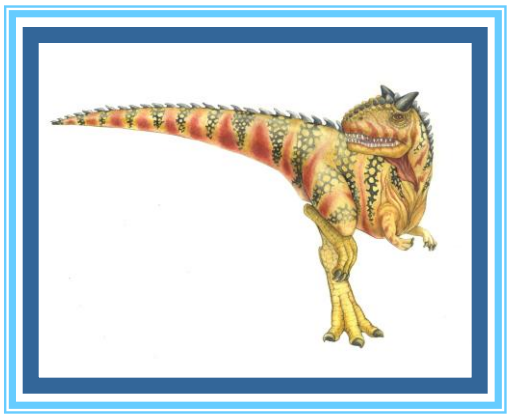
Chapter 21: Windows 10





Chapter 21: Windows

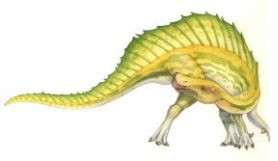
- History
- Design Principles
- System Components
- Terminal Services and Fast User Switching
- File System
- Networking
- Programmer Interface





Objectives

- Explore the principles upon which Windows 10 is designed and the specific components involved in the system
- Provide a detailed explanation of the Windows 10 file system
- Illustrate the networking protocols supported in Windows 10
- Describe the interface available in Windows 10 to system and application programmers
- Describe the important algorithms implemented in Windows 10

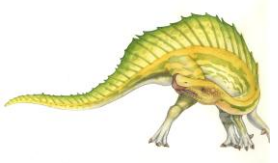


Troy University Resources



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.
- Many versions of Windows along the way, XP, Vista, 7, 8, and now 10
- Many older versions still in use, less secure, some unpatched





Windows 10

- 32- and 64-bit preemptive multitasking operating system for Intel microprocessors
- Key goals for the system:
  - portability
  - security
  - support for multiple application environments
  - multiprocessor support
  - extensibility
  - international support
  - compatibility with legacy Windows applications, subject to architecture and subsystem support
- Uses a hybrid NT kernel architecture
- Several versions with different prices, for devices/laptops/desktops and servers
- Win 10 has an app store, Windows Desktop Bridge to run older binaries, **Pico Providers** replacing the old multiple subsystem concept, **Hyper-V** virtualization, multiuser, distributed services, remote GUI, and other advanced features

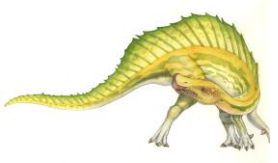


Troy University Resources



Design Principles

- Security -
  - **Access control lists** (**ACLs**) – both attribute-based and claim-based
  - Mandatory Integrity Control (MIC) uses **integrity levels and policy**
  - File system and communication encryption
  - Exploit mitigations – **address-space layout randomization** (**ASLR**), **Data Execution Prevention** (**DEP**), **Control-Flow Guard** (**CFG**), and **Arbitrary Code Guard** (**ACG**)
  - Several digital signature facilities
  - **Device Guard** option for fine grain control over what signers code is allowed on the system





Design Principles (Cont.)

- Extensibility — layered architecture
  - **Remote procedure calls** (**RPCs**)
  - **Advanced local procedure calls** (**ALPCs**)
- Portability — Windows 10 can be moved from one hardware architecture to another with relatively few changes
  - Written in C and C++
  - Processor-specific portions are written in assembly language for a given processor architecture (small amount of such code).
  - Platform-dependent code is isolated in a dynamic link library (DLL) called the “hardware abstraction layer” (HAL)

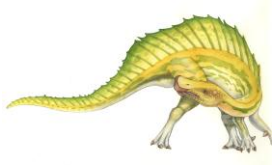


Troy University Resources



Design Principles (Cont.)

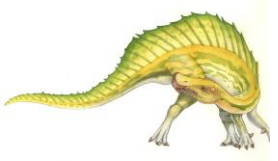
- Reliability — Windows 10 uses hardware protection for virtual memory, and software protection mechanisms for operating system resources
- Compatibility — POSIX applications may need porting or a compatible environment such as WSL to run on Windows 10
- Performance — Windows 10 subsystems can communicate 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
- Energy efficiency – for portable devices, etc, includes dynamic tick feature, process lifetime management, desktop activity monitor, connected standby





Windows 10 Architecture

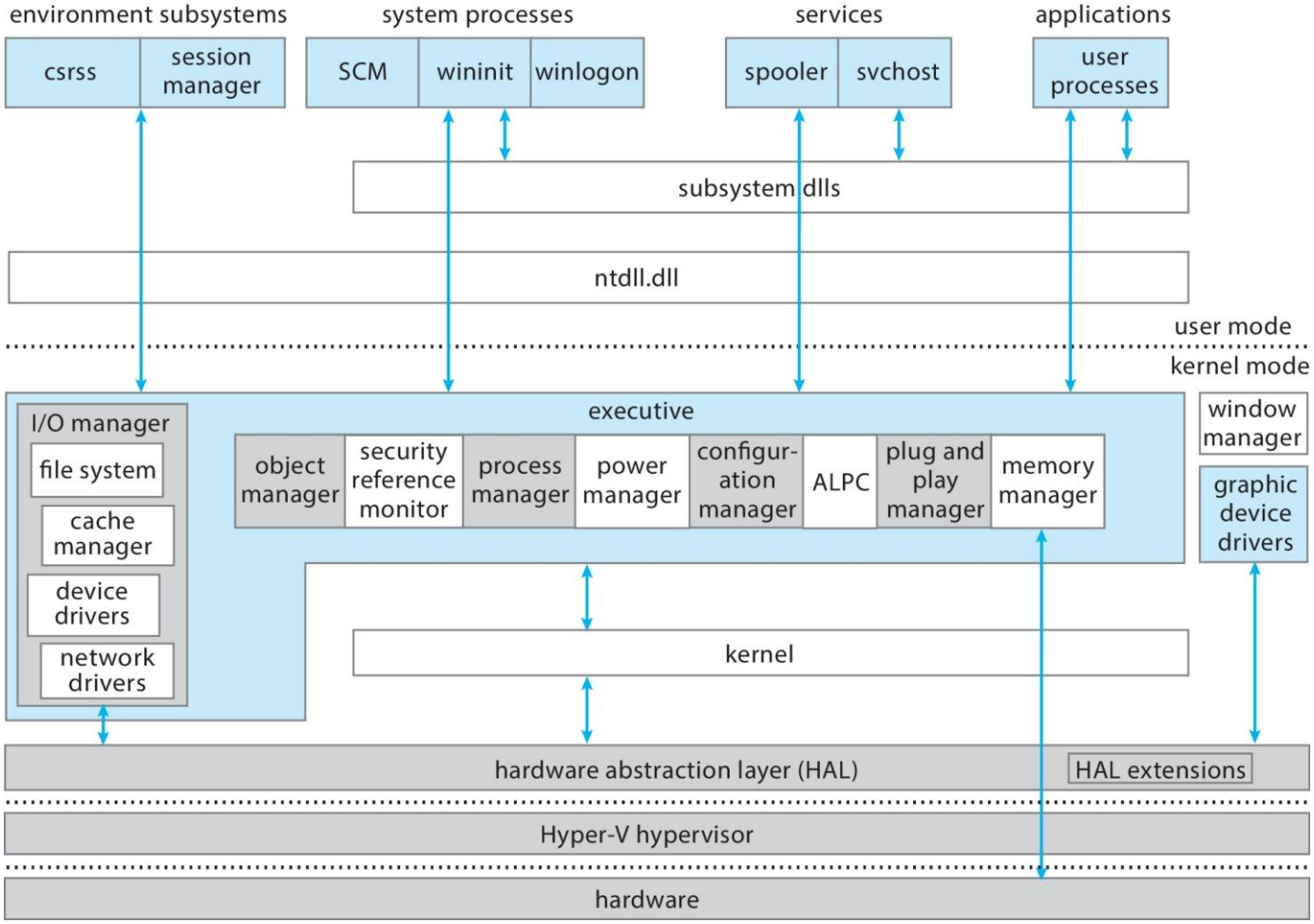
- Layered system of module
- Protected mode — **hardware abstraction layer** (**HAL**), kernel, executive
- User mode — collection of subsystems
  - Environmental subsystems emulate different operating systems
  - Protection subsystems provide security functions

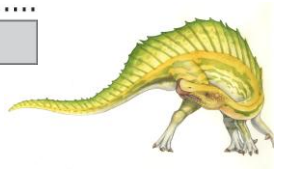


Troy University Resources



Depiction of 10 Architecture







System Components

- Layered system of modules operating at specific privilege layers
- Kernel and user mode of course
- **Virtual Trust Levels** (**VTLs**) option implemented by Hyper-V virtualization
  - Enables virtual secure mode
  - **Normal World** (VTL 0) and **Secure World** (VTL 1)
  - Within each world are user and kernel modes
  - Secure world has a secure kernel and executive and a collection of **trustlets**
  - Bottommost layer runs in special processor mode (VMX Root Mode on Intel) including Hyper-V hypervisor, creating hardware-based normal-to-secure-world boundary

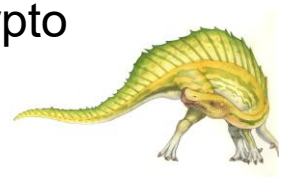


Troy University Resources



System Components — Kernel

- Foundation for the executive and the subsystems
- Core kernel code is resident. Whether execution can be preempted depends on IRQL and execution context
- 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)
- VSM Enclaves allow valid signed third-party code to perform crypto calculations





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 eight states: *initializing*, *ready, deferred-ready, standby, running, waiting, transition*, and *terminated*
- An application thread can execute in both **user mode** and **kernel mode. System threads** execute in kernel mode
  - An application thread has a user stack and a kernel stack
  - Kernel layer runs trap handler to switch stacks and change CPU mode

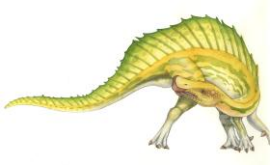


Troy University Resources



Kernel — Scheduling

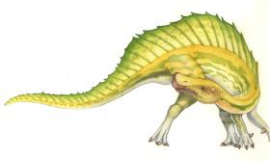
- The dispatcher uses a 32-level priority scheme to determine the order of thread execution.
  - Priorities are divided into two classes
    - The real-time class contains threads with priorities ranging from 16 to 31
    - The variable class uses priorities 1 to 15. Priority 0 is reserved for the zero-page thread
- Characteristics of Windows 10'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
  - 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 10 does not guarantee that a real-time thread will start to execute within any particular time limit
  - This is known as *soft realtime*

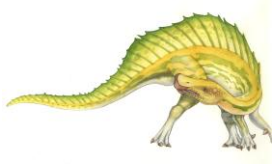


Troy University Resources



Windows x86 Interrupt Request Levels

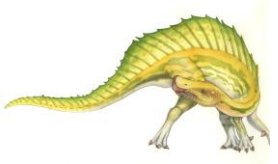
interrupt levels	types of interrupts
31	machine check or bus error
30	power fail
29	interprocessor notification (request another processor to act; e.g., dispatch a process or update the TLB)
28	clock (used to keep track of time)
27	profile
3–26	traditional PC IRQ hardware interrupts
2	dispatch and deferred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

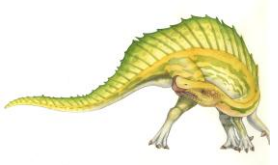


Troy University Resources



Executive — Object Manager

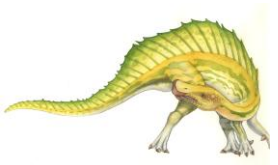
- Windows represents many managed resources as objects. The object manager supervises their use
  - Generates an object **handle**
  - Checks security
  - Keeps track of which processes are using each object
- Objects are manipulated by a standard set of methods, namely `create`, `open`, `close`, `delete`, `query name`, `parse` and `security`.





Executive — Naming Objects

- The Windows executive allows almost any object to be given a name, which may be either permanent or temporary
  - Exceptions are process, thread and some others object types
- 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
- Securable objects have security descriptors that can include access control lists

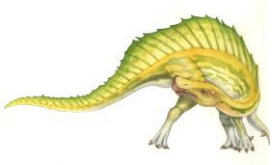


Troy University Resources



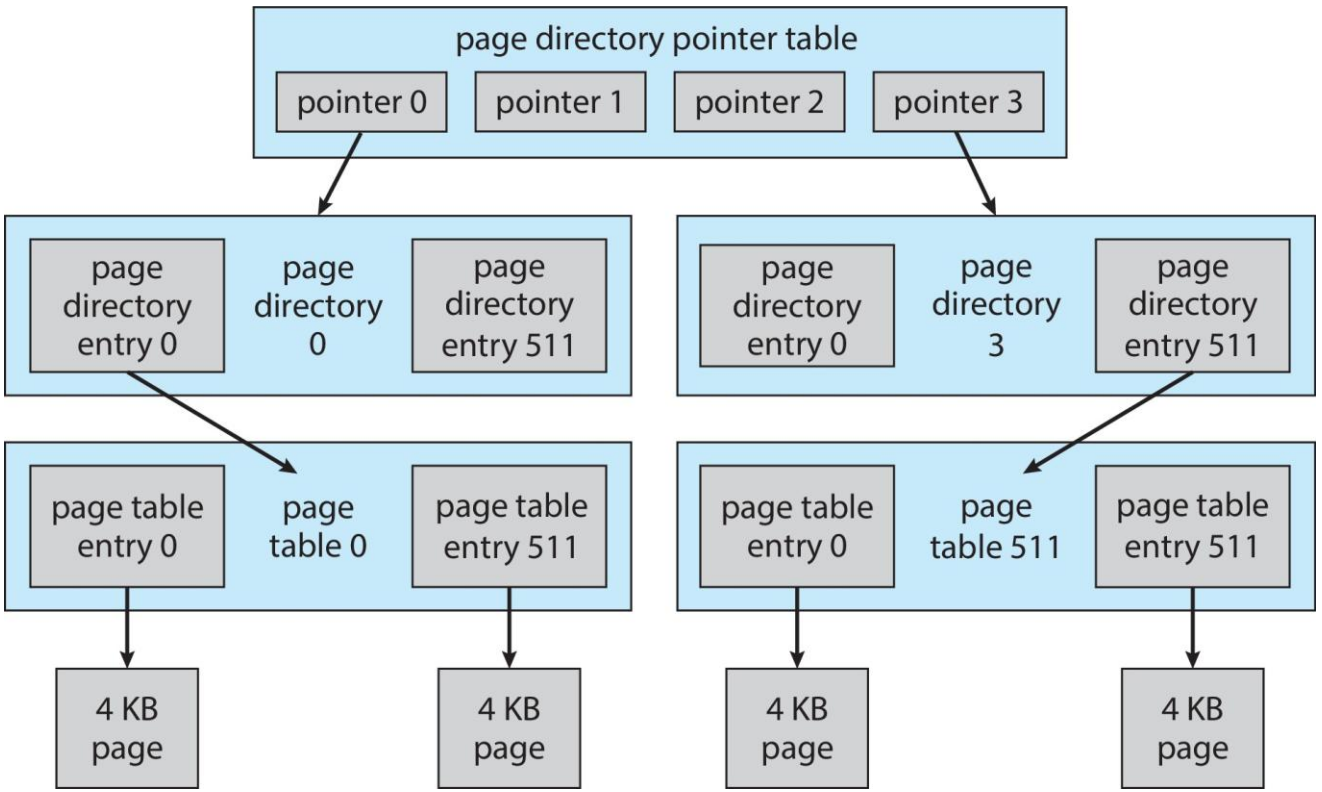
Executive — Virtual Memory Manager

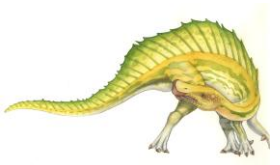
- The design of the VM manager assumes that the underlying hardware supports virtual to physical mapping a paging mechanism, transparent cache coherence on multiprocessor systems, and virtual addressing aliasing.
- The VM manager uses hardware-supported pages, typically 4 KiB. Large-page support depends on the architecture and API
- The Windows 10 VM manager uses a two step process to allocate memory
  - The first step reserves a portion of the process's address space
  - Committing charges against the system commit limit. Physical pages are supplied on demand, rather than assigning every page a page-file slot





32-bit Virtual Memory





Troy University Resources



Virtual Memory Manager (Cont.)

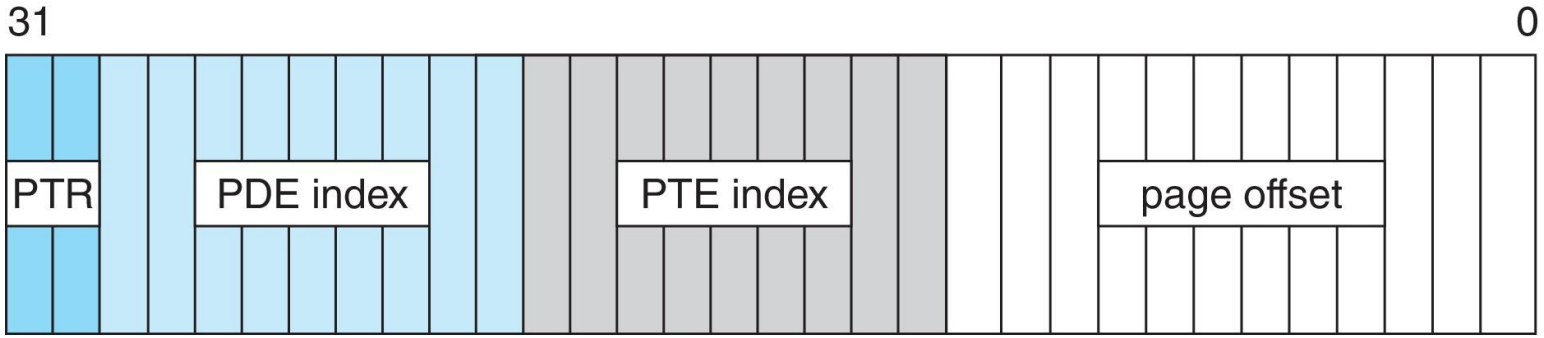
- The following is a legacy IA-32 example without PAE
  - Each process has a *page directory* that contains 1024 *page directory entries* of size 4 bytes
  - Each page directory entry points to a *page table* which contains 1024 *page table entries* (PTEs) of size 4 bytes
  - Each PTE points to a 4 KB *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
- Physical pages can be valid, zeroed, free, standby, modified, or bad

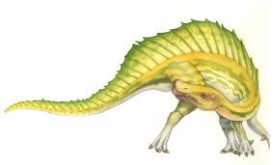




IA-32 Address Translation with PAE

- 2 bits select a page-directory-pointer-table entry, then 9 bits select a PDE, 9 bits select a PTE, and 12 bits give the page offset

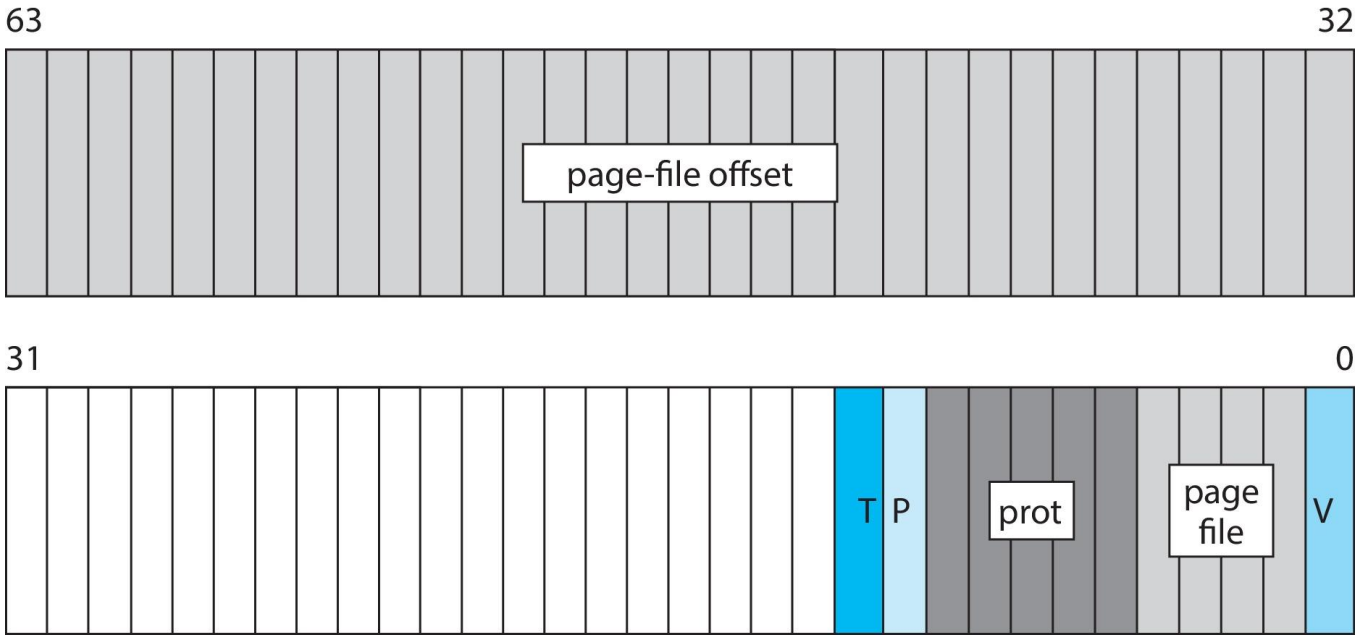




Troy University Resources



Software PTE Example



Illustrative software PTE layout, with V=0, T=0, P=0. Bit meanings vary by version and architecture

5 bits for page protection, 32 bits for page-file offset, frame address, 4 bits to select a paging file, and 20 bits for additional bookkeeping





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
- A process is represented by a process object. A job object manages a group of processes
- Windows containers use job objects and **silos as part of their isolation**

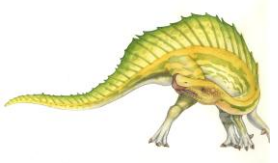


Troy University Resources



Executive — Local Procedure Call Facility

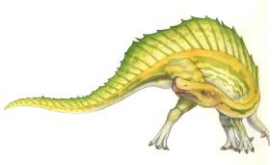
- The LPC passes requests and results between client and server processes within a single machine
- When a LPC channel is created, one of three types of message passing techniques must be specified
  - First type is suitable for small messages, up to 256 bytes; port's message queue is used as intermediate 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

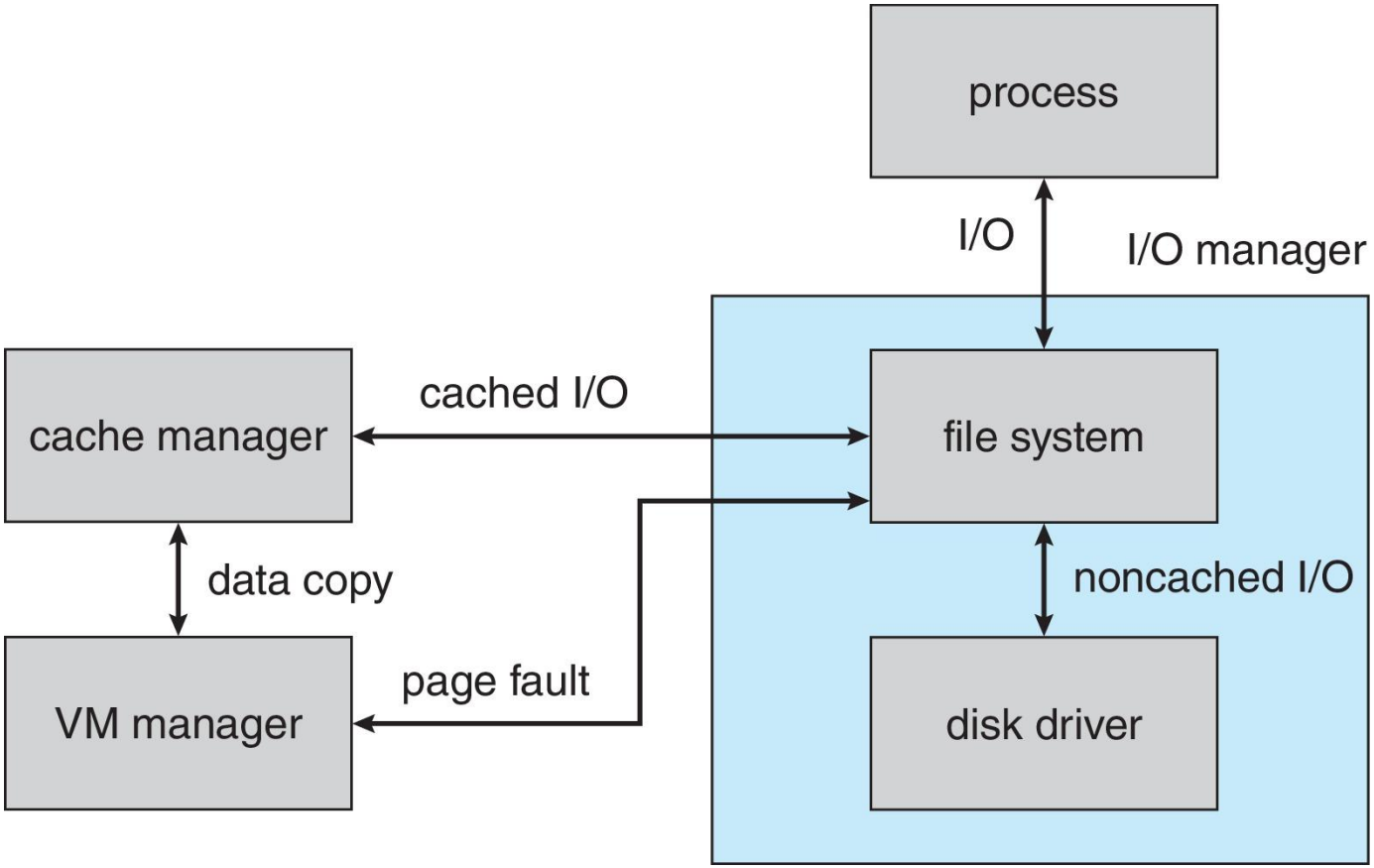
- The I/O manager is responsible for
  - file systems
  - cache management
  - device drivers
  - network drivers
- Keeps track of which installable file systems are loaded, and manages buffers for I/O requests
- Works with VM Manager to provide memory-mapped file I/O
- Controls the Windows 10 **cache 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

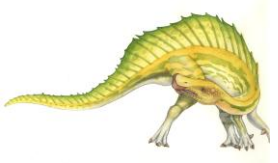


Troy University Resources



File I/O

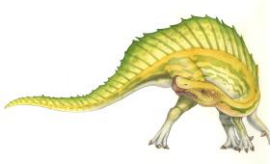

process
I/O
I/O manager
cache manager
cached I/O
file system
data copy
noncached I/O
VM manager
page fault
disk driver





Executive — Security Reference Monitor

- The object-oriented nature of Windows 10 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

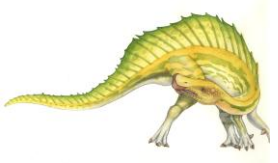


Troy University Resources



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





Executive – Power Manager

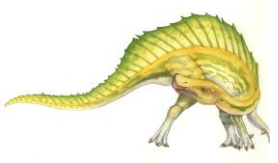
- Detects current power conditions
- Puts system to sleep, hibernation
- Processor power manager for core parking, CPU throttling and boosting, etc.
- Device state management





Executive – Registry

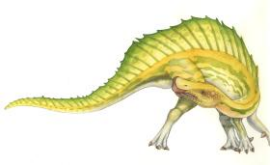
- Configuration information kept in internal repositories called **hives**
- The configuration manager manages the **registry**
- Separate hives for system information, each user's preferences, software information, security and boot options
- Registry hives can be captured in a **system restore point. A restore point is not created** before every registry change





Executive – Booting

- Windows starts booting through BIOS or UEFI firmware
- UEFI includes **Secure Boot feature** that provides integrity checking via digital signature verification of all firmware and boot-time components
- Firmware does **Power-on self-test** (**POST**)
- If machine is hibernating, state restored via `winresume.efi` program
- Kernel initializes, starts several processes including **idle**, **system process**, **secure system process** and **session manager subsystem** (**SMSS**)

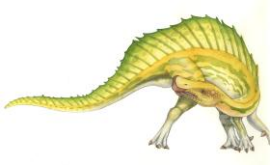


Troy University Resources



Legacy NT Subsystems

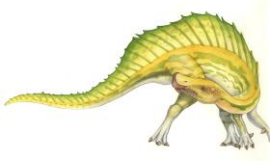
- Earlier Windows NT releases used user-mode environmental subsystems above native executive services to support different application environments
- Win32 was the main application environment in Windows 7
  - It also provides all the keyboard, mouse and graphical display capabilities.
- On 32-bit x86 Windows, NTVDM provided an MS-DOS environment. It is unavailable on 64-bit Windows





Environmental Subsystems (Cont.)

- Legacy 16-bit Windows environment, on 32-bit x86 Windows:
  - 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
- Earlier NT releases included a POSIX.1 subsystem. This legacy subsystem is not included in Windows 10

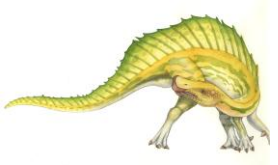


Troy University Resources



Environmental Subsystems (Cont.)

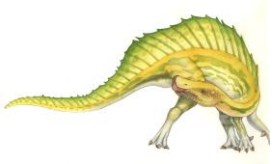
- Earlier NT releases included an OS/2 subsystem. Windows 10 does not include it
- Logon and Security Subsystems authenticates users logging on to Windows 7 systems
  - Users are required to have account names and passwords.
  - Authentication establishes identity. Object access checks then use the access token and the object's security descriptor
  - Kerberos is the default for Active Directory domain authentication, subject to the authentication context





File System

- The fundamental structure of the Windows 10 file system (NTFS) is a *volume*
  - Created by the Windows 10 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
  - Smaller clusters can reduce internal fragmentation. Cluster size depends on volume format and configuration

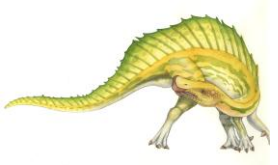


Troy University Resources



File System — Internal Layout

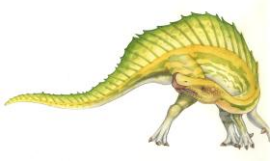
- NTFS uses logical cluster numbers (LCNs) as disk addresses
- An NTFS file consists of attributes, including one or more data streams that applications access as byte streams
- Every file in NTFS is described by one or more records in an array stored in a special file called the Master File Table (MFT)
- 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

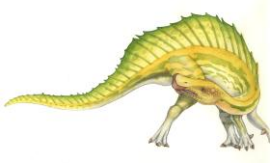


Troy University Resources



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 7 *log file service.*





File System — Security

- Security of an NTFS volume is derived from the Windows object model
- NTFS associates a security descriptor with each securable file or directory
- The descriptor includes owner and group SIDs and access-control information. Access tokens belong to security subjects, not files

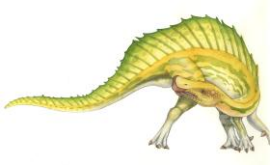


Troy University Resources



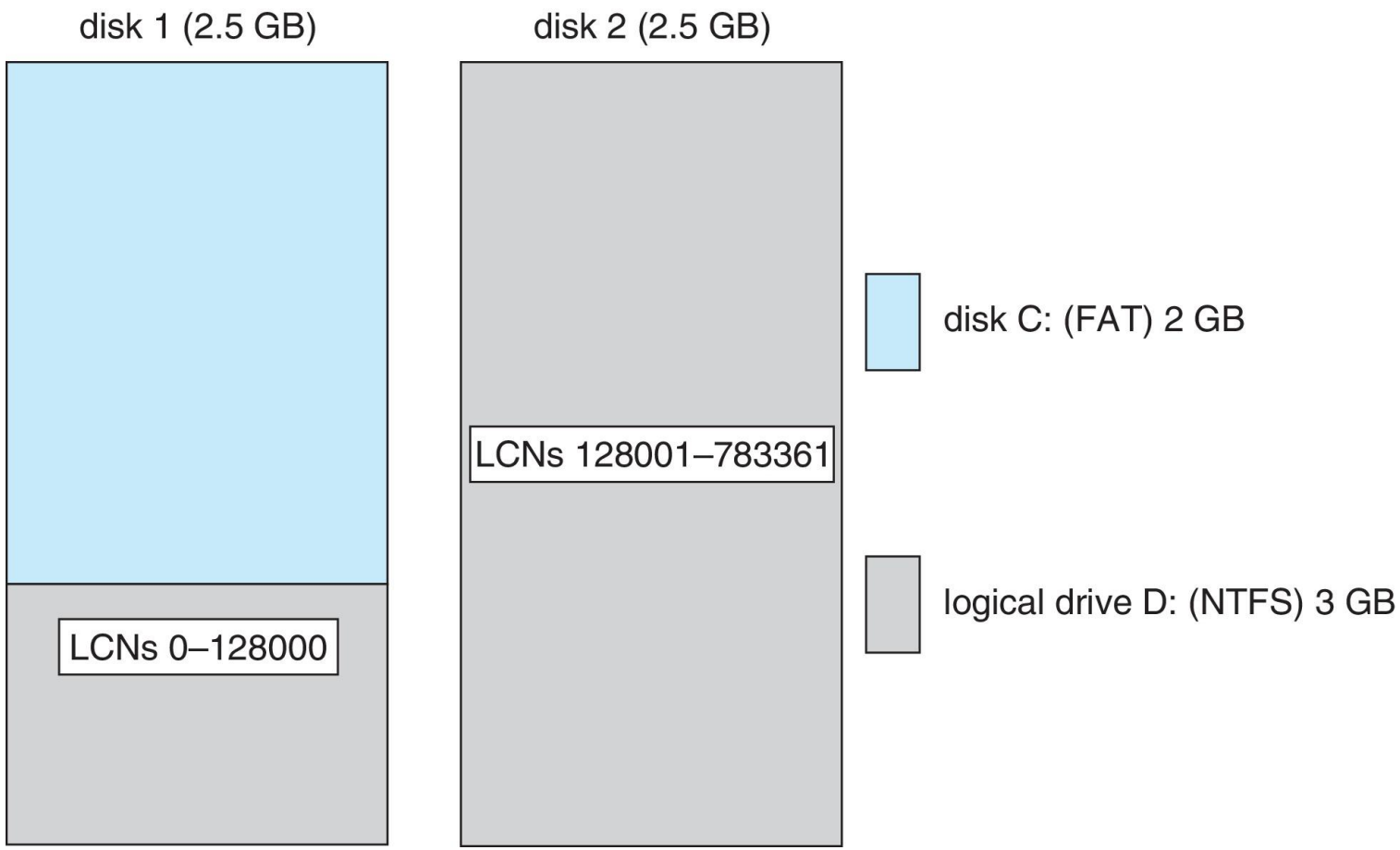
Volume Management and Fault Tolerance

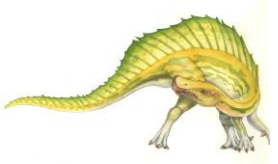
- `Historically, FtDisk` provided software volume sets and fault-tolerant disk configurations in Windows NT
- Logically concatenate multiple disks to form a large logical volume, a *volume set*
- Interleave multiple physical partitions in round-robin fashion to form a *stripe set* (also called RAID level 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
- Drive firmware can remap bad sectors. NTFS can mark bad *clusters and remap affected file data*





Volume Set On Two Drives

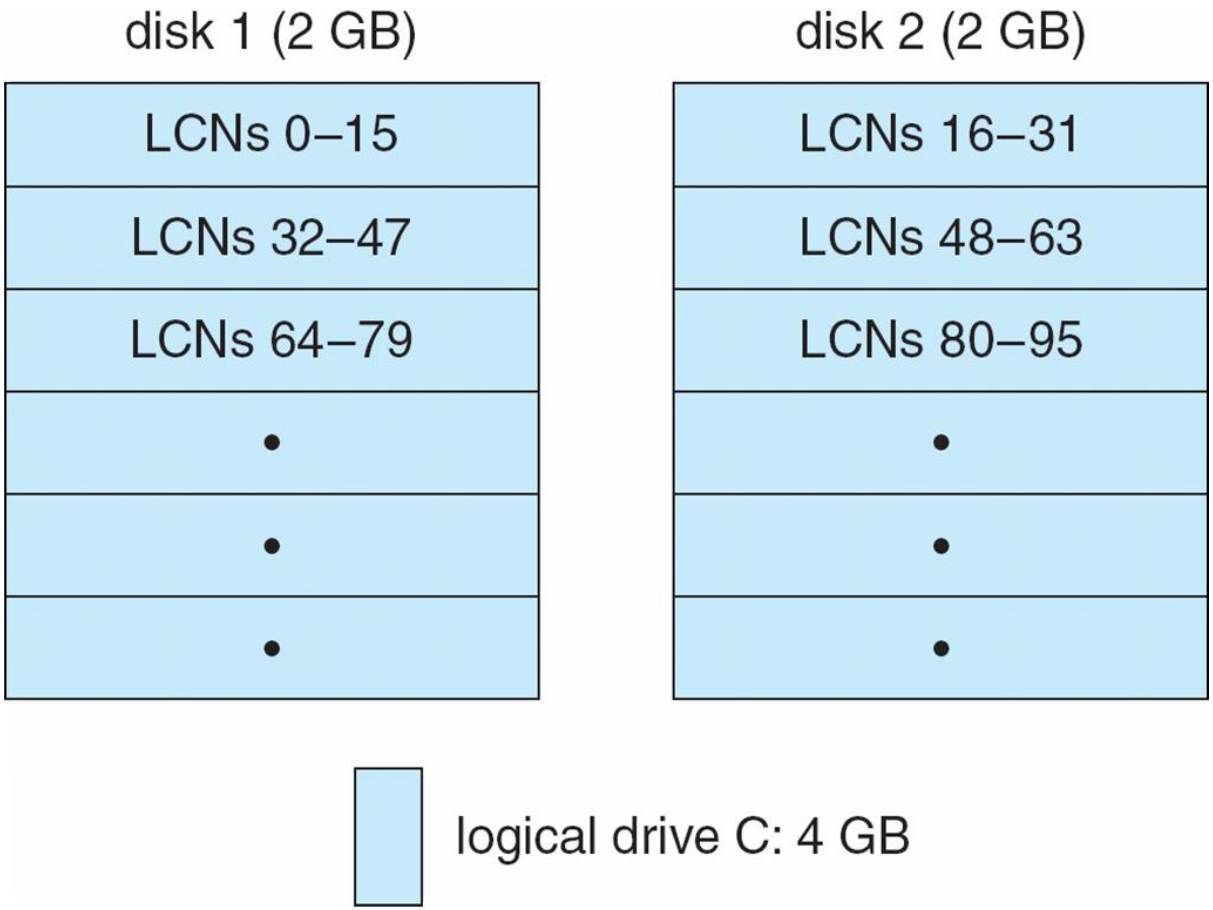


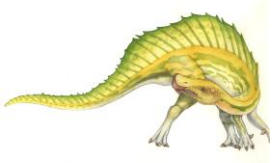


Troy University Resources



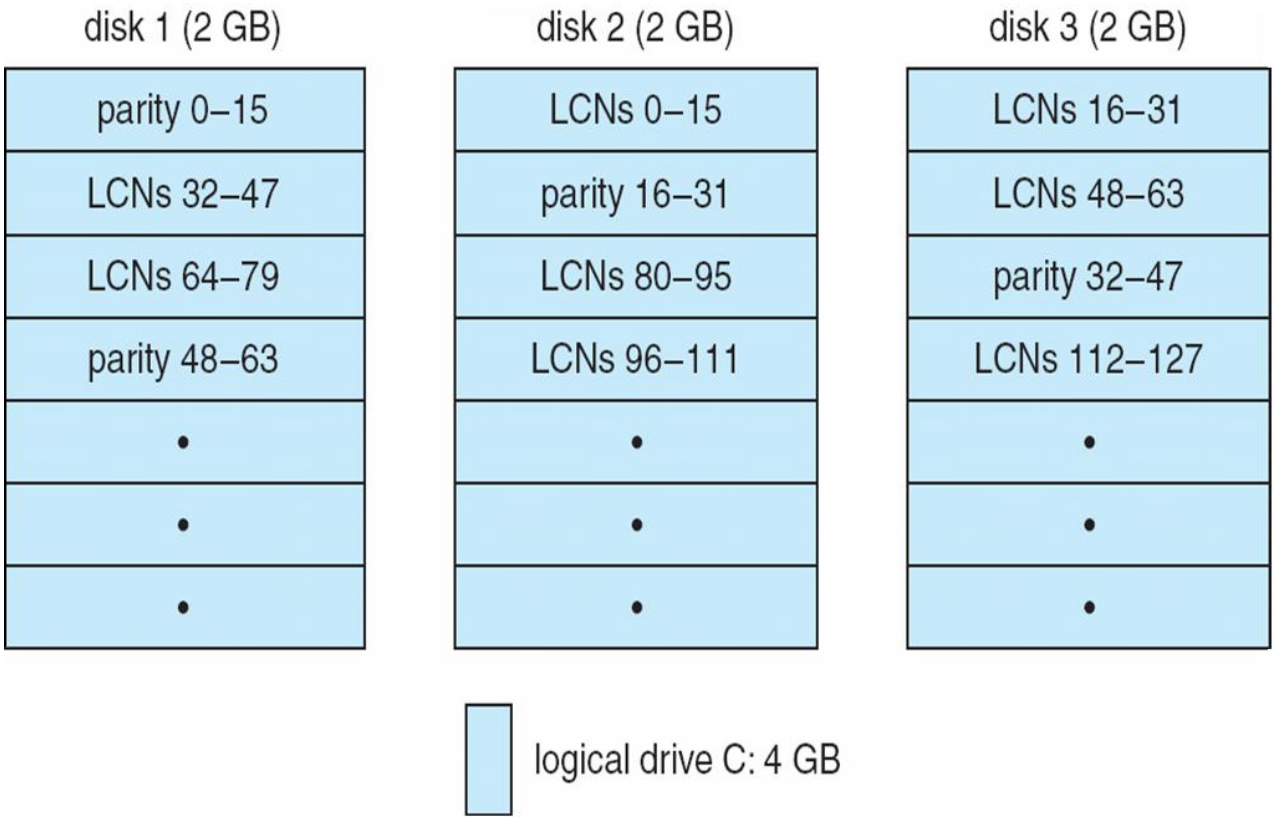
Stripe Set on Two Drives

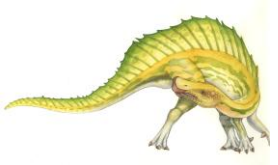






Stripe Set With Parity on Three Drives

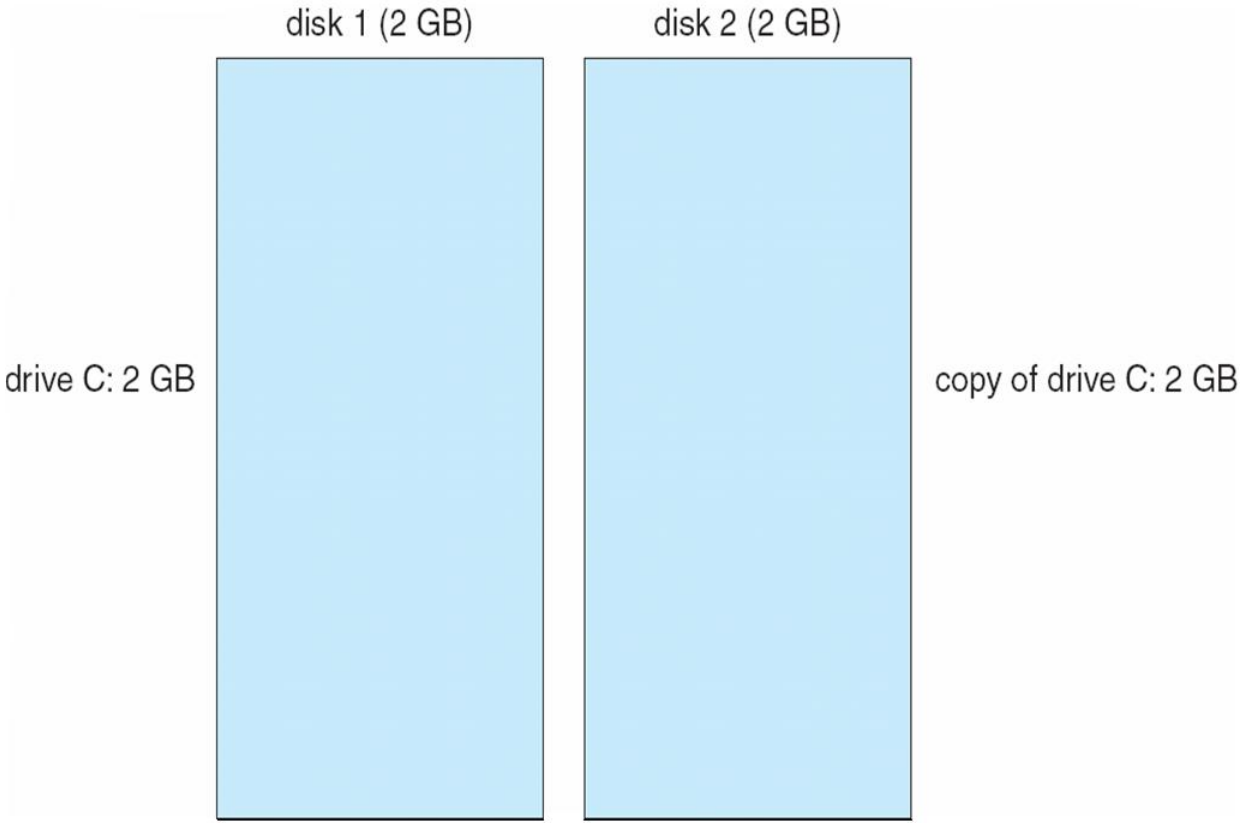




Troy University Resources



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

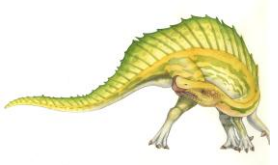


Troy University Resources



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 10 supports both peer-to-peer and client/server networking; it also has facilities for network management
- To describe networking in Windows 10, 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
  - Legacy TDI (Transport Driver Interface) is obsolete. Modern kernel networking uses interfaces such as Winsock Kernel (WSK)
- Windows 7 implements transport protocols as drivers that can be loaded and unloaded from the system dynamically



Troy University Resources



Networking — Protocols

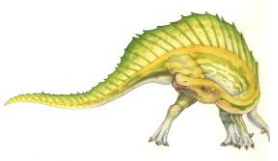
- SMB provides network file and printer access. Older SMB/NetBIOS descriptions grouped requests into these categories:
  1. `Session control`
  2. `File`
  3. `Printer`
  4. `Message`
- NetBIOS is a legacy network naming and session API
  - Used to:
    - Establish logical names on the network
    - 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.)

- Windows uses the TCP/IP Internet protocol version 4 and version 6 to connect to a wide variety of operating systems and hardware platforms
- PPTP is a legacy VPN protocol with known security weaknesses and is unsuitable for new secure deployments
- Legacy DLC supported communication with some IBM mainframes and HP printers. It is not a standard Windows 10 networking component

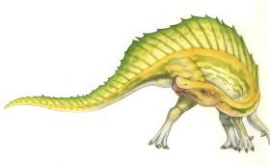


Troy University Resources



Networking — Distributed Processing Mechanisms

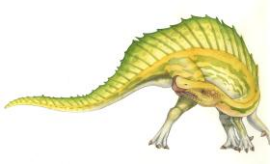
- Windows has supported named pipes, mailslots, Winsock, and RPC for distributed applications. NetDDE is a historical mechanism
- NetBIOS applications can communicate over the network using TCP/IP
- Named pipes are connection-oriented messaging mechanism that are named via the uniform naming convention (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 7 RPC mechanism follows the widely-used Distributed Computing Environment standard for RPC messages, so programs written to use Windows 7 RPCs are very portable.
  - RPC messages are sent using NetBIOS, or Winsock on TCP/IP networks, or named pipes on LAN Manager networks.
  - Windows 7 provides the Microsoft *Interface Definition Language* to describe the remote procedure names, arguments, and results.

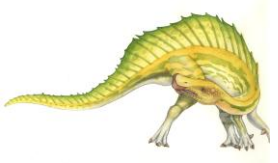


Troy University Resources



Networking — Redirectors and Servers

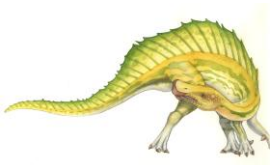
- In Windows 7, an application can use the Windows 7 I/O API to access files from a remote computer as if they were local, provided that the remote computer is running an MS-NET server.
- A *redirector* is the client-side object that forwards I/O requests to remote files, where they are satisfied by 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

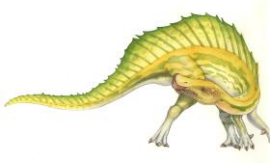


Troy University Resources



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

- Windows uses the concept of a domain to manage global access rights within groups
- A domain is a group of machines running Windows that share a common security policy and user database
- Windows provides three models of setting up trust relationships
  - *One-way: if A trusts B, users from B can be authorized to access resources in A*
  - *Two-way trust combines both directions. Transitivity is a separate property that can extend trust through intermediate domains*
  - *Crosslink – allows authentication to bypass hierarchy to cut down on authentication traffic*

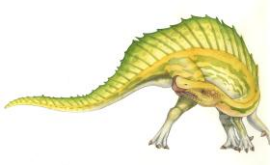


Troy University Resources



Name Resolution in TCP/IP Networks

- On an IP network, name resolution is the process of converting a computer name to an IP address
  - For illustration, host.example.test `could m`ap to `192.0.2.10`
- 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
- Legacy WINS uses one or more servers to maintain a dynamic NetBIOS-name-to-IPv4-address database, queried by clients
- Clients register names with WINS. DHCP can supply WINS server addresses and related client settings





Programmer Interface — Access to Kernel Object

- A process obtains a process-local *handle* by creating or `opening` an object, or by handle inheritance or duplication
- `CloseHandle` releases a handle. Object deletion depends on remaining handles, kernel references, and object lifetime rules
- 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:
    - DuplicateHandle can give another process a handle to the same object, subject to the required process access rights

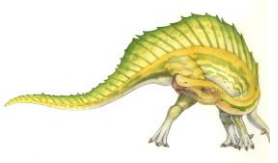


Troy University Resources



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 six process priority classes:
  1. `IDLE_PRIORITY_CLASS` (priority level 4)
  2. `BELOW_NORMAL_PRIORITY_CLASS` (NT priority level 6)
  3. `NORMAL_PRIORITY_CLASS` (level 8 — typical for most processes
  4. `ABOVE_NORMAL_PRIORITY_CLASS` (level 10)
  5. `HIGH_PRIORITY_CLASS` (level 13)
  6. `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`
  - 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, increases the scheduling quantum by some factor, typically 3

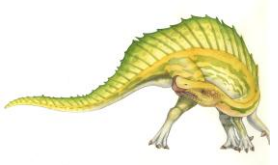


Troy University Resources



Process Management (Cont.)

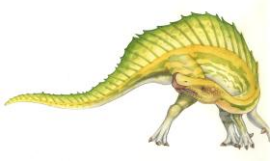
- The kernel dynamically adjusts the priority of a thread depending on whether it is I/O-bound or CPU-bound
- To synchronize the concurrent access to shared objects by threads, the kernel provides synchronization objects, such as semaphores and mutexes
  - In addition, threads can synchronize by using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





Process Management (Cont.)

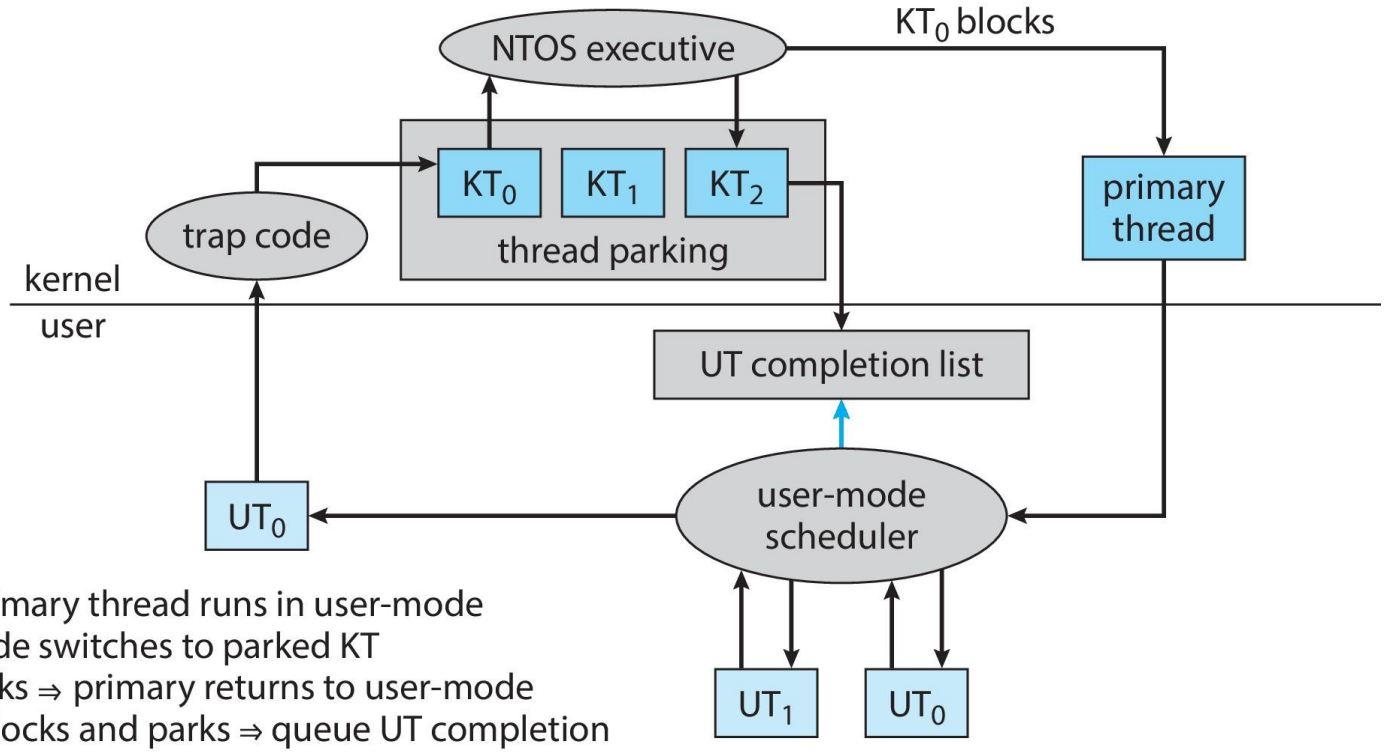
- A fiber is user-mode code that gets scheduled according to a user-defined scheduling algorithm
  - Each thread runs one fiber at a time. Fibers on different threads can execute in parallel
  - Windows includes fibers to facilitate the porting of legacy UNIX applications that are written for a fiber execution model
- Windows also introduced user-mode scheduling for 64-bit systems which allows finer grained control of scheduling work without requiring kernel transitions



Troy University Resources



User-mode Scheduling



only primary thread runs in user-mode
trap code switches to parked KT
KT blocks ⇒ primary returns to user-mode
KT unblocks and parks ⇒ queue UT completion





Programmer Interface — Interprocess Communication

- 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
- A thread's message queue is created on demand when it first uses the relevant USER or GDI functions
- 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

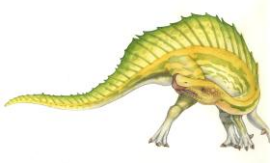


Troy University Resources



Programmer Interface — Memory Management

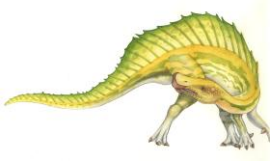
- Virtual memory:
  - `VirtualAlloc` reserves or commits virtual memory
  - `VirtualFree` decommits or releases the memory
  - These functions enable the application to determine the virtual address at which the memory is allocated
- An application can use memory by memory mapping a file into its address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





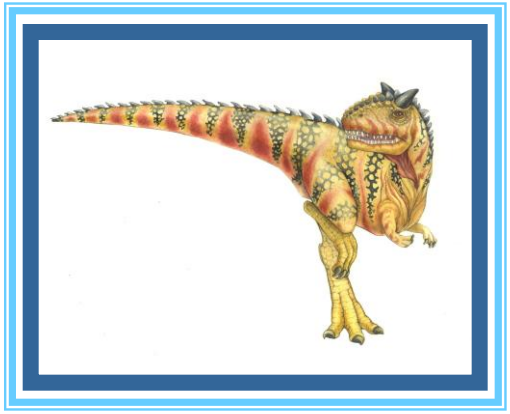
Memory Management (Cont.)

- A heap in the Win32 environment is a region of reserved address space
  - The executable header sets the *default heap reserve, commonly 1 MiB. Memory is committed as needed*
  - 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



Troy University Resources

End of Chapter 21



Silberschatz, Galvin and Gagne ©2018