Linux Administrator

Tema 4: Gestión de Dispositivos y Sistemas de Ficheros

Sonia Fernández Sapena

GESTIÓN DE DISPOSITIVOS

REPRESENTACION DE LOS DISCOS

Cada disco y cada partición está representado por un archivo especial de tipo bloque

IDE: Los discos con controladores IDE (PATA) se llaman hdX:

- hda: IDE0, Master
- hdb: IDE0, Slave
- hdc: IDE1, Master
- hdd: IDE1, Slave

SCSI, ATA, USB, FIREWIRE : SCSI, SCA,SAS, FiberChannel, USB, Firewire, Thunderbolt, y otros exóticos se llaman sdX. La enumeración sigue el orden de las tarjetas SCSI y los adaptadores asociados.

- Sda: primer disco SCSI
- Sdb: Segundo disco SCSI
- Sdc: tercer disco SCSI

GESTIÓN DE DISPOSITIVOS

PARTICIONADO

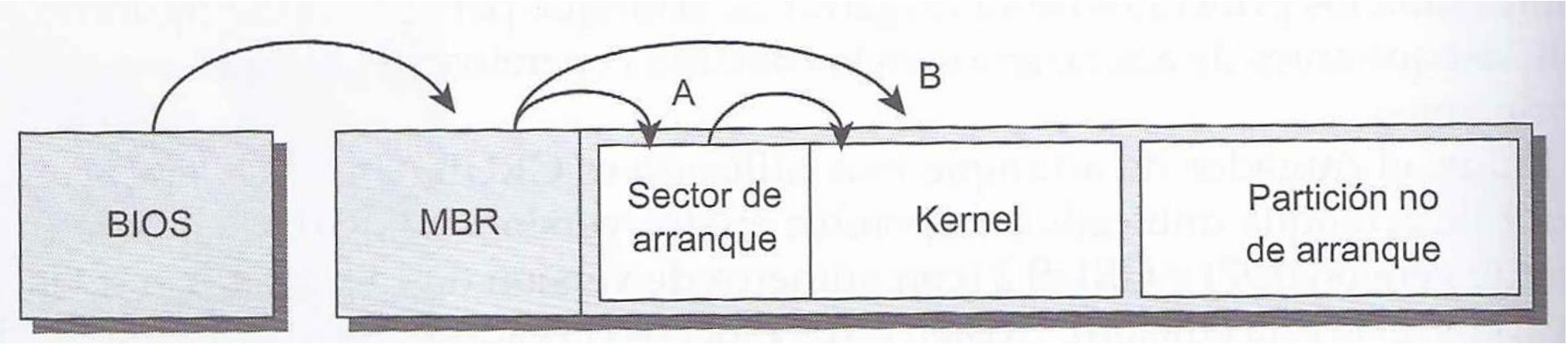
PARTICIONADO LÓGICO: Consiste en una división lógica del disco. Se fracciona el disco físico, real en varios discos virtuales lógicos: las particiones.

Vemos las particiones como discos independientes que contienen su propio sistema de archivos. Cuando creamos un esquema de particiones, este, se guarda en una parte del disco duro.

GESTIÓN DE DISPOSITIVOS

PARTICIONADO MBR (Master Boot Record)

MBR es la parte más interna del disco, está situado en los primeros 512 bytes y es dónde se ubica la tabla de particiones.



GESTIÓN DE DISPOSITIVOS

PARTICIONADO MBR

Existen tres tipos de particiones:

1. Las particiones primarias, en un total de 4 son las descritas en el MBR. Originalmente, existían sólo las particiones primarias. Este tipo de particiones se limitan a solamente cuatro por unidad física. Los sistemas operativos deben estar instalados en ellas para arrancar.
2. Las particiones extendidas, una sola por disco. Las particiones extendidas fueron creadas para resolver la limitación de tener únicamente cuatro particiones separadas por unidad física.
3. Las particiones lógicos: Un disco puede tener hasta 63 particiones IDE (15 en SCSI). Para superar el número de 15 particiones, es posible utilizar LVM (Logical Volume Management) que permite agrupar varios discos físicos en una sola unidad (Volume Group).

Se enumeran las particiones de 1 a n:

- Particiones de 1 a 3: primarias.
- Partición 4: extendida
- Particiones de 5 a n: lógicas

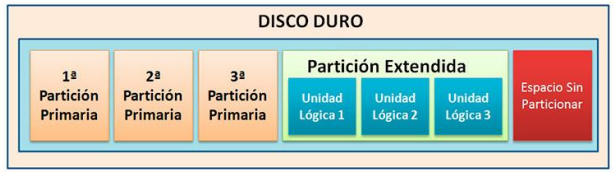
Ejemplos: hda1: Primera partición primaria del primer disco IDE, hdb5: quinta partición lógica del segundo disco IDE

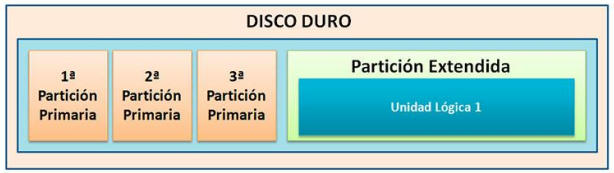
GESTIÓN DE DISPOSITIVOS

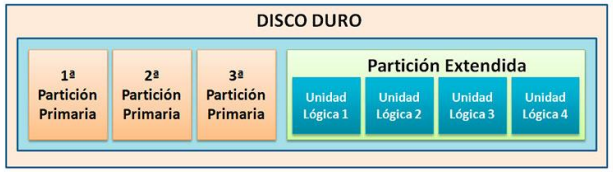
PARTICIONADO MBR

Existen tres tipos de particiones:

1. **Las particiones primarias**, en un total de 4 son las descritas en el MBR. Originalmente, existían sólo las particiones primarias. Este tipo de particiones se limitan a solamente cuatro por unidad física. Los sistemas operativos deben estar instalados en ellas para arrancar. Son las divisiones primarias del disco. En un disco duro, pueden existir de una a cuatro particiones primarias o hasta tres primarias y una extendida, cualquier sistema operativo puede detectar este tipo de particiones primarias, y asignarles una unidad, siempre y cuando el sistema operativo reconozca su formato (sistema de archivos). Una de las particiones primarias se llama la partición activa y es la de arranque. El ordenador busca en esa partición activa el arranque del sistema.
2. **La partición extendida**, también conocida como partición secundaria, es otro tipo de partición que actúa como una partición primaria; sirve para contener múltiples unidades lógicas en su interior. Fue ideada para romper la limitación de 4 particiones primarias en un solo disco físico. Solo puede existir una partición de este tipo por disco, y solo sirve para contener particiones lógicas. Por lo tanto, es el único tipo de partición que no soporta un sistema de archivos directamente.
3. **Las particiones lógicos**: Un disco puede tener hasta 63 particiones IDE (15 en SCSI). Ocupa una porción de la partición extendida o la totalidad de la misma, y se ha formateado con un tipo específico de sistema de archivos (FAT32, NTFS, ext3, ext4, etc.) y se le ha asignado una unidad, así el sistema operativo reconoce las particiones lógicas o su sistema de archivos. Se pueden tener un máximo de 15.







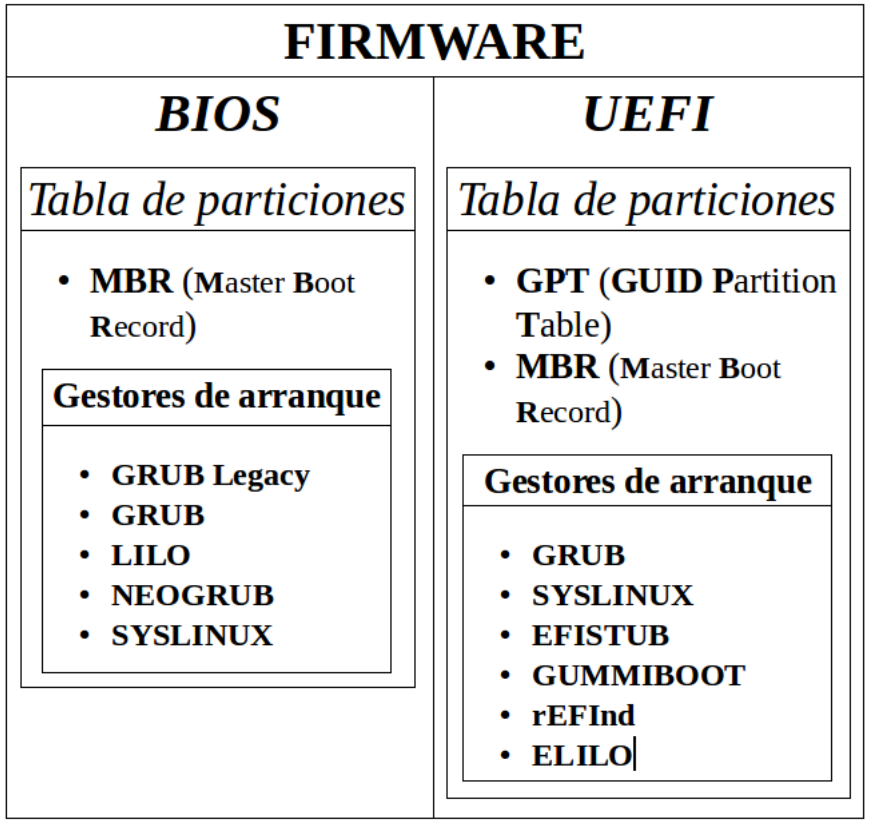
GESTIÓN DE DISPOSITIVOS

PARTICIONADO GPT

GPT (GUID Partition Table). Este es un estándar que permite describir la tabla de particiones de un disco duro, que forma parte de las especificaciones de UEFI.

- **UEFI, sustituye a la BIOS**
- **GPT reemplaza al particionamiento MBR**

UEFI es necesario en sistemas con discos de más de 2 TB, debido a que BIOS y MBR solo pueden direccionar 2 TB.



GESTIÓN DE DISPOSITIVOS

PARTICIONADO GPT

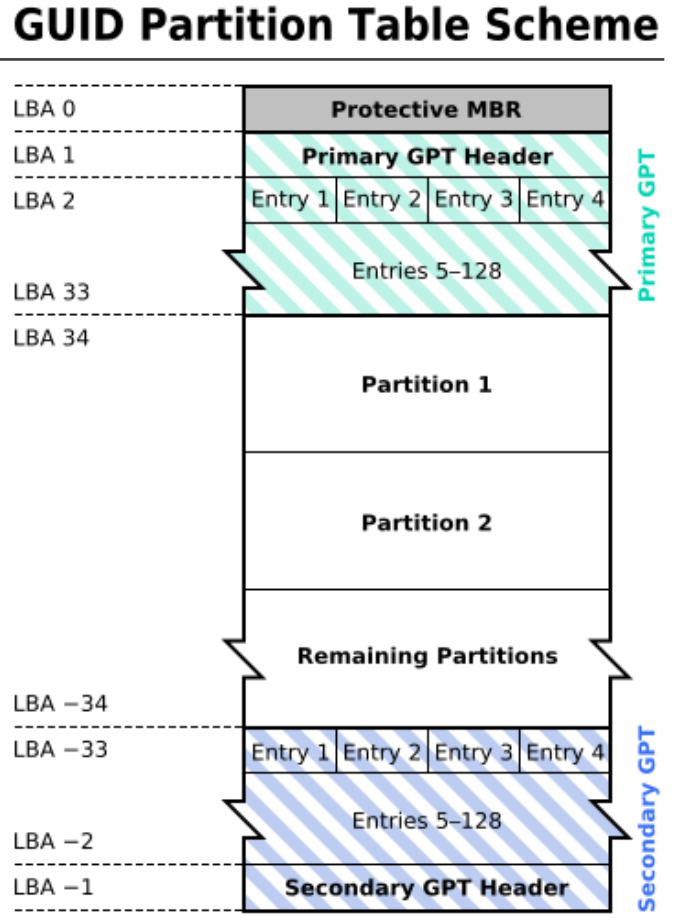
GUID: Identificador global único, es un valor codificado en 128 bits que sirve como identificador único para un componente de software.

LBA (Logical Block Addressing)

LBA 0: Son los primeros 512 bytes del disco, conservan su rol de MBR, pero son conocidos como **MBR protector**. Impedir las escrituras en el disco por herramientas que no reconocen el formato GPT.

LBA 1: Es la cabecera GPT está en la posición LBA 1 y contiene información sobre la estructura GPT.

LBA 2 a 33: Contienen los descriptores de las particiones..



GESTIÓN DE DISPOSITIVOS

HERRAMIENTAS DE PARTICIONADO MBR

fdisk (fixed disk): Es la herramienta tradicional en modo texto utilizada para el particionado *MBR* del disco. Para utilizar *fdisk* solo hay que escribir el comando seguido del disco a particionar (**#fdisk /dev/hdX** o sdX) y se desplegará un menú donde se nos presentan una serie de opciones. Podemos ver las particiones de un disco directamente desde línea de comandos, sin entrar al programa con **#fdisk -l**. ***fdisk*** no admite tabla de particiones *GPT*.

cfdisk: Editor de particionamiento con una interfaz diferente a *fdisk* y que además, a diferencia de esta, *cfdisk* nos permite redimensionar particiones extendidas cuando existe espacio libre tras ellas. Es fácil moverse por la interfaz mediante los cursores y además nos ofrece varias opciones obvias como *new* (para nuevas particiones), *type* (para escoger el tipo de sistema de archivo), *bootable* (si queremos hacer arrancable una partición), entre otras.

sfdisk: Una de las principales diferencias de *sfdisk* con respecto a *fdisk* es que no es interactiva, por lo que nos permite automatizar mediante scripts el particionado de un disco o crear un clon de nuestro disco como medida de seguridad. *sfdisk* no alinea las particiones y tampoco está diseñado para particiones de gran tamaño. Los principales uso de *sfdisk* son: mostrar el tamaño de las particiones, ver las particiones de un disco, checkear las particiones de un disco y reparticionar un disco.

GESTIÓN DE DISPOSITIVOS

HERRAMIENTAS DE PARTICIONADO MBR

fdisk: Herramienta modo texto para el particionado MBR

#fdisk [opciones] dispositivo

- **[Opciones]**
  – listar las particiones de un dispositivo: **-l**
  – Especificar el número de sectores: **-S**
  – Especificar el número de cilindros: **-C**
  – Especificar el número de cabezales: **-H**
  – Modo de compatibilidad con DOS: **-c=mode** (mode=dos o mode=nondos, por defecto usa nondos)
  – Imprimir ayuda: **-h**

GESTIÓN DE DISPOSITIVOS

HERRAMIENTAS DE PARTICIONADO MBR

Si usamos **fdisk** sin opciones indicando directamente el dispositivo (*$ sudo fdisk /dev/sdb*) entraremos al modo interactivo. Algunas opciones del modo interactivo son:

– **n**: Crear una nueva partición

Nota: fdisk mide los puntos de inicio y fin de una partición en cilindros o sectores, aunque podemos definir el tamaño con un sigo + un valor numérico y un sufijo (MB, GB..).

d : Elimina una partición
l : Enumera los códigos de tipo de particiones
t : Cambiar el tipo de una partición
L : Si hemos usado **t** para cambiar el tipo de partición, podemos usar **L** para que se despliegue una lista de tipos de particiones y elegir una de entre ellas
p : Muestra las particiones.
a : Marca una partición como activa para que se arranque desde ella
m o **?** : Desplegar el menú o ayuda
q : Salir sin guardar cambios
w : Salir escribiendo los cambios

GESTIÓN DE DISPOSITIVOS

HERRAMIENTAS DE PARTICIONADO MBR

sfdisk: Herramienta de particionado no interactiva (a diferencia de *fdisk*).

#sfdisk [opciones] dispositivo

Si no utilizamos opción alguna con *sfdisk* querrá decir que queremos particionar el disco. Podemos indicar con **-O** <nombre_archivo> , que queremos realizar una copia de los sectores que vamos a modificar y en caso de accidente restaurarlo con **-I**

- **[*Opciones*]:**
  – Listar las particiones: **-l**
  – Podemos chequear la partición con: **-v**
  – Imprimir los tipos de sistema de archivos: **-T**
  – Cambiar el ID de una partición: **-c** (–print-id y –change-id)
  – Mostrar el tamaño del disco: **-s**
  – Crear un volcado de las particiones de un disco: **-d** (#sfdisk -d /dev/sda > sda.out)
  – Marcar una partición como activa o inactiva: **-A** <partición>

Ejemplo:

#sfdisk /dev/sdc -O sdc-partition-sectors.sabe / **#sfdisk /dev/sdc -I sdc-partition-sectors.save**

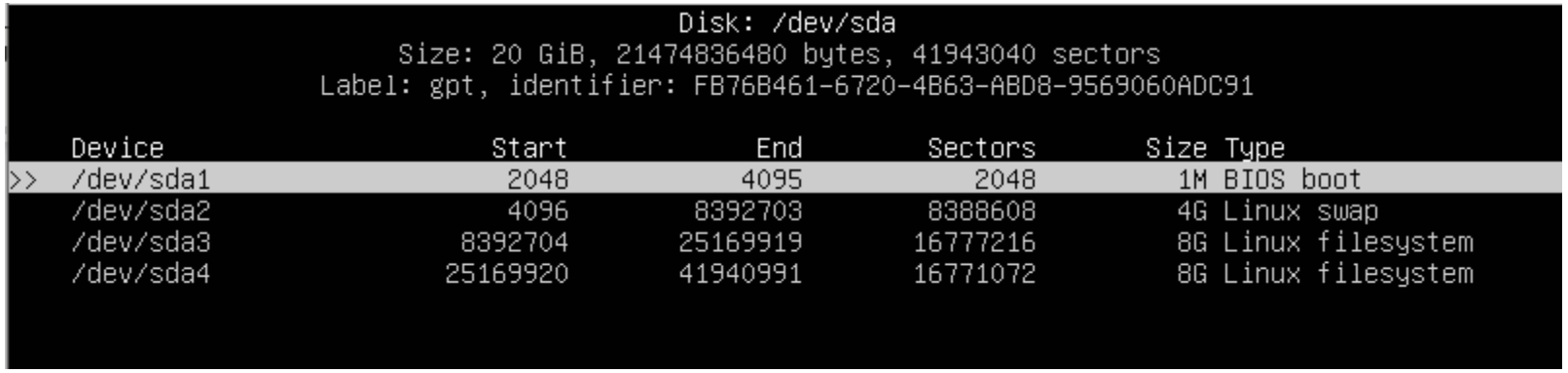
GESTIÓN DE DISPOSITIVOS

HERRAMIENTAS DE PARTICIONADO MBR

cfdisk: Administrar una tabla de particiones mediante un menú.

#cfdisk [opciones] dispositivo

Si no marcamos opciones entramos a un menú donde podremos crear, modificar, eliminar, mover, etc… particiones.



GESTIÓN DE DISPOSITIVOS

HERRAMIENTAS DE PARTICIONADO GPT:

GNOME Partition Editor o **gparted**: Digamos que es la herramienta gráfica por excelencia para escritorios *GNOME*, formada principalmente por la librería *GNU parted*. Según que sistema de archivo manipule, *gparted* nos permite una infinidad de cosas tanto en particiones *MBR*, *GPT*, *APM* y *BSD* (entre otros), como por ejemplo, crear, eliminar, mover, redimensionar particiones, seleccionar el tipo de sistema de archivos, etc.. Habrá opciones que solo podamos usar si arrancamos desde un disco de rescate o las particiones a tratar no están en uso. Tiene soporte para *CHS* y *LBA*.

GPT fdisk es un set de herramientas de modo texto para el particionado del disco, tanto para *MBR* como para *GTP*. En este set se encuentran las herramientas:

gdisk: Es el "hermano" de *fdisk* pero para particiones *GPT*, tanto para discos rígidos como SSD, donde se encarga por defecto de que sus particiones estén perfectamente alineadas. Su uso es puramente desde línea de comandos con permisos de administrador. Algunos de sus comandos útiles son:

- **m**: Volver al menú principal
- **d**: Para eliminar una partición
- **n**: para crear una partición
- **o**: Para crear una nueva tabla de particiones *GPT*
- **b**: Bakcup de la tabla de particiones
- **t**: para cambiar el tipo de partición

cgdisk (interfaz interactiva) y **sgdisk** (útil en scripts) : que son válidas tanto para Linux como para Windows, FreeBSD y OS X.

GESTIÓN DE DISPOSITIVOS

HERRAMIENTAS DE PARTICIONADO GPT:

parted: Herramienta de particionado GPT.

#parted [opciones] dispositivo [comandos [opciones_comandos]]

[Opciones]:

– Listar las particiones y su tamaño de un dispositivo: **-l** (–list)
– Pasar un script para que interactúe con parted y no el usuario: **-s**
– Chequear de forma simple una partición: **check**
– Copiar una partición de un dispositivo (por defecto el dispositivo actual) a otro dispositivo: **cp** <origen> <destino>
– Imprimir la ayuda para un determinado comando: **help**
– Crear un sistema de archivo determinado (FAT16/32, ext2, reiserfs o linux-swap): **mkfs** <partición> <fs-type>
– Crear una partición vacía con un sistema de archivos determinado **mkpart** <tipo-partición> <fs-type> start end
– Crear una partición dándole formato **mkpartfs** <tipo-partición> <fs-type> start end
– Crear una nueva tabla de particiones: **mklabel** <tipo_de_tabla>
– Mover una partición indicando el inicio (que queremos que tenga) y el final: **move** <partición> start end
– Darle nombre a una partición: **name** <partición> nombre
– Desplegar la tabla de particiones: **print**
– Rescatar una partición indicando el inicio y el final de la misma: **rescue** start end
– Redimensionar un sistema de archivos de una partición (por defecto en MB): **resize** <partición> start end
– Eliminar una partición: **rm** <partición>
– Seleccionar un dispositivo o partición RAID/LVM: **select** <dispositivo>

GESTIÓN DE DISPOSITIVOS

HERRAMIENTAS DE PARTICIONADO GPT:

gdisk: Es una herramienta de particionado a la que pasamos como argumento el dispositivo a particionar y podemos ejecutar una serie de acciones sobre ella. Estas acciones son listadas con los comandos **help** o **?** una vez dentro de *gdisk*. Algunas de estas opciones son similares a las de *fdisk*:

[Opciones]:
– Crear un backup de la tabla de particiones: **b**
– Modificar el nombre de la partición: **c**
– Eliminar una partición: **d**
– Mostrar información sobre una partición: **i**
– Listar los tipos conocidos de particiones: **l**
– Crear una nueva partición: **n**
– Crear una nueva tabla de particiones *GPT* vacía: **o**
– Imprimir la tabla de particiones: **p**
– Salir sin guardar: **q**
– Salir guardando: **w**
– Opciones de recuperación: **r**
– Comprobar disco: **v**
– Funciones extras: **x**

Nota: Cuando use el comando n , se debe pulsar ‘Enter’ para dar a la partición el último número libre y es necesario pulsar ‘Enter’ nuevamente para que acepte el sector de arranque por defecto para la nueva partición antes de establecer el tamaño que necesite para el último sector.

GESTIÓN DE DISPOSITIVOS

HERRAMIENTAS DE PARTICIONADO GPT:

Volumen lógico:

lvcreate : Crear un volumen lógico (como si fuese una partición) para luego asignarle un tipo de sistema de archivo. Lo bueno es que podremos redimensionar el tamaño de esta "partición" tanto como el grupo de volumen nos permita.

#***lvcreate*** *[opción1 argumento] [opción2 argumento]….[opciónN argumento] <grupo_de_volúmenes> [volumen_físico]*

[Opciones]:
– Indicar el tamaño: **-L** <tamaño>
– Indicar el nombre: **-n** <nombre>
– Dar permisos de lectura (r) o ambos (por defecto rw): **-p** {r|rw}
– Crear un snapshot: **-s**

Ejemplo:

#***lvcreate*** *-L 50GB -n "lvhome" -p r mi_vg_1*

(donde *mi_vg_1* es mi grupo de volúmenes anteriormente creado)

#*lvcreate* –size 100m –snapshot –name snapvol /dev/vg00/lvhome

GESTIÓN DE DISPOSITIVOS

HERRAMIENTAS DE PARTICIONADO GPT:

lvconvert: Convertir un volumen lógico

lvresize: Redimensiona un volumen lógico

lvdisplay: Muestra los volúmenes lógicos creados en el sistema

Nota: Los volúmenes lógicos suelen almacenarse bajo */dev/mapper* y */dev/nombre_grupo_volúmenes*, por lo que nuestro volumen tendrá el nombre o ruta completa */dev/mapper/vg1-lvhome* o */dev/vg1/lvhome*. En caso de no encontrarse los volúmenes necesitaremos cargar el modulo y escanear los volúmenes.

Cargar módulo de *LVM*:

#***modprobe*** *dm-mod*

Escanear *LVM*:

#vgscan
#vgchange -ay

Si queremos dar formato a un volumen lógico usamos *mkfs*

#***mkfs.ext4*** */dev/mapper/vg1-lvhome*

GESTIÓN DE DISPOSITIVOS

HERRAMIENTAS DE PARTICIONADO GPT:

Redimensionar un volumen lógico y su sistema de archivos:

Aumentar la capacidad de un *LV*:

lvresize: Redimensiona un volumen lógico***#lvresize*** *-L +5GB /dev/vg1/lvhome*
(aumentamos el tamaño en 5GB a *lvhome*, no es necesario tener desmontado el volumen)

resize2fs: Redimensiona sistemas de archivo ext2,ext3 y ext4

resize_reiserfs: Para sistema de archivo reiserfs

xfs_growfs: Para *XFS* (no permite disminuir su tamaño solo aumentarlo)

#resize2fs */dev/vg1/lvhome*

Nota: Debemos de tener el volumen desmontado, podemos hacerlo pasando a nivel 1 (*telinit 1*) y desmontarlo.

GESTIÓN DE DISPOSITIVOS

HERRAMIENTAS DE PARTICIONADO GPT:

Disminuir el tamaño de un *LV*:

Nota: En este caso, desmontaremos primero el volumen, reduciremos el sistema de archivos y luego el volumen. Supongamos que nuestro *lvhome* tiene una capacidad de 9GB y queremos dejarlo en 4GB.

#***resize2fs*** */dev/vg1/lvhome 4G*

#***lvreduce*** *-L -5G /dev/vg1/lvhome o #**lvreduce** -L 4G /dev/vg1/lvhome*

Nota: Ahora ya podemos montar nuestro volumen *lvhome*

GESTIÓN DE DISPOSITIVOS

SISTEMA DE ARCHIVOS

Para poder acceder a los datos almacenados en un disco duro, debemos primero crear en él un sistema de archivos. Un sistema de archivos es la forma en la que se estructuran de forma lógica los datos almacenados en su estructura física. Es decir proporciona una forma de acceso mediante direcciones lógicas a datos almacenados en estructuras físicas. Para poder crear dicha estructura de archivos es necesario dar formato al disco. Un sistema de archivos debe facilitar al usuario una visión estructurada de sus datos, que permite distinguirlos, encontrarlos, tratarlos y trabajar con ellos.

METADATOS: Los metadatos de un archivo son sus propiedades, aunque en Linux, se denominan inodo. El contenido de los metadatos suele ser:

- Permisos
- Fechas de acceso y modificación
- Propietario, grupo
- Tamaño
- Número de bloques utilizados
- El tipo de archivos
- Contador de vínculos
- Un árbol de direcciones de bloques de datos.

GESTIÓN DE DISPOSITIVOS

SISTEMA DE ARCHIVOS

LOS NOMBRES DE LOS ARCHIVOS

Los nombres pueden tener una longitud de 255 caracteres. La extensión no existe y no es importante, se puede utilizar con fines orientativos, pero Linux no la necesita.

ARCHIVO DE REGISTRO

Los actuales sistemas de archivos disponen a menudo de mecanismos que permiten garantizar en la medida de los posible la integridad de los datos. El sistema más utilizado es el “journalist”. El sistema almacena un registro donde almacena todos los cambios antes de realizarlos realmente.

GESTIÓN DE DISPOSITIVOS

SISTEMA DE ARCHIVOS

La mayoría de los sistema de archivos UNIX tienen una estructura general parecida, aunque los detalles exactos pueden variar un poco. Los conceptos centrales son **superbloque**, **nodo-i**, **entradas de directorio** y **bloque de datos.**

El ***superbloque*** contiene metadatos críticos del sistema de archivos como información acerca del tamaño, cantidad de espacio libre y donde se encuentra los datos. Si el superbloque es dañado, y su información se pierde, no podría determinar que partes del sistema de archivos contiene información y lo mas seguro es que no podamos ni montar el sistema de archivos por este motivo, se guardan varias copias del superbloque en determinados bloques del sistema de archivos.

Un ***nodo-i*** *(inode, Nodo de indice)*: Contienen información sobre un archivo, salvo su nombre. Un inodo guarda la siguiente información:

- **Identificador** del dispositivo que alberga al sistema de archivos.
- **Número de inodo** que identifica al archivo dentro del sistema de archivos.
- **Longitud** del archivo en bytes (tamaño del archivo).
- Identificador del **usuario propietario** del archivo con derechos diferenciados.
- Identificador del **grupo propietario** del archivo con derechos diferenciados.
- **Modo de acceso** al archivo.
- **Marcas de tiempo**: última modificación (*mtime*), acceso (*atime*) y de alteración del propio inodo (**ctime**).
- **Número de enlaces** (*hard links*) es decir, entradas de directorio asociados con este inodo.

GESTIÓN DE DISPOSITIVOS

SISTEMA DE ARCHIVOS

***Entradas de directorios*:** Contienen el nombre de un archivo y el número de nodo-i que representa a este.

***Bloque de datos*:** Conjunto de sectores contiguos que componen la unidad de almacenamiento más pequeña de un disco. El número de bloques es determinado al crearse el sistema de archivos, y depende del tipo elegido, así como de las opciones que use en su creación. Un bloque solo puede contener un archivo, o parte de un archivo, es decir, si el archivo es más pequeño que el bloque, el espacio restante del bloque no será utilizado. El tamaño por defecto de bloque es de 4KB.

GESTIÓN DE DISPOSITIVOS

SISTEMA DE ARCHIVOS

Nativos de Linux:

- **ext2** : extendido 2 es el tradicional sistema de archivos nativo de Linux, es un sistema fiable y perfecto para particiones **/boot** o discos de menos de 1GB. No posee respaldo de transacciones.
- **ext3** : Es el evolutivo del segundo sistema de archivos nativo para Linux (ext2), incluye respaldo de transacciones por lo que es un sistema de archivos igual de fiable que ext2 y además se recupera mucho mas rápidamente de los cortes de alimentación y las caídas del sistema.
- **ext4** : Es la nueva generación de la familia “extendidos”, pensado para trabajar con archivos grandes (+2TB), discos muy grandes (+16TB) y extensiones que mejoran el rendimiento.
- **Reiserfs** : Diseñado para trabajar con un gran número de archivos pequeños (menos 32KB) ya que utiliza varios trucos para encajar los finales de los archivos en los espacios sin utilizar de otros. *Reiserfs* hasta su versión 3.x viene incluido en el *kernel*. Tiene varias características que no todos los sistemas de archivos poseen, como *journaling* que previene el riesgo de corrupción del propio sistema de archivo, reparticionamiento tanto con el filesystem montado (online, solo es posible aumentarlo) como desmontado (offline, permite aumentar o disminuir el tamaño) o el *Tail packing* que reduce la fragmentación interna.

GESTIÓN DE DISPOSITIVOS

SISTEMA DE ARCHIVOS

Nativos de Linux:

- **JFS** (*Journaling File System*): Sistema de archivos de 64bits con respaldo de transacciones creado por IBM para sistemas de altos rendimientos y donado a Linux (su versión OS/2) aunque existen versiones para otros sistemas como AIX, HP-UX… *JFS* utiliza un método para organizar los bloques vacíos estructurándolos en un árbol y además al ser 64bits soporta bastante bien el manejo de grandes ficheros y particiones *LFS* (*Large File Support*). Podríamos resaltar algunas características como el *Journaling* que al igual que r*eiserfs* sigue el principio de *metadata only* que en vez de una comprobación completa, solo se tienen en cuenta las modificaciones en los metadatos provocados por las actividades del sistema, eficiente administración de directorios, mejor utilización de la memoria mediante adjudicación dinámica de inodos, etc.
- **XFS** (*Extents File System*): Al igual que *JFS*, es un sistema técnicamente sofisticado aunque se ganó su reputación por su robustez, velocidad y flexibilidad en *IRIX*, no todas estas características fueron bien encajadas en Linux. *Red Hat Enterprise Linux* incorporó en su versión 7 *XFS* como su sistema de archivos por defecto, destacando su capacidad de manejar una partición de hasta 500TB
- **Btrfs** (*B-tree FS*): Es un sistema de archivos copy-on-write anunciado por *Oracle Corporation* para *GNU/Linux*. El objetivo de *Btrfs* es sustituir a *ext3*, eliminando el mayor número de sus limitaciones como el tamaño máximo de los ficheros o la compatibilidad con nuevas tecnologías. *Btrfs* tiene la intención de centrarse en la tolerancia a fallos, reparación y fácil administración

GESTIÓN DE DISPOSITIVOS

SISTEMA DE ARCHIVOS

Otros sistemas de archivos:

- **FAT** (*File Allocation Table*): Antiguo y primitivo pero omnipresente ya que los principales OS entienden a *FAT* lo que lo convierte en un buen file system para discos extraibles con los que intercambiar datos. Existen dos variantes de *FAT* que principalmente difieren en el tamaño de la estructura de datos de *FAT* (FAT16 bits,FAT32 bits, extFAT, VFAT).
- **NTFS** (*New Technology File System*): Es el preferido por los principales OS Windows (desde mas antiguos hasta los mas nuevos). Desafortunadamente el soporte de *NTFS* para Linux es bastante rudimentario no pudiendo escribir nuevos datos en particiones *NTFS*. Si necesitásemos tener un buen soporte de lectura/escritura podríamos probar con el controlador *NTFS-3G* que reside en el área del usuario en vez de en el del kernel.
- **HFS/HFS+** (*Hierarchical File System*): Es el sistema de archivos para usuarios de Mac OS, aunque si quisiéramos cambiar archivos con ellos, lo mejor sería utilizar *FAT*. Apple ha ampliado *HFS* con *HFS+* conocido como *HFS extendido* (plus) para dar un mejor soporte a discos duros de gran capacidad y a diversas funcionalidades Unix.
- **ISO-9660**: Estándar para *CD-ROM*, presente en varios niveles. El Nivel 1 es similar a FAT, por lo que solo soporta nombres de archivos 8.3. Los niveles 2 y 3 soportan nombres de archivos de hasta 32 caracteres. El código para este sistema de archivo en Linux es iso9660. El soporte de *ISO-9660* de Linux también funciona con extensiones *Rock Ridge* (permite nombres largos, permisos, enlaces simbólicos, etc..) y *Joliet*.
- **UDF** (*Universal Disc Format*): Siguiente generación para discos ópticos. Suele usarse en discos regrabables y DVDs. La compatibilidad lectura/escritura de *UDF* con Linux aun es precaria.

GESTIÓN DE DISPOSITIVOS

SISTEMA DE ARCHIVOS

Sistema de Ficheros	Máxima capacidad fichero	Máxima capacidad volumen	Journaling	Permisos	Bifurcaciones
FAT12	32 Mb	32 Mb	No	No	No
FAT16	2 Gb	2 Gb	Si	No	Si
FAT32	4 Gb	2 Tb	Si	No	Si
NTFS	16 TB	256 Tb	No	Si	Si
HPFS	2 Gb	-	No	Si	Si
HFS	2 Gb	21 Tb	No	AppleShare	Si
HFS+	8 Tb	8 Tb	Si	Unix	Si
EXT2	2 Tb	16 Tb	Si	Unix	Si
EXT3	16 Gb	2/32 Tb	Si	Unix	Si
EXT4	16 Gb	1 Eb	Si	POSIX	No
REISER3	8 Tb	16 Tb	Si	Si	Si
REISER4	8 Tb	16 Tb	Si	Si	Si
ZFS	16 Eb	16 Tb	Si	POSIX	Si
XFS	8 Eb	16 Tb	Si	Si	Si

GESTIÓN DE DISPOSITIVOS

COMANDOS SISTEMA DE ARCHIVOS: CREACION

mkfs : Dar formato a una partición.

#mkfs [opciones] partición

[Opciones]:

– Indicar el tipo de sistema de archivo para dar formato: **-t** <tipo-fs>
– Comprobar bloques defectuosos durante el formato: **-c**
– Reducir el espacio libre que deja *mkfs* para tareas de recuperación (5% por defecto): **-m** <nuevo-%>

Nota: Podemos llamar a *mkfs* de distintas maneras, según que tipo de sistema de archivos utilicemos por ejemplo:
mkfs -t ext4 es igual a *mkfs.ext4*

GESTIÓN DE DISPOSITIVOS

COMANDOS SISTEMA DE ARCHIVOS: CREACION

Ejemplo 1: Formatear la partición **/dev/sdb1** con el sistema de archivos **ext4**, con un espacio libre (desperdiciado, podemos incluso darle un 0, útil si no tenemos un sistema operativo en esta partición) reducido al **1%** y con una etiqueta "**Mi USB**":

$ sudo mkfs.ext4 -m1 -L "Mi USB" /dev/sdb1

Ejemplo 2 : creamos un sistema de archivos de tipo ext2 en una primera partición sdb1.

 # mkfs –t ext2 /dev/sdb1

Ejemplo 3*:* Crear un sistema de archivos transaccional ext3 con un tamaño de bloques de 2048 bytes y un inodo por cada 16 kb. Los usuarios pueden utilizar la totalidad del sistema. La etiqueta es DATA (con ext3 no hace falta –j)

 # mkfs –t ext3 –b 2048 –i 16384 –m 0 –L "DATA" /dev/sdb1

GESTIÓN DE DISPOSITIVOS

COMANDOS SISTEMA DE ARCHIVOS: CREACION

mke2fs : Es como *mkfs* pero exclusiva para la familia de los sistemas de archivos extendidos. Nos permite un mayor número de opciones ya que se centran unicamente en estos sistemas de archivos. Por ejemplo el equivalente a *mkfs.ext3* sería *mke2fs -j*, crea un sistema de archivo extendido con respaldo de transacciones. Este comando es útil si queremos conocer los bloques en los que se guardan los backups de superbloques, deberemos de pasar la opción **-n** (simulación) y **-b** <blocksize> para que sea mas preciso.

#mke2fs [opciones] partición

- **[Opciones]:**
  – Crear la partición con una etiqueta: **-L** <etiqueta>
  – Crear un sistema de archivos con un UUID determinado: **-U** <UUID>
  – Crear el sistema de archivo ext2, ext3 o ext4: **-t** <tipo-fs>
  – Utilizar mke2fs en un script: **-q**
  – Simular un formateo: **-n**
  – Pasarle el tamaño del bloque: **-b** <blocksize>

mkswap: Crear una partición de intercambio

swapon: Activar una partición de intercambio

GESTIÓN DE DISPOSITIVOS

COMANDOS SISTEMA DE ARCHIVOS: MONTAJE

mount: Nos permite montar un sistema de archivo en un punto de montaje.

[Opciones extras válidas también para /etc/fstab]:
– Utiliza las opciones por defecto *(rw, suid, dev, exec, auto, nouser, async)*: **defaults**
– Interpretar caracteres o bloques especiales en el filesystem: **dev**
– Permitir ejecutar archivos binarios: **exec**
– Montar un archivo como si de una partición se tratase (por ejemplo una imagen.img): **loop**
– Montar o no el dispositivo de forma automática, bien durante el arranque o durante la ejecución de mount -a: **auto/noauto**
– Permitir o no, montar el dispositivo a usuarios indicando #mount /y_el_punto_de_montaje: **user/nouser**
– Permite a cualquier usuario desmontar el filesystem montado por otro usuario (user no permite esto): **users**
– Permitir o no, montar el dispositivo igual que user pero a diferencia de que debe de ser el propietario: **owner**
– Activar nuevas opciones sin tener que desmontar y volver a montar el dispositivo: **remount**
– Montar en solo lectura: **ro**
– Montar en lectura/escritura, solo para filesystem soportados: **rw**
– Definir un propietario para los archivos de la nueva unidad montada (ideal para filesystem que no reconocen los permisos Linux, como vfat, ntfs..): **uid=valor** (donde valor puede ser por ejemplo el id del usuario, 500, podemos ver los id en /etc/passwd)
– Igual que uid pero para el grupo: **gid=valor**
– Montar el dispositivo con una umask específica. útil para los filesystem que no admiten permisos Unix. Funciona igual que para archivos Linux, es decir en binario y restando esos bits a la mascara 777. umask=027 generará permisos 750: **umask=valor**
– Igual que para la umask de archivos pero solo para directorios: **dmask=valor**
– Igual que dmask pero solo para archivos: **fmask=valor**
– Desactivar las extensiones RockRidge para CD-ROM ISO-9660: **norock**
– Desactiva las extensiones Joliet para CD-ROM ISO-9660: **nojoliet**

GESTIÓN DE DISPOSITIVOS

COMANDOS SISTEMA DE ARCHIVOS: MONTAJE

mount: Nos permite montar un sistema de archivo en un punto de montaje.

Ejemplo 1: Montaje por periférico. Montar la raíz del sistema sdb1 con ext4 en el directorio /mnt/DATA

#mount –t ext4 /dev/sdb1 /mnt/DATA

Ejemplo 2: Montaje por etiqueta.

#mount –t ext4 –L DATA /mnt/DATA

Ejemplo 3: Montaje por UUID

#mount –t ext4 –U 67f6e4b8-…… /mnt/DATA

Ejemplo 4: Volver a montar un sistema de archivos en modo de solo lectura

#mount –o ro,remount /mnt/DATA

Ejemplo 5: Montarlo todo ***#mount -a***

GESTIÓN DE DISPOSITIVOS

COMANDOS SISTEMA DE ARCHIVOS: MONTAJE

umount: Desmonta un sistema de archivos. Debemos comprobar que ningún proceso esté utilizando ningún archivo del sistema que queremos desmontar.

#umount [-afrnv] [-t file_system] dispositivo | punto de montaje

[Opciones]:

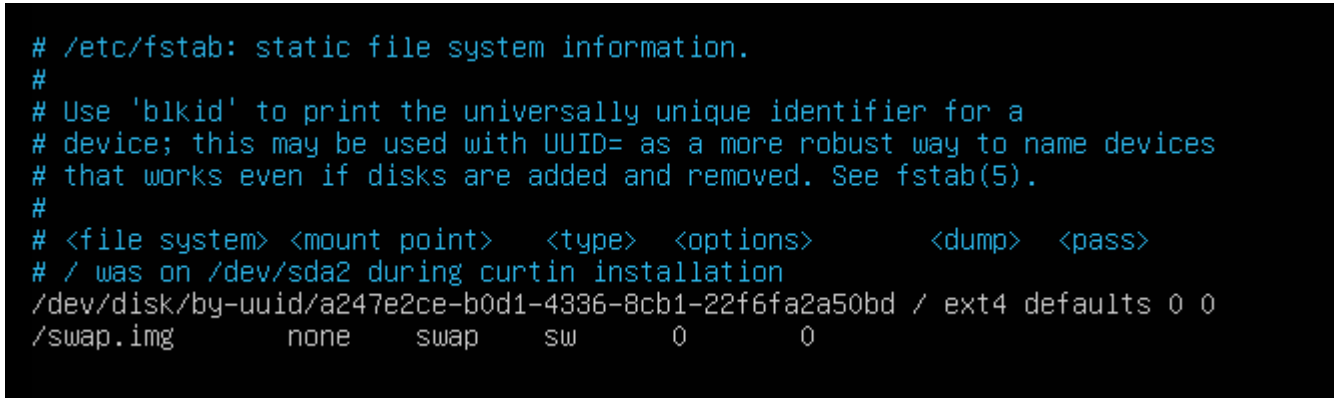
– Desmontar todos los dispositivos con entradas en /etc/mtab y que estén inactivos: **-a**
– Forzar el desmontaje: **-f**
– En caso de que umount no pueda desmontar el dispositivo lo vuelve a montar en solo lectura: **-r**
– Muestra la información: **-v**
– Desmonta sin escribir en /etc/mtab: **-n**
– Especificar el sistema de archivos: **-t <file_system>**

Nota: Al igual que ocurre con *mount*, los usuarios normales no pueden utilizar *umount* a menos que se haya indicado en su montaje alguna opción como *user*, *users* y *owner*.

GESTIÓN DE DISPOSITIVOS

COMANDOS SISTEMA DE ARCHIVOS: MONTAJE

/etc/fstab: Contiene una configuración estática de las diferentes opciones de montaje de los sistemas de archivos. Este archivo es invocado siempre que se inicia el sistema, ya que es aquí donde se especifican los periféricos y sus puntos de montaje. Contiene seis campos:



1. *Periférico: el que se va a montar*
2. *Punto de montaje*
3. *Typefs (ext2,ext3,ext4,VFAT, etc)*
4. *Opciones: opciones de montaje*
5. *Dump: Frecuencia de volcado para backups*
6. *Fsck: Frecuencia de verificación de archivos ;0:ignorar,1: en primero, 2 en segundo (en paralelo los que tienen el mismo número)*

GESTIÓN DE DISPOSITIVOS

COMANDOS SISTEMA DE ARCHIVOS: MONTAJE

Montaje durante el arranque

Durante la secuencia de inicio, uno de los scripts lee el archivo /etc/fstab, casi al principio del boot, entre la carga del núcleo y el inicio de los servicios. Todos aquellos sistemas de archivos que no tengan configurada la opción noauto se montan automáticamente porque el montaje automático está implícito.

El primero es el sistema de archivos raíz /, luego la swap y los demás sistemas de archivos,

GESTIÓN DE DISPOSITIVOS

COMANDOS SISTEMA DE ARCHIVOS: REPARACIÓN Y COMPROBACIÓN

badblocks: Busca bloques defectuosos en un dispositivo, normalmente una partición de disco.

 # badblocks dispositivo [primer-bloque] [último-bloque]

[Opciones]

- Especificar el tamaño (en bytes) del bloque (por defecto 1024): **-b**
- Especificar el número de bloques defectuosos 'posibles' antes de finalizar el test automáticamente: **-e**
- Leer desde una lista de bloques malos ya conocidos para evitar que vuelvan a ser leídos y desperdiciar mas tiempo: **-i** <fichero>
- Realizar una prueba no destructiva, es decir de solo lectura (no se puede combinar con -w): **-n**
- Escribir la lista de bloques defectuosos en un fichero: **-o** <archivo>
- Mostrar el porcentaje del proceso de escaneado: **-s**
- Usar modo destructivo, es decir de escritura: **-w**

GESTIÓN DE DISPOSITIVOS

COMANDOS SISTEMA DE ARCHIVOS: REPARACIÓN Y COMPROBACIÓN

dumpe2fs: Informa sobre la configuración y estado de un sistema de archivo. Aunque se puede ejecutar estando montado el sistema de archivos a diagnosticar, se recomienda que este permanezca desmontado durante la ejecución. Este comando es útil por ejemplo si queremos conocer el tamaño de bloque, el UUID del dispositivo, el tamaño de ínodo, las veces que se ha montado, y un largo etc ya que muestra bastante información útil

#dumpe2fs [opciones] dispositivo

[Opciones]:

- Imprimir bloques marcados como defectuosos: **-b**
- Usar el superbloque para examinar el dispositivo (casos en los que el sistema de archivo está bastante corrupto): **-o superblock=**<superblock>
- Usar un tamaño concreto de bloque (se usa para casos de filesystem corruptos): **-o blocksize=**<blocksize>
- Mostrar solo información relativa al superbloque: **-h**

GESTIÓN DE DISPOSITIVOS

COMANDOS SISTEMA DE ARCHIVOS: REPARACIÓN Y COMPROBACIÓN

tune2fs: Permite configurar muchos parámetros del sistema de archivos (ext2, ext3 y ext4) de los que informa dumpe2fs o usando la opción **-l**.

#tune2fs [opciones] dispositivos

[Opciones]:

- Ajustar el número de veces que un filesystem puede ser montado sin que fsck compruebe su estado: **-c**
- Engañar el conteo de número de veces que un sistema de archivos ha sido montado sin ser revisado por fsck: **-C**
- Ajustar el intervalo de tiempo que puede transcurrir sin que un filesystem sea comprobado por fsck: **-i** <tiempo> (se añade el sufijo **d** (días), **w** (semanas) o **m** (meses)
- Añadir respaldo de transacciones a un sistema de archivos ext2: **-j**
- Definir un % de bloques reservados en el disco para root (esto es lo que hace mkfs -m) que por defecto es un 5%: **-m**
- Definir el espacio reservado en números de bloques en vez de %: **-r** <bloques>
- Forzar a *tune2fs* a terminar su ejecución aun cuando existan errores. Esta opción es útil sobre todo cuando hemos eliminado el registro .journal o es externo al sistema de archivos: **-f**
- Reservar bloques de un filesystem a un determinado grupo que podemos pasar a través de su guid o nombre de grupo: **-g** <grupo>
- Sobreescribir la configuración del journal creado por la opción -j: **-J size**=journal-size **location**=journal_location **device**=external_journal

GESTIÓN DE DISPOSITIVOS

COMANDOS SISTEMA DE ARCHIVOS: REPARACIÓN Y COMPROBACIÓN

debugfs: Depurador interactivo de filesystem. Puede ser utilizado tanto para comprobar la configuración de un sistema de archivo como para modificarla. Una vez que estemos interactuando con *debugfs* nos permitirá pasarle comandos.

#debugfs [opciones] dispositivo

[Opciones]:

- Especificar que el sistema de archivos se abra en modo de lectura y escritura: **-w**
- Forzar al sistema de archivos ser abierto en modo de solo lectura, útil para casos precarios o inconsistencia del filesystem: **-c**
- Pasar a *debugfs* un archivo de comandos para ser ejecutados y finalizar: **-f** <archivo_de_comandos>

[Comandos]:

- Mostrar información sobre los superbloques del filesystem: **show_super_stats | stats** (similar a dumpe2fs)
- Mostrar el nodo de indice de un archivo o directorio: **stat** <nom_file>
- Deshacer la eliminación de un archivo: **(undel)eted** <inodo> <nombre>
- Cerrar los filesystem abiertos: **close**
- Obtener ayuda: **list_request | help | lr | ?**
- Salir de debugfs: **quit**

GESTIÓN DE DISPOSITIVOS

COMANDOS SISTEMA DE ARCHIVOS: REPARACIÓN Y COMPROBACIÓN

fsck: Comprobar y reparar uno o mas sistemas de archivos. En realidad es un front-end de varios comprobadores de filesystem (*fsck.fs_type*)

#fsck [opciones] [opciones-especificas] filesystem

[Opciones]:

- Revisar todos los sistemas de archivos marcados para revisión (1) en /etc/fstab: **-A**
- Indicador de proceso: **-C**
- Salida detallada: **-V**
- Simulación de lo que haría fsck: **-N**
- Especificar el tipo de sistema de archivo: **-t** <*fs_type*> (**-t no** <*fs_type*> para revisar todos los filesystem excepto los que sean del tipo especificado)
- Podemos pasar opciones especificas que *fsck* no entiende pero son válidas para un sistema de archivo concreto, entre ellas:

[Opciones-específicas]:

- Realizar una comprobación automática: **-a** o **-p**
- Realizar comprobación interactiva: **-r**
- Realizar una comprobación forzosa: **-f**
- Realizar ejecuciones de *fsck* en dispositivos en serie en vez de en paralelo: **-s**
- No comprobar sistemas de archivos montados y retornar 0 como código de salida: **-M**
- Escapar la comprobación del sistema de archivos ráiz (/) cuando se especifica la opción -A: **-R**

GESTIÓN DE DISPOSITIVOS

COMANDOS SISTEMA DE ARCHIVOS: REPARACIÓN Y COMPROBACIÓN

e2fsck: Usado para realizar comprobaciones en los sistemas de archivos ext2, ext3 y ext4. No es aconsejable usar el comando mientras los sistemas de archivos están montados, como excepción podemos usarlo si pasamos la opción **-n** no acompañada de **-l, -L** o **-c**

#e2fsck [opciones] dispositivo

[Opciones]:

- Corregir de forma automática un sistema de archivos siempre y cuando no se necesite la intervención del administrador de sistemas. Esta opción no podrá usarse junto a -n o -y: **-p** o **-a** (en desuso)

- Usar un superbloque diferente, útil cuando el primer bloque del dispositivo está dañado: **-b** <superbloque>
- Forzar a *e2fsck* a buscar un superbloque de tamaño específico: **-B** <tamaño>
- Localizar bloques defectuosos y añadirlos a la lista de bloques erróneos para que no sean reusados en nuevos archivos o directorios. Podemos especificar esta opción 2 veces para realizar una prueba de lectura-escritura no destructiva sobre el bloque: **-c**
- Abrir el sistema de archivo en modo de solo lectura y asumir como NO todas las preguntas interactivas: **-n**
- Conservar los bloques defectuosos en su lista de bloques malos y añadir a esta lista nuevos bloques defectuosos (hay que acompañarla con la opción -c): **-k**

GESTIÓN DE DISPOSITIVOS

COMANDOS SISTEMA DE ARCHIVOS: REPARACIÓN Y COMPROBACIÓN

xfs_info: Muestra información técnica sobre el filesystem. Es necesario que este esté montado (a diferencia de otras tantas herramientas de bajo nivel).

xfs_ metadump: Herramienta de depuración para sistemas de archivos XFS. Copia la información de un sistema de archivos XFS a un archivo, para ser analizado posteriormente.

#xfs_metadump [opciones] dispositivo archivo_dump

Nota: xfs_metadump no crea copias de seguridad!!

[Opciones]:

- Mostrar el progreso del volcado: **-g**
- Ignorar las lecturas de errores y copiar los metadatos que son accesibles unicamente: **-e**
- Especificar que el filesystem a procesar es guardado en un archivo regular: **-f**
- Mostrar por pantalla los errores encontrados: **-w**

GESTIÓN DE DISPOSITIVOS

COMANDOS SISTEMA DE ARCHIVOS: REPARACIÓN Y COMPROBACIÓN

xfs_admin: Es el equivalente en “bruto” a *tune2fs* pero para sistemas de archivos XFS. Permite cambiar parámetros de configuración del filesystem, para ello xfs_admin hace uso del comando xfs_db.

#xfs_admin [opciones] dispositivo

[Opciones]:

- Habilitar la versión 2 de respaldo de transacciones: **-j**
- Obtener el nombre y UUID del sistema de archivos: **-l** y **-u** respectivamente
  Nota: Para tal fin también podemos usar el comando **blkid**
- Definir la etiqueta o UUID del sistema de archivo: **-L** <Etiqueta> o **-U** <UUID>
- **Nota:** La etiqueta tiene un máximo de 12 caracteres y el UUID puede utilizarse el que tiene antes de ser formateado o modificado el filesystem o bien utilizar el argumento “generate” como valor de UUID de modo que xfs_admin genere uno nuevo

Nota: Los sistemas de archivos deben de ser desmontado antes de ejecutar cualquier modificación

GESTIÓN DE DISPOSITIVOS

COMANDOS SISTEMA DE ARCHIVOS: REPARACIÓN Y COMPROBACIÓN

xfs_ncheck: Provee una lista de inodos de indice y rutas de archivos si se emplea sin opciones

xfs_check: Comprueba si un sistema de ficheros es consistente.

#xfs_check [opciones] dispositivo

[Opciones]:

- Especificar que se trata de una imagen de archivo regular **-f** <archivo.img>
- Especificar la ruta del log en un filesystem externo: **-l** <log-file>
- Mostrar solo aquellos errores de suma importancia: **-s**

GESTIÓN DE DISPOSITIVOS

COMANDOS SISTEMA DE ARCHIVOS: REPARACIÓN Y COMPROBACIÓN

xfs_repair: Repara un sistema de archivo corrupto. El sistema de archivo debe ser desmontado.

#xfs_repair [opciones] dispositivo

[Opciones]:

- No reparar, solo indicar que es lo que anda mal: **-n**
- Desactivar la obtención previa de inodos de indices y directorio. Útil si la ejecución de *xfs_repair* se atasca o deja de responder: **-P**
- Anular la cantidad de memoria RAM máxima que usa *xfs_repair* (un 75% por defecto): **-m** <maxmem>
- Permitir que *xfs_repair* repare un sistema de archivos montado en modo de solo lectura, típico para reparar el directorio raíz. Hay que reiniciar una vez terminado: **-d**

GESTIÓN DE DISPOSITIVOS

COMANDOS SISTEMA DE ARCHIVOS: CONTROL

df: Informa sobre el uso del espacio de los diferentes sistemas de archivos y dispositivos montados en el sistema.

#df [opciones] [dispositivo]

[Opciones]:

- Incluir todos los pseudosistemas de archivos con tamaño 0 (/proc, /sys, /proc/bus/usb, etc..): **-a** (–all)
- Agregar una letra como sufijo para indicar la unidad de medida (por ejemplo K para Kb): **-h**
- Mostrar el espacio en Kb: **-k**
- Mostrar el espacio en Mb: **-m**
- En vez de mostrar el uso de espacio, mostrar los nodos de indices: **-i** (–inodes)
- Omitir los sistemas de archivos en red, es decir solo muestra los locales: **-l** (–local)
- Hacer una llamada a sync antes de desplegar la información: **–sync**
- Mostrar el tipo de sistema de archivo: **-T**
- Mostrar solo aquellos dispositivo que tengan un cierto sistema de archivo: **-t** <fs_type> (–type=fs_type)
- Mostrar solo los dispositivos que no tengan el tipo de sistema de archivo especificado: **-x** <fs_type> (–exclude=fs_type)

Nota: Está opción es útil para sistemas de archivos que crean un número fijo de nodos de indices como ext2, ext3, ext4, XFS y otros sistemas, pero no para sistemas como Reiserfs, que crean nodos dinámicamente.

GESTIÓN DE DISPOSITIVOS

COMANDOS SISTEMA DE ARCHIVOS: CONTROL

blkid: Mas que un comando para monitorizar el uso del disco, es un comando que nos proporcionará una buena información acerca del disco como por ejemplo: el nombre del dispositivo de bloques, el UUID de los dispositivos, etiquetas y sistemas de archivos.

$ sudo blkid [opciones] [dispositivo]

Opciones:

•Mostrar con un formato diferente: **-o** <formato>

Formatos:

– **list** : Muestra el dispositivo, tipo de sistema de archivos, etiqueta, punto de montaje y UUID

– **device**: Muestra solo los dispositivos (/dev/sda1, /dev/sdb1…)

– **export**: Recomendable para redireccionar la salida a un archivo. Muestra la etiqueta, el la siguiente línea el UUID y en la última línea (para ese dispositivo) el tipo de sistema de archivos. Los dispositivos están separados por una línea en blanco.

– **value**: Igual que export pero muestra solo los valores. Es decir omite el LABE=, UUID=,etc… Además no se separan los dispositivos con líneas en blanco

– **full**: Este es el formato por defecto

/dev/sda1: LABEL="MintCinn" UUID="6324422a8-5e6c-4bd3-b7a3-f6d22906b9f9" TYPE="xfs"

- Mostrar todos los nombres de sistemas de archivos y RAID conocidos: **-k**
- Mostrar solo determinados tags (LABEL, UUID y TYPE) junto con el dispositivo: **-s** <tag>
- Mostrar el dispositivo que tiene una etiqueta determinada: **-L** <etiqueta>
- Mostrar el dispositivo que tiene un UUID determinado: **-U** <uuid>

GESTIÓN DE DISPOSITIVOS

LA SWAP

En un entorno de 32 bits, un proceso puede acceder a 4GB de espacio de memoria, como máximo. En un entorno de 64 bits, no hay limite de 4 GB, pero el tamaño de la memoria continua siendo fijo.

TAMAÑO ÓPTIMO

- Si la RAM tiene menos de 512 MB, la swap debe ser el doble.
- Si la RAM tiene entre 1 y 4 GB, la swap debe tener el tamaño de la RAM
- Si la RAM supera los 4 GB, la swap debe tener 4 GB más o menos

CREAR PARTICION DE SWAP

Mkswap: Para preparar la partición que se va a recibir de la swap

#mkswap /dev/sda5

Swapon: Activar la swap, sin tener que iniciar el sistema

Swapoff: Desactiva una zona de swap

/proc/swaps: Refleja el estado actual de las zonas de swap activas.

GESTIÓN DE DISPOSITIVOS

CUOTAS DE DISCO

Las **cuotas** permiten poner limites al uso de espacio en los sistemas de archivos. Estos límites son de dos tipos:

- **Inodos**: Limita el numero de archivos
- **Bloques**: Limita el número de bloques

Se implementan las cuotas por sistema de archivos individual. Se puede gestionar cada usuario de manera totalmente independiente. Ocurre lo mismo con los grupos. XFS permite gestionar las cuotas por directorios. Podemos configurar dos tipos de límites.

- Límite duro: Cantidad máxima que el usuario o grupo no podrá superar en ningún caso.
- Límite suave: cantidad máxima que el usuario o grupo no podrá superar de manera temporal hasta que se supere el tiempo de gracia. Pase lo que pase, el usuario no podrá nunca superar el límite duro.
- Periodo de gracia: Durante este tiempo el usuario puede continuar trabajando en el sistema. El objetivo es que vuelva al Limite suave. Una vez superado este tiempo, el límite se convierte en límite duro.

GESTIÓN DE DISPOSITIVOS

CUOTAS DE DISCO

Todo debe hacerse como root, y lo primero que haremos es editar el archivo "/etc/fstab" y añadiremos "usrquota" o "grpquota", dependiendo si se desea cuotas por usuario o grupos, o incluso ambas.

#> vi /etc/fstab

```
#> vi /etc/fstab
/dev/sda2  /           ext3    noatime         1       1
/dev/sda1  /boot       ext3    noatime         1       2
/dev/sda3  /home       ext3    noatime         1       2
....
(Añadimos en la cuarta columna el tipo de cuotas que deseamos)
/dev/sda2  /           ext3    noatime         1       1
/dev/sda1  /boot       ext3    noatime         1       2
/dev/sda3  /home       ext3    noatime,usrquota,grpquota       1       2
...
```

GESTIÓN DE DISPOSITIVOS

CUOTAS DE DISCO

Algo similar a lo anterior deberá tener tu archivo de configuración, y como ya se indicó solo agregamos el soporte para cuotas en el sistema de archivos que nos interese. Lo anterior por si solo, es obvio que no hace nada, habría que reiniciar el sistema para que se apliquen los cambios pero realmente no es necesario, lo siguiente re-monta el sistema de archivos "/home":

```
#> mount -o remount /home
#> mount
/dev/sda1 on /boot type ext3 (rw,noatime)
/dev/sda2 on / type ext3 (rw,noatime)
/dev/sda3 on /home type ext3 (rw,noatime,usrquota,grpquota)
none on /proc type proc (rw)
```

GESTIÓN DE DISPOSITIVOS

CUOTAS DE DISCO

quotacheck : Este comando crea, verifica o repara el control de cuotas en los sistemas que lo soporten, en este caso creara el soporte:

#> quotacheck -ugmv /home

Pues el sistema esta listo para manipular cuotas de usuario, esto lo podemos comprobar porque en la raíz del sistema de archivos soportado con cuotas deben existir los archivos "aquota.user" y "aquota.group" que son binarios, no trates de modificarlos o manipularlos:

```
#> cd /home
#> ls -l
total 72
-rw-------  1 root   root    8192 2008-05-17 21:38 aquota.group
-rw-------  1 root   root    8192 2008-05-17 21:38 aquota.user
drwx--x--x  4 user1  user1   4096 2008-05-12 16:13 user1/
drwx--x--x  4 user2  user2   4096 2008-05-12 16:13 user2/
drwx--x--x  3 user3  user3   4096 2008-05-05 12:01 user3/
drwx--x--x  3 user4  user4   4096 2008-05-05 12:01 user4/
```

GESTIÓN DE DISPOSITIVOS

CUOTAS DE DISCO

Quotaon: lo anterior deja listo el sistema para el soporte de cuotas pero estás siguen sin ser activadas se requiere activar el soporte de cuotas, para lo cual invocamos el comando quotaon.

```
#> quotaon -ugv /home
/dev/sda3 [/home]: group quotas turned on
/dev/sda3 [/home]: user quotas turned on
```

Quotaoff: Cuando por alguna razón sea necesario desactivar las cuotas, entonces utiliza la contraparte, que es el comando quotaoff:

```
#> quotaoff -v /home
/dev/sda3 [/home]: group quotas turned off
/dev/sda3 [/home]: user quotas turned off
```

GESTIÓN DE DISPOSITIVOS

CUOTAS DE DISCO : Cuota a usuarios

Ahora hay que aplicar la cuota por usuario, aunque el sistema de archivos ya soporta cuotas y están habilitadas, por defecto ningún usuario tiene establecidas cuotas. Asi que para iniciar habrá que administrar cada usuario a través del comando **edquota**, que abrirá el editor de texto que se tenga por defecto y mostrará lo siguiente:

```
#> edquota -u user1
Disk quotas for user user1 (uid 502):
  Filesystem                   blocks       soft       hard     inodes     soft     hard
  /dev/sda3                        56          0          0         14        0        0
```

el comportamiento por default es modificar cuotas para ese usuario en todos los sistemas de archivos que tengan activo el control de cuotas (quotaon). Si se desea control de cuotas para un filesystem en específico entonces se agrega la opción -f:

```
#> edquota -u user1 -f /home
(solo aplica la cuota en el sistema de archivos indicado)
```

GESTIÓN DE DISPOSITIVOS

CUOTAS DE DISCO : Cuota a usuarios

Como usuario administrador 'root' puedes ver el uso de cuotas de cualquier usuario, ya sea individualmente o por medio de un reporte global. Por usuario o individualmente se usa el comando **quota, estando como "root“.**

```
#> quota -u user1
Disk quotas for user user1 (uid 502):
     Filesystem  blocks   quota   limit   grace   files   quota   limit   grace
      /dev/sda3      56      70     100              14       0       0
```

Como usuario individual del sistema, puedes observar tus cuotas con el mismo comando quota, sin argumentos.
Ahora bien, si se desea un reporte global de las cuotas de todos los usuarios o por grupos, siendo "root" utiliza el comando **repquota**

```
#> repquota /home
*** Report for user quotas on device /dev/sda3
Block grace time: 7days; Inode grace time: 7days
                        Block limits                File limits
User            used    soft    hard  grace    used  soft  hard  grace
----------------------------------------------------------------------
root      --  184280       0       0              11     0     0
sergio    -- 42579852       0 50000000          34902     0     0
user1     --      56      70     100              14     0     0
```

GESTIÓN DE DISPOSITIVOS

CUOTAS DE DISCO : Cuota a usuarios

Usaremos entonces la opción *-p* (protptype) para hacer duplicados a partir del ya establecido.

```
#> edquota -p user1 user2
```

Con lo anterior "copias" la información de límites de cuotas del "user1" al "user2", no hay límite de cuantos usuarios puedes colocar como argumentos asi que lo siguiente es válido

GESTIÓN DE DISPOSITIVOS

LOS PERMISOS DE ACCESO : PERMISOS Y USUARIOS

Podemos saber quien es el propietario de un archivo con el comando *$ls -l* (list), que genera un listado largo, que incluye la información de propiedad y permisos

$ls –l

```
-rwxrwxrwx 1 nebul4ck tripas 5059518 dic  6  2013 1. Virginia Alexandre - Lady Marmalade.mp3
-rwxrw-r-- 1 nebul4ck tripas     432 ene 18 14:43 abcd.sh
-rw-r--r-- 1 nebul4ck tripas   10240 ene 18 04:05 arch.tar
drwxr-xr-x 2 nebul4ck tripas      80 ene 19 10:46 descom
lrwxrwxrwx 1 nebul4ck tripas      23 ene 16 21:35 logo -> dire1/este/logo-web.png
```

El superusuario puede cambiar el propietario de un archivo o directorio mediante el comando chown.
La sintaxis del comando **chown** es:

```
$sudo chown [opciones] nuevpropietario:nuevogrupo <archivos a modificar>
```

GESTIÓN DE DISPOSITIVOS

LOS PERMISOS DE ACCESO : PERMISOS Y USUARIOS

Los permisos de los archivos o directorios los pueden cambiar el superusuario y el propietario del archivo, se cambian con el comando **chmod** y su sintaxis es

```
$sudo chmod [opciones] modo <nombre archivo>
```

Modo octal: *$sudo chmod **644** archivo.txt*

Un **modo simbólico** consta de tres componentes:

1. Código que indica el conjunto de permisos que desea modificar: propietario (**u**), el grupo (**g**), globales (**o**) o todos (**a**)
2. Símbolo que indica la acción a realizar; añadir (**+**), quitar(**–**) o definir el valor indicado (**=**)
3. Por último, el código que especifica cuál debería ser el permiso: rw, rwx, rx, etc…

GESTIÓN DE DISPOSITIVOS

LOS PERMISOS DE ACCESO : PERMISOS Y USUARIOS

Ejemplo 1: Dar permisos **rw** al propietario, **r** al grupo y quitar **w** al grupo:

$sudo chmod u+rw,g+r,g-w report.txt

Nota: Si el grupo anteriormente ya tenía permisos de escritura y el resto de usuarios por ejemplo permisos de lectura, este comando no anulará lo que ya había. Para especificar el valor concreto, anulando posibles permisos dado anteriormente usamos (=)

Ejemplo 2: Dar permisos concretos (anula lo que ya había):

$sudo chmod u=rwx,g=rx,o=r archivo.txt

GESTIÓN DE DISPOSITIVOS

LOS PERMISOS DE ACCESO : PERMISOS Y USUARIOS

chown: Cambia el propietario y el grupo de un archivo

chown [opciones] propietario:grupo archivo

Opciones:

- No imprimir mensajes de error: **-f**
- Modificar de forma recursiva: **-R**

chgrp: Cambia el grupo propietario de un archivo

chgrp [opciones] grupo <archivos>

Opciones: Acepta las mismas que *chown*, destacamos **-f** y **-R**

Opciones: Igualmente destacamos **-f** y **-R**

GESTIÓN DE DISPOSITIVOS

LOS PERMISOS DE ACCESO : PERMISOS Y USUARIOS

Umask y el grupo por defecto

Podemos definir el modo y el grupo de un archivo por defecto. Cuando un usuario crea un archivo, dicho archivo tiene un propietario predeterminado (por defecto su creador y el grupo principal de este) y unos permisos predeterminados que vienen definidos por la máscara del usuario. Esta máscara se puede modificar con el comando **umask** (*umask* es una operación a nivel de bits).

El comando **umask** recibe como entrada un valor octal que representa los bits a eliminar de los permisos 777 de los directorios, o 666 de los permisos de archivos.
Pongamos un ejemplo: Supongamos que tenemos una umask de 022 o 002 (suelen ser las umask por defecto), si creamos un directorio este se creará con los permisos 755 o 775 (777 – 022 y 777- 002), si creamos un archivo los permisos serán 644 y 664.

GESTIÓN DE DISPOSITIVOS

LOS PERMISOS DE ACCESO : PERMISOS Y USUARIOS

Los usuarios normales pueden introducir el comando ***umask*** para cambiar los permisos de los nuevos archivos que creen. El superusuario también puede cambiar la configuración por defecto para todos los usuarios, modificando un archivo de configuración del sistema (*/etc/profile*).

Advertencia: La configuración de umask en el archivo /etc/profile puede ser invalidada por los archivos de configuración de perfil de inicio de cada usuario (~/bash_profile).
Podemos ver la configuración de umask en octal con el comando umask y de forma simbólica con umask -S. Al igual que los modos de los archivos, podemos definir la umask por defecto de forma simbólica, pero de hacerlo así deberemos de indicar los bits que si queremos definir, por ejemplo,

```
$umask u=rw,g=rx,o=rx (644)
```