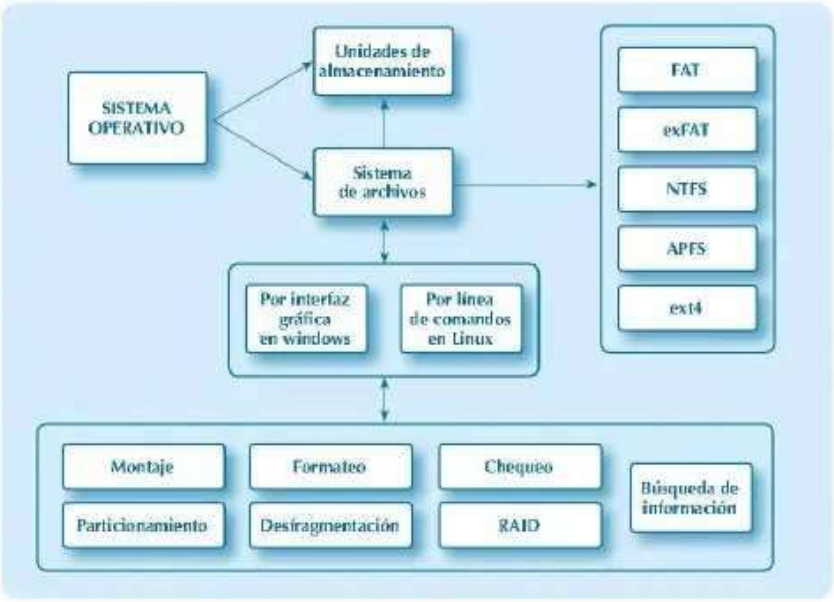
3

Sistemas operativos. Gestión de archivos y almacenamiento

Objetivos

- ✓ Conocer y comparar los sistemas de archivos más empleados.
- ✓ Reconocer la estructura y funciones de los directorios de los sistemas operativos más empleados.
- ✓ Crear e identificar diferentes tipos de particiones y unidades lógicas.
- ✓ Instalar y evaluar utilidades relacionadas con la gestión del almacenamiento e información.
- ✓ Operar con distintas herramientas, por comandos o entorno gráfico, para localizar archivos e información en el sistema de archivos.

Mapa conceptual



Glosario

Archivo. La unidad lógica mínima que contiene información.

Desfragmentación. Proceso de unión de bloques de datos de un mismo archivo, evitando la disgregación o esparcimiento de estos entre sí en el medio de almacenamiento.

Directorio. Contenedores lógicos de ficheros o directorios.

Enlace simbólico. Tipo de archivo en sistemas de archivos *ext4*, entre otros, que referencian a otros archivos o directorios del mismo u otro sistema de archivos.

Estructura de directorios de un sistema operativo. Conjunto de directorios que constituyen la organización y despliegan los archivos y subdirectorios del propio sistema operativo.

Formateo. Proceso por el que se instala un sistema de archivos en una partición.

i-nodo. Estructura de datos, en sistemas de archivos *ext4*, entre otros, que almacenan meta-información del archivo que representan.

Journaling. Registro del sistema de archivos que evita la inconsistencia de ficheros y facilita la recuperación de datos en los medios de almacenamiento.

Partición. División interna del dispositivo de almacenamiento que permite organizar la información y facilitar la gestión del almacenamiento.

RAID. Configuración de un grupo de discos independientes para aumentar la integridad, la capacidad de almacenamiento, la velocidad de transferencia o disminuir el riesgo a fallos.

3.1. Introducción

Al igual que la gestión de procesos y la gestión de usuarios, la gestión de los archivos es uno de los pilares fundamentales de cualquier sistema operativo.

Los sistemas operativos son capaces de gestionar la información contenida en los medios de almacenamiento gracias a los sistemas de archivos. Estos proveen la manera de almacenar la información, así como mecanismos que permitan realizar operaciones sobre ella.

Existen multitud de sistemas de archivos que confieren diferentes características al espacio de almacenamiento y repercuten en la seguridad de los datos, su rendimiento o su gestión. En este tema estudiaremos diferentes sistemas operativos, centrándonos en los más usados: *ext4* y *NTFS*.

Los sistemas operativos proveen herramientas para la gestión del almacenamiento, ya sea por línea de comandos o por interfaz gráfica, para realizar particiones de los medios de almacenamiento, formatear, montar y desmontar los sistemas de archivos, desfragmentar, chequear los sistemas de archivos, buscar información e incluso crear diferentes esquemas RAID.

Para trabajar la gestión de archivos y almacenamiento, a lo largo del tema trabajaremos con los sistemas operativos *Microsoft Windows* a través de su interfaz gráfica y con *Ubuntu* mediante su interfaz textual. Sobre este último recaerá la mayoría del peso de los aspectos que se van a estudiar, ya que consiste en una herramienta fundamental para usuarios expertos o administradores de sistemas operativos.

3.2. Sistemas de archivos

Los sistemas de archivos emplean el *archivo (fichero)* como la herramienta fundamental de abstracción lógica de la información. Se dice que un archivo es, por tanto, la unidad lógica mínima de almacenamiento que contiene información. Es decir, los archivos se emplean para evitar que el usuario conozca la estructura interna y las propiedades características de los medios de almacenamiento, facilitando la gestión y organización por su parte.

Otro elemento empleado por los sistemas de archivos son los *directorios (carpetas)*. Estos son ficheros que actúan de contenedores lógicos de ficheros o directorios. Independientemente del sistema de archivos donde se defina el directorio, este almacena información relativa a la localización física de la información y los atributos propios de cada archivo o directorio que contenga.

Los sistemas de archivos tienen como objetivo:

- Acceder a la información de los ficheros.
- Crear, eliminar y modificar ficheros.
- Acceder a los ficheros mediante diferentes protocolos de comunicación en red u otros ficheros.

- Facilitar el acceso multiusuario.
- Facilitar el acceso a multitud de medios de almacenamiento.
- Realizar copias de seguridad.
- Utilizar herramientas de recuperación de información.
- Priorizar la eficiencia y la seguridad de acceso a la información.
- Maximizar el rendimiento en las operaciones sobre los archivos.
- Permitir la monitorización y contabilidad sobre ficheros.
- Administrar el espacio de almacenamiento, gestionar la asignación del espacio libre y el espacio ocupado de los archivos.

Para administrar el espacio libre (susceptible de ser asignado) y el espacio ocupado, se han de definir espacios de asignación. Estas son las unidades físicas mínimas de almacenamiento gestionadas por los sistemas de archivos. Los sistemas de archivos definen el tamaño del espacio de asignación (también llamado *unidad de asignación* o *clúster*) durante la instalación del propio sistema de archivos (formateo). Este espacio determina el tamaño mínimo que ocupará un archivo en el medio de almacenamiento. A nivel físico (hardware), el dispositivo administra sectores, sin embargo, el sistema de archivos gestiona clústeres.

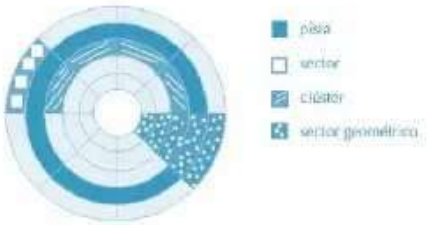


Figura 3.1
Esquema físico de un disco duro mecánico.

La planificación de este espacio es muy importante, siendo ideal un equilibrio entre el promedio del tamaño de los archivos que vaya a alojar el sistema de archivos (evitando así que se desperdicie espacio interno al clúster, también conocido como *fragmentación interna*) y el tamaño del volumen (facilitando la administración del sistema de archivos). Si el usuario no define el tamaño del clúster, el sistema operativo lo asignará automáticamente, atendiendo al tamaño del disco, el tipo de sistema de archivos y el esquema de particionamiento (*MBR* o *UEFI*).

RECUERDA

✓ En Microsoft Windows, al formatear una unidad (pulsando el botón secundario del ratón sobre ella), se puede indicar el tamaño de unidad de asignación.

Figura 3.2
Indicación de la unidad de asignación al formatear una unidad en Microsoft Windows.

3.2.1. FAT (File Allocation Table)

Es un sistema de archivos creado para el sistema operativo MS-DOS. La administración del espacio del almacenamiento es sencilla, por lo que se convierte en un sistema de archivos muy extendido en la mayoría de los sistemas operativos. De tal manera, que se suele emplear en dispositivos utilizados para intercambiar datos en computadoras con varios sistemas operativos, videoconsolas, televisores, etc.

Las principales limitaciones de FAT32 son:

- ✓ Imposibilidad de gestionar particiones superiores a 8 TB (32 GB en Microsoft Windows) y archivos de más de 4 GB.
- ✓ Bajo rendimiento.
- ✓ Inseguro: no permite encriptación, sus atributos y permisos son limitados y no permite *journaling*.

TOMA NOTA

Journaling es un registro o diario del sistema de archivos. Los sistemas de archivos que hacen uso de este sistema se denominan transaccionales. Consiste en registrar una serie de acciones previas a la operación sobre el sistema de archivos que se vaya a realizar. De tal manera, que si se produce un error durante alguna operación (copiado, borrado, creación, etc.), el sistema puede deshacer los cambios y volver a la situación original. Así se evitan muchas situaciones de inconsistencia de ficheros, facilitando la recuperación de datos y chequeos de medios de almacenamiento.

3.2.2. exFAT

Resultado de una evolución del sistema de archivos FAT32, que elimina sus principales limitaciones. Se pueden tratar archivos de hasta 16 EB y mantiene la ligereza frente a sistemas de archivos más avanzados, como NTFS y APFS. Aunque sigue resultando inseguro, es ideal para medios de almacenamiento FLASH portables con gran capacidad y compatibles entre sistemas operativos.

3.2.3. NTFS

Se considera el sistema de archivos estándar de Microsoft Windows. Sus mejoras son considerables con respecto a FAT32, primando la seguridad y la confiabilidad. Sus principales ventajas son: todas

- Emplea *journaling*. Favoreciendo una pronta recuperación ante errores inesperados.
- Permite cifrado y compresión.
- Reduce significativamente la fragmentación y aumenta la velocidad de búsqueda de archivos con respecto a FAT32.
- Puede llegar a gestionar volúmenes de hasta 16 EB y archivos de hasta 16 TB.
- Emplea Unicode para el nombre de archivos, con hasta 255 caracteres.

3.2.4. APFS

Sistema de archivos empleado por Apple Inc. para sus medios de almacenamiento, que supone una versión mejorada de su predecesora HFS+. Sus características son similares a NTFS y ext4, por lo que permite administrar archivos y volúmenes de hasta 8 EB. Permite encriptación y está optimizado para almacenamiento Flash.

Recurso digital 3.1

Esquemas de partición con sistemas de archivos FAT, NTFS y ext4.

3.2.5. ext4 (Fourth extended file system)

Sistema de archivos predeterminado para sistemas operativos de tipo Linux en su cuarta versión. Incluye journaling, maneja archivos de hasta 16 TB y volúmenes de hasta 1 EB. Supera a sus antecesores ext2 y ext3, ya que:

- ✓ Mejora el rendimiento.
- ✓ Reduce la fragmentación.
- ✓ Permite trabajar con ficheros de mayor tamaño gracias al uso de *extents*.

A diferencia de NTFS, ext4 no emplea extensiones como parte del nombre de los archivos. En Microsoft Windows, un nombre de archivo se divide en *<nombre>.<extensión>*, donde *extensión* es un conjunto de caracteres (normalmente tres o cuatro) que se asocian con programas para que el sistema operativo reconozca la manera de ejecutar el archivo.

Ejemplos de extensiones muy usadas son:

- *.c*: fichero fuente en lenguaje de programación C.
- *.txt*: fichero de texto ASCII.
- *.tar*: fichero resultado de utilizar el comando tar.
- *.html*: fichero de lenguaje de marcas de hipertexto.

No obstante, muchos nombres de archivos en Linux incorporan sufijos separados por un punto en el nombre del fichero por convención, pero no como requisito establecido por el sistema de archivos.

Una partición con sistema de archivos *ext4* se divide en *grupos de bloques*. Cada grupo de bloques se divide, a su vez, en las siguientes partes:

a) *Superbloque*: contiene la información más relevante del grupo de bloques.
b) *Descriptores de grupos*: almacena la información más importante del resto de bloques.
c) *Bitmap de bloques de datos*: contiene un mapa de bits donde se representa cada clúster, así como su estado (libre u ocupado).

d) *Bitmap de i-nodos*: mapa de bits representando a cada *i-nodo*, que además indica su estado (libre u ocupado).
e) *Tabla de i-nodos*: tabla que contiene una entrada por cada *i-nodo*. El i-nodo almacena la información propia de cada archivo.
f) *Bloques de datos*: clústers con información. Cada bloque de datos está asociado con un archivo.

La estructura fundamental es el *i-nodo* o *nodo índice*. Este almacena toda la metainformación asociada al archivo que representa: tipo de archivo, propietario, tamaño, fechas, número de bloques de datos, localización de los bloques de datos, etc.

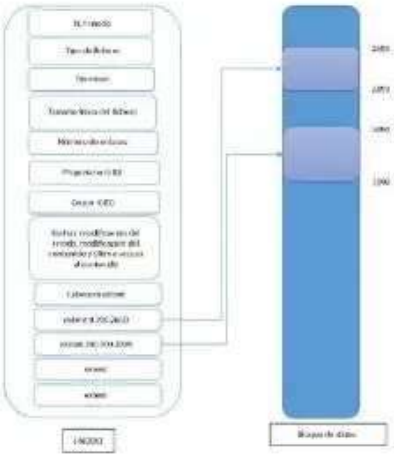


Figura 3.3
Estructura de i-nodo en ext4.

TEN EN CUENTA

✓ Es importante la compatibilidad entre sistemas de archivos y sistemas operativos. Los sistemas de archivos ext4 y APFS no son reconocidos por Microsoft Windows, sin embargo, se puede emplear NTFS en la mayoría de distribuciones Linux. Por otro lado, exFAT se considera ideal para medios extraíbles de tipo FLASH por su compatibilidad entre la mayoría de sistemas operativos.

En ext4, el i-nodo emplea extents, los cuales intervienen con ficheros grandes. Estos emplean un conjunto de clústers contiguos (en lugar de estar separados) y se gestionan de manera simple, almacenando: el inicio del bloque dentro del archivo, el número de bloques almacenados y el inicio del número de bloque físico en disco.

Un i-nodo está compuesto por:

✓ Un identificador único de i-nodo o número de i-nodo. Único en el sistema de archivos.
✓ Tipo de fichero: regular, enlace simbólico, directorio o dispositivo.
✓ Permisos de lectura, escritura y ejecución para el propietario, grupo y otros usuarios.
✓ Tamaño del fichero (en bytes).
✓ Número de enlaces (duros). Hace referencia al número de veces que el i-nodo es referenciado en el árbol de directorios. Cuando se elimina un fichero de un directorio, decrementa en uno este número y se elimina la entrada correspondiente en dicho directorio. Así, ha de llegar a cero este número para que se elimine el i-nodo.
✓ Identificador del propietario del archivo (UID). Número identificativo único que representa a un usuario.

- ✓ Identificador del grupo (GID). Número identificativo único que representa a un grupo de usuarios.
- ✓ Fechas de última modificación de la meta-información del i-nodo *(ctime)*, última modificación de su contenido *(mtime)* y último acceso a su contenido *(atime)*.
- ✓ Cabecera extent. Contiene información asociada a los extents que lo siguen: número válido de entradas que siguen a la cabecera, máximo número de entradas que siguen a la cabecera y la profundidad del extent actual (0 apunta a bloques de datos y en otro caso hasta 4 apuntará a nodos extents intermedios que actúan como indirecciones). Si el archivo está más fragmentado que los cuatro extents del i-nodo, estos apuntarán a nodos índices del árbol de extent hasta llegar a hojas extent.
- ✓ Cuatro nodos hoja extent que contienen el inicio del bloque dentro del archivo, el número de bloques almacenados y el inicio del número de bloque físico en disco.

3.3. Estructura de directorios en Linux y Microsoft Windows

Cuando los sistemas operativos GNU/Linux y Microsoft Windows son instalados en medios de almacenamiento, crean un conjunto de directorios donde se despliegan los archivos y subdirectorios del propio sistema operativo en una estructura en forma de árbol invertido del sistema de archivos. Esto es, un directorio del que cuelgan un conjunto de subdirectorios y, dentro de estos últimos, se alojan otros subdirectorios, y así sucesivamente.

RECUERDA

- ✓ En ambos, el directorio principal del que cuelgan el resto de directorios es el *directorio raíz*, simbolizado en Microsoft Windows por "\" y en Linux por "/".

De esta manera, para hacer referencia a la localización de un directorio dentro de la estructura arbórea, se emplea el término *ruta, camino o path*. Así, se localiza fácilmente un archivo o directorio. Por ejemplo:

- En Linux podemos hacer referencia a la ruta de un directorio como */home/usuario 2/* donde "/" es el directorio raíz, "*home*" es un subdirectorio del directorio raíz y "*usuario 2*" es un subdirectorio del directorio *home*.
- En Microsoft Windows se puede indicar un directorio como *C:\Documents and Settings*. En Microsoft Windows el directorio raíz se antecede por una letra que simboliza una unidad o volumen. "*C:*" indica el directorio raíz de la unidad C. Y "*Documents and Settings*" es un subdirectorio del directorio raíz.

El *directorio de trabajo actual* es el directorio donde se encuentra situado actualmente, al ir navegando por los directorios y subdirectorios de una unidad, dentro del árbol de directorios. El directorio de trabajo actual se simboliza por ".". En Linux podemos mostrar la ruta de trabajo actual mediante el comando *pwd* (print working directory).

Por *directorio padre* de un directorio se conoce al directorio que se encuentra por encima de él en la estructura jerárquica del árbol de directorios. Se simboliza por "..". De la misma manera,

un *directorio hijo* de un directorio es el directorio que se encuentra por debajo del primero en la estructura jerárquica.

La ruta de un archivo o directorio se puede indicar de dos maneras:

1. Ruta absoluta: ruta completa indicada desde el directorio raíz.
2. Ruta relativa: ruta indicada desde el directorio de trabajo actual. Se suele hacer uso de ".." y "." para desplazarse.

El comando *cd* (change directory) permite cambiar de directorio. Su sintaxis es la siguiente: *cd [directorio]*, si no se indica directorio, cambiará al directorio home del usuario y equivaldría a ejecutar cd ~.

Actividad resuelta 3.1

Nos encontramos en /home/luis como directorio actual y queremos acceder al directorio raíz. ¿Cómo se puede indicar el cambio de directorio mediante ruta absoluta y relativa?

SOLUCIÓN

Ruta Absoluta: cd /
Ruta relativa: cd ../..

3.3.1. Estructura de directorios en GNU/Linux

La mayoría de los sistemas operativos GNU/Linux siguen el estándar *FHS (Filesystem Hierarchy Standard)* que define el contenido y las funciones de los directorios en la estructura jerárquica del árbol de directorios. Los directorios más importantes son:

- ✓ /: Directorio raíz o *root*. Todos los directorios y subdirectorios parten de este directorio raíz.
- ✓ /bin: contiene los archivos binarios (ejecutables) a nivel de usuario.
- ✓ /boot: almacena los archivos ejecutables y de configuración necesarios para el arranque del sistema.
- ✓ /dev: se encuentran los componentes del sistema y todos los dispositivos de almacenamiento representados por archivos (memorias FLASH, particiones, DVD, etc.). A través de este directorio podemos acceder a la información propia del medio de almacenamiento.
- ✓ /etc: almacena los archivos de configuración globales del sistema y que afectan a todos los usuarios.
- ✓ /home: directorio que aloja los directorios de los diferentes usuarios del sistema, a excepción del usuario root (el cual emplea /root).
- ✓ /lib: contiene archivos muy importantes para el sistema operativo, como librerías y módulos del kernel.
- ✓ /media: directorio que se emplea para montar dispositivos como discos duros o medios removibles como CD o DVD.

- ✓ /mnt: también utilizado para albergar puntos de montaje pero, en este caso, temporales, como, por ejemplo, un sistema de archivo en red o carpetas compartidas en máquinas virtuales.
- ✓ /proc: directorio empleado por el sistema para guardar información relativa a los procesos y al kernel del sistema. No contiene archivos físicos, sino que se generan sobre la marcha archivos virtuales.
- ✓ /sys: al igual que /proc, almacenan archivos virtuales relativos al kernel del sistema, así como información de drivers y dispositivos.
- ✓ /sbin: almacena ejecutables que se suelen emplear para tareas administrativas por el superusuario.
- ✓ /tmp: las aplicaciones emplean esta carpeta para almacenar archivos temporales.
- ✓ /usr: almacena archivos de solo lectura de la mayoría de las aplicaciones y utilidades instaladas en el sistema.
- ✓ /opt: contiene el resto de aplicaciones no almacenas en /usr. Normalmente, son aquellas que no son parte de los paquetes instalados con la distribución Linux objeto de uso.
- ✓ /srv: directorio encargado de alojar datos, scripts y carpetas para servidores instalados en nuestro sistema (servidores web, ftp, repositorios, etc.).
- ✓ /var: directorio considerado como registro del sistema. En él se incluye información del sistema, logs, información de caché, etc.

Actividad propuesta 3.1

Busca imágenes en Internet con la estructura de directorios de los sistemas operativos Microsoft Windows y Ubuntu (en sus versiones más actuales). Compáralos. El 'Escritorio', en ambos, es una carpeta que contiene ficheros o directorios. Para un usuario cualquiera, ¿cuál es la ruta del 'Escritorio' en ambos sistemas operativos?

3.3.2. Estructura de directorios en Microsoft Windows

El directorio raíz de una partición con Microsoft Windows instalado dispone del siguiente árbol de directorios:

- \Archivos de programa (Program Files): en sistemas de 32 bits, se encuentran todos los programas instalados. Sin embargo, en sistemas de 64 bits se almacenarán las aplicaciones de 64 bits.
- \Archivos de programa (x86): esta carpeta se encuentra en sistemas de 64 bits y contiene las aplicaciones instaladas de 32 bits.
- \PerfLogs: por defecto, está vacía, pero puede contener registros de rendimiento del sistema.
- \ProgramData: carpeta oculta que contiene datos de programas genéricos para todos los usuarios del sistema.
- \Usuarios (Users): carpeta que contiene subcarpetas por cada usuario del sistema, así como la carpeta \Acceso público (\Public) y \Default (carpeta oculta):
  - \Acceso público: carpeta compartida por todos los usuarios del sistema donde se definen aspectos comunes a ellos. Por defecto, está compartida en red.

- \Default: contiene el perfil base sobre el que se crean nuevos perfiles en el sistema. Así, al crear un nuevo usuario, su carpeta situada en C:\Usuarios, contendrá el perfil y la estructura definida en \Default.
- \[nombreUsuario]: contiene un conjunto de carpetas que definen el perfil del usuario (conjunto de valores de configuración del entorno: escritorio, aplicaciones, impresoras, conexiones de red, etc.), así como la carpeta oculta AppData. Esta última carpeta contiene los datos de las aplicaciones asociadas al usuario en sí (a diferencia de \ProgramData). En ella se encuentran tres subcarpetas: Roaming, que aloja perfiles de configuración de aplicaciones que puedan sincronizarse entre equipos; Local y LocalLow, que almacenan el resto de archivos empleados por las aplicaciones.

- Windows: contiene el grueso de la instalación del sistema operativo. En esta carpeta destacan las subcarpetas:
  - \System32: contiene archivos DLL de 32 bits o 64 bits, dependiendo de si la versión de Microsoft Windows es de 32 bits o 64 bits respectivamente.
  - \SysWOW64: solo en las versiones de 64 bits para almacenar archivos DLL de 32 bits.
  - \WinSxS: conocido como el almacén de componentes de Microsoft Windows, que contiene archivos utilizados para la instalación, las actualizaciones del sistema, los Service Packs o las características de Microsoft Microsoft Windows.

SABÍAS QUE...

Las bibliotecas de vínculos dinámicos (DLL) son archivos que contienen código ejecutable y datos. Microsoft Windows los emplea frecuentemente, ya que aumenta la modularidad en las aplicaciones, ahorra recursos del sistema y simplifica la instalación y la ejecución de las aplicaciones.

Actividades propuestas

3.2. Navega por la estructura de directorios de Ubuntu y lista las carpetas aquí estudiadas. Para mayor detalle, puedes hacer uso del comando *man hier* en un terminal de Linux, el cual especificará la utilidad de cada carpeta.

3.3. Navega por la estructura de directorios de Microsoft Windows y lista las carpetas aquí estudiadas, observando su contenido.

3.4. Gestión de archivos por línea de comandos en Linux

En entornos domésticos o de oficina se emplea la interfaz gráfica para el manejo y gestión de archivos. No obstante, los administradores de sistemas suelen hacer uso de la interfaz por línea de comandos, ya que su versatilidad y potencia de uso la convierte en una herramienta ideal.

Los comandos de Linux siguen una sintaxis:

```
comando [opciones] [argumentos]
```

Cada comando varía el tipo de opciones y argumentos que pueda utilizar, e incluso puede no utilizarse ninguno en caso de ser opcionales. Las opciones pueden ser cortas (una sola letra) o largas (una palabra), antecedidas por uno o dos guiones. Se pueden emplear ambos tipos de opciones con un mismo comando, aunque las opciones cortas pueden unirse. Tanto los comandos como las opciones y los argumentos que se usen son sensibles a mayúsculas y minúsculas.

Uno de los comandos que más se emplea es *ls* (list).

```
ls [opciones] [ficheros]
```

Permite listar el contenido de un directorio e información de archivos. Sus opciones más utilizadas son las siguientes:

- ✓ l: muestra en formato largo.
- ✓ t: ordena por fecha de modificación.
- ✓ r: invierte el orden de salida.
- ✓ R: lista recursivamente el contenido de cada directorio.
- ✓ i: muestra el número de i-nodo.
- ✓ a: muestra los archivos ocultos. En Linux los archivos ocultos son aquellos que empiezan con ".". Si no indicamos esta opción, el comando *ls* no listará los archivos ocultos.
- ✓ h: muestra el tamaño de cada fichero en K, M, G, etc.
- ✓ size: muestra el tamaño de cada fichero en bloques.
- ✓ S: lista los archivos ordenados por tamaño.

Por defecto y sin argumentos, lista el directorio actual ordenado alfabéticamente. Si se indica más de uno, se listará el contenido de cada uno de ellos.

Cuando se emplea la opción "*– l*", aparecen clasificadas en columnas la información propia de cada fichero listado:

1. La primera columna es la lista de control de acceso o *máscara de permisos*. A su vez, se divide en dos partes:
   a) El primer carácter puede ser: "d" indicando un directorio, "l" un enlace simbólico, "-" un archivo regular, "c" un dispositivo de tipo carácter, o "b" un dispositivo de tipo bloque.
   b) El resto de caracteres son los permisos asociados al usuario, al grupo y a otros, tomados en grupos de tres.
2. La segunda columna establece el número de enlaces duros asociados al archivo.

3 y 4. La tercera y cuarta columna indican el propietario y el grupo, respectivamente.

5. La quinta columna hace mención al tamaño que ocupa en bytes.
6. La sexta columna se refiere a la fecha de la última modificación de su contenido *(mtime)* o de su creación (si no se ha modificado).

7. La séptima y última columna se refiere al propio nombre del fichero. El nombre del fichero no reside en el i-nodo, sino en el directorio.

TOMA NOTA

En Linux, los nombres de los ficheros pueden tener una extensión entre 1 y 255 caracteres, pero, además, debemos conocer que:

- No se puede emplear el carácter "/".
- No es recomendable emplear los caracteres: =, ^, ~, ', ", `, *, -, ?, [,], (,), !, &, ~, <, >. Esto se debe a que tienen un significado especial en el terminal. No obstante, podemos emplearlos, pero entrecomillando el nombre del fichero.
- Un archivo puede contener espacios. Para indicar al terminal que es un espacio en lugar del siguiente argumento, se entrecomilla el nombre completo del fichero. También se puede anteceder cada espacio (se dice que se "escapa") con una barra invertida "\" o se usan apóstrofes simples.
- Los nombres no tienen por qué usar extensiones. No obstante, es recomendable para muchos de ellos emplearlas para identificar fácilmente el tipo de información que contienen.
- Si queremos ocultar un archivo, se antecede con un punto ".".

Cualquier archivo o directorio se localiza e identifica dentro del árbol de directorios gracias a su ruta. Por ello, no pueden existir dos archivos con el mismo nombre en un mismo directorio.

Además, todos los directorios disponen, al menos, de dos entradas ocultas: "." el directorio actual y ".." el directorio padre. Estas se definen automáticamente cuando se crea un directorio. El directorio actual hace referencia a él mismo y se emplea muy a menudo cuando se trabaja con rutas relativas, y también lo hacen por defecto multitud de comandos. También el directorio padre se utiliza necesariamente para poder moverse por el árbol de directorios.

Actividades propuestas

3.4. Ejecuta y explica las siguientes instrucciones: *ls /home /usr, ls -l /home, ls –R /home, ls –ltra.*

3.5. Ejecuta y explica las siguientes instrucciones: *cd . , cd .. y cd ./..*

3.4.1. Tipos de ficheros

Los tipos de ficheros en Linux son muy importantes. Cualquier elemento físico (discos, impresoras, etc.) o lógico (directorio, enlace, etc.) se representa en Linux mediante un archivo, lo que facilita y estandariza la gestión de todos ellos. Se distinguen cuatro tipos elementales de ficheros:

a) *Regulares.* Son ficheros ordinarios que contienen información de diversa naturaleza. Dentro de ellos se incluyen los ficheros ejecutables como un tipo de fichero regular que tiene activo algún permiso de ejecución y contienen código ejecutable.

b) *Directorios.* Almacenan en su bloque de datos el número de i-nodo y el nombre de los archivos que contiene. Cuando son listados con el comando *ls* muestra la información accediendo a la estructura interna de cada i-nodo.

Ejemplo

En la siguiente imagen se observa cómo el directorio actual "." (asociado al directorio /usr) tiene 131074 como número de i-nodo con 11 enlaces duros. Y el directorio padre del directorio actual (/usr) es el directorio raíz que tiene como número de i-nodo 2 con 24 enlaces duros. Su bloque de datos contiene los números de i-nodo y los nombres de los archivos que contiene.

```
luis@luis-VirtualBox:~$ ls -lai /usr
total 108
131074 drwxr-xr-x  11 root root  4096 oct 18 00:28 .
     2 drwxr-xr-x  24 root root  4096 dic 23 11:15 ..
131089 drwxr-xr-x   2 root root 45056 dic 12 22:59 bin
131058 drwxr-xr-x   2 root root  4096 oct 18 00:27 games
131091 drwxr-xr-x   8 root root  4096 nov 10 11:28 include
131092 drwxr-xr-x 127 root root  4096 nov 15 10:09 lib
131093 drwxr-xr-x   2 root root  4096 oct 18 00:28 libexec
131094 drwxr-xr-x  10 root root  4096 oct 18 00:23 local
131095 drwxr-xr-x   2 root root 12288 dic 12 22:59 sbin
131096 drwxr-xr-x 258 root root 12288 nov 14 23:48 share
131097 drwxr-xr-x   8 root root  4096 dic 23 11:14 src
```

Figura 3.4
Ejemplo de salida del comando ls con opciones -lai.

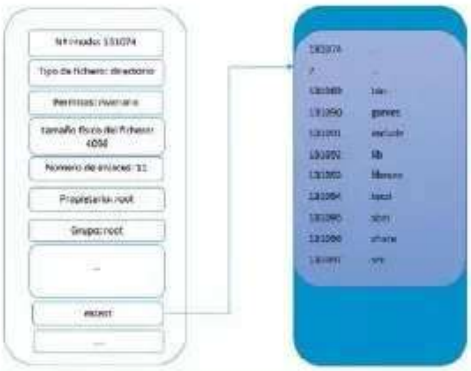


Figura 3.5
Estructura de i-nodo de ejemplo.

c) *Enlaces.* Existen dos tipos:

- *Enlaces duros.* Son asociaciones de nombres de ficheros a i-nodos. Es decir, se trata de reutilizar un i-nodo asignándole nombres distintos y localizándose en diferentes directorios o en el mismo. Los enlaces duros se establecen sobre ficheros regulares y no sobre directorios. Además, pueden hacer referencia solo dentro del mismo sistema de archivos. Para crear enlaces duros, se emplea el comando *ln*, y su sintaxis es la siguiente: *ln fichero fichero_enlace*.

- *Enlaces simbólicos o enlaces blandos.* El directorio que contiene un enlace simbólico almacena un i-nodo (diferente al i-nodo con el archivo que enlaza) y un nombre de fichero. Para acceder al fichero con el que enlaza, el i-nodo del enlace simbólico almacena en un campo la ruta de acceso al archivo destino, si esta es inferior a 60 bytes y, si es superior, se almacenará en el bloque de información de un *extent*. De esta manera, pueden referenciar archivos de otros sistemas de archivos, particiones y equipos en red. Para crear enlaces simbólicos, se debe incluir la opción "–s" con *ln*: *ln -s fichero fichero_enlace.*

Actividad resuelta 3.2

Crea un archivo llamado notas.txt sobre el que crearemos, posteriormente, un enlace duro llamado notas.txt.bck y un enlace simbólico notas.txt.s_bck.

SOLUCIÓN

Para crear un archivo nuevo sin contenido, podemos emplear el comando *touch*, siendo su sintaxis: *touch nuevo_fichero*

El comando *touch* también permite actualizar la fecha de acceso y modificación del fichero, si existe, sin necesidad de modificarlo o acceder a su contenido.

Primero creamos el fichero notas.txt: *touch notas.txt.*

A continuación, creamos el enlace duro *notas.txt.bck* asociado al archivo *notas txt*: *ln notas.txt notas.txt.bck*

Y, por último, creamos el enlace simbólico *notas.txt.s_bck* y listamos el contenido del directorio actual en formato largo, mostrando los números de i-nodo, como aparece en la siguiente imagen:

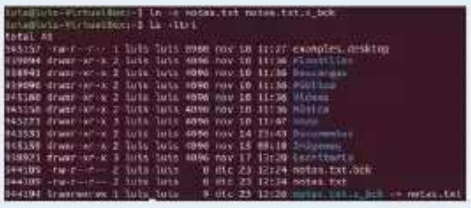


Figura 3.6
Creación de enlace simbólico.

Podemos observar cómo el número de i-nodo de *notas.txt* y *notas.txt.bck* es igual. No así el de *notas.txt.s_bck*. Además, este último indica que es simbólico, ya que "apunta" mediante los caracteres "- >" a *notas.txt* y además consta el carácter "*l*" en su máscara de permisos. En la siguiente imagen se detalla esquemáticamente la relación entre i-nodo, carpeta y enlace simbólico.

Figura 3.7
Relación entre i-nodo, carpeta y enlace simbólico.

Los enlaces duros, una vez creados, no se distinguen del archivo original, puesto que están asociados al mismo i-nodo. Por tanto, la eliminación del archivo original, manteniendo un enlace duro asociado, no afecta en nada al segundo. No así ocurre con los enlaces simbólicos, puesto que si eliminamos el archivo original (al no disponer de enlaces duros), el enlace blando se considera "roto", no pudiendo acceder a la información.

d) Dispositivos. Son archivos que representan a dispositivos físicos. En el directorio /dev se localiza la mayoría de ellos. Se distinguen dos tipos de archivos de dispositivos:

1. Dispositivos por caracteres: suelen ser aquellos que no disponen de sistema de archivos, como impresoras, teclados, terminales de texto, etc. Transfieren los datos carácter a carácter.
2. Dispositivos por bloques: almacenan la información en bloques de datos físicamente, como, por ejemplo: discos duros, cintas magnéticas, unidades flash, etc.

Además, existen dispositivos virtuales, los cuales tienen un tratamiento especial, como */dev/null*, también conocido como *cubeta de bits*, puesto que se suele emplear para eliminar o descartar toda información que sea enviada a él.

La propia orden ls dispone de la opción "- *-color*", que se encuentra activada por defecto en Ubuntu y permite discriminar el tipo de archivo según el color:

- Blanco: archivo regular.
- Verde: archivo ejecutable.
- Azul: directorio.
- Cian: enlace simbólico.
- Rojo: enlace roto.

3.4.2. Eliminación de ficheros

Se pueden eliminar archivos mediante la orden *rm*. Su sintaxis es la siguiente:

```
rm [-irf] lista_de_ficheros
```

Sus opciones más utilizada son las siguientes:

- ✓ i: solicita confirmación antes de realizar la acción.
- ✓ r o R: eliminación recursiva sobre directorios.
- ✓ f: fuerza la eliminación aun estando protegido el archivo contra escritura.

Ejemplo: *rm notas.txt notas.txt.bck*

Además, también se pueden emplear caracteres comodín, como "*" o "?". Estos caracteres permiten generar patrones de identificación de ficheros. "*" hace referencia a cualquier cadena de caracteres y "?" solo referencia un único carácter cualquiera. El intérprete de comandos, antes de ejecutar la orden donde se incluyan los caracteres comodines, realiza una acción llamada "expansión de comodines", donde traduce estos comodines a todas las posibles combinaciones de caracteres, según el patrón aportado. De esta manera, se pueden referir varios archivos a la vez.

Actividades propuestas

3.6. Empleando caracteres comodín, realiza las siguientes acciones por línea de comandos:

- Eliminar todos los archivos del directorio actual.
- Elimina, solicitando confirmación, todos los ficheros que comiencen por "doc".
- Elimina todos los archivos que comiencen por "script" y tengan un carácter más como nombre de fichero.

3.7. En un mismo directorio, crea un archivo llamado *origen1*. Crear un enlace duro sobre *origen1* llamado *origen_d*. Crear un enlace simbólico sobre *origen1* llamado *origen_s*. Lista el contenido del directorio en formato largo mostrando los números de i-nodo. Elimina el archivo *origen1*. Vuelve a listar el contenido del directorio en formato largo, mostrando los números de i-nodo. ¿Qué ha ocurrido? Eliminar el archivo *origen_d*. Volver a listar el contenido del directorio en formato largo, mostrando los números de i-nodo. ¿Qué ha ocurrido?

3.4.3. Creación y eliminación de directorios

Se pueden crear directorios mediante la orden *mkdir*. Su sintaxis es la siguiente:

```
mkdir lista_de_directorios
```

La eliminación de directorios se puede efectuar mediante la orden *rm -r*, si dispone de contenido, o empleando el comando *rmdir*, si el directorio se encuentra vacío.

```
rm lista_de_directorios
```

Ejemplos:

- *mkdir scripts:* crea el directorio.
- *rmdir scripts*: elimina el directorio vacío *scripts*.
- *rm -ri ./trabajo*: elimina, pidiendo confirmación, todos los archivos del directorio *trabajo* y el propio directorio.

3.4.4. Copia de archivos

El comando utilizado para copiar la información de un archivo es *cp*. Su sintaxis es:

```
cp [-irR] lista_archivos_origen destino
```

Por defecto se sobrescribirá en el destino la lista de archivos de origen, en caso de equivalencia en los nombres. Sus opciones más utilizadas son:

- ✓ i: solicita confirmación antes de sobrescribir un archivo con el mismo nombre en el destino.
- ✓ r o R: copia de forma recursiva en directorios.

Ejemplos:

- *cp -i /etc/passwd ./passwd.bck*: copia el archivo *passwd*, solicitando confirmación, a otra ubicación (desde /etc. al directorio actual) cambiando además de nombre.
- *cp -r ../backup/*: copia el directorio actual a *./backup/* directorios completos.

3.4.5. Renombrado o movimiento de archivos

Se emplea la orden *mv* para mover o renombrar archivos. Su funcionamiento es similar a la orden *cp*, salvo que los archivos de origen desaparecen:

```
mv [-iu] lista_archivos_origen destino
```

Sus opciones más utilizadas son:

- ✓ i: se emplea para asegurarse que en el destino no exista un archivo con el mismo nombre, ya que se sobrescribiría.
- ✓ u: solo mueve archivos o directorios con el mismo nombre entre origen y destino si estos son más actuales en el origen.

Ejemplo: *mv passwd.bck ./backup/*

3.4.6. Impresión de archivos

La interfaz de comandos dispone de multitud de comandos para poder inspeccionar el contenido de los ficheros. Para que este pueda ser correctamente interpretado, ha de encontrarse en formato ASCII. En caso contrario, aparecerá un conjunto de caracteres no legibles.

Los principales comandos empleados para mostrar el contenido de ficheros en la interfaz textual son:

```
cat [lista_archivos] [ | >|>>|<|<<) archivo]
```

Concatena la lista de archivos, mostrándolos por pantalla. Su salida se puede redireccionar a un archivo mediante:

- > sobrescribe si existe el archivo o, en caso contrario, lo crea.
- >> añade al contenido de un fichero existente o, en caso contrario, lo crea.

Por tanto, se emplea la orden *cat* para crear archivos de escaso contenido de forma rápida. Para ello, se emplea el comando *cat*, seguido de ">" o ">>" y de un archivo. De esta manera, el intérprete de comandos espera que introduzcamos texto hasta que pulsemos la combinación de teclas CTRL+D. En la siguiente imagen, podemos ver cómo hemos creado dos archivos de texto para, más tarde, fusionarlos en uno solo.

Figura 3.8
Usos de cat.

Otros comandos muy utilizados son:

```
more [lista_archivos]
less [lista_archivos]
```

Permiten visualizar el contenido de ficheros por páginas. A diferencia de *cat*, se puede navegar entre páginas. El comando *less* es más versátil y potente que *more*, resultando ideal para ficheros de gran tamaño.

Existen otras órdenes que ayudan a obtener el principio o el final de archivos:

```
head [-n] lista_ficheros
tail [-n] lista_ficheros
```

Se emplean para visualizar las *n* primeras líneas *(head)* o últimas *(tail)* de uno o varios ficheros de texto. Por defecto, mostrará diez líneas.

Ejemplos:

- ✓ *head -5 /etc/passwd* : muestra las cinco primeras líneas del fichero */etc/passwd*.
- ✓ *tail -3 /etc/passwd* : muestra las tres últimas líneas del fichero */etc/passwd*.

TOMA NOTA

Navegación entre páginas para comandos *more* y *less*:

- Tecla barra espaciadora: pasa a la siguiente página.
- Tecla enter: pasa a la siguiente línea.
- Tecla q: sale de la visualización.
- /texto: busca el texto o expresión regular.
- n: busca la siguiente coincidencia del texto o expresión regular.
- :n: pasa al siguiente fichero.
- :p: retrocede al fichero anterior.

3.4.7. Cuenteo de un fichero

La orden *wc* permite mostrar, por defecto, el número de líneas, palabras y número de bytes de un fichero. Su sintaxis es la siguiente:

```
wc [-lwcL] lista_ficheros
```

Podemos seleccionar algunos contadores de manera aislada o conjunta:

- l: número de líneas.
- w: número de palabras.
- c: número de bytes.
- L: longitud de la línea más larga.

Ejemplos:

✓ *wc texto1.txt texto2.txt fusion_textos.txt* : muestra las líneas, palabras y número de bytes de cada uno de los ficheros indicados.
✓ *wc –l texto1.txt* : solo imprime por pantalla el número de líneas de *texto1.txt*.

SABÍAS QUE...

El fichero */etc/passwd* almacena las cuentas de los usuarios del sistema. Cada fila correspondiente con un usuario, consta de siete campos delimitados por ":". Cada columna hace referencia, por este orden, a: usuario, password, UID, GID, descripción, home y shell.

3.4.8. Ordenación de un fichero

La orden *sort* permite mostrar, de forma ordenada, las líneas de un fichero. Por defecto, ordena siguiendo el orden establecido por los caracteres en ASCII, donde las letras minúsculas tienen un orden mayor que las mayúsculas.

```
sort [-fnru][-t <delimitador>][-k <num_campo>] lista_archivos
```

Las opciones más importantes son:

- f: ignora las mayúsculas y las minúsculas.
- r: invierte el orden.
- n: ordena numéricamente en lugar de alfabéticamente.
- u: elimina entradas repetidas.
- t *delimitador*: indica un delimitador o separador de campos.
- k: indica el número de campo por el que se va a realizar la ordenación. Por defecto, toma los espacios, tabuladores y carácter de fin de línea como delimitadores de campo.

Actividad resuelta 3.3

Genera un archivo de texto llamado ordenacion.txt *con las siguientes palabras (una por línea) y en el siguiente orden: coche, moto, Coche, Moto, 1, 100, 2, 200, Autobus, autobus.*

SOLUCIÓN

- Muestra por pantalla de forma ordenada alfabéticamente el fichero, sin distinguir mayúsculas y minúsculas, y eliminando líneas repetidas: *sort –fu ordenacion.txt*
- Ordena numéricamente el fichero *ordenacion.txt*: *sort –n ordenacion.txt*. Puedes apreciar la diferencia con la ordenación del ejemplo anterior.
- Ordena inversamente el fichero */etc/passwd* por nombre de usuario (primer campo), estableciendo como delimitador ":": *sort –t: -k1 –r /etc/passwd*
- Ordena por el noveno campo correspondiente a los nombres de ficheros la salida listada en formato largo del directorio actual: *ls –l | sort –k9*
- Ordena numéricamente por el tamaño de los ficheros la salida listada del directorio actual en formato largo: *ls –l | sort –k5n*

3.4.9. Entrada y salidas estándar. Redirecciones

En general, todos los comandos y, en consecuencia, los procesos que se generan a partir de ellos reciben la información mediante un flujo de entrada de información y los resultados son enviados mediante un flujo de salida. El sistema operativo asigna automáticamente a cada comando entradas y salidas estándar asociadas a los flujos de entrada y salida, respectivamente.

Por defecto, se asignan tres ficheros: *entrada estándar (stdin)*, *salida estándar (stdout)* y *salida de errores estándar (stderr)*. Además, estos flujos de entrada o salida se identifican por un número o descriptor de fichero: *0*, *1* y *2* para *stdin*, *stdout* y *stderr*, respectivamente.

La entrada estándar nutre al comando de información para su ejecución, la salida estándar transmite el resultado y, si durante la ejecución de un comando sobreviene un error o un aviso, este se enviará a la salida de errores estándar.

Normalmente, la entrada estándar está asociada con el teclado y la salida estándar y la salida de errores estándar con la pantalla. No obstante, no siempre es así, ya que muchos comandos pueden tomar la entrada o salida estándar de otros ficheros. Además, el comportamiento normal del flujo de un comando se puede "alterar" mediante operadores que redireccionan su flujo a otros comandos, dispositivos o ficheros.

A) Redirecciones de la salida estándar

Podemos emplear el operador ">" para volcar la salida de una orden sobre un fichero en lugar de a la salida estándar:

```
orden > fichero
```

Si el fichero no existe, se crea un nuevo y, si existe, sobrescribe su contenido. También se puede utilizar "1>", puesto que "1" hace referencia al descriptor de fichero.

También se puede emplear el operador ">>" para realizar la misma acción, pero, a diferencia del operador ">", añade su contenido al fichero sin sobrescribirlo.

```
orden >> fichero
```

Ejemplos:

- ✓ Ejecutar *cat /etc/passwd* . La entrada estándar es el fichero /etc/passwd y la salida estándar y de error es la pantalla.
- ✓ Ejecutar *cat > notas.txt* . La entrada estándar es el teclado y la salida estándar se redirecciona al fichero *notas.txt*.
- ✓ Ejecutar *ls /usr >> notas.txt* . La entrada estándar es el resultado de la ejecución del comando *ls /usr*, el cual no se muestra por pantalla, ya que se redirecciona al fichero *notas.txt*, donde se añade al contenido existente en este.

B) Redirecciones de la entrada estándar

También podemos redireccionar un fichero como entrada de una orden, en lugar de la entrada estándar. Para ello, se emplea el operador "<":

```
orden < fichero
```

Existe una redirección particular que permite introducir texto hasta que se encuentre una línea únicamente con el delimitador establecido.

```
<<delimitador
```

```
luis@luis-VirtualBox:~$ cat <<END
> capitulo1
> capitulo2
> END
capitulo1
capitulo2
```

Figura 3.9
Uso de cat con <<.

Ejemplos:

- Ejecutar *cat < notas.txt* . La orden *cat* no recibe ningún fichero como argumento, por tanto, toma el fichero *notas.txt* como entrada al ser redireccionado a la entrada de la orden.
- Ejecutar *cat <<END* . La orden *cat* toma el teclado como entrada estándar hasta que se encuentre la cadena "END" en una línea.

C) Redirección de la salida de error estándar

Durante la ejecución de un comando, tanto la salida estándar como la salida de error estándar pueden tener o no actividad. Al ser flujos diferentes, la salida de error estándar también se puede redireccionar empleando el operador *2>*:

```
orden 2> fichero
```

Del mismo modo, se puede emplear el operador "*2>>*" para añadir contenido al fichero. Ejemplos:

- ✓ Ejecutar *ls s* (dado un archivo s inexistente). El archivo "s" no existe en el directorio actual. Por tanto, al listarlo, el terminal ofrece un mensaje de error por la salida estándar. Si ejecutamos la misma orden redireccionando la salida de error estándar al fichero error.txt, se almacenará en este el mensaje de error y no será mostrado por pantalla: *ls s 2> error.txt*.
- ✓ Ejecutar *ls –R /usr > salida.txt 2> salida_err.txt*. El comando "*ls –R /usr*" almacena su salida en el fichero salida.txt y los errores o avisos se almacenarán en *salida_err.txt*.
- ✓ Ejecutar *ls –R / 1>listado.txt 2>errores.txt*. El listado del directorio raíz de manera recursiva se redirige al archivo *listado.txt*, mientras que todos los errores generados (principalmente, porque no se poseen permisos de acceso a determinados directorios) se envían al fichero *errores.txt*.

D) Redirección de la salida estándar y la salida de error estándar al mismo destino

Ambas salidas se pueden redireccionar al mismo destino mediante el comando &> o &>>. El primero sustituye y el segundo añade contenido al fichero.

```
orden &> fichero
orden &>> fichero
```

Ejemplo: *ls –R / &> listado_y_errores.txt*

E) Redirección de la salida estándar y la salida de error estándar de una orden con la entrada estándar de otra orden

Es muy común emplear el operador "*tubería*" o "*pipe*", "|" para concatenar salidas estándar con entradas estándar entre órdenes.

```
ordenA | ordenB
```

Al igual que antes, con el operador |& podemos redireccionar la salida estándar y la salida de error estándar de una orden a la siguiente.

```
ordenA |& ordenB
```

Además, podemos emplear el operador "tubería" a la vez que enviamos la información de la salida estándar a la salida estándar y a un archivo mediante la orden *tee*:

```
ordenA | tee fichero | ordenB
```

Ejemplos:

- Ejecutar *ls -1 /usr | wc –l* . La orden *ls -1 /usr* lista en una columna una línea por cada fichero o directorio. Su salida estándar se redirecciona a la entrada estándar de *wc –l* que a su vez envía a su salida estándar (pantalla) el resultado: 9. Hemos obtenido el número de ficheros o directorios de */usr*.
- Ejecutar *ls -1 /usr | tee listado | wc –l*. La salida estándar de *ls -1 /usr* se envía a la orden *tee* que almacena en el fichero *listado* el resultado de la orden anterior, a la vez que envía esta a la siguiente orden.

F) Redirección de la salida de error a la salida estándar

Se suele emplear el operador *2>&1* para que la salida de error se dirija al mismo lugar que la salida estándar.

Ejemplo: *ls –R / > listado_y_errores.txt 2>&1*

3.4.10. Procesamiento de textos

Existen dos comandos ampliamente utilizados para extraer información sobre textos generados por otros comandos, ficheros o cadenas de caracteres en general. Estos son *cut* y *grep*.

El comando *cut* se emplea para obtener información a partir de la división de un fichero o cadena de caracteres en columnas. Estas columnas se pueden establecer por caracteres o por campos delimitados por un delimitador de campo.

Su sintaxis es la siguiente:

```
cut-c<lista_caracteres>|-f<lista_columnas>[<-ddelimitador>]fichero_texto
```

Donde:

- ✓ –c <lista_caracteres>: corta por caracteres especificados por *lista_caracteres*.
- ✓ –f <lista_columnas> [<-d delimitador>]: corta por campos establecidos por *lista_columnas*. Por defecto, los delimitadores son: espacio, tabuladores o espacio fin de línea, a menos que se especifique otro mediante la opción *–d*. No se pueden ejecutar conjuntamente las opciones *–c* y *–f*.

Ejemplos:

- Obtener el primer campo del fichero /etc/passwd: *cut –f1 –d: /etc/passwd*. El comando *cut* toma la entrada estándar del fichero */etc/passwd*. Los campos se establecen por el delimitador ":" gracias a la opción "*–d*". La opción *–f1* hace referencia al primer campo. Si quisiéramos añadir más campos, bastaría con emplear comas para campos independientes o guiones para intervalos de campos, como: *cut –f1,4-7 –d: /etc/passwd*.
- Obtener los caracteres 1°, 2°, 3°, 4° y 20° del fichero */etc/passwd*:

```
cut -c1-4,20 /etc/passwd
```

Por otro lado, el comando *grep* localiza un patrón en uno o varios ficheros, mostrando las líneas donde se encuentra. Su sintaxis es la siguiente:

```
grep [-nvlicw] patrón fichero_texto [fichero_texto …]
```

Las opciones más empleadas son:

- ✓ l: solo muestra los ficheros que contienen el patrón especificado.
- ✓ i: elimina la distinción entre mayúsculas y minúsculas.
- ✓ c: muestra el número de líneas totales que cumplen con el patrón para cada fichero.
- ✓ w: localiza el patrón como palabra y no como parte de una cadena de texto.
- ✓ n: imprime el número de línea del patrón localizado.
- ✓ v: busca líneas que no contengan el patrón especificado.

Además, grep permite emplear expresiones regulares básicas (patrones que definen un conjunto de cadenas de texto). Las más utilizadas son las que se muestran en el cuadro 3.1.

CUADRO 3.1
Expresiones regulares

Símbolo	Significado	Ejemplos
.	Cualquier carácter, excepto el carácter fin de línea	Cas.
*	Cero o más repeticiones del carácter que le precede	C*
[lista]	Coincide con uno de los caracteres presentes en la lista	[aCgh]
	Se puede indicar la negación de la coincidencia de un patrón	[^aCgh]
	Se pueden indicar rangos de caracteres, si se incluyen guiones y estos caracteres se especifican de mayor a menor	[0-9] [^0-9]
^	Comienzo de línea	^C
$	Fin de línea	a$

Los símbolos que tienen un significado especial se pueden emplear si se anteceden con el carácter "\" en los patrones. Como, por ejemplo "*" o "\^".

El comando *grep* puede utilizarse con la opción "*–E*". De esta forma, su uso sería equivalente al comando *egrep*. Este último permite usar expresiones regulares extendidas, pudiendo formar patrones más complejos y potentes. Es recomendable la lectura del manual de ayuda de *grep* o *egrep*: *man grep*.

Actividad resuelta 3.4

- Localizar aquellas líneas que contengan el patrón "root" en el fichero */etc/passwd*:

```
grep root /etc/passwd
```

- Crear un fichero de texto para trabajar con él. Este fichero se va a titular *Pirata.txt* y contendrá el siguiente texto:

```
Con diez cañones por banda,
viento en popa a toda vela,
no corta el mar, sino vuela,
un velero bergantín:
bajel pirata que llaman
por su bravura el Temido,
en todo mar conocido
del uno al otro confín.
```

Figura 3.10
Contenido de un fichero de texto.

- Listar aquellas líneas que tengan el patrón "el mar", mostrando además el número de línea donde se encuentren dentro del fichero:

```
grep -n "el mar" Pirata.txt
```

En este caso, se ha de entrecomillar el patrón, ya que contiene un espacio. Si no se entrecomillara, resultaría un error al tomar dos argumentos en lugar de uno, siendo interpretado "mar" como un archivo.

- Mostrar aquellas líneas que tienen una "u" seguida de cualquier otro carácter:

```
grep u. Pirata.txt
```

- Mostrar aquellas líneas con palabras con "u" seguido de cualquier otro carácter:

```
grep -w u. Pirata.txt
```

- Mostrar aquellas líneas que comienzan por "en":

```
grep ^en Pirata.txt
```

- Mostrar aquellas líneas que terminan en ",":

```
grep ,$ Pirata.txt
```

- Mostrar aquellas líneas que comiencen por "C", seguidas de cero o más repeticiones de cualquier carácter y terminen en ",":

```
grep ^C.*,$ Pirata.txt
```

- Mostrar aquellas líneas que contengan cadenas de caracteres que no contienen una letra mayúscula, seguida del carácter "a" y del carácter ",":

```
grep [^A-Z][a], Pirata.txt
```

- Mostrar aquellas líneas que no contengan la palabra "en":

```
grep -v -w en Pirata.txt
```

3.5. Gestión de archivos por interfaz gráfica en Microsoft Windows

El 'Explorador de archivos' de Microsoft Windows facilita la gestión de los archivos sin necesidad de conocer la administración interna del sistema. Este dispone de accesos rápidos a carpetas muy utilizadas: 'Escritorio', 'Imágenes' o 'Descargas', así como a las unidades conectadas. En la figura 3.11 se detallan sus partes:

1. Barra de herramientas de acceso rápido.
2. Cinta de opciones.
3. Barra de direcciones.
4. Cuadro de búsqueda.
5. Lista de archivos: carpetas de usuario.
6. Lista de archivos: dispositivos y unidades.
7. Iconos de vista de los archivos.
8. Panel de navegación.

Su estructura es bastante intuitiva, lo que permite realizar cualquier operación sobre los elementos de diversas maneras.

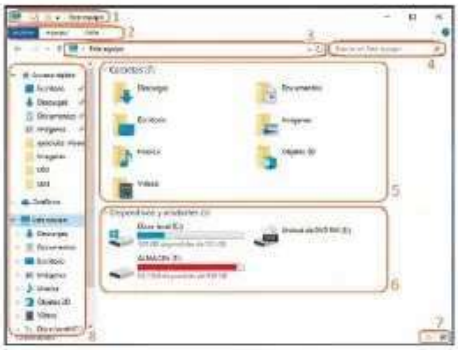


Figura 3.11
Explorador de archivos.

El propio 'Explorador de archivos' se puede configurar mediante la 'Barra de herramientas' de acceso rápido (pulsando en el botón con forma de triángulo), habilitando o deshabilitando algunas opciones.

El 'Explorador de archivos' se adapta a la navegación de la estructura de carpetas. De esa manera, dependiendo si nos encontramos en una unidad, carpeta o archivo, el entorno se va adaptando, así como las operaciones que podemos hacer con cada uno.

La cinta de opciones nos resulta de mucha utilidad, puesto que contiene en modo icono los comandos que podemos realizar sobre cada elemento sobre el que nos encontremos: copiar, pegar, mover a otro lugar, copiar a otro lugar, eliminar, cambiar el nombre, crear una nueva carpeta o un acceso directo, ver sus propiedades, etc. Todo ello en la pestaña 'Inicio'.

Además, permite de forma muy sencilla compartir de diversas maneras un elemento en la pestaña 'Compartir'.

TOMA NOTA

Todas estas opciones se pueden realizar a través del menú contextual si nos situamos sobre un elemento y pulsamos el botón secundario del ratón.

También podemos modificar la manera de organizar los archivos y directorios en la pestaña 'Vista'. Por ejemplo, con la vista 'Detalles' de archivos, disponemos de más información propia de cada elemento. Incluso podemos añadir o quitar propiedades si nos situamos en la barra de detalles con el botón secundario del ratón.

En la propia pestaña 'Vista', el botón 'Opciones' permite configurar multitud de características. Entre ellas, destacan las opciones de configuración avanzada de la pestaña 'Ver', donde podemos mostrar archivos, carpetas o unidades ocultas, ocultar archivos protegidos del sistema operativo u ocultar extensiones de los archivos. Algunas de estas opciones se encuentran también en la cinta de opciones.

Además, en la pestaña 'Herramientas de unidad' (accesible cuando se selecciona una unidad) se gestionan las unidades de almacenamiento, pudiendo cifrarlas, desfragmentarlas, liberar espacio o formatearlas.

Para una mayor agilidad en la gestión de archivos se suelen hacer uso de teclas combinadas junto con el ratón o combinaciones de teclas rápidas.

CUADRO 3.2
Combinaciones en Windows

Combinaciones	Descripción
Ctrl+x	Cortar el elemento seleccionado
Ctrl+c	Copiar el elemento seleccionado
Ctrl+v	Pegar el elemento seleccionado
Ctrl+z	Deshacer una acción
F2	Modificar el nombre del elemento
Ctrl+e	Seleccionar todos los elementos
Ctrl+d	Eliminar el elemento seleccionado y enviarlo a la papelera de reciclaje
Ctrl+click ratón sobre elementos	Seleccionar distintos elementos

3.6. Gestión de almacenamiento por línea de comandos en Linux

La estructura de directorios de Linux incluye los diferentes discos y particiones que el sistema operativo es capaz de gestionar. Linux puede tratar con una partición de disco cuando esta contiene un sistema de archivos y se anexa a su árbol de directorios mediante un directorio común, al que se denomina *punto de montaje*. Este directorio es uno más entre el conjunto de directorios, al que se le otorga la capacidad de acceder a través de él a un *subsistema de archivos*.

RECUERDA

✓ Ya sabemos que la partición es una división física del disco duro, que permite organizar la información, incrementar la eficiencia de acceso a los datos en el disco, aumentar la seguridad e instalar diferentes sistemas de archivos, así como sistemas operativos.

Los dispositivos de almacenamiento se administran a través del directorio del sistema */dev* (al igual que el resto de dispositivos físicos del equipo), el cual contiene archivos que se agrupan por tipos, atendiendo al comienzo del nombre. Los más importantes en cuanto a la gestión de almacenamiento son los siguientes:

- */dev/hd**: interfaz para unidades de disco duro IDE.
- */dev/sd**: interfaz para discos SCSI, SATA y unidades con conexión USB (unidades FLASH o discos duros externos).
- */dev/tty**: consolas o terminales físicos. Para cambiar entre consolas, se ha de utilizar la combinación de teclas CTRL+ALT+ F1..F6. Si queremos volver a la consola gráfica o entorno gráfico se emplea la combinación de teclas CTRL+ALT+ F7.
- */dev/ttyS**: puertos serie.
- */dev/sr** y */dev/scd**: interfaz para unidades CD o DVD.

De tal manera, que la sintaxis para determinar la identificación de cada dispositivo o partición es la siguiente:

```
/dev/<id_dispositivo><letra_orden><numero_partición>.
```

Ejemplos:

- ✓ */dev/hda:* primer disco duro IDE.
- ✓ */dev/sdb:* segundo disco duro SCSI, SATA o con conexión USB.
- ✓ */dev/sdc2:* segunda partición del tercer disco duro SCSI, SATA o con conexión USB.
- ✓ */dev/ttyS0:* primer puerto serie.

En discos con sistema de particiones MBR, a las particiones primarias se le asignan los números del 1 al 4 y, las lógicas, desde el 5 en adelante.

3.6.1. Montaje y desmontaje

La orden que realiza el montaje de un sistema de archivos es *mount*. Gracias a ella, se monta cualquier sistema de archivos reconocible por el núcleo de Linux en un punto de montaje del sistema. Todos los sistemas de archivos se montan directamente por nosotros o indirectamente durante el arranque del sistema, a excepción del sistema de archivos raíz "/", que se asocia a un punto de montaje compilado en el propio kernel y que monta la partición especificada durante la instalación. La sintaxis de la orden *mount* es la siguiente:

```
mount [-avwrt] [tipo] [dispositivo] [punto_de_montaje]
```

Las opciones más empleadas son:

- -a: monta los sistemas de archivos presentes en */etc/fstab*, salvo que se indique el parámetro *noauto*, que impediría el montaje por esta opción.
- -v: muestra información del proceso de montaje.
- -w: monta el sistema de archivos con permisos de lectura y escritura.
- -r: monta el sistema de archivos con permisos de solo lectura.
- -t <tipo>: indica el tipo de sistema de archivos para montar. Este puede ser, entre otros: *ext, ext2, ext3, ext4, ntfs, nfs, iso9660, msdos, vfat, hfs, hfsplus* o *smbfs*.

El sistema mantiene actualizada una lista de sistemas de archivos montados a través del archivo */proc/self/mounts* (en versiones anteriores se empleaba /etc/mtab, que actualmente es un enlace simbólico al anterior). Este se interpreta cuando ejecutamos *mount* sin argumentos y se actualiza al montar nuevos sistemas de archivos o al desmontar sistemas de archivos existentes. Por tanto, la orden *mount* sin modificadores y argumentos lista los sistemas de archivos montados actualmente en el sistema.

```
luis@luis-VirtualBox:~$ mount
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime)
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
udev on /dev type devtmpfs (rw,nosuid,relatime,size=998080k,nr_inodes=249520,mode=755)
devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,gid=5,mode=620,ptmxmode=000)
tmpfs on /run type tmpfs (rw,nosuid,noexec,relatime,size=204116k,mode=755)
/dev/sda1 on / type ext4 (rw,relatime,errors=remount-ro)
```

Figura 3.12
Ejemplo de ejecución de mount.

Por defecto, Ubuntu monta automáticamente un sistema de archivos. Esto se da, por ejemplo, cuando Linux detecta que un pendrive ha sido conectado en un puerto USB o cuando se inserta un nuevo disco DVD en la unidad lectora. Esta opción se puede deshabilitar, aunque realmente resulta cómoda.

Las unidades Flash USB u otros dispositivos portables son gestionados por el gestor de archivos de Linux que, en el caso de *Ubuntu 18.10*, es *Nautilus*, y se encuentra en el entorno de escritorio de *GNOME*. Este gestor automatiza su conexión.

Si realizamos un montaje, hemos de indicar el tipo de sistema de archivos, la partición y el punto de montaje donde se situará el nuevo sistema de archivos (este último ha de existir). No obstante, se han de seguir una serie de pasos cuando deseamos montar un sistema de archivos:

1. Detectar el nombre asignado por el sistema a la partición objeto de montaje. Una manera de conocerlo es realizar un listado de las unidades o particiones recientemente creadas. Si sabemos que es un disco SCSI, SATA o con conexión USB, podemos ejecutar *ls -ltr /dev/sd**. Otras formas de averiguarlo, son mediante los comandos:

   - *lsblk*: lista la información de los dispositivos por bloques: nombres, tipos, puntos de montaje, tamaños, etc. Si ejecutamos *lsblk - -fs* , mostrará el sistema de archivos.
   - *lshw –C disk*: muestra información detallada de los discos conectados al sistema.
   - *fdisk –l*: muestra información de los dispositivos o particiones que figuran en */proc/partitions*, el cual registra los dispositivos conectados.
   - *lsusb*: muestra información de los buses USB y los dispositivos conectados a ellos. Esto permite recabar información sobre los tipos de dispositivos USB conectados: *lsusb –tv*.
   - *dmesg*: lista el contenido del buffer de mensajes del núcleo de Linux. Este comando muestra gran cantidad de información. En nuestro caso, podemos averiguar el

último dispositivo conectado ejecutando este comando justo después de conectar un dispositivo, mostrando una serie de mensajes relativos a la conexión de un nuevo dispositivo USB y su posterior configuración. En caso de que el sistema tenga mucha actividad, podríamos filtrar, por ejemplo, con *dmesg | grep sd*.

2. Crear el punto de montaje, si este no lo está.
3. Montar el sistema de archivos en el punto de montaje. Por defecto, solo el usuario *root* tiene permisos para ejecutar este comando.

```
luis@luis-VirtualBox:~$ sudo mount -t vfat /dev/sdb1 /home/luis/pendrive
luis@luis-VirtualBox:~$ cd /home/luis/pendrive
luis@luis-VirtualBox:~/pendrive$
```

Figura 3.13
Ejemplo de montaje de sistema de archivos.

Dado el ejemplo de la imagen anterior y a partir de ese momento, el sistema de archivos del pendrive cuelga del directorio */home/luis/pendrive* y toda acción sobre él afectará al propio sistema de archivos del dispositivo. La opción *–t* resulta optativa, puesto que *mount* intenta averiguar el tipo de sistema de archivos.

Actividades propuestas

3.8. Investigar otros dos gestores de archivos empleados en GNU/Linux que estén asociados a otros entornos de escritorio.

3.9. mount emplea el comando blkid y el archivo /proc/filesystems para intentar montar una partición con un sistema de archivos no indicado. Utilizando el comando blkid -k y accediendo al contenido del archivo /proc/filesystems, realiza un listado de todos los sistemas de archivos que Ubuntu puede manejar.

Para dejar de usar completamente o extraer un dispositivo con sistema de archivos, este se debe desmontar. Para el desmontaje se emplea el comando *umount*, el cual puede completarse si no existen directorios en uso del sistema de archivos objeto a desmontar o procesos lanzados desde él. Para ejecutar el desmontaje se puede indicar la partición o el punto de montaje:

```
umount <dispositivo> o umount <punto_de_montaje>
```

Para el ejemplo anterior son igualmente válidos:

```
umount /dev/sdb1
umount /home/Luis/pendrive
```

Los sistemas operativos GNU/Linux automatizan el proceso de montaje de particiones gracias al archivo editable */etc/fstab*. Generalmente, los discos duros internos y unidades de CD/DVD son los que se especifican en este archivo. De manera que las particiones que se detallen en él se montarán durante el arranque del sistema operativo.

Una partición que no figure en este archivo solo podrá ser montada y desmontada porun usuario administrador o root.

El fichero /etc/fstab se estructura por columnas separadas por espacios de la siguiente manera.

- ✓ File system: partición.
- ✓ Mount point: punto de montaje.
- ✓ Type: tipo de sistema de archivos que contiene la partición. En caso de indicar "auto", detecta el sistema de archivos automáticamente.
- ✓ Options: opciones de montaje. Son semejantes a las del comando mount. Las opciones más comunes son:
  - auto: monta el sistema de archivos durante el arranque. Este es el valor por defecto.
  - noauto: el sistema de archivos se montará solo manualmente.
  - ro: monta el sistema de archivos en modo solo lectura.
  - rw: monta el sistema de archivos en modo lectura-escritura.
  - user: permite a cualquier usuario montar el sistema de archivos.
  - users: cualquier usuario del grupo *users* puede montar el sistema de archivos.
  - nouser: solo el usuario *root* puede montar el sistema de archivos.
  - defaults: establece una serie de opciones de montaje predeterminadas.
  - errors=VALOR, establece una acción en caso de que en el sistema de archivos se produzca un error. Si VALOR es:
    - continue: el sistema sigue funcionando.
    - remount-ro: el sistema se reinicia en modo de solo lectura.
    - panic: se apaga el sistema.
- ✓ Dump: habilita o deshabilita la copia de seguridad mediante el comando *dump*. Normalmente no se encuentra instalado este programa, por lo que la opción más común es 0 (deshabilitado). Con valor 1, *dump* hace una copia de seguridad del sistema de archivos.
- ✓ Pass: establece el orden en el que se comprueban los sistemas de archivos. Con un valor 0 no se comprueba el sistema de archivos, 1 se comprueba en primer lugar y 2 se comprueba en segundo lugar. Normalmente, el sistema de archivos raíz ha de comprobarse en primer lugar y el resto en segundo lugar.

PARA SABER MÁS

Desmontaje de sistemas de archivos

Los dispositivos de almacenamiento disponen de una memoria caché de tamaño variable que actúa como unidad temporal volátil o buffer cuya principal ventaja es la velocidad de escritura o lectura y el sincronismo con el procesador, chipset y memoria RAM. En el proceso de escritura, la caché almacena aquellas modificaciones que aún no han sido realmente actualizadas en la propia memoria permanente del dispositivo. Cuando la memoria caché se llena, se vuelca su contenido al medio de almacenamiento permanente. Este proceso se efectúa en intervalos de tiempo altamente frecuentes.
Una vez terminadas las acciones sobre el sistema de archivos, este se ha de desmontar para separar el sistema de archivos del punto de montaje y evitar pérdidas de información de aquellos datos que se encuentren en caché y que aún no hayan sido volcados al almacenamiento permanente del medio, indicándole que realicen así este trasvase de información.

Un ejemplo de ello se representa en la figura 3.14, donde las líneas que comienzan por el carácter "#" son comentarios y no se interpretan. Nos encontramos con una partición donde se encuentra el sistema raíz en *ext4* y un archivo de intercambio para swap.

Figura 3.14 Contenido de /etc/fstab.

Como ya sabemos, los dispositivos y particiones se identifican con un nombre descriptivo por el tipo de dispositivo, como, por ejemplo, */dev/sda1* o */dev/sr0*. Esta nomenclatura puede dar lugar a equívocos, ya que se asocia a la conexión de estos con los puertos físicos y cableado en la placa base o al orden de almacenamiento en la BIOS. Como podemos extraer un dispositivo interno físicamente y conectarlo en otro puerto, se podría alterar la asociación de */etc/fstab* con un dispositivo. Por tanto, resulta conveniente asignar a los dispositivos por bloques, un nombre que lo identifique inequívocamente (pero evitando asignar dos etiquetas iguales a dos particiones diferentes). Esto se puede realizar:

1. Mediante una etiqueta. Se puede asignar una etiqueta a una partición mediante el programa *GParted* (cuando la partición se encuentre desmontada) o mediante otros comandos, dependiendo del sistema de archivos.

   - *NTFS: ntfslabel <partición> <etiqueta>*
   - *FAT: fatlabel <partición> <etiqueta>*
   - *ext2/3/4: e2label <partición> <etiqueta>* y *tune2fs –L <etiqueta> <partición>*
   - *swap: swaplabel –L <etiqueta> <partición>*

   Ejemplo: sudo e2label /dev/sda1 sistemaraiz

2. Mediante un UUID. El UUID o identificador único universal se asigna a la partición cuando esta es formateada. Por tanto, no tenemos que asignarla nosotros. Sin embargo, si se vuelve a formatear una partición o se modifican sus características (como su tamaño), cambiará su UUID.

   Podemos ver las etiquetas y los UUID de las particiones del sistema mediante *lsblk --fs*.

   Otras formas de mostrar las etiquetas y los UUID son listando los directorios */dev/disk/by-label/* y */dev/disk/by-uuid*, respectivamente, en sistemas MBR o */dev/disk/by-partlabel/* y */dev/disk/by-partuid/* para sistemas UEFI con GPT.

   Ejemplos:

   ls –l /dev/disk/by-label
   ls –l /dev/disk/by-uuid

   La identificación de etiquetas o UUID en el archivo */etc/fstab* es sencilla, basta con indicar *LABEL=<nombre-etiqueta>* o *UUID=<código_UUID>* en la columna *file system*. Para ello, editamos el archivo */etc/fstab* con privilegios de *root* y añadimos las modificaciones necesarias.

Actividad propuesta 3.10

Empleando la ayuda de *mount (man mount)*, averigua el significado de cada una de las opciones preestablecidas mediante la opción defaults.

3.6.2. Particionar

Existen varias herramientas para crear y manipular particiones en Linux, teniendo en cuenta que la mayoría de tareas de administración con particiones necesitan privilegios de *root*.

Las más conocidas son *fdisk*, *gdisk*, *parted* y *GParted*. Todas trabajan con esquemas de particionamiento MBR y GPT, a excepción de *gdisk*, que lo hace solo con GPT. Las tres primeras se manejan a través de la interfaz de texto mediante un menú interactivo. Tanto *fdisk* como *gdisk* (*GPT fdisk*) presentan un conjunto de opciones muy similares, sin embargo, *parted* dispone de otras opciones y permite, además, formatear las particiones.

Ubuntu con escritorio GNOME, incorpora la herramienta de partición de discos GNOME-disks (Discos de GNOME) a la que podemos acceder como "Discos". Dentro de ella, se pueden seleccionar cualesquiera de las particiones en el listado de la izquierda; y a la derecha aparecerá información al respecto. Además, permite desmontar, eliminar y establecer opciones adicionales de la partición de forma sencilla:

- Formatear.
- Editar la partición.
- Redimensionarla.
- Comprobar y reparar el sistema de archivos.
- Crear y restaurar una imagen.
- Probar el rendimiento de la partición.

Figura 3.15
Discos de GNOME.

Figura 3.16
GParted.

La herramienta 'GParted' es un clásico entre los gestores de particiones en Linux, aunque no viene instalada en Ubuntu por defecto. Puede trabajar con MBR, GPT y permite formatear las particiones. Podemos descargar la aplicación a través de 'Software de Ubuntu'.

Al igual que GNOME-disks, GParted presenta un entorno muy intuitivo. A la derecha se selecciona el disco y en el cuerpo central aparece un esquema gráfico con las dimensiones de las particiones y el espacio ocupado en cada una de ellas. Marcando una partición o en espacio no particionado, podemos realizar sobre ella multitud de acciones, como:

- ✓ Crear nuevas particiones en espacio no particionado.
- ✓ Eliminar o crear nuevas particiones.
- ✓ Redimensionar o mover particiones.
- ✓ Copiar y pegar particiones en otros discos.
- ✓ Formatear.
- ✓ Asignar un nombre.

Los discos deben de disponer de una tabla de particiones para poder gestionar las particiones que en él se establezcan. Si el disco está "limpio", se ha de crear previamente la tabla de particiones.

En el menú 'Dispositivo', podemos seleccionar la opción 'Crear tabla de particiones' y, a continuación, seleccionar el tipo de tabla de particiones, siendo las más conocidas *msdos* (*MBR*) o *gpt*.

Para crear una nueva partición, marcamos el espacio sin asignar o no particionado, y en el menú 'Partición' seleccionamos 'Nueva'. Es entonces donde se establecen todos los parámetros de la partición, como el tamaño, el sistema de archivos para implantar, la etiqueta o el tipo de partición (primaria, extendida o lógica) que, en el caso de GPT, solo puede ser primaria (puesto que, como ya sabemos, no existen particiones lógicas o extendidas).

Además, se pueden redimensionar las particiones, reduciendo su tamaño desde el principio de la misma o por el final, e incluso, si hay espacio sin asignar en el disco, también se pueden ampliar.

Actividad propuesta 3.11

Accede a la herramienta 'Discos de GNOME', observa la información que muestra y navega por las distintas opciones que ofrece para realizar sobre las distintas unidades.

Instala 'GParted' y, a partir de un nuevo disco, crea la tabla de particiones, crea nuevas particiones en GPT, instala diferentes sistemas de archivos, etiquétalos y redimensiónalos.

En cuanto a las herramientas por línea de comandos, la más característica es *fdisk*. Ya conocemos la utilidad *fdisk –l*, sin embargo, esta herramienta dispone de una gran cantidad de opciones para manejar las particiones de los discos del sistema. Para ejecutar *fdisk* hemos de indicar el disco o partición sobre el que queremos trabajar: *fdisk <disco | partición>*.

Al comienzo de la ejecución del programa nos puede realizar diversos avisos, como, por ejemplo, si detecta que el disco tiene más de 2 TB, conviene emplear un esquema de particionamiento GPT.

Figura 3.17. Uso de fdisk.

Su funcionamiento consiste en introducir la tecla adecuada según el menú del programa. Al pulsar "m" podemos ver un listado de todas las acciones.

Es muy importante guardar y salir antes de terminar con la ejecución del programa; de lo contrario, no tendrán efecto las acciones sobre la tabla de particiones.

En la imagen 3.17 se crea una tabla de particiones nueva GPT, con una partición que ocupa todo el espacio del disco con *fdisk*.

En cuanto al programa *gdisk (GPT fdisk)*, emplea un conjunto de acciones muy similares a *fdisk* pero hemos de tener precaución, ya que las modificaciones que se realicen sobre discos MBR serán modificadas a GPT y, por tanto, pueden causar errores inesperados en BIOS MBR o UEFI en modo heredado. Por otro lado, durante la creación de una partición en *gdisk* nos solicita que indiquemos mediante un código en hexadecimal el tipo de partición. Esta información es una recomendación para que sea reconocida la partición. En *fdisk* también se puede establecer.

Actividad resuelta 3.5

Crear una tabla de particiones GPT en un disco mediante gdisk, estableciendo dos particiones que ocupen espacios similares en cuanto a tamaño.

SOLUCIÓN

Lanzamos el programa pasándole como argumento un disco flash USB de 14.4 GB: *sudo gdisk /dev/sdc*, mostrando la tabla de particiones actual del disco. La primera opción es crear una tabla de particiones nueva "o".

A continuación, creamos una nueva partición "n", y el programa nos preguntará el número de partición, siendo la primera (opción por defecto). Luego preguntará por el sector de comienzo de la partición, e indicaremos la opción por defecto para que no exista espacio sin particionar. Más tarde, hemos de indicar el último sector o tamaño de la partición que en nuestro caso indicamos 7GB (+7G). Por último, pregunta el tipo de partición ,que también estableceremos la opción por defecto.

Figura 3.18
Creación de una primera partición con gdisk.

Después, mostramos la tabla de particiones para comprobar las acciones tomadas.

Ahora repetimos los pasos anteriores, como indica la figura 3.20, pero sin indicar el tamaño de la partición, con objeto de ocupar el resto de espacio en disco.

Figura 3.19
Creación de una primera partición con gdisk.

Y, por último, hacemos efectivas las modificaciones hechas con "w".

Figura 3.20
Guardar los cambios con gdisk.

3.6.3. Formatear

Ya sabemos que muchas herramientas de particionamiento, como *GNOME-disks*, *GParted* y *parted*, permiten formatear las particiones creadas. Pero existen otras en modo texto que se encargan únicamente de realizar esa función.

El comando *mkfs* presenta la siguiente sintaxis:

```
mkfs [-t tipo_sistema_archivos][opciones] dispositivo
```

Donde *dispositivo* es normalmente una partición, y las opciones más comunes son:

- t tipo_sistema_archivos: tipo de sistema de archivos que se desea implantar como *ext2*, *ext3*, *ext4*, *vfat (msdos)* o *ntfs*, entre otros.
- opciones: se pueden especificar otras muchas opciones, como etiquetas, formato rápido, tamaño del cluster, etc. Estas opciones varían según el constructor, al que llama *mkfs*. Por tanto, para mayor detalle, hemos de familiarizarnos con la ayuda (*man*) de estos comandos.

Para formatear una partición, el dispositivo ha de estar desmontado.

Ejemplo: *mkfs –t ntfs /dev/sdc1*

mkfs es un *front-end* de otros programas que implantan sistemas de archivos. Es decir, *mkfs* llama al programa constructor del sistema de archivos, como son *mkfs.ext2*, *mkfs.ext3*, *mkfs.ext4*, *mkfs.vfat*, *mkfs.ntfs*, etc., dependiendo del sistema de archivos para implantar. Y algunos de estos son enlaces simbólicos a otros, como *mke2fs*, *mkntfs* o *mkfs.fat*. Podemos analizarlo si ejecutamos *ls -l /sbin/mk**.

Por tanto, las siguientes acciones son equivalentes:

```
sudo mkfs.ntfs /dev/sdc1
sudo mkntfs /dev/sdc1
sudo mkfs -t ntfs /dev/sdc1
```

Además de los sistemas de archivos antes descritos (propios del núcleo), podemos incorporar otros que sean reconocidos por GNU/Linux. Como muestra de ello, el sistema de archivos exFAT no se incorpora dentro del núcleo, por lo que si deseamos hacer uso de él para montar, leer, escribir o formatear un sistema de archivos de este tipo, podemos instalar el paquete asociado: *sudo apt install exfat-utils*.

Esta orden instalará los paquetes *exfat-utils* y *exfat-fuse*. El término *FUSE* (Filesystem in Userspace) hace mención a que el nuevo sistema de archivos se implementará y será gestionado aparte de los sistemas de archivos propios del núcleo.

Ejemplo: *sudo mkfs –t exfat –n pendriveExFAT /dev/sdd1*

A partir de este momento, ya podríamos hacer uso de este sistema de archivos cuando sea montado.

3.6.4. Desfragmentación

La fragmentación de un sistema de archivos se entiende como la disgregación o el esparcimiento de datos, relacionados entre sí, en el medio de almacenamiento. En principio, este hecho no presenta problema alguno, puesto que el sistema de archivos conoce y gestiona los bloques de datos de cada archivo. Sin embargo, los discos duros mecánicos se ven penalizados por esta característica, ya que el cabezal de lectura y escritura ha de oscilar continuamente para seguir los bloques de un archivo, deteriorándose los tiempos de lectura y escritura de dichos bloques.

Cuando existe mucha fragmentación en un disco, se generan multitud de huecos donde se pueden almacenar, o no, bloques de archivos distintos. Al proceso de unión de los bloques de datos de un mismo archivo se le denomina *desfragmentación*.

La fragmentación y sus características dependen de cada sistema de archivos. En sistemas de archivos *NTFS* y *FAT* la desfragmentación es común para mejorar su rendimiento.

Los sistemas de archivos Linux, en general, no necesitan desfragmentarse, puesto que emplean sistemas inteligentes de asignación de bloques a ficheros, de manera que raramente estos se han de desfragmentar. Normalmente, dejan un espacio en bloques considerable entre ficheros para futuros crecimientos, evitando que el crecimiento de un fichero suponga la división de bloques en otra parte del disco y que, si se reduce un fichero, pueda ocupar el espacio libre una parte de otro fichero, evitando más fragmentación.

Sin embargo, sistemas de archivos Linux con dispositivos con poco espacio en disco y mucho movimiento, creación y eliminación frecuente de ficheros es posible que se deban

desfragmentar. Por ello, el sistema de archivos *ext4* puede emplear la utilidad *e4defrag* para este propósito.

El comando *e4defrag* presenta la siguiente sintaxis:

```
e4defrag [-cv] objetivo
```

Donde *objetivo* puede ser un archivo, un directorio o un dispositivo con sistema de archivos *ext4 que* emplea *extents*. Las opciones más comunes son:

- ✓ *c*: muestra un recuento de la fragmentación actual con respecto a la fragmentación ideal. Si se combina con la opción *v*, lo muestra para cada archivo. Con esta opción no se hace efectiva la fragmentación.
- ✓ *v*: muestra el recuento para cada archivo antes y después de la desfragmentación.

Ejemplo de uso de *e4defrag*

La figura 3.21 muestra una estimación de la desfragmentación:

Figura 3.21
Estimación de desfragmentación.

La figura 3.22 muestra la ejecución de desfragmentación en el directorio actual:

```
luis@luis-VirtualBox:~$ sudo e4defrag -v .
```

Figura 3.22
Ejecución de desfragmentación.

La figura 3.23 muestra la parte final del proceso de desfragmentación, ofreciendo un resumen

Figura 3.23
Parte final del proceso de desfragmentación.

Actividad propuesta 3.12

Desfragmenta una unidad con sistema de archivos *ext4*.

3.6.5. Chequeo

El componente más importante en cualquier sistema informático son los datos. Estos tienen un valor incalculable y, si se pierden, sin tener una copia de los mismos, los resultados pueden ser catastróficos.

Los discos duros, a lo largo de su vida útil, suelen ser fuente de diversos problemas, originados internamente o por agentes externos. Nos referimos a:

a) Malware.
b) Fallos en componentes electrónicos del disco duro (normalmente en su placa de circuito impreso por humedad, condensación o suministro eléctrico).
c) Fluctuaciones de tensión en el suministro de energía. Ya sean por picos de tensión, bajadas de esta, fluctuaciones periódicas o cortes de luz.
d) Daños físicos, como caídas o simplemente elementos deteriorados por el uso. Especialmente en discos mecánicos, ya que estos son más propensos al desgaste o problemas debidos por alguna de sus partes móviles (motor, cabezal de lectura-escritura, etc.).
e) Errores firmware o de actualización de drivers.

La gran mayoría de discos duros emplea la tecnología *S.M.A.R.T.* (Self Monitoring Analysis and Reporting Technology) con capacidad para detectar e informar de errores o fallos.

Cuando se combina esta tecnología junto con software compatible, ya sea la propia BIOS del equipo u otro software de terceros instalado en el sistema operativo, obtenemos información muy valiosa sobre el estado actual del disco, e incluso puede avisar de un fallo inminente.

Recursos web

Herramientas compatibles con la tecnología SMART:

- GSmartControl para Linux: https://gsmartcontrol.sourceforge.io/
- CrystalDiskInfo para Microsoft Windows: https://crystalmark.info/en/software/crystaldiskinfo/

La monitorización del sistema de archivos se ha de realizar continuamente o chequearse periódicamente dependiendo de la importancia de los datos que almacene el disco, por lo que es muy recomandable instalar una aplicación de monitorización del estado de los discos duros.

En cuanto a utilidades propias de Linux, este dispone principalmente de los comandos *fsck* y *e2fsck*. Al igual que ocurre con *mkfs*, *fsck* es un *front-end* de otros programas que chequean sistemas de archivos, como *e2fsck* que permite chequear sistemas de archivos de la familia *ext*. Por tanto, este último es el apropiado y el que se ejecuta para chequear sistemas *ext2*, *ext3* o *ext4*. Podemos ejecutar *ls –l /sbin/*fsck** para listar los ejecutables relacionados con el chequeo.

El comando *fsck* presenta la siguiente sintaxis:

```
fsck [-A][-V][-t tipo_sistema_archivos][-a][-r] [sistema_de_archivos]
```

Donde las opciones más comunes son:

- A: Chequea todos los sistemas de archivos establecidos en */etc/fstab*. Esta opción se lanza durante el arranque de Linux. Con esta opción no se puede emplear el argumento *sistema_de_archivos*.
- V (verbose): detalla las acciones realizadas por fsck.
- t tipo_sistema_archivos: tipo de sistema de archivos que se desea chequear como *ext2*, *ext3*, *ext4*, *fat (msdos)* o *ntfs*, entre otros.
- a: repara sin pedir confirmación.
- r: pide confirmación antes de reparar los daños.

Ejemplo: *sudo fsck –t ext4 /dev/sdb1*

El comando *e2fsck* presenta la siguiente sintaxis:

```
e2fsck [-pcnyvD] sistema_archivos
```

Donde las opciones más comunes son:

- ✓ p: repara el sistema de archivos automáticamente. No es compatible con las opciones –n o –y.
- ✓ c: hace uso del programa *badlocks*, que busca bloques defectuosos. En caso de localizar alguno, lo añade a una lista de bloques defectuosos y evita así su uso.
- ✓ y: responde afirmativamente a todas las respuestas que *e2fsck* pudiera plantear.
- ✓ n: responde negativamente a todas las respuestas que *e2fsck* pudiera plantear.
- ✓ v: detalla las acciones realizadas por *e2fsck*.
- ✓ D: optimiza los directorios para un mejor acceso a los datos.

Ejemplo: *sudo e2fsck /dev/sdb1 -c*

Por tanto, las anteriores acciones son equivalentes a *sudo fsck.ext4 /dev/sdb1*.

Además de los sistemas de archivos propios del núcleo, podemos incorporar otros que sean reconocidos por GNU/Linux y así, pues, chequearlos con la utilidad correspondiente (como, por ejemplo, la utilidad *exfatfsck*).

Es altamente recomendable desmontar un sistema de archivos antes de proceder a su chequeo, así aseguramos la integridad de los datos, al no estar usándose los archivos asociados. Además, cuando se hayan efectuado cambios mediante estos programas de chequeo, es aconsejable reiniciar el sistema (*shutdown –r now*).

RECUERDA

✓ El fichero */etc/fstab* automatiza el chequeo de los sistemas de archivos durante el arranque mediante *fsck*, estableciéndose su prioridad en su columna *pass*. Por ello, es importante establecer a 1 el valor *pass* para el sistema de archivos raíz de Linux, ya que, de lo contrario, no podremos hacer el chequeo con garantías una vez el sistema se haya iniciado.

3.6.6. RAID

Cuando hablamos de seguridad en la información y eficiencia en la accesibilidad a los datos en los discos duros, debemos detenernos en el concepto de RAID de forma obligatoria. Es una de las maneras más eficaces de evitar la pérdida de datos y aumentar el rendimiento en tareas de lectura o escritura en discos duros.

RAID (Redundant Array of Independent Disks) consiste en establecer un modo de trabajo, nivel o configuración de un grupo de discos para aumentar la integridad, la capacidad total de almacenamiento, la velocidad de transferencia o disminuir el riesgo a fallos. Para establecer esta configuración, se puede realizar mediante software (propio o no del sistema operativo) o mediante hardware específico (tarjeta controladora o chipset de la placa base). La forma de realizar un nivel RAID es distribuyendo o redundando los datos entre varios discos de diferentes maneras.

A) Tipos de RAID

Los niveles RAID más empleados son los siguientes:

- RAID 0. Se encarga de dividir o distribuir los datos entre dos o más discos sin duplicar la información, es decir, *no existe redundancia de datos*. Por este motivo, no se considera una configuración propia RAID, pero aumenta la velocidad de lectura y escritura.
- RAID 1. Establece una copia exacta entre dos o más discos. Esto permite aumentar la fiabilidad de los datos al quedar estos duplicados en tantos discos como se desee. Además, aumenta la velocidad de lectura.

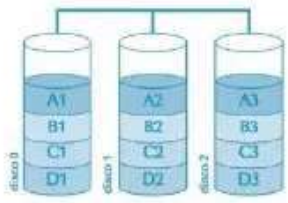


Figura 3.24
Esquema RAID 0.

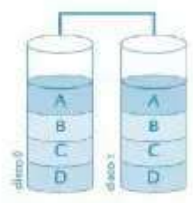


Figura 3.25
Esquema RAID 1.

- RAID 5. Al igual que RAID 0, realiza una distribución de los bloques de datos y genera información de paridad que se distribuye en todos los discos (al menos tres). Los bloques de paridad permiten reconstruir un disco en caso de fallo. Para ello, han de realizar cálculos de los datos, generando dicha paridad, también llamada *código de detección de error* o *CRC*. De este modo, no se desaprovecha tanto espacio redundante, como RAID 1 y, además, mejora la velocidad de lectura, si bien las escrituras son más costosas al deber generar códigos CRC.

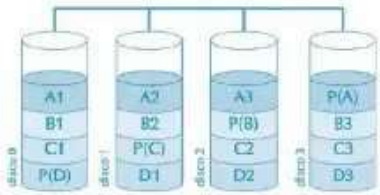


Figura 3.26
Esquema RAID 5.

- Combinaciones RAID: RAID 1+0, RAID 0+1 y RAID 5+0. También se pueden establecer combinaciones de niveles RAID anidando estos y aprovechando las ventajas de varias configuraciones. Así, destacamos los siguientes niveles anidados:

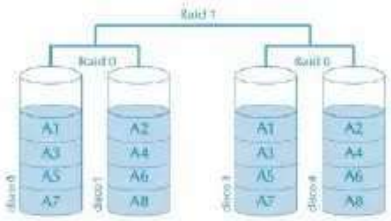


Figura 3.27
Esquema RAID 10.

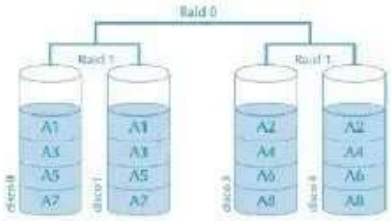


Figura 3.28
Esquema RAID 01.

SABÍAS QUE...

Es frecuente emplear el término *JBOD o RAID Lineal* al método de combinar diferentes discos físicos en uno solo lógico. *JBOD*, por tanto, no presenta redundancia ni mejora el rendimiento del conjunto, sin embargo, el tamaño global es la suma de todos ellos.

B) Administración de RAID

En Linux, la gestión y administración de RAID se realiza mediante el paquete *mdadm (Multiple Device Administrator)*, el cual podemos instalar mediante: *sudo apt install mdadm*.

Antes de proceder, podemos comprobar si existe algún dispositivo RAID (multidispositivo) en el sistema, para lo que visualizamos el archivo */proc/mdstat* mediante *cat /proc/mdstat*. Y en

su resultado podemos comprobar en la línea *Personalities* los tipos de niveles RAID soportados por el núcleo Linux actualmente.

La creación del RAID se puede hacer sobre los dispositivos o sobre particiones, y no necesariamente del mismo tamaño. En caso de que empleen diferente tamaño, *mdadm* avisará si este es más del 1% entre cualquiera de ellos y tomará el tamaño más pequeño. De cualquier modo, lo normal es emplear varios discos con la misma capacidad. Los comandos más comunes empleados para la gestión de RAID en Linux son los siguientes:

1. Creación de RAID.

```
mdadm --create /dev/mdX --level=Y --raid-devices=Z dispositivos
```

Donde:

- ✓ *create /dev/mdX* indica la creación del multidispositivo, siendo X un número.
- ✓ *level=Y* es el nivel RAID para aplicar, pudiendo ser Y:
  - *linear* para RAID lineal.
  - *raid0, 0* o *stripe* para RAID0.
  - *mirror, raid1* o *1* para RAID1.
  - *raid4* o *4* para RAID4.
  - *raid5* o *5* para RAID5.
  - *raid6* o *6* para RAID6.
  - *raid10* o *10* para RAID10.
- ✓ *raid-devices=Z dispositivos*, donde *Z* indica el número de dispositivos asociados al RAID y cada uno de ellos separado por espacios (*/dev/sdX /dev/sdY* ...).

2. Establecer un disco como defectuoso de un RAID:

```
mdadm /dev/mdX --fail /dev/sdY
```

3. Eliminar un disco de un RAID:

```
mdadm /dev/mdX --remove /dev/sdY
```

4. Añadir un disco a un RAID:

```
mdadm /dev/mdX --add /dev/sdY
```

5. Comprobar el estado de todos los multidispositivos:

```
cat /proc/mdstat
```

6. Obtener información de todos los multidispositivos:

```
mdadm --detail --scan
```

7. Obtener información de un multidispositivo:

```
mdamd --detail /dev/mdX y mdadm --detail /dev/mdX --scan
```

8. Examinar el estado de un dispositivo asociado a un RAID:

```
mdadm --examine /dev/sdX
```

9. Detener un RAID:

```
mdadm --stop /dev/mdX
```

10. Eliminar el superbloque de un dispositivo sobreescribiendo ceros:

```
mdadm --zero-superblock /dev/sdY
```

Actividad resuelta 3.6

Crear un RAID5

SOLUCIÓN

Antes de comenzar, hemos de disponer de al menos tres dispositivos y, en nuestro caso, hemos creado una partición que ocupa la totalidad de cada uno: */dev/sdc1*, */dev/sdd1* y */dev/sde1*, todos ellos de 10 G.

A continuación, creamos el dispositivo RAID, indicando el nombre del multidispositivo (/dev/mdX donde X es un número que empieza en 0 y ha de estar libre), el número de dispositivos que lo constituirán y las unidades de almacenamiento:

```
sudo mdadm -- create /dev/md0 --level=5 --raid-devices=3 /dev/
                    sdc1 /dev/sdd1 /dev/sde1
```

Podemos visualizar la información del RAID mediante: *mdadm - -detail - -scan*, y la construcción del mismo mediante *mdadm - -detail /dev/md0* y *cat /proc/mdstat*. Con *mdadm* se informa del estado de construcción en la línea *Rebuild Status*. Una vez construido, todos los dispositivos aparecerán activos y en sincronía con el resto, como aparece en la siguiente imagen.

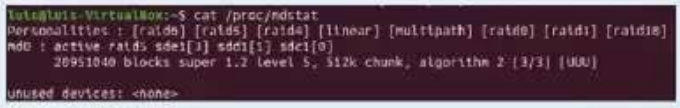


Figura 3.29
Estado de los multidispositivos.

Una vez completada la construcción del RAID, si volvemos a analizar el estado de los multidispositivos del sistema, comprobamos la existencia de uno nuevo. Ahora debe aparecer *md0* activo con nivel RAID5 formado por los dispositivos *sde1*, *sdd1* y *sdc1*.

Y para poder usar *md0*, solo debemos implantar un sistema de archivos al multidispositivo y montarlo en un directorio existente con permisos suficientes para hacer uso de él:

```
sudo mkfs.ext4 /dev/md0
sudo mount /dev/md0 /media/Luis/RAID
```

Actividad resuelta 3.7

Montar al inicio del sistema un multidispositivo.

SOLUCIÓN

Al ser *md0* un dispositivo más, podemos hacer que permanezca ante reinicios. Los pasos serían los siguientes:

1. Ubuntu modifica el nombre de los multidispositivos cuando reinicia el sistema. Para asignar un nombre multidispositivo a un grupo de discos en RAID, debemos añadir una línea en el archivo de configuración */etc/mdadm/mdadm.conf* ejecutando el script */usr/share/mdadm/mkconf* o añandiendo la línea directamente mediante *mdadm - -detail /dev/md0 - - scan >> /etc/mdadm/mdadm.conf*, y después actualizamos el *initramfs (sistema de archivos RAM de inicio de Linux)* mediante el comando *update-initramfs –u*:

```
sudo /usr/share/mdadm/mkconf | sudo tee /etc/mdadm/mdadm.conf
sudo update-initramfs -u
```

2. Como ya sabemos, debemos añadir una línea en */etc/fstab*, indicando que monte durante el inicio del sistema el multidispositivo *md0* (en este caso).

Actividad resuelta 3.8

Simulación de fallos en RAID5

SOLUCIÓN

Siguiendo con el ejemplo práctico anterior, vamos a simular fallos en discos de sistema RAID5 en el multidispositivo *md0*. Supongamos que el sistema, mediante algún software que genera avisos S.M.A.R.T., indica que el disco */dev/sdc* no funciona bien y que disponemos de otro disco con las mismas características que el resto del RAID.

Primero, marcamos como defectuoso el disco */dev/*sdc y comprobamos que su estado es *"(F)" faulty*:

```
sudo mdadm /dev/md0 --fail /dev/sdc1
cat /proc/mdstat
```

A continuación, quitamos el disco del RAID, restando aún dos discos más, y lo comprobamos:

```
sudo mdadm /dev/md0 --remove /dev/sdc1
cat /proc/mdstat
```

Por último, añadimos un nuevo disco */dev/sdg1*, reconstruyendo automáticamente el RAID5 y comprobamos:

```
sudo mdadm /dev/md0 --add /dev/sdg1
cat /proc/mdstat
```

Si añadiésemos otro disco en RAID5 (sumando cuatro discos), sin estar ninguno de los existentes defectuoso, quedaría en modo repuesto (*spare*) ante futuros fallos de otro disco, es decir, el sistema lo sincronizaría automáticamente, si fallase uno de los otros. Esto se debe a que RAID5 emplea tres dispositivos, como mínimo.

Si deseamos eliminar un multidispositivo RAID y así evitar que aparezcan en modo inactivo o que algunos dispositivos no se puedan usar por estar asociados a otros RAID no activos, debemos:

1. Desmontar el dispositivo si está en uso.
2. Detener el multidispositivo RAID.

```
sudo mdadm --stop /dev/md0
```

3. Borrar el superbloque de cada dispositivo que constituía el RAID:

```
sudo mdadm --zero-superblock /dev/sde1
sudo mdadm --zero-superblock /dev/sdd1
sudo mdadm --zero-superblock /dev/sdg1
```

4. En caso de que estuviera asociado al arranque del sistema, actualizar */etc/fstab* eliminando la línea asociada y actualizar *initramfs*.

3.7. Gestión de almacenamiento por interfaz gráfica en Microsoft Windows

En Microsoft Windows la gestión del almacenamiento se realiza principalmente mediante el 'Administrador de discos'. Se puede acceder a través del 'Panel de control', 'Herramientas administrativas' y 'Administración de equipos'.

En él figuran detalladamente cada uno de los discos instalados en el sistema y sus particiones. De tal manera que podemos estudiar el tamaño total, capacidad, espacio libre y usado. Para cada disco o partición podemos realizar una serie de acciones asociadas: crear particiones, formatear, eliminar particiones, cambiar las etiquetas o letras de la unidad, etc.

Otra utilidad de Microsoft Windows es 'Almacenamiento'. Se puede acceder dentro de Sistema en 'Configuración'. En ella, podemos ver el espacio libre de cada unidad y, además, al acceder a cada una de ellas, se estudia el 'Uso de almacenamiento', donde el sistema realiza un análisis de los tipos de archivos ordenados por el espacio que ocupan. A su vez, por cada tipo, resultan ordenados por tamaño. También se puede acceder a cada archivo de los que lo conforman o realizar sobre ellos actividades propias de cada tipo, como, por ejemplo: desinstalar una aplicación, administrar la restauración del sistema, acceder a un archivo en concreto o administrar las carpetas de descargas.

Además, el 'Explorador de archivos' de Microsoft Windows facilita la gestión de los archivos sin necesidad de conocer la gestión interna del sistema. Desde este, es muy sencillo realizar diferentes acciones sobre las unidades, como formatear, desfragmentar o chequear unidades o particiones.

Para particionar un disco en Microsoft Windows se realiza desde el 'Administrador de discos' y seleccionando el espacio no particionado del disco con el botón secundario del ratón se marca 'Nuevo volumen simple…'.

Figura 3.30
Creación de partición en Microsoft Windows.

Sucesivamente, nos solicitará que introduzcamos el tamaño de la partición, la letra para asignarle a dicha unidad y si deseamos formatearla, indicaremos el tipo de sistema de archivos, el tamaño del clúster y la etiqueta identificativa.

Se puede dar la circunstancia, por diversos motivos, que Microsoft Windows no reconozca una unidad o una partición y, por tanto, el 'Explorador de archivos' no la mostrará. En tales casos, desde el 'Administrador de discos' debemos inicializar el disco (situándonos sobre el recuadro del disco e 'Inicializar disco') o asignarle una letra a una unidad pulsando en 'Cambiar la letra y rutas de acceso de unidad'.

Estando creada una partición, se puede formatear desde el 'Administrador de discos' o el propio 'Explorador de archivos' mediante el menú contextual de la unidad.

Y desde la opción 'Propiedades' (pestaña 'Herramientas') del menú contextual del 'Explorador de archivos' sobre una unidad, podemos chequearla y desfragmentarla.

TOMA NOTA

Todas estas opciones también están presentes en la cinta de opciones del 'Explorador de archivos', en la pestaña 'Herramientas de unidad'.

Por otro lado, con Microsoft Windows, podemos hacer uso del 'Administrador de discos' para añadir a los discos otras características y funcionalidades, convirtiéndolos en 'discos dinámicos'. Con este concepto, Microsoft Windows permite crear distintos tipos de volúmenes, y algunos de ellos de tipo RAID, a saber:

- Volumen distribuido: consiste en unir diferentes espacios de diferentes discos bajo una misma unidad lógica.
- Volumen reflejado: conocido también como RAID-1. Genera redundancia de datos.
- Volumen seccionado: también llamado RAID-0. Genera un gran rendimiento, pero un fallo en cualquiera de sus discos hará que el volumen falle.
- Volumen RAID-5: permite distribuir los datos entre tres o más discos con tolerancia a fallos de forma distribuida. Es accesible desde versiones de Windows Server.
- Volumen simple: una única unidad en un disco.

Hemos de tener en consideración que la conversión de discos básicos a dinámicos no implica la pérdida de datos, sin embargo, de dinámicos a básicos sí. El procedimiento es muy sencillo: desde el 'Administrador de discos' hemos de convertir los discos a dinámicos, si estos no lo son, y han de estar inicializados.

Pulsando con el botón secundario del ratón sobre uno de los discos, nos dará a elegir entre las distintas posibilidades de discos dinámicos (puede ser que alguna de ellas aparezca sombreada, debido a que el sistema detecta imposibilidad en la acción). Posteriormente, comenzará el asistente para la creación de un nuevo volumen del tipo seleccionado, resultando un proceso muy sencillo e intuitivo en el que solo tenemos que seguir los pasos.

Actividad resuelta 3.9

Crear un volumen reflejado en Microsoft Windows 10 desde el 'Administrador de discos'.

SOLUCIÓN

Suponemos que hemos instalado tres discos con espacio no asignado, tal y como aparece en la siguiente imagen. Inicializamos los discos y los convertimos a dinámicos.

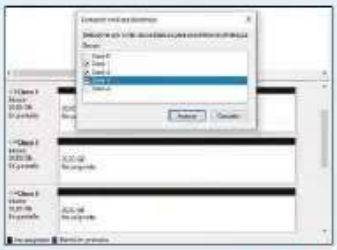


Figura 3.31
Creación de discos dinámicos.

Seleccionamos la opción 'Nuevo volumen reflejado' y se desplegará el asistente de creación en el que seguimos los pasos empleando todo el espacio de cada disco (se puede emplear menos), seleccionamos una letra a la unidad, implantamos el sistema de archivos NTFS e indicamos una etiqueta a la unidad (en nuestro caso, RAID1).

Una vez terminada la configuración con el asistente, aparecerá un resumen de las características del nuevo volumen y, tras pulsar en 'Finalizar', el sistema creará el nuevo volumen, que se reflejará como una unidad más.

Figura 3.32
Selección de discos en el asistente para nuevo volumen reflejado.

Figura 3.33
Asignación de letra en el asistente para nuevo volumen reflejado.

Figura 3.34
Selección de la configuración para formatear el nuevo volumen reflejado.

Figura 3.35
Nuevo volumen reflejado.

Por otro lado, Microsoft Windows dispone de otra herramienta llamada 'Espacios de almacenamiento', a la que podemos acceder a través de 'Almacenamiento' y 'Administrar espacios de almacenamiento', o mediante el 'Panel de control'.

'Espacios de almacenamiento' permite crear unidades virtuales agrupando dos o más unidades en un grupo de almacenamiento. Esto facilita, sin apenas conocimientos y mediante una gestión muy sencilla, la administración de espacios simples, similar a RAID0 (ya que no protegen los datos, pero aumentan el rendimiento), espacios de reflejo (doble o triple), similar a RAID1, y espacios de paridad, similar a RAID5.

Dentro de 'Espacios de almacenamiento', su creación es muy sencilla. Basta con acceder a 'Crear un nuevo grupo y espacios de almacenamiento'. Posteriormente, seleccionamos la etiqueta, la unidad, el sistema de archivos para implantar, el tipo de espacio de almacenamiento y su tamaño.

Figura 3.36
Creación de un espacio de almacenamiento.

Figura 3.37
Administración de un espacio de almacenamiento.

Una vez creado, se puede administrar el espacio de almacenamiento agregando unidades, cambiando el nombre del grupo, etc.

En estas condiciones, el sistema avisa cuando una unidad física se encuentra defectuosa. En ese caso, se debe agregar una nueva unidad física para solventar el problema.

3.8. Búsqueda de información por línea de comandos en Linux

Los sistemas de archivos han de ofrecer herramientas para localizar archivos por diferentes criterios (fecha de creación, fecha de modificación, tamaño, nombre, etc.) y búsqueda de información en el contenido de estos.

Uno de los comandos más empleados en Linux es sin duda *find*. Este comando permite multitud de opciones para localizar cualquier fichero en el sistema de archivos. Su sintaxis es la siguiente:

```
find [ruta] [criterio] [acción]
```

El comando *find* realiza una búsqueda sobre la ruta dada (pueden ser varias). Si no se especifica ruta, la búsqueda se realiza sobre el directorio actual. El resultado y los errores se envían a las salidas estándares por defecto. En caso de no indicar criterio alguno, no se hará ningún tipo de filtro. Además, la acción permite tratar los ficheros encontrados mediante alguna acción.

3.8.1. Criterios de búsqueda

Los criterios más importantes de búsqueda son los que se exponen a continuación:

A) Por nombre de fichero

Están las siguientes opciones:

- ✓ name *patrón*: permite buscar ficheros permitiendo la búsqueda con patrones.
- ✓ iname patrón: actúa del mismo modo que name, pero sin distinguir mayúsculas de minúsculas.

Ejemplos:

- *find . –name texto.txt*. Busca en el directorio actual recursivamente archivos con el patrón "texto.txt".
- *find . –name 't*.txt'*. Busca en el directorio actual recursivamente archivos que comiencen por "t" y terminen en ".txt".
- *find . –iname texto.txt*. Busca en el directorio actual recursivamente archivos con el patrón "texto.txt" sin distinguir mayúsculas de minúsculas.

B) Nivel de profundidad en subdirectorios

Por defecto, *find* busca recursivamente en directorios a partir de la ruta especificada, pero podemos limitar el nivel de profundidad máximo (hasta donde llega la búsqueda) y mínimo (desde dónde empieza la búsqueda).

- ✓ maxdepth n: especifica hasta qué subdirectorios se realiza la búsqueda. La ruta especificada se encuentra en el nivel 1, un subdirectorio dentro de este en el nivel 2 y así sucesivamente.
- ✓ mindepth n: se especifica desde qué nivel comienza la búsqueda. Si se indica el nivel 2, buscará desde los subdirectorios de la ruta especificada recursivamente.

Ejemplos:

- *find . –maxdepth 1 –name texto.txt*. Limita la búsqueda de ficheros con nombre "texto.txt" en el directorio actual sin entrar en subdirectorios.
- *find . –maxdepth 2 –name texto.txt*. Limita la búsqueda de ficheros con nombre "texto.txt" al directorio actual y subdirectorios.
- *find . –mindepth 2 –name texto.txt*. Comienza la búsqueda de ficheros con nombre "texto.txt" desde los subdirectorios del directorio actual.
- *find . –maxdepth 2 –mindepth 2 –name texto.txt*. Limita la búsqueda de ficheros con nombre "texto.txt" únicamente a subdirectorios del directorio actual.

C) Tiempos de acceso, modificación y cambio

Podemos hacer búsquedas atendiendo a la fecha de última modificación del i-nodo (*c*), última modificación de su contenido (*m*) y último acceso a su contenido (*a*). Para cualquiera de ellos, se puede especificar el tiempo en minutos (min) o en días (time). De tal manera que podemos especificar las siguientes opciones: *cmin*, *mmin*, *amin*, *ctime*, *mtime* y *atime*. Los valores numéricos que acompañan a las opciones pueden ser: *+n* indicando mayor que, *-n* indicando menor que y *n* indicando igualdad.

Ejemplos:

- ✓ *find . –amin -1*. Localiza archivos que se accedieron hace menos de un minuto.
- ✓ *find . –mtime -1*. Localiza archivos que se modificaron hace menos de un día.

D) Comparación de ficheros

También se pueden localizar ficheros comparándolos con otro fichero. A saber:

- newer fichero: busca ficheros que se modificaron más recientemente que *fichero*.
- anewer fichero: busca ficheros accedidos más recientemente que *fichero* fue modificado.
- cnewer fichero: busca ficheros en los que que el estado del *i-nodo* se modificó más recientemente que *fichero* fue modificado.

Ejemplo:

find . –anewer notas.txt. Busca ficheros cuya fecha de acceso fue más reciente que la modificación de notas.txt.

E) Tamaños

Podemos especificar comparaciones con tamaños. Para lo que podemos emplear la siguiente opción: -size [+-]n[bckMG]. Donde cada parámetro indica:

- ✓ +n indica mayor que n, -n indica menor que n y n indica igualdad, siendo n un valor numérico.
- ✓ b bloques de 512 bytes.
- ✓ c bytes.
- ✓ k kilobytes.
- ✓ M megabytes.
- ✓ G gigabytes.

Ejemplo:

find . –size +1M. Busca archivos mayores de 1 megabyte.

F) Tipo de fichero

Se pueden realizar búsquedas por el tipo de fichero mediante la opción –type con alguno de los siguientes modificadores: l (enlace simbólico), d (directorio), f (fichero regular), b (dispositivo de tipo bloque) y c (dispositivo de tipo carácter).

Ejemplo:

find ./nivel2 –type d . Busca directorios a partir del subdirectorio "nivel2".

G) Permisos

Las búsquedas se pueden efectuar sobre los permisos mediante la opción *perm [-/] permisos*. Se pueden establecer los permisos en octal o de manera simbólica. El signo "-" indica que el fichero debe contener al menos los permisos dados y "/" indica que debe tener alguno de los permisos que se dan. Si no se indica ninguno de estos dos signos, los permisos deben de ser idénticos a los especificados.

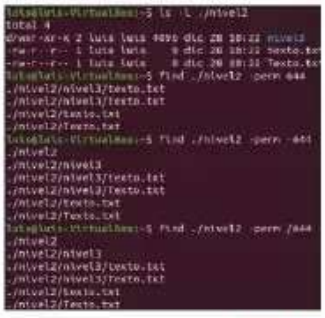


Figura 3.38
Uso de find con la opción -perm.

Ejemplos:

La figura 3.38 muestra varios ejemplos en los que se puede observar cómo para los permisos dados en el subdirectorio *nivel2*, los archivos encontrados con la opción -perm 644 son aquellos que tienen exactamente esos permisos (rw-r- - r- -). Con -perm -644 se suman los directorios nivel2 y nivel3, puesto que disponen de los permisos anteriores más dos de ejecución (rwxr-xr-x). Y -perm /644 son los mismos que los anteriores, puesto que todos ellos tienen al menos rw- en usuario, r en grupo o r en otros.

H) Otras opciones de búsqueda

- user usuario. Localiza archivos por un usuario dado.
- inum inodo. Busca ficheros por número de inodo.
- uid UID. Busca ficheros por el UID de usuario.
- gid GID. Busca ficheros por GID.

También podemos combinar opciones de búsqueda mediante las siguientes opciones:

- ✓ *criterio1 –and criterio2*: hace que se cumplan dos criterios. Es similar a no indicar nada entre criterios.
- ✓ *criterio1 –a criterio2*: idéntico al anterior.
- ✓ *criterio1 –or criterio2*: hace que se cumpla un criterio u otro.
- ✓ *criterio1 –o criterio2*: idéntico al anterior.
- ✓ *-not criterio*: niega el criterio.
- ✓ *!criterio*: idéntico al anterior.

Además, se puede indicar mediante paréntesis la preferencia entre criterios.

Ejemplos:

- *find . /nivel2 –perm /644 –type d*. Los criterios –perm /644 y –type d se han de cumplir a la vez.
- *find ./nivel2 –perm /644 –o –perm 755*. Los criterios para obtener un resultado pueden ser uno u otro.
- *find . –size +1M –a \(-user root –o –user luis \)*. El criterio *–size +1M* es obligatorio y que se dé alguno de los otros dos criterios: que el usuario sea *root* o *luis*. Como se puede observar, los paréntesis se deben "escapar" con el carácter "\".

3.9. Búsqueda de información por interfaz gráfica en Microsoft Windows

El cuadro de búsqueda del 'Explorador de archivos' es una herramienta muy potente para realizar búsquedas desde la unidad o carpeta actual por diferentes criterios para archivos o carpetas: fecha, modificación, creación, tamaño (vacío, minúsculo, pequeño, mediano, grande, enorme, gigantesco o tamaño exacto), clase (carpeta, vínculo, película, imagen, etc.), extensión, carpeta o archivo. Además, se pueden combinar diferentes criterios mediante operadores lógicos: NOT, OR o &. La cinta de opciones de 'Herramientas de búsqueda' se habilita al situarse sobre el cuadro de búsqueda.

Figura 3.39
Cuadro de búsqueda por tamaño.

Figura 3.40
Cinta de opciones de búsqueda.

Actividades propuestas

3.13. Realiza búsquedas con *find* en Linux, atendiendo a las diferentes opciones que hemos estudiado.

3.14. Realiza búsquedas a través del 'Explorador de archivos' de Microsoft Windows mediante el cuadro de búsqueda y la cinta de opciones de 'Herramientas de búsqueda'.

Recurso digital 3.2

Administrador de medios virtuales de Oracle VM VirtualBox.

Resumen

- Los sistemas operativos modernos ofrecen herramientas para la gestión de archivos y almacenamiento. En este capítulo hemos trabajado tanto desde el punto de vista gráfico (con Microsoft Windows) como textual (con Ubuntu), aunque se ha hecho hincapié en este último, dada su gran potencia y versatilidad.
- Primero se ha estudiado el concepto de sistema de archivos, así como los ejemplos más característicos: FAT, exFAT, NTFS, ext4 y APFS. Más tarde se ha dado a conocer la estructura de directorios de Linux y Microsoft Windows.
- Los comandos más importantes tratados para la gestión de archivos en Ubuntu han sido:

```
ls      cd      touch   pwd     ln      rm      mkdir   rmdir
cp      mv      cat     more    less    head    tail    wc
sort    cut     grep
```

- Por otro lado, hemos estudiado la gestión de almacenamiento mediante los comandos:

```
mount           lsblk           fdisk           lsusb
umount          gdisk           parted          GParted
mkfs            e4defrag        fsck            e2fsck
mdadm
```

- Además, se han estudiado archivos o directorios importantes, tanto en la gestión de almacenamiento como en la configuración general del sistema Ubuntu, como:

```
/etc/passwd     /dev/null       /dev            /proc/self/mounts
/etc/fstab      /proc/mdstat
```

- Desde el punto de vista de la administración gráfica con Microsoft Windows, para la gestión de archivos y almacenamiento hemos trabajado con el 'Explorador de archivos' y el 'Administrador de discos', respectivamente. No obstante, siempre es posible una administración por línea de comandos en Microsoft Windows, aunque no se haya estudiado en este capítulo.

- Por último, se han abordado herramientas para localizar archivos por diferentes criterios, así como búsqueda de información dentro de estos, como *find* en Ubuntu y a través las herramientas de búsqueda del 'Explorador de archivos' en Microsoft Windows.

Ejercicios propuestos

1. ¿Cuáles son los objetivos de los sistemas de archivos?
2. Gestión de archivos en Ubuntu. En Ubuntu, crea la siguiente estructura de directorios a partir del directorio home del usuario:

   a) Copia el archivo */etc/passwd* en el directorio *Plataforma* (estando situado en el directorio /etc).
   b) Copia los archivos que contengan la letra c del directorio */bin* al directorio *Ejercicios*.
   c) Copia todos los archivos que empiecen por "m" o por "n" del directorio */bin* al directorio *Practicas*.
   d) Mueve un solo archivo que empiece por la letra "m" del directorio *Practicas* al directorio *Libros*.
   e) Borra el archivo *mkdir* con confirmación del directorio *Practicas*.
   f) Renombra el archivo *mount* de *Practicas* por *montar*.
   g) Crea un enlace simbólico de *montar* llamado *e_montar*. Crea dos enlaces duros de correo llamados *montar_duro1* y *montar_duro2*. ¿Cuántos enlaces duros contiene *montar*? Compara los números de i-nodo de montar, montar_duro1 y montar_duro2 y justifícalo. Borra *montar*. ¿Se ha convertido e_montar en un enlace roto? ¿Por qué? ¿Y si eliminamos *montar_duro1* y *montar_duro2*?
   h) Copia toda la información que contiene el directorio *Practicas* al directorio *Apuntes*.
   i) Crea un archivo de texto de dos líneas con el comando *cat* en el directorio *SI*. Muestra el número de palabras y líneas de dicho archivo de texto.

3. Procesamiento de textos en Ubuntu:

   a) Concatena los archivos */etc/passwd*, */etc/shadow* y */etc/fstab* en un solo archivo llamado *concatenado*. Todo ello en una sola instrucción.
   b) Muestra el número de usuarios que disponen del shell "bash" como intérprete de comandos. Emplea el fichero */etc/passwd*.
   c) Listar de manera inversamente ordenada solo los grupos primarios de aquellos usuarios cuyo UID comienza por 1. Emplea el fichero */etc/passwd*.

4. Búsqueda de información en Ubuntu:

a) Encuentra los archivos ocultos de tu directorio de trabajo.
b) Busca en todo el sistema los ficheros de tu usuario. Evita mostrar los mensajes de error.
c) Busca todos los archivos que comiencen por "a" en múltiples rutas de forma conjunta: en tu *home* y en */dev*.
d) Busca todos los archivos que comiencen por "ca" pero que no terminen con ".php". Evita mostrar los mensajes de error.
e) Encuentra todos los archivos modificados en la última hora. Evita mostrar los mensajes de error.
f) Busca todos los ficheros que tengan como usuario vuestro usuario que empiecen por "e" y que tenga más de 1K. Evita mostrar los mensajes de error.
g) Busca los ficheros en /etc que tengan permiso de lectura sin entrar en subdirectorios. Evita mostrar los mensajes de error.
h) Crea un fichero llamado fichero1. Después, crea 2 ficheros llamados *fichero2* y *fichero3*. Encuentra aquellos ficheros que se hayan creados posteriormente a fichero1.
i) Modifica fichero2, fichero3 y fichero1 por ese orden, con el contenido que desees. Busca los ficheros que se hayan modificado más recientemente a la modificación de fichero2.
j) Entra en fichero3. Sal. Busca los ficheros cuyo acceso sea más reciente.

5. Búsqueda de información en Microsoft Windows:

a) Busca en todo el equipo aquellas imágenes entre 1 y 128 MB creadas el mes pasado.
b) Busca en el directorio actual aquellos archivos con extensión *.txt* y creados hoy.
c) Busca en todas las subcarpetas aquellos directorios que tengan como nombre *datos* o *copia*.

6. Particiones y volúmenes en Ubuntu. Para el siguiente ejercicio, a partir de una máquina virtual con Ubuntu, añade un nuevo disco duro de 30 GB.

a) Describe los pasos y comandos para crear y poder utilizar un disco duro con la siguiente estructura GPT y tamaño de sus particiones:

Donde cada partición debe ser montada a partir del directorio para crear *$HOME/particiones*.

b) Hacer que la partición *Datos* y *Backup* se monte automáticamente al iniciarse el sistema en modo de solo lectura.

c) Chequea la partición *Backup* y desfragméntala.
d) Disponemos de un archivo *helloween.mp3* en el directorio *musica* de nuestro pendrive con sistema de archivos NTFS que está sin montar y queremos copiarlo al directorio *documentos* de nuestro sistema dentro de nuestro *home*. Al finalizar, hay que desmontarlo. Indica todas las acciones para realizar todo el proceso.

7. Gestión de almacenamiento en Microsoft Windows. A partir de una máquina virtual con Microsoft Windows, añade un nuevo disco duro de 30 GB. A través del 'Administrador de discos', crea tres particiones de 10 GB cada una.

8. Gestión de archivos en Microsoft Windows:

a) A través del 'Explorador de archivos' y continuando el ejercicio anterior, formatea una de las particiones de la unidad con sistema de archivos NTFS, etiquétala con nombre Datos y asigna un tamaño de la unidad de asignación de 4096.
b) Formatea otra de las particiones de la unidad con sistema de archivos *FAT32* y etiquétala como *Compartida*.

9. RAID en Ubuntu. Añade a la máquina virtual de Ubuntu tres discos de 20 GB cada uno:

a) Crea un sistema RAID1 con dos de ellos.
b) Haz que dicho RAID1 sea permanente ante reinicios.
c) Simula el fallo de uno de los discos, márcalo como defectuoso, elimínalo del RAID1 y asocia otro disco para que se sincronice al RAID1.

10. 'Discos dinámicos y espacios de almacenamiento' en Microsoft Windows. Añade a la máquina virtual de Microsoft Windows tres discos de 20 GB cada uno:

a) Crea un sistema RAID1 con dos de ellos a través del 'Administrador de discos'.
b) Comprueba la nueva unidad a través del 'Explorador de archivos'. Deshaz el RAID.
c) Crea un espacio de reflejo con los tres discos mediante 'Espacios de almacenamiento'.

ACTIVIDADES DE AUTOEVALUACIÓN

1. El sistema de archivos exFAT:
   - ☐ *a)* Es más ligero que NTFS y APFS.
   - ☐ *b)* Es más seguro que NTFS.
   - ☐ *c)* Solo permite gestionar archivos de hasta 4 GB.