Linux Versus Other Unix-Like Kernels

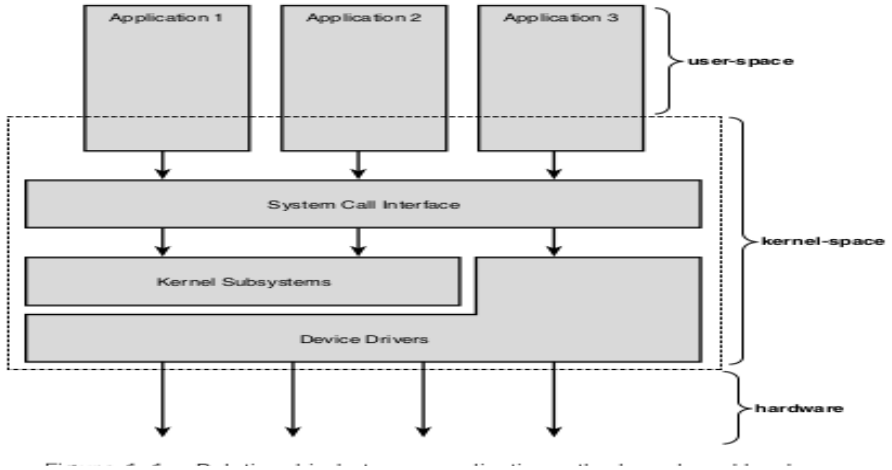
1. **Free and Open Source:** Linux is free, while some Unix-like systems may have costs.
2. **Customizable:** Linux is highly customizable due to its open-source nature.
3. **Runs on Older Hardware:** Linux works well on older, low-end hardware.
4. **Efficient:** Linux prioritizes efficiency and performance.
5. **Strong Community Support:** Linux has active community support for updates and solutions.

Multiuser Systems:
- Multiuser systems allow multiple users to run applications concurrently.
- Key features include user authentication, protection against buggy and malicious programs, resource allocation limits, and hardware protection.
- Unix is an example of a multiuser system that enforces hardware resource protection.

Users and Groups:
- In multiuser systems, each user has a private space with storage and mail.
- User access to private space must be restricted to the owner.
- Users are identified by a UserID (UID) and can belong to one or more groups identified by a (gid)
- The root or superuser account is used for system administration.

Processes:
- Processes are fundamental in operating systems and represent running programs.
- A process can be seen as an instance of a program in execution.
- Processes execute instructions in an address space, which is the memory they can access.
- Modern systems allow multiple execution flows within a single process.
- Concurrent active processes are known as multiprogramming or multiprocessing.

Kernel Architecture



Microkernel Operating Systems:
- Microkernel OSs have a small kernel with essential functions.
- System processes implement other OS functions like memory allocation and device drivers
- Easily portable to different hardware architectures.
- Efficient memory usage by swapping out or destroying unneeded system processes

Hard Links:
- A filename included in a directory is called a hard link, or simply a link.
- The same file can have multiple hard links, meaning it has several filenames.

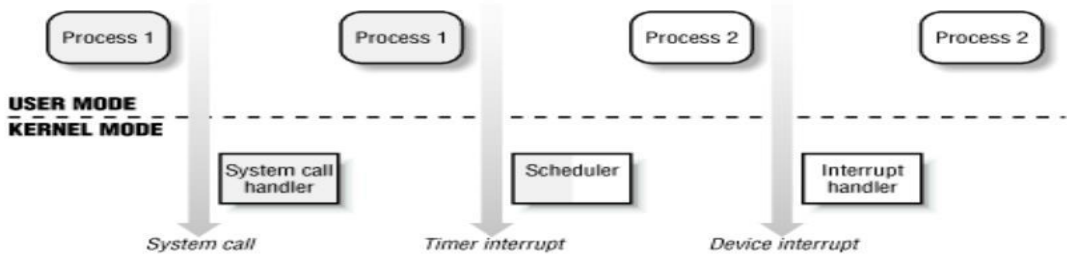
- You create a hard link using the `ln` command, like `$ ln f1 f2`.
- Hard links have two limitations: you can't create hard links for directories, and they must be within the same filesystem.

Soft Links (Symbolic Links):
- To overcome the limitations of hard links, symbolic links (soft links) were introduced.
- Symbolic links are short files that contain the pathname of another file.
- They can refer to files located in different filesystems.
- You create a symbolic link using the `ln -s` command, like `$ ln -s f1 f2`.

hard links are multiple names for the same file within the same filesystem, while symbolic links are shortcuts that can point to files in different filesystems.

- **File Descriptors:** These are numbers assigned to open files by the operating system. They help processes read, write, and manage file positions.
- **Inodes:** These are data structures used to store information about a file, like owner, permissions, and data block locations.
- **File Handling System Calls:** Processes open files using `open()`, read/write data with `read()` and `write()`, change file positions with `lseek()`, and close files with `close()`.

- **Process/Kernel Mode:** CPUs can run in either User Mode or Kernel Mode.
- User Mode: Programs cannot directly access kernel data or programs.
- Kernel Mode: Programs have full access to kernel resources.
- Special instructions allow switching between User and Kernel Modes.
- Most of the time, programs run in User Mode and switch to Kernel Mode only when they need a service from the kernel.



- **Processes:** Processes are dynamic entities managed by the kernel.
- The kernel manages processes but is not a process itself.
- Processes use system calls to request kernel services.
- System calls set up parameters and switch the CPU from User to Kernel Mode.
- Kernel threads are privileged, run in Kernel Mode, and handle various tasks like exceptions, interrupts, and I/O operations.

In summary, processes in User Mode request kernel services using system calls, which switch the CPU to Kernel Mode for servicing. Kernel threads run in Kernel Mode to handle various tasks in the system.