这也进一步加强了Windows与软件产业之间的联系纽带。

（下转69页）



潘爱民

微软亚洲研究院，研究员。主要研究兴趣包括Windows交互式性能、操作系统和网络安全。



戴夫·普罗伯特

（Dave Probert）

微软Windows核心操作系统部门的内核设计师，目前主要致力于下一代Windows的设计和开发。

参考文献

- [1] Mark Russinovich, Inside the Windows Vista Kernel: Part 1, <http://technet.microsoft.com/en-us/magazine/cc162494.aspx>
- [2] Mark Russinovich, Inside the Windows Vista Kernel: Part 2, <http://technet.microsoft.com/en-us/magazine/cc162480.aspx>
- [3] Mark Russinovich, Inside the Windows Vista Kernel: Part 3, <http://technet.microsoft.com/en-us/magazine/cc162458.aspx>
- [4] Mark Russinovich, Inside Windows Server 2008 Kernel Changes, <http://technet.microsoft.com/en-us/magazine/cc194386.aspx>
- [5] David Probert, et al., Windows Server 2008 Kernel Architecture, TechEd Israel 2008, mms://iwebcast1.you-niversity.com/iarchives/WSR440_SG01W.WMV
- [6] Mark Russinovich & David Solomon著，潘爱民译，深入解析Windows操作系统，电子工业出版社，2007
- [7] Microsoft Corp., Windows Research Kernel, <http://www.microsoft.com/resources/sharedsource/licensing/researchkernel.mspix>

¹² Internet Information Services

(上接49页)

最后,简单提一下,本文前述的W indow s是指安装在个人计算机(包括桌面机器、笔记本计算机和PC服务器)上的操作系统,W indow s操作系统另有一分支,称为W indow s C E。它是指用于便携式设备(包括手机、掌上计算机)的操作系统。基于W indow s在操作系统领域中的技术优势,这一分支在过去的几年中也得到了迅速发展,本文不再讲述。

结语

W indow s作为一个现代操作系统,无论在技术方面,还是在市场方面,都是成功的。本文介绍了W indow s操作系统的历史,并且从全局角度分析了W indow s操作系统的内核结构以及它的一些重要特征,其后讲述了W indow s系统内核的一些最新进展,最后总结了W indow s操作系统在发展过程中对于软件产业的影响。■

征稿启事

为跟踪计算机新技术前沿的研究方向,促进计算机技术水平的提高和发展,本刊欢迎广大专家赐稿,共同讨论和研究计算机新技术前沿的研究方向和发展水平。

本刊编辑部联系方式,电话(010)62562503转19,电子邮箱:ccc@ictac.cn。