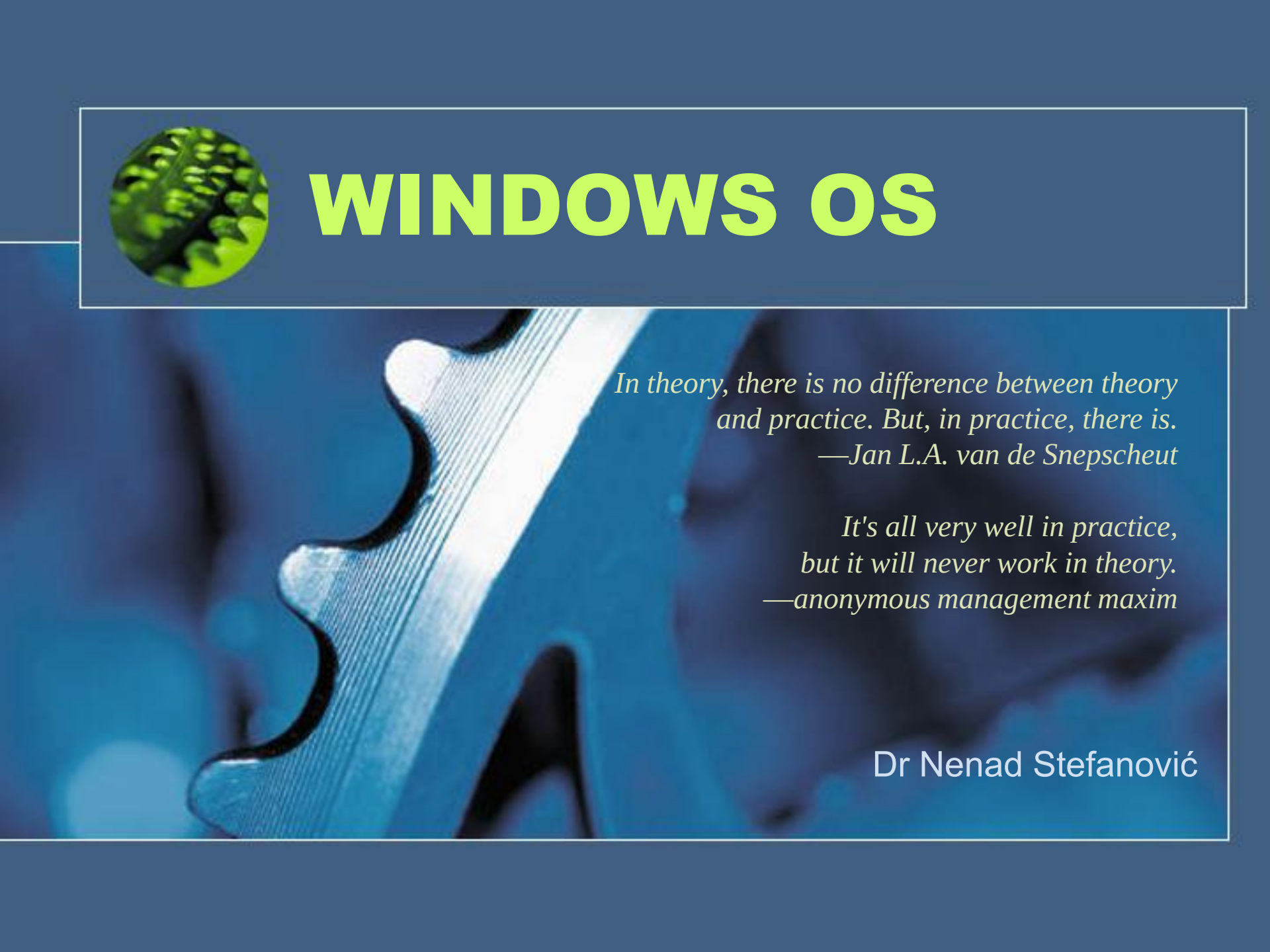
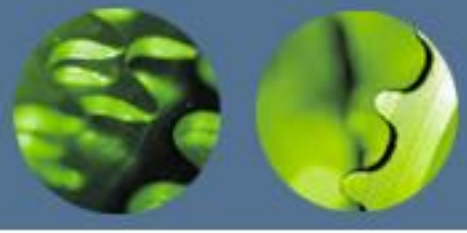


WINDOWS OS

In theory, there is no difference between theory and practice. But, in practice, there is.
—Jan L.A. van de Snepscheut

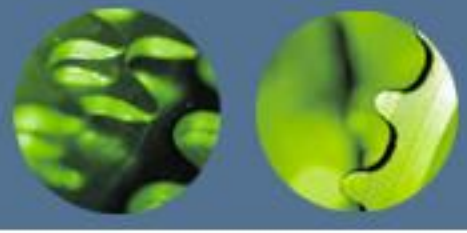
It's all very well in practice, but it will never work in theory.
—anonymous management maxim

Dr Nenad Stefanović



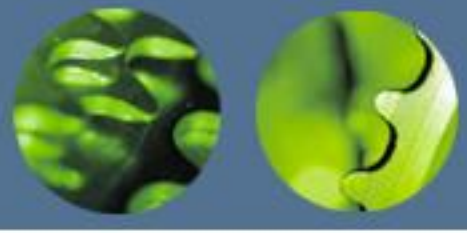
Agenda

- Windows Architecture
- Introducing Windows 7
- Upgrading to Windows 7
- Choosing and Installing a Windows 7 Edition
- Configuring and Troubleshooting Hardware and Devices
- The Windows 7 Interface
- Windows 7 File System
- Windows 7 Performance and Maintenance
- Security in Windows 7
- Mobile Computing in Windows 7
- Configuring and Troubleshooting Networking
- Command Line for Windows 7
- Digital Media in Windows 7
- Windows 7 Advanced Tips&Tricks

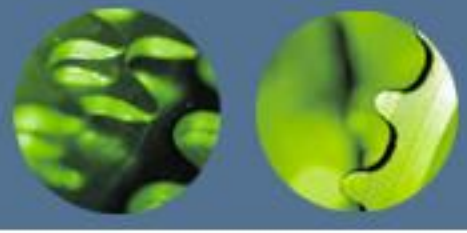


History

- In 1988, Microsoft decided to develop a “new technology” (NT) portable operating system that supported both the OS/2 and POSIX APIs
- Originally, NT was supposed to use the OS/2 API as its native environment but during development NT was changed to use the Win32 API, reflecting the popularity of Windows 3.0

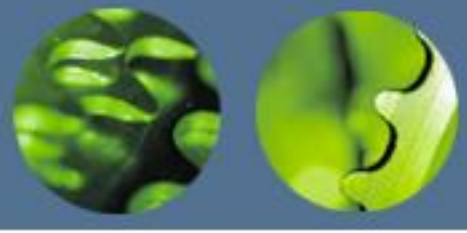


WINDOWS INTERNALS



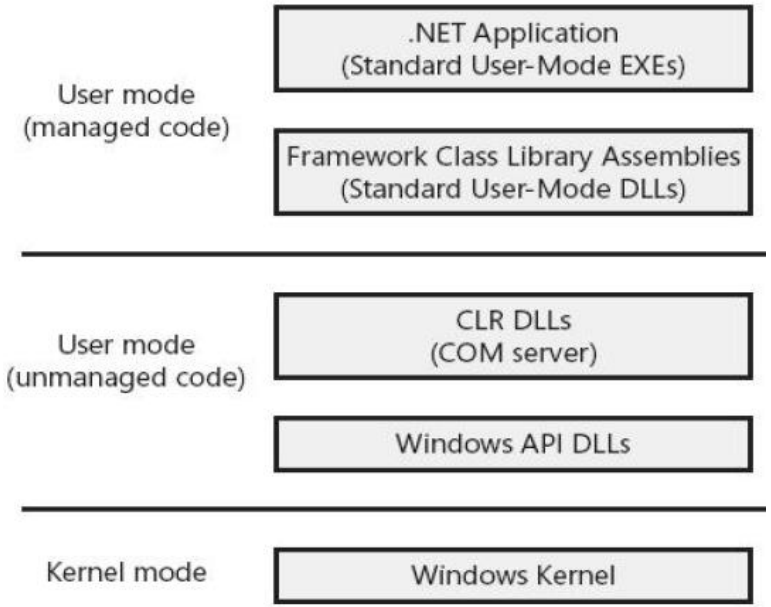
Windows API

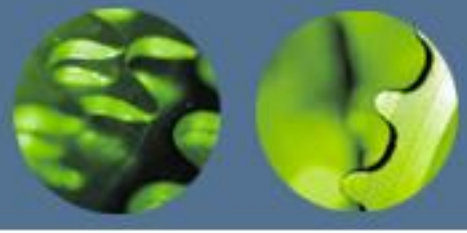
- The Windows application programming interface (API) is the system programming interface to the Windows operating system family.
- The Windows API consists of thousands of callable functions, which are divided into the following major categories:
  - Base Services
  - Component Services
  - User Interface Services
  - Graphics and Multimedia Services
  - Messaging and Collaboration
  - Networking
  - Web Services



.Net

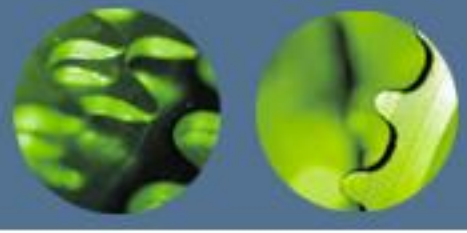
- The .NET Framework consists of a library of classes called the Framework Class Library (FCL) and a Common Language Runtime (CLR) that provides a managed code execution environment with features such as just-in-time compilation, type verification, garbage collection, and code access security.





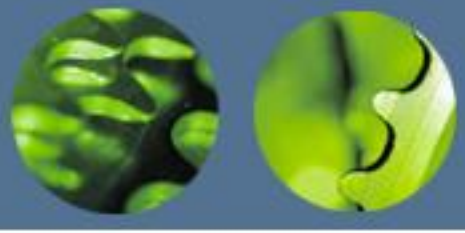
Windows

- 32-bit preemptive multitasking operating system for Intel microprocessors
- Key goals for the system:
  - portability
  - security
  - POSIX compliance
  - multiprocessor support
  - extensibility
  - international support
  - compatibility with MS-DOS and MS-Windows applications
- Uses a micro-kernel architecture
- Available in several versions: Home, Premium, Business Ultimate
- New version – Windows 2008, is now available; Windows 7 beta



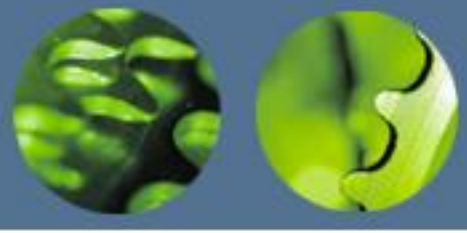
Design Principles

- Extensibility — layered architecture
  - Executive, which runs in protected mode, provides the basic system services
  - On top of the executive, several server subsystems operate in user mode
  - Modular structure allows additional environmental subsystems to be added without affecting the executive
- Portability — Windows can be moved from on hardware architecture to another with relatively few changes
  - Written in C and C++
  - Processor-dependent code is isolated in a dynamic link library (DLL) called the “hardware abstraction layer” (HAL)



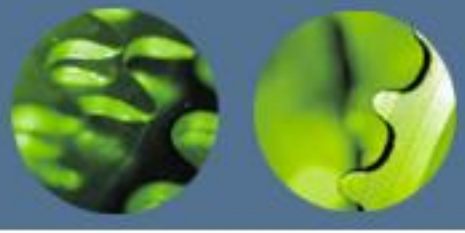
Design Principles (Cont.)

- Reliability — Windows uses hardware protection for virtual memory, and software protection mechanisms for operating system resources
- Compatibility — applications that follow the IEEE 1003.1 (POSIX) standard can be complied to run on Windows without changing the source code
- Performance — Windows subsystems can communicate with one another via high-performance message passing
  - Preemption of low priority threads enables the system to respond quickly to external events
  - Designed for symmetrical multiprocessing
- International support — supports different locales via the national language support (NLS) API



Operating System Model

- In most multiuser operating systems, applications are separated from the operating system itself—the operating system kernel code runs in a privileged processor mode (referred to as kernel), with access to system data and to the hardware; application code runs in a nonprivileged processor mode (called user mode), with a limited set of interfaces available, limited access to system data, and no direct access to hardware.
- When a user-mode program calls a system service, the processor traps the call and then switches the calling thread to kernel mode.
- When the system service completes, the operating system switches the thread context back to user mode and allows the caller to continue.

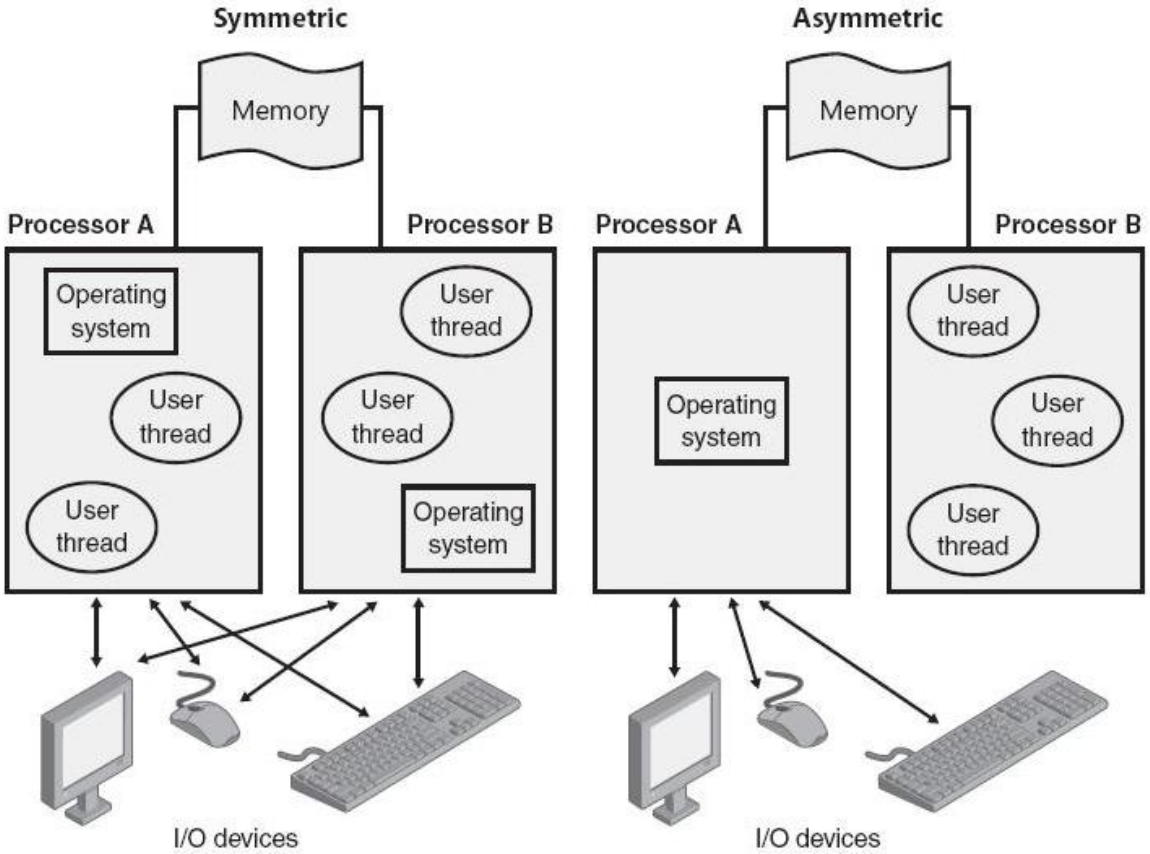


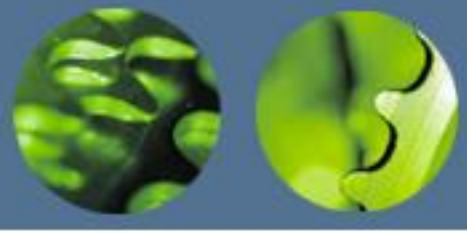
Operating System Model

- All these operating system components are, of course, fully protected from errant applications because applications don't have direct access to the code and data of the privileged part of the operating system (although they can quickly call other kernel services).
- This protection is one of the reasons that Windows has the reputation for being both robust and stable as an application server and as a workstation platform yet fast and nimble from the perspective of core operating system services, such as virtual memory management, file I/O, networking, and file and print sharing.
- The kernel-mode components of Windows also embody basic object-oriented design principles. For example, they don't in general reach into one another's data structures to access information maintained by individual components. Instead, they use formal interfaces to pass parameters and access and/or modify data structures.

Symmetric Multiprocessing

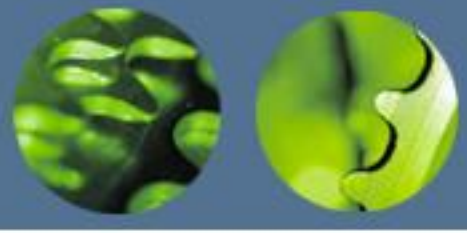
- Multitasking is the operating system technique for sharing a single processor among multiple threads of execution. When a computer has more than one processor, however, it can execute multiple threads simultaneously.





Symmetric Multiprocessing

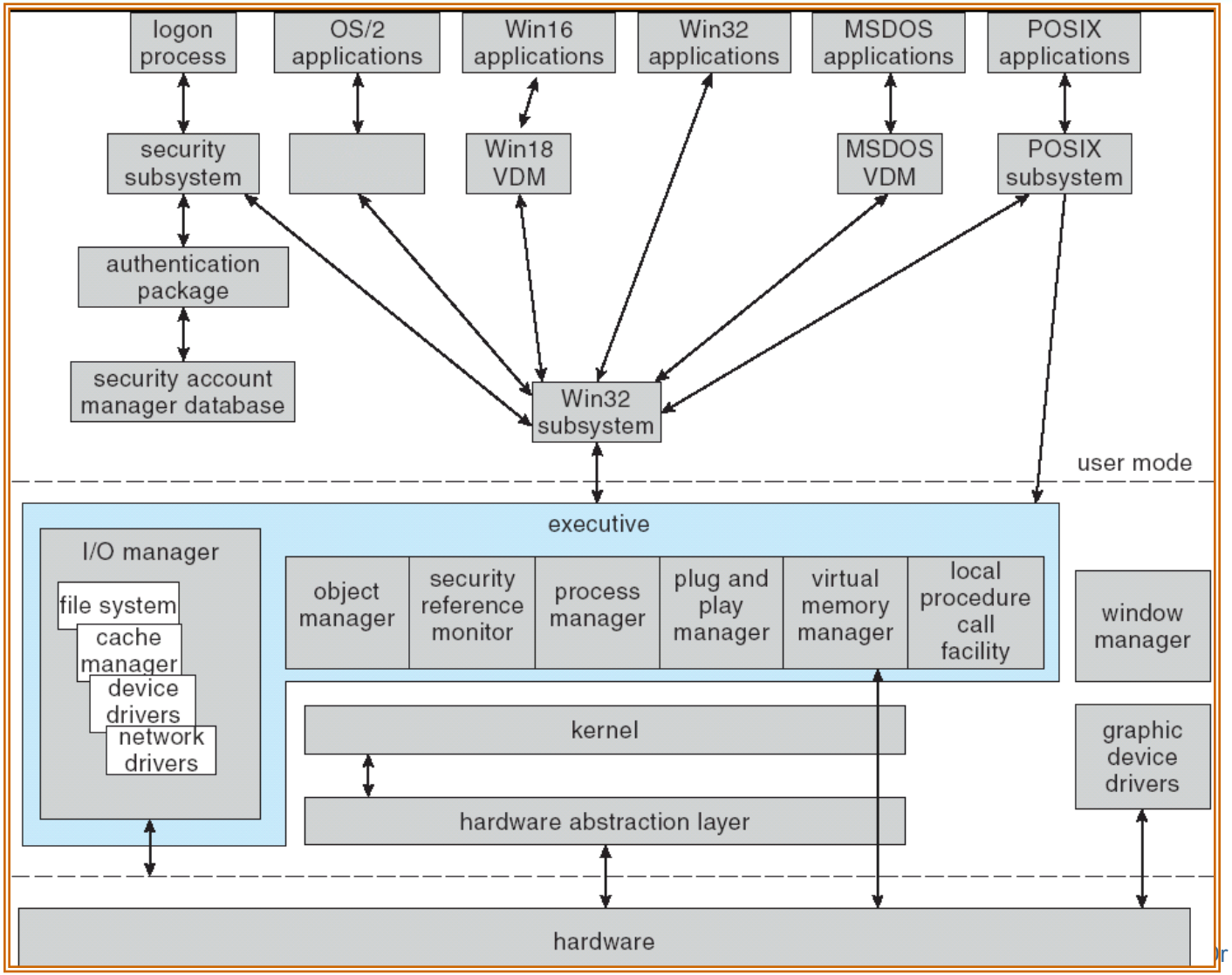
- Windows Vista, 7 and Windows Server 2008 also support two modern types of multiprocessor systems: hyperthreading and NUMA (non-uniform memory architecture).
- Hyperthreading is a technology introduced by Intel that provides many logical processors on one physical processor. Each logical processor has its CPU state, but the execution engine and onboard cache are shared.
- In NUMA systems, processors are grouped in smaller units called nodes. Each node has its own processors and memory and is connected to the larger system through a cachecoherent interconnect bus.

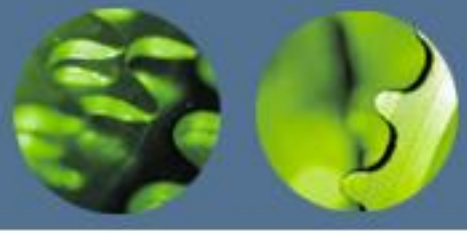


Windows Architecture

- Layered system of modules
- Protected mode — HAL, kernel, executive
- User mode — collection of subsystems
  - Environmental subsystems emulate different operating systems
  - Protection subsystems provide security functions

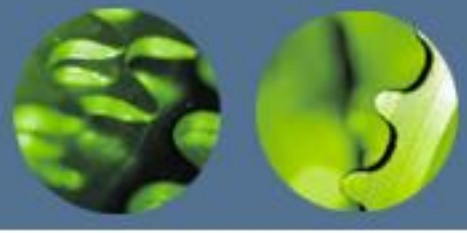
Depiction of Windows Architecture





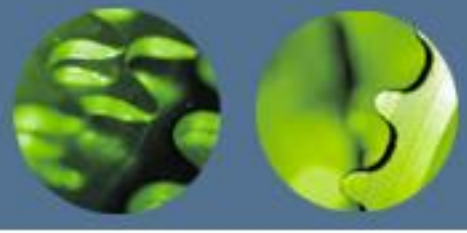
Windows Subsystem

- The Windows subsystem consists of the following major components:
  - The environment subsystem process (Csrss.exe) loads three DLLs (Basesrv.dll, Winsrv.dll, and Csrsrv.dll)
  - The kernel-mode device driver (Win32k.sys) – (Window Manager, GDI, DirectX wrappers, etc.)
  - Subsystem DLLs (such as Kernel32.dll, Advapi32.dll, User32.dll, and Gdi32.dll) translate documented Windows API functions into the appropriate and mostly undocumented kernel-mode system service calls to Ntoskrnl.exe and Win32k.sys.
  - Graphics device drivers are hardware-dependent graphics display drivers, printer drivers, and video miniport drivers.



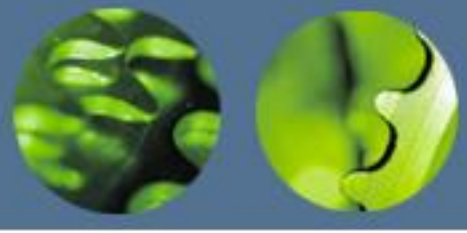
POSIX Subsystem

- POSIX, an acronym loosely defined as “a portable operating system interface based on UNIX,” refers to a collection of international standards for UNIX-style operating system interfaces. The POSIX standards encourage vendors implementing UNIX-style interfaces to make them compatible so that programmers can move their applications easily from one system to another.
- Windows Vista/7 and Windows Server 2008 provide the Windows Subsystem for Unix-based Applications (SUA), which includes an enhanced POSIX subsystem environment that provides nearly 2,000 UNIX functions and 300 UNIX-like tools and utilities.



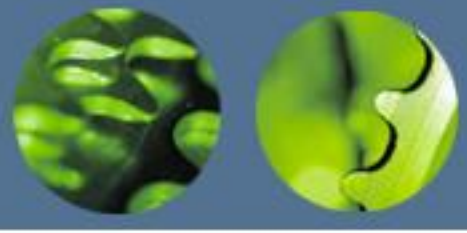
Ntdll.dll

- Ntdll.dll is a special system support library primarily for the use of subsystem DLLs. It contains two types of functions:
  - System service dispatch stubs to Windows executive system services
  - Internal support functions used by subsystems, subsystem DLLs, and other native images



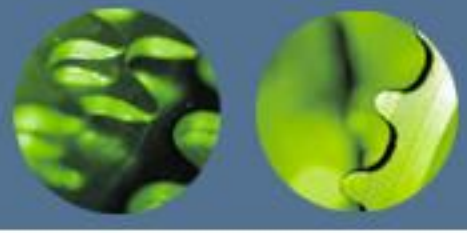
Executive

- The Windows executive is the upper layer of Ntoskrnl.exe. (The kernel is the lower layer.)
- The executive includes the following types of functions:
  - Functions that are exported and callable from user mode.
  - Device driver functions that are called through the use of the DeviceIoControl function.
  - Functions that can be called only from kernel mode that are exported.
  - Functions that are exported and callable from kernel mode but are not documented in the WDK.
  - Functions that are defined as global symbols but are not exported.
  - Functions that are internal to a module that are not defined as global symbols.



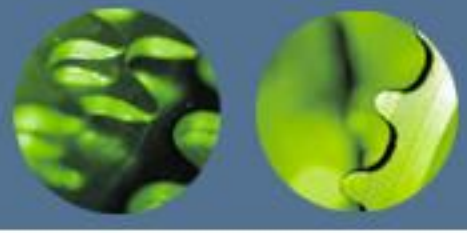
Executive

- The executive contains the following major components:
  - The configuration manager
  - The process and thread manager
  - The security reference monitor
  - The I/O manager
  - The Plug and Play (PnP) manager
  - The power manager
  - The Windows Driver Model Windows Management Instrumentation routines
  - The cache manager
  - The memory manager
  - The logical prefetcher and Superfetch
  - The object manager
  - The Advanced LPC facility
  - A broad set of common run-time library functions
  - Executive support routines, such as system memory allocation
  - The hypervisor library, part of the Hyper-V stack in Windows Server 2008
  - The kernel transaction manager
  - Event Tracing
  - The Windows diagnostic infrastructure
  - The Windows hardware error architecture



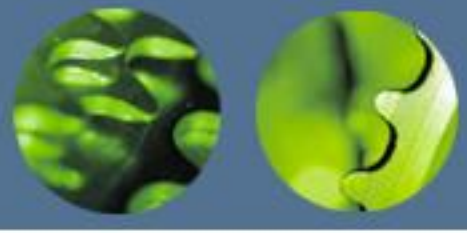
System Components — Kernel

- Foundation for the executive and the subsystems
- Never paged out of memory; execution is never preempted
- Four main responsibilities:
  - thread scheduling
  - interrupt and exception handling
  - low-level processor synchronization
  - recovery after a power failure
- Kernel is object-oriented, uses two sets of objects
  - **dispatcher objects** control dispatching and synchronization (events, mutants, mutexes, semaphores, threads and timers).
  - **control objects** (asynchronous procedure calls, interrupts, power notify, power status, process and profile objects)



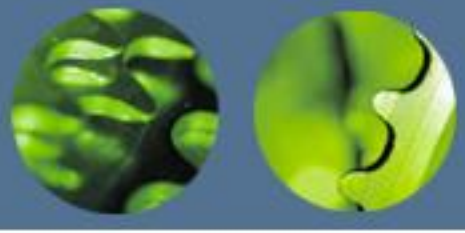
System Components — Kernel

- The kernel consists of a set of functions in Ntoskrnl.exe that provide fundamental mechanisms (such as thread scheduling and synchronization services) used by the executive components, as well as low-level hardware architecture–dependent support (such as interrupt and exception dispatching), that is different on each processor architecture.
- The kernel provides a low-level base of well-defined, predictable operating system primitives and mechanisms that allow higher-level components of the executive to do what they need to do.



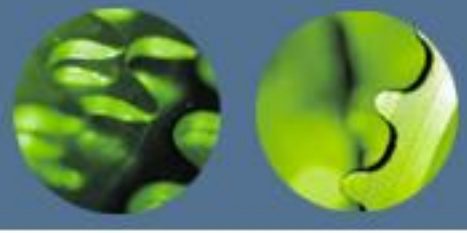
Hardware Abstraction Layer

- As mentioned at the beginning of this chapter, one of the crucial elements of the Windows design is its portability across a variety of hardware platforms.
- The hardware abstraction layer (HAL) is a key part of making this portability possible.
- The HAL is a loadable kernel-mode module (Hal.dll) that provides the low-level interface to the hardware platform on which Windows is running.
- It hides hardware-dependent details such as I/O interfaces, interrupt controllers, and multiprocessor communication mechanisms—any functions that are both architecture-specific and machine-dependent.
- Windows Vista/7 and Windows Server 2008 have the ability to detect at boot-up time which HAL should be used.



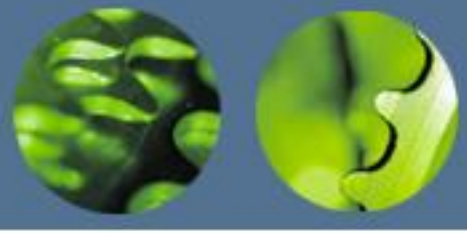
Device Drivers

- Device drivers are loadable kernel-mode modules (typically ending in .sys) that interface between the I/O manager and the relevant hardware.
- They run in kernel mode in one of three contexts:
  - In the context of the user thread that initiated an I/O function
  - In the context of a kernel-mode system thread
  - As a result of an interrupt (and therefore not in the context of any particular process or thread—whichever process or thread was current when the interrupt occurred)



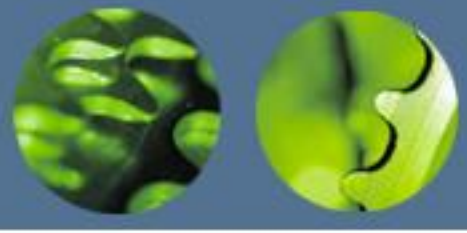
Device Drivers

- There are several types of device drivers:
  - Hardware device drivers manipulate hardware (using the HAL) to write output to or retrieve input from a physical device or network. There are many types of hardware device drivers, such as bus drivers, human interface drivers, mass storage drivers, and so on.
  - File system drivers are Windows drivers that accept file-oriented I/O requests and translate them into I/O requests bound for a particular device.
  - File system filter drivers, such as those that perform disk mirroring and encryption, intercept I/Os and perform some added-value processing before passing the I/O to the next layer.
  - Network redirectors and servers are file system drivers that transmit file system I/O requests to a machine on the network and receive such requests, respectively.
  - Protocol drivers implement a networking protocol such as TCP/IP, NetBEUI, and IPX/SPX.
  - Kernel streaming filter drivers are chained together to perform signal processing on data streams, such as recording or displaying audio and video.



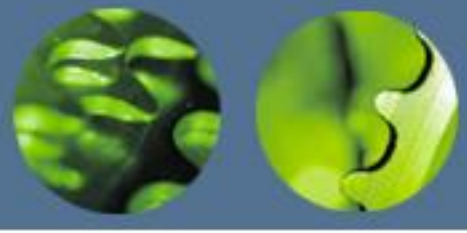
Device Drivers

- Windows 2000 added support for Plug and Play, Power Options, and an extension to the Windows NT driver model called the Windows Driver Model (WDM).
- The Windows Driver Foundation (WDF) simplifies Windows driver development by providing two frameworks: the Kernel-Mode Driver Framework (KMDF) and the User-Mode Driver Framework (UMDF). Developers can use KMDF to write drivers for Windows 2000 SP4 and later, while UMDF supports Windows XP and later.



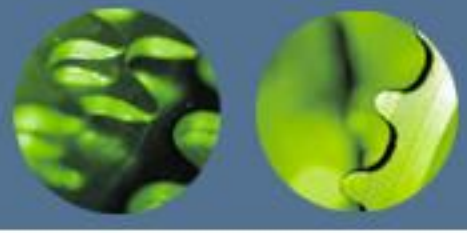
System Processes

- The following system processes appear on every Windows system. (Two of these—Idle and System—are not full processes, as they are not running a user-mode executable.)
  - Idle process (contains one thread per CPU to account for idle CPU time)
  - System process (contains the majority of the kernel-mode system threads)
  - Session manager (Smss.exe)
  - Local session manager (Lsm.exe)
  - Windows subsystem (Csrss.exe)
  - Session 0 initialization (Wininit.exe)
  - Logon process (Winlogon.exe)
  - Service control manager (Services.exe) and the child service processes it creates (such as the system-supplied generic service-host process, Svchost.exe)
  - Local security authentication server (Lsass.exe)



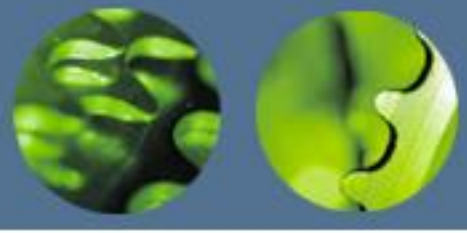
Session Manager (Smss)

- The session manager (%SystemRoot%\Smss.exe) is the first user-mode process created in the system.
- The kernel-mode system thread that performs the final phase of the initialization of the executive and kernel creates the actual Smss process.
- The session manager is responsible for a number of important steps in starting Windows, such as opening additional page files, performing delayed file rename and delete operations, and creating system environment variables.
- It also launches the subsystem processes (normally just Csrss.exe) and either the Wininit or Winlogon processes, the former of which in turn creates the rest of the system processes.
- Smss also creates user sessions. When Smss creates the first interactive user session (the console session) or when a request to create a session is received, it creates a copy of itself inside that session.



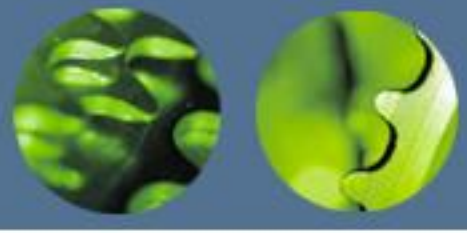
Winlogon, LogonUI, LSASS, and Userinit

- The Windows logon process (%SystemRoot%\Winlogon.exe) handles interactive user logons and logoffs. Winlogon is notified of a user logon request when the secure attention sequence (SAS) keystroke (Ctrl+Alt+Delete) combination is entered.
- Because Winlogon is a critical system process on which the system depends, credential providers and the UI to display the logon dialog box run inside a child process of Winlogon called LogonUI. When Winlogon detects the SAS, it launches this process, which initializes the credential providers.
- Once the username and password have been captured, they are sent to the local security authentication server process (%SystemRoot%\Lsass.exe) to be authenticated. LSASS calls the appropriate authentication package (implemented as a DLL) to perform the actual verification, such as checking whether a password matches what is stored in the Active Directory or the SAM.



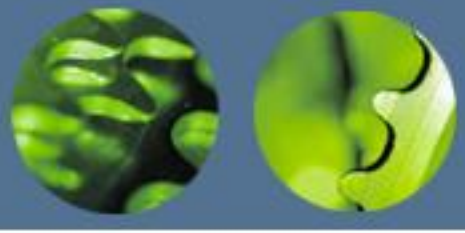
Winlogon, LogonUI, LSASS, and Userinit

- Upon a successful authentication, LSASS calls a function in the security reference monitor (for example, NtCreateToken) to generate an access token object that contains the user's security profile.
- If User Account Control (UAC) is used and the user logging on is a member of the administrators group or has administrator privileges, LSASS will create a second, restricted version of the token.
- This access token is then used by Winlogon to create the initial process(es) in the user's session.
- The initial process(es) are stored in the registry value Userinit under the registry key HKLM\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\Winlogon.
- The default is Userinit.exe, but there can be more than one image in the list.



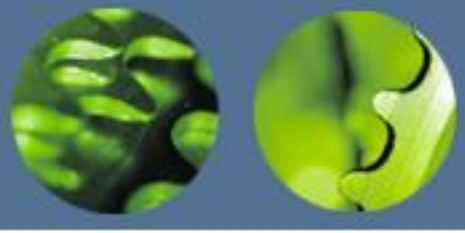
Service Control Manager (SCM)

- The service control manager is a special system process running the image %SystemRoot%\Services.exe that is responsible for starting, stopping, and interacting with service processes.
- Service programs are really just Windows images that call special Windows functions to interact with the service control manager to perform such actions as registering the service's successful startup, responding to status requests, or pausing or shutting down the service.
- Services are defined in the registry under HKLM\SYSTEM\CurrentControlSet\Services.



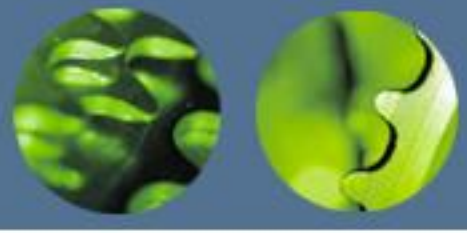
Basic types of user-mode processes

- Fixed (or hardwired) system support processes, such as the logon process and the Session Manager, that are not Windows services.
- Service processes that host Windows services, such as the Task Scheduler and Print Spooler services. Services generally have the requirement that they run independently of user logons. Many Windows server applications, such as Microsoft SQL Server and Microsoft Exchange Server, also include components that run as services.
- User applications, which can be one of five types: Windows 32-bit, Windows 64-bit, Windows 3.1 16-bit, MS-DOS 16-bit, or POSIX 32-bit.
- Environment subsystem server processes, which implement part of the support for the operating system environment, or personality presented to the user and programmer.



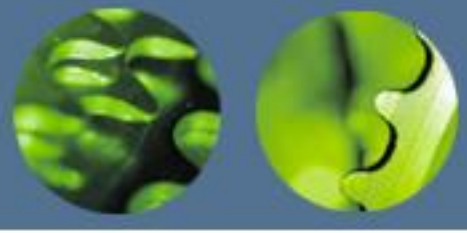
The kernel-mode components of Windows

- The Windows executive contains the base operating system services, such as memory management, process and thread management, security, I/O, networking, and interprocess communication.
- The Windows kernel consists of low-level operating system functions, such as thread scheduling, interrupt and exception dispatching, and ultiprocessor synchronization.
- Device drivers include both hardware device drivers, which translate user I/O function calls into specific hardware device I/O requests, as well as file system and network drivers.
- The hardware abstraction layer (HAL) is a layer of code that isolates the kernel, device drivers, and the rest of the Windows executive from platform-specific hardware differences.
- The windowing and graphics system implements the graphical user interface (GUI) functions (better known as the Windows USER and GDI functions), such as dealing with windows, user interface controls, and drawing.



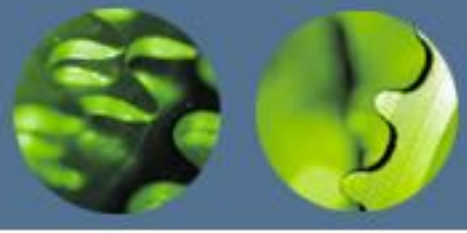
Core Windows System Files

File Name	Components
Ntoskrnl.exe	Executive and kernel
Ntkrnlpa.exe (32-bit systems only)	Executive and kernel with support for Physical Address Extension (PAE), which allows addressing of up to 64 GB of physical memory
Hal.dll	Hardware abstraction layer
Win32k.sys	Kernel-mode part of the Windows subsystem
Ntdll.dll	Internal support functions and system service dispatch stubs to executive functions
Kernel32.dll, Advapi32.dll, User32.dll, Gdi32.dll	Core Windows subsystem DLLs



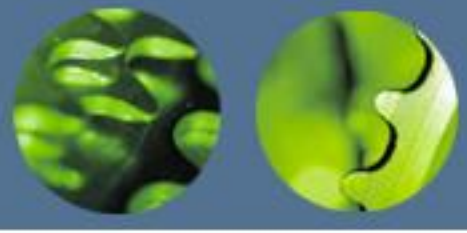
Kernel — Process and Threads

- The process has a virtual memory address space, information (such as a base priority), and an affinity for one or more processors
- Threads are the unit of execution scheduled by the kernel's dispatcher
- Each thread has its own state, including a priority, processor affinity, and accounting information
- A thread can be one of six states: *ready, standby, running, waiting, transition*, and *terminated*



Kernel — Scheduling

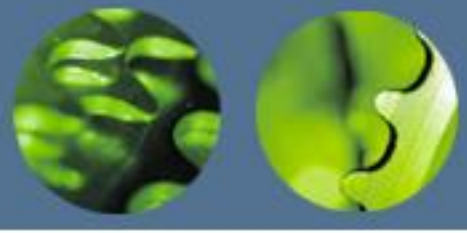
- The dispatcher uses a 32-level priority scheme to determine the order of thread execution
- Priorities are divided into two classes
  - The real-time class contains threads with priorities ranging from 16 to 31
  - The variable class contains threads having priorities from 0 to 15
- Characteristics of Windows priority strategy
  - Trends to give very good response times to interactive threads that are using the mouse and windows
  - Enables I/O-bound threads to keep the I/O devices busy
  - Complete-bound threads soak up the spare CPU cycles in the background



Kernel — Scheduling (Cont.)

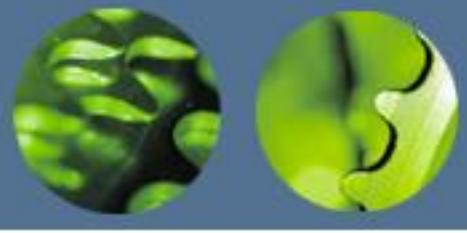
- Scheduling can occur when a thread enters the ready or wait state, when a thread terminates, or when an application changes a thread's priority or processor affinity

- Real-time threads are given preferential access to the CPU; but Windows does not guarantee that a real-time thread will start to execute within any particular time limit
  - This is known as **soft realtime**



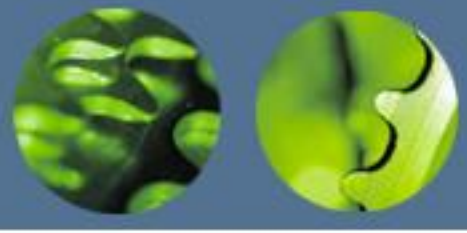
Windows Interrupt Request Levels

interrupt levels	types of interrupts
31	machine check or bus error
30	power fail
29	interprocessor notification (request another processor to act; e.g., dispatch a process or update the TLB)
28	clock (used to keep track of time)
27	profile
3–26	traditional PC IRQ hardware interrupts
2	dispatch and deferred procedure call (DPC) (kernel)
1	asynchronous procedure call (APC)
0	passive



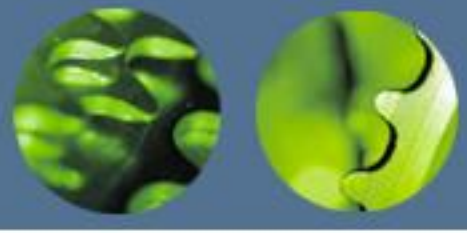
Kernel — Trap Handling

- The kernel provides trap handling when exceptions and interrupts are generated by hardware of software
- Exceptions that cannot be handled by the trap handler are handled by the kernel's *exception dispatcher*
- The interrupt dispatcher in the kernel handles interrupts by calling either an interrupt service routine (such as in a device driver) or an internal kernel routine
- The kernel uses spin locks that reside in global memory to achieve multiprocessor mutual exclusion



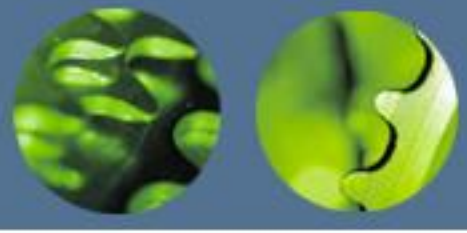
Windows Objects

- **Encapsulation:** An object consists of one or more items of data, called attributes, and one or more procedures that may be performed on those data, called services. The only way to access the data in an object is by invoking one of the object's services. Thus, the data in the object can easily be protected from unauthorized use and from incorrect use (e.g., trying to execute a nonexecutable piece of data).
- **Object class and instance:** An object class is a template that lists the attributes and services of an object and defines certain object characteristics. The operating system can create specific instances of an object class as needed. For example, there is a single process object class and one process object for every currently active process. This approach simplifies object creation and management.
- **Inheritance:** This is not supported at the user level but is supported to some extent within the Executive. For example, Directory objects are examples of container objects. One property of a container object is that the objects they contain can inherit properties from the container itself. As an example, suppose you have a directory in the file system that has its compressed flag set. Then any files you might create within that directory container will also have their compressed flag set.
- **Polymorphism:** Internally, Windows uses a common set of API functions to manipulate objects of any type; this is a feature of polymorphism. However, Windows is not completely polymorphic because there are many APIs that are specific to specific object types.



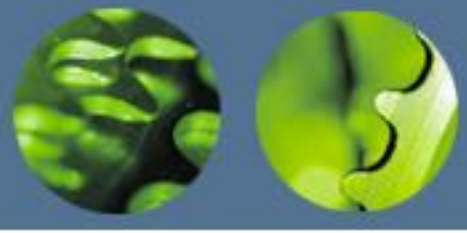
Executive — Object Manager

- Windows uses objects for all its services and entities; the object manger supervises the use of all the objects.
  - Generates an object **handle**
  - Checks security
  - Keeps track of which processes are using each object
- Objects are manipulated by a standard set of methods, namely `create, open, close, delete, query name, parse` and `security`



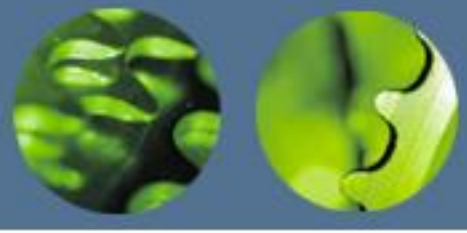
Executive — Naming Objects

- The Windows executive allows any object to be given a name, which may be either permanent or temporary
- Object names are structured like file path names in MS-DOS and UNIX
- Windows implements a **symbolic link object**, which is similar to **symbolic links** in UNIX that allow multiple nicknames or aliases to refer to the same file
- A process gets an object handle by creating an object by opening an existing one, by receiving a duplicated handle from another process, or by inheriting a handle from a parent process
- Each object is protected by an access control list



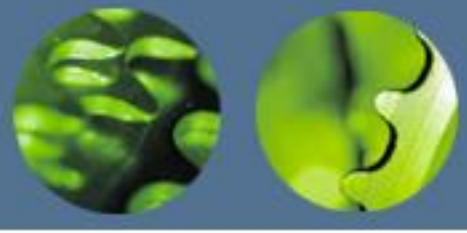
Executive — Virtual Memory Manager

- The design of the VM manager assumes that the underlying hardware supports virtual to physical mapping a paging mechanism, transparent cache coherence on multiprocessor systems, and virtual addressing aliasing
- The VM manager in Windows uses a page-based management scheme with a page size of 4 KB
- The Windows VM manager uses a two step process to allocate memory
  - The first step reserves a portion of the process's address space
  - The second step commits the allocation by assigning space in the Windows paging file



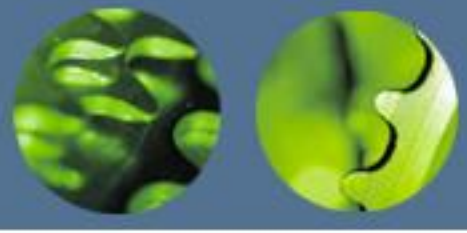
Executive — Process Manager

- Provides services for creating, deleting, and using threads and processes
- Issues such as parent/child relationships or process hierarchies are left to the particular environmental subsystem that owns the process



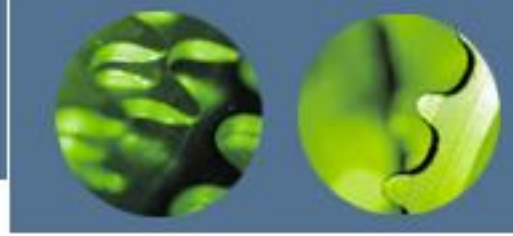
Executive — Local Procedure Call Facility

- The LPC passes requests and results between client and server processes within a single machine
- In particular, it is used to request services from the various Windows subsystems
- When a LPC channel is created, one of three types of message passing techniques must be specified
  - First type is suitable for small messages, up to 256 bytes; port's message queue is used as intermediate storage, and the messages are copied from one process to the other
  - Second type avoids copying large messages by pointing to a shared memory section object created for the channel
  - Third method, called *quick LPC* was used by graphical display portions of the Win32 subsystem

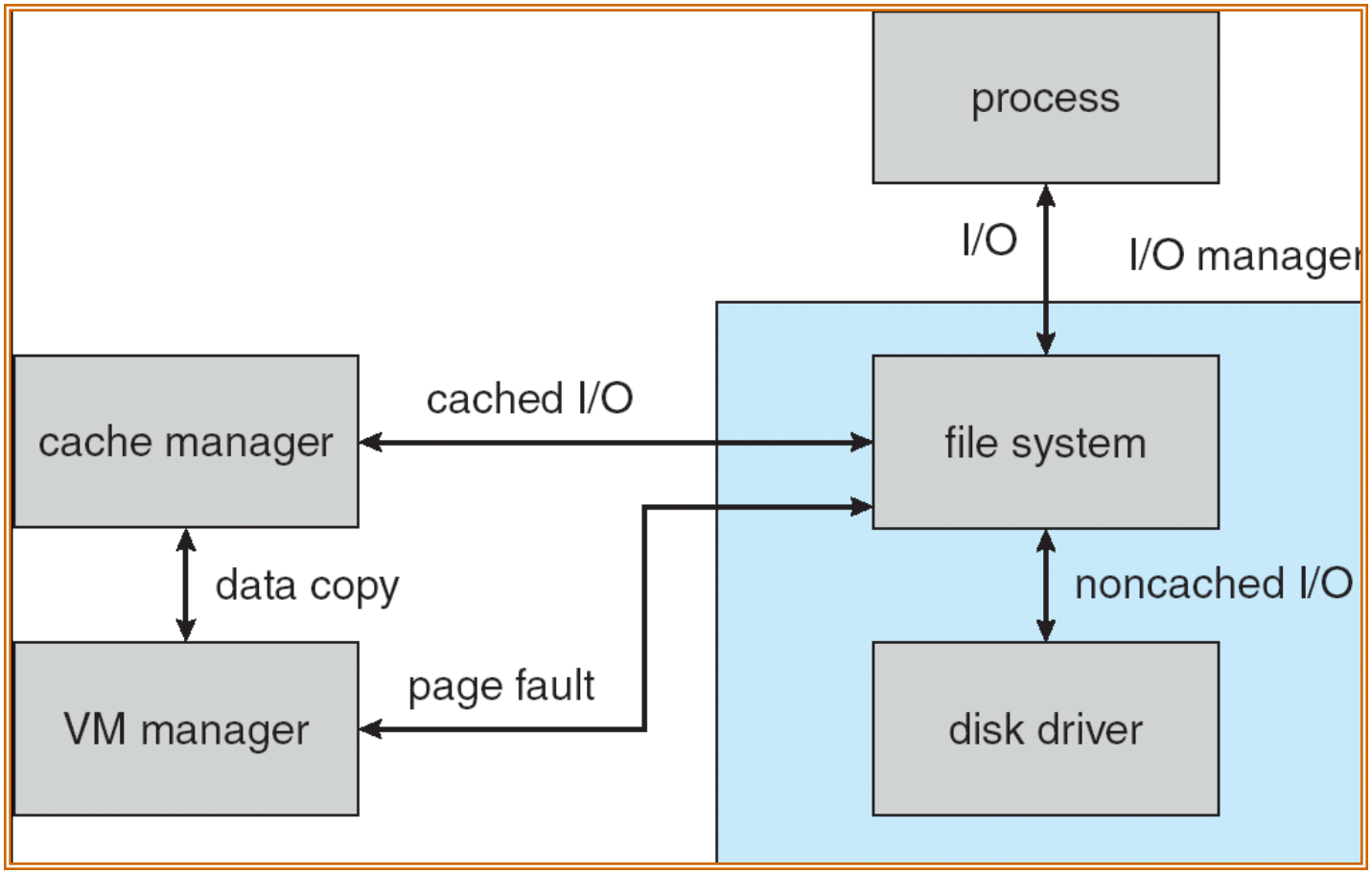


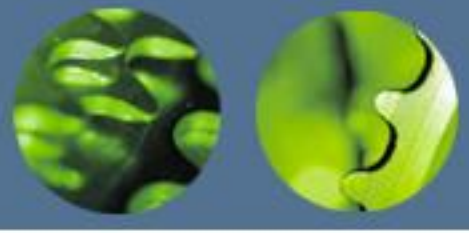
Executive — I/O Manager

- The I/O manager is responsible for
  - file systems
  - cache management
  - device drivers
  - network drivers
- Keeps track of which installable file systems are loaded, and manages buffers for I/O requests
- Works with VM Manager to provide memory-mapped file I/O
- Controls the 2000 cache manager, which handles caching for the entire I/O system
- Supports both synchronous and asynchronous operations, provides time outs for drivers, and has mechanisms for one driver to call another



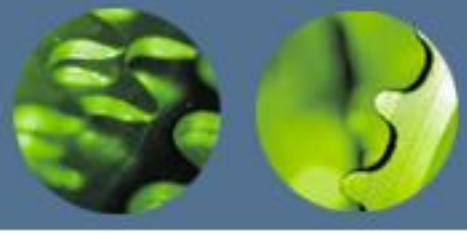
File I/O


process
I/O
I/O manager
cache manager
cached I/O
file system
data copy
noncached I/O
VM manager
page fault
disk driver



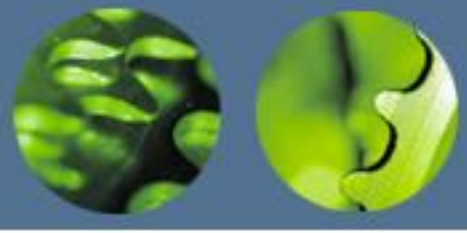
Executive — Security Reference Monitor

- The object-oriented nature of Windows enables the use of a uniform mechanism to perform runtime access validation and audit checks for every entity in the system
- Whenever a process opens a handle to an object, the security reference monitor checks the process's security token and the object's access control list to see whether the process has the necessary rights



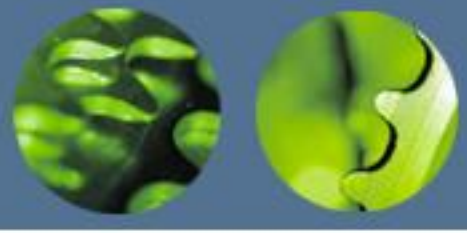
Executive – Plug-and-Play Manager

- Plug-and-Play (PnP) manager is used to recognize and adapt to changes in the hardware configuration
- When new devices are added (for example, PCI or USB), the PnP manager loads the appropriate driver
- The manager also keeps track of the resources used by each device



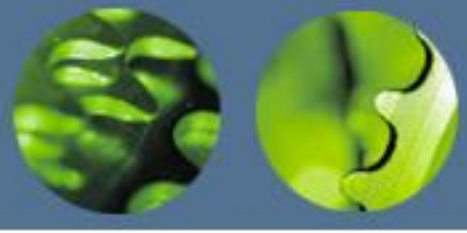
Environmental Subsystems

- User-mode processes layered over the native Windows executive services to enable Windows to run programs developed for other operating system
- Windows uses the Win32 subsystem as the main operating environment; Win32 is used to start all processes
  - It also provides all the keyboard, mouse and graphical display capabilities
- MS-DOS environment is provided by a Win32 application called the *virtual dos machine* (VDM), a user-mode process that is paged and dispatched like any other Windows thread



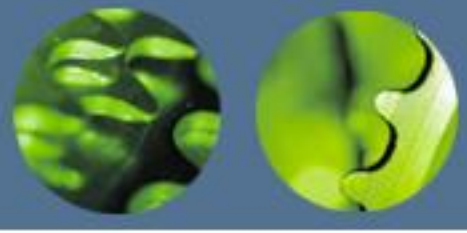
Environmental Subsystems (Cont.)

- 16-Bit Windows Environment:
  - Provided by a VDM that incorporates *Windows on Windows*
  - Provides the Windows 3.1 kernel routines and sub routines for window manager and GDI functions
- The POSIX subsystem is designed to run POSIX applications following the POSIX.1 standard which is based on the UNIX model



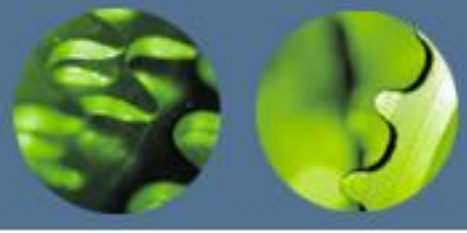
Environmental Subsystems (Cont.)

- OS/2 subsystems runs OS/2 applications
- Logon and Security Subsystems authenticates users logging to Windows systems
  - Users are required to have account names and passwords
- The authentication package authenticates users whenever they attempt to access an object in the system
  - Windows uses Kerberos as the default authentication package



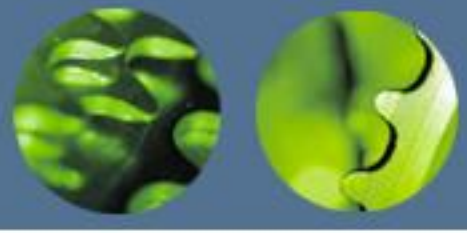
File System

- The fundamental structure of the Windows file system (NTFS) is a *volume*
  - Created by the Windows disk administrator utility
  - Based on a logical disk partition
  - May occupy a portions of a disk, an entire disk, or span across several disks
- All *metadata*, such as information about the volume, is stored in a regular file
- NTFS uses *clusters* as the underlying unit of disk allocation
  - A cluster is a number of disk sectors that is a power of two
  - Because the cluster size is smaller than for the 16-bit FAT file system, the amount of internal fragmentation is reduced



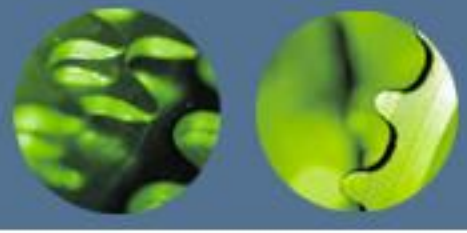
File System — Internal Layout

- NTFS uses logical cluster numbers (LCNs) as disk addresses
- A file in NTFS is not a simple byte stream, as in MS-DOS or UNIX, rather, it is a structured object consisting of attributes
- Every file in NTFS is described by one or more records in an array stored in a special file called the Master File Table (MFT)
- Each file on an NTFS volume has a unique ID called a file reference.
  - 64-bit quantity that consists of a 48-bit file number and a 16-bit sequence number
  - Can be used to perform internal consistency checks
- The NTFS name space is organized by a hierarchy of directories; the index root contains the top level of the B+ tree



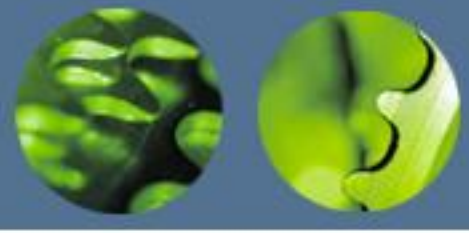
File System — Recovery

- All file system data structure updates are performed inside transactions that are logged
  - Before a data structure is altered, the transaction writes a log record that contains redo and undo information
  - After the data structure has been changed, a commit record is written to the log to signify that the transaction succeeded
  - After a crash, the file system data structures can be restored to a consistent state by processing the log records



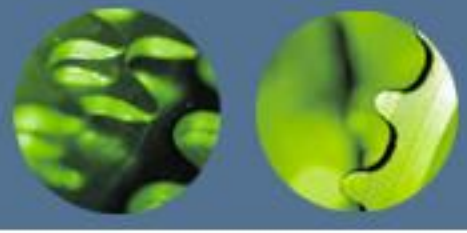
File System — Recovery (Cont.)

- This scheme does not guarantee that all the user file data can be recovered after a crash, just that the file system data structures (the metadata files) are undamaged and reflect some consistent state prior to the crash
- The log is stored in the third metadata file at the beginning of the volume
- The logging functionality is provided by the Windows *log file service*



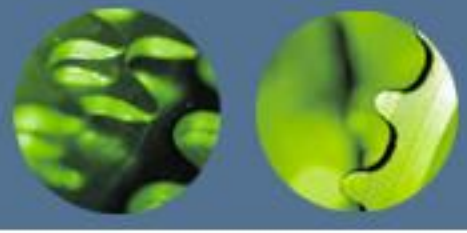
File System — Security

- Security of an NTFS volume is derived from the Windows object model
- Each file object has a security descriptor attribute stored in this MFT record
- This attribute contains the access token of the owner of the file, and an access control list that states the access privileges that are granted to each user that has access to the file



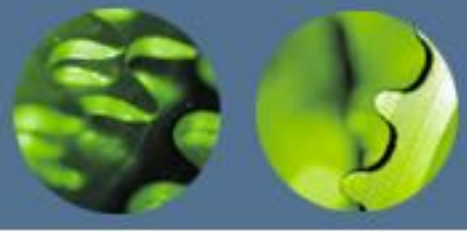
Volume Management and Fault Tolerance

- FtDisk, the fault tolerant disk driver for Windows, provides several ways to combine multiple SCSI disk drives into one logical volume
- Logically concatenate multiple disks to form a large logical volume, a **volume set**
- Interleave multiple physical partitions in round-robin fashion to form a **stripe set** (also called RAID level 0, or “disk striping”)
  - Variation: **stripe set with parity**, or RAID level 5.
- Disk mirroring, or RAID level 1, is a robust scheme that uses a **mirror set** — two equally sized partitions on tow disks with identical data contents
- To deal with disk sectors that go bad, FtDisk, uses a hardware technique called *sector sparing* and NTFS uses a software technique called *cluster remapping*



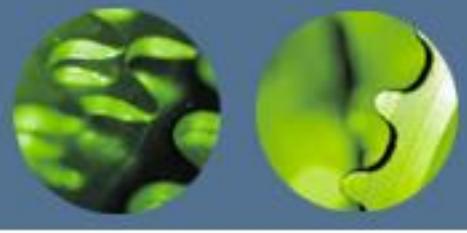
File System — Compression

- To compress a file, NTFS divides the file's data into *compression units*, which are blocks of 16 contiguous clusters
- For sparse files, NTFS uses another technique to save space
  - Clusters that contain all zeros are not actually allocated or stored on disk
  - Instead, gaps are left in the sequence of virtual cluster numbers stored in the MFT entry for the file
  - When reading a file, if a gap in the virtual cluster numbers is found, NTFS just zero-fills that portion of the caller's buffer



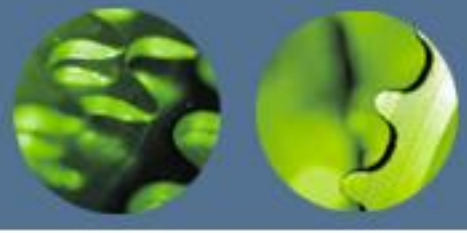
File System — Reparse Points

- A reparse point returns an error code when accessed. The reparse data tells the I/O manager what to do next
- Reparse points can be used to provide the functionality of UNIX mounts
- Reparse points can also be used to access files that have been moved to offline storage



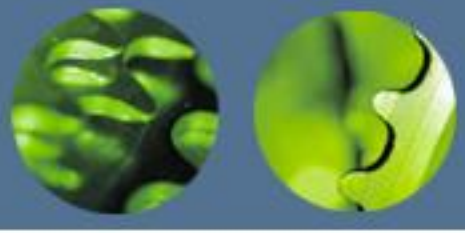
Networking

- Windows supports both peer-to-peer and client/server networking; it also has facilities for network management
- To describe networking in Windows, we refer to two of the internal networking interfaces:
  - NDIS (Network Device Interface Specification) — Separates network adapters from the transport protocols so that either can be changed without affecting the other
  - TDI (Transport Driver Interface) — Enables any session layer component to use any available transport mechanism
- Windows implements transport protocols as drivers that can be loaded and unloaded from the system dynamically



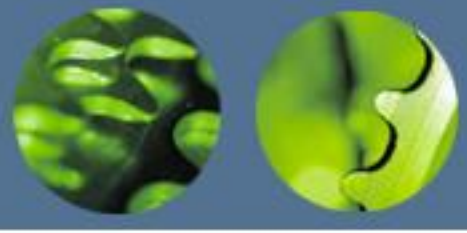
Networking — Protocols

- The server message block (SMB) protocol is used to send I/O requests over the network. It has four message types:
  - Session control
  - File
  - Printer
  - Message
- The network basic Input/Output system (NetBIOS) is a hardware abstraction interface for networks. Used to:
  - Establish logical names on the network
  - Establish logical connections of sessions between two logical names on the network
  - Support reliable data transfer for a session via NetBIOS requests or *SMBs*



Networking — Protocols (Cont.)

- NetBEUI (NetBIOS Extended User Interface): default protocol for Windows 95 peer networking and Windows for Workgroups; used when Windows wants to share resources with these networks
- Windows uses the TCP/IP Internet protocol to connect to a wide variety of operating systems and hardware platforms
- PPTP (Point-to-Point Tunneling Protocol) is used to communicate between Remote Access Server modules running on Windows machines that are connected over the Internet
- The Windows NWLink protocol connects the NetBIOS to Novell NetWare networks

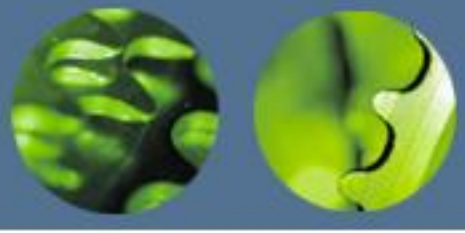


Networking — Protocols (Cont.)

- The Data Link Control protocol (DLC) is used to access IBM mainframes and HP printers that are directly connected to the network
- Windows systems can communicate with Macintosh computers via the Apple Talk protocol if an Windows Server on the network is running the Windows Services for Macintosh package

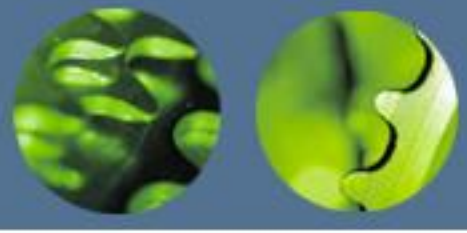
Networking — Dist. Processing Mechanisms

- Windows supports distributed applications via named NetBIOS,named pipes and mailslots, Windows Sockets, Remote Procedure Calls (RPC), and Network Dynamic Data Exchange (NetDDE) and Web services
- NetBIOS applications can communicate over the network using NetBEUI, NWLink, or TCP/IP
- Named pipes are connection-oriented messaging mechanism that are named via the uniform naming convention (UNC)
- Mailslots are a connectionless messaging mechanism that are used for broadcast applications, such as for finding components on the network
- Winsock, the windows sockets API, is a session-layer interface that provides a standardized interface to many transport protocols that may have different addressing schemes



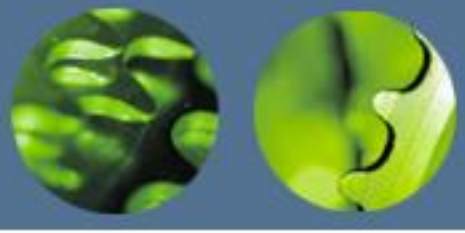
Distributed Processing Mechanisms (Cont.)

- The Windows RPC mechanism follows the widely-used Distributed Computing Environment standard for RPC messages, so programs written to use Windows RPCs are very portable
  - RPC messages are sent using NetBIOS, or Winsock on TCP/IP networks, or named pipes on LAN Manager networks
  - Windows provides the Microsoft *Interface Definition Language* to describe the remote procedure names, arguments, and results



Networking — Domains

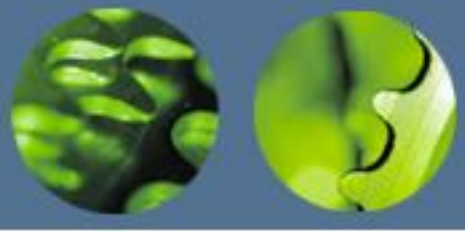
- Windows Servers uses the concept of a domain to manage global access rights within groups
- A domain is a group of machines running NT server that share a common security policy and user database
- Windows provides three models of setting up trust relationships
  - *One way, A trusts B*
  - *Two way, transitive, A trusts B, B trusts C so A, B, C trust each other*
  - *Crosslink – allows authentication to bypass hierarchy to cut down on authentication traffic*



Name Resolution in TCP/IP Networks

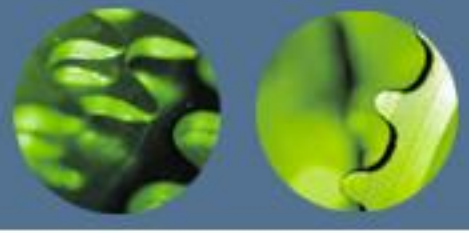
- On an IP network, name resolution is the process of converting a computer name to an IP address
  e.g., www.bell-labs.com resolves to 135.104.1.14

- Windows provides several methods of name resolution:
  - Windows Internet Name Service (WINS)
  - broadcast name resolution
  - domain name system (DNS)
  - a host file
  - an LMHOSTS file



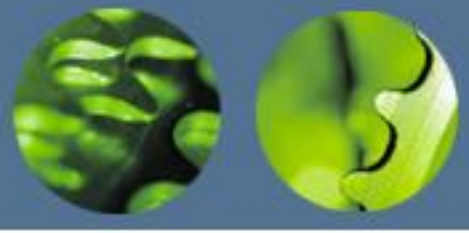
Name Resolution (Cont.)

- WINS consists of two or more WINS servers that maintain a dynamic database of name to IP address bindings, and client software to query the servers
- WINS uses the Dynamic Host Configuration Protocol (DHCP), which automatically updates address configurations in the WINS database, without user or administrator intervention



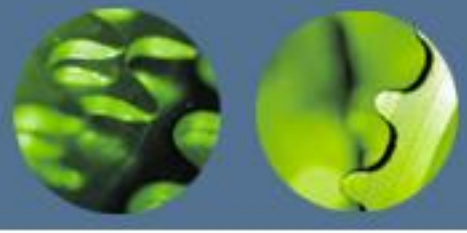
Programmer Interface — Access to Kernel Obj.

- A process gains access to a kernel object named XXX by calling the CreateXXX function to open a *handle* to XXX; the handle is unique to that process
- A handle can be closed by calling the CloseHandle function; the system may delete the object if the count of processes using the object drops to 0
- Windows provides three ways to share objects between processes.
  - A child process inherits a handle to the object
  - One process gives the object a name when it is created and the second process opens that name
  - DuplicateHandle function:
    - Given a handle to process and the handle's value a second process can get a handle to the same object, and thus share it



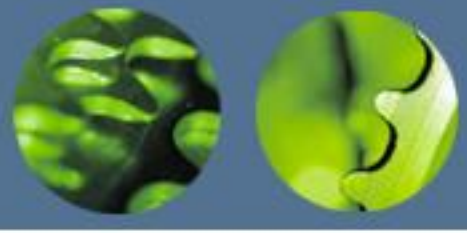
Programmer Interface — Process Management

- Process is started via the `CreateProcess` routine which loads any dynamic link libraries that are used by the process, and creates a *primary thread*
- Additional threads can be created by the `CreateThread` function
- Every dynamic link library or executable file that is loaded into the address space of a process is identified by an *instance handle*



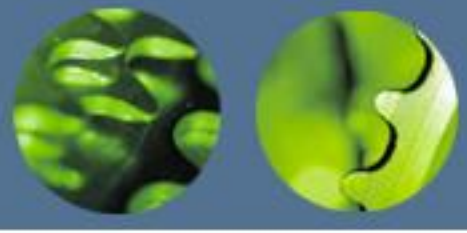
Process Management (Cont.)

- Scheduling in Win32 utilizes four priority classes:
  - IDLE_PRIORITY_CLASS (priority level 4)
  - NORMAL_PRIORITY_CLASS (level8 — typical for most processes
  - HIGH_PRIORITY_CLASS (level 13)
  - REALTIME_PRIORITY_CLASS (level 24)
- To provide performance levels needed for interactive programs, Windows has a special scheduling rule for processes in the NORMAL_PRIORITY_CLASS
  - Windows distinguishes between the **foreground** *process* that is currently selected on the screen, and the **background** *processes* that are not currently selected
  - When a process moves into the foreground, Windows increases the scheduling quantum by some factor, typically 3



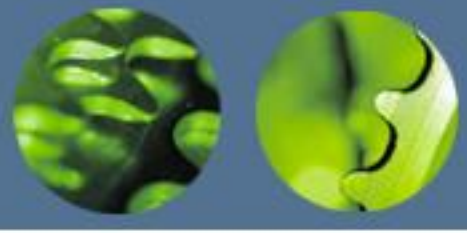
Process Management (Cont.)

- The kernel dynamically adjusts the priority of a thread depending on whether it is I/O-bound or CPU-bound
- To synchronize the concurrent access to shared objects by threads, the kernel provides synchronization objects, such as semaphores and mutexes
  - In addition, threads can synchronize by using the `WaitForSingleObject` or `WaitForMultipleObjects` functions
  - Another method of synchronization in the Win32 API is the critical section



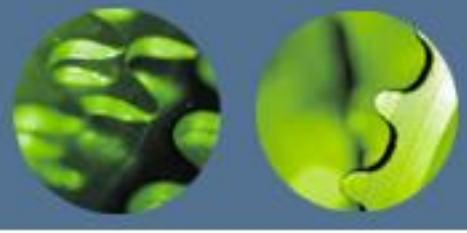
Process Management (Cont.)

- A fiber is user-mode code that gets scheduled according to a user-defined scheduling algorithm
  - Only one fiber at a time is permitted to execute, even on multiprocessor hardware
  - Windows includes fibers to facilitate the porting of legacy UNIX applications that are written for a fiber execution model



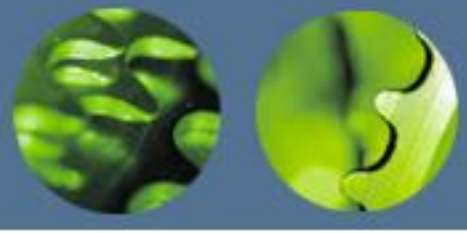
Programmer Interface — Interprocess Comm.

- Win32 applications can have interprocess communication by sharing kernel objects
- An alternate means of interprocess communications is message passing, which is particularly popular for Windows GUI applications
  - One thread sends a message to another thread or to a window
  - A thread can also send data with the message
- Every Win32 thread has its own input queue from which the thread receives messages
- This is more reliable than the shared input queue of 16-bit windows, because with separate queues, one stuck application cannot block input to the other applications
- .NET applications can use Web Services (SOAP, HTTP, etc.)



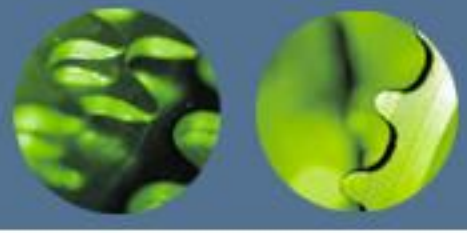
Programmer Interface — Memory Management

- Virtual memory:
  - `VirtualAlloc` reserves or commits virtual memory
  - `VirtualFree` decommits or releases the memory
  - These functions enable the application to determine the virtual address at which the memory is allocated
- An application can use memory by memory mapping a file into its address space
  - Multistage process
  - Two processes share memory by mapping the same file into their virtual memory



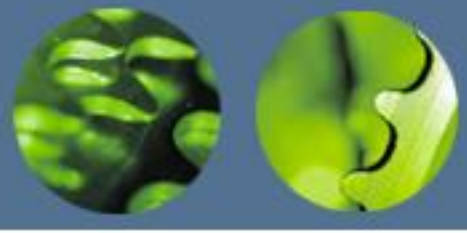
Boot Process

- The Windows boot process doesn't begin when you power on your computer or press the reset button. It begins when you install Windows on your computer. At some point during the execution of the Windows Setup program, the system's primary hard disk is prepared with code that takes part in the boot process.
- Microsoft operating systems split hard disks into discrete areas known as partitions and use file systems (such as FAT and NTFS) to format each partition into a volume. A hard disk can contain up to four primary partitions. Because this apportioning scheme would limit a disk to four volumes, a special partition type, called an extended partition, further allocates up to four add
- The early phases of the boot process differ significantly on x86 and x64 systems with a BIOS (basic input output system) versus systems with an EFI (Extensible Firmware Interface). EFI is a newer standard that does away with much of the legacy 16-bit code that BIOS systems use and allows the loading of preboot programs and drivers to support the operating system loading phase.itional partitions within each extended partition.



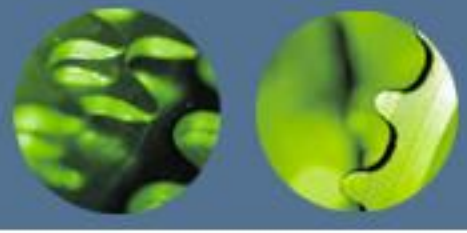
Bios Boot Process Components

Component	Processor Execution	Responsibilities	Location
Master Boot Record (MBR)	16-bit real mode	Reads and loads partition boot sectors	Per storage device
Boot sector	16-bit real mode	Reads the root directory to load Bootmgr.	Per partition
Bootmgr	16-bit real mode and 32-bit without paging	Reads the Boot Configuration Database (BCD), presents boot menu, and allows execution of preboot programs such as the Memory Test application (Memtest.exe). If a 64-bit installation is booted, switches to 64-bit long mode before loading Winload.	Per system
Winload.exe	32-bit protected mode with paging, 64-bit protected mode if booting a Win64 installation	Loads Ntoskrnl.exe and its dependencies (Bootvid.dll on 32-bit systems, Hal.dll, Kdcom.dll, Ci.dll, Clfs.sys, Pshed.dll) and boot-start device drivers.	Per Windows installation
Winresume.exe	32-bit protected mode, 64-bit protected mode if resuming a Win64 installation	If resuming after a hibernation state, resumes from the hibernation file (Hiberfil.sys) instead of typical Windows loading.	Per Windows installation



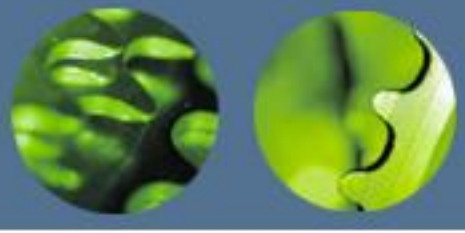
Boot Process

- Utilities that prepare hard disks for the definition of volumes, such as the Windows Setup program, write a sector of data called a Master Boot Record (MBR) to the first sector on a hard disk.
- The MBR includes a fixed amount of space that contains executable instructions (called boot code) and a table (called a partition table) with four entries that define the locations of the primary partitions on the disk.
- When a BIOS-based computer boots, the first code it executes is called the BIOS, which is encoded into the computer's ROM. The BIOS selects a boot device, reads that device's MBR into memory, and transfers control to the code in the MBR.



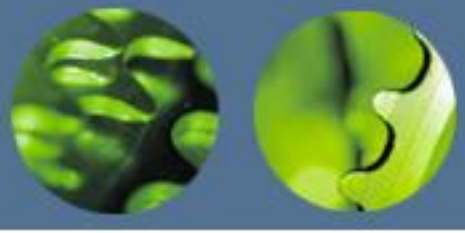
Boot Process

- The MBRs written by Microsoft partitioning tools, such as the one integrated into Windows Setup and the Disk Management MMC snap-in, go through a similar process of reading and transferring control.
- First, an MBR's code scans the primary partition table until it locates a partition containing a flag that signals the partition is bootable. When the MBR finds at least one such flag, it reads the first sector from the flagged partition into memory and transfers control to code within the partition. This type of partition is called a system partition, and the first sector of such a partition is called a boot sector. The volume defined for this partition is called the system volume.
- Operating systems generally write boot sectors to disk without a user's involvement. For example, when Windows Setup writes the MBR to a hard disk, it also writes the file system boot code (part of the boot sector) to the first bootable partition of the disk.
- Before writing to a partition's boot sector, Windows Setup ensures that the boot partition is formatted with NTFS, the only supported file system that Windows can boot from, or formats the boot partition (and any other partition) with NTFS.



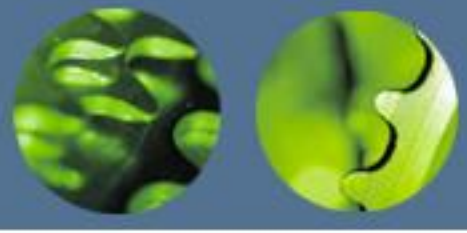
Boot Process

- Another of Setup's roles is to prepare the Boot Configuration Database (BCD), which on BIOS systems is stored in the \Boot\BCD file on the root directory of the system volume.
- This file contains options for starting the version of Windows that Setup installs and any preexisting Windows installations.
- If the BCD already exists, the Setup program simply adds new entries relevant to the new installation.



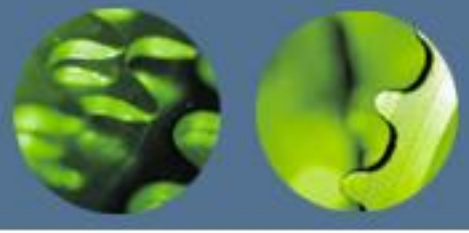
Boot Process

- Another of Setup's roles is to prepare the Boot Configuration Database (BCD), which on BIOS systems is stored in the \Boot\BCD file on the root directory of the system volume. This file contains options for starting the version of Windows that Setup installs and any preexisting Windows installations. If the BCD already exists, the Setup program simply adds new entries relevant to the new installation.
- The role of the boot-sector code is to give Windows information about the structure and format of a volume and to read in the Bootmgr file from the root directory of the volume. Thus, the boot-sector code contains just enough read-only file system code to accomplish this task.



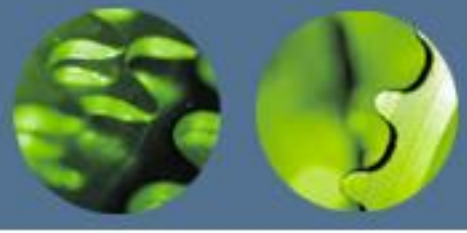
Ready Boot

- Windows uses the standard logical boot-time prefetcher if the system has less than 512 MB of memory, but if the system has 700 MB or more of RAM, it uses an in-RAM cache to optimize the boot process.
- After every boot, the ReadyBoost service uses idle CPU time to calculate a boot-time caching plan for the next boot. It analyzes file trace information from the five previous boots and identifies which files were accessed and where they are located on disk. It stores the processed traces in %SystemRoot%\Prefetch\Readyboot as .fx files and saves the caching plan under HKLM\SYSTEM\CurrentControlSet\Services\Ecache \Parameters in REG_BINARY values named for internal disk volumes they refer to.



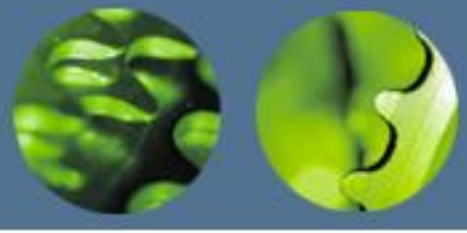
Images That Start Automatically

- In addition to the Userinit and Shell registry values in Winlogon's key, there are many other registry locations and directories that default system components check and process for automatic process startup during the boot and logon processes.
- The Msconfig utility (Windows\System32\Msconfig.exe) displays the images configured by several of the locations.



Troubleshooting Boot and Startup Problems

- Last known good (LKG) is a useful mechanism for getting a system that crashes during the boot process back to a bootable state. Because the system's configuration settings are stored in HKLM\SYSTEM\CurrentControlSet\Control and driver and service configuration is stored in HKLM\SYSTEM\CurrentControlSet\Services, changes to these parts of the registry can render a system unbootable.
- You can press the F8 key during the boot and select last known good from the resulting menu. The system marks the control set that it was using to boot the system as failed by setting the Failed value of HKLM\SYSTEM\Select and then changes HKLM\SYSTEM\Select\Current to the value stored in HKLM\SYSTEM\Select\LastKnownGood. It also updates the symbolic link HKLM\SYSTEM\CurrentControlSet to point at the LastKnownGood control set.

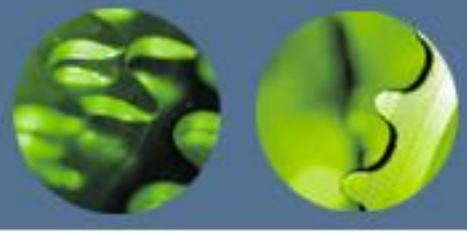


Troubleshooting Boot and Startup Problems

- Safe mode is a boot configuration that consists of the minimal set of device drivers and services. By relying on only the drivers and services that are necessary for booting, Windows avoids loading thirdparty and other nonessential drivers that might crash.
- When Windows boots, you press the F8 key to enter a special boot menu that contains the safe-mode boot options. You typically choose from three safe-mode variations: Safe Mode, Safe Mode With Networking, and Safe Mode With Command Prompt. Standard safe mode includes the minimum number of device drivers and services necessary to boot successfully.

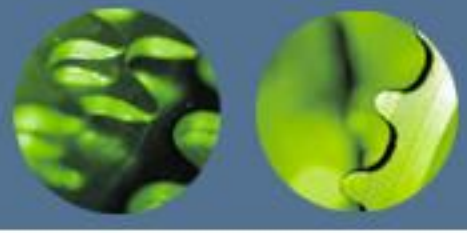
Windows Recovery Environment (WinRE)

- The Windows Recovery Environment provides an assortment of tools and automated repair technologies to automatically fix the most common startup problems. It includes five main tools:
  - ■ Startup Repair An automated tool that detects the most common Windows startup problems and automatically attempts to repair them.
  - ■ System Restore Allows restoring to a previous restore point in cases in which you cannot boot the Windows installation to do so, even in safe mode.
  - ■ Complete PC Restore Called ASR (Automated System Recovery) in previous versions of Windows, this restores a Windows installation from a complete backup, not just a system restore point, which may not contain all damaged files and lost data.
  - ■ Windows Memory Diagnostic Tool Performs memory diagnostic tests that check for signs of faulty RAM. Faulty RAM can be the reason for random kernel and application crashes and erratic system behavior.
  - ■ Command Prompt For cases where troubleshooting or repair requires manual intervention



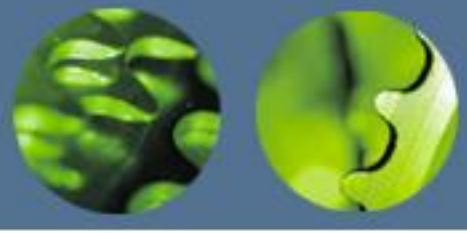
MBR Corruption

- Symptoms A system that has Master Boot Record (MBR) corruption will execute the BIOS power-on self test (POST), display BIOS version information or OEM branding, switch to a black screen, and then hang. Depending on the type of corruption the MBR has experienced, you might see one of the following messages: “Invalid partition table,” “Error loading operating system,” or “Missing operating system.”
  - Cause The MBR can become corrupt because of hard-disk errors, disk corruption as a result of a driver bug while Windows is running, or intentional scrambling as a result of a virus.
  - Resolution Boot into the Windows Recovery Environment, choose the Command Prompt option, and then execute the bootrec /fixmbr command. This command replaces the executable code in the MBR.



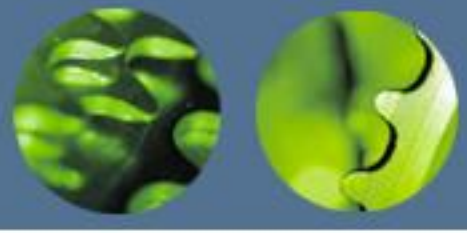
Boot Sector Corruption

- Symptoms Boot sector corruption can look like MBR corruption, where the system hangs after BIOS POST at a black screen, or you might see the messages “A disk read error occurred,” “BOOTMGR is missing,” or “ BOOTMGR is compressed” displayed on a black screen.
  - Cause The boot sector can become corrupt because of hard-disk errors, disk corruption as a result of a driver bug while Windows is running, or intentional scrambling as a result of a virus.
  - Resolution Boot into the Windows Recovery Environment, choose the Command Prompt option, and then execute the bootrec /fixboot command. This command rewrites the boot sector of the volume that you specify. You should execute the command on both the system and boot volumes if they are different.



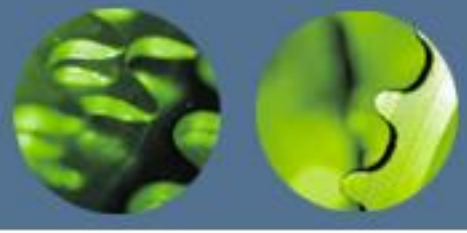
Windows resource Protection

- To preserve the integrity of the many components involved in the boot process, as well as other critical Windows files, libraries, and applications, Windows implements a technology called Windows Resource Protection (WRP). WRP is implemented through access control lists (ACLs) that protect critical system files on the machine.
- WRP will also protect entire critical folders if required, even locking down the folder so that it is inaccessible by administrators (without modifying the access control list on the folder). The only supported way to modify WRP-protected files is through the Windows Modules Installer service, which can run under the TrustedInstaller account. This service is used for the installation of patches, service packs, hotfixes, and Windows Update.



System Hive Corruption

- Symptoms If the System registry hive is missing or corrupted, Winload will display the message “Windows could not start because the following file is missing or corrupt: \WINDOWS\SYSTEM32\CONFIG\SYSTEM,” on a black screen after the BIOS POST.
  - Causes The System registry hive, which contains configuration information necessary for the system to boot, has become corrupt or has been deleted.
  - Resolution Boot into the Windows Recovery Environment, choose the Command Prompt option, and then execute the chkdsk command. If the problem is not corrected, obtain a backup of the System registry hive. Windows makes copies of the registry hives every 12 hours (keeping the immediately previous copy with a .OLD extension) in a folder called \Windows\System32\Config\RegBack, so copy the file named System to \Windows\System32\Config.
- If System Restore is enabled (System Restore is discussed in Chapter 11), you can often obtain a more recent backup of the registry hives, including the System hive, from the most recent restore point.



Shutdown

- If someone is logged on and a process initiates a shutdown by calling the Windows Exit-WindowsEx function, a message is sent to that session's Csrss instructing it to perform the shutdown. Csrss in turn impersonates the caller and sends an RPC message to Winlogon, telling it to perform a system shutdown. Winlogon then impersonates the currently logged-on user (who might or might not have the same security context as the user who initiated the system shutdown) and calls ExitWindowsEx with some special internal flags. Again, this call causes a message to be sent to the Csrss process inside that session, requesting a system shutdown.
- If the thread doesn't exit before the timeout, Csrss fades out the screen and displays the hung-program screen.

Q & A

Questions?

nenad@automobili.zastava.net

stefanovic.n@gmail.com