LONDON METROPOLITAN UNIVERSITY

informatics college • pokhara

Module Code & Module Title

CT5052 Network Operating Systems

Year and Semester

2023-24 Spring

Module leader: **Prashant Adhikari**

Student Name: Dipen Khatri

London Met ID: 23056968

Assignment Submission Date: 30th, September

I confirm that I understand my coursework needs to be submitted online via my secondary teacher under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

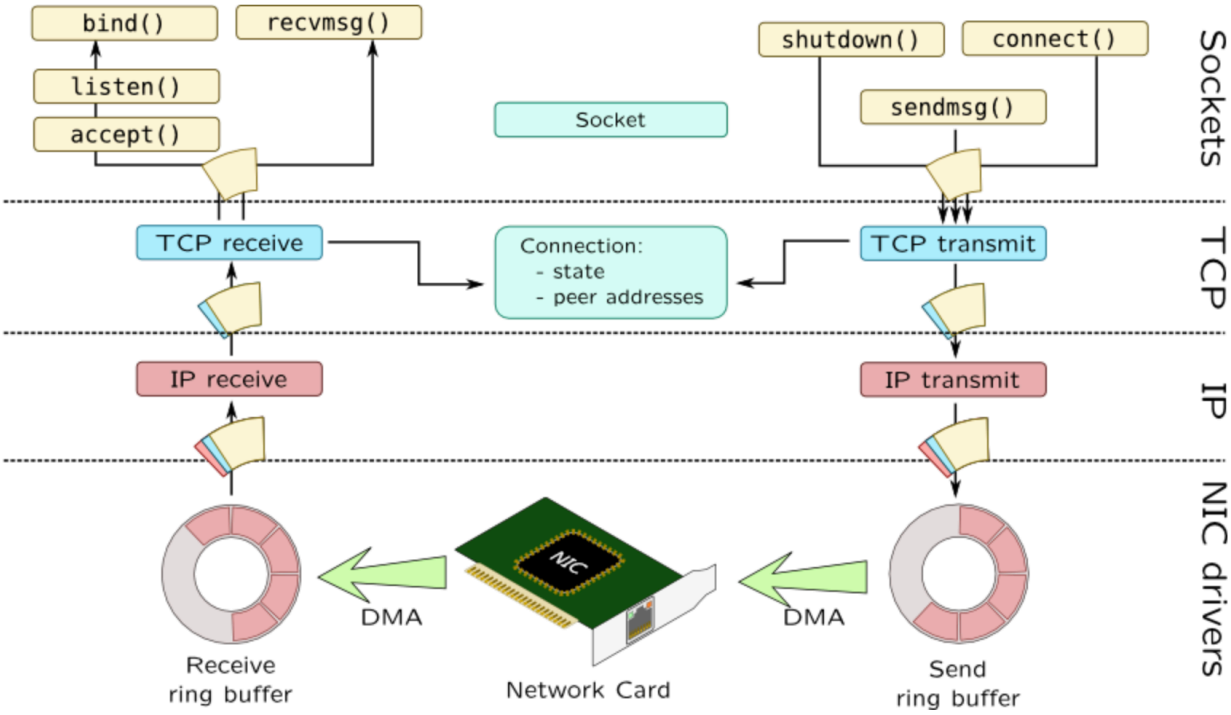
Contents

1. Introduction

The Kernel: The Heart of an Operating System

An essential component of every operating system (OS) is the kernel, which serves as the hub for managing software and hardware resources. By serving as a bridge, it makes sure that user programs and the underlying hardware are in constant communication. An operating system wouldn't work without the kernel since it handles essential functions including memory allocation, device management, file system control, process scheduling, and system security. Because of its intricate functioning, the kernel serves as the brains of the operating system, coordinating the efforts of all its parts to maintain seamless operation.

To put it simply, the kernel can be thought of as the computer system's brain. It oversees all significant background tasks, guaranteeing that user apps communicate with hardware such as the CPU, memory, and input/output devices in an efficient manner. The kernel plays a critical role in system security, stability, and performance even though consumers may not be able to see it directly.



2. Goal

This report's main goal is to examine the idea of the operating system kernel, its different varieties, its function in popular operating systems, and its role during system boot. Understanding the architecture and operations of the kernel helps us to better understand how operating systems handle the intricate interplay between hardware and software.

Key Functions of the Kernel

The kernel is responsible for several key functions that are vital to the smooth running of the computer system. These include:

1. Process Management

Process management, or managing all of the active programs on the system, is one of the kernel's most significant duties. Each running program is called a "process," and the CPU time allotted to each process is decided by the kernel. It accomplishes this by using a feature called multitasking, in which the kernel distributes CPU time among processes to create the illusion that several applications are open at once.

The kernel uses a process scheduler to allot CPU time to various applications. The scheduler makes sure that activities may go on without interfering with one another, including online browsing, background updates, and music listening. The kernel balances the burden and maintains effective system performance by rapidly switching between processes.

2. Memory Management

One further important function of the kernel is memory management. Every application that you open needs a specific amount of RAM to function. The kernel manages the distribution of this memory across various programs and makes sure that each one has a dedicated area. Additionally, it guards against other programs overwriting the memory areas, which can result in a crash or corrupt data to increase the amount of memory accessible beyond the actual RAM, the kernel additionally makes use of virtual memory. The kernel transfers some data temporarilyto the hard drive (or solid-state drive) if the system runs out of RAM, enabling the system to function normally even in situations where memory is limited.

3. Device Management

Device management is yet another crucial task carried out by the kernel. Keyboards, printers, USB drives, network cards, and monitors are just a few examples of the hardware that computers may communicate with. Device drivers, which are specialized software intended to help with hardware and operating system communication, are how the kernel interacts with various devices.

Programs can use devices without having to comprehend how they operate inside thanks to the kernel's abstraction of the complexity of hardware interaction. For instance, the application you're using doesn't need to understand the specifics of the printer's operation in order to print a page; the kernel manages communication with the relevant driver. One of the main explanations for why new hardware can be plugged in and expected to function with the operating system without any issues is this abstraction.

4. File System Management

File systems, which let users and programs store and retrieve data from disks or other storage devices, are managed by the kernel. It guarantees that programs may read, write, and edit files as needed and arranges data in an ordered manner, usually using directories and files.

The kernel tracks the file's position and decides where on the disk the data should go when a software saves a file. Additionally, it guards against program overwrites and prevents illegal access to specific file system sections.

5. Security and Access Control

Keeping the operating system secure is one of the kernel's most crucial responsibilities. What users and programs are permitted to do is determined by the access control policies it imposes. For instance, the kernel makes sure that an ordinary user cannot access another user's data without authorization or change important system files.

In order to guarantee that only authorized users are able to access the system, the kernel also controls user authentication. The kernel assists in defending the system from viruses, malware, and illegal access by overseeing these security settings.

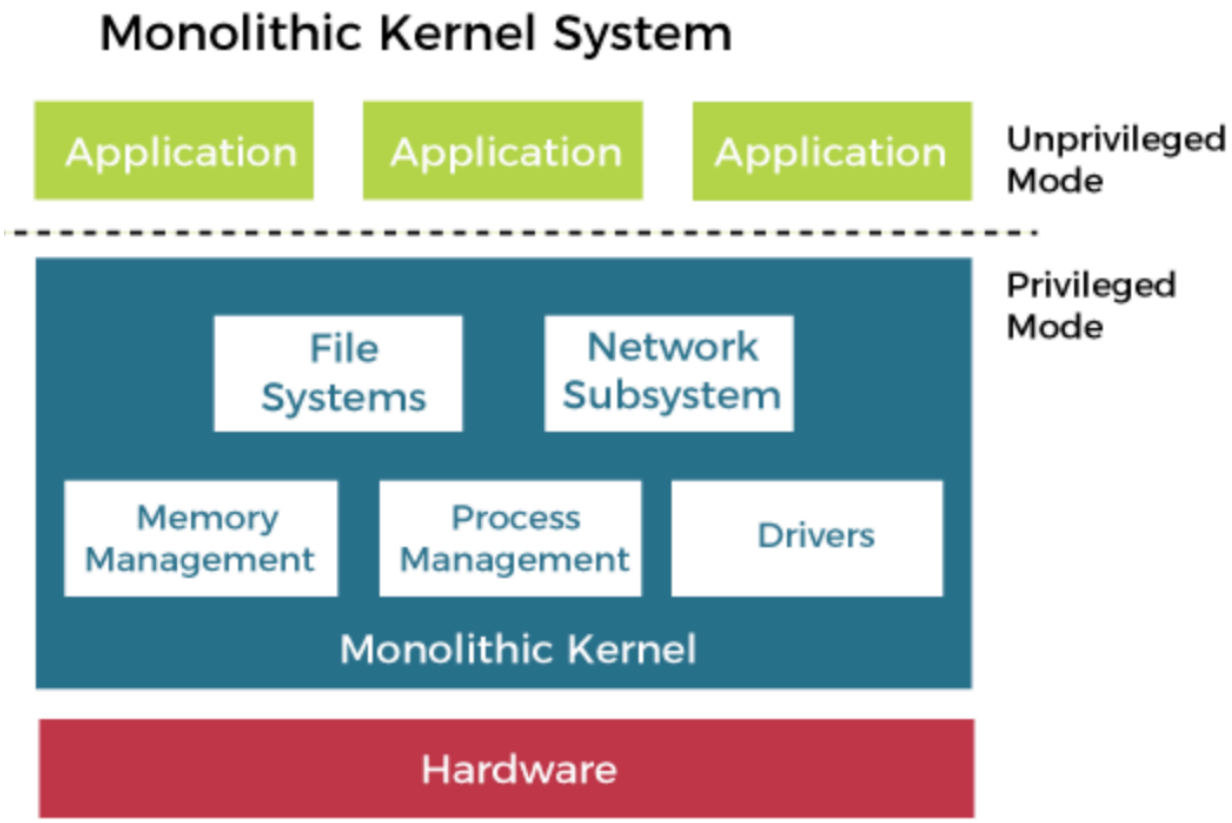
Types of Kernels

There are various varieties of kernels, each with a unique structure and approach to managing system operations. There are three primary kinds of kernels:

1. Monolithic Kernel

All of the essential operating system functions, including memory management, file system administration, device drivers, and process scheduling, are contained within the kernel of a monolithic system. As a result, the same address space is used by a single, sizable block of code. Monolithic kernels are usually faster than other types since there is no context switching between user mode and kernel mode because everything runs in kernel mode.

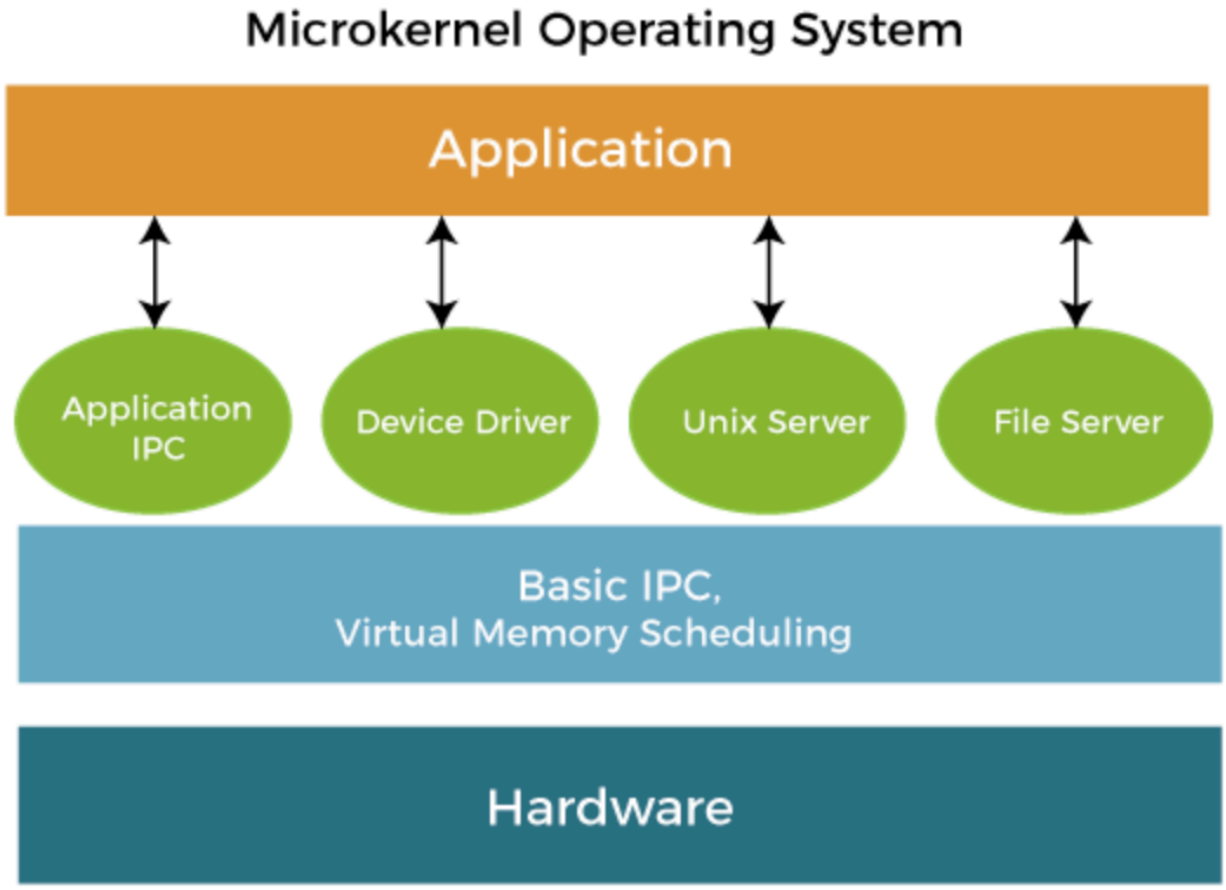
The drawback of monolithic kernels is that they can be more difficult to maintain and less secure. A kernel bug has the power to take down the entire system. Unix and Linux are two instances of operating systems that have a monolithic kernel.



2. Microkernel

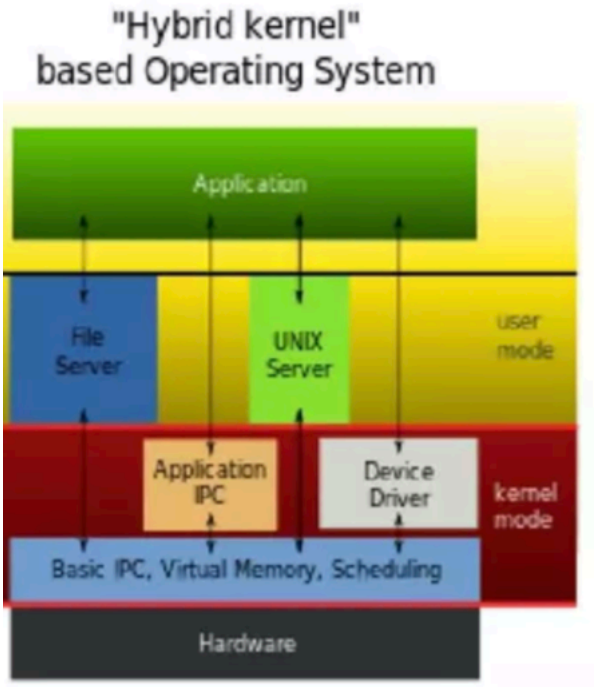
A microkernel adopts a more austere strategy, including only the most necessary features in the kernel, like basic process scheduling and inter-process communication. Device drivers and file systems are among the other functions that are relocated to user space and operate as independent processes there. Because adding or updating services doesn't involve changing the kernel as a whole, microkernels are more secure and simpler to extend or modify thanks to their modular design.

Microkernels can be slower due to greater communication between the kernel and user space services, although offering increased security and modularity. Microkernels are found in some operating systems, such as Mach and Minix.



3. Hybrid Kernel

The goal of a hybrid kernel is to bring together the advantages of microkernels and monolithic kernels. For performance, it places necessary services like device drivers in the kernel area; but, for modularity, it also permits certain services to operate in user space. The goal of hybrid kernels is to provide a balance between flexibility and speed. Windows NT and macOS are two examples of operating systems with hybrid kernels.



Popular Kernels and Their History

Kernel for iOS

The XNU (X is Not Unix) kernel, which blends parts of the Mach microkernel with BSD Unix components, serves as the foundation for the iOS kernel. XNU was created by Apple and debuted alongside NeXT STEP, a Unix-like operating system created by NeXT (which Apple eventually purchased) in the late 1980s. Through its growth into the iOS kernel, XNU has become highly tuned for mobile devices, striking a balance between energy efficiency and performance.

Kernel for Windows

As part of the initial release of Windows NT in 1993, the hybrid Windows NT kernel was unveiled. It was made with portability and compatibility for many hardware architectures in mind. Compared to previous Windows systems' DOS-based kernels, Windows NT represented a major change. The NT kernel's dominance in the desktop OS industry can be attributed to its optimization over time to support a broad variety of applications and retain backward compatibility with older software.

Kernel for Ubuntu

Ubuntu, a Linux-based operating system, makes use of the monolithic Linux kernel. Since its original development by Linus Torvalds in 1991, Linux has become one of the most popular kernels, particularly for embedded and server computers. Since its 2004 debut, the Ubuntu distribution has been a major factor in the widespread adoption of the Linux kernel in personal computing and cloud infrastructure.

Boot Process

In order to ensure that the system is prepared for user input, the kernel is essential to the boot process.

Steps in the Boot Process:

BIOS/UEFI Initialization: The firmware sets up the hardware and searches for a device that can be booted.

Boot Loader: After being loaded into memory, the boot loader—GRUB in Linux and BOOTMGR in Windows—chooses which kernel to load.

Kernel loading involves initializing device drivers and memory management as well as loading the chosen kernel into memory.

Kernel Initialization: In this phase, the kernel loads drivers, configures memory areas, and launches the first user-space process (init in Unix/Linux).

User-space Initialization: The user interface (desktop environment or terminal) is loaded and the system is made operational.

These procedures, which are described in Tanenbaum's Modern Operating Systems, make guarantee that the system can move from a state of bare metal to one in which applications may operate.

Conclusion

To sum up, the kernel is an operating system's central component. It oversees vital operations like file systems, security, device communication, process and memory management, and file systems. The trade-offs between performance, modularity, and security are different for the three types of kernels: hybrid, microkernel, and monolithic. The kernel is a crucial part of both simple and complex computer systems because of its roles in resource management, security, and system stability.