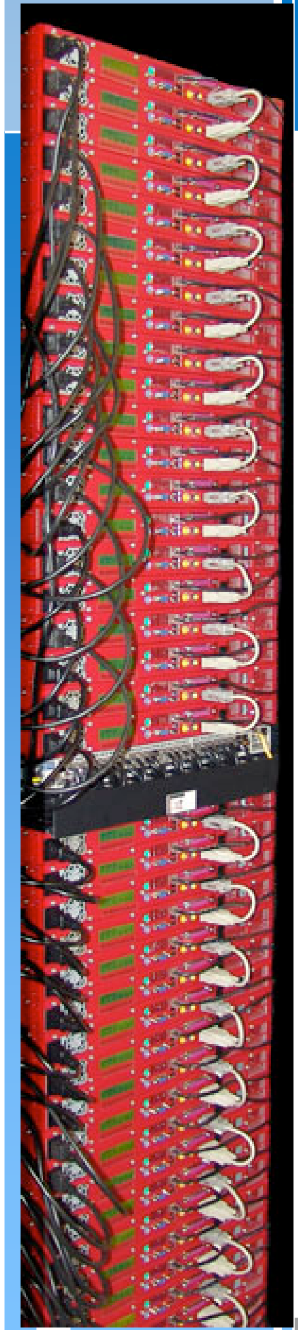


Data center server

- One of the specification is the size that server will take in a rack. 1U is the smallest size and blade servers, which fit one unit are all the rage
- Dual (Quad Core Xeon, 2x4MB Cache, 2.66 GHz, 1333 MHz FSB), 16 GB memory, 2x73GB 15k rpm hard disk - $10000
- 1 rack - 60 racks
  - ($ 0.6 m)

Servers

- Mission critical systems
  - Three tier systems - production, backup and test
  - Virtual hosting to protect against interference with other processes
  - Data center support service level agreements (SLA) - OS should be aware of these
  - On demand computing
  - Autonomic management
- Each rack can consume 10 Kw
  - Additional 10 Kw in cooling
  - Data center can be powered exclusively by a 300 MW power station.

Hot topics

- Hot research areas:
  - Energy management for servers/laptops
  - Virtual machine support for isolation (Java, Xen, VMWare, Parallels, Wine etc.)
  - Grid/cluster computing to harness lots of machines
  - Autonomic OS/storage etc.

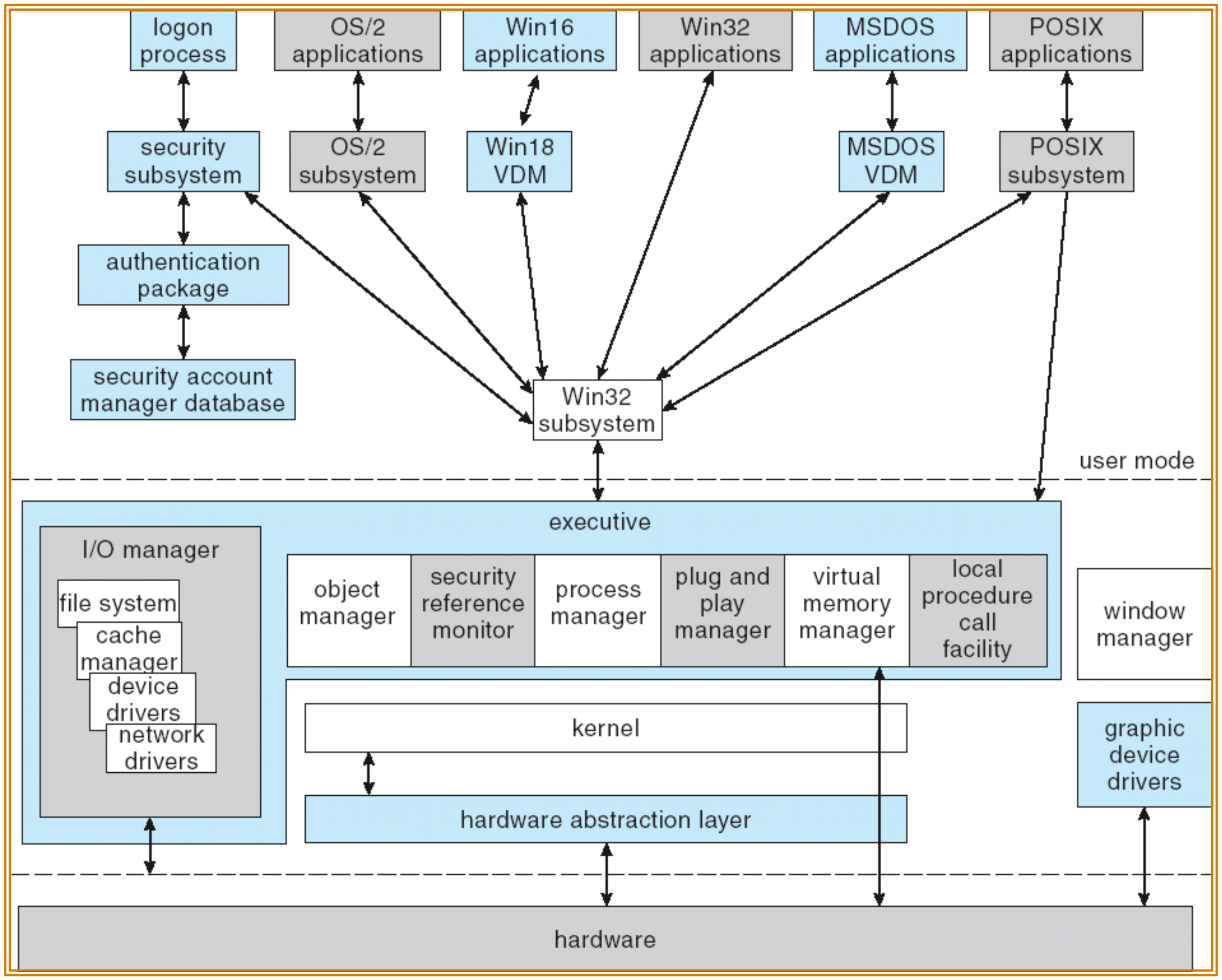
Windows XP

- 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 many variations (Pro, Home, Media Center, X64, ..)

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)

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 in 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
  - Tends 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
  - Compute-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 XP does not guarantee that a real-time thread will start to execute within any particular time limit
  - This is known as *soft real-time*

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`

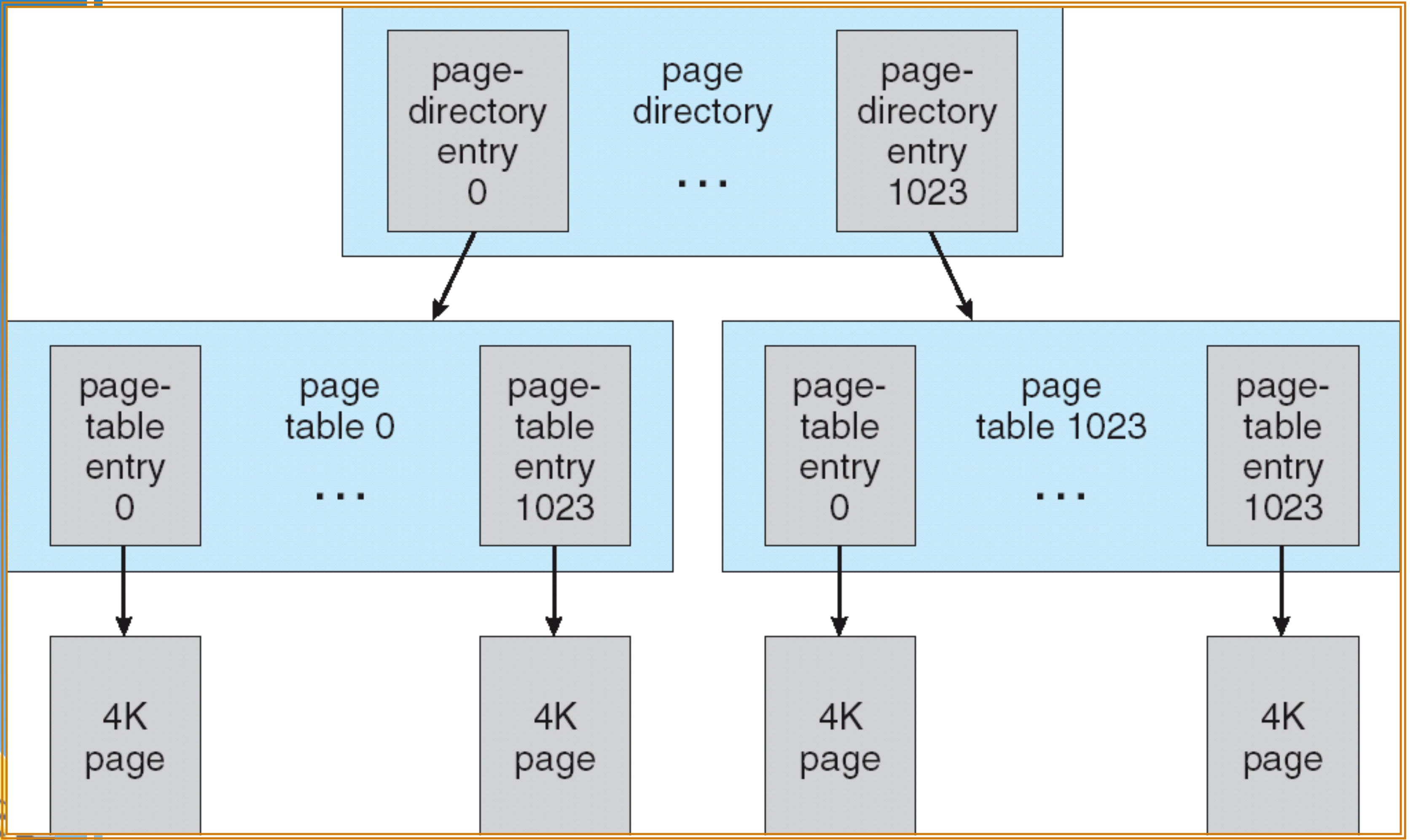
Executive — Naming Objects

- 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

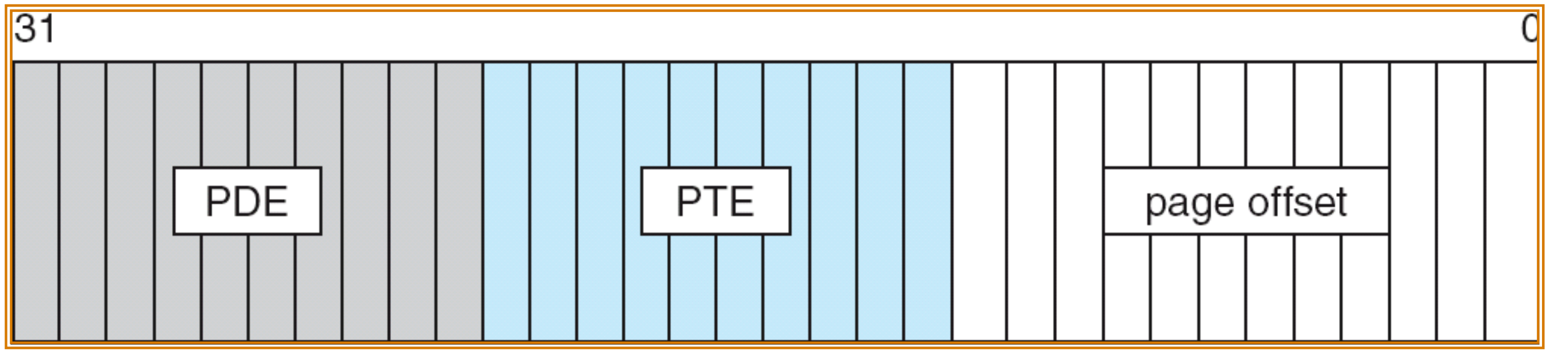


Virtual Memory Manager (Cont.)

- 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

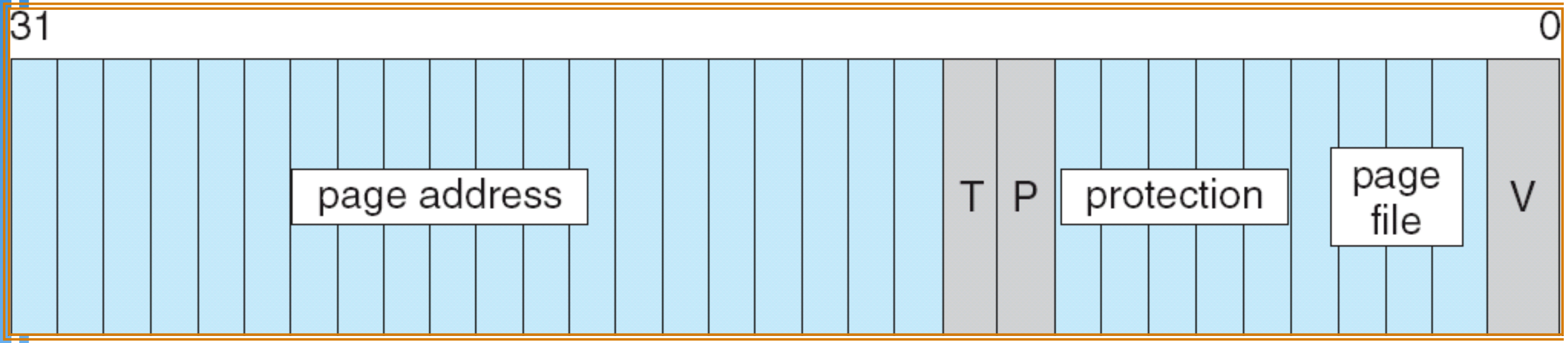
Virtual-to-Physical Address Translation

- 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.

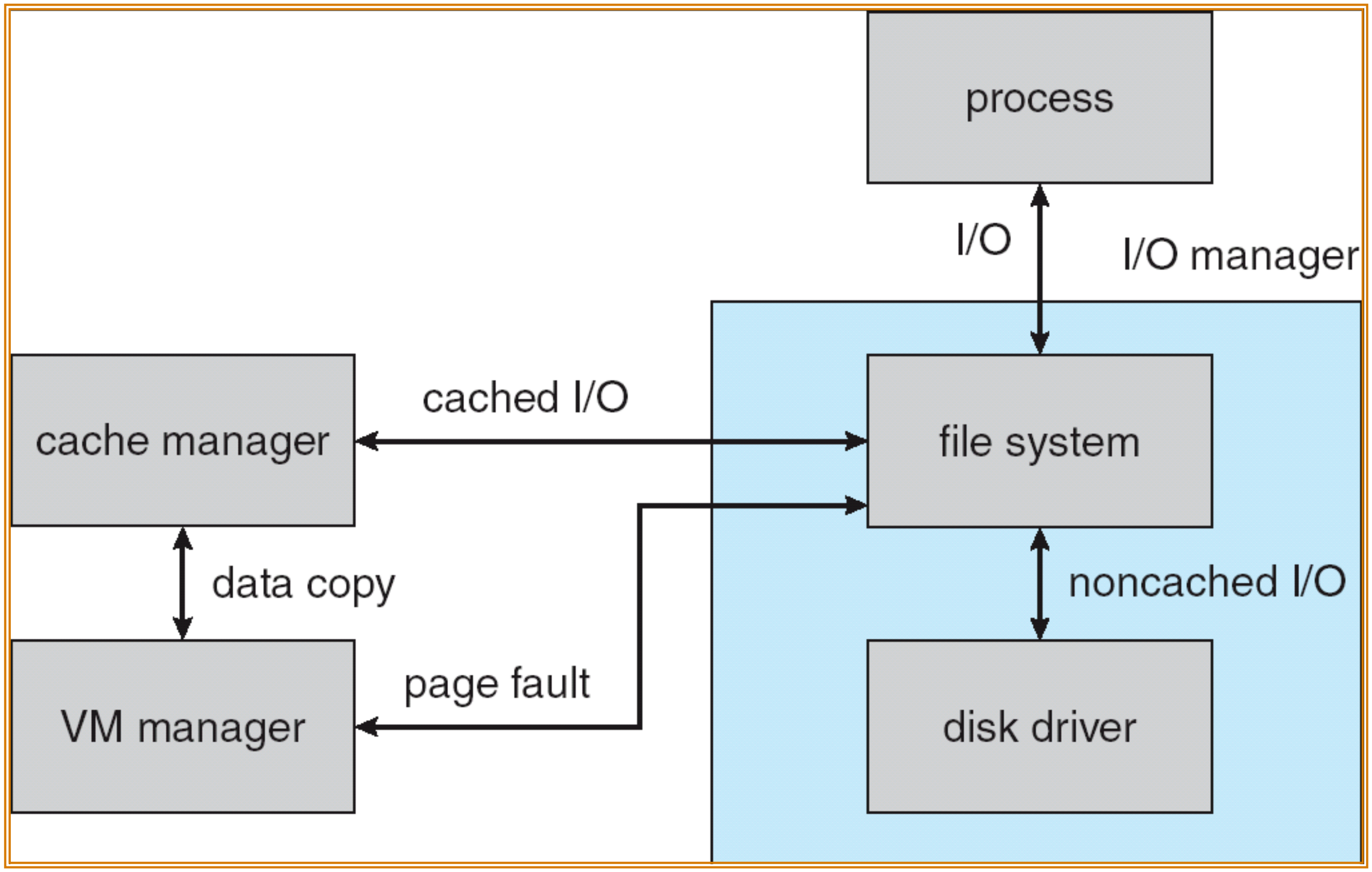
Executive — Local Procedure Call Facility

- 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


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 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 on 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

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

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

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