Arranque del Sistema Windows/Linux, análisis técnico

Pelayo Iglesias Manzano – UO266600

Pelayo Calleja Villafañe – UO292393

Índice

Introducción

El proceso de arranque es un factor muy importante en cualquier sistema operativo, sea Linux o Windows, ya que representa el inicio de la comunicación entre el hardware y el software que permite que todo funcione correctamente. Los sistemas operativos ya nombrados tienen arquitecturas y mecanismos de arranque distintos, debido a sus diferencias en diseño y objetivos. A lo largo de este trabajo, se analizarán las etapas de arranque en ambos sistemas, los gestores de arranque específicos, y las configuraciones de hardware y software involucradas en cada caso. Es necesario comprender todos estos puntos para diagnosticar problemas en el sistema o también para ver y entender los procesos que se llevan a cabo en el equipo desde que lo encendemos.

Dicho esto, el trabajo se encuentra dividido de manera cronológica, resaltando los factores más importantes involucrados en cada etapa del arranque de ambos sistemas operativos, Linux y Windows.

Pasos del arranque

Pasos que se realizan a la hora de arrancar el sistema Windows/Linux:

Paso 0: Encender el ordenador o dispositivo inteligente.

Paso 1: Ejecución del firmware (**BIOS/UEFI**).

Paso 2: Chequeo básico del hardware del equipo (**POST**).

Paso 3: Lectura y ejecución del "cargador de arranque" (**Bootloader**).

Paso 3A: Lectura del cargador de arranque (**Bootloader**).

1. En **BIOS => MBR.**
2. En **UEFI => GPT.**

Paso 3B: Tanto en Windows como en Linux, ejecución del cargador de arranque (**Bootloader**).

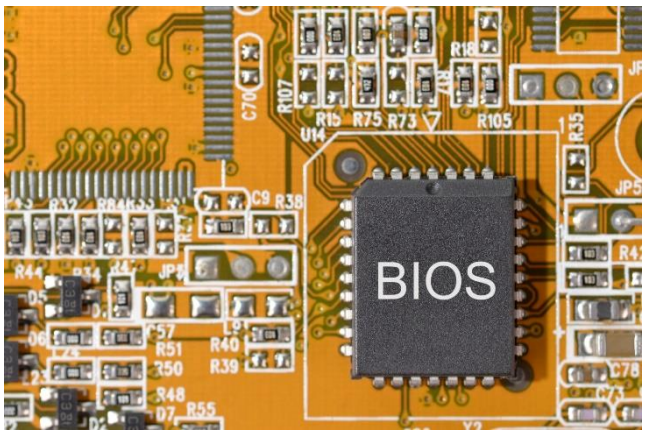
Paso 4: Ejecución del **Kernel** del Sistema Operativo.

BIOS/UEFI en sistemas Windows/Linux

Antes de hablar de BIOS o UEFI, debemos aclarar qué es un firmware para tener una base.

Un **firmware** es un tipo de código o programa que se almacena en una memoria de tipo ROM de un ordenador en forma de chip. Se encarga de inicializar y arrancar el ordenador antes de ceder el control al sistema operativo. Los firmwares son partes esenciales de los sistemas informáticos ya que deben controlar el flujo de datos entre las distintas partes del ordenador, gestionar y distribuir la energía y manejar la comunicación entre el ordenador y los dispositivos conectados.

El sistema básico de entrada y salida, o **BIOS (Basic Input Output System),** es el primer programa informático que se ejecuta al encender el ordenador. Su función principal es inicializar y probar el hardware del sistema durante el arranque y cargar el sistema operativo desde el disco duro (cualquier dispositivo de almacenamiento). Se almacena en una **memoria ROM**, está escrita en ensamblador y es código de 16 bits. Suele combinarse con discos duros particionados con **Master Boot Record (MBR).**



La BIOS posee varios componentes, que son:

- **Chip de la BIOS:** Se trata del hardware físico donde se almacena la BIOS. Normalmente, está integrado en la placa base y es el encargado de almacenar el firmware.
- **Memoria CMOS:** La CMOS (Semiconductor Complementario de Óxido Metálico) es un tipo de memoria volátil que almacena configuraciones básicas del sistema, como la fecha, la hora y el orden de arranque.
- **Batería CMOS:** Este pequeño, pero importante componente alimenta la memoria CMOS, asegurando que las configuraciones del sistema no se pierdan.
- **POST (Power-On Self Test):** Es una prueba automática que comprueba la integridad del hardware y verifica que todos los componentes del ordenador funcionan correctamente. La BIOS también comprueba los principales dispositivos de entrada del ordenador, como el teclado y el ratón, y los principales dispositivos de salida, como el monitor del ordenador o las impresoras que estén conectadas a él. Una vez acabado el POST, la BIOS inicializa los componentes del sistema, configurando los parámetros necesarios para que funcionen correctamente.

Después de realizar el POST, la BIOS busca el **sector de arranque (MBR en BIOS)** en el disco y carga el **cargador de arranque (bootloader).** Después de este proceso, cede el control al sistema operativo.

La BIOS proporciona una interfaz que permite a los usuarios configurar diferentes parámetros del sistema, como el orden de arranque, la configuración del hardware y las opciones de seguridad. Para acceder a esta interfaz, durante el arranque se debe pulsar **F2**, **Delete** o **Esc** (esto puede variar).

A partir de 2005 se introduce el **Unified Extensible Firmware Interface (UEFI)**, pero no fue hasta 2008 que lo vimos implantado en los ordenadores personales, gracias a Microsoft: Windows Vista de 64 bits y Windows Server 2008. Decir que la gran evolución la vimos allá por el lanzamiento de Windows 8, que fue cuando se implementó el **Arranque Seguro**.

La **UEFI (Unified Extensible Firmware Interface**) es una especificación que define una interfaz entre el sistema operativo y el firmware. Se lee la tabla de particiones **GUID (GPT),** de la cual hablaremos en otro apartado. UEFI reemplaza la antigua interfaz **BIOS** debido a sus claras ventajas ante esta:

- La EFI comunica el arranque además de con el ya clásico MBR, con el sistema **GPT** que solventa las limitaciones técnicas del MBR.
- Los BIOS hacen uso de modos de 16 bits para funcionar, diseño heredado del Intel 8088, pero a diferencia de esto, la EFI funciona directamente con modos de **32 bits y 64 bits**, permitiendo que las aplicaciones de la EFI tengan acceso completo al direccionamiento de 64 bits.
- Capacidad de arranque desde unidades de almacenamiento grandes, dado que no sufren de las limitaciones del MBR.
- **Un gestor de arranque propio de la UEFI** permite también la selección y carga directa de los sistemas operativos, eliminando la necesidad de recurrir a gestores de arranque.
- UEFI permite que los controladores y servicios genuinos se ejecuten en el arranque, lo que ya conocemos como "**arranque seguro**".

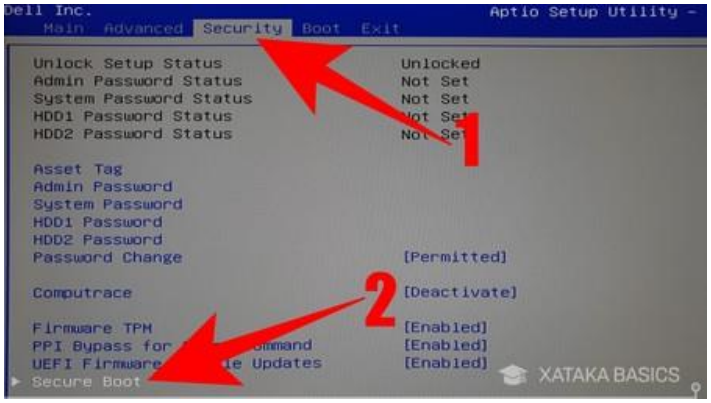
Los pasos que sigue la UEFI en su funcionamiento son muy parecidos a los de la BIOS. Al encender la computadora, el firmware UEFI se carga desde un chip en la placa base, para posteriormente iniciar un **proceso similar al POST en BIOS** en el que ejecuta una prueba rápida para asegurarse de que los componentes críticos estén funcionando correctamente. En esta etapa, también se detectan y configuran automáticamente los dispositivos, como discos duros, tarjetas de red y tarjetas gráficas. Tras realizar el POST, comprueba si el **Secure Boot (arranque seguro)** está habilitado. UEFI busca el gestor de arranque en la **Partición del Sistema EFI (ESP)**, una partición especial en el disco que contiene los archivos necesarios para arrancar el sistema operativo y que está formateada en **FAT32**. Después UEFI lee el archivo de **gestor de arranque (Bootloader)** designado, que suele ser **BOOTX64.EFI** en sistemas de 64 bits. El gestor de arranque carga el **kernel**

del sistema operativo en la memoria y establece los parámetros iniciales necesarios para su ejecución, para finalmente entregar el control al sistema operativo.

Si queremos acceder a la UEFI para cambiar su configuración o activar el **Secure Boot**, debemos pulsar las teclas **F2**, **Delete** o **Esc (como en BIOS)** durante el arranque del equipo.

Arranque seguro

Secure Boot (arranque seguro) es una característica de seguridad de la interfaz UEFI que ayuda a proteger el proceso de arranque de la computadora contra software malicioso. Su función principal es garantizar que solo se ejecuten sistemas operativos y controladores de confianza cuando la computadora se inicia, evitando que cualquier software no autorizado o modificado pueda manipular el sistema desde el arranque. Para esto **Secure Boot** utiliza **firmas digitales** para verificar la integridad y autenticidad de cada componente del arranque, desde el firmware hasta el sistema operativo.



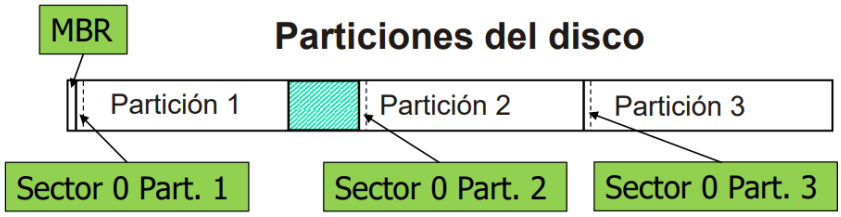
Las ventajas de este son:

- **Protección contra malware**: Reduce significativamente el riesgo de que se instale malware.
- **Integridad del sistema**: Ayuda a garantizar que el sistema operativo no esté manipulado.
- **Autenticidad**: Al verificar los sistemas operativos y controladores, se asegura de que estos provengan de fuentes confiables y no hayan sido modificados.

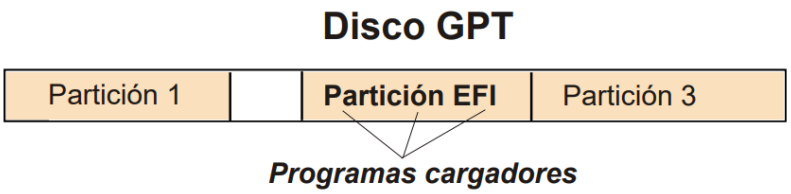
MBR/GPT

Antes de poder utilizar un disco duro, debe particionarlo, es decir, dividirlo en secciones lógicas en las que almacenar datos. Existen dos métodos alternativos para almacenar información de partición en una unidad: MBR (registro de arranque maestro) y GPT (tabla de partición GUID).

El **MBR** es el primer sector de todo el disco. MBR está limitado a 4 particiones primarias y puede manejar discos duros con una capacidad máxima de 2 TB.



GPT es una tecnología de partición de disco duro más nueva, introducida en 2000. GPT admite discos duros de 9,4 ZB (zettabytes) y puede manejar hasta 128 particiones. GPT ofrece mayor confiabilidad e integridad de datos que MBR debido a su capacidad para crear copias de seguridad de las tablas de particiones. Además, **tiene compatibilidad total con el sistema operativo.**



Además de la capacidad de almacenamiento y número de particiones, ambas tecnologías presentan unas claras diferencias y ventajas/desventajas una respecto a la otra:

- **Compatibilidad: MBR** es compatible con sistemas operativos antiguos, como versiones anteriores a Windows XP y Windows Server 2003, así como con sistemas BIOS tradicionales. **GPT** requiere sistemas UEFI y generalmente no es compatible con sistemas antiguos o BIOS tradicionales sin UEFI. Funciona de manera nativa en sistemas modernos como Windows 8 y posteriores, y en muchas distribuciones Linux recientes.

- **Seguridad:** El **MBR** sólo tiene una ubicación de almacenamiento para la información de partición (en el inicio del disco). Esto significa que, si se corrompe el MBR, el sistema no podrá arrancar y la información de las particiones puede perderse. **GPT** ofrece mayor seguridad porque guarda múltiples copias de la tabla de particiones en diferentes ubicaciones del disco. Además, incluye códigos de verificación **CRC (Cyclic Redundancy Check)** para verificar la integridad de las tablas de particiones y detectar errores.

- **Identificación de particiones:** Con MBR las particiones están identificadas de manera básica y no es posible asignarles identificadores únicos globales. Con GPT

cada partición tiene un **GUID (Globally Unique Identifier)**, un identificador único que permite diferenciarlas sin confusión y facilita la recuperación de datos en caso de problemas.

En general, **GPT es superior a MBR** en términos de capacidad, seguridad y flexibilidad, y está diseñado para soportar las necesidades de almacenamiento y seguridad de los sistemas modernos. MBR sigue siendo útil en sistemas antiguos o aquellos que solo requieren particiones limitadas y son compatibles con BIOS, pero GPT es ahora el estándar recomendado para la mayoría de los sistemas modernos que utilizan UEFI.

Bootloaders en Linux

En Linux existen varios tipos de **Bootloaders** que se utilizan para gestionar el proceso de arranque del sistema. Cada uno tiene sus características, funciones y compatibilidades particulares. Los más comunes son:

- **GRUB:** Es el gestor y cargador de arranque para Linux y otros sistemas operativos basados en Unix **más usado**.
- **LILO:** Configuración manual y sin menú de arranque gráfico. Ha sido **reemplazado por GRUB**, pero aún puede encontrarse en sistemas antiguos o especializados.
- **SYSLINUX: SYSLINUX** está diseñado principalmente para arrancar sistemas desde dispositivos extraíbles como **USB, CD-ROM o red**. Usado en recuperación, discos de instalación de Linux, y arranque por red. Sus variantes son: **ISOLINUX, PXELINUX Y EXTLINUX**.
- **Loadlin:** Era útil cuando muchos **sistemas carecían de soporte completo para arrancar Linux directamente desde el BIOS**. Es ideal para sistemas con procesadores x86 y arquitecturas DOS compatibles, pero a diferencia de otros bootloaders, **no es compatible con sistemas UEFI o configuraciones de hardware modernas**.
- **BURG: Basado en GRUB**, pero permite personalizar el menú de arranque con temas gráficos y tiene una interfaz de usuario simplificada.
- **systemd-boot: Parte de systemd** y diseñado específicamente para sistemas UEFI. Es ligero y fácil de configurar, pero no tan versátil como GRUB en términos de **multi-boot** y **compatibilidad con BIOS**.

Bootloaders en Windows

En los sistemas operativos Windows, los bootloaders son programas esenciales en el proceso de arranque del sistema. Un bootloader o cargador de arranque, es responsable de cargar el núcleo del sistema operativo (kernel) en la memoria RAM y en última instancia, iniciar el sistema operativo. Estos programas son clave para la inicialización de cualquier sistema y en Windows específicamente, existen varios componentes y

tecnologías involucradas en el proceso de arranque. A continuación, hablaremos sobre ellos.

- **BOOTMGR:** Se encarga de buscar y cargar el archivo de arranque del sistema operativo (Boot Configuration Data o BCD) que contiene la información que Windows necesita para iniciarse. Una vez que se ha cargado el BCD, BOOTMGR es el encargado de realizar una serie de comprobaciones para garantizar que el sistema operativo se pueda cargar correctamente. Puede pasar que BOOTMGR no se encuentre presente o se haya perdido, cuando esto ocurre, es posible que sea necesario reparar o reinstalar el sistema operativo o realizar otros pasos para solucionar el problema
- **UEFI BootLoader:** También se le puede llamar Interfaz de Firmware Extensible Unificada es una tecnología que controla el hardware de tu ordenador cuando lo enciendes y que sustituye a BIOS en algunos ordenadores desde hace años.
- **Winload.ex**: Es un componente crucial en el proceso de arranque de sistemas operativos Windows modernos. Este archivo ejecutable es conocido como el cargador del sistema operativo y se encarga el núcleo (kernel) de Windows y otros componentes esenciales para iniciar el sistema operativo. Se encuentra en la carpeta System32 dentro del directorio de instalación de Windows y cumple funciones vitales para el arranque del sistema.
- **NTLDR:** Es el archivo encargado del arranque del Sistema Operativo en las primeras versiones de Microsoft Windows NT, incluyendo Windows XP, y Windows Server 2003. El NTDLR se encuentra usualmente en el disco duro principal, pero también puede encontrarse en dispositivos portátiles como CD-ROM, memorias USB o disquetes. Este archivo requiere como mínimo, que dos archivos se encuentren en el directorio raíz del volumen de inicio:

    - NTDLR, que se encarga de cargar el sistema operativo
    - boot.ini, que contiene un menú de opciones de inicio.

    Si el archivo no se encuentra en el disco, el ordenador enviará un mensaje de error informándolo.

Kernel

El **kernel o núcleo** es una parte fundamental del sistema operativo que se encarga de conceder el acceso al hardware de forma segura para todo el software que lo solicita. Todos los sistemas operativos tienen un kernel, pero el más famoso es el de Linux.

Este núcleo de los sistemas operativos **se ejecuta en modo privilegiado con acceso especial a todos los recursos del sistema** para así poder realizar las peticiones de acceso que el software que lo necesita lo va pidiendo, también decide el orden de las peticiones recibidas según la prioridad e importancia de estas

El kernel sirve para **administrar los recursos de hardware** solicitados por los diferentes elementos de software y hacer de intermedio decidiendo a que y cuando se concede este acceso evitando así sobrecarga del sistemas, recursos innecesarios y acceso a software malicioso al propio kernel y llegar a poder controlar así todo el sistema. Además, sirve como **elemento de seguridad** teniendo que pasar por varias capas antes de poder tener acceso, además, tiene que distribuir los recursos de manera eficiente y ordenada para que el hardware trabaje junto al software de la mejor manera posible.

Generalmente, relacionamos un kernel del sistema operativo a un PC, pero la verdad es que está presente y sirve para hacer funcionar todos los computadores que podemos encontrar hoy en día, por poner un ejemplo los dispositivos móviles con Android e iOS, también disponen de un kernel basado en Linux/Unix.

El kernel también **concede acceso a todos los periféricos** que tengamos conectados e interactuar con el software que los solicite. Por ejemplo, si conectamos un móvil para usarlo como webcam con DroidCam, este kernel se encarga de conceder los permisos necesarios al software para gestionar y poder tener la imagen y el audio.

El resumen el kernel es el encargado de hacer funcionar básicamente todo, tiene que ser capaz de arrancar, por ejemplo, un PC desde que lo encendemos hasta que vemos visible el escritorio, todo esto comunicándose con los elementos hardware que dispone el PC y también son necesarios para hacerlo funcionar, una vez que tengamos el escritorio deberá ser capaz de hacer funcionar los programas que nosotros queramos abrir y hacerlos funcionar en nuestro PC.

A continuación, hablaremos de los kernel tanto de Linux como de Windows y explicaremos un poco las diferencias entre ambos.

Kernel Linux

El kernel Linux es de código abierto, lo que significa que su código fuente está disponible para cualquiera que quiera verlo, modificarlo o distribuirlo. Esta es una de las razones principales por las que ha habido tantas variaciones del sistema operativo Linux. Es un destacado ejemplo de software libre, un modelo de desarrollo que promueve la libertad de usar, estudiar, compartir y modificar el software. Su creador fue Linus Torvalds y lo lanzó en 1991 bajo la **licencia pública general GNU**, que representa un cambio radical frente al software propietario, donde el código no se comparte.

Las distribuciones de Linux pueden variar significativamente en términos de apariencia y funcionalidad, pero todas comparten el mismo núcleo: el kernel de Linux. Las distribuciones más populares incluyen Ubuntu, Fedora y Debian para uso general. Para entonos de servidor se usa **RockyLinux, CloudLinux, AlmaLinux, CentOS y Red Hat**. Para pruebas de penetración y seguridad se usa **Kali Linux**.

El kernel de Linux, está construido con una arquitectura monolítica, que es una de las razones clave de su eficiencia y rendimiento. A pesar de ser un kernel monolítico, también

incorpora características de los **microkernels** mediante módulos, lo que permite la carga y descarga de funcionalidades en tiempo de ejecución.

El kernel Linux consta de varios componentes clave, algunos de ellos son:

1. **Gestión de la memoria**
2. **Programador de procesos**
3. **Sistema de Archivos**
4. **Controladores de Dispositivos**
5. **Sistema de Red**
6. **Módulos del Kernel**

Kernel Windows

En la década de los noventa, Microsoft estaba basando sus sistemas operativos en los kernel Windows 9x, donde el código básico tenía muchas similitudes con MS-DOS. Necesitaba recurrir a él para poder operar. Microsoft estaba desarrollando otra versión de su sistema dirigido a los servidores llamada **Windows NT**, que nació en 1993.

La principal característica del **kernel de Windows NT** es que es bastante modular y está basada en dos capas principales, la de usuario y la de kernel. El sistema utilizar cada una para diferentes tipos de programa. Por ejemplo, las aplicaciones se ejecutan en el modo usuario y los componentes principales del sistema operativo en modo kernel. Mientras la mayoría de los drivers suelen usar el modo kernel, aunque con excepciones. Es por esto por lo que se refiere a él como kernel híbrido, pero sobre todo también porque permite tener subsistemas en el espacio del usuario que se comunicaban con el kernel a través de un mecanismo de IPC. Cuando ejecutas una aplicación, esta accede al modo usuario, donde Windows crea un proceso específico para la aplicación. Cada aplicación tiene su **dirección virtual privada**, ninguna puede alterar los datos que pertenecen a otra y tampoco acceder al espacio virtual del propio sistema operativo. Es por lo tanto el modo que menos privilegios otorga.

El modo núcleo en cambio es ese en el que el código se ejecuta en él tiene acceso directo a todo el hardware y toda la memoria del equipo. Aquí todo el código comparte un mismo espacio virtual, y puede incluso acceder a los espacios de dirección de todos los procesos del modo usuario. Esto es peligroso, ya que si un driver en el modo kernel toca lo que no debe podría afectar al funcionamiento de todo el sistema operativo. Este modo núcleo **está formado por servicios executive**, como el controlador de caché, el gestor de comunicación, gestor de E/S, las llamadas de procedimientos locales, o los gestores de energía y memoria entre otros. Estos a su vez están formados por varios módulos que realizan tareas específicas, controladores de núcleo, un núcleo y una Capa de Abstracción del Hardware o HAL

Diferencias entre el kernel Linux y Windows

La primera diferencia que podemos encontrar es que **Windows utiliza un kernel monolítico**, lo que significa que la mayoría de los componentes esenciales del sistema operativo, como los controladores de dispositivo y el sistema de archivos, están integrados directamente en el kernel principal, esto, puede conducir a un mayor acoplamiento y a una mayor complejidad de gestión de recursos. **Linux por su parte utiliza un enfoque modular**, donde muchas funciones del kernel se implementan como módulos que pueden cargarse y descargarse dinámicamente según sea necesario. Esto permite una mayor flexibilidad y optimización en la gestión de recursos, ya que los módulos se pueden ajustar para adaptarse a diferentes configuraciones de hardware y requerimiento de software.

La siguiente diferencia que encontramos es que **Windows utilizar un sistema de priorización de procesos basado en niveles de prioridad estáticos**, donde los procesos se clasifican en niveles de prioridad fijos que determinan su acceso a los recursos del sistema. Esto puede llevar a situaciones donde ciertos procesos monopolizan recursos sin considerar la carga del sistema. Por su lado **Linux utiliza un sistema de priorización dinámica de procesos** mediante el uso de la planificación de CPU basada en colas de prioridad. Esto permite que el kernel adapta la asignación de recursos según la carga del sistema y la importancia de cada proceso, garantizando una distribución más equitativa y eficiente de los recursos.

Otra diferencia es que **Windows emplea un sistema de gestión de memoria virtual paginada**, donde se utiliza un archivo de intercambio (swap) en disco para almacenar datos cuando la memoria física está saturada. Esto puede resultar en degradación del rendimiento cuando se produce un alto uso de la memoria virtual. **Linux también utiliza un sistema de memoria virtual paginada, pero su gestión es más eficiente** debido a su diseño modular y a las herramientas avanzadas de administración de memoria, como el kernel de baja latencia y el soporte para el acceso directo a memoria (DMA), que optimizan el rendimiento y la estabilidad del sistema incluso en situaciones de alta carga.

La última diferencia es que **Windows utiliza principalmente el sistema de archivos NTFS**, que ofrece características avanzadas como el control de acceso, la comprensión y el registro de cambios. Sin embargo, NTFS puede presentar cierta complejidad en la gestión de recursos y en la optimización del rendimiento en comparación con sistemas de archivos más ligeros. Mientras que **Linux es compatible con una amplia variedad de sistemas de archivos**, incluyendo ext4, Btrfs, XFS, entre otros. Estos sistemas de archivos ofrecen diferentes características y niveles de rendimiento, permitiendo a los usuarios elegir el más adecuado según sus necesidades de gestión de recursos y almacenamiento.

En conclusión, la gran diferencia técnica entre el kernel de Windows y el de Linux radica en su diseño monolítico frente al modular, su enfoque de integración de funciones y

servicios, y su filosofía de desarrollo y colaboración. Estas diferencias tienen un impacto significativo en el rendimiento, la flexibilidad y la seguridad de cada sistema operativo.

Apartado experimental: medidas de arranque con diferentes Bootloaders

Como prueba experimental hemos decidido hacer una comparativa de los tiempos de arranque del sistema Linux utilizando los bootloaders GRUB y systemd-boot.

Tras configurar correctamente ambos, gracias al comando "*systemd-analyze*" hemos tenido estos resultados:

GRUB

```
Startup finished in 911ms (kernel) + 1.783s (initrd) + 15.146s (userspace) = 17.841s
multi-user.target reached after 3.896s in userspace.
```

systemd-boot

```
Startup finished in 868ms (kernel) + 1.715s (initrd) + 4.158s (userspace) = 6.742s
multi-user.target reached after 2.823s in userspace.
```

Analizando estos datos, podemos confirmar el hecho de que el bootloader **systemd-boot**, como ya hemos dicho anteriormente, es mucho más ligero que **GRUB** y por esta razón tiene un tiempo notablemente más corto, además **systemd-boot** está hecho especialmente para sistemas Linux con UEFI mientras que **GRUB** es más polivalente.

systemd-boot actúa como un cargador EFI ligero que solo selecciona el kernel y la imagen de inicialización (**initramfs**) directamente desde el sistema de archivos EFI, sin interpretar configuraciones adicionales. **GRUB** lee su configuración desde un archivo de texto y, en algunos casos, requiere un proceso adicional de carga de módulos, lo que puede alargar un poco el tiempo de arranque.

Sabiendo esto, **systemd-boot** es una buena elección cundo buscamos rendimiento y simplicidad, pero **GRUB** destaca ampliamente en compatibilidad y tiene una configuración avanzada.

Conclusión

Con este trabajo hemos comprendido que el proceso de arranque es esencial para que los sistemas operativos Windows y Linux funcionen correctamente. Windows utiliza un arranque más controlado y seguro, con herramientas como el Windows Boot Manager y el Secure Boot, lo cual refuerza la seguridad y compatibilidad. En cambio, Linux destaca por su flexibilidad y opciones de personalización, permitiendo adaptarse a diversas configuraciones mediante bootloaders como GRUB y systemd-boot. Conocer estas diferencias permite a los administradores y usuarios elegir el sistema adecuado para cada caso y mejorar el rendimiento, seguridad y gestión del entorno de trabajo.

Bibliografía

https://www.profesionalreview.com/guias/bios/

https://es.wikipedia.org/wiki/BIOS

https://www.publico.es/antivirus/que-es-la-bios-de-un-ordenador/

https://es.wikipedia.org/wiki/Unified_Extensible_Firmware_Interface

https://www.pccomponentes.com/bios-uefi-que-es?srsltid=AfmBOort-Wf4e7ucJUhTFpA0_CCCgdD306ge6wbK0OgBRR-R_EVEo-16

https://learn.microsoft.com/es-es/windows-hardware/design/device-experiences/oem-secure-boot

https://ibertronica.es/blog/actualidad/arranquesegurowindows11/

https://www.4ddig.net/es/migrar-so/mbr-vs-gpt.html?gad_source=1&gclid=Cj0KCQjwm5e5BhCWARIsANwm06geju768ACVoWdh9fdF5Q7Mg5qXTNXMmlCXR9MBtWu4USMoDMSR-IMaAu1QEALw_wcB#part%201

https://4ddig.tenorshare.com/es/migrar-so/mbr-vs-gpt.html

https://www.seguinet.es/linux-bootloader/

https://phoenixnap.com/kb/what-is-grub

https://www.geeknetic.es/Kernel/que-es-y-para-que-sirve

https://administraciondesistemas.com/que-es-el-kernel-de-linux/

https://www.genbeta.com/a-fondo/como-es-el-kernel-de-windows-y-cuales-son-sus-diferencias-con-el-de-linux

https://laboratoriolinux.es/index.php/-noticias-mundo-linux-/software/36065-las-grandes-diferencias-tecnicas-del-kernel-de-windows-frente-al-kernel-de-linux.html

https://www.adslzone.net/esenciales/windows/error-falta-bootmgr/

https://www.adslzone.net/esenciales/windows-10/bios-uefi-windows-10/

https://es.wikipedia.org/wiki/NTLDR