SISTEMA DE ARCHIVOS

MÓDULO 5

Sistemas Abiertos – Administración de SO I

Contenido del Módulo 5.

- Introducción al sistema de archivos.
  - Definición. Importancia del sistema de archivos.
- Sistemas de archivos soportados por GNU/Linux.
  - ext4, XFS, Btrfs, FAT32 y exFAT, NTFS.
- El Virtual File System (VFS).
  - Concepto y función. Estructura. Interacción con sistemas de archivos físicos.
- Sistema de archivos ext4.
  - Estructura. Ventajas y características.
- Estructura de directorios en GNU/Linux.
  - Estándar de Jerarquía de Sistema de Archivos (FHS). Descripción.
- Tipos de archivos en GNU/Linux.
  - Normales. Directorios. Especiales. Enlaces.
- Montaje de sistemas de archivos.
  - Comando *mount*. Opciones de *mount*. Verificación. Desmontar. Comparación.

Introducción al sistema de archivos.

- **Definición de un sistema de archivos.**
  - Estructura y método para controlar cómo se almacenan y recuperan los datos en un dispositivo de almacenamiento.
  - Sin un sistema de archivos, los datos serían un bloque continuo de información sin organización.
    - Sería imposible saber dónde comienza o termina un archivo.
  - Se encarga de gestionar y organizar los datos en bloques o sectores del dispositivo de almacenamiento.
  - Asigna a cada archivo una localización física concreta y proporciona una manera eficiente de acceder a ellos.

Introducción al sistema de archivos.

- **Importancia en la gestión de datos y recursos.**
  - El sistema de archivos es fundamental en la gestión de datos y recursos del sistema operativo por varias razones:
    - **Organización:** Permite la creación de una jerarquía de archivos y directorios. Se puede crear, modificar, mover y eliminar archivos de manera eficiente.
    - **Acceso eficiente:** Optimizan el tiempo de acceso a los datos. Esto incluye índices o estructuras de árbol que permiten localizar rápidamente un archivo.
    - **Gestión de espacio:** Administra el uso del espacio en disco. Distribuye los archivos en bloques físicos.

Introducción al sistema de archivos.

- **Importancia en la gestión de datos y recursos.**
  - El sistema de archivos es fundamental en la gestión de datos y recursos del sistema operativo por varias razones:
    - **Seguridad:** Proporciona mecanismos de seguridad a través de los permisos de acceso y características avanzadas como el SUID y el SGID. Además permiten registro de transacciones para evitar la corrupción de datos en caso de una falla del sistema.
    - **Recuperación de fallos:** Muchos sistemas de archivos implementan funciones de **journaling** o **snapshots** que ayudan a evitar la pérdida de datos tras un fallo del sistema o una interrupción inesperada.

Sistemas de archivos en GNU/Linux.

- GNU/Linux es compatible con una amplia gama de sistemas de archivos, c/u diseñado para diferentes necesidades y casos de uso.
- **ext4 (Fourth Extended File System).**
  - Es el sistema de archivos por defecto en muchas distribuciones. Optimizado para velocidad y eficiencia en el manejo de archivos.
  - **Ventajas:**
    - **Compatibilidad amplia:** Soportado de forma nativa por la mayoría de las distribuciones de Linux.
    - **Journaling:** Implementa un sistema de registro de transacciones (journal), que ayuda a proteger la integridad de los datos en caso de un fallo.

Sistemas de archivos en GNU/Linux.

- **ext4 (Fourth Extended File System).**
  - Es el sistema de archivos por defecto en muchas distribuciones. Optimizado para velocidad y eficiencia en el manejo de archivos.
  - **Ventajas:**
    - **Gran capacidad de almacenamiento:** Puede manejar sistemas de archivos de hasta 1 exabyte y archivos individuales de hasta 16 terabytes.
    - **Desfragmentación:** Mejorado en comparación con ext3, lo que permite reducir la fragmentación en disco.

Sistemas de archivos en GNU/Linux.

- **ext4 (Fourth Extended File System).**
  - **Desventajas:**
    - **No es COW (Copy-on-Write):** no permite snapshots nativos.
    - **Limitado en características avanzadas:** carece de ciertas funcionalidades de gestión de volúmenes dinámicos y compresión de datos.

Sistemas de archivos en GNU/Linux.

- **XFS (Extended File System).**
  - Sistema de archivos de alto rendimiento, optimizado para archivos grandes y operaciones de L/E de archivos secuenciales.
  - **Ventajas:**
    - **Rendimiento en archivos grandes:** Ideal para aplicaciones que manejan grandes volúmenes de datos, como bases de datos y procesamiento multimedia.
    - **Alto rendimiento en L/E:** Gracias a su diseño para gestionar eficientemente grandes bloques de datos.

Sistemas de archivos en GNU/Linux.

- **XFS (Extended File System).**
  - **Ventajas:**
    - **Gestión de espacio avanzada:** Permite administración dinámica del espacio, facilitando expansión de volúmenes en tiempo real.
  - **Desventajas:**
    - **Complejidad en recuperación de datos:** La recuperación puede ser difícil y lenta en caso de fallo.
    - **No permite reducir el tamaño de la partición:** solo permite la expansión, pero no la contracción de particiones.
    - **Menor flexibilidad para escritorios:** Aunque es excelente en servidores, puede no ser tan versátil en entornos de uso general

Sistemas de archivos en GNU/Linux.

- **Btrfs (B-tree File System).**
  - Sistema de archivos de próxima generación, diseñado para ofrecer funcionalidades avanzadas como snapshots, compresión y gestión de volúmenes.
  - **Ventajas:**
    - **Copy-on-Write (COW):** Soporta snapshots nativos, lo que facilita la realización de copias de seguridad incrementales sin duplicar el espacio en disco.
    - **Compresión transparente:** Ahorra espacio en disco al comprimir datos automáticamente.

Sistemas de archivos en GNU/Linux.

- **Btrfs (B-tree File System).**
  - **Ventajas:**
    - **Gestión de volúmenes y subvolúmenes:** Permite crear y gestionar subvolúmenes dentro de un mismo sistema de archivos.
    - **Detección y corrección de errores:** Implementa sumas de verificación para los datos y metadatos, lo que facilita la detección y corrección de errores en disco.

Sistemas de archivos en GNU/Linux.

- **Btrfs (B-tree File System).**
  - **Desventajas:**
    - **Inestabilidad relativa:** Aunque ha mejorado, Btrfs aún tiene limitaciones de estabilidad en comparación con ext4 en ciertos escenarios.
    - **Rendimiento menor en archivos pequeños:** Puede ser más lento en operaciones que manejan numerosos archivos pequeños, como las cargas de trabajo de servidores de bases de datos.
    - **Fragmentación:** Debido a su diseño COW, puede experimentar mayor fragmentación que otros sistemas de archivos.

Sistemas de archivos en GNU/Linux.

- **FAT32 y exFAT.**
  - Sistemas de archivos diseñados por Microsoft, y son comunes en dispositivos extraíbles por su gran compatibilidad con múltiples SO.
  - **Ventajas:**
    - **Alta compatibilidad:** FAT32 y exFAT son compatibles con la mayoría de los sistemas operativos, incluyendo Windows, macOS y Linux.
    - **Adecuado para dispositivos portátiles:** Ideal para unidades USB y tarjetas SD.
    - **Ligero:** Requiere menos recursos y es fácil de implementar en dispositivos con recursos limitados.

Sistemas de archivos en GNU/Linux.

- **FAT32 y exFAT.**
  - **Desventajas:**
    - **Limitaciones en tamaño de archivo (FAT32):** FAT32 no soporta archivos de más de 4 GB, lo cual es una limitación significativa en entornos modernos.
    - **Sin seguridad avanzada:** No soporta permisos avanzados o características de seguridad como el journaling.
    - **No es adecuado para sistemas críticos:** Su simplicidad lo hace inadecuado para entornos de servidor o almacenamiento a largo plazo.

Sistemas de archivos en GNU/Linux.

- **NTFS (New Technology File System).**
  - Nativo de Windows, ofrece funcionalidades avanzadas como permisos de archivos y journaling. Aunque es soportado en Linux, su rendimiento puede no ser óptimo en todos los casos.
  - **Ventajas:**
    - **Permisos y seguridad avanzados:** Ofrece control de acceso granular y características avanzadas de seguridad.
    - **Journaling y recuperación de errores:** Ayuda a proteger la integridad de los datos en caso de fallos.
    - **Tamaño de archivo y partición elevado:** No tiene las limitaciones de tamaño de FAT32, soportando archivos y particiones muy grandes

Sistemas de archivos en GNU/Linux.

- **NTFS (New Technology File System).**
  - **Desventajas:**
    - **Compatibilidad parcial en Linux:** Aunque Linux puede leer y escribir en NTFS, el soporte completo puede depender de herramientas como `ntfs-3g`.
    - **No diseñado para Linux:** Puede ser ineficiente o menos estable en Linux, especialmente en entornos de escritura intensiva.
    - **Menor flexibilidad:** No es tan flexible ni eficiente en Linux como ext4 o XFS, por lo que no es ideal para almacenamiento primario en Linux.

El Virtual File System (VFS).

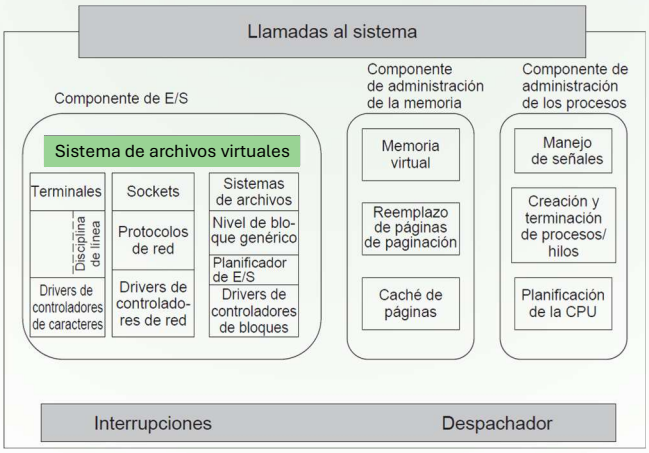
- **Concepto y función del VFS en Linux.**
  - Es una capa de abstracción dentro del kernel de Linux que permite al SO interactuar con múltiples filesystems de forma uniforme.
  - En otras palabras, VFS actúa como una "interfaz virtual" que traduce las operaciones comunes sobre archivos a comandos específicos para el sistema de archivos subyacente.
  - El usuario o las aplicaciones no necesitan conocer los detalles específicos de cada sistema. Esto hace de Linux un sistema operativo versátil y escalable.
  - Además, simplifica la compatibilidad con dispositivos externos formateados en sistemas no nativos.

El Virtual File System (VFS).

- **Concepto y función del VFS en Linux.**
  - **Principales funciones de VFS:**
    - **Unificación de operaciones:** permite a las aplicaciones realizar operaciones sobre archivos sin preocuparse por el sistema de archivos real en el que se encuentran.
    - **Portabilidad y flexibilidad:** puede ser compatible con gran variedad de sistemas de archivos, incluso de otros SO, fundamental para doble arranque y L/E en unidades extraíbles.
    - **Simplificación del kernel:** organiza y centraliza las operaciones de E/S, aliviando al kernel de instrucciones específicas para cada sistema de archivos.

El Virtual File System (VFS).

- **Concepto y función del VFS en Linux.**
  - **Principales funciones de VFS:**



El Virtual File System (VFS).

- **Estructura del VFS.**
  - Para entender cómo funciona, veamos sus estructuras principales:
    - **Superbloque (superblock):** Almacena información general sobre el sistema de archivos, como su tamaño y tipo, y se carga al montar el sistema de archivos.
    - **Inodo (inode):** Cada archivo o directorio en Linux se representa con una estructura de datos, **inodo**, que contiene información como permisos, propietario, tamaño y punteros a los bloques de datos. VFS mantiene una tabla de inodos en memoria para acceder rápidamente a los archivos.

El Virtual File System (VFS).

- **Estructura del VFS.**
  - Para entender cómo funciona, veamos sus estructuras principales:
    - **Dentry (directory entry):** Es una estructura que vincula el nombre del archivo con su inodo. Esto facilita la navegación en el sistema de archivos, permitiendo que se traduzcan nombres de archivos a sus respectivas ubicaciones físicas.
    - **File:** Esta estructura representa los archivos abiertos en el sistema, almacenando información sobre su posición y estado en cada proceso que los abre.

El Virtual File System (VFS).

- **Interacción con sistemas de archivos físicos.**
  - El VFS se encuentra entre las aplicaciones del usuario y los sistemas de archivos físicos, y cumple un papel de "traductor" en este proceso. La interacción funciona de la siguiente manera:
    - **Aplicaciones:** Cuando una aplicación realiza una operación se envía una solicitud al SO. La aplicación no sabe (ni necesita saber) el filesystem, VFS se encarga de la traducción.
    - **VFS como interfaz común:** Al recibir la solicitud, el VFS determina el filesystem al que pertenece el archivo y localiza el inodo. Luego, envía la solicitud al controlador específico.

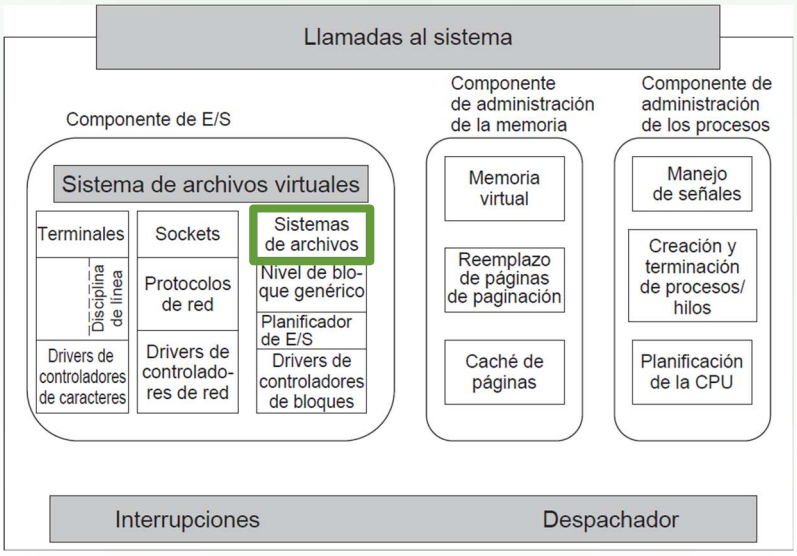
El Virtual File System (VFS).

- **Interacción con sistemas de archivos físicos.**
  - El VFS se encuentra entre las aplicaciones del usuario y los sistemas de archivos físicos, y cumple un papel de "traductor" en este proceso. La interacción funciona de la siguiente manera:
    - **Controladores de filesystem:** Los controladores de filesystem implementan las operaciones específicas según las estructuras y particularidades del sistema.
    - **Respuesta de los datos:** Finalmente, el controlador devuelve el resultado de la operación al VFS, que a su vez se lo transmite a la aplicación.

Sistema de archivos ext4.

- Cuarta versión del filesystem original (Fourth Extended Filesystem).
- Disponible a partir de la versión 2.6.28 del kernel.
- Sistema de archivos por defecto en la mayoría de las distribuciones de GNU/Linux, por su:
  - **Estabilidad.**
  - **Eficiencia.**
  - **Soporte para sistemas de almacenamiento modernos.**
- Diseñado para mejorar rendimiento y fiabilidad, incluye características avanzadas que permiten una administración de archivos y datos más efectiva.

Sistema de archivos ext4.



Sistema de archivos ext4.

- **Estructura básica de ext4.**
- Se compone de varios bloques y estructuras de datos que garantizan la integridad y eficiencia del sistema de archivos. Tiene cuatro elementos:
  - **Bloque de arranque (Boot Block)**
  - **Superbloque.**
  - **Inodos.**
  - **Bloques de datos.**

Sistema de archivos ext4.

- **Estructura básica de ext4.**
  - **Bloque de arranque (Boot Block):** Es el primer bloque de un sistema de archivos ext4 y contiene el código de arranque del sistema operativo.
  - Aunque ext4 permite crear sistemas de archivos en cualquier partición, el bloque de arranque suele ubicarse en la primera partición y contiene instrucciones básicas para cargar el SO.

Sistema de archivos ext4.

- **Estructura básica de ext4.**
  - **Superbloque:** El superbloque es una estructura crítica que almacena información esencial sobre el sistema de archivos, como:
    - Tamaño total del sistema de archivos.
    - Estado actual del sistema (montado o desmontado).
    - Número total de inodos y bloques de datos.
    - Información de journaling (registro de transacciones).
  - Esta info es fundamental para que el SO interprete correctamente el sistema de archivos. Además, ext4 almacena copias del superbloque en varias partes del disco para una recuperación rápida en caso de errores.

Sistema de archivos ext4.

- **Estructura básica de ext4.**
  - **Inodos:** Cada archivo y directorio en ext4 se representa mediante un inodo, que es una estructura de datos que almacena la información de metadatos del archivo, como:
    - Permisos de acceso y propietario.
    - Fechas de creación, modificación y acceso.
    - Tamaño del archivo.
    - Punteros a los bloques de datos donde se almacena el contenido del archivo.
  - Los inodos permiten al sistema identificar y acceder a los archivos de manera eficiente, ya que ext4 utiliza una estructura de árbol para organizarlos.

Sistema de archivos ext4.

- **Estructura básica de ext4.**
  - **Bloques de datos:** Son las unidades donde se almacena el contenido real de los archivos.
  - Cada inodo contiene punteros a los bloques de datos correspondientes, lo que permite al sistema acceder a los datos de manera rápida.
  - En ext4, los bloques de datos suelen tener un tamaño de 4 KB, aunque pueden ajustarse para optimizar el rendimiento según el uso del sistema de archivos.

Sistema de archivos ext4.

- **Ventajas y características de ext4.**
  - **Journaling:** reduce la probabilidad de pérdida de datos.
  - **Soporte para archivos y filesystems grandes:**
    - Sistemas de archivos hasta **1 exababyte** (10^{18} bytes, 10^{6} TB).
    - Archivos individuales hasta **16 TB**.
  - **Asignación de bloques extendida:** llamada *delayed allocation*, reduce fragmentación y mejora el rendimiento.
  - **Extents:** grupos de bloques contiguos.
  - **Montaje más rápido:** estructura de metadatos optimizada.
  - **Retrocompatibilidad:** soporta las versiones 2 y 3.

Estructura de directorios en GNU/Linux.

- Se organiza siguiendo el **Estándar de Jerarquía de Sistema de Archivos (FHS, Filesystem Hierarchy Standard)**.
- Define una jerarquía lógica para los directorios y sus contenidos.
- Es consistente en la mayoría de las distribuciones de Linux
- Facilita la navegación y administración del sistema.

Estructura de directorios en GNU/Linux.

- **Estándar de Jerarquía de Sistema de Archivos (FHS).**
  - Establece una disposición estándar de los directorios y archivos en el sistema.
  - Garantiza que tanto usuarios como aplicaciones sepan dónde encontrar o ubicar archivos específicos sin importar la distribución de Linux que utilicen.
  - Este estándar categoriza los directorios en dos grandes grupos:
    - **Directorios compartibles y no compartibles.**
    - **Directorios variables y estáticos.**

Estructura de directorios en GNU/Linux.

- **Estándar de Jerarquía de Sistema de Archivos (FHS).**
  - **Directorios compartibles y no compartibles:**
    - **Compartibles:** Contienen archivos que pueden compartirse entre distintos sistemas (por ejemplo, `/usr y /home`).
    - **No compartibles:** Contienen archivos específicos de cada sistema, como los archivos de configuración en `/etc`.
  - **Directorios variables y estáticos:**
    - **Variables:** Directorios con datos que cambian frecuentemente, como logs y archivos temporales (por ejemplo, `/var`).
    - **Estáticos:** Contienen datos que rara vez cambian, como archivos binarios y de configuración (por ejemplo, `/usr/bin`).

Estructura de directorios en GNU/Linux.

- **Descripción de directorios principales.**
  - **/ (Raíz):** La raíz del sistema de archivos. Solo los administradores pueden escribir en la raíz.
  - **/bin:** Tiene binarios esenciales, como comandos básicos (ls, cp, mv) para el arranque y recuperación. Disponibles para todos los usuarios.
  - **/boot:** Almacena archivos para el arranque del sistema, incluyendo el kernel, el cargador de arranque (GRUB, LILO), y archivos de config.
  - **/dev:** Contiene archivos de dispositivos que representan el hw, como discos (/dev/sda).
  - **/etc:** Almacena archivos de config. y de aplicaciones. Son críticos para la config. de servicios, redes, y aplicaciones del sistema.

Estructura de directorios en GNU/Linux.

- **Descripción de directorios principales**.
  - **/home:** donde se almacenan los archivos personales de los usuarios del sistema, con sus configuraciones.
  - **/lib:** contiene bibliotecas compartidas por los binarios en `/bin` y `/sbin`. Permiten que los programas se ejecuten correctamente.
  - **/media:** utilizado para montar dispositivos extraíbles, (USB, HDD externos y CD/DVD).
  - **/mnt:** punto de montaje temporal para sistemas de archivos.
  - **/opt:** Almacena aplicaciones instaladas por el usuario o el admin. Suelen contener paquetes de sw que no forman parte de la instalación estándar del sistema y necesitan directorios separados.

Estructura de directorios en GNU/Linux.

- **Descripción de directorios principales**.
  - **/proc:** directorio virtual que contiene información sobre los procesos en ejecución y el estado del sistema.
  - **/root:** directorio personal del usuario `root`.
  - **/run:** contiene archivos de info del sistema desde el último arranque, como sockets y temporales para la inicialización de servicios.
  - **/sbin:** contiene los binarios de comandos avanzados, ejecutables únicamente por el usuario root.
  - **/srv:** tiene datos específicos de servicios ofrecidos por el sistema.
  - **/tmp:** archivos temporales. Se eliminan periódicamente.

Estructura de directorios en GNU/Linux.

- **Descripción de directorios principales.**
  - **/ usr:** almacena apps y arch. de usuario. Tiene los subdirectorios:
    - **/usr/bin:** binarios disponibles para todos los usuarios.
    - **/usr/lib:** bibliotecas compartidas por las app en `/usr/bin`.
    - **/usr/local:** sw instalado manualmente.
    - **/usr/share:** datos compartidos (documentación, configuración).
  - **/var:** contiene archivos que cambian con frecuencia:
    - Logs del sistema en `/var/log`.
    - Archivos de correos electrónicos en `/var/mail`.
    - Directorios de almacenamiento temporal en `/var/tmp`.

Tipos de archivos en GNU/Linux.

- **Archivos normales.**
  - Contienen datos de usuario o del sistema. Se identifican con un guión al comienzo de la salida del comando `ls -l`:
    - `-rw-r--r-- 1 user user 1234 Oct 23 12:00 archivo.txt`
  - Pueden ser de texto, imágenes, ejecutables, etc.
- **Directorios.**
  - Carpetas que contienen otros archivos y subdirectorios. Se identifican con una `d`:
    - `drwxr-xr-x 2 user user 4096 Oct 23 12:00 documentos`
  - Se utilizan para organizar el sistema de archivos y crear una jerarquía de directorios.

Tipos de archivos en GNU/Linux.

- **Archivos especiales.**
  - Utilizados para representar el hardware del sistema. Hay dos tipos:
    - **Dispositivos de bloque:** representan dispositivos de almacenamiento que se leen y escriben en bloques (discos, unidades USB, etc). Se distinguen con una `b`:
      - `brw-rw---- 1 root disk 8, 1 Oct 23 12:00 /dev/sda1`
    - **Dispositivos de carácter:** representan dispositivos orientados a caracteres, como teclados, impresoras, etc. Se pueden distinguir con una `c`:
      - `crw-rw-r-- 1 root tty 4, 1 Oct 23 12:00 /dev/tty1`

Tipos de archivos en GNU/Linux.

- **Enlaces.**
  - Permiten crear "alias" o "accesos directos" a archivos o directorios. Existen dos tipos principales de enlaces:
    - **Enlaces duros (Hard Links):** apunta a un inodo específico. Son prácticamente idénticos al archivo original, y el archivo se elimina cuando se eliminan todos sus enlaces duros. Solo pueden crearse dentro del mismo filesystem y no pueden apuntar a directorios.
    - **Enlaces simbólicos (Soft Links o Symbolic Links):** es un archivo que apunta a la ruta de otro archivo o directorio. Funcionan como accesos directos. Si el archivo original es eliminado, el enlace simbólico queda roto y no apunta a nada.

Tipos de archivos en GNU/Linux.

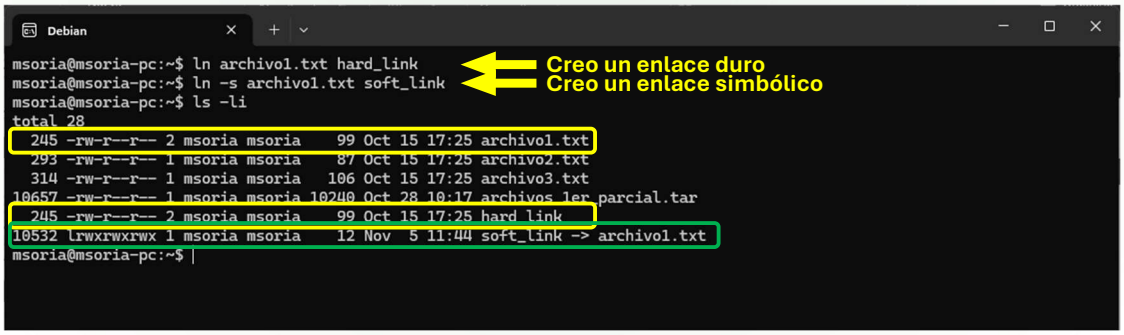
- **Enlaces.**
  - Para crear enlaces se utiliza el comando `ln`.
    - **Enlaces duros:** la sintaxis es
      - `ln archivo_original enlace_duro`.
    - **Enlaces simbólicos:** la sintaxis es
      - `ln -s archivo_original enlace_simbólico`.
  - Los enlaces duros tienen el mismo número de inodo que el archivo original, y no es posible diferenciarlos técnicamente.
  - Los enlaces simbólicos tienen un inodo diferente y se distinguen con una `l` delante al usar `ls -l`.

Tipos de archivos en GNU/Linux.

- **Enlaces.**
  - Un ejemplo de enlaces: se utiliza el comando `ln`.
    - **Enlaces duros:** la sintaxis es
      - `ln archivo_original enlace_duro`.
    - **Enlaces simbólicos:** la sintaxis es
      - `ln -s archivo_original enlace_simbólico`.
  - Los enlaces duros tienen el mismo número de inodo que el archivo original, y no es posible diferenciarlos técnicamente.
  - Los enlaces simbólicos tienen un inodo diferente y se distinguen con una `l` delante al usar `ls -li`.

Tipos de archivos en GNU/Linux.

- **Enlaces.**



Montaje de sistemas de archivos.

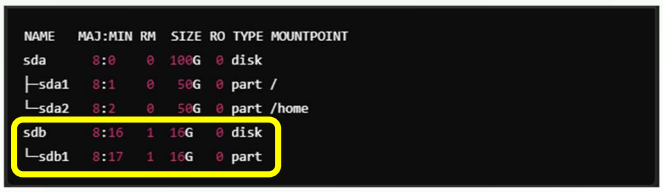
- En GNU/Linux, el **montaje de sistemas de archivos** es el proceso mediante el cual el sistema operativo hace accesible un dispositivo de almacenamiento en una ubicación específica del sistema de archivos.
- Esto se realiza mediante el comando `mount`, que asocia el dispositivo con un directorio llamado **punto de montaje**.
- Esto se aplica típicamente a los dispositivos extraíbles como las unidades USB, CD/DVD, tarjetas SD, etc., pero también a dispositivos de almacenamiento “fijos” como un disco duro o un SSD.

Montaje de sistemas de archivos.

- **Uso del comando *mount*.**
  - El comando `mount` permite al administrador del sistema montar dispositivos y sistemas de archivos en puntos específicos del sistema. La sintaxis básica es:
  - `mount [opciones] dispositivo punto_de_montaje`
    - **dispositivo:** Es la ruta del dispositivo de almacenamiento, generalmente ubicado en el directorio /dev (por ejemplo, /dev/sda1 para la primera partición del primer disco duro).
    - **punto_de_montaje:** Es el directorio donde se montará el dispositivo. Una vez montado, los archivos y carpetas del dispositivo serán accesibles desde este punto.

Montaje de sistemas de archivos.

- **Uso del comando *mount*.**
  - El punto de montaje **debe existir** antes de montar un filesystem.
  - Por defecto, si no está configurado el automontaje de filesystems, y si el dispositivo es Plug&Play, luego de conectarlo, el SO crea un archivo de dispositivo en `/dev`, como `/dev/sdb` o similar.
  - En este caso, para identificar el dispositivo, se puede utilizar el comando:
  - `lsblk`



Montaje de sistemas de archivos.

- **Opciones comunes de *mount*.**
  - **Especificar el tipo de sistema de archivos:** Si el sistema de archivos no se detecta automáticamente, puedes especificarlo con `-t`. Por ejemplo, para montar un dispositivo en formato NTFS:
    - `sudo mount -t ntfs /dev/sdb1 /media/usb`
  - **Montaje de solo lectura:** Para montar un dispositivo en modo solo lectura, utiliza la opción `-o ro`:
    - `sudo mount -o ro /dev/sdb1 /media/usb`
  - **Especificar opciones de montaje:** Puedes pasar opciones adicionales con `-o`, como permisos o límites de tamaño:
    - `sudo mount -o uid=1000,gid=1000 /dev/sdb1 /media/usb`

Montaje de sistemas de archivos.

- **Verificación del montaje.**
  - **Comando *df*:** con la opción -h:
    - `df -h`

```
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1        50G   10G   38G  21% /
/dev/sdb1       100G   20G   80G  20% /media/usb
```

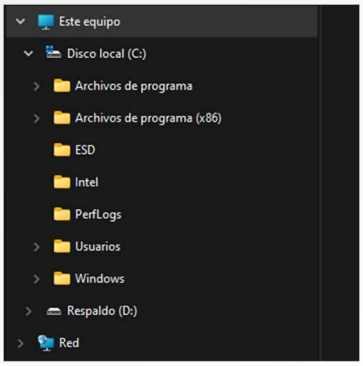
  - **Comando *mount*:** sin argumentos:
    - `mount`

```
/dev/sda1 on / type ext4 (rw,relatime)
/dev/sdb1 on /media/usb type ntfs (rw,relatime)
```

- **Desmontar un sistema de archivos.**
  - `sudo umount /media/usb`

Montaje de sistemas de archivos.

- Comparación de sistemas de archivos: Windows vs. Linux.



Fin del Módulo 5