Team Project-Case Study

Computer Operating Systems

Professor. Furht

Linux vs. Windows

Florida Atlantic University
By: Mackenzie Falla, Kadeem Mighty & Esaie Joseph
2023

Table of Contents

Introduction

An operating system (OS) is a crucial software that enables a computer to perform essential functions such as task scheduling, application execution, and peripheral control. In this report we will compare two distinct operating systems, exploring their functionalities and unique characteristics.

Linux and Windows have their pros and cons just like every other operating system. The choice of operating system depends on the organizational architecture and objectives. When it comes to organizational architecture and the overall capabilities of operating systems, they can be tailored to fulfill similar tasks. The key lies in determining which operating system excels in a particular task.

Linux, being an open-source operating system, offers several advantages. It does not require licenses or frequent upgrades, and it is known for its robust security, making it less vulnerable to breaches. Linux is known for its stability, with a lower likelihood of crashes and system halts compared to Windows. It is widely used as a platform for hosting various servers, including network servers, web servers, and database servers. Once Linux is installed, it can be run for extended periods of time, even decades.

Windows is a closed-source operating system that requires licensing for proper functionality. It is susceptible to cyber-attacks and security breaches. In Windows, there are four types of users: Administrators, Children, Guests, and Standards.

Windows is favored for its user-friendly interface and attractive Graphical User Interface (GUI), making it accessible to users with limited programming knowledge. However, Linux is commonly utilized in data centers and servers due to its stability and the ability to operate without frequent upgrades or restarts.

In terms of file organization, Windows stores files in directories or folders, whereas Linux utilizes a tree structure. One significant distinction between Linux and Windows is the accessibility of their source code. Linux allows organizations to modify the source code according to their needs, while Windows retains copyright protection on its source code.

Requirements of Linux & Windows:

One crucial decision to consider is the analysis of system requirements for utilizing Linux and Windows. The optimal utilization of system resources stands as a primary objective for any organization. Each organization strives to effectively utilize the available pool of resources to accomplish both short-term and long-term objectives. Below, you will find a concise description of the requirements for using Linux and Windows.

Hardware:

Hardware is a crucial asset for organizations, often more expensive than software. The efficient utilization of hardware is a significant factor that motivates organizations to choose Linux from a range of operating systems. Unlike Windows, Linux does not necessitate frequent hardware updates to keep up with evolving technology, making it compatible with older hardware. Despite this, Windows requires careful hardware selection due to its multitasking and multiprogramming capabilities, which rely on a seamless combination and synchronization between hardware and software.

Software:

Preprocessor:

Linux systems offer versatile processor support, including the 80386, 80486, Pentium, Pentium Pro, and even up to the Pentium IV. What's even more remarkable is that organizations can leverage non-intel architectures when using Linux, as it extends its compatibility to Alpha AXP, PowerPC, SPARC, MIPS, and Motorola architecture.

In contrast, Windows exposes mandatory processor upgrades for compatibility with modern versions, given that processor speed directly influences system performance. Intel's latest milestone is the 10th generation of Core i7 processors.

RAM (Random Access Memory):

Linux is known for its efficient memory usage, as it can run on as little as 8MB of RAM. For more resource-intensive task, a recommended minimum32MB of RAM is suggested. Nonetheless, Linux is also capable of supporting RAM capacities of 1GB or even higher, which sets it apart from other operating systems.

On the other hand, Windows has varying RAM requirements depending on its versions. Older versions such as Windows 98, Vista, and XP 2007 were optimized to run smoothly with just 512MB of RAM. With the advancements in Windows 8, 8.1, and 10, the RAM requirements have increased significantly, with a recommended minimum of 8GB for optimal performance.

ROM (Read Only Memory):

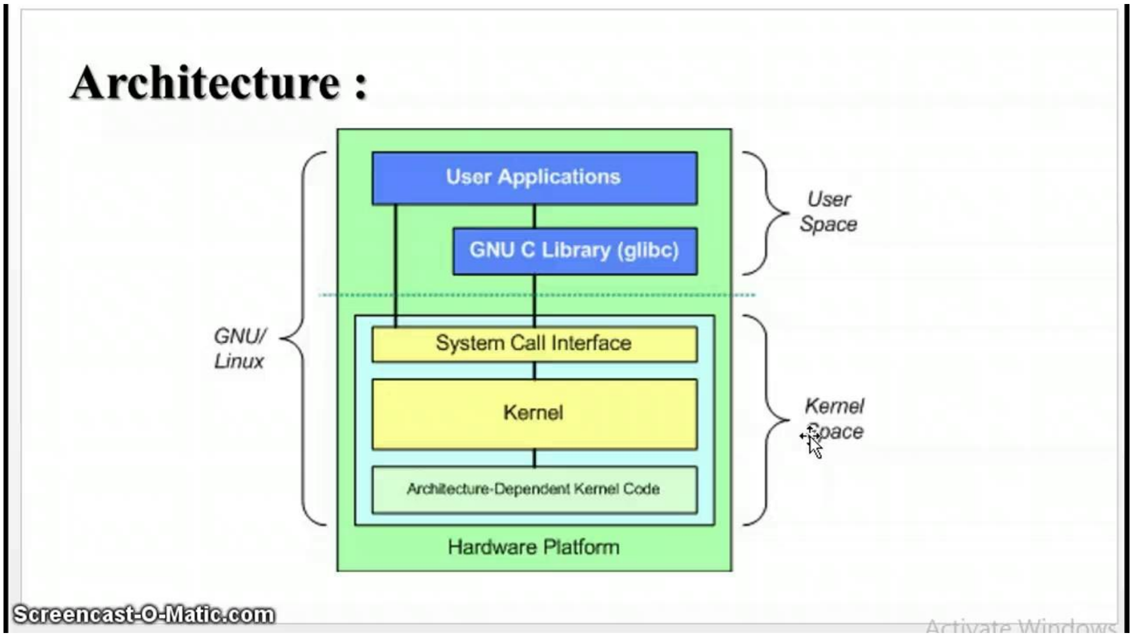
Linux systems can be booted from a floppy disk, providing a simple way to store data. When it comes to requirements for organizing data on a hard drive, organizations should select a SCSI (Small Computer System Interface) hard drive based on their specific needs. For demanding workloads and complex operations, a 1GB SCSI hard disk is recommended, although larger capacities can also be used. It is worth nothing that Linux optimizes the utilization of both hardware and software resources. Linux is capable of controlling various types of device controllers, including IDE, RLL, and MFM. On the other hand, Windows users typically store a wide range of data on their computers, necessitating larger disks.

Software Applications:

Windows possesses a broad-reaching capacity to install a wide array of software applications, thereby accommodating the diverse needs and preferences of its users. Its broad ecosystem encompasses a multitude of both proprietary and open-source applications, ensuring vast selection for users to choose from. The flexibility of the Windows operating system, combined with the support of a substantial developer community, enables users to seamlessly integrate and customize their computing experience.

Linux exhibits a more focused approach in terms of software availability. While it does include a selection of built-in applications and web browsers, its default installation is typically leaner in comparison to Windows. The Linux philosophy prioritizes user choice allowing individuals to configure their systems to meet their specific requirements. This approach may result in a relatively limited selection of pre-installed applications, requiring users to actively seek and install their preferred software components. Furthermore, Linux's software ecosystem leans heavily towards open-source and community-driven projects. Because of this it may lack certain proprietary applications commonly found in the Windows environment.

Operating Systems Architecture



Architecture :

User Applications

GNU C Library (glibc)

System Call Interface

Kernel

Architecture-Dependent Kernel Code

Hardware Platform

User Space

Kernel Space

GNU/ Linux

To gain a comprehensive insight into the performance of operating systems (OS), it is imperative to delve into the intricacies of its architecture. The architecture of an OS serves as a blueprint, revealing the intricate interplay of its constituent components and their collaborative orchestration when tasked with specific operations. Now we are shedding light on the inner workings and synergy of these pivotal structures.

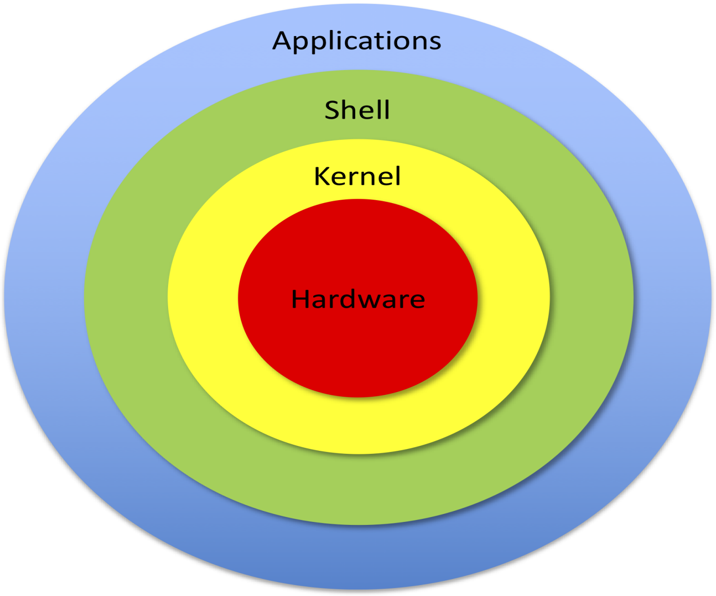
Linux Architecture:

Linux's architectural framework stands in stark contrast to that of Windows, distinguished by its elegant simplicity. This distinction is primarily rooted in the choice of a monolithic kernel as the foundation of the Linux operating system. Beneath the surface, the Linux architecture comprises several components, each contributing to its functionality and versatility. These integral elements include the kernel, system libraries, system utilities, and command shells.

Linux Kernel:

The Linux Kernel occupies a pivotal role as the core component of the operating system, serving as the linchpin that orchestrates the intricate synergy between hardware and software. Distinguished itself with a monolithic kernel architecture, Linux's kernel functions as both a resource manager and an allocation maestro, expertly mediating the interplay between the hardware and the OS.

This formidable kernel assumes the role of an essential abstraction layer, positioned deftly between application programs and the operating systems. In this capacity, it empowers applications to seamlessly interact with the OS, providing a bridge that enhances the interoperability and efficiency of the entire system.



System Libraries:

These are specialized programs through which application programs gain access to the kernel's features, without the need for access to the kernel module's source code.

System Utilities:

Utilities serve a specific purpose in the Linux operating system, enabling it to perform various specialized functions.

Shells:

Shells act as an interface between users and the Linux kernel, accepting commands from users and executing kernel functions. They come in different types, including the Bourne Shell, POSIX Shell, C Shell, and Korn Shell.

Windows Architecture:

The architecture of Windows is slightly more intricate compared to Linux. It consists of several components, including the Hardware Abstraction Layer (HAL), the Executive Services, Environmental Subsystems, and Kernel. The diagram below should provide a clear illustration:

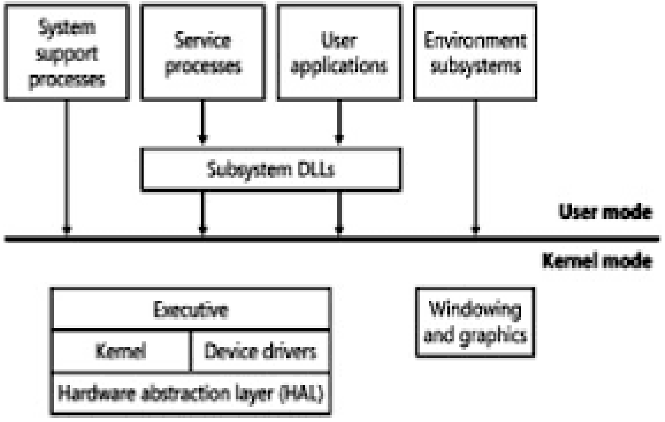


FIGURE 2-1 Simplified Windows architecture

Hardware Abstraction:

The Hardware Abstraction Layer (HAL) serves as a crucial intermediary between computer hardware and software. It operates in close proximity to the Windows Kernel, offering a foundational platform for the kernel to carry out essential functions like inter-process communication, interrupt control, and space management. It's important to note that if a program accesses the HAL, it prevents simultaneous access by other programs.

Kernel:

Similar to other operating systems, the kernel plays a vital role in Windows. It is responsible for scheduling CPU activities and works in conjunction with the microkernel to enhance the performance of the Windows operating system. Windows utilizes a Hybrid Kernel architecture, which further contributes to its efficiency.

Executive Services:

Executive Services encompasses a range of critical tasks such as system security, process management, Input/Output device handling, and object management. These services include both the kernel and the HAL, working together to ensure efficient and secure operation of the system.

Subsystem Environments:

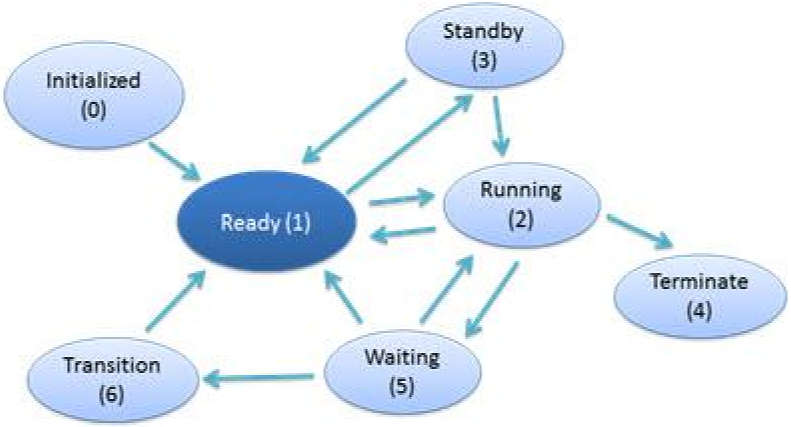
Subsystem Environments enable Windows to run applications that are designed for different operating systems. They receive API calls for applications and convert them into a format that is compatible with Windows. Notable subsystems include POSIX, OS/2, and WIN32, which facilitate seamless execution of diverse applications on the Windows platform.

Major Functions of Each Operating System

Analyzing the major functions of both operating systems, Linux and Windows, allows us to identify and compare their behaviors and differences. By examining the similarities and distinctions in their features, we can establish criteria for evaluating the efficiency and performance of each operating system.

Process States:

The process states, namely new, ready, running, waiting and terminated, exhibit similarities in both systems; a Process Control Block (PCB) contains comprehensive information about each process, including its unique process ID. To provide a visual representation, the process states can be metaphorically explained as follows:



Threads:

Threads, by definition, are processes that share resources with other processes. In the Linux kernel, threads are treated as standard processes, without any specific definition. This distinction sets Linux apart from other operating systems. In contrast, Windows supports threading, allowing computer systems with the Windows operating system to execute multiple threads based on the number of available processors.

Scheduling Algorithms Used:

Linux utilizes a single scheduling algorithm known as ‘priority-based’ for managing tasks. In contrast, Windows employs a more sophisticated approach with process-oriented scheduling algorithms. Windows utilizes various scheduling techniques, such as First-Come-First-Serve (FCFS), priority-based, Round Robin, and Shortest Job First (SJF), to optimize CPU performance and enhance overall system efficiency.

Schedulers:

In Linux, the scheduling of processes is primarily based on priority, with both static and dynamic priorities being considered. Processes that have been running for an extended period of time experience a decrease in priority, while waiting processes are dynamically preferred. On the other hand, Windows utilizes a round-robin scheduler, which allocates a fixed time for each process. As a result, Windows schedulers are commonly referred to as short-term schedulers.

Synchronization Techniques (Semaphores):

Linux follows a non-preemptive approach, meaning that a process cannot be replaced by a higher-priority process To control race conditions among shared data, Linux utilizes four primary types of synchronization: Atomic Operations, Locking, Interrupt Disabling, and Non-Preemptable of Processes in Kernel Mode. In contrast, Windows employs dispatcher objects for process/thread synchronization, which signal the availability of a process by indicating whether it is currently available or not.

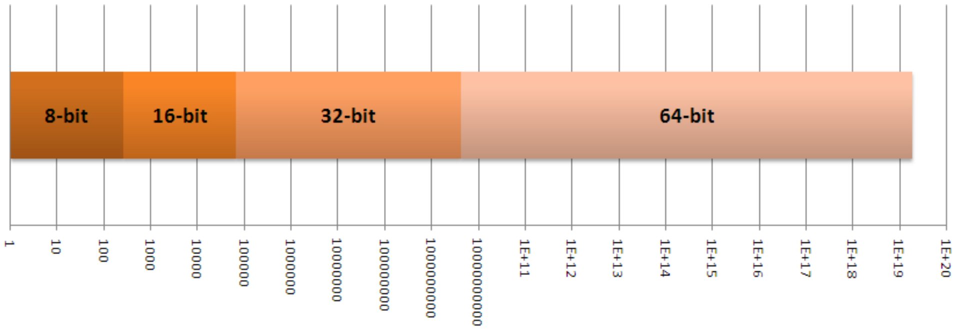
Handling Deadlocks:

In Linux, the availability of a comprehensive mechanism for deadlock detection and resolution is limited. Deadlock prevention primarily relies on the development of bug-free source code. On the other hand, Windows offers a more robust solution with its built-in deadlock detection monitors. These monitors analyze the code logic and prompt the necessary actions to handle deadlocks effectively. Moreover, Windows provides an additional tool called the deadlock kernel debugger extension, which proves to be highly effective in preventing the utilization of inadequate resources within the code. It is worth noting that windows ensures deadlock detection support across all versions, further enhancing its reliability in this aspect. By contrast, Linux may require additional measures and careful coding practices to address deadlock-related concerns.

Memory Organization and Management:

In Linux, the Memory Management Unit (MMU) plays a crucial role in creating virtual addresses within the system's memory. These virtual addresses consist of two essential components: the offset and the virtual page frame number. Whenever the processor encounters a virtual address, it must extract both the offset and the virtual page frame number to proceed. The primary objective of the CPU is to translate these virtual addresses into their corresponding physical addresses. To achieve this, Linux utilizes page tables, which serve as a reference to locate the frames in memory. By mapping virtual memory to physical addresses, Linux enables the flexibility of mapping virtual memory to physical pages in any desired order. The following figures provide a visual representation of how Linux accomplishes this mapping process, effectively bridging the gap between virtual and physical memory.

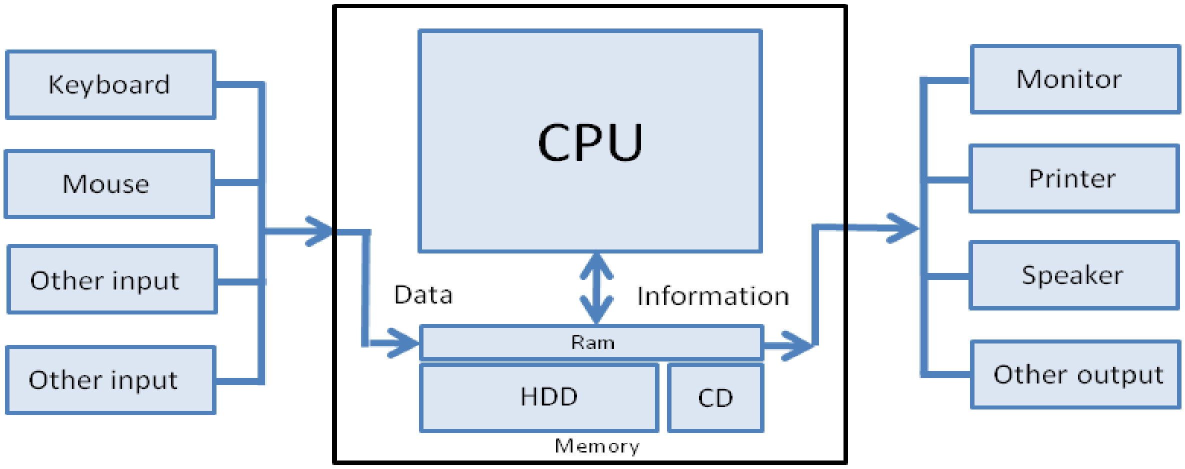
In Windows, there are two distinct types of memory management. The first type is found in 32-bit Windows OS, where the addressable memory space is intentionally shared between the kernel and active applications. In a 32-bit system, each virtual space has the capacity to address nearly 4GB of memory. On the other hand, in 64-bit Windows OS Memory Management, the virtual space expands to address a staggering 8TB of memory. By default a 64-bit system offers up to 16TB of addressable memory locations, providing a significant boost in memory capacity compared to its 32-bit counterpart. This enhanced memory management capacity in 64-bit Windows allows for more efficient and expansive memory utilization in various applications and scenarios.



Input/Output Management:

Linux manages Input/Output using a request structure. When a device requests a read or write operation, the Linux monolithic kernel invokes the rw_block() function, passing it an array of pointers. The rw_block() function, in turn, calls the make_request() routine for each device.

The make_request() call initially clusters the buffer with existing buffers in the request queue. It's worth nothing that this operation is specific to drivers compiled into the kernel and does not apply to loadable devices. If the request already exists, no new_request() is created. Linux handles two types of Input/Output, the first being Logical I/O and the second Device I/O.



The Windows Operating System excels in Input/Output management, showcasing an intriguing efficiency. It prioritizes interrupt handling over regular program execution. Windows leverages device drivers as a crucial intermediary for connecting and synchronizing software with hardware. This makes Input/Output an essential asset for those developing disk drivers.

Input/Output devices often operate at speeds that may not perfectly align with the pace of the Windows Operating System. Consequently, the primary mode of communication between device drivers and Windows revolves around Input/Output Request Packets(IRP's).

Windows employs a layered model for Input/Output management, organized into stacks. For instance, a mouse driver must interact with a USB hub, which communicates with a USB hub controller. The USB hub controller then establishes connections with the remaining computer hardware through the Peripheral Component Interconnect (PCI).

Linux Vs. Windows

The ongoing debate between Linux and Windows has persisted for decades. These two operating systems each possess their unique strengths and weaknesses. Linux excels in stability, security, and durability, making it a preferred choice. In contrast, Windows offers modern features, an intuitive GUI, and an array of application programs, making it a go-to for user friendliness.

A standout feature of Linux is its open-source nature, allowing for endless customization. Organizations can mold Linux to suit their specific needs, a freedom not as readily available in Windows. An essential factor to consider is the cost, where Windows tends to be high-priced in terms of software and hardware requirements, while Linux stands out as a freely available option online.

Windows enjoys popularity among the general public due to its user-friendly interface, enabling even those without deep programming knowledge to use it with ease. On the security front, Linux gains an edge over Windows, being less vulnerable to viruses and security threats.

Linus also boasts robust remote access protocols, offering enhanced security compared to Windows. Tools like Host, RealVNC empower, Zoho, and many more empower Linux users to remotely access their desktops securely from anywhere using the same protocols.

Linux primarily finds its place among experts, owing to its compatibility with the command line interface (CLI). It commands a significant market share in high-performance computing, embedded devices, and virtualization. Its support for virtualization makes it a valuable resource for sharing resources across systems.

In evaluating both Linux and Windows, it is clear that Linux does not seem likely to match Windows in the near future. Windows stands out as a more organized, user-centric, and advanced operating system. When it comes to performance, Windows excels due to its complex architecture, both in software and hardware requirements. Linux, as mentioned earlier, can run on minimal and less powerful hardware. On the other end, Windows' evolving versions demand more advanced and faster hardware, which seamlessly harmonizes with. Thus, in terms of performance, efficiency, and overall management, Windows will take the lead over Linux.

Limitations:

Operating System experts are diligently working to enhance memory management within computer systems. To achieve this goal, they are keen on employing more efficient and advanced data structure algorithms. Their primary goal is centered around two key aspects for optimizing memory:

(1) Enhancing Scalability - Scalability challenges stem from the need for locking mechanisms during atomic access to shared data structures. By effectively managing these factors, Linux's memory management can be significantly improved in the future.

(2) Minimizing Software Overhead - Software overhead primarily arises from tree transversal, contention lock, and storage costs. To address these issues and enhance memory efficiency in Linux, a preferred approach is to reorganize data structures, mitigating these overheads.

In contrast, Windows has already implemented a well-organized memory management system. To better serve its customers, Windows should focus on reducing licensing expenses and establishing a standardized file format for software applications across all Windows.

Additionally, Windows should prioritize upgrading its desktop remote access feature, which currently operates less efficiently compared to Linux's robust remote access tools. Linux remains a popular choice in server offices due to its low ownership costs, minimal hardware and software requirements, and the absence of renewal expenses for software applications.

References

Geeks for Geeks. “Difference between Linux and Windows.” *GeeksforGeeks*, 11 June 2019, www.geeksforgeeks.org/difference-between-linux-and-windows/.

“Linux vs. Windows.” *IONOS Digitalguide*, www.ionos.com/digitalguide/server/know-how/linux-vs-windows/.

“Simplified Windows Architecture Overview.” *Bucarotechelp.com*, bucarotechelp.com/computers/architecture/81080801.asp.

Ram and rom difference: *What is Ram & Rom?. Crucial. (n.d.).* https://www.crucial.com/articles/about-memory/what-is-the-difference-between-rm-and-rom#:~:text=RAM%2C%20which%20stands%20for%20random%20access%20memory%2C%20and,memory%20that%20permanently%20stores%20instructions%20for%20your%20computer.

Das, A. (2023a, June 2). *Advantages and disadvantages of using linux*. It’s FOSS. https://itsfoss.com/advantages-linux/