Capítulo 7: Administración de dispositivos de almacenamiento

Introducción

Los dispositivos de almacenamiento son fundamentales para la instalación y la correcta operación del sistema operativo. Como ya mencionamos anteriormente, en un sistema Linux hay solamente una estructura de directorios sin importar de cuántos dispositivos de almacenamiento físico dispongamos. Pero para poder utilizarlos, es necesario que primero asociemos a cada uno con un directorio del sistema. Este directorio recibe el nombre de **punto de montaje**.

En este capítulo aprenderemos cómo montar un dispositivo de almacenamiento, tanto temporariamente como de manera permanente. Además, a pesar de que hoy en día el costo de los discos y SSDs es relativamente bajo, siempre es bueno ser cautos en el uso del espacio disponible. Por eso también explicaremos cómo comprimir archivos y directorios desde la línea de comandos. Por supuesto este no es el único propósito de la compresión de archivos, pero vale la pena mencionarlo.

Los sistemas de archivos

Uno de los pasos previos más importantes al instalar Linux y en preparación para su uso consiste en crear particiones y sistemas de archivos. Una *partición* es una división lógica que se hace a un dispositivo de almacenamiento con el propósito de definir un espacio determinado para un uso en particular, mientras que un *sistema de archivos* puede definirse (en este contexto) como la forma en que se estructurarán los contenidos (archivos y directorios) de dicha partición.

En cualquier sistema Linux el espacio utilizado por ciertos directorios (**/home** y **/var**, por nombrar dos ejemplos) puede llegar a aumentar o variar constantemente. El crear una partición dedicada para cada uno de ellos presenta dos ventajas al menos:

1. Limitar el espacio que pueden ocupar sus contenidos (de manera que su crecimiento no perjudique el de otros directorios)

2. Impedir que una partición llena afecte críticamente el funcionamiento de todo el sistema.

En cuanto a los sistemas de archivos, los más utilizados hoy en día son **ext4** y **xfs**. Como dijimos anteriormente, un sistema de archivos determina la forma en que se estructura el contenido en el espacio de almacenamiento. Veamos algunas características de ambos sistemas de archivos:

ext4 (tomado de la documentación oficial):

- Tamaño máximo de sistema de archivos (generalmente se extiende a toda la partición): 1 EiB = 1024 PB (1 PiB = 1024 TiB, 1 TiB = 1024 GiB)
- Tamaño máximo de archivos: 16 TiB
- Cantidad máxima de subdirectorios: 64.000
- Journaling: permite la recuperación de archivos que estaban siendo escritos en el momento de un fallo del sistema. Si bien esta característica no elimina totalmente las posibilidades de encontrar archivos corruptos, sí contribuye a la confiabilidad del sistema de archivos y a la rapidez del chequeo de los mismos.
- Permite aumentar o disminuir el tamaño del sistema de archivos.

xfs:

- Tamaño máximo de sistema de archivos (generalmente se extiende a toda la partición): 8 EiB en sistemas operativos de 64 bits y de 16 TiB en 32 bits.
- Tamaño máximo de archivos: 9 EiB
- Journaling
- Es posible aumentar el tamaño del sistema de archivos pero no disminuirlo

---

*Para ver más detalles sobre **xfs** podemos consultar el sitio oficial del proyecto.*

Crear particiones y sistemas de archivos

En Linux, la herramienta tradicional para administrar particiones basadas en MBR (presente por defecto en computadoras fabricadas hasta 2009 aproximadamente) es `fdisk`, mientras que desde 2010 en adelante (particiones basadas en GPT) se comenzó a utilizar también otra utilidad llamada `gdisk`

---

*Es importante aclarar que todas las operaciones relacionadas con la creación de particiones, sistemas de archivos, y montaje de dispositivos deben ser realizadas por **root** o con un usuario con permisos de superusuario.*

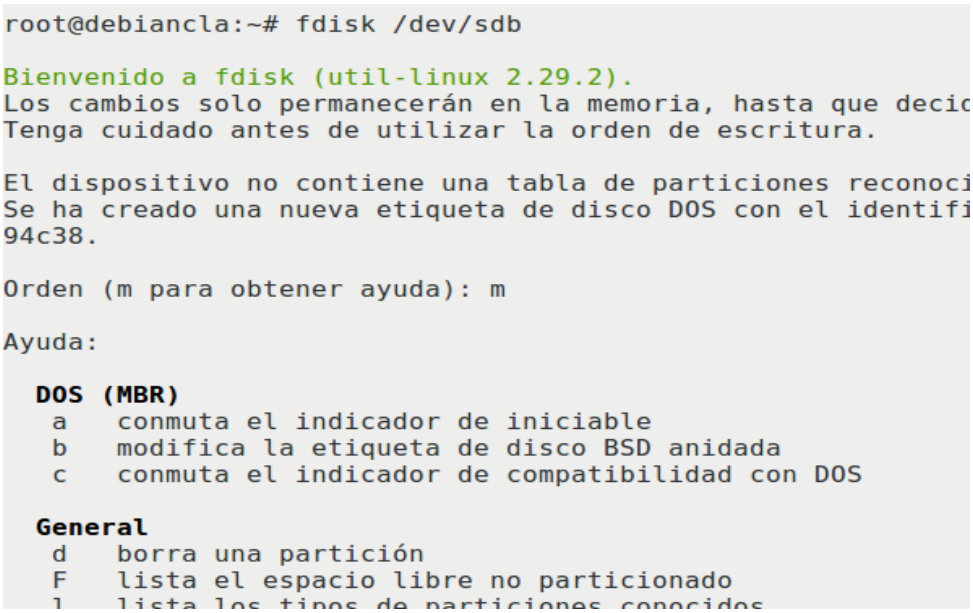
Cabe destacar que el estándar GPT permite crear hasta 128 particiones primarias en un mismo disco, y permite alcanzar tamaños de particiones no soportados por MBR

(el cual soporta hasta 2 TB y hasta 4 particiones primarias – si se necesitan más particiones se puede reemplazar una de las primarias por una extendida que pueda dividirse en múltiples particiones lógicas).

Para crear una partición con `fdisk`, utilizamos el comando del mismo nombre seguido por el identificador de dispositivo en cuestión. Por ejemplo, para **/dev/sdb**:

```
fdisk /dev/sdb
```

En la pantalla siguiente podemos presionar la tecla **m** para ver el menú de ayuda como se muestra en la imagen (por limitaciones de espacio no se alcanzan a apreciar todas las opciones disponibles):



Las operaciones más usuales que podemos llevar a cabo a partir de este momento son las siguientes:

- **m**: mostrar el menú de ayuda
- **p**: mostrar la tabla de particiones del dispositivo
- **l**: mostrar los tipos de particiones soportadas por fdisk
- **n**: agregar una nueva partición
- **d**: borrar una partición existente
- **w**: guardar los cambios en la tabla de particiones y salir
- **q**: salir sin guardar cambios

Algunos puntos importantes para destacar:

- Al crear una nueva partición, tendremos que confirmar si deseamos que sea una partición primaria o una extendida. Si no hemos creado ninguna partición todavía, deberíamos elegir primaria. Para el comienzo de la partición es conveniente aceptar el valor ofrecido por defecto por `fdisk`presionando Enter. Para la finalización de la misma es conveniente especificar el tamaño deseado de la misma utilizando el signo **+** seguido de un número y de la nomenclatura **M**, o **G** (por **MB** y **GB**, respectivamente).
- La primera partición de **/dev/sdb** recibirá el identificador **/dev/sdb1**, la segunda **/dev/sdb2**, y así sucesivamente. Por defecto, al crear una nueva partición, se le asignará el tipo **Linux (83)**. Si deseamos cambiarlo, podemos listar los tipos disponibles con **L**, elegir el adecuado, y luego asignarlo con la opción **t**.
- Si en el dispositivo seleccionado existe más de una partición y deseamos borrar una de ellas, tendremos que especificar el número de la misma (a partir del orden en el que aparecen al mostrar la tabla de particiones).
- Antes de confirmar cualquier cambio realizado, es importante revisar la tabla de particiones, ya que una vez que se guarden los cambios y se salga del menú de `fdisk`es probable que los mismos sean irreversibles (lo cual es particularmente crítico si hemos modificado una partición que contenía datos).

Por ejemplo:

```
Orden (m para obtener ayuda): n
Tipo de partición
   p   primaria (0 primaria(s), 0 extendida(s), 4 libre(s))
   e   extendida (contenedor para particiones lógicas)
Seleccionar (valor predeterminado p): p
Número de partición (1-4, valor predeterminado 1):
Primer sector (2048-16777215, valor predeterminado 2048):
Último sector, +sectores o +tamaño{K,M,G,T,P} (2048-16777215, valor predeterminado 16777215): +2,5G

Crea una nueva partición 1 de tipo 'Linux' y de tamaño 2,5 GiB.

Orden (m para obtener ayuda): w
Se ha modificado la tabla de particiones.
Llamando a ioctl() para volver a leer la tabla de particiones.
Se están sincronizando los discos.

root@debiancla:~# fdisk -l /dev/sdb
Disco /dev/sdb: 8 GiB, 8589934592 bytes, 16777216 sectores
Unidades: sectores de 1 * 512 = 512 bytes
Tamaño de sector (lógico/físico): 512 bytes / 512 bytes
Tamaño de E/S (mínimo/óptimo): 512 bytes / 512 bytes
Tipo de etiqueta de disco: dos
Identificador del disco: 0xdc7ed9a2

Disposit.  Inicio Comienzo   Final Sectores Tamaño Id Tipo
/dev/sdb1             2048 5220351  5218304   2,5G 83 Linux
root@debiancla:~#
```

En la imagen anterior vemos cómo crear una partición de 2,5 GB que recibió la identificación **/dev/sdb1**. Luego de guardar el cambio volvimos a la línea de

comandos y con `fdisk -l /dev/sdb` podemos ver la tabla de particiones actual. **Importante:** Utilizar un punto en vez de una coma al especificar el tamaño de las particiones si estamos utilizando un sistema cuyo separador de decimales es el primero.

Una vez que disponemos de una nueva partición, es hora de crear un sistema de archivos sobre la misma a fin de poder utilizarla. La utilidad por excelencia para realizar esta tarea es `mkfs` (entre sus variantes podemos destacar `mkfs.ext4` y `mkfs.xfs` para sistemas de archivos ext4 y mkfs, respectivamente).

Entre las muchas opciones de las que disponemos, `-L` en particular nos será de gran utilidad para asignarle una etiqueta al sistema de archivos, mediante la cual podremos identificarlo más fácilmente a la hora de montarlo y disponerlo finalmente para su uso.

Para crear un sistema de archivos del tipo **ext4** en **/dev/sdb1** y asignarle la etiqueta **ALUMNO** (el límite es de 16 caracteres) utilizaremos el siguiente comando:

```
mkfs.ext4 -L ALUMNO /dev/sdb1
```

El uso de `gdisk` es prácticamente idéntico a `fdisk`, por lo que no entraremos en detalles sobre el primero en este capítulo.

Montar particiones

Para montar dispositivos de almacenamiento luego de que hayamos creado un sistema de archivos contamos con dos alternativas:

1. De *manera permanente* implica agregar al dispositivo (junto con las opciones de montaje correspondientes) en el archivo **/etc/fstab**. Esto hará que el dispositivo esté disponible cuando se inicia el sistema y durante su uso, salvo que se lo desmonte explícitamente.

2. *Bajo demanda* significa montar el dispositivo cuando se lo necesite, luego de lo cual se prescinde de su uso y se lo desmonta.

Veamos a continuación ambas opciones.

El archivo /etc/fstab

El archivo **/etc/fstab** contiene información sobre los sistemas de archivos que deben montarse al inicio del sistema. Cada línea en el archivo posee los siguientes 6 campos:

1. El primer campo representa el dispositivo de almacenamiento (local o remoto) que debe montarse. Puede especificarse mediante el identificador de dispositivo (por ejemplo, **/dev/sdb1**, la primera partición de **/dev/sdb**) o mediante la etiqueta asignada al momento de crear el sistema de archivos (**LABEL=ALUMNO**

como asignamos anteriormente), o mediante el **UUID** del sistema de archivos (lo cual puede averiguarse mediante el comando `blkid`)

---

*El **UUID** (**U**niversally **U**nique **ID**entifier) puede utilizarse para identificar un dispositivo de almacenamiento, independientemente de su nombre o punto de montaje. Por esto puede utilizarse con tranquilidad para representar el dispositivo en el archivo **/etc/fstab**.*

---

2. El segundo campo muestra el punto de montaje en el que se anexará el sistema de archivos.
3. El tercer campo especifica el tipo de sistema de archivos (ext4, xfs, etc.)
4. El cuarto campo indica las opciones de montaje (separadas por comas si hay más que una). En el *man page* del comando `mount`podemos encontrar una descripción detallada de cada una.
5. El quinto campo es hoy una opción en desuso.
6. El sexto campo indica si el sistema de archivos debe chequearse al iniciar el sistema y en qué orden: para **/** (la raíz del árbol de directorios) siempre debería ser **1**, mientras que para el resto siempre **2**. Si este campo no existe o es **0**, significa que no se debe chequear el sistema de archivos.

Dicho esto, podemos montar **/dev/sdb1** en **/mnt/pruebas** (este directorio debe existir) mediante cualquiera de las tres maneras siguientes, agregando la línea correspondiente a **/etc/fstab**:

- Especificando el identificador de dispositivo:

  /dev/sdb1 /mnt/pruebas ext4 defaults 0 2

- Mediante la etiqueta asignada al sistema de archivos:

  LABEL=ALUMNO /mnt/pruebas ext4 defaults 0 2

- Mediante el **UUID**. Averigüémoslo primero con `blkid`

```
root@debiancla:~# mkfs.ext4 /dev/sdb1
mke2fs 1.43.4 (31-Jan-2017)
Se está creando un sistema de ficheros con 652288 bloques de 4k y 163200 nodos-i
UUID del sistema de ficheros: 93a0999f-23a9-4b96-9483-680956fe17ad
Respaldo del superbloque guardado en los bloques:
        32768, 98304, 163840, 229376, 294912

Reservando las tablas de grupo: hecho
Escribiendo las tablas de nodos-i: hecho
Creando el fichero de transacciones (16384 bloques): hecho
Escribiendo superbloques y la información contable del sistema de ficheros: hecho

root@debiancla:~# blkid /dev/sdb1
/dev/sdb1: UUID="93a0999f-23a9-4b96-9483-680956fe17ad" TYPE="ext4" PARTUUID="dc7ed9a2-01"
root@debiancla:~#
```

`blkid /dev/sdb1`

Luego agreguemos la siguiente línea en el archivo:

`UUID=93a0999f-23a9-4b96-9483-680956fe17ad /mnt/pruebas ext4 defaults 0 2`

Luego de guardar los cambios en el archivo **/etc/fstab**, los cambios que acabamos de introducir entrarán en efecto en el próximo reinicio, aunque también es posible hacer que se apliquen de manera inmediata con el comando `mount -a` (de esta manera forzamos el montaje de todos los dispositivos listados en el archivo).

Montar dispositivos de almacenamiento bajo demanda

Si lo que deseamos es montar un dispositivo bajo demanda (de uso por tiempo limitado), utilizaremos el comando `mount` con la opción `-t` para indicar el tipo de sistema de archivos a utilizar seguido del identificador del dispositivo, **LABEL**, o **UUID**, y el punto de montaje.

Por ejemplo, si quisiéramos montar el sistema de archivos con **LABEL=ALUMNO** en /mnt/pruebas solamente de manera temporaria, podemos utilizar el siguiente comando:

`mount -t ext4 LABEL=ALUMNO /mnt/pruebas`

Para desmontar un sistema de archivos emplearemos el comando umount seguido por el punto de montaje:

`umount /mnt/pruebas`

Importante: El proceso anterior fallará si estamos situados dentro del punto de montaje. Para salir del mismo primero, podemos utilizar `cd` (a fin de volver a nuestro directorio personal) y luego ejecutar el comando anterior.

---

*Una vez montado **/dev/sdb1** en **/mnt/pruebas**, podemos utilizar este último directorio como lo deseemos (crear archivos y directorios, etc).*

Cuando conectamos un nuevo dispositivo al equipo (pensemos en una unidad USB por nombrar un ejemplo), podemos identificar el nombre del dispositivo al observar la salida de

```
dmesg | tail
```

A continuación, podremos proceder como explicamos anteriormente para montar el dispositivo, con la excepción de que si el mismo ya había estado en uso no será necesario crear nuevamente el sistema de archivos.

Chequear y reparar sistemas de archivos defectuosos

Si el sistema se apaga abruptamente (por ejemplo, durante un corte de energía, y no se cuenta con UPSs), pueden ocasionarse errores en un sistema de archivos si se están realizando operaciones de lectura y escritura sobre el mismo. Para intentar solucionarlos, debemos desmontar el sistema de archivos (si el mismo se encuentra montado) o no intentar hacerlo, y utilizar el comando `fsck`

La regla de oro para el uso de `fsck`es la siguiente: Nunca debemos intentar correr fsck sobre un dispositivo que se encuentre montado a fin de minimizar el riesgo de pérdida o corrupción de archivos.

Supongamos que deseamos chequear el sistema de archivos en **/dev/sdb1**. Luego de desmontarlo con `umount /dev/sdb1`, ejecutaremos el siguiente comando:

```
fsck /dev/sdb1
```

Si se encontraran errores, podemos presionar la letra **y** para confirmar la reparación o **n** para omitirla. Alternativamente podríamos cancelar la operación con Ctrl + c y luego ejecutar `fsck`con la opción `-y` (para confirmar las reparaciones automáticamente). Por otro lado, podemos simular la reparación (a fin de ver qué es lo que haría) en vez de ejecutarla, podemos utilizar la opción `-n` de esta manera:

```
fsck -n /dev/sdb1
```

Es importante tener en cuenta las diferencias de uso entre `mkfs`y `fsck`para evitar usar un comando cuando en realidad necesitamos utilizar el otro. Las consecuencias pueden ser graves en términos de pérdida de información valiosa.

Archivado y compresión

En Linux, el archivado y compresión se realiza utilizando el comando `tar` (archivado) en conjunto con `gzip`, `bzip2` o `xz` (compresión). La posibilidad de disponer de un solo archivo comprimido (a menudo llamado *tarball*) que contenga muchos otros (incluyendo directorios) nos permite distribuirlo fácilmente mediante correo electrónico o transferirlo a través de una red, con la ventaja adicional de que su tamaño es menor que la suma de sus contenidos.

A modo informativo, las nuevas versiones del kernel se distribuyen desde kernel.org comprimidas con `xz`

Si bien es perfectamente posible archivar un grupo de archivos y directorios SIN comprimirlos, esta opción rara vez se utiliza, ya que siempre es preferible disponer de un archivo final de menor tamaño. Por ese motivo, nos enfocaremos exclusivamente en ambas operaciones (archivado y compresión).

Antes de comenzar, necesitaremos crear unos archivos de prueba y un subdirectorio en el directorio actual:

```
echo "Yo soy el archivo1" > archivo1.txt
echo "Yo soy el archivo2 y peso más que archivo1" > archivo2.txt
echo "Yo soy el archivo3 y penso más que archivo1 y que archivo2" > archi
vo3.txt
mkdir resultados
```

A continuación, veamos cómo comprimir este grupo de archivos con cada una de las herramientas que mencionamos hace instantes:

- Para comprimir y empaquetar **archivo1.txt**, **archivo2.txt**, y **archivo3.txt** con `gzip` y `tar` en un archivo de nombre **tarcongzip.tar.gz** dentro del subdirectorio **resultados** debemos hacer lo siguiente:

  tar -czvf resultados/tarcongzip.tar.gz archivo1.txt archivo2.txt archivo3.txt

- Con `tar` y `bzip2`

  tar -cjvf resultados/tarconbzip2.tar.bz2 archivo1.txt archivo2.txt archivo3.txt

- Finalmente, con `tar` y `xz`:

  tar -cJvf resultados/tarconxz.tar.xz archivo1.txt archivo2.txt archivo3.txt

Como observamos en los ejemplos, los comandos en cada caso son prácticamente idénticos. La única diferencia reside en el uso de las opciones `-z`, `-j`, y `-J` que representan la compresión con `gzip`, `bzip2`, y `xz` respectivamente.

Antes de descomprimir, crearemos tres directorios auxiliares dentro de resultados:

```
mkdir resultados/1
mkdir resultados/2
mkdir resultados/3
```

Y luego descomprimiremos los contenidos en **tarcongzip.tar.gz**, **tarconbzip2.tar.bz2**, y **tarconxz.tar.xz** en **resultados/1**, **resultados/2**, y

resultados/3. La opción `-C` (mayúscula) nos permite indicar un directorio destino para los archivos descomprimidos antes de realizar la operación propiamente dicha:

```
tar -xzvf resultados/tarcongzip.tar.gz -C resultados/1
tar -xjvf resultados/tarconbzip2.tar.bz2 -C resultados/2
tar -xJvf resultados/tarconxz.tar.xz -C resultados/3
```

Resaltemos algunos puntos importantes a tener en cuenta en todos los casos:

- La opción `-c` indica que estamos creando un tarball, mientras que `-x` nos dice que lo estamos descomprimiendo.
- `-v` muestra la lista de archivos que fueron procesados.
- `-f` es la última opción y debe preceder al nombre del tarball a crear.
- `gzip`es la herramienta más antigua de las tres y provee la menor compresión, mientras que `xz` es la más nueva y presenta una mayor compresión, pero a costa de un mayor uso de recursos del sistema al momento de comprimir. Finalmente, `bzip2`es una especie de promedio entre las otras dos.
- Cabe destacar que todas las opciones de `tar` (que podemos consultar en el *man page* correspondiente) se pueden utilizar con cualquiera de las herramientas de compresión mencionadas anteriormente.

Si bien existen otras herramientas más potentes y versátiles (aunque complejas) para realizar copias de respaldo en Linux, todo administrador de sistemas debe conocer `tar` y las herramientas de compresión como la palma de su mano.