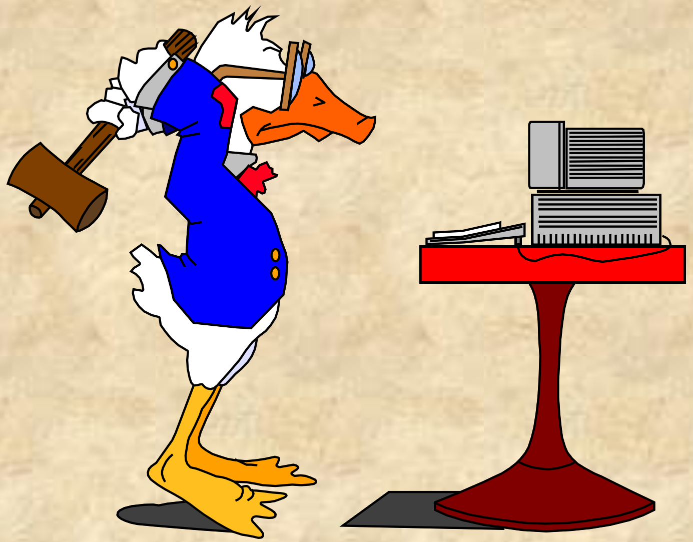
Process Management in Linux

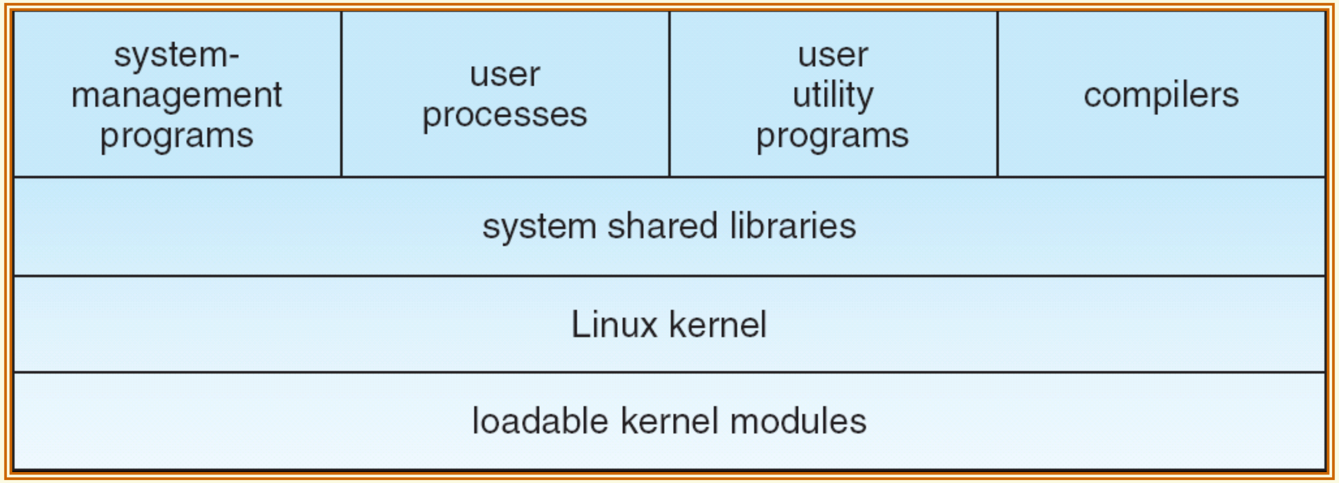
Dr.Hashamdar

Massachusetts Institute of Technology (MIT)



The Linux Operating System

- **The Linux kernel tracks:**
  - The location of the process execution (context)
  - Which files the process is accessing or has access to
  - What users and groups the process belongs to (credentials)
  - current directory for the process is
  - memory space the process has access to and how it uses it.



system-management programs

user processes

user utility programs

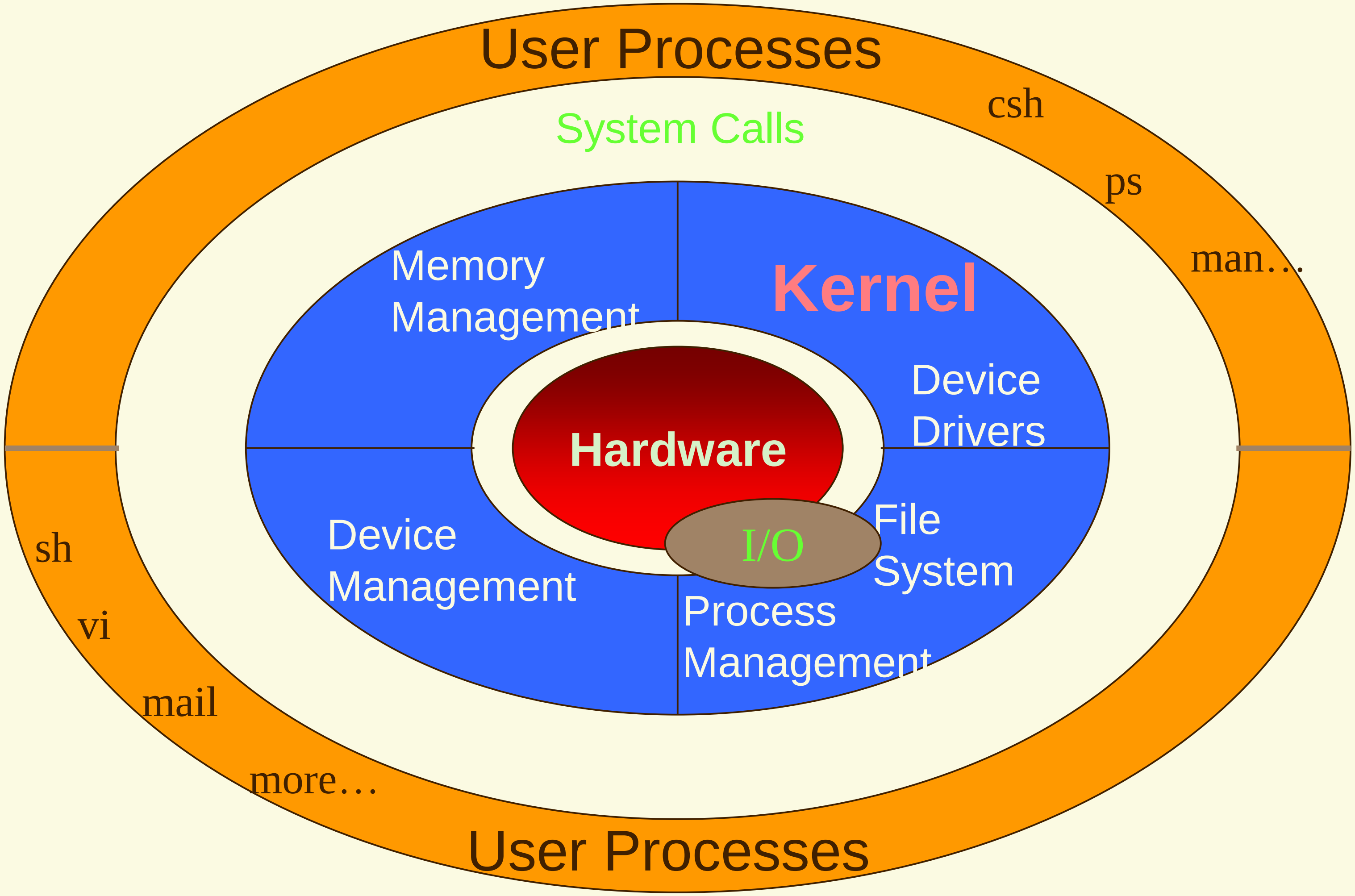
compilers

system shared libraries

Linux kernel

loadable kernel modules

What is a process?



What is a process?

- Processes are not just **programs**
  - Program instructions plus other components as needed (primarily data)
- Sometimes called **jobs** or **tasks**

Process Management

- UNIX process management separates the creation of processes and the running of a new program into two distinct operations.
  - The **fork** system call creates a new process
  - A new program is run after a call to **exec**
- Under UNIX, a process encompasses all the information that the operating system must maintain *t* track the context of a single execution of a single program

Process Management

- Under Linux, process properties fall into 3 groups

(i) Process’s Identity

(ii) Process’s Environment

(iii) Process’s Context

(i) Process Identity

- Three different attributes for a *process identity*

  a) Process ID (PID)

  b) Credentials

  c) Personality

a) Process ID (PID)

- PID is a unique integer
- PID cannot be changed by the process
- Kernel uses PID to track a process
  - Tracks exit status
  - Relationship to other processes

a) Process ID (PID)

- When process exits
  - PID is kept until it can be safely discarded.
  - PID that is kept in an active status is called a zombie.
  - All child processes (now called orphan processes) become child processed of the init process.

(i) Process Identity

- Three different attributes for a *process identity*

  a) Process ID (PID)

  b) Credentials

  c) Personality

b) Credentials

- Are used to insure security
- Made up of
  - users
  - and groups
- User ID (UID) and Group ID (GID)
  - either one can be set within a limit
  - symbolic user and group names that are mapped to a unique integer value (usually positive).

(i) Process Identity

- Three different attributes for a *process identity*

  a) Process ID (PID)

  b) Credentials

  c) Personality

c) Personality

- Personality identifiers allow slight modifications to the semantics of certain system calls.
- “Personalities are primarily used by emulation libraries to request that system calls can be compatible with certain specific flavors of unit.” (Galvin, 710)

(i) Process Identity

- Three different attributes for a *process identity*
    a) Process ID (PID)
    b) Credentials
    c) Personality

Process Management

- Under Linux, process properties fall into 3 groups

(i) Process's Identity

(ii) Process's Environment

(iii) Process's Context

(ii) Process Environment

- The process’s environment is inherited from its parent, and is composed of **2 null-terminated vectors**:
  - The argument vector lists the command-line arguments used to invoke the running program; conventionally starts with the name of the program itself
  - The environment vector is a list of “**NAME=VALUE**” pairs that associates named environment variables with arbitrary textual values

(ii) Process Environment

- Passing environment variables among processes and inheriting variables by a process’s children are flexible means of passing information to components of the user-mode system software

- The environment-variable mechanism provides a customization of the operating system that can be set on a per-process basis, rather than being configured for the system as a whole

Process Management

- Under Linux, process properties fall into 3 groups

(i) Process's Identity

(ii) Process's Environment

(iii) Process's Context

(iii) Process Context

- The (constantly changing) state of a running program at any point in time
- The **scheduling context** is the most important part of the process context; it is the information that the scheduler needs to suspend and restart the process
- The kernel maintains **accounting** information about the resources currently being consumed by each process, and the total resources consumed by the process in its lifetime so far

(iii) Process Context

- The **file table** is an array of pointers to kernel file structures
  - When making file I/O system calls, processes refer to files by their index into this table
- The **signal-handler table** defines the routine in the process’s address space to be called when specific signals arrive
- The **virtual-memory context** of a process describes the full contents of the its private address space

Process Management

- Under Linux, process properties fall into 3 groups

(i) Process’s Identity

(ii) Process’s Environment

(iii) Process’s Context

Processes and Threads

- Linux uses the same internal representation for processes and threads;
- a thread is simply a new process that happens to share the same address space as its parent

Processes and Threads

- A distinction is only made when a new thread is created by the **clone** system call
  - **Fork** creates a new process with its own entirely new process context
  - **clone** creates a new process with its own identity, but that is allowed to share the data structures of its parent
- Using **clone** gives an application fine-grained control over exactly what is shared between two threads

Organization of Table of Processes

- Each process is referenced by **descriptor**
  - Describes process attributes together with information needed to manage process
- Kernel dynamically allocates these descriptors when processes begin execution
- All process descriptors are organized in **doubly linked list**
- Scheduler used **Macro instructions** to manage and update process descriptor lists as needed

Process Synchronization

- To allow two processes to synchronize with each other, Linux provides:
  - **Wait queue:** Linked circular list of process descriptors
  - **Semaphores:** Used to solve problems of mutual exclusion and problems of producers and consumers
    - In Linux they contain 3 fields:
      - Semaphore counter
      - Number of waiting processes
      - List of processes waiting for semaphore