Web & Net Design - System Analyst
Computer Engineering Teach Advice

מצגות לימודיות להוראת מיקרו-מעבדים, אסמבלר, תקשורת, מערכות הפעלה.

שאול קובל

מבוא לתקשורת מחשבים

מערכות הפעלה ** WIN

saul@coval.net
http://www.coval.net

ערב טוב !!

נא לשמור על השקט ולכבות טלפונים!! תודה

תכינו את דפי העזר וכלי כתיבה ורשמו הערות על פי הצורך

WINDOWS

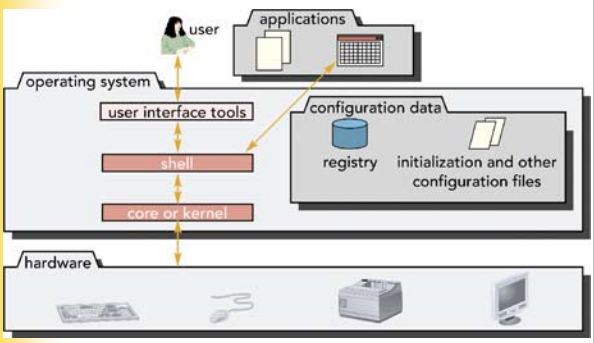
Operating System

Components of an OS

Every Operating System has Three Major Components

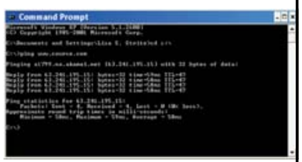
- A ***shell*** is the portion of the OS that relates to the user and to applications
- The ***kernel*** is the part that loads when you first turn on your computer and is the “brain” of the OS
- ***Configuration*** Information

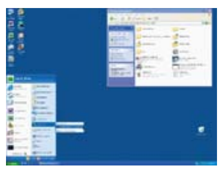
Operating System Components



An OS Provides a user interface

- With a command-line interface, you type commands or press specific keys to enter data and commands to instruct the OS to perform operations
- A menu-driven interface provides menus as a way to enter data and commands
- With a graphical user interface (GUI), you issue commands by selecting icons, buttons, windows, or other graphical objects on the screen





Multiple Users

- Most modern operating systems provide the concept of multiple users
- Each user defined on the machine is assigned an account
- This account defines the privileges that the user has – these include access to files, hardware configuration info etc.
- The highest privilege is the administrator privilege level

Booting an Operating System

- During the boot process the BIOS searches the boot device for the ***Master Boot Record*** which transfers control to the OS
- The OS takes over where the BIOS left off – for the most part at this point the hardware is initialized and the basic machine is intact

Booting Windows

- In the case of Windows NT, 2K and XP, the BIOS transfers control to a program called ***ntldr*** which loads the OS
- During the operation of the startup BIOS the x86 is in ***real*** mode
- The *ntldr* transfers the x86 CPU into ***protected*** mode and creates a file system and then initializes other systems – e.g. video

Windows Boot Window

- Windows Allows the user to force Windows to boot into Safe Mode
- In safe mode Windows operates with basic files and drivers and optionally without networking
- Several safe mode options exist:
  - Safe Mode without networking
  - Safe Mode with networking
  - Safe mode with only command line access
- Safe Mode is most useful in debugging.
- Safe Mode is entered by pressing F8 during boot

Executing a Program

- In most all operating systems the instructions for a program are stored in a disk file called an ***executable file***
- In Windows these files have the extension (***.exe***)
- When a program is to be executed the instructions in the associated executable files are read from the disk and placed into memory
- The portion of the operating system that performs this function is called the ***program loader***
- Programs stored in executable files are referred to as ***relocatable*** files

Executing a Program

- Not all instructions that makeup a program are stored in an executable file
- In most cases these other instructions are stored in ***Dynamic Link Library*** files
- Dynamic link library files have a file name extension of (***DLL***).
- When a program is loaded by the program loader, any necessary instructions that are not stored in the executable file are loaded from the appropriate DLL file
- Much of the actual operating system typically exists in dynamic link libraries

Multitasking Operating Systems

- A multitasking operating system attempts – or at least appears to - to execute multiple programs simultaneously
- A multitasking OS gives each program a slice of time in which to execute – when the program is not executing the OS *suspends* it
- The allocation of these time slices to competing programs is controlled by a part of the OS kernel called the *scheduler*
- There are many scheduling policies that exist for scheduling the execution of competing programs

Program Scheduling Policies

- Priority Based Preemptive Multitasking
  - Tasks are assigned priorities according to how important it is for that particular task to execute
  - More important programs are assigned a higher priority than less important programs
  - If multiple programs are ready to run at a particular time, the highest priority program will *always* execute – at the expense of a lower priority programs
  - This policy may cause lower priority programs to be *starved* for CPU time

Program Scheduling Policies

- Preemptive Multitasking
  - Each program is given a slice of CPU time to execute
  - Each program is given an equal amount of time to execute
  - This is the “fairest” of policies
  - This does not ensure that any program will execute for any length of time

Threads and Processes

- A Process
  - Is an instance of a program
  - Each process has its own memory space and system resources
  - A process “thinks” it has exclusive access to the machine (e.g. memory space, I/O space etc.)
- A thread
  - Is typically an execution instance spawned by a process
  - Inherits the address space and system resources that are owned by the parent process
  - Is *owned* by a process

Virtual Memory

- A computer has a finite amount of RAM available for programs
- Multitasking Operating Systems typically have multiple programs loaded in memory simultaneously
- Virtual Memory Allows each process to appear to have the entire computer address space to itself
- Controlled by a part of the kernel called the *virtual memory manager*

Virtual Memory Basics

- Virtual memory allows the computer to appear to have essentially an unlimited amount of RAM
- If the OS is low on RAM, it can store some data on the hard drive, a method called virtual memory
- Data transferred to virtual memory is stored in a file on the hard drive called a swap file or paging file

Virtual Memory Operation

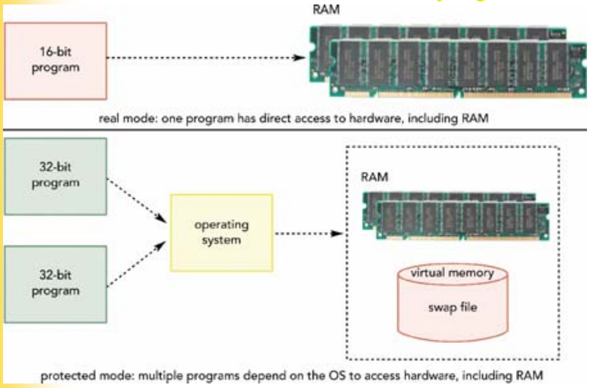
- Only the memory pages that are currently needed are maintained in physical RAM
- All other page are stored on disk
- If a memory page is referenced and that page is not currently in physical RAM, a ***page fault exception*** occurs
- The exception handler finds the referenced page in the pagefile.sys and loads it into physical RAM

Evolution of OS Memory Mgt.

- Early CPUs only had 20 lines on the bus available to handle addresses, which was 1MB of memory, divided up as follows:

Range of Memory Addresses	Type of Memory	Range Using Hexadecimal Notation
0 to 640K	Conventional or base memory	0 to A0000
604K to 1024K	Upper memory (A through F ranges)	A0000 to FFFFF
Above 1024K	Extended memory	100000 and up

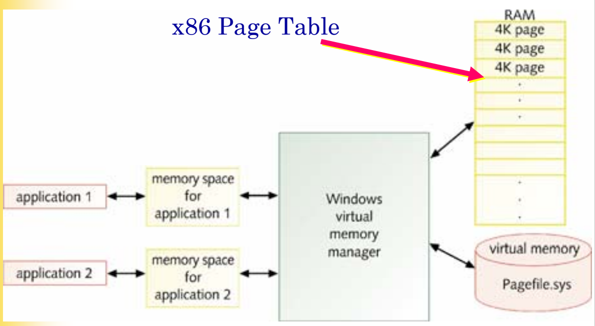
Evolution of OS Memory Mgt.



Windows Virtual Memory

- With this model, an application or device driver indicates that it needs memory, but it does not have to tell Windows which physical or which memory addresses it wants
- Memory is allocated in 4K segments or pages
- Windows assigns a certain number of pages to an application or device driver that needs memory
- Windows virtual memory manager handles managing the memory addresses used for each page and can choose to store these pages in RAM or in the hard drive in the swap file named Pagefile.sys

Windows Memory Management



The Unix Operating System

- Unix was developed initially around 1969 at the AT&T Bell Laboratories – AT&T Unix
- Later another, slightly different, version of Unix surfaced at UC Berkeley – Berkeley Unix
- In general, AT&T Unix focused on kernel functions while Berkeley Unix focused on networking capability
- Unix pioneered the X-window display system

Unix Operating Systems

- Today the largest Unix vendors include:
  - Sun Microsystems
  - Hewlet Packard
  - Silicon Graphics
- Unix is most common in large server systems
- Most of the communications backbone is hosted on Unix machines
- Real programmers prefer Unix!!

The Linux Operating System

- The initial Linux kernel was developed by Linus Torvalds while attending the University of Helsinki
- Linux is an ***Open Source*** operating system – the source code is freely available
- Linux ***is not*** Unix but rather Unix like – the commands and operation is very similar
- Various distributions of Linux exist – Red Hat is probably the most popular

Plug and Play Hardware

Why Do We Need Plug and Play ?

- Every PC contains a number of hardware components that must all work together
- Each component may need to have its own:
  - Interrupt ID
  - RAM that only this device writes to DMA
  - ROM address (where it's own BIOS resides)
  - RAM buffer address (used to pass data to/from this address)
  - PnP Identification code

Plug and Play Hardware

- The component cooperation depends on each device being correctly configured
- Each time a component is added or taken away, the configuration must be changed
- Plug and Play is a standard – agreed upon by the various hardware manufactures – that allows a computer to accomplish this automatically

How Does Plug and Play Work

- Plug and Play compatible BIOS locates all potential components that are Plug and Play compatible and collects information about each PnP device
  - Each component contains a unique identifier as part of it's on board ROM
  - Each component defines what resources it needs and what range of resources it will accept
  - BIOS transfers this configuration information to the operating system so that the appropriate drivers can be configured

How Does Plug and Play Work

- The Windows configuration manager completes the PnP installation by:
  - Assigning Resources (e.g. interrupt ID)
  - Managing conflicts
  - Locates a device driver for each component
  - Asks user for driver if it is not found
- Note that once a device is configured the information is stored within the device itself

System Requirements for PnP

- System Hardware
  - Chipset and Bus controllers must be capable of supporting PnP
- Peripheral Hardware
  - The devices being added must also be capable of supporting PnP
- System BIOS
  - The system BIOS starts the PnP process
- Operating System
  - The PnP compatible OS completes the PnP process

Device Configuration Storage

- A PnP device need only be configured once
- This configuration information is stored in a dedicated part of the non-volatile CMOS memory
- The data stored in this memory is called the ***Extended System Configuration Data***
- Both the BIOS and the OS use this data area to store and read configuration information about devices

User Configuration for PnP

• When PnP is available, the BIOS allows you to specifically designate which IRQs and DMA channels are available for PnP control

•The configuration is typically specified as:

• When possible we will make a reference to the Intel x86 architecture

System Requirements for PnP

- A PnP BIOS typically allows you to specifically designate which IRQ #'s and DMA channels are available for PnP control
- The BIOS configuration options typically are:
  - PCI/PnP – Allowed to be assigned by PnP
  - ISA/Legacy – "off limits" to PnP assignment
- Typically the following resources are PCI/PnP
  - IRQs – 3,4,5,7,9,11,12,14 and 15
  - DMA Channels – 0,1,3,5,6 and 7

Typical PnP Sequence

- See the following URL for a description of the typical steps taken by the BIOS at boot time to manage a PnP system

http://www.pcguide.com/ref/mbsys/res/pnpOperation-c.html

Typical PnP Sequence

• Create a resource table of all available PnP devices containing IRQ,DMA channel and I/O address – excluding "off limit" resources

• Search for and identify all PnP devices on both PCI and ISA bus

DOS Memory Management

- Under DOS and Windows 9.x, RAM is logically broken into the following *areas*:
  - Conventional Memory
  - Upper Memory
  - High Memory Area (HMA)
  - Extended Memory (XMS)
- The original IBM PC only could address up to 1Mbyte because the 8086 was a 16 bit processor with 20 address lines (2^{20} = 1Meg)
- The Intel 80386 and later CPUs are 32 bit and can address 4 Gigabytes

DOS Memory Management

- ***Conventional Memory*** refers to the first 640K of RAM – this is how much the original IBM PC came with
- ***Upper Memory*** refers to memory between 640K and 1Mega Byte – this area is mostly ROM addresses and other system addresses
- ***Extended Memory*** refers to all memory above 1 Mega byte
- ***High Memory Area*** refers to the first 64K of extended memory.

DOS Memory Management

- The original IBM PC included 640K of memory available to programs
- Most examples will attempt to convey concepts and will not imply any particular architecture
- When possible we will make a reference to the Intel x86 architecture

Windows NT Memory Management

- Windows NT, 2000 and XP memory management is much simpler because it is a true 32 bit operating system
  - The entire address space can be utilized
  - Restrictions that existed in DOS no longer exist
  - Memory protection can be enforced
- The result of a memory protection violation results in an error known as a ***General Protection Fault***

Windows Debugging Support

Dr. Watson Program

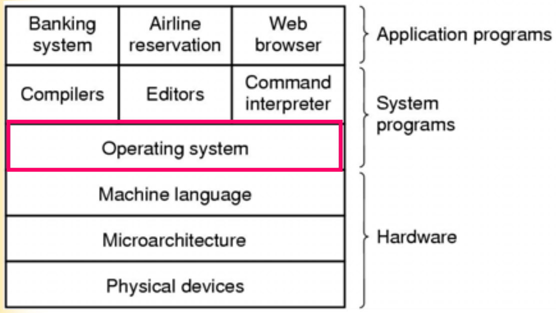
- Dr. Watson is included in the \windows\system directory
- Can be run in the background and will detect, decode and log errors
- For details, see the help menu in Windows XP
- If frequent errors occur – put Dr. Watson in startup menu

Multiple Processor Support

- Symmetric Multi-Processing Support
  - Two or more “identical” processors exist on a single system
  - All processors are considered equal in that they have access to the same system resources
  - The OS schedules threads to run on a particular processor
  - Most of the scheduling is out of the user as well as the programmers control
- When a thread is ready to run the OS schedules it on the processor according to its scheduling policy

Operating System Concepts

- Layers of a computer system



Windows XP

Module 22: Windows XP

- History
- Design Principles
- System Components
- Environmental Subsystems
- File system
- Networking
- Programmer Interface

Windows XP

- 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 four versions, Professional, Server, Advanced Server, National Server.
- In 1996, more NT server licenses were sold than UNIX licenses

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.

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 —XP 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).

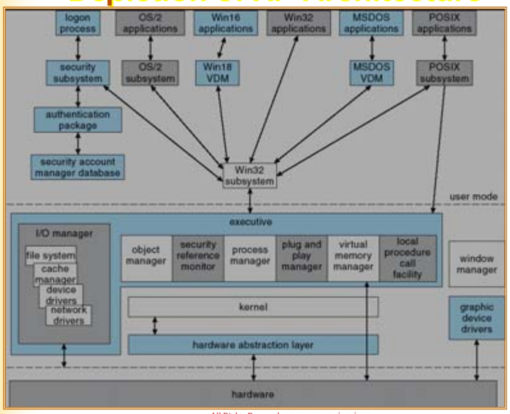
Design Principles (Cont.)

- Reliability —XP 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 XP without changing the source code.
- Performance —XP 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.

XP 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 XP 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.)

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 XP's 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 XPdoes not guarantee that a real-time thread will start to execute within any particular time limit. (This is known as *soft realtime*.)

Windows XP 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.

Executive — Object Manager

- XP 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`.

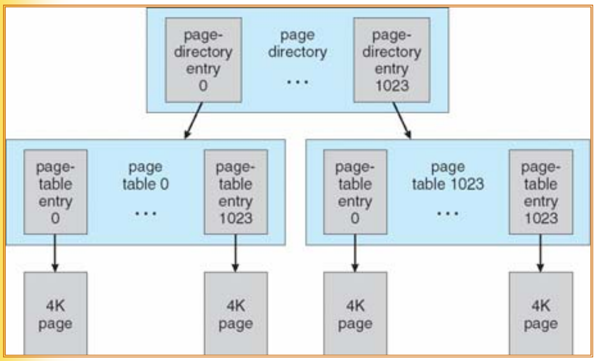
Executive — Naming Objects

- The XP 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.
- XP 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 XP uses a page-based management scheme with a page size of 4 KB.
- The XP 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 2000 paging file.

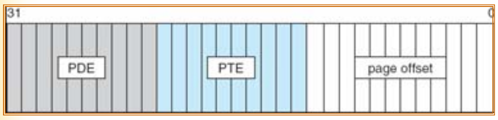
Virtual-Memory Layout



Virtual Memory Manager (Cont.)

- The virtual address translation in XP uses several data structures.
  - Each process has a *page directory* that contains 1024 *page directory entries* of size 4 bytes.
  - Each page directory entry points to a *page table* which contains 1024 *page table entries* (PTEs) of size 4 bytes.
  - Each PTE points to a 4 KB *page frame* in physical memory.
- A 10-bit integer can represent all the values form 0 to 1023, therefore, can select any entry in the page directory, or in a page table.
- This property is used when translating a virtual address pointer to a bye address in physical memory.
- A page can be in one of six states: valid, zeroed, free standby, modified and bad.

Virtual-to-Physical Address Translation



- 10 bits for page directory entry, 20 bits for page table entry, and 12 bits for byte offset in page.

Page File Page-Table Entry

5 bits for page protection, 20 bits for page frame address, 4 bits to select a paging file, and 3 bits that describe the page state. V = 0

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.

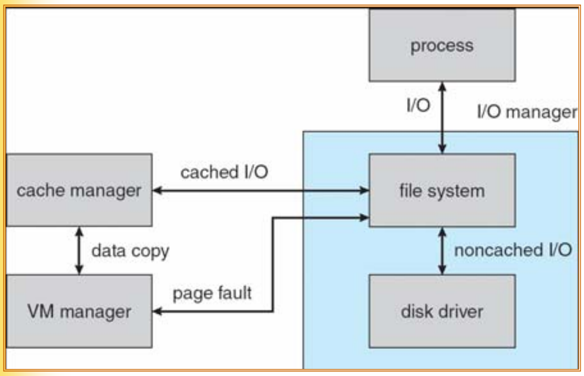
Executive — Local Procedure Call Facility

- 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 XP 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 XP 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



Executive — Security Reference Monitor

- The object-oriented nature of XP 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 XP executive services to enable XP to run programs developed for other operating system.
- XP 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 XP 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 to Windows XP 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 XP uses Kerberos as the default authentication package.

File System

- The fundamental structure of the XP file system (NTFS) is a *volume*.
  - Created by the XP 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 XP *log file service*.

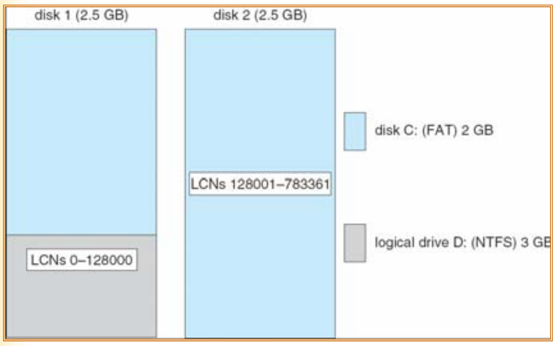
File System — Security

- Security of an NTFS volume is derived from the XP 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.

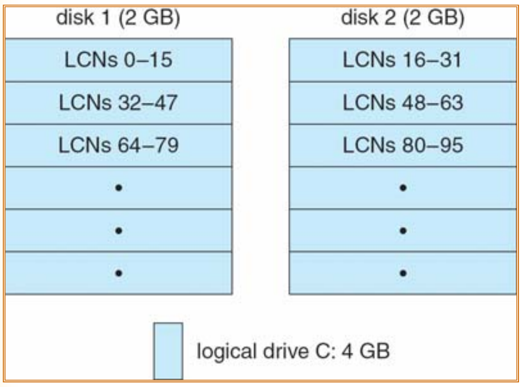
Volume Management and Fault Tolerance

- `FtDisk`, the fault tolerant disk driver for XP, 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*.

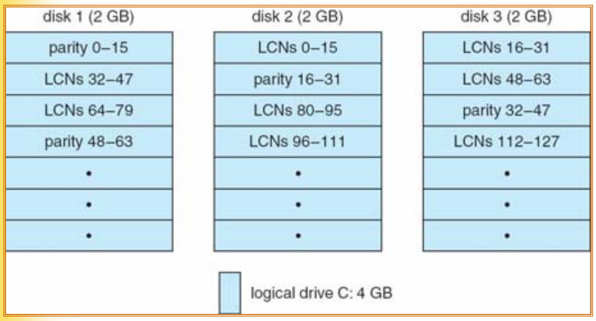
Volume Set On Two Drives



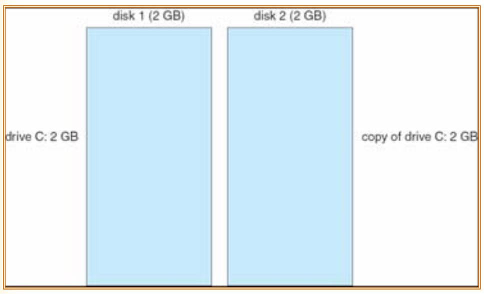
Stripe Set on Two Drives



Stripe Set With Parity on Three Drives



Mirror Set on Two Drives



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

- XP supports both peer-to-peer and client/server networking; it also has facilities for network management.
- To describe networking in XP, 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.
- XP 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 XP wants to share resources with these networks.
- XP 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 XP machines that are connected over the Internet.
- The XP 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.
- XP systems can communicate with Macintosh computers via the Apple Talk protocol if an XP Server on the network is running the Windows XP Services for Macintosh package.

Networking — Dist. Processing Mechanisms

- XP supports distributed applications via named NetBIOS, named pipes and mailslots, Windows Sockets, Remote Procedure Calls (RPC), and Network Dynamic Data Exchange (NetDDE).
- 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 XP RPC mechanism follows the widely-used Distributed Computing Environment standard for RPC messages, so programs written to use XP RPCs are very portable.
  - RPC messages are sent using NetBIOS, or Winsock on TCP/IP networks, or named pipes on LAN Manager networks.
  - XP provides the Microsoft *Interface Definition Language* to describe the remote procedure names, arguments, and results.

Networking — Redirectors and Servers

- In XP , an application can use the XP I/O API to access files from a remote computer as if they were local, provided that the remote computer is running an MS-NET server.
- A *redirector* is the client-side object that forwards I/O requests to remote files, where they are satisfied by a server.
- For performance and security, the redirectors and servers run in kernel mode.

Access to a Remote File

- The application calls the I/O manager to request that a file be opened (we assume that the file name is in the standard UNC format).
- The I/O manager builds an I/O request packet.
- The I/O manager recognizes that the access is for a remote file, and calls a driver called a Multiple Universal Naming Convention Provider (MUP).
- The MUP sends the I/O request packet asynchronously to all registered redirectors.
- A redirector that can satisfy the request responds to the MUP.
  - To avoid asking all the redirectors the same question in the future, the MUP uses a cache to remember with redirector can handle this file.

Access to a Remote File (Cont.)

- The redirector sends the network request to the remote system.
- The remote system network drivers receive the request and pass it to the server driver.
- The server driver hands the request to the proper local file system driver.
- The proper device driver is called to access the data.
- The results are returned to the server driver, which sends the data back to the requesting redirector.

Networking — Domains

- NT 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.
- XP 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`

- XP 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.
- XP 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, XP has a special scheduling rule for processes in the `NORMAL_PRIORITY_CLASS`.
  - XP 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, XP 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.
  - XP 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.

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.

Memory Management (Cont.)

- A heap in the Win32 environment is a region of reserved address space.
  - A Win 32 process is created with a 1 MB *default heap.*
  - Access is synchronized to protect the heap's space allocation data structures from damage by concurrent updates by multiple threads.
- Because functions that rely on global or static data typically fail to work properly in a multithreaded environment, the thread-local storage mechanism allocates global storage on a per-thread basis.
  - The mechanism provides both dynamic and static methods of creating thread-local storage.

סיכומים וחזרות לחומר הלימוד

R&D Advisers

Web & Net Design - System Analyst
Computer Engineering Teach Advice

Saul Coval Computers