Linux Internals

Introduction to Linux

Module Contents

- **Background and History of Linux**
- **Concepts of OS**
  - Where does an OS fit in?
  - What does an OS do?
  - Who needs OS?
- **Program versus Process**
- **Process/kernel model**
- **Linux OS Roles and Features**
  - Process Management
  - Memory Management
  - File Systems
  - H/W and N/W Management

History of Linux

- Linux is a member of the large family of Unix-like operating systems
- Linux was initially developed by Linus Torvalds in 1991 as an operating system for IBM compatible personal computers based on the Intel 80386 microprocessor
- It is open source and the source code is open under the GNU Public License
- Source code is available for study as well as modification http://www.kernel.org/
- Source code is present in the path /usr/src. But directory structure inside this path can differ depending on the distribution

Advantages of Using Linux

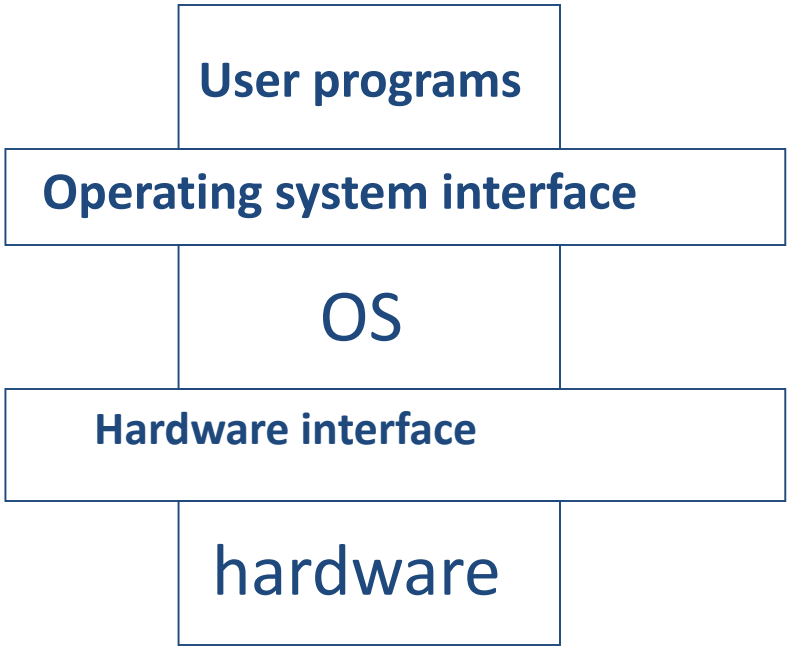
- **Linux is free**
- **Linux is fully customizable in all its components**
- **Linux runs on low-end, cheap hardware platforms**
- **Linux is powerful**
- **Linux has a high standard for source code quality**
- **The Linux kernel is very small and compact**

Advantages of Using Linux (Contd…)

- **Linux is highly compatible with many common operating systems**
- **Linux is well supported**

Where does an OS fit in?

- **The layer between the hardware and the user programs (application programs)**
- **OS is a software system that directly interacts with the hardware**



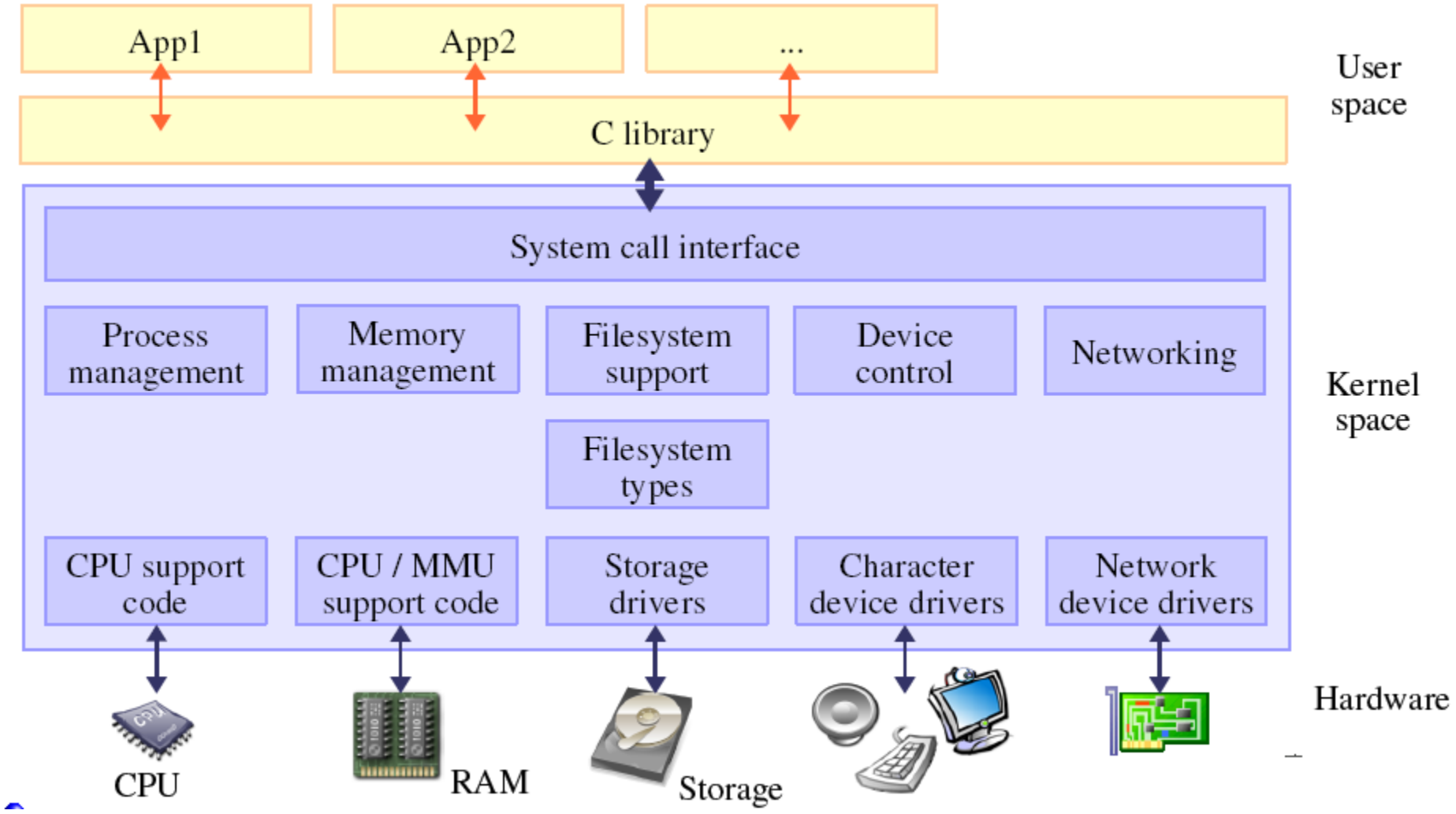
What does an OS do?

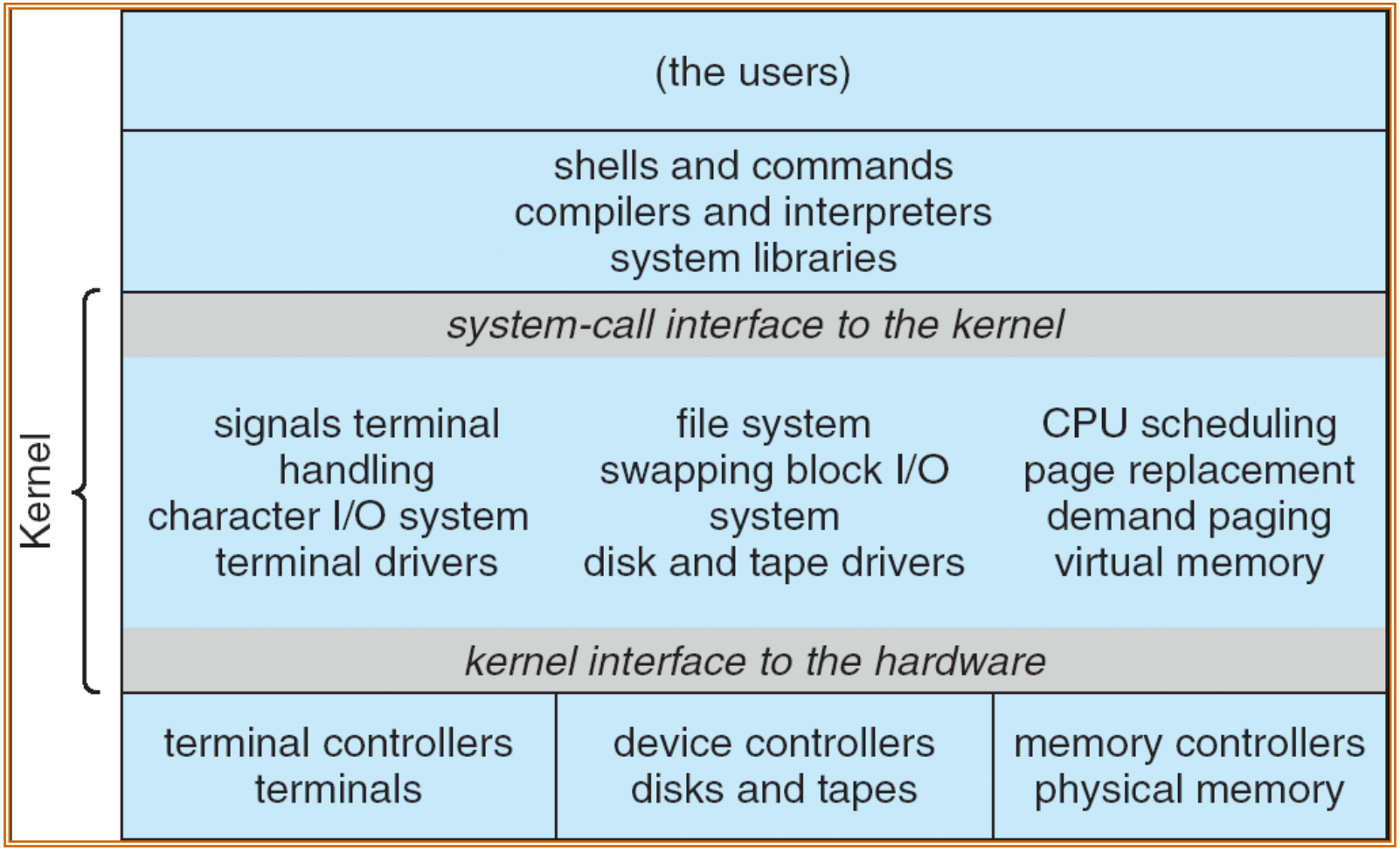
- **OS manages resources such as memory, process, I/O devices etc**
- **How does computer hardware run programs?**
  - A program in memory, starting from program counter (pc), run till the halt instruction.
- **Share resources (CPU, memory, disk, ….)**
  - For each program running (a process), the OS makes each process feel that it solely owns the CPU, owns the memory -- Virtual machine for each process
- **Manage the processes (create, exit, switch, scheduling)**

Who needs OS?

- **OS makes computer easier to use**
  - All general purpose computers need OS.
- **A better question: Who does not need OS?**
  - Some very specialized (and small) systems do not need OS.
  - Example microwave oven control system (the OS functions are implemented in the application).

Linux OS Role & Features



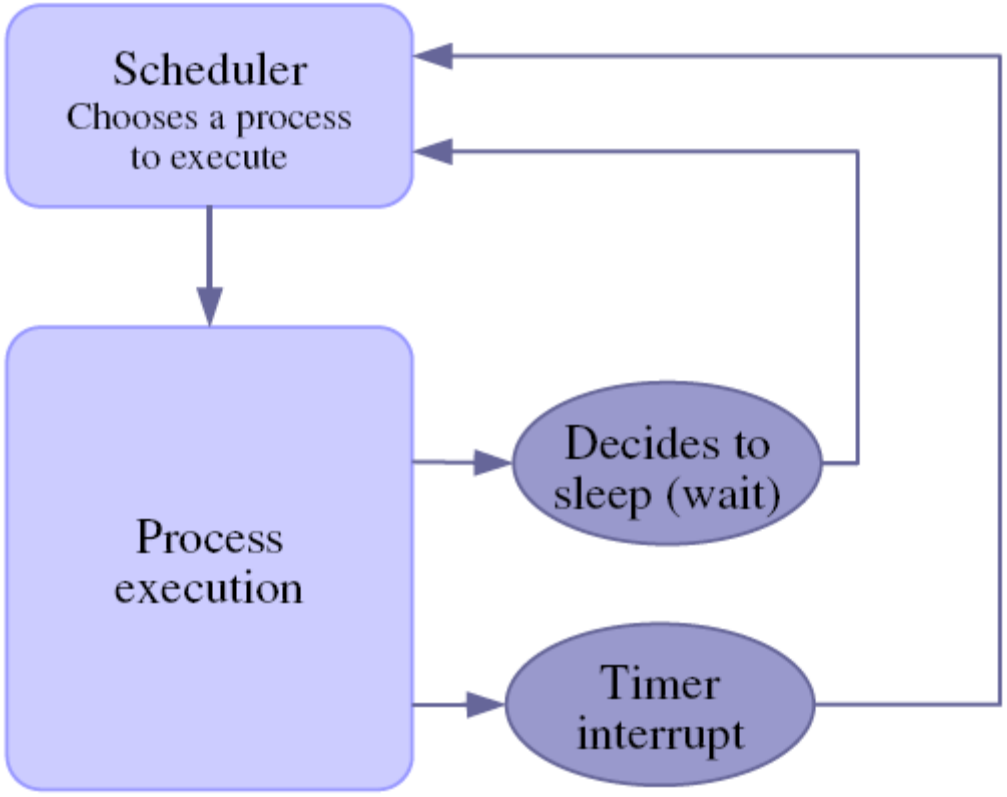

(the users)
shells and commands
compilers and interpreters
system libraries
system-call interface to the kernel
signals terminal handling
character I/O system
terminal drivers
file system
swapping block I/O system
disk and tape drivers
CPU scheduling
page replacement
demand paging
virtual memory
kernel interface to the hardware
terminal controllers
terminals
device controllers
disks and tapes
memory controllers
physical memory
Kernel

Program versus Process

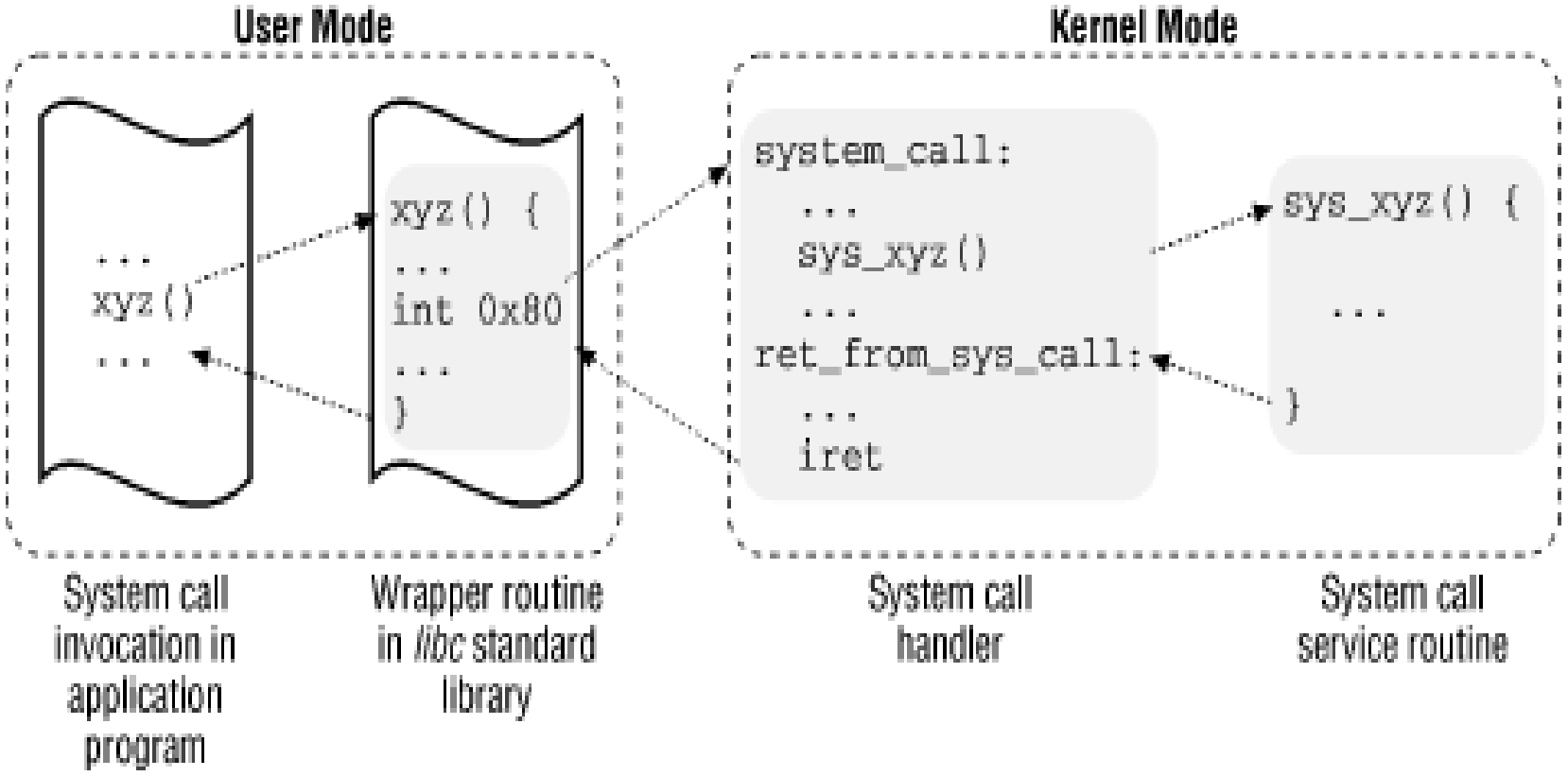
- **A program by itself is not a process**
- **A program is a passive entity**
  - Data stored on the disk or in memory
- **A process is an active entity**
  - Instructions are waiting to be executed
  - Instructions must be executed sequentially
  - Many processes can execute the same program
- **Several processes can execute the same program concurrently**

Process Scheduling

- **Linux implements preemptive multitasking**



System Call Interface Process/kernel model



Process management

- **The OS must provide the ability to**
  - Create and delete user and system processes
  - Suspend and resume processes
  - Provide a method for process synchronization
  - Provide a method for process communication
  - Provide a method to handle deadlock
- **Tracking a Process**
  - As a process-manager, the OS must keep track of each process that is currently in existence, to facilitate cooperation among them and to mediate competing demands

Memory Management

- **Main memory is increasing in size as time goes on.**
- **As memory size increases, so does the complexity of maintaining the memory.**
- **As we add more devices, we must allocate more memory to run the hardware, i.e.**
  - Sound cards
  - modems
- **Thus programs are getting bigger and more complex, and to run these programs more (Main) memory is required**

Memory Management(Contd..)

- **The OS must keep track of memory and decide:**
  - What memory is currently being used
  - Who is using the memory
  - Which processes will be allocated memory when memory frees up
  - How much memory should be allocated to a particular process
  - When memory should be deallocated

File Management

- **The way files are managed is one of the most visible parts of an OS**
- **File management must be logical and efficient**
- **Efficiency is needed as the average number of files on a given disk increases**
- **User control may also need to be implemented, controlling which files should be accessed by any given user**

File Management (Contd..)

- **The OS is responsible for:**
  - Creating and deleting files and directories
  - Supporting basic functionality needed for manipulating those files and directories
  - Mapping files onto secondary storage (i.e. saving the file)
  - Backing up files on stable media

I/O System Management

- **A OS should hide the underlying differences between various types of hardware.**
- **For example, saving a file to disk should be no different if the disk is IDE or SCSI**
- **I/O handling is hidden from the rest of the system through a standard interface known as the I/O Subsystem**

Networking

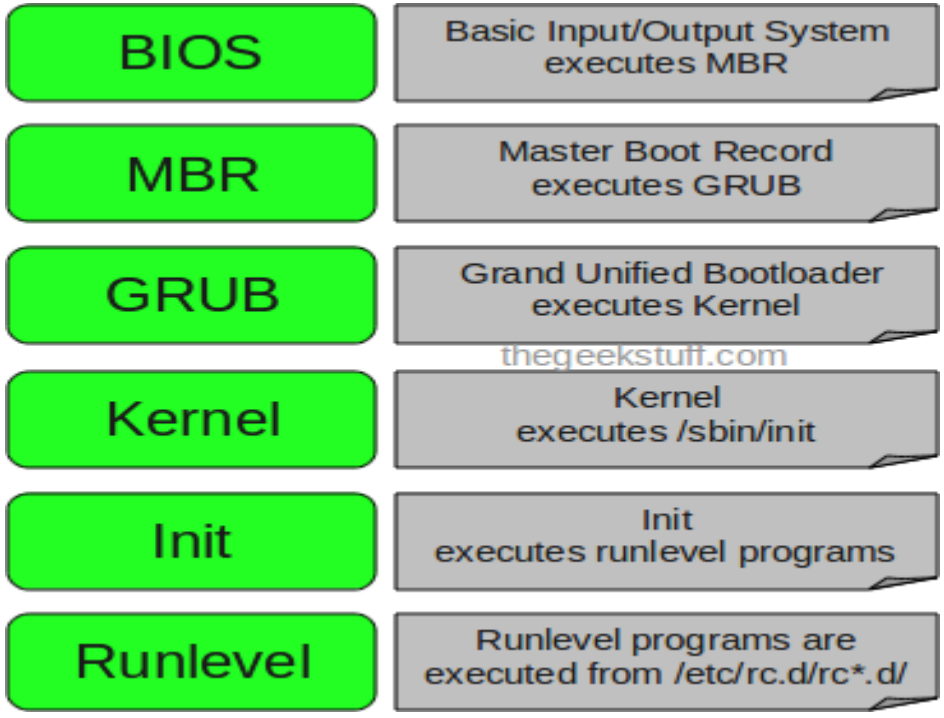
- **Computers are connected through a communication network**
- **Network communication must be controlled and managed**
- **Without communication between OSs, distributed systems would not be possible**
- **Most OSs generalize network access as a form of file access**

Protection System

- **If we want to run concurrent programs, we must be able to protect:**
  - Processes
  - CPU
  - Disk
  - Memory
  - Devices
- **If we cannot guarantee these basics, then we cannot guarantee the integrity of the processes**

Command Interpreter System

- **The command interpreter is the interface between the OS and the user**
- **The command interpreter can be a part of the OS kernel or may be run as an external program.**
- **Commands are given to the OS by using control statements**
- **The program used to interpret the control statements is known as the "command interpreter" or the "command shell"**


BIOS
Basic Input/Output System
executes MBR
MBR
Master Boot Record
executes GRUB
GRUB
Grand Unified Bootloader
executes Kernel
thegeekstuff.com
Kernel
Kernel
executes /sbin/init
Init
Init
executes runlevel programs
Runlevel
Runlevel programs are
executed from /etc/rc.d/rc*.d/

Runlevels

Run Level	Mode	Action
0	Halt	Shuts down system
1	Single-User Mode	Does not configure network interfaces, start daemons, or allow non-root logins
2	Multi-User Mode	Does not configure network interfaces or start daemons.
3	Multi-User Mode with Networking	Starts the system normally.
4	Undefined	Not used/User-definable
5	X11	As runlevel 3 + display manager(X)
6	Reboot	Reboots the system

Most Linux servers lack a graphical user interface and therefore start in runlevel 3. Servers with a GUI and desktop Unix systems start runlevel 5. When a server is issued a reboot command, it enters runlevel 6.

Review

- Who developed the Linux OS and When?
- Why one needs operating System?
- What are the advantages of Linux?
- What is process? How process is selected for execution in Linux? How it is related to program?
- When does process enter in Kernel mode?
- What are the different functions of Linux OS?
- What are the activities performed by OS for Process Management?