COMPARISON AMONG LINUX,WINDOWS AND MAC OS

EIham musaa
Refan Alziyadi
Shaima Alzubaidi

COMPARISON TOPICS :

Processes&threads
CPU scheduling
Memory management
File system management
Sicurity & protection

WINDOWS

Introduction

The Windows operating system (OS) is a fundamental software system created by Microsoft. Initially launched in 1985 as Windows 1.0, it has evolved into one of the most widely used operating systems worldwide. Windows OS is designed to run on various hardware, including desktops, laptops, servers, and mobile devices. Its graphical user interface (GUI) allows users to interact with the computer using visual elements like buttons and icons, making it more user-friendly compared to text-based systems. Over the decades, Windows has become the go-to OS for personal and professional computing, maintaining a dominant presence in the market.

Processes & Threads

Windows processes consist of one or more threads, with threads being the basic executable unit in the system. The OS allocates system resources to these threads based on factors like CPU availability, memory, and fairness. Windows' support for multiprocessor systems allowsfor parallel execution across multiple processo
From a programmer's perspective, each process includes

- A virtual address space distinct from other processes
- Code segments (including DLLs) and data segments (globalvariables)
- Environment strings, heaps, and open handles.

Threads in a process share the code, global variables, and resources but maintain independent scheduling.
Each thread includes:

- A stack for procedure calls.
- Thread Local Storage (TLS) for managing unique data environments.
- An argument stack from the creating thread.
- A context structure maintained by the kernel.

Windows supports both user threads (managed at the application level) and kernel threads (managed by the OS kernel).

CPU Scheduling

Windows uses a priority-driven, preemptive scheduling system. The highest-priority runnable thread gets CPU time unless limited by processor affinity (threads assigned to specific processors). This ensures efficient multitasking and optimal use of system resources.

Synchronization

Windows handles task-to-task communication and synchronization within computers and across networks. The OS offers a comprehensive set of synchronization mechanisms that allow threads and processes to operate safely without conflicts, ensuring efficient communication between networked systems.

Memory Management

Windows employs a large, virtual memory address space that transparently moves data between physical memory and secondary storage like disk drives. This process ensures efficient use of memory, even when dealing with large files and applications.

File Systems

Windows supports multiple file systems, with NTFS (New Technology File System) being the most common. NTFS provides features such as security, fault tolerance, encryption, and support for large files. Other supported file systems include:

1. FAT/FAT32: Legacy file systems primarily used for removable storage like USB drives and memory cards.
2. CDFS: Used for CD-ROMs, compliant with ISO 9660 standards.
3. UDF: Supports DVD drives and other optical media. Windows also supports distributed file systems, including Network File System (NFS) and Common Internet File System (CIFS), and extensive storage area network (SAN) support for modern server environments.

I/O Systems

The Windows I/O system abstracts physical and virtual devices, ensuring consistency in application interfaces across different hardware platforms. Key features include:

- Plug and Play (PnP) for automatic hardware detection.
- Power management for optimizing energy use.
- High-performance asynchronous I/O to handle multiple requests.
- Extensibility with support for dynamic driver loading and unloading. The I/O manager coordinates between devices and applications,ensuring smooth and scalable operation across systems.

Security & Protection

Windows provides robust security mechanisms, including file protection attributes and security flags. Kernel objects, such as open files, require strict security measures to prevent unauthorized access. Windows also allows security attributes to protect shared resources and maintain system integrity.

MAC OS

Introduction

Mac OS is the operating system used by Apple's Mac computers. Think of it as the software that manages your computer's hardware and helps run your applications.

- *User Interface: Mac OS is known for its clean and user-friendly interface. It uses a system called Finder to help you organize and find your files easily.*
- *Integration: It works seamlessly with other Apple products like iPhones and iPads. For example, you can start an email on your Mac and finish it on your iPhone.*
- *Stability and Security: Mac OS is designed to be stable and secure. It gets regular updates to fix issues and protect against threats.*
- *Applications: Mac OS comes with useful built-in apps like Safari for browsing the web, Mail for emails, and iMovie for video editing.*

1. Processes & Threads

Processes: A process is like a big container that holds everything a program needs to run, such as code and data. In Mac OS, each application you open runs in its own process. For example, if you open Safari and Word, Safari and Word each run in their own process.

Threads: Threads are smaller parts of a process. They handle different tasks within a single process. For instance, Safari might have one thread for loading web pages and another for checking emails. Threads within the same process share resources but work on separate tasks.

2. CPU Scheduling

CPU Scheduling: This is how the system decides which process or thread gets to use the CPU (the brain of the computer) and for how long. Mac OS uses a system called "priority-based scheduling." This means that the CPU will prioritize processes and threads based on their importance and needs.

- Time-Slicing: The CPU gives each process or thread a little bit of time (a slice) to run before switching to the next one. This happens very quickly, so it feels like everything is running at once.

3. Memory Management

Memory Management: This is how the operating system handles the computer's memory (RAM). Mac OS uses a method called "virtual memory."

○ Virtual Memory: This lets the system use more memory than is physically available by using a part of the storage drive as temporary memory. So if you have a lot of apps open, Mac OS can still manage them by using this virtual memory.

○ Paging: When the physical memory gets full, Mac OS moves some data to the storage drive. This process is called paging. It helps in managing the memory more efficiently.

4. I/O Systems

I/O Systems: Input/Output (I/O) systems handle data exchange between the computer and external devices like keyboards, mice, and printers.

○ Device Drivers: Mac OS uses special software called device drivers to communicate with hardware. These drivers translate the data from the hardware into a format the computer can understand.

○ File System: Mac OS organizes data on storage drives using a file system. The most common file system in Mac OS is called APFS (Apple File System). It helps in storing, organizing, and accessing files efficiently.

5. Security and Protection

Security: Mac OS includes several layers of security to protect your data and keep malicious software away.

○ Gatekeeper: This feature checks apps to make sure they are from trusted sources before allowing them to run. If an app is not verified, Gatekeeper will block it.

○ FileVault: This is a tool that encrypts your entire hard drive. Encryption means that your data is turned into a code that only you can unlock with your password.

○ Sandboxing: Each app runs in its own sandbox, which is like a secure, isolated space. This prevents apps from accessing or modifying data from other apps or system files.

○ Regular Updates: Mac OS gets regular updates to fix security holes and protect against new threats. Keeping your system up-to-date is important for staying secure.

LINUX

Introduction

Linux is an open-source kernel-based operating system. Linux provides users with a free and
open environment that can be modified and customized to suit different needs.
Today, Linux is
one of the most popular operating systems, especially in servers, supercomputing systems,
and embedded systems.

1. Processes & Threads

- Processes: A process is an instance of a running program, with its own memory and resources. Each process is uniquely identified by a Process ID (PID). Processes can be in different states (running, sleeping, stopped, etc.).
- Threads: Threads are smaller units of a process that share the same memory but can execute different tasks simultaneously. Linux uses POSIX threads (pthreads) for multithreading, allowing multiple threads within a process to run in parallel.
- Process Control: Linux manages processes using system calls like `fork()` (to create new processes), `exec()` (to run a new program in a process), `wait()` (to wait for process termination) and `kill()` (to send signals to processes). Processes follow a parent-child hierarchy.

2. CPU Scheduling

- Schedule : The Linux kernel uses the Completely Fair Scheduler (CFS) to allocate CPU time, ensuring all processes get fair access to CPU resources while keeping the system responsive.
- Scheduling Classes: Linux differentiates between real-time and normal processes, with real-time tasks having higher priority.
- Context Switching: The kernel switches between processes based on scheduling algorithms, allowing multiple processes to run by alternating CPU time.

3. Memory Management

- Virtual Memory: Linux uses virtual memory, enabling processes to use more memory than physically available. Pages are moved between RAM and disk as needed, providing efficient memory use.
- Paging: Memory is divided into pages, which are loaded only when needed (demand paging). This optimizes memory usage.
- Swapping: When RAM is full, Linux can swap inactive pages to disk, freeing memory for active processes.
- Memory Allocation: Processes request memory dynamically using functions like `malloc()`, and the kernel handles memory allocation to ensure efficient use and avoid fragmentation.

4. I/O Systems

- Virtual Memory: Linux uses virtual memory, enabling processes to use more memory than physically available. Pages are moved between RAM and disk as needed, providing efficient memory use.
- Paging: Memory is divided into pages, which are loaded only when needed (demand paging). This optimizes memory usage.
- Swapping: When RAM is full, Linux can swap inactive pages to disk, freeing memory for active processes.
- Memory Allocation: Processes request memory dynamically using functions like `malloc()`, and the kernel handles memory allocation to ensure efficient use and avoid fragmentation.

5. Security and Protection

- User Privileges: Linux uses a permission model that assigns ownership of files and processes
to users and groups. Permissions for reading, writing, and executing are set for the owner, group,
and others.
- SetUID and SetGID: These special permissions allow users to run files with the file owner’s or
group’s privileges, enabling temporary elevated access when needed.
- Access Control Lists (ACLs): ACLs provide more detailed control over file permissions,
allowing multiple users or groups to have different access levels.
- SELinux: A security module that enforces strict access controls through policies, preventing
unauthorized actions by processes.
- Encryption: Linux supports encryption at multiple levels, including file and disk encryption, as
well as network-level security through protocols like SSL/TLS.

Best in	Why
Process and Thread Management: Linux	Linux efficiently handles multiple processes and threads, making it great for multitasking and complex systems.
CPU Scheduling: Linux	The Completely Fair Scheduler (CFS) ensures fair CPU usage for all processes, making it ideal for systems with strict timing needs.
File System Management: Linux	Linux supports fast and secure file systems like Ext4 with advanced access control.
Security: Linux	Linux provides top-tier security with SELinux, strong file permissions, and encryption, making it a preferred choice for servers.
Memory Management: Linux	Linux uses virtual memory and paging to manage memory efficiently, even under heavy loads.

In summary, Linux is ideal for developers, while Windows suits regular users better.

Reference

[1] Goloboff, P. A., & Morales, M. E. (2023). TNT version 1.6, with a graphical interface for MacOS and Linux, including new routines in parallel. Cladistics, 39(2), 144-153.

[2] Adekotujo, Akinlolu, et al. "A comparative study of operating systems: Case of windows, unix, linux, mac, android and ios." International Journal of Computer Applications 176.39 (2020): 16-23.

[3] Chen, Weiteng, et al. "Syzgen: Automated generation of syscall specification of closed-source macos drivers." Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security. 2021.

[4] Johnson M. Hart, Windows System Programming - Google Books.

[5] Understanding the Windows I/O System - Microsoft Press Store.

[6] CPU Scheduling in Windows Operating System - Studocu.

THANK YOU FOR LISTENING