Chapter 2

Operating Systems Concepts

Read…

- **Introduction to Operating Systems**
  - Silberschatz: Chapters 1–3
  - Tanenbaum: Chapter 1
- **Linux Programming:** Chapter 1
- (USP: Chapter 1 & Appendix A)

Components of a Computer System

- **Hardware**
  - Processor(s)
  - Memory
  - I/O devices
- **Operating system**
  - Kernel — e.g., Linux
  - System programs — e.g., GNU tools
- **Application programs**
- **Users**

Major Benefits of an OS

- **Convenience:** facilitates the use of hardware
- **Efficiency:** ensures that resources are used efficiently
- **Security:** ensures that resources are not misused
- **Communication:** enables access to other computers
- **Real-Time Support:** enables real-time constraints to be met

Characteristics of Modern OSs

- **Multiprogramming:** Several programs can be executed together
  - Improves performance of the computer system
  - Requires **job scheduling**
- **Time-sharing:** Several programs can be executed simultaneously by sharing the CPU (CPUs)
  - Enables several users to use the computer simultaneously
  - Requires **CPU scheduling**

Operating System Components

- Process management
- Main memory management
- File management
- I/O management
- Secondary storage management
- Networking
- Security
- Command Interpreter

Process Management

A **process** is a running program

— The OS must be able to **start** and **stop processes**

A process runs sequentially

— The OS must **control access to the CPU**

A process requires access to system resources

— The OS **allocates** and **controls system resources**

A process may need to communicate with other processes

— The OS must provide **communication mechanisms**

Main Memory Management

Main memory is the work space for the CPU

- Consists of a large array of words
- Each word is addressed by its index in the array

The code and data of an executing process must be **resident** in main memory

- For efficiency, several processes are usually simultaneously resident in main memory
- A process may not completely fit in main memory

File Management

- A **file** is an abstract, uniform unit of stored information
  – Files are organised into **directories**
- Files can be stored on several kinds of physical media
- Information access is largely controlled by file access

Secondary storage management

- **Main memory** has significant **limitations:**
  – It is relatively **small** in size
  – It is a **volatile** storage medium
- Secondary storage is needed to cover these limitations
  – A computer's performance strongly depends on how efficiently it manages secondary storage
  – Main memory is already a kind of "secondary storage" compared with registers and **cache**

I/O Device Management

- A computer includes several I/O **hardware devices**
- An operating system wraps an **I/O subsystem** around each I/O device to:
  – Hide the physical aspects of the device
  – Provide a uniform software-based means of accessing I/O devices

Networking System

- A computer may communicate with other computers on a physical **data network** (e.g., Ethernet or ATM)
- A computer may communicate across an **internet** of physical networks
- Several computers working together can form a **distributed computing system**
- Computer networking opens up a Pandora's box of **security concerns**

Information Security System

Concerned with the protection of:

- Electronically stored and manipulated information
- Operating system
- Application-level information systems

Why information security is unique:

- Concerned with **misuse** instead of **proper use**
- Hard to engineer

Security Implementation Problems

- Involves most components of an information system
- Information security requirements clash with many other system requirements
- Cuts across component boundaries and levels of abstraction

A system is only as secure as its weakest component

Command Interpreter

- The **command interpreter** is the user interface of an operating system
  - May or may not be part of the OS kernel
- Kinds of command interpreters:
  - Command line interpreter (called a **shell** in Unix)
  - Mouse-based window and menu system

Simple OS Architecture

- Levels:
  - Application programs
  - System programs
  - *Kernel interface*
  - Kernel
  - *Hardware interface*
  - Hardware
- Example: Unix

Kernel Interface

- The interface functions are special instructions called **system calls** or **traps**
  - **Programming interface** to the kernel
  - Instruction set of the **kernel virtual machine**
- Steps in **executing a system call**
  - Process P invokes a system call instruction I
  - Kernel gains control of CPU via **interrupt**
  - The code for I is executed
  - P regains control of the CPU

Major Kernel Subinterfaces

- **Process management**
- **File and I/O device management**
  - I/O devices are treated as files (makes programs device independent)
- **Interprocess communication (IPC)**
  - May be local or remote
  - Two models: **message passing**, **shared memory**

System Programs

- Provide **operating system services** to **users**
- Many system programs are **interfaces to system calls**
- **Command interpreter** is a major system program
  - **Interactive interface** to the kernel
  - Two implementation models:
    - Single program
    - Collection of system programs (as in Unix)

      Distinguish:
      - Shell built-in functions
      - Separate commands

Layered OS Architecture

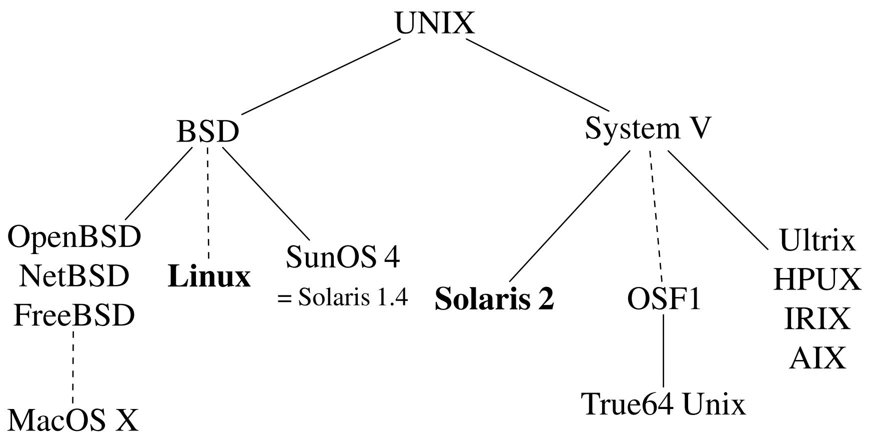
- Operating system is composed of several layers starting at the hardware and ending at the user interface
- Each layer is a module:
  - Interface is available to higher-layer implementations
  - Implementation is defined using lower-layer interfaces
- ***Advantages:***
  - Modularity: layers can be developed as separate units
- ***Disadvantages:***
  - Hard to define a clean set of layers
  - The more layers, the higher the overhead

    — Example of tension between **clarity** and **efficiency**

Microkernel OS Architecture

- Operating system is built on top of a **microkernel** made as small as possible which includes:
  - Process management
  - Memory management
  - Interprocess communication
- System programs compose the rest of the OS
- Example: Mach (developed at CMU in mid 1980s, used in OSF1, Tru64 Unix, GNU HURD, MacOs X)
- ***Advantages:***
  - Easy to extend OS: add more system programs
  - Easier to maintain; more secure and reliable
  - Individual operating systems can be implemented as collections of system programs

UNIX Flavours



POSIX