

Chapter 2: Operating-System Structures

Chapter 2: Operating-System Structures

- ▶ Operating System Services
- ▶ User Operating System Interface

Objectives

- ▶ To describe the services an operating system provides to users, processes, and other systems
- ▶ To discuss the various ways of structuring an operating system
- ▶ To explain how operating systems are installed and customized and how they boot

Operating System Services

- ⦿ One set of operating-system services provides functions that are helpful to the user:
 - User interface - Almost all operating systems have a user interface (UI)
 - Varies between Command-Line (CLI), Graphics User Interface (GUI), Batch
 - Program execution - The system must be able to load a program into memory and to run that program, end execution, either normally or abnormally (indicating error)
 - I/O operations - A running program may require I/O, which may involve a file or an I/O device.
 - File-system manipulation - The file system is of particular interest. Obviously, programs need to read and write files and directories, create and delete them, search them, list file information, permission management.

Operating System Services (Cont.)

- ▶ One set of operating-system services provides functions that are helpful to the user (Cont):
 - ▶ Communications - Processes may exchange information, on the same computer or between computers over a network
 - ▶ Communications may be via shared memory or through message passing (packets moved by the OS)
 - ▶ Error detection - OS needs to be constantly aware of possible errors
 - ▶ May occur in the CPU and memory hardware, in I/O devices, in user program
 - ▶ For each type of error, OS should take the appropriate action to ensure correct and consistent computing
 - ▶ Debugging facilities can greatly enhance the user's and programmer's abilities to efficiently use the system

Operating System Services (Cont.)

- ▶ Another set of OS functions exists for ensuring the efficient operation of the system itself via resource sharing
 - ▶ **Resource allocation** - When multiple users or multiple jobs running concurrently, resources must be allocated to each of them
 - ▶ Many types of resources - Some (such as CPU cycles, main memory, and file storage) may have special allocation code, others (such as I/O devices) may have general request and release code.
 - ▶ **Accounting** - To keep track of which users use how much and what kinds of computer resources
 - ▶ **Protection and security** - The owners of information stored in a multiuser or networked computer system may want to control use of that information, concurrent processes should not interfere with each other
 - ▶ **Protection** involves ensuring that all access to system resources is controlled
 - ▶ **Security** of the system from outsiders requires user authentication, extends to defending external I/O devices from invalid access attempts
 - ▶ If a system is to be protected and secure, precautions must be instituted throughout it. A chain is only as strong as its weakest link.

User Operating System Interface - CLI

CLI allows direct command entry

- ▶ Sometimes implemented in kernel, sometimes by systems program
- ▶ Sometimes multiple flavors implemented - **shells**
- ▶ Primarily fetches a command from user and executes it
 - ▶ Sometimes commands built-in, sometimes just names of programs
 - ▶ If the latter, adding new features doesn't require shell modification

User Operating System Interface - GUI

- ◎ User-friendly **desktop** metaphor interface
 - Usually mouse, keyboard, and monitor
 - **Icons** represent files, programs, actions, etc
 - Various mouse buttons over objects in the interface cause various actions (provide information, options, execute function, open directory (known as a **folder**))
 - Invented at Xerox PARC
- ◎ Many systems now include both CLI and GUI interfaces
 - Microsoft Windows is GUI with CLI “command” shell
 - Apple Mac OS X as “Aqua” GUI interface with UNIX kernel underneath and shells available
 - Solaris is CLI with optional GUI interfaces (Java Desktop, KDE)