CSE 120
Principles of Operating Systems

Fall 2004

Lecture 2: Operating System Modules and Interfaces

Geoffrey M. Voelker

Modules, Interfaces, Structure

- We roughly defined an OS as the layer of software between hardware and applications
- Now we're going to survey the support OSes provide to applications
  - Modules – OS services and abstractions
  - Interfaces – operations supported by components
  - Structure – how components get hooked together
    - » Will discuss in depth later in the quarter
    - » Makes more sense after being exposed to much of the material
- This is all a high-level preview of what we are going to cover in the course

OS Module Overview

- Common OS modules
  - Processes
  - Memory
  - I/O
  - Secondary storage
  - Files
  - Protection
  - Accounting
  - Command interpreter (shell, window system)
- We'll survey each module and discuss its interface

Process Module

- An OS executes many kinds of activities
  - User programs
  - Batch jobs or command scripts
  - System programs (daemons): print spoolers, name servers, file servers, Web servers, etc.
  - Run "ps -edaf" on Solaris – What is all that stuff?
- Each "execution entity" is encapsulated in a process
  - A process includes both the program (code, data) and execution context (PC, regs, address space, resources, etc.)
- Process module manages processes
  - Creation, scheduling, deletion, etc.

Process Interface

- Process module interface
  - Create a process
  - Delete a process
  - Suspend a process
  - Resume a process
  - Inter-process communication
    - » Transfer, share data
  - Inter-process synchronization
  - Process relationships
    - » Parent, child, process groups

Memory

- Primary memory is the direct access storage for CPU
  - Programs must be stored in memory to execute
  - Interacts with process module
- Operating systems
  - Allocate memory for programs (explicitly and implicitly)
  - Deallocate memory when needed (by rest of system)
  - Maintain mappings from virtual to physical memory (page tables)
  - Decide how much memory to allocate to each process
    - » Large space of policy decisions
  - Decide when a process should be removed from memory
    - » More policy decisions

Input/Output (I/O)

- Much of an OS deals with device I/O
  - One of the main reasons we use OSes
  - Hundreds of thousands of lines of code in NT for I/O, drivers
- The OS provides a standard interface between programs (user or system) and devices
  - File system (disks), sockets (network), frame buffer (video)
- Device drivers are the routines responsible for controlling I/O devices
  - OS defines an interface for each class of devices (e.g., disks)
  - A driver implements interface, encapsulates device-specific knowledge (initiation and control, interrupt handling, errors)

Secondary Storage

- Secondary storage (disk) is the persistent memory
  - It endures system failures (for the most part)
- Low-level OS routines are often responsible for low-level disk functions
  - Read/write blocks
  - Schedule requests (optimize arm movement)
  - Device errors
- Usually independent of file system
  - Although there might be cooperation (e.g., free space management)
  - Low-level knowledge can help FS performance (placement)

File System

- Secondary storage devices are too crude to use directly for long-term storage
  - Read/write physical device blocks too low-level for programs
- The file system provides a much higher-level, more convenient abstraction for persistent storage
  - Objects (files, directories) and interfaces (read, write, etc.)
- Files are the basic storage entity
  - A file is a named collection of persistent information
- Directories are special files that contain the names of other files + metadata (data about files, attributes)
  - Directories have all properties of files ("inheritance")

File System Interface

- File system interface provides standard file operations
  - Existence: File/directory creation, deletion
  - Manipulation: open, read, write, append, rename, close, etc.
  - Sometimes higher-level operations
    - » File copy, change notification (NT)
    - » Records (IBM)
- File system also provides general services
  - Backup
  - Consistency
  - Compression
  - Encryption
  - Accounting and quotas

Protection

- Protection is a general mechanism throughout OS
- All objects (resources) need protection
  - Processes
  - Memory
  - Devices
  - Files
- Protection mechanisms help to prevent errors as well as prevent malicious destruction
  - E.g., running as root

Accounting

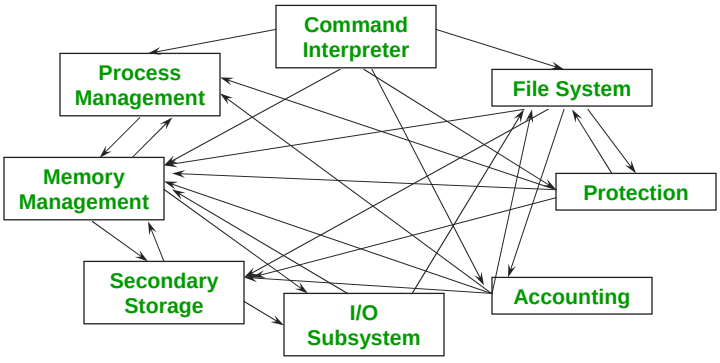
- General facility for keeping track of resource usage for all system objects
  - Quotas in the file system (Unix: “quota –v”)
  - Memory usage (Unix: “man limit”)
  - Process resource usage (Unix: “rusage <command>”)
- Resource usage might be used to bill customers
  - In world of PCs, might seem strange
  - In world of mainframes and minicomputers, crucial
    - » Departments, users billed for CPU time
      - e.g., IBM mainframe “turbo” switch

Command Interpreter (Shell)

- Process that:
  - Handles user input (commands)
  - Manages subprocesses
  - Executes script files (files of commands)
- On some systems, CI is part of OS
  - Users constrained to use that CI (DOS)
- Others, it is just another user-level process
  - Unix shell
  - Any program can be a CI (sh, csh, ksh, bash, etc.)
- Or, there may not be a command language at all
  - Original MacOS (hey, where's the shell?)
  - What about MacOS 10?

The Challenge of Structure

- It is clear what modules an OS should provide
- Not so clear how to hook them together (well)…



For next class...

- Browse the course web
  - http://www.cse.ucsd.edu/classes/fa04/cse120/
- Read Chapters 1, 2, and 3
- Start reading Nachos Chapter, Appendix C (online)
- Start thinking about partners for project groups

- No discussion this Friday (nothing to discuss)
- Accounts available soon