Overview of today's lecture

- Major components of an operating system
- Structure and internal architecture of an operating system
- Monolithic Vs Micro-kernels
- Virtual Machine Monitors
- Re-cap of the lecture

Major OS Components

- Process management
- Memory management
- I/O
- Secondary Storage
- File System
- Protection
- Accounting
- Shell (OS UI)
- GUI
- Networking

Process Operation

- The OS provides the following kinds operations on processes (i.e. process abstraction interface)
  - Create a process
  - Delete a process
  - Suspend a process
  - Resume a process
  - Clone a process
  - Inter-process communication
  - Inter-process synchronization
  - Create / delete a child process

I/O

- A Big Chunk Of OS Kernel deals with I/O
  - Millions of Lines in windows XP (including drivers)
- The OS provides standard interface between programs and devices
- Device drivers are the routines that interact with specific device types:
  - Encapsulates device specific knowledge
  - E.g. how to initialize a device, how to request the I/O, how to handle interrupts and errors
  - E.g. SCSI device drivers, Ethernet card drivers, video card drivers, sound card drivers.

- Note: windows has ~35000 device drivers.

Secondary Storage

- Secondary storage (disk, tape) is persistent memory
  - Often magnetic media survives power failures (hopefully)

- Routines that interact with disks are typically at a very low level in the OS
  Used by many components
  - Handle scheduling of disk operations, head movement,
  - Error handling and often management of space on disk

- Usually independent of file system
  - Although there may be cooperation
  - File system knowledge of device details can help optimize performance
    E.g. place related files close together on disk

File System

- Secondary storage device are crude and awkward
  E.g. write 4096 byte block to a sector
- File system are convenient abstraction
- A file is a basic long term storage unit
- A directory is just a special kind of file

Command interpreter (shell)

- A particular program that handles the interpretation of users commands and helps to manage processes
- On some systems, command interpreter may be a standard part of the OS
- On others, its just not privileged code that provides an interface to the user
- On others there may be no command language

File system operations

The file system interface defines standard operations
- File (or directory) creation and deletion
- Manipulating of files and directories
- Copy
- Lock

File system also provide higher level services
- Accounting and quotes
- Backup
- Indexing or search
- File versioning

Accounting

- Keeps track of resource usage
- Both to enforce quotas “you’re over the disk limit”
  Or
  to produce bills
- Important for time shared computers like mainframes

Networking

An OS typically has a built-in communication infra-structure that

implements:

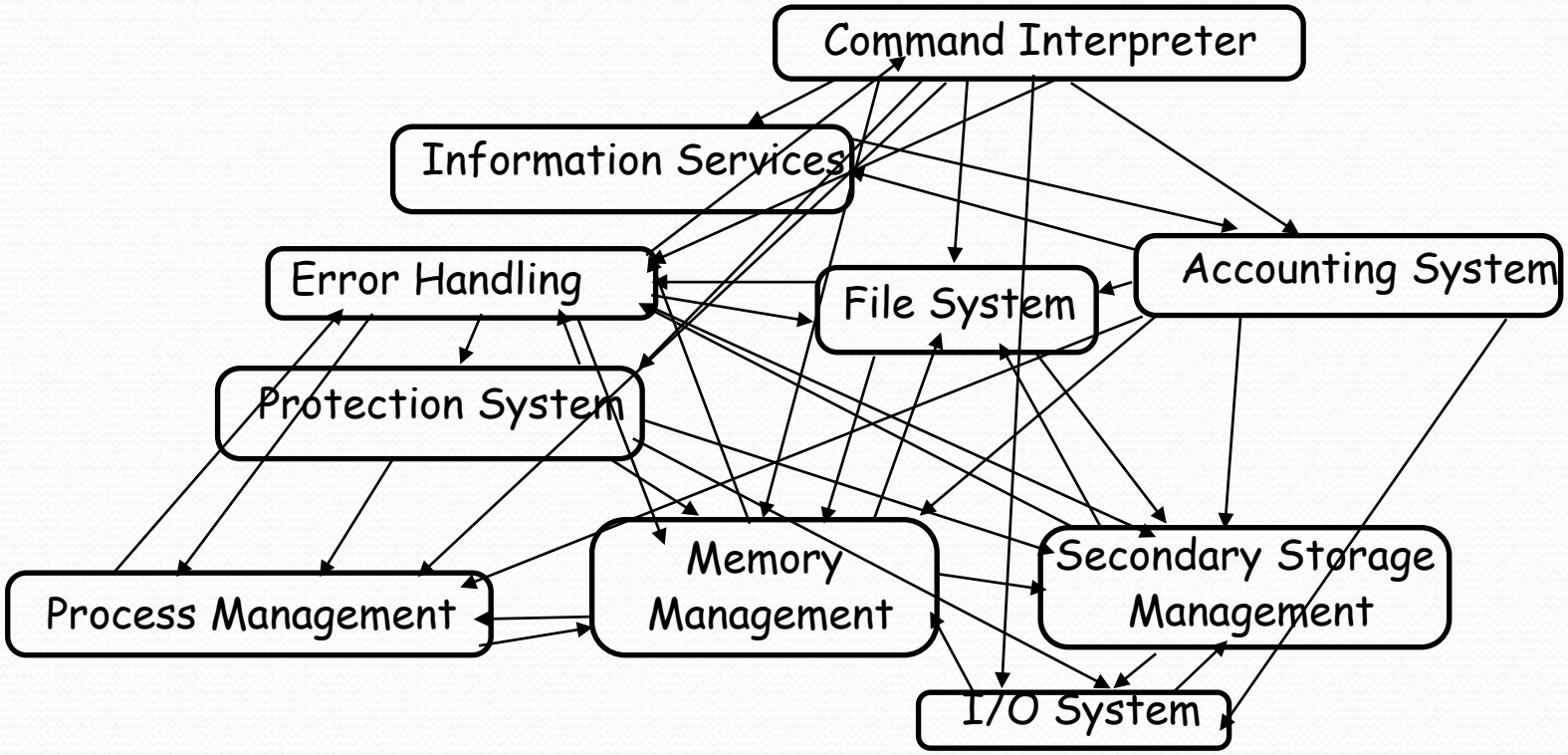
a. A network protocol software stack

b. A route lookup module to map a given destination address to a next hop.

c. A name lookup service to map a given name to a destination machine.

OS structure

- It's not always clear how to stitch OS modules together:

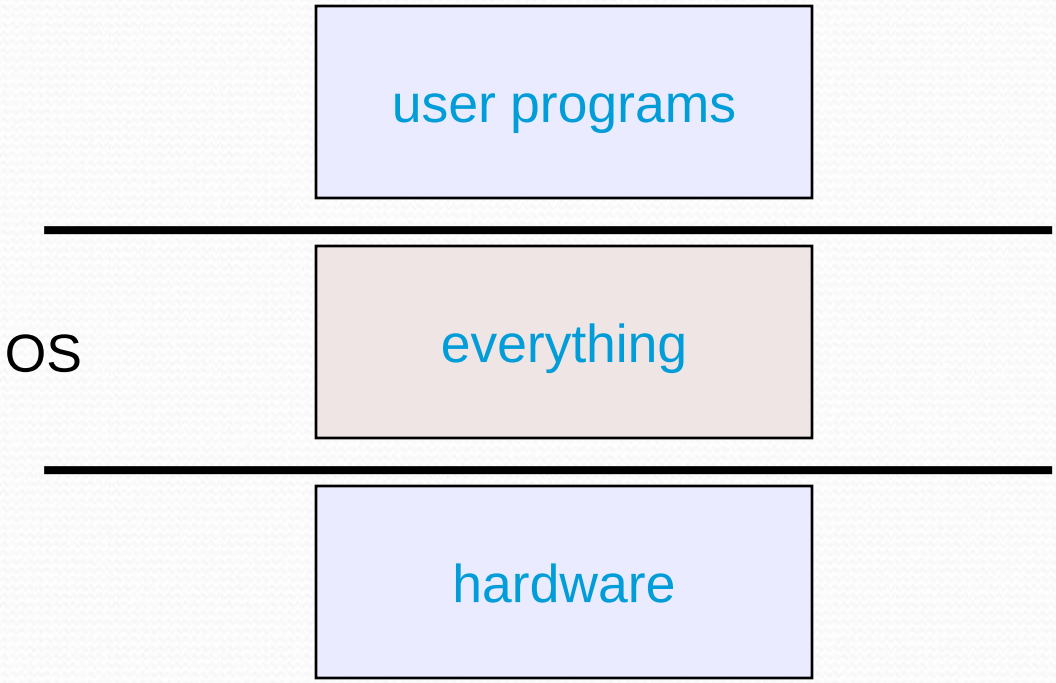


OS Structure

- An OS consists of all of these components, plus:
  - Many other components
  - System programs (e.g. boot strap code, the init program).
- Major issues:
  - How do we organize all this?
  - What are all the code modules, and where do they exist?
  - How do they cooperate?
- Massive software engineering and design problem
  - Design a large complex program that:
    - Performs well, is reliable, is extensible, is backwards compatible…

Early structure: Monolithic

- Traditionally, OS's (like UNIX, DOS) were built as a monolithic entity:



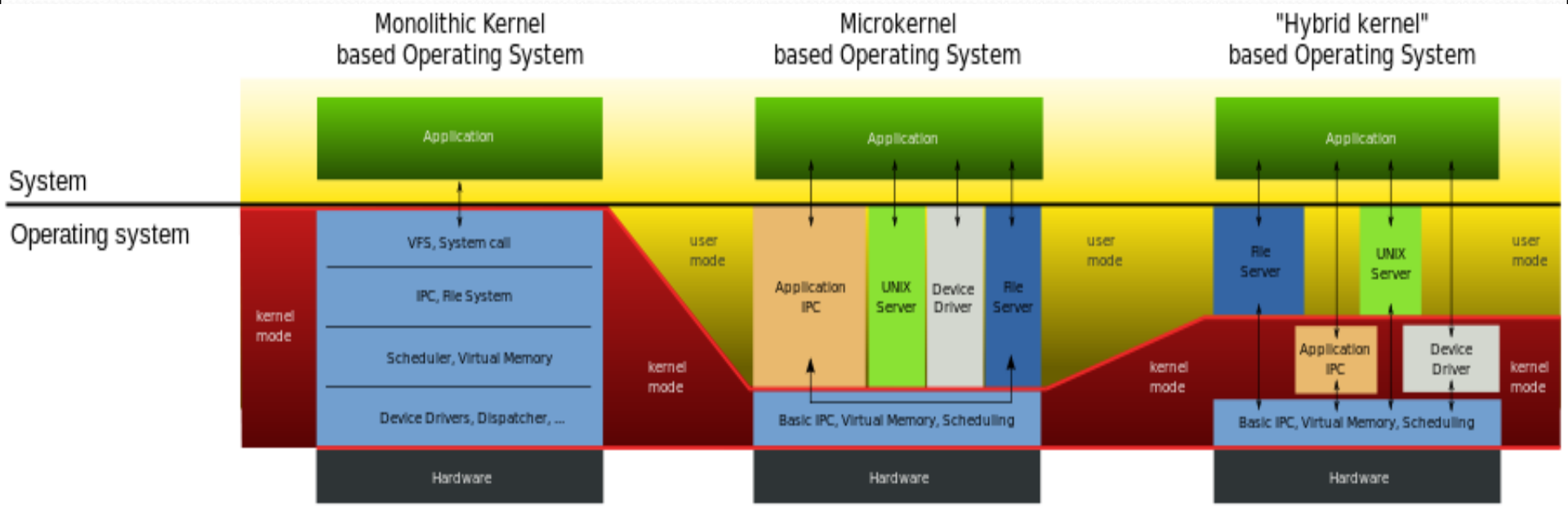
Monolithic Design

- Major Advantages:
  - Cost of module interaction is low
- Disadvantages
  - Hard to understand
  - Hard to modify
  - Unreliable
  - Hard to maintain
- What id alternative?
  - Find ways to organize the OS in order to simplify its design and implementation.

Monolithic Design

- A **monolithic kernel** is an operating system architecture where the entire operating system is working in the kernel space and alone as supervisor mode.
- The monolithic differs from other operating system architectures (such as the microkernel architecture) in that it defines alone a high-level virtual interface over computer hardware, with a set of primitives or system calls to implement all operating system services such as process management, concurrency, and memory management itself and one or more device drivers as modules.

Comparison b/w different kernel Operating Systems



Monolithic Design

- Major Advantages:
  - Cost of module interaction is low
- Disadvantages
  - Hard to understand
  - Hard to modify
  - Unreliable
  - Hard to maintain
- What id alternative?
  - Find ways to organize the OS in order to simplify its design and implementation.

Layering

- The traditional approach is layering
  - Implement OS as a set of layers
  - Each layer presents an enhanced virtual machine to the layer above.
- The first description of the system was Djakarta's system.
  - Layer 5: job managers
  - Layer 4: device managers
  - Layer 3: console manager
  - Layer 2: pager manager
  - Layer 1: Kernel
  - Layer 0: Hardware

Problems with layering

Imposes hierarchical structure

- but real system are more complex:
  - file system requires VM services (buffers)
  - VM would like to use files for its backing store
- strict layering isn't flexible enough
  - Poor performance
- each layer crossing has overhead associated with it
  - Disjunction between model and reality
- systems modeled as layers, but not really built that way

Microkernel OS

- In computer science, a **microkernel** is the near-minimum amount of software that can provide the mechanisms needed to implement an operating system (OS).
- These mechanisms include low-level address space management, thread management, and inter-process communication (IPC).
- If the hardware provides multiple rings or CPU modes, the microkernel is the only software executing at the most privileged level (generally referred to as supervisor or kernel mode).
- Traditional operating system functions, such as device drivers, protocol stacks and file systems, are removed from the microkernel to run in user space.

- Minimum functionality in the kernel. Most of the OS functionality in user level servers. Examples of servers are file servers, terminal servers, memory servers etc.
- Each part becomes more manageable. Crashing of one service doesn't bring the system down.
- Distribution of the system becomes transparent.
- Kernel provides basic primitives e.g. transport of messages, loading programs into memory, device handling.
- Policy decisions are made in the user space while mechanisms are implemented in micro-kernel.

- Micro-kernel lends itself well to OO design principles. Components based design possible.
- Disadvantage: Performance
- Solutions:
- Reduce micro-kernel size
- Increase micro-kernel size

Virtual Machine Monitors

- Export a virtual machine to user programs that resembles hardware.
- A virtual machine consists of all hardware features e.g. user/kernel modes, I/O, interrupts and pretty much everything a real machine has.
- A virtual machine may run any OS.

  Examples:

  JVM, VM Ware, User-Mode Linux (UML).

  Advantage: portability

  Disadvantage: slow speed