Operating System Concepts

SunBeam Institute of Information & Technology, Hinjwadi, Pune & Karad.

Trainer: Akshita Chanchlani
Email: akshita.chanchlani@sunbeaminfo.com

 # Booting:

- There are two steps of booting:

1. Machine Boot:

Step-1: when we switched on the power supply current gets passed to the motherboard on which from ROM memory one micro-program gets executes first called as **BIOS(Basic Input Output System).**

Step-2: first step of BIOS is **POST(Power On Self Test),** under POST it checks wheather all peripheral devices are connected properly or not and their working status.

Step-3: After POST it invokes **Bootstrap Loader** programs, which searches for available **bootable devices** presents in the system, and it selects only one bootable device at a time as per the priority decided in BIOS settings.

2. System Boot:

Step-4: After selection of a bootable device (budefault HDD), **Bootloader Program** in it gets invokes which displays list of names operating systems installed on the disk, from which user need to select any one OS.

Step-5: Upon selection of an OS, **Bootstrap Program** of that OS gets invokes, which locates the kernel and load into the main memory

 # UNIX Operating System:

- UNIX: UNICS – **Uniplexed Information & Computing Services/System.**

- UNIX was developed at **AT&T Bell Labs** in US, in the decade of 1970's by **Ken Thompson, Denies Ritchie** and team.

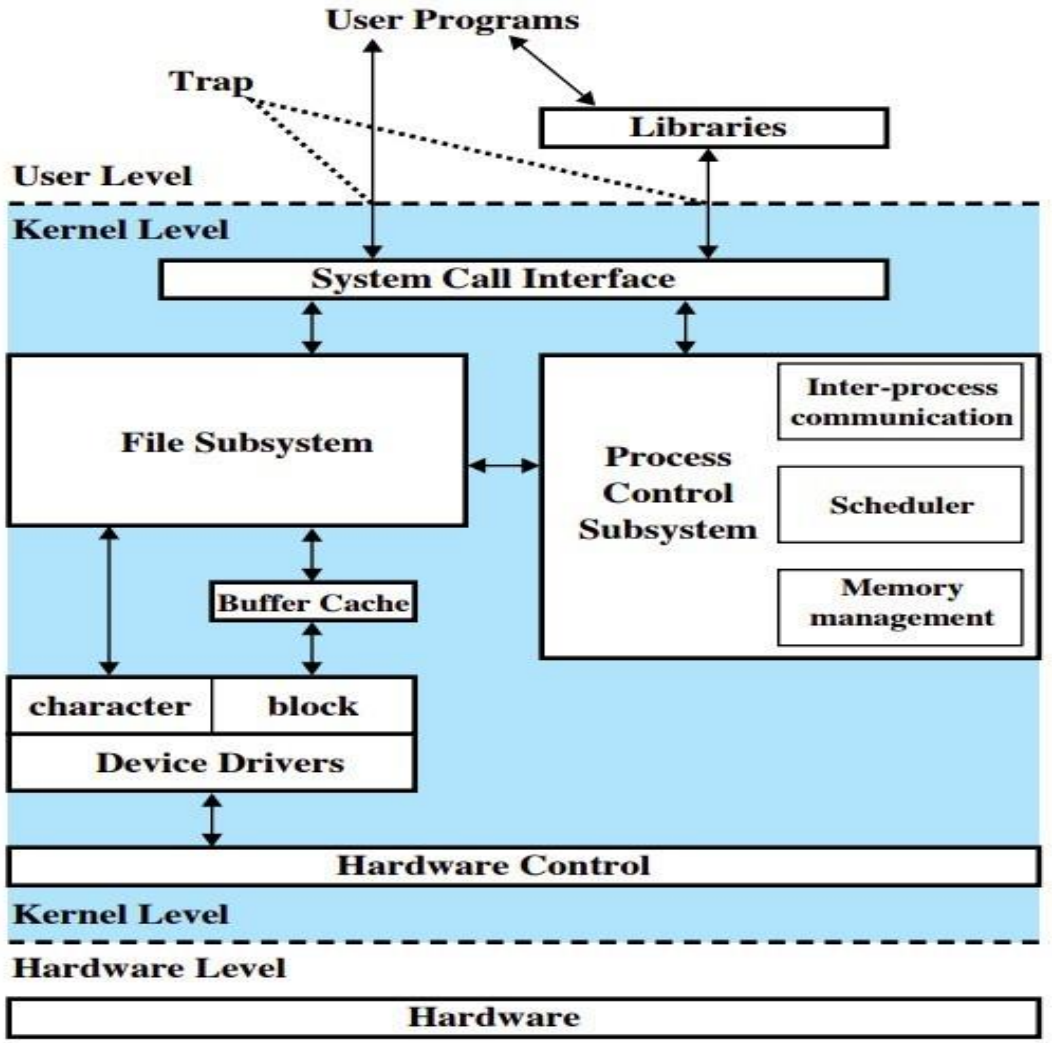
- It was first run on a machine **DEC-PDP-7** (Digital Equipment Corporation – Programmable Data Processing-7).

- UNIX is the first **multi-user, multi-programming & multi-tasking** operating system.

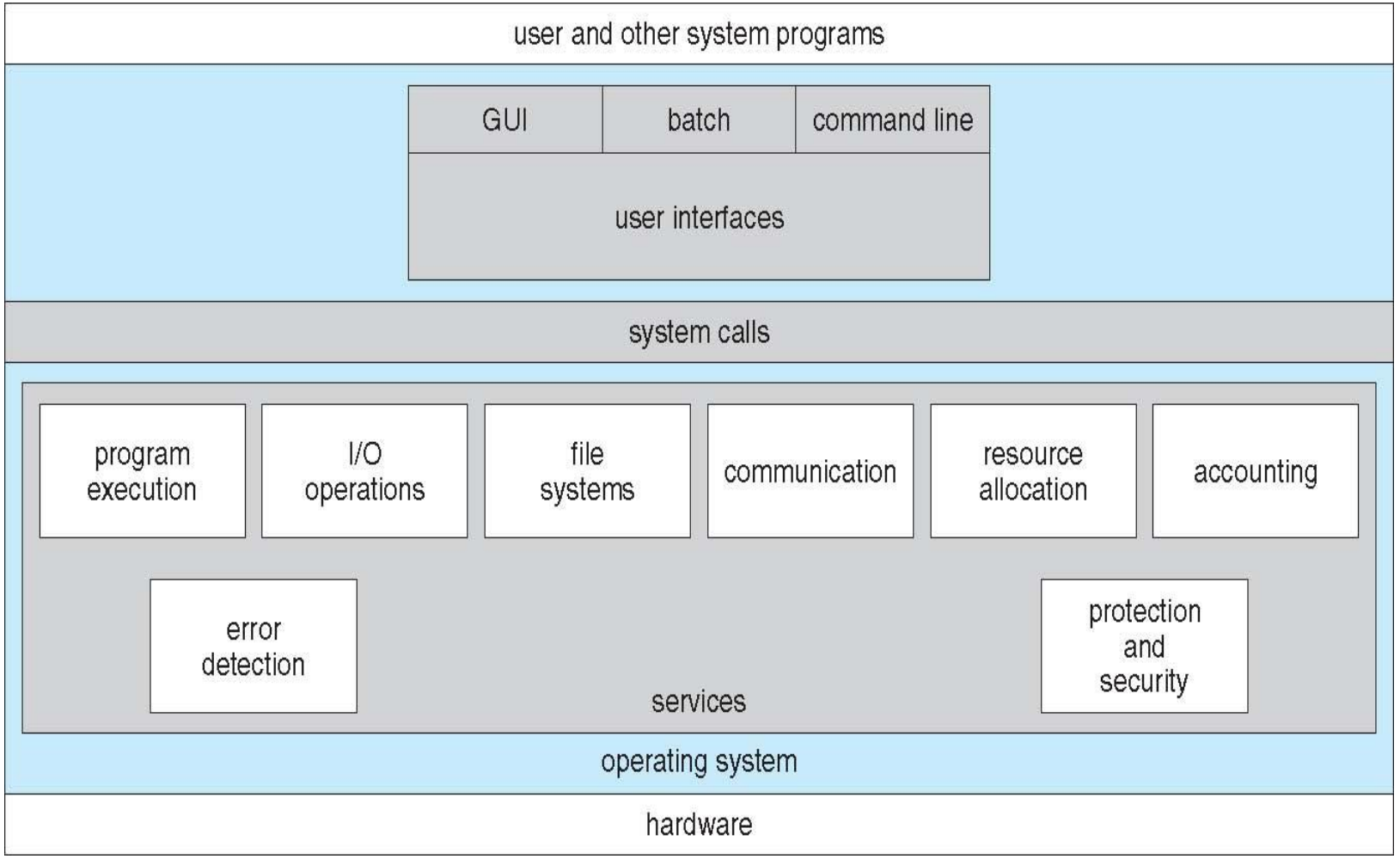
- UNIX was specially designed for developers by developers

- System architecture design of UNIX is followed by all modern OS's like Windows, Linux, MAC OS X, Android etc..., and hence UNIX is referred as **mother of all modern operating systems.**

Operating System Concepts



View of OS Services



Operating System Concepts

- Kernel acts as an interface between programs and hardware.

- Operating System has subsystems like **System Call Interface Block, File Subsystem Block, Process Control Subsystem Block (which contains IPC, Memory Management & CPU Scheduling), Device Driver, Hardware Control/Hardware Abstraction Layer.**

- There are two major subsystems:

1. Process Control Subsystem
2. File Subsystem

- In UNIX, whatever is that can be stored is considered as a file and whatever is active is reffered as a process.

- File has space & Process has life.

Operating System Concepts

- From UNIX point of view all devices are considered as a file
- In UNIX, devices are catagorises into two catagories:

1. Character Devices: Devices from which data gets transfered character by character --> character special device file

e.g. keyboard, mouse, printer, monitor etc...

2. Block Devices: Devices from which data gets transfered block by block --> block special device file

e.g. all storage devices.

- **Device Driver:**It is a program/set of programs enable one or more hardware devices to communicate with the computer's operating system.

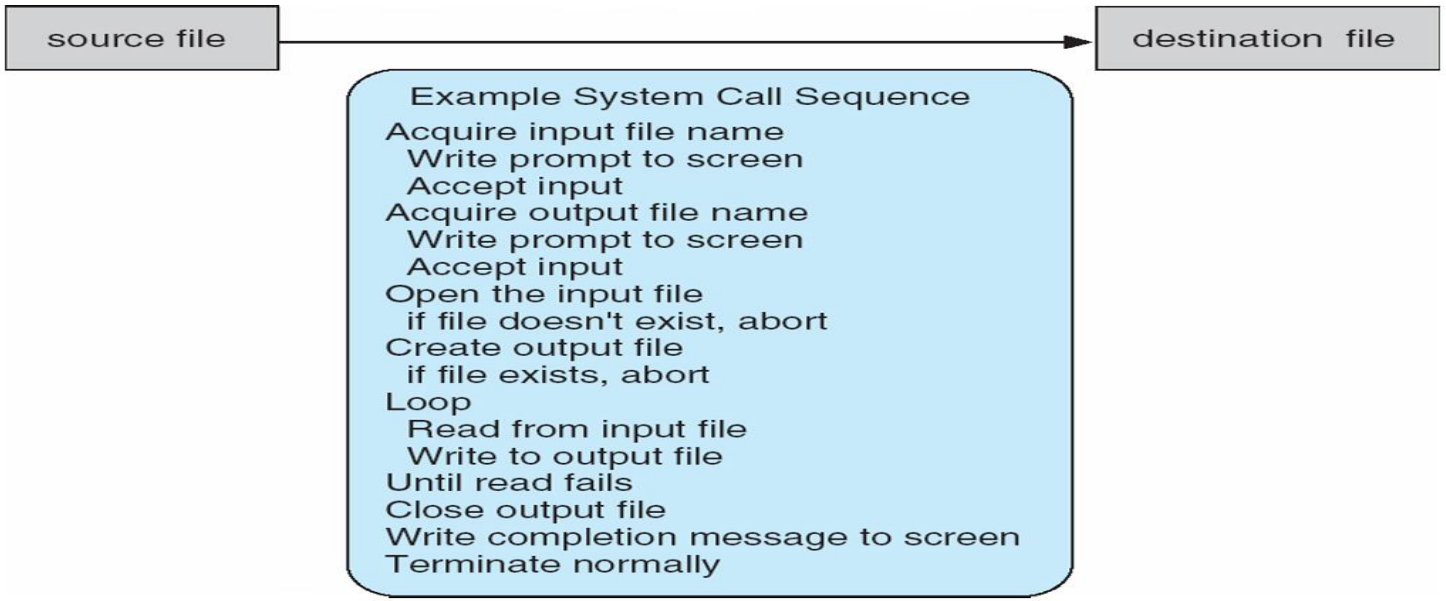
System Call

Programming interface to the services provided by the OS

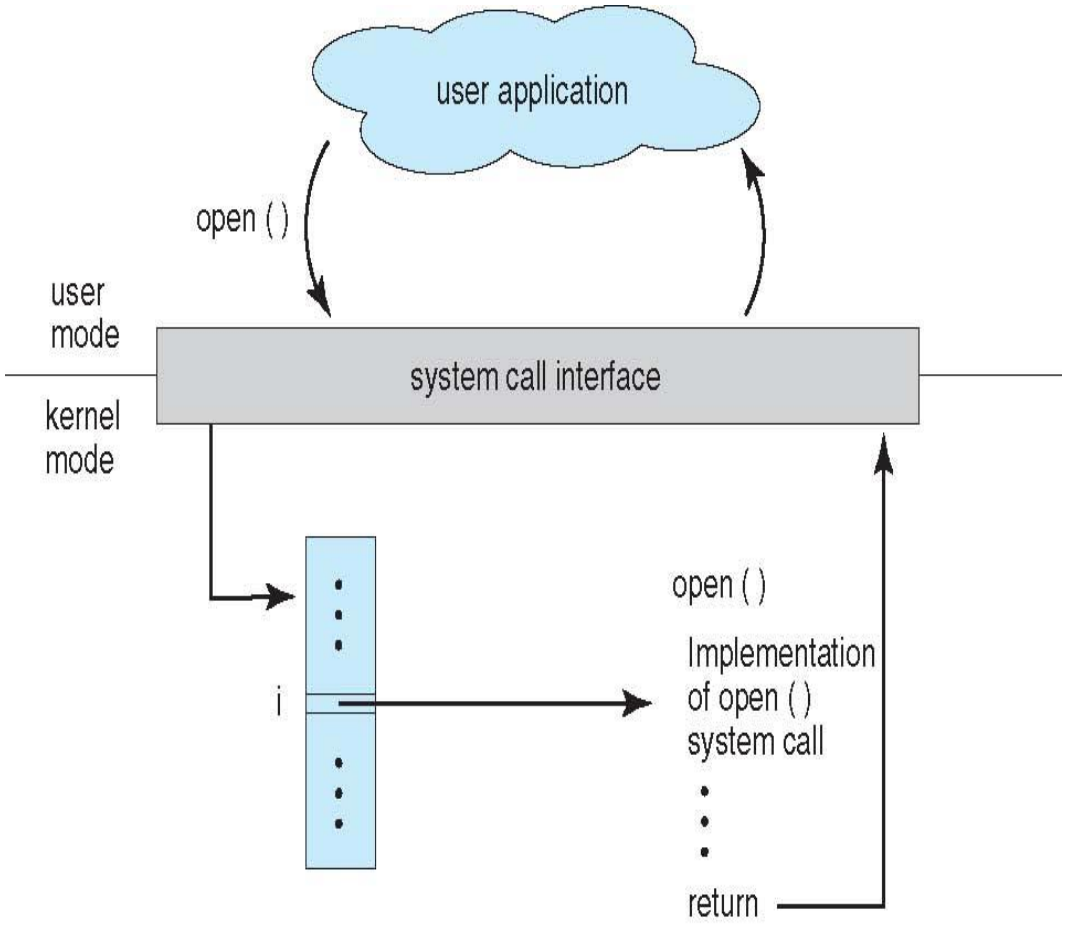
Mostly accessed by programs via a high-level **Application Program Interface (API)** rather than direct system call use.

The system call interface invokes intended system call in OS kernel and returns status of the system call

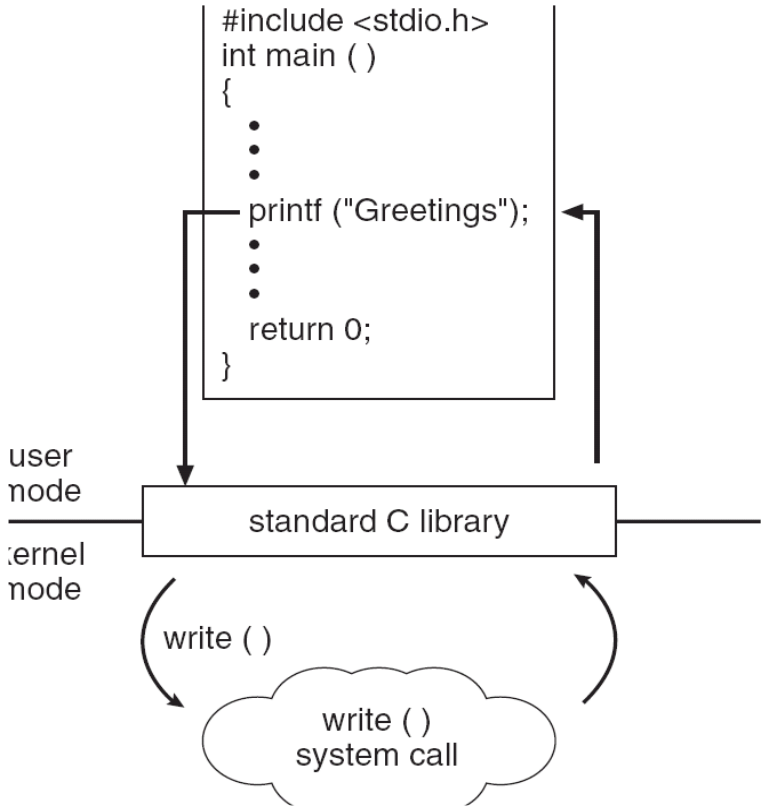
E.g. Copying a file from source to destination



System Call and OS Relationship



C Example



System Call Categories

Process control

(fork(),exit(),wait()

- load, execute, end, abort
- create process, terminate process
- get and set process
- allocate and free memory

File management

Read(),write()

- create file, delete file, open, close file
- read, write

Device management

Read(),write()

- request device, release device
- read, write
- get device attributes, set device attributes

Communications

Pipe(),shmget()

- send, receive messages
- transfer status information

Protection and Security

Chmod(),chown()

- Grant permissions
- Change ownership

Information maintenance

Getpid(),sleep()

- get time or date, set time or date
- get system data, set system data

- Hardware Control Layer/Block does communication with control logic block i.e. controller of a hardware.

 # System Calls: are the functions defined in a C, C++ & Assembly languages, which provides interface of services made available by the kernel for the user (programmer user).

- If programmers want to use kernel services in their programs, it can be called directly through system calls or indirectly through set of libary functions provided by that programming language.

- There are 6 catagories of system calls:

1. **Process Control System Calls:** e.g. fork(), _exit(), wait() etc...
2. **File Operations System Calls:** e.g. open(), read(), write(), close() etc...
3. **Device Control System Calls:** e.g. open(), read(), write(), ioctl() etc...

4. Accounting Information System Calls: e.g. getpid(), getppid(), stat() etc...

5. Protection & Security System Calls: e.g. chmod(), chown() etc...

6. Inter Process Communication System Calls: e.g. pipe(), signal(), msgget() etc...

- In UNIX 64 system calls are there.

- In Linux more than 300 system calls are there

- In Windows more than 3000 system calls are there

- When system call gets called the CPU switched from user defined code to system defined code, and hence system calls are also called as **software interrupts/trap.**

 # Dual Mode Operation:

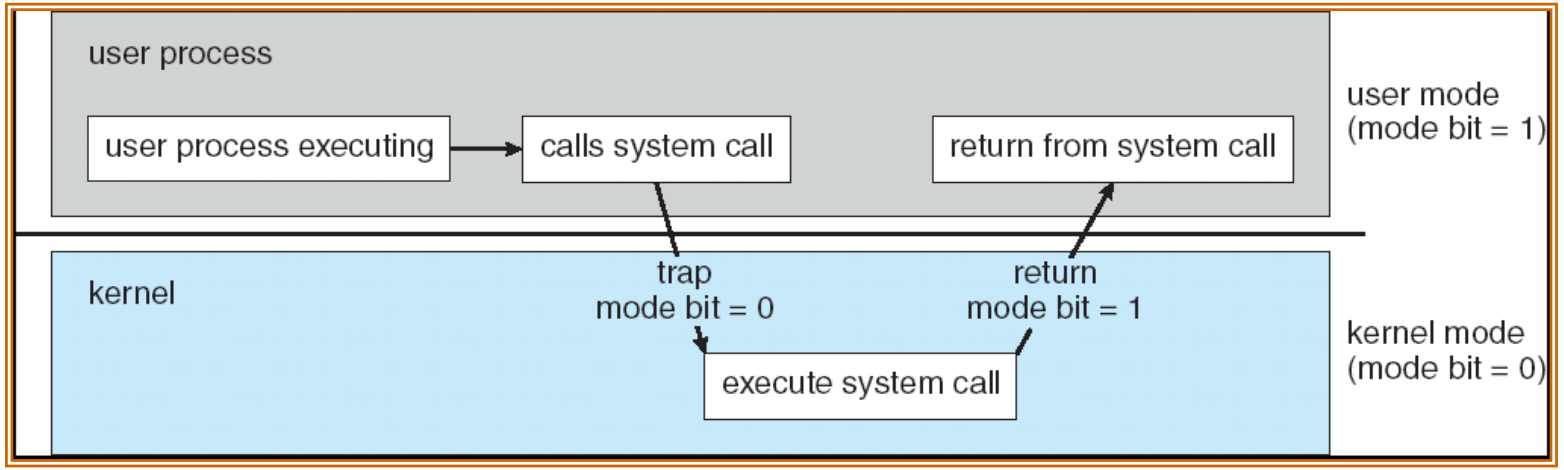
- System runs in two modes: System Mode and User Mode

1. System Mode:

- When the CPU executes system defined code instructions, system runs in a system mode.
- System mode is also reffered as kernel mode/monitor mode/supervisor mode/ privileged mode.

2. User Mode:

- When the CPU executes user defined code instructions, system runs in a user mode.
- User mode is also reffered as non-priviledged mode.
- Throughout execution, the CPU keeps switch between kernel mode and user mode



Dual Mode Operation:

- Throughout an execution of any program, the CPU keeps switcesh in between kernel mode and user mode and hence system runs in two modes, it is reffered as **dual mode operation.**

- To differentiate between user mode and kernel mode one bit is there onto the CPU which is maintained by an OS, called as **mode bit,** by which the CPU identifies wheather currently executing instruction is of either system defined code instruction/s or user defined code instruction/s.

- In Kernel mode value of **mode bit = 0**, whereas

- In User mode **mode bit = 1.**

Modes

System Mode/ Kernel Mode

privileged mode/master mode/supervisor mode

- Can execute any instruction
- Can access any memory locations, e.g., accessing hardware devices,
- Can enable and disable interrupts
- Can change privileged processor state
- Can access memory management units
- Can modify registers for various descriptor tables

User Mode

Unprivileged Mode

- Access to memory is limited,
- Cannot execute some instructions
- Cannot disable interrupts,
- Cannot change arbitrary processor state,
- Cannot access memory management units

Transition from user mode to system mode must be done through well defined call gates (system calls)

Kernel space vs User space

Part of the OS runs in the kernel model

known as the OS kernel

Other parts of the OS run in the user mode, including service programs , user applications, etc.

they run as processes

they form the user space (or the user land)