

MÓDULO I. Administración de GNU/Linux

Sesión 2. Herramientas de administración del sistema de archivos

2025/07/01

Índice

1. Introducción	1
2. Objetivos principales	2
3. Partición de dispositivos de almacenamiento secundario	3
3.1 ¿Cuántas particiones necesito y de qué tipo?	4
3.2 ¿Qué directorios de primer nivel del estándar FHS deberían estar soportados por una partición independiente?	5
4. Asignación de un Sistema de Archivos a una partición (formateo lógico)	11
5. Ajuste de parámetros configurables de un SA y comprobación de errores	12
6. Montaje y desmontaje de Sistemas de Archivos	14
7.2 Gestores de Paquetes	16
7.2.1 Gestión de paquetes en Debian y derivados	16

1. Introducción

Como vimos en la sesión anterior, el estándar FHS nos permite conocer cómo se estructura la información en un sistema GNU/Linux. En estos sistemas existen:

- **Archivos regulares.** Archivos de programa y datos.
- **Directorios.** Archivos específicamente diseñados para soportar la estructura jerárquica de organización de la información en un Sistema de Archivos (SA). Esta estructura implementa lo que en el mundo UNIX/Linux se conoce como espacio de nombres de archivo.
- **Enlaces simbólicos.** Archivos que permiten referenciar a otros archivos desde distintas ubicaciones en el espacio de nombres (distintos directorios).
- **Archivos especiales de dispositivo.** Estos archivos representan dispositivos y permiten al usuario del lenguaje de órdenes del shell y a los programadores de aplicaciones trabajar sobre los dispositi-

tivos como si se tratase de un archivo normal. De hecho, estos tipos de archivo implementan la abstracción de dispositivos que proporcionan los SAs en los sistemas UNIX/Linux. Los sistemas UNIX distinguen dos subtipos de archivos especiales de dispositivo: orientados a dispositivos de bloques y orientados a dispositivos de caracteres.

- **Archivos para comunicaciones FIFO.** Permiten comunicar procesos.
- **Archivos para comunicaciones Socket.** Permiten comunicar procesos.

Esta sesión aborda dos áreas temáticas de la administración de SOs íntimamente relacionadas. Por una parte se presentarán los conceptos y órdenes para administración (herramientas del administrador) relativas al trabajo con dispositivos de almacenamiento secundario y creación, personalización y comprobación de inconsistencias de los SAs en sí mismos. Por otra parte, se mostrarán los conceptos, procedimientos y herramientas que se pueden utilizar para administrar el software de nuestro ordenador. La administración del software incluye la instalación, eliminación, actualización o reconstrucción de paquetes, lo cual resulta de vital importancia para tener un sistema estable y usuarios satisfechos.

Con respecto a la primera área temática se comentarán los conceptos y procedimientos para: Establecer particiones en dispositivos de almacenamiento secundario; asignar un determinado SA a una partición en un dispositivo de este tipo, es decir, lo que se conoce en SOs como formateo lógico; establecer configuraciones personalizadas de la información que el sistema de archivos mantiene sobre si mismo (metainformación o metadatos o atributos del sistema de archivos); hacer visible la información que almacena un SA en nuestro espacio de nombres de archivo actual, lo que se conoce comúnmente por montar un SA; y, finalmente, poder comprobar la consistencia de las estructuras de datos que contiene el SA.

2. Objetivos principales

- Comprender qué es una partición de un dispositivo de almacenamiento secundario, conocer la información que se almacena en la tabla de particiones, y saber utilizar una herramienta que permita llevar a cabo la partición de un dispositivo (`fdisk`).
- Comprender el concepto de formateo lógico de particiones y comprender la familia de órdenes `mkfs*`, su utilización y sus opciones principales.
- Conocer los metadatos básicos de un SA tipo Linux y saber utilizar las herramientas que proporciona Linux para personalizar dicha información, `tune2fs`, y realizar comprobaciones y reparaciones de las posibles inconsistencias que se puedan producir en dicha información como consecuencia del uso del sistema de archivos, `fsck`.
- Comprender el concepto de espacio de nombres de la estructura jerárquica de directorios y saber ampliar el espacio de nombres con nuevos sistemas de archivos mediante la orden `mount`.
- Conocer los principales aspectos e implicaciones relacionados con llevar a cabo una administración correcta y productiva del software instalado en un ordenador con sistema operativo Linux.
- Ser capaz de tomar decisiones acerca de qué software instalar y cómo hacerlo.
- Adquirir destrezas básicas en el manejo de las principales herramientas disponibles para administrar el software de un sistema Linux.

3. Partición de dispositivos de almacenamiento secundario

Para poder utilizar un dispositivo de almacenamiento secundario (drive), como puede ser un disco duro o una memoria flash, en un SO es necesario, en primer lugar, establecer secciones (particiones) dentro del dispositivo físico, que sean identificables, y que permitan alojar un sistema de ficheros (SA) concreto.

En la actualidad podemos encontrar multitud de dispositivos de almacenamiento, que incluyen:

- Discos Magnéticos (HDD) a través de interfaces como SATA.
- Unidades de Estado Sólido (SSD), tanto SATA como los mucho más rápidos NVMe.
- Memorias Flash (pendrives USB, tarjetas SD, etc.).
- Dispositivos RAID (configurados por hardware o software en Linux).

Una partición en cualquiera de estos dispositivos está constituida por un conjunto de sectores que forman un “disco lógico”. Un sector es la mínima unidad de almacenamiento, siendo su tamaño más común 4096 bytes (4 KiB) en dispositivos modernos, aunque también existen de 512 bytes.

El Estándar Moderno: GPT (GUID Partition Table) Mientras que históricamente los discos se organizaban usando una Tabla de Particiones **MBR (Master Boot Record)**, este sistema está obsoleto y ha sido reemplazado en todos los sistemas de los últimos 15 años por la Tabla de Particiones GUID (GPT).

El antiguo sistema MBR presentaba serias limitaciones que GPT soluciona:

- **Límite de tamaño:** MBR no puede direccionar discos de más de 2 TiB.
- **Número de particiones:** MBR solo permite 4 particiones “primarias”. Para tener más, era necesario usar un truco complejo creando una partición “extendida” que a su vez contenía “particiones lógicas”. De hecho, este concepto de particiones primarias, extendidas y lógicas ya no se usa con GPT.
- **Fragilidad:** El MBR almacena toda la información de particionado en el primer sector del disco. Si ese sector se corrompe, se pierde toda la estructura del disco.

La tecnología GPT, que se usa en conjunto con el firmware *Unified Extensible Firmware Interface*, o **UEFI** para abreviar, ofrece ventajas cruciales:

- **Soporte para discos enormes:** Direcciona hasta Zettabytes, un tamaño virtualmente ilimitado.
- **Gran número de particiones:** Permite hasta 128 particiones estándar por defecto, sin necesidad de particiones extendidas o lógicas.
- **Robustez:** Utiliza la suma de verificación (CRC32) para detectar corrupción en la tabla de particiones y guarda una copia de seguridad de la misma al final del disco, permitiendo una recuperación sencilla en caso de error.
- **Identificadores únicos:** Usa GUIDs (Identificadores Únicos Globales) de 128 bits para identificar tanto el disco como cada una de las particiones, evitando conflictos.

Tipos de Partición en GPT Cuando creamos una partición en un disco GPT, ya no usamos códigos numéricos como 0x83 (Linux) o 0x82 (swap), como se hacía con el antiguo sistema MBR. En su lugar, a cada partición se le asigna un GUID de tipo de partición que describe su propósito. Las herramientas modernas como `fdisk` o `gdisk` manejan esto de forma transparente. Algunos de los tipos más comunes son:

- **Linux filesystem:** Para cualquier sistema de ficheros nativo de Linux (`ext4`, `Btrfs`, `XFS`, etc.). Es el tipo por defecto.
- **Linux swap:** Para el área de intercambio.
- **EFI System Partition (ESP):** Esta partición es fundamental en sistemas modernos. Es una pequeña partición (normalmente de 100-500 MB, formateada en `FAT32`) donde el firmware UEFI busca y carga los gestores de arranque de los sistemas operativos instalados. Es obligatoria para poder arrancar un SO en un sistema UEFI.

3.1 ¿Cuántas particiones necesito y de qué tipo?

La estructura de particiones depende de si el disco va a contener un sistema operativo arrancable o si se usará únicamente para almacenar datos. En los ordenadores modernos, el arranque no lo gestiona la antigua BIOS, sino un firmware más avanzado llamado UEFI (el mismo que comentamos al principio de la guía), que requiere una estructura de particionado específica.

A) Configuración para un Disco de Arranque (con Sistema Operativo) Para instalar un sistema operativo como Linux en un disco y que pueda arrancar en un equipo moderno, necesitas la siguiente configuración basada en GPT:

1. Partición del Sistema EFI (ESP) - (Obligatoria)

- Es una partición pequeña (entre 100 MB y 512 MB suele ser suficiente) formateada en **FAT32**.
- El firmware UEFI del ordenador buscará aquí los archivos del gestor de arranque (`.efi`) para poder iniciar el sistema operativo. Sin ella, el sistema no arrancará.

2. Partición Raíz (/) - (Obligatoria)

- Esta es la partición principal donde se instala el sistema operativo (los directorios `/bin`, `/etc`, `/lib`, `/usr`, etc.).
- Debe tener un tamaño suficiente para el SO y las aplicaciones que instales (30 GB es un mínimo seguro, pero 50-100 GB es más recomendable).
- Se formatea con un sistema de ficheros nativo de Linux, como **ext4**, que es el más común y robusto.

3. Espacio de Intercambio (Swap) - (Recomendado)

- Es un espacio que el sistema usa como memoria RAM virtual cuando esta se llena. Hoy en día tienes dos opciones modernas:
 - **Fichero Swap (Opción flexible y moderna):** En lugar de una partición, se crea un fichero (`/swapfile`) dentro de la partición raíz. Es la opción por defecto en muchas

distribuciones actuales (como Ubuntu) porque su tamaño se puede cambiar fácilmente sin tener que volver a particionar el disco.

- **Partición Swap (Opción tradicional):** Sigue siendo una opción válida y algunos la prefieren para hibernar el sistema (suspend-to-disk), ya que es ligeramente más sencilla de configurar para esa tarea.

4. **Partición Home (/home) - (Opcional pero muy recomendado)**

- Crear una partición separada para /home es una excelente práctica. Aquí se guardan todos los archivos y configuraciones personales de los usuarios.
- **Ventaja principal:** Si necesitas reinstalar o cambiar de distribución de Linux en el futuro, puedes formatear la partición raíz (/) sin tocar la partición /home, conservando así todos tus datos intactos.

B) Configuración para un Disco Secundario (solo para datos) Si el disco no va a arrancar un sistema operativo, la configuración es mucho más sencilla. Olvida por completo los conceptos de particiones primarias o lógicas. Con GPT, simplemente creas las particiones que necesites para organizar tus datos.


- Puedes crear **una única partición grande** que ocupe todo el disco o **varias particiones** si quieres separar, por ejemplo, “Trabajo” de “Multimedia”.
- El sistema de ficheros a elegir dependerá del uso:
 - **ext4:** Si solo lo vas a usar en sistemas Linux.
 - **NTFS:** Si necesitas compatibilidad total de lectura/escritura con Windows.
 - **exFAT:** Si necesitas compatibilidad con cualquier sistema de bajo coste además de cualquier SO (ideal para discos externos que contienen datos multimedia, por ejemplo).

3.2 ¿Qué directorios de primer nivel del estándar FHS deberían estar soportados por una partición independiente?

Si bien toda la estructura de directorios del FHS puede residir en una única partición (la partición raíz, etiquetada como /), es a menudo beneficioso, en ciertas situaciones, establecer particiones independientes para la información almacenada en directorios específicos. Esto puede mejorar la flexibilidad, la seguridad y la facilidad de mantenimiento del sistema.

A continuación, se describen algunos de los directorios de primer nivel del FHS que son comúnmente soportados por una partición independiente, junto con las razones para hacerlo:

Directorio	Descripción
/home	Este directorio contiene los directorios de inicio de todos los usuarios del sistema. Al asignar a /home su propia partición, puedes limitar el espacio de almacenamiento disponible para los usuarios, evitando que consuman todo el espacio de la partición raíz. Además, y crucialmente, si /home está en una partición separada, facilita las actualizaciones del sistema operativo . Puedes formatear y reinstalar la partición raíz sin afectar los datos de los usuarios, que permanecen intactos en su partición independiente.
/usr	Este directorio alberga la mayoría de los ejecutables binarios del sistema , bibliotecas, archivos de documentación y recursos compartidos. Históricamente, se separaba para mantener los binarios del sistema y la documentación en una partición de solo lectura. Aunque hoy en día es menos común tener /usr en una partición completamente separada, sigue siendo una consideración válida en entornos donde la seguridad y la inmutabilidad de los binarios del sistema son prioritarias.
/var	El directorio /var contiene datos variables , como archivos de registro (/var/log), directorios de spool (para correo electrónico e impresión), cachés de paquetes (/var/cache), y bases de datos. En sistemas donde Linux actúa como servidor o maneja grandes volúmenes de datos variables (por ejemplo, un servidor web o de base de datos), asignar a /var una partición independiente es una práctica estándar. Esto previene que el crecimiento de los archivos de registro o datos de aplicaciones llene la partición raíz , lo que podría comprometer la estabilidad del sistema.

 **Actividad 2.1. Partición de un dispositivo: “USB pen drive” o “memory stick”** En esta actividad, configuraremos un dispositivo USB de tipo “pen drive” o un archivo de disco simulado para crear particiones. Nos centraremos en el uso de **tablas de particiones GPT (GUID Partition Table)**, que han reemplazado a las MBR como el estándar moderno y ofrecen mayor flexibilidad y soporte para discos grandes.

Crearemos una tabla de particiones GPT con dos particiones primarias: 1. Una partición formateada con el sistema de archivos **ext4** (un sistema de archivos robusto y ampliamente utilizado en Linux). 2. Una segunda partición formateada con **vfat (FAT32)**, para asegurar la compatibilidad con una amplia gama de sistemas operativos, incluyendo Windows y macOS.

El tamaño de las particiones queda a vuestra elección, pero cada una debería tener un mínimo de 512 MB.

Utilizaremos la herramienta **fdisk** para la creación y manipulación de la tabla de particiones. Es importante recordar que esta herramienta es potente y su uso incorrecto puede llevar a la pérdida de datos.

Siempre asegúrate de identificar correctamente el dispositivo sobre el que vas a trabajar.

A continuación, se describen dos procedimientos: uno para particionar un dispositivo USB real y otro para particionar un dispositivo simulado mediante un archivo.

A) Partición de un dispositivo USB real Este procedimiento es para trabajar directamente con un “pen drive” o “memory stick”.

1. **Identifica el dispositivo USB:** Inserta el “pen drive” en un puerto USB. Luego, utiliza el comando `lsblk` o `fdisk -l` para identificar su ruta, que probablemente será algo como `/dev/sdX` (donde X es una letra, como b, c, d, etc.). **¡Es crucial que identifiques el dispositivo correcto para evitar formatear el disco equivocado!**

```
lsblk
# 0 bien
sudo fdisk -l
```

Busca el dispositivo por su tamaño o nombre. Para ello ejecuta:

```
ls /dev/disk/by-* -d
```

Por ejemplo, si tu USB es de 8GB y tiene un nombre parecido (MI_USB_8G), podría aparecer con el nombre: `/dev/disk/by-label/MI_USB_8G`

pero en realidad sería un enlace, por ejemplo, a `/dev/sdb`.

2. **Desmonta el dispositivo (si está montado):** Antes de particionar, cualquier partición del USB debe estar desmontada. Puedes verificar los puntos de montaje con `mount` o `lsblk`. Si encuentras alguna partición montada (e.g., `/dev/sdb1` montada en `/media/usuario/USB_DRIVE`), desmonta todas las particiones del USB.

```
sudo umount /dev/sdb1 # Reemplaza /dev/sdb1 con la partición de tu USB
# Repite si hay más particiones montadas del mismo USB (e.g., /dev/sdb2)
```

3. **Inicia fdisk para particionar:** Ahora, inicia `fdisk` con la ruta de tu dispositivo USB (por ejemplo, `/dev/sdb`).

```
sudo fdisk /dev/sdb # ¡Asegúrate de que /dev/sdb sea tu USB!
```

Dentro de `fdisk`:

- Antes de nada presiona `m` para familiarizarte con la ayuda, si no, nada de lo que viene a continuación tendrá sentido.
- Presiona `g` para crear una **nueva tabla de particiones GPT**.
- Presiona `n` para crear la primera partición.
 - Acepta el número de partición por defecto (1).
 - Acepta el primer sector por defecto.

- Para el último sector, especifica el tamaño deseado (por ejemplo, +1G para 1 Gigabyte).
- Presiona n nuevamente para crear la segunda partición.
 - Acepta el número de partición por defecto (2).
 - Acepta el primer sector por defecto.
 - Para el último sector, puedes dejarlo por defecto para que ocupe el resto del espacio, o especificar un tamaño (e.g., +1G).
- Presiona p para **ver la tabla de particiones** que has configurado.
- Presiona w para **escribir los cambios** en el disco y salir.

Nota: fdisk no asigna directamente los tipos de sistema de archivos (ext4, vfat), solo un tipo de partición genérico para Linux. El formato real se hará en el siguiente paso.

4. **Verifica las nuevas particiones:** Después de escribir los cambios, puedes usar lsblk de nuevo para ver las nuevas particiones creadas (e.g., /dev/sdb1, /dev/sdb2).

```
lsblk /dev/sdb
```

5. **Formatea las particiones:** Ahora, formatea cada partición con el sistema de archivos deseado. Reemplaza /dev/sdb1 y /dev/sdb2 con las rutas de tus particiones. Revisa el man de la orden mkfs.

```
sudo mkfs.ext4 -L "Datos_Linux" /dev/sdb1
sudo mkfs.vfat -n "COMPATIB" /dev/sdb2
```

- -L para ext4 asigna una etiqueta de volumen.
- -n para vfat asigna un nombre de volumen.

6. **Monta y prueba las particiones:** Puedes montar las particiones para verificar que funcionan correctamente.

```
sudo mkdir -p /mnt/usb_ext4
sudo mkdir -p /mnt/usb_vfat
sudo mount /dev/sdb1 /mnt/usb_ext4
sudo mount /dev/sdb2 /mnt/usb_vfat
# Puedes crear un archivo de prueba
sudo touch /mnt/usb_ext4/test_ext4.txt
sudo touch /mnt/usb_vfat/test_vfat.txt
ls /mnt/usb_ext4
ls /mnt/usb_vfat
# Desmonta las particiones cuando hayas terminado
sudo umount /mnt/usb_ext4
sudo umount /mnt/usb_vfat
```

Plantéate que, como en GNU/Linux todo son archivos, ¿qué pasaría si hacemos un cat al fichero de una de las particiones creadas? ¿Has visto algo familiar tras crear ficheros o carpetas en la

partición?

B) Partición de un archivo de disco simulado Este método permite practicar el particionamiento sin riesgo de afectar un disco físico real, usando un archivo como si fuera un dispositivo de almacenamiento.

1. **Crea un archivo de disco simulado:** Usa `dd` para crear un archivo de bloques que actuará como tu “disco virtual”. Por ejemplo, un archivo de 256MB. Revisa el manual de `dd`.

```
dd if=/dev/zero of=disk.img bs=1M count=256
```

Esto creará un archivo llamado `disk.img` de 256 MB lleno de ceros en bloques de 1 MB.

2. **Asocia el archivo a un dispositivo loop:** Revisa el manual de `losetup`, y úsalo para asociar `disk.img` a un dispositivo de bucle libre (`/dev/loopX`). Aunque el sistema intentará usar el primer dispositivo loop disponible, es conveniente que seas consciente de cuales de estos sistemas están ya ocupados.

```
sudo losetup -f --show disk.img
```

La salida te dirá qué dispositivo loop se ha asignado (por ejemplo, `/dev/loop0`). Anótalo, lo necesitarás para los siguientes pasos. Otra opción válida es montarlo en un punto fijo conocido, revisa el manual para ello.

3. **Particiona el archivo simulado con `fdisk`:** Ahora, utiliza `fdisk` en el dispositivo loop que se te asignó (por ejemplo, `/dev/loop0`).

```
sudo fdisk /dev/loop0
```

Dentro de `fdisk` (los pasos son idénticos a los del USB real):

- Presiona `g` para crear una **nueva tabla de particiones GPT**.
- Presiona `n` para crear la primera partición.
 - Acepta el número de partición por defecto (1).
 - Acepta el primer sector por defecto.
 - Para el último sector, especifica el tamaño deseado (por ejemplo, `+100M` para 100 Megabytes).
- Presiona `n` nuevamente para crear la segunda partición.
 - Acepta el número de partición por defecto (2).
 - Acepta el primer sector por defecto.
 - Para el último sector, puedes dejarlo por defecto para que ocupe el resto del espacio, o especificar un tamaño (e.g., `+100M`).
- Presiona `p` para **ver la tabla de particiones** que has configurado.
- Presiona `w` para **escribir los cambios** en el disco y salir.

4. **Actualiza la información de las particiones loop:** Después de crear las particiones, el kernel necesita ser informado sobre ellas. `partx` es útil para esto.

```
sudo partx -a /dev/loop0
```

Ahora, `lsblk` debería mostrar las particiones `loop0p1` y `loop0p2`.

```
lsblk /dev/loop0
```

5. Formatea las particiones simuladas: Formatea las nuevas particiones con `ext4` y `vfat`.

```
sudo mkfs.ext4 -L "Virtual_EXT4" /dev/loop0p1
```

```
sudo mkfs.vfat -n "VIRTUALVFAT" /dev/loop0p2
```

6. Monta y prueba las particiones simuladas: Puedes montar y probar las particiones virtuales para asegurarte de que funcionan.

```
sudo mkdir -p /mnt/virtual_ext4
```

```
sudo mkdir -p /mnt/virtual_vfat
```

```
sudo mount /dev/loop0p1 /mnt/virtual_ext4
```

```
sudo mount /dev/loop0p2 /mnt/virtual_vfat
```

```
# Crea archivos de prueba
```

```
sudo touch /mnt/virtual_ext4/prueba_ext4.txt
```

```
sudo touch /mnt/virtual_vfat/prueba_vfat.txt
```

```
ls /mnt/virtual_ext4
```

```
ls /mnt/virtual_vfat
```

7. Desmonta y “desasocia” el dispositivo loop: Una vez que hayas terminado, es crucial desmontar las particiones y luego desasociar el archivo del dispositivo `loop`.

```
sudo umount /mnt/virtual_ext4
```

```
sudo umount /mnt/virtual_vfat
```

```
sudo losetup -d /dev/loop0 # Reemplaza /dev/loop0 con tu dispositivo loop
```

```
# Puedes eliminar el archivo disk.img si ya no lo necesitas
```

```
rm disk.img
```

A partir de este momento ya disponemos de particiones sobre las que crear sistemas de archivos. Para ahondar más en el proceso de partición de discos sobre Linux puedes consultar el “Linux Partition HOWTO” (<http://www.tldp.org/HOWTO/Partition/>).


Aquí tienes otros enlaces más actualizados:

- **UEFI:** <https://wiki.debian.org/UEFI>
- **GPT:** How to Manipulate GPT Partitons
- **fdisk:** <https://linuxize.com/post/fdisk-command-in-linux/>

4. Asignación de un Sistema de Archivos a una partición (formateo lógico)

Una vez que disponemos de las particiones en nuestro dispositivo de almacenamiento secundario debemos proceder a asignar el sistema de archivos adecuado a cada una de las particiones. En Linux, aparte del SA específico para las particiones especialmente dedicadas a intercambio (swap), se utilizan normalmente tres sistemas de archivos: ext2, ext3 y ext4.

- **ext2.** Es el sistema de archivos de disco de alto rendimiento usado en Linux para discos duros, memorias flash y medios extraíbles. El *second extended file system* se diseñó como una extensión del *extended file system* (ext). ext2 ofrece el mejor rendimiento en términos de velocidad de transferencia de E/S y uso de CPU de entre todos los sistemas de archivos que soporta Linux.
- **ext3.** Es una versión de ext2 que incluye “registro por diario” (journaling). El journaling es un mecanismo por el cual un sistema informático puede implementar transacciones. Se basa en llevar un journal o registro de diario en el que se almacena la información necesaria para restablecer los datos afectados por la transacción en caso de que ésta falle. La aplicación del journaling en los sistemas de archivos modernos permite evitar la corrupción de las estructuras de datos que soportan la información del sistema de archivos: estructura de directorios, estructura de bloques libres de disco y estructuras que soportan los atributos de los archivos.
- **ext4.** Es el estándar de facto actual de las distribuciones Linux. Este SA tiene unas estructuras similares a las del ext3 pero, además, presenta las siguientes mejoras:
 - **Extensiones.** Las extensiones permiten describir un conjunto de bloques de disco contiguos, mejorando de esta forma el rendimiento de E/S al trabajar con archivos de gran tamaño y reduciendo la fragmentación de disco.
 - **Asignación retardada de espacio en disco (allocate-on-flush).** Esta técnica permite postergar en el tiempo la asignación de bloques de disco hasta el momento real en el que se va a realizar la escritura.

 **Actividad 2.2. Creación de sistemas de archivos** El objetivo es simplemente formatear lógicamente las particiones creadas con anterioridad de forma consistente con el tipo de SA que se estableció que iba a ser alojado. En la primera partición crearemos un SA de tipo ext3 y en la segunda un ext4.

La orden que permite establecer un SA de los reconocidos dentro del sistema Linux sobre una partición de disco es `mke2fs` (consulta el manual en línea para familiarizarte con sus opciones). El resultado de la ejecución de esta orden es el formateo lógico de la partición escogida utilizando el SA que se ha seleccionado.

Utiliza el manual en línea para conocer cómo ejecutar la orden de creación de SA. `mke2fs` es la orden genérica para creación de sistemas de archivos. Como requisito es necesario que establezcas dos etiquetas de volumen para los SAs: `LABEL_ext3` para la primera partición y `LABEL_ext4` para la segunda. Debería aparecer un listado en pantalla similar al siguiente.

```
mke2fs 1.47.0 (5-Feb-2023)
```

```
Descartando los bloques del dispositivo: hecho
```

Etiqueta del sistema de ficheros=LABEL_ext4
Tipo de S0: Linux
Tamaño del bloque=4096 (log=2)
Tamaño del fragmento=4096 (log=2)
Stride=0 bloques, anchura de stripe=0 bloques
2560 nodos-i, 2560 bloques
128 bloques (5.00%) reservados para el superusuario
Primer bloque de datos=0
Número máximo de bloques en el sistema de archivos=4194304
1 grupo de bloques
32768 bloques por grupo, 32768 fragmentos por grupo
2560 nodos-i por grupo

Reservando las tablas de grupo: hecho
Escribiendo las tablas de nodos-i: hecho
Creando el fichero de transacciones (1024 bloques): hecho
Escribiendo superbloques y la información contable del sistema de archivos: hecho

5. Ajuste de parámetros configurables de un SA y comprobación de errores

Una vez que tenemos disponibles nuestros nuevos sistemas de archivos podemos utilizar una orden, tune2fs, que nos permite ajustar determinados parámetros de los sistemas de archivos ext2/ext3/ext4. Puedes utilizar la opción -l para obtener un listado por pantalla que nos muestre la información relevante de un determinado SA. A continuación se muestra un ejemplo de listado sobre uno de nuestros “discos virtuales” con su sistema de archivos ya creado.

```
tune2fs 1.47.0 (5-Feb-2023)
Filesystem volume name: LABEL_ext4
Last mounted on: <not available>
Filesystem UUID: c3d20751-4ee4-4324-9e50-8ea2c2507f05
Filesystem magic number: 0xEF53
Filesystem revision #: 1 (dynamic)
Filesystem features: has_journal ext_attr resize_inode dir_index filetype extent 64bi
Filesystem flags: signed_directory_hash
Default mount options: user_xattr acl
Filesystem state: clean
Errors behavior: Continue
Filesystem OS type: Linux
Inode count: 2560
Block count: 2560
Reserved block count: 128
```

```

Overhead clusters: 1189
Free blocks: 1365
Free inodes: 2549
First block: 0
Block size: 4096
Fragment size: 4096
Group descriptor size: 64
Reserved GDT blocks: 1
Blocks per group: 32768
Fragments per group: 32768
Inodes per group: 2560
Inode blocks per group: 160
Flex block group size: 16
Filesystem created: Wed Jun 25 10:12:49 2025
Last mount time: n/a
Last write time: Wed Jun 25 10:12:49 2025
Mount count: 0
Maximum mount count: -1
Last checked: Wed Jun 25 10:12:49 2025
Check interval: 0 (<none>)
Lifetime writes: 17 kB
Reserved blocks uid: 0 (user root)
Reserved blocks gid: 0 (group root)
First inode: 11
Inode size: 256
Required extra isize: 32
Desired extra isize: 32
Journal inode: 8
Default directory hash: half_md4
Directory Hash Seed: d2c61877-11d1-4249-9bb8-e0d31ec6d1e2
Journal backup: inode blocks
Checksum type: crc32c
Checksum: 0x73ea3332

```

La tabla siguiente muestra algunas opciones interesantes de tune2fs.


Tabla 2. Algunas opciones de la orden tune2fs.

Opción	Descripción
-l <dispositivo>	Muestra por pantalla el contenido del superbloque del SA.

Opción	Descripción
<code>-c max-mount-counts <dispositivo></code>	Establece el número máximo de montajes que se puede llegar a realizar sin que se realice una comprobación de la consistencia del SA.
<code>-L label <dispositivo></code>	Poner una etiqueta al SA.

Con el tiempo, las estructuras de metainformación del SA pueden llegar a corromperse. Esta situación podría provocar degradación en el rendimiento del sistema e incluso situaciones de pérdida de información. Para solucionarlo Linux automatiza el proceso de comprobación de la consistencia del sistema de archivos en base al número de montajes que se han realizado. No obstante, pueden ocurrir situaciones en las que sea necesario que el administrador ejecute manualmente las comprobaciones y repare las posibles inconsistencias. Linux proporciona la herramienta `fsck` para llevar a cabo esta labor.

Hay que tener muy claro que esta herramienta actúa sobre las estructuras de metainformación del SA, no sobre el contenido de los archivos (datos de archivo). A continuación se muestran algunas de las situaciones de inconsistencia de metadatos del SA que se pueden dar: * Bloques que están asignados simultáneamente a varios archivos. * Bloques marcados como libres pero que están asignados a un archivo determinado. * Bloques marcados como asignados a un archivo determinado pero que realmente están libres. * Inconsistencia del número de enlaces a un determinado archivo. * i-nodos marcados como ocupados pero que realmente no están asociados a ningún archivo.

 **Actividad 2.3. Personalización de los metadatos del SA** Consultando el manual en línea para la orden `tune2fs` responde a las siguientes preguntas:

- ¿Cómo podrías conseguir que en el siguiente arranque del sistema se ejecutará automáticamente `e2fsck` sin que se haya alcanzado el máximo número de montajes?
- ¿Cómo podrías conseguir reservar para uso exclusivo de un usuario `username` un número de bloques del sistema de archivos?

6. Montaje y desmontaje de Sistemas de Archivos


Una vez que disponemos de nuestros nuevos sistemas de archivos ya solo nos queda ponerlos a disposición de los usuarios haciendo que puedan ser accesibles dentro de la jerarquía de directorios. Ya se ha comentado con anterioridad que la jerarquía de directorios proporciona un espacio de nombres para los archivos. Hasta este momento solamente disponemos de los sistemas de archivos creados en las particiones correspondientes. Para poder crear archivos y directorios en estos sistemas de archivos es necesario hacerlos visibles en el espacio de nombres. Para ello se utiliza la orden `mount`.

El montaje de un sistema de archivos consiste en añadir una nueva rama al espacio de nombres actualmente en uso, de modo que, una vez completada la orden, toda la información del sistema de archivos

montado resulte accesible dentro de dicho espacio de nombres. Así mismo, se podrán crear nuevos archivos y directorios en esta nueva rama del espacio de nombres. Para montar un sistema de archivos es necesario solamente indicar en qué directorio se montará (aquí es donde crecerá la nueva rama del espacio de nombres), y cuál es el nombre del archivo especial de dispositivo que representa la partición en donde reside el sistema de archivos. El directorio que se utiliza como punto de partida para el sistema de archivos montado se denomina punto de montaje.

Cuando ya no es necesario acceder a la información del sistema de archivos montado se puede llevar a cabo una operación de desmontaje del SA. Esta operación provoca que el sistema de archivos deje de estar disponible en el espacio de nombres y además permite realizar todas las actualizaciones en disco de los metadatos del SA de forma que quede en estado consistente.

NOTA: No es posible desmontar un SA si está siendo utilizado (busy).

 **Actividad 2.4. Montaje de sistemas de archivos** Utiliza el manual en línea para descubrir la forma de montar nuestros SAs de manera que cumplas los siguientes requisitos: * El SA etiquetado como LABEL_ext3 debe estar montado en el directorio /mnt/SA_ext3 y en modo de solo lectura. * El SA etiquetado como LABEL_ext4 debe estar montado en el directorio /mnt/SA_ext4 y debe tener sincronizadas sus operaciones de E/S de modificación de directorios.

Como vimos en la sesión anterior, /etc/fstab es el archivo de configuración que contiene la información sobre todos los sistemas de archivos que se pueden montar y de las zonas de intercambio a activar. El formato del archivo se describe a continuación:

```
# <file system> <mount point> <type> <options> <dump> <pass>
```

- <file system>, es el número que identifica el archivo especial de bloques .
- <mount point>, es el directorio que actúa como punto de montaje.
- <type>, tipo de sistema de archivos (ext2, ext3, ext4, vfat, iso9660, swap, nfs, etc.)
- <options>, son las opciones que se utilizarán en el proceso de montaje. Se especifican como una lista separada por comas y sin espacios.
- <dump>, normalmente no se usa, pero si su valor es distinto de 0 indica la frecuencia con la que se realizará una copia de seguridad del SA.
- <pass> , durante el arranque del sistema este campo especifica el orden en el que la orden fsck realizará las comprobaciones sobre los SAs.

A continuación se muestran las posibles opciones que pueden especificarse en el campo <options>: * rw. Lectura-escritura * ro. Sólo lectura * suid/nosuid. Permitido el acceso en modo SUID, o no permitido * auto/noauto. Montar automáticamente o no montar automáticamente (ni ejecutando mount -a) * exec/noexec. Permitir la ejecución de ficheros, o no permitir * usrquota, grpquota. Cuotas de usuario y de grupo * defaults = rw, suid, dev, exec, auto, nouser, async * user, users, owner. Permitir a los usuarios montar un sistema de archivos * uid=500, gid=100. Propietario y grupo propietario de los archivos del SA. * umask. Máscara para aplicar los permisos a los archivos.

 **Actividad 2.5. Automontaje de Sistemas de Archivos** Escribe las dos líneas necesarias en el archivo `/etc/fstab` para que se monten automáticamente nuestros dos SA en el arranque del sistema con los mismos requisitos que se han pedido en la **Actividad 2.4**.

7.2 Gestores de Paquetes

La instalación y actualización de software en sistemas Linux se realiza de forma eficiente mediante los gestores de paquetes. Estos gestores automatizan el proceso de encontrar, instalar, configurar y actualizar el software, así como la gestión de sus dependencias.

En el ecosistema GNU/Linux, los paquetes de software se identifican por su formato, que suele estar asociado a la distribución. Dado que estamos trabajando con Ubuntu, nos centraremos en el formato **.deb (Debian Package)**, que es el estándar para Debian, Ubuntu y sus derivados. Los paquetes **.deb** contienen los archivos binarios, bibliotecas, documentación y scripts necesarios para instalar una aplicación o componente del sistema.

Es posible que aún encuentres referencias a paquetes RPM (Redhat Package Manager), utilizados principalmente en distribuciones basadas en Red Hat como Fedora o CentOS. Si bien es posible convertir paquetes entre formatos con herramientas como `alien`, en un entorno Ubuntu, siempre es preferible utilizar paquetes **.deb** y los gestores diseñados para ellos para garantizar la estabilidad y compatibilidad.

Existen gestores de paquetes que operan a diferentes niveles:

- **Gestores de bajo nivel:** Se encargan de la instalación, eliminación y gestión de paquetes individuales. No se resuelven las dependencias automáticamente. El principal gestor de bajo nivel para paquetes **.deb** es `dpkg`.
- **Gestores de alto nivel:** Utilizan los gestores de bajo nivel como base, pero añaden funcionalidades avanzadas como la resolución automática de dependencias, la búsqueda en repositorios remotos y la gestión de actualizaciones complejas. Para sistemas basados en Debian/Ubuntu, los gestores de alto nivel por excelencia son **APT (Advanced Packaging Tool)** y `aptitude`.

La siguiente tabla resume los gestores de paquetes más relevantes para el formato **.deb**:

	Formato .deb (Debian, Ubuntu)
Modo Línea de Órdenes	<code>dpkg</code> ; <code>apt-get</code> ; <code>apt</code> ; <code>aptitude</code>
Modo Gráfico	<code>synaptic</code> ; Software Center

7.2.1 Gestión de paquetes en Debian y derivados

`dpkg` es el gestor de paquetes de **bajo nivel** en sistemas basados en Debian y Ubuntu. Permite instalar, eliminar y gestionar paquetes **.deb** individuales. Sin embargo, `dpkg` no resuelve dependencias; si un paquete requiere de otros para funcionar, `dpkg` lo notificará y deberás instalar esas dependencias manualmente.

Ejemplos de uso de dpkg:

- **Instalar un paquete .deb local:**

```
sudo dpkg -i /ruta/al/paquete.deb
```

- **Eliminar un paquete (sin sus archivos de configuración):**

```
sudo dpkg -r nombre_del_paquete
```

- **Eliminar un paquete (con sus archivos de configuración):**

```
sudo dpkg -P nombre_del_paquete
```

- **Listar todos los paquetes instalados:**

```
dpkg -l
```

- **Mostrar información sobre un paquete instalado:**

```
dpkg -s nombre_del_paquete
```

- **Listar los archivos que pertenecen a un paquete instalado:**

```
dpkg -L nombre_del_paquete
```

APT (Advanced Packaging Tool) es el conjunto de herramientas de **alto nivel** más utilizado en Debian y Ubuntu. Proporciona una interfaz mucho más amigable y potente para la gestión de paquetes, ya que puede resolver dependencias, buscar en múltiples repositorios remotos y gestionar actualizaciones completas del sistema.

Históricamente, la interfaz de línea de comandos principal de APT era `apt-get`. Si bien `apt-get` sigue siendo funcional y se utiliza ampliamente, especialmente en scripts o en entornos minimalistas y contenedores donde la compatibilidad es clave, la herramienta `apt` (introducida en Ubuntu 14.04 y Debian 8) ha sido diseñada para ser la interfaz de usuario preferida para el día a día. `apt` combina las funcionalidades más comunes de `apt-get`, `apt-cache` y `apt-cdrom` en una sola herramienta, ofreciendo una salida más estética y progresiva.

Comandos importantes de APT (usando apt como interfaz principal):

Comando (apt)	Comando (apt-get)	Descripción
<code>sudo apt update</code>	<code>sudo apt-get update</code>	Actualiza la lista de paquetes disponibles en los repositorios. Es crucial ejecutarlo antes de instalar o actualizar software.
<code>sudo apt upgrade</code>	<code>sudo apt-get upgrade</code>	Actualiza todos los paquetes instalados a sus últimas versiones disponibles. No elimina paquetes ni instala nuevos paquetes de forma agresiva.

Comando (apt)	Comando (apt-get)	Descripción
sudo apt full-upgrade	sudo apt-get dist-upgrade	Realiza una actualización completa del sistema , incluyendo la eliminación de paquetes obsoletos y la instalación de nuevas dependencias si es necesario (maneja cambios importantes en las dependencias).
sudo apt install <paquete>	sudo apt-get install <paquete>	Instala un nuevo paquete y todas sus dependencias.
sudo apt remove <paquete>	sudo apt-get remove <paquete>	Elimina un paquete , pero conserva sus archivos de configuración.
sudo apt purge <paquete>	sudo apt-get purge <paquete>	Elimina un paquete y sus archivos de configuración. Útil para una limpieza completa.
sudo apt autoremove	sudo apt-get autoremove	Elimina paquetes que fueron instalados automáticamente como dependencias y que ya no son necesarios para ningún otro paquete.
sudo apt clean	sudo apt-get clean	Borra los archivos .deb descargados de la caché local del paquete (/var/cache/apt/archives/). Útil para liberar espacio.
sudo apt autoclean	sudo apt-get autoclean	Borra solo los archivos .deb que ya no se pueden descargar (porque sus versiones son más antiguas) de la caché.
apt search <término>	apt-cache search <término>	Busca paquetes que contengan el término especificado en su nombre o descripción.
apt show <paquete>	apt-cache show <paquete>	Muestra información detallada sobre un paquete específico (versión, dependencias, descripción, etc.).

Repositorios de paquetes:

Los repositorios APT son servidores que almacenan colecciones de paquetes de software. La lista de repositorios configurados en tu sistema se encuentra usualmente en el archivo `/etc/apt/sources.list`, aunque recientemente se ha modificado a la ruta `/etc/apt/sources.list.d/ubuntu.sources` por coherencia, ya que el resto de archivos adicionales, necesarios para la configuración de paquetes del

sistema, se encuentran en el directorio `/etc/apt/sources.list.d/`.

En cualquier caso, es fundamental que la lista de repositorios esté actualizada (`sudo apt update`) antes de intentar instalar o actualizar software.

 **Actividad 2.7. Trabajo con el gestor de paquetes APT** En esta actividad, exploraremos el gestor de paquetes APT, que es fundamental en tu entorno Ubuntu.

1. **Explora las opciones de APT:** Familiarízate con las opciones disponibles de `apt` ejecutando los siguientes comandos:

```
apt --help
man apt
```

También puedes consultar la ayuda para comandos específicos, como:

```
apt help install
apt help search
```

2. **Lista paquetes instalados y disponibles:** Utiliza `apt` para listar los paquetes actualmente instalados en tu sistema y aquellos disponibles en los repositorios.

```
apt list --installed
apt list --all-versions # Muestra todas las versiones disponibles
apt list | grep <nombre_parcial_paquete> # Filtra la lista
```

3. **Realiza una búsqueda de paquetes:** Busca un paquete específico utilizando `apt search`. Por ejemplo, busca información sobre el paquete `sysstat`.

```
apt search sysstat
```

4. **Muestra información detallada de un paquete:** Una vez que hayas encontrado un paquete, usa `apt show` para obtener información más detallada sobre él.

```
apt show sysstat
```

5. **Instala y elimina un paquete:** Instala el paquete `sysstat` si no lo tienes. Luego, elimínalo y vuelve a instalarlo. Observa los mensajes de salida.

```
# Instalar sysstat
sudo apt install sysstat
```

```
# Eliminar sysstat (solo el paquete, manteniendo la configuración)
sudo apt remove sysstat
```

```
# Instalar sysstat de nuevo
sudo apt install sysstat
```

```
# Eliminar sysstat y sus archivos de configuración
sudo apt purge sysstat
```

Presta atención a la diferencia entre `apt remove` y `apt purge`.

6. **Limpia la caché de paquetes:** Después de varias instalaciones y eliminaciones, es una buena práctica limpiar la caché de paquetes descargados para liberar espacio.

```
sudo apt clean
sudo apt autoremove
```

Aunque `apt` es la herramienta de línea de comandos preferida para la gestión de paquetes en Ubuntu debido a su interfaz mejorada y unificada, es importante recordar que a menudo **`apt` realiza operaciones de alto nivel que requieren cierta inicialización o la configuración de un entorno más completo**. Por esta razón, comandos como `apt-get` y `apt-cache` siguen siendo ampliamente utilizados y son de vital importancia, especialmente en **sistemas más básicos o en entornos donde la compatibilidad y la simplicidad son prioritarias, como en los contenedores Docker, Podman o LXC**.

En estos entornos minimalistas, donde el espacio y los recursos son críticos y la salida amigable de `apt` es menos relevante, **`apt-get`** proporciona una forma directa y sin florituras de interactuar con el sistema de paquetes, siendo ideal para scripts automatizados o Dockerfiles. Mientras que `apt-get` maneja las acciones principales de instalación, eliminación y actualización, **`apt-cache`** se centra en la búsqueda y consulta de información sobre los paquetes disponibles en los repositorios, actuando como una interfaz para la caché de información de paquetes de APT. Revisa la documentación de ambas herramientas y repite los ejemplos anteriores con ellas hasta que te familiarices con ambas formas de instalar/desinstalar y buscar paquetes.