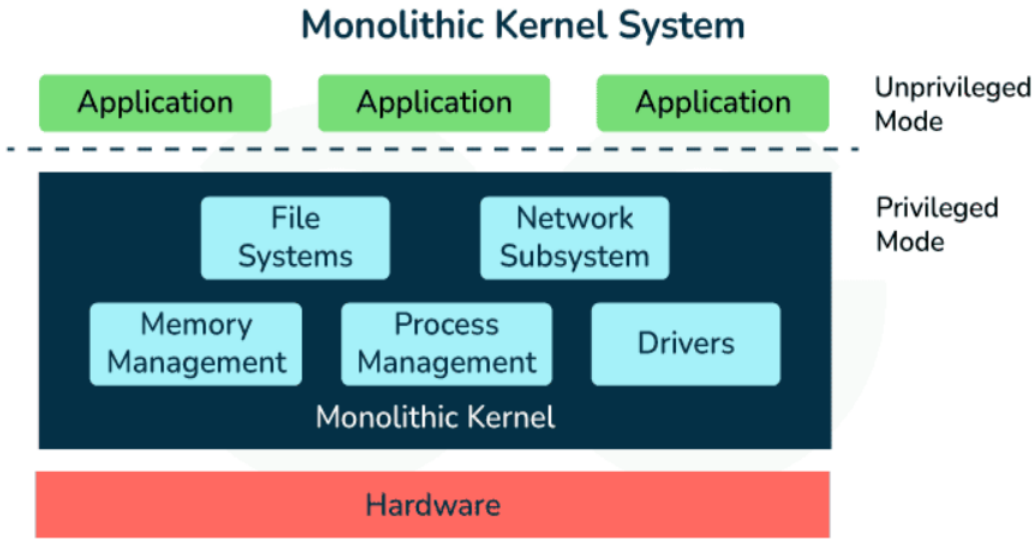
1.2 SYSTEM ARCHITECTURE, KERNEL VS USER SPACE

MONOLITHIC VS MICROKERNEL

- The kernel is the core component of an operating system (OS) responsible for managing system resources and enabling communication between hardware and software.
- Based on how these services and components are organized, kernels are broadly classified into two main types - Monolithic Kernel and Microkernel.

Monolithic Kernel

- A monolithic kernel is an operating system kernel in which all the operating system services run in kernel space, meaning they all share the same memory space.
- This type of kernel is characterized by its tight integration of system services and its high performance.



Advantages of a Monolithic Kernel

- It provides important OS functions like CPU scheduling, memory management, and file management through system calls.
- It runs as one large process in a single address space, which makes it faster.
- It is a single static binary file.
- Examples include Unix, Linux, Open VMS, and z/TPF.
- No complex inter-process communication is needed, so system calls run faster.

Disadvantages of a Monolithic Kernel

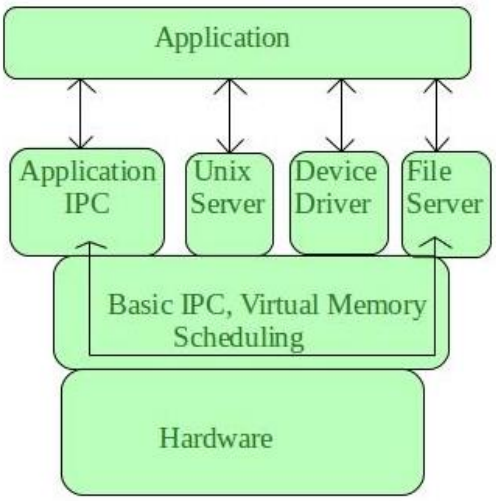
- If one service fails, the whole system may crash.
- It is not modular, so adding new features requires changing the whole system.
- A bug in any part can affect the entire system because everything runs in kernel mode.
- The kernel becomes large and complex as more services are added.

Microkernel

- A microkernel is a type of operating system kernel in which only the most basic services run in kernel space, with other services running in user space.
- This type of kernel is characterized by its modularity, simplicity, and ability to run multiple operating systems on the same hardware.

The microkernel itself typically includes only the most fundamental services, such as:

- **Inter-process Communication (IPC):** Mechanisms for processes to communicate and synchronize with each other.
- **Basic Scheduling:** Managing the execution of processes.
- **Minimal Memory Management:** Essential functions for memory allocation and protection.



Other functionalities that are often part of a monolithic kernel, like device drivers, file systems, and network protocols, are implemented in user space as separate processes. This contrasts with a monolithic kernel, where all these services run in kernel space.

Advantages of Microkernel

- Modularity
- Performance
- Security
- Reliability
- Scalability
- Portability

Disadvantages of a Microkernel

- Slower Message Passing
- More Complex
- Limited Performance

Differences Between Monolithic Kernel and Microkernel

Basics	**Micro Kernel**	**Monolithic Kernel**
Size	Smaller	Larger as OS and both user & OS lie in the same address space.
Execution	Slower	Faster
Extendible	Easily extendible	Complex to extend
Security	If the service crashes then there is no effect on working on the microkernel.	If the process/service crashes, the whole system crashes as both user and OS were in the same address space.

Basics	Micro Kernel	Monolithic Kernel
Code	More code is required to write a microkernel.	Less code is required to write a monolithic kernel.
Examples	L4Linux, macOS	Windows, Linux BSD

Kernel Space vs User Space

- In an operating system, there are two main areas where code runs: user space and kernel space.
- *User space is where user applications run, while kernel space is where the operating system and other important parts run.*
- In kernel space, code can directly access system resources like memory and hardware, allowing it to perform special tasks that user space code can't.
- System calls are important for connecting user space and kernel space. When an application makes a system call, it switches from user space to kernel space, allowing the kernel to do what the application requested.