Particionado

1. Particionado lógico

En este manual consideraremos soporte de almacenamiento a cualquiera de tipo magnético o de memoria como, por ejemplo, discos duros, SSD, tarjeta de memoria, etc. Es decir, todo lo que se pueda parecer a un disco duro según la visión clásica: un espacio de datos que se puede seccionar en varias entidades lógicas e independientes y que dispone cada una de su propio sistema de archivos.

Se puede considerar un disco como una larga banda de espacio de almacenamiento dividida en casillas que pueden contener una cantidad determinada de información. Se puede utilizar el disco tal cual, como espacio de almacenamiento. Nada impide crear un sistema de archivos en un disco sin pasar por la etapa de particionado. Sin embargo, es importante dar una organización lógica a este espacio y a los sistemas de archivos que va a contener, aunque se trate sólo de separar los datos (los archivos de datos) de los tratamientos (los programas que los utilizan y el sistema).

El particionado consiste en una división lógica del disco. Se fracciona el disco físico, real, en varios discos virtuales, lógicos: las particiones. Se ve cada partición como un disco independiente que contiene su propio sistema de archivos.

Existen dos métodos de particionado: MBR, previsto para los equipos basados en BIOS, y GPT, para los equipos basados en UEFI. Estas últimas son también compatibles con el particionado BIOS, pero no al revés.

2. Particionado MBR

a. MBR y BIOS

El método de particionado MBR se llama de esta forma porque la partición principal se ubica en este sector específico del disco. El particionado MBR data de los años 1980, cuando los PC trabajaban en 16 bits, y se le han hecho cuatro apaños, para trabajar en 32 bits. La posición de un bloque se codifica en 32 bits, y el tamaño de un bloque es de 512

bytes. Podemos entonces calcular el tamaño máximo de un disco: 2^{32}*512 bytes: 2 TB. No es posible emplear este método con discos de más de 2 TB. No es posible emplear este método con discos de más de 2 TB. Mucha gente piensa que ha sido estafada al comprar los discos de 3 TB. Pero no, simplemente es que no han prestado atención a los temas de compatibilidad.

Si usted compra un disco duro de más de 2 TB, debe disponer de un equipo compatible y utilizar el modo de particionado GPT.

Este método de particionado sigue siendo el más empleado, pero la disponibilidad de equipos basados en UEFI, los nuevos sistemas operativos y los discos de capacidades superiores a 2 TB lo dejarán poco a poco obsoleto.

El hecho de estar limitado a soportes físicos de 2 TB no significa que el tamaño del sistema de archivos esté limitado en sí. Usted puede extender un sistema de archivos en dos discos de 2 TB para crear uno de 4 TB gestionado por el sistema operativo.

b. MBR

El primer sector es el **MBR**, *Master Boot Record* o zona de arranque. Con un tamaño de 512 bytes, contiene en sus primeros 446 bytes una rutina (un programa) de arranque destinado a ejecutar el sistema operativo desde la partición activa, o desde el gestor de arranque (bootloader). Los 6 últimos bytes de este bloque, opcionales, son usados para firmar el disco con 4 bytes. A continuación, 2 bytes nulos, y más adelante, los 64 bytes que contienen la tabla de las cuatro particiones primarias. El conjunto termina con una firma 0xAA55 en dos bytes.

Formato de un MBR

c. Las particiones

Una partición es una división lógica del disco. Existen tres tipos:

- Las particiones primarias, en un total de cuatro, son las descritas en el MBR.
- Las particiones extendidas (primarias extendidas), una sola por disco (aunque en teoría sea posible crear particiones extendidas dentro de otras particiones extendidas).
- Las particiones lógicas.

Un disco puede contener hasta 63 particiones en IDE, 15 en SCSI (es un límite de la implementación oficial del SCSI) o mediante la librería libata. El límite actual es de 15 particiones para todos los discos con los últimos núcleos y la API libata. Sin embargo, algunas distribuciones permiten utilizar la antigua API (PATA) para volver al antiguo sistema.

Observe que se trata efectivamente de un límite por disco, y no del número total de particiones gestionadas por el sistema.

Para superar el límite de las 15 particiones, es posible utilizar el “device mapper” de Linux, que requiere en particular la gestión del **LVM** (*Logical Volume Management*). El LVM permite agrupar varios discos (Physical Volumes) en una sola unidad (Volume Group) vista por el sistema como un enorme disco único que puede dividir en particiones (Logical Volumes) sin limitación de número. Además, puede añadir discos en el grupo después, aumentar o reducir el tamaño de las particiones sin preocuparse de su ubicación física real...

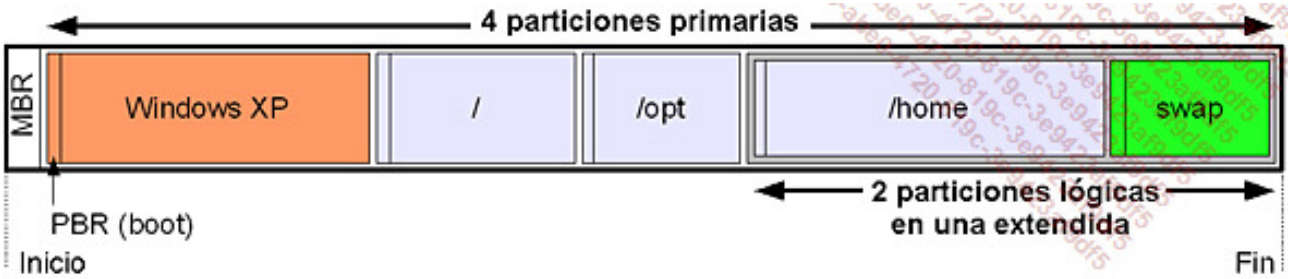
Se numeran las particiones de 1 a n (15 o 63). Una partición con un valor superior o igual a 5 indica que se trata obligatoriamente de una partición lógica. Como sólo puede haber cuatro particiones primarias, se suele crear la última (la 4) como extendida:

- Particiones 1 a 3: primarias

- Partición 4: extendida
- Particiones 5 a n: lógicas

El número de la partición aparece a continuación del nombre del archivo periférico de disco:

- hda1: primera partición primaria del primer disco IDE;
- hdb5: quinta partición, primera partición lógica del segundo disco IDE;
- sda3: tercera partición primaria del primer disco SCSI / libata;
- sdc8: octava partición, o sea, cuarta partición lógica del tercer disco SCSI/libata.



Descripción esquemática de un disco

d. EBR

Como cada partición extendida debe describir las particiones lógicas que contiene, debe disponer también de una tabla de partición. El **EBR** (*Extended Boot Record*) retoma la estructura del MBR, excepto porque sólo hay dos grabaciones posibles en la tabla de las particiones. La primera indica efectivamente la posición y el tamaño de una partición lógica, mientras que la segunda está vacía si es la única partición lógica, o apunta a otro EBR. Por lo tanto, puede haber varios EBR en una partición extendida.

- Los EBR forman una lista encadenada, la segunda entrada de partición apuntando al EBR siguiente.
- Sólo hay una partición lógica descrita por EBR.

e. PBR

El **PBR** (*Partition Boot Record*), también llamado **VBR** (*Volume Boot Record*) o Partition

Boot Sector es el primer sector de cada partición primaria o lógica. Puede contener una rutina de inicio de un sistema operativo, un cargador de inicio o incluso nada si la partición no tiene como misión iniciar un sistema operativo. Cuando el MBR no contiene rutinas, la BIOS intenta iniciar y ejecutar la rutina del PBR de la partición marcada como activa.

f. Tipos de particiones

Cada partición dispone de un tipo que permite determinar su contenido. Es un identificador numérico codificado en un byte generalmente presentado en hexadecimal. Parece importante facilitar una lista aquí para que pueda entender correctamente la finalidad de este identificador.

```
 0  Empty          24  NEC DOS         81  Minix / old Lin bf  Solaris
 1  FAT12          27  Hidden NTFS Win 82  Linux swap / So c1  DRDOS/sec (FAT-
 2  XENIX root     39  Plan 9          83  Linux           c4  DRDOS/sec (FAT-
 3  XENIX usr      3c  PartitionMagic  84  OS/2 hidden or  c6  DRDOS/sec (FAT-
 4  FAT16 <32M     40  Venix 80286     85  Linux extended  c7  Syrinx
 5  Extended       41  PPC PReP Boot   86  NTFS volume set da  Non-FS data
 6  FAT16          42  SFS             87  NTFS volume set db  CP/M / CTOS / .
 7  HPFS/NTFS/exFAT 4d  QNX4.x         88  Linux plaintext de  Dell Utility
 8  AIX            4e  QNX4.x 2nd part 8e  Linux LVM       df  BootIt
 9  AIX bootable   4f  QNX4.x 3rd part 93  Amoeba          e1  DOS access
 a  OS/2 Boot Manag 50  OnTrack DM      94  Amoeba BBT      e3  DOS R/O
 b  W95 FAT32      51  OnTrack DM6 Aux 9f  BSD/OS          e4  SpeedStor
 c  W95 FAT32 (LBA) 52  CP/M           a0  IBM Thinkpad hi ea  Rufus alignment
 e  W95 FAT16 (LBA) 53  OnTrack DM6 Aux a5  FreeBSD        eb  BeOS fs
 f  W95 Ext'd (LBA) 54  OnTrackDM6     a6  OpenBSD         ee  GPT
10  OPUS           55  EZ-Drive       a7  NeXTSTEP        ef  EFI (FAT-12/16/
11  Hidden FAT12   56  Golden Bow      a8  Darwin UFS      f0  Linux/PA-RISC b
12  Compaq diagnost 5c  Priam Edisk    a9  NetBSD          f1  SpeedStor
14  Hidden FAT16 <3 61  SpeedStor      ab  Darwin boot     f4  SpeedStor
16  Hidden FAT16   63  GNU HURD or Sys af  HFS / HFS+      f2  DOS secondary
17  Hidden HPFS/NTF 64  Novell Netware  b7  BSDI fs        fb  VMware VMFS
18  AST SmartSleep  65  Novell Netware  b8  BSDI swap      fc  VMware VMKCORE
1b  Hidden W95 FAT3 70  DiskSecure Mult bb  Boot Wizard hid fd  Linux raid auto
1c  Hidden W95 FAT3 75  PC/IX          bc  Acronis FAT32 L fe  LANstep
1e  Hidden W95 FAT1 80  Old Minix      be  Solaris boot    ff  BBT
```

Como el tipo de partición debería reflejar el sistema de archivos que contiene, una partición de tipo 0x0c debería contener un sistema de archivos de tipo FAT32 LBA (discos grandes). Una partición de tipo 0x83 debería contener un sistema de archivos Linux. Pero ¿cuál? Ya ha visto que existen varios...

Observe el predominio de los tipos de partición para Windows. Windows se basa esencialmente en este tipo para determinar su contenido. Nada impide crear una partición de tipo Linux y colocar en ella un sistema de archivos FAT32. Sin embargo, si lo hace, Windows no reconocerá la partición (considerada de tipo desconocido) y no podrá acceder al contenido.

Linux reconocerá en general (hay una pocas excepciones) el contenido de una partición por el sistema de archivos que reside en ella. Puede crear un sistema de archivos ext4 en una partición de tipo 0x0e y constatar que todo funciona. El tipo 0x83 puede acoger cualquier sistema de archivos Linux: ext2, ext3, ReiserFS, jfs, XFS... Sin embargo, por razones de compatibilidad, tenga cuidado en respetar la asociación tipo de partición <-> sistema de archivos y ser coherente.

3. Particionado GPT

a. GPT y UEFI

GPT significa *GUID Partition Table*, **GUID** significa *Globally Unique Identifier*. Este es un estándar que permite describir la tabla de particiones de un disco duro, que forma parte de las especificaciones de **UEFI***Unified Extensible Firmware Interface*. Para resumir:

- UEFI substituye a la BIOS.
- GPT reemplaza al particionado MBR.

Ha constatado que el diseño de una BIOS y su método de direccionamiento de un disco duro, basado en valores codificados en 32 bits, impiden el uso de discos de más de 2 TB y reducen el número de particiones posibles. La disponibilidad de soportes de almacenamiento de mayores capacidades, el nuevo hardware y los nuevos sistemas operativos han adelantado la necesidad de disponer de un sistema de arranque y de particionado más moderno.

El desarrollo inicial de GPT se remonta a finales de los 90 y lo inició Intel, dentro de EFI. Su evolución a UEFI y su aceptación como estándar para los constructores y fabricantes en 2014 lo convirtió en 2010 en el habitual para todo el hardware (placas base) y sistemas operativos recientes. Todas las placas base basadas en UEFI aceptan GPT.

Por razones de compatibilidad, es posible en ciertos casos emplear GPT con algunas BIOS, especialmente para el empleo de una partición de arranque particular, la **BIOS Boot partition**, vinculada a un gestor de arranque particular, como GRUB2.

El tamaño de dirección de un bloque lógico está codificado en 64 bits. El tamaño de un bloque lógico es al menos de 512 bytes pero puede ser más grande. El tamaño máximo de un disco con un tamaño de bloque de 512 bytes es de 2^{64}-1*512 bytes: 8 ZiB (Zebibyte), o sea 8 mil millones de terabytes.

No contamos más la posición en sectores o bloques, pero en términos de **LBA** (*Logical Block Addressing*, direccionamiento por bloques lógicos). Empleamos entonces LBA 0 para los 512 primeros bytes, LBA 1 para los siguientes 512, y así sucesivamente.

Todas las versiones actuales de las distribuciones Linux soportan GPT.

b. GUID

GUID, identificador global único, es un valor codificado en 128 bits que sirve como identificador único para un componente software. No podemos saber si un GUID es realmente único (en el mundo en todo caso), pero la probabilidad, vistos los valores del conjunto, de que dos valores aleatorios de 128 bits sean iguales en componentes software cercanos o que se espera que trabajen juntos es casi nula.

GUID es equivalente al UUID, nombre con el que es más conocido en Linux y Unix. Observe que en su forma hexadecimal, un GUID se parece a esto: 41649fe8-9341-4bce-977a-687f9da63fdb

Distinguimos:

- Un bloque de cuatro bytes (32 bits).
- Tres bloques de dos bytes (3 veces 1 bits, 48 bits).
- Un bloque de seis bytes (48 bits).

En GPT, el GUID sirve para describir:

- El identificador único del disco.
- El identificador del tipo de partición (equivalente a los tipos de particiones MBR).
- El identificador único de una partición.

c. LBA 0

El **LBA 0**, o sea los primeros 512 bytes del disco, conservan su rol de MBR, pero son conocidos como MBR protector (*Protective MBR*). Su función, dependiente de su estructura, es la de impedir las escrituras en el disco por herramientas que no reconocen el formato GPT. Este MBR describe de hecho una partición, de tipo 0xEE, que abarca el conjunto del disco hasta un máximo de 2 TB, las antiguas herramientas y BIOS no son capaces de ver más allá. Este tipo de partición es desconocido para las herramientas del tipo MBR, no pueden modificar el disco, salvo borrando esta partición.

La presencia de un Protective MBR es la firma de un disco GPT.

Tratándose sin embargo de un MBR, un BIOS reciente puede ser programado para reconocer este tipo de "partición" y utilizar el cargador de arranque (bootloader). En este caso, en Linux con GRUB2, este fin de código debe finalmente cargar el contenido de otra partición especial, la BIOS Boot partition.

d. LBA 1

La cabecera GPT está en la posición LBA 1. Existe una copia en LBA -1, a partir del final del disco. Esta cabecera, firmada 45 46 49 20 50 41 52 54, que se traduce en «PARTE EFI» en ASCII, contiene información sobre la estructura GPT:

- Su propia ubicación y tamaño.
- La ubicación y el tamaño de la cabecera sub (al final del disco).
- Asociada a controlar el dinero.
- Las primeras y últimas direcciones utilizadas para las particiones.
- El GUID del disco.
- La dirección de la tabla de particiones (por lo general LBA 2).

- El número y el tamaño de las particiones descriptores.
- La suma de comprobación de la tabla de particiones.

El conjunto ocupa 92 bytes, los restantes 420 se rellenan con ceros.

La presencia de las sumas de comprobación permite que el firmware UEFI, el gestor de arranque o las herramientas especializadas compruebe el estado de la cabecera y su copia. En caso de error en el LBA 1, LBA -1 puede ser utilizado. En caso de error en ambos, el disco queda inutilizable. Es lo mismo para el control de las tablas de partición.

e. LBA 2 a 33

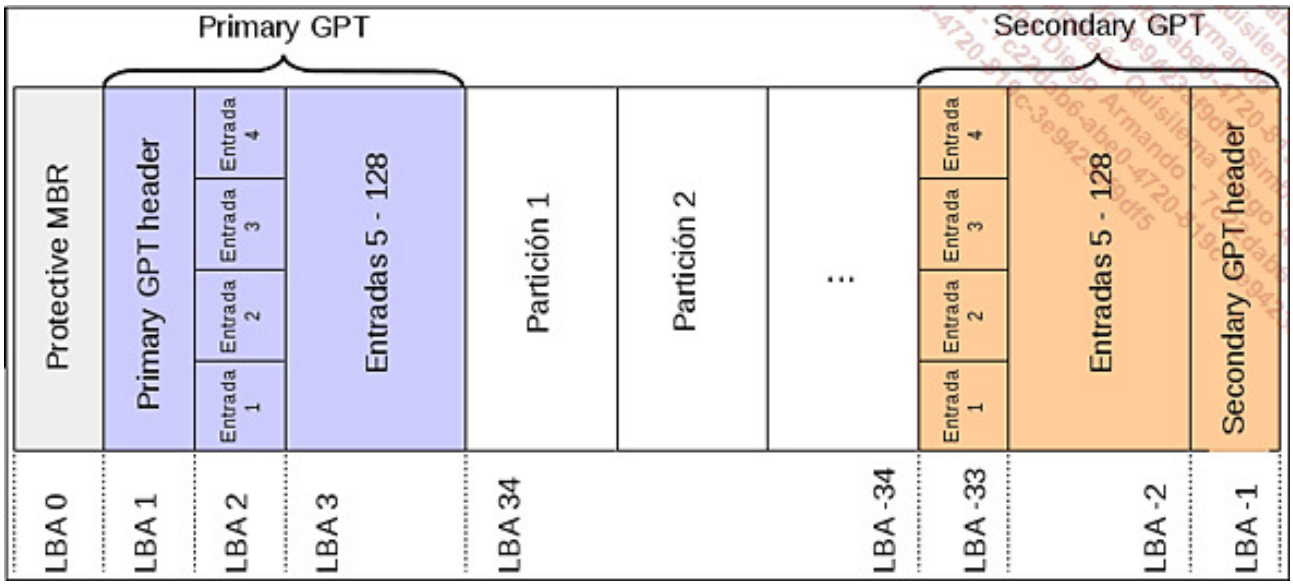
Los LBA 2 a 33 contienen los descriptores de las particiones. Cada partición está descrita en 128 bytes, donde 4 son para LBA. Los 32 LBA permiten describir 128 particiones. Este es el mínimo impuesto por el estándar UEFI, el número y el tamaño de los descriptores de las particiones pueden ser modificados en el LBA 1.

Una copia de seguridad de los descriptores se encuentra en LBA -2 a -33 (fin del disco).

El descriptor de partición está estructurado como sigue:

- el tipo GUID de la partición (16 bytes), el equivalente del tipo de partición MBR,
- el GUID de la partición misma (16 bytes),
- el LBA de inicio de la partición (8 bytes),
- el LBA del final de la partición (8 bytes),
- los atributos (por ejemplo, solo lectura) de la partición (8 bytes),
- el nombre de la partición en formato UTF-16 (2 bytes por carácter, 36 caracteres, 72 bytes).

Así llegamos a un tamaño de 128 bytes.



Estructura de un disco GPT

f. Tipos de particiones

Los tipos de particiones están descritos por los GUID. Al ser el tamaño de los GUID imponente, aquí solo listamos los GUID más frecuentes para las particiones previstas para Linux.

Type	GUID
Linux filesystem data	0FC63DAF-8483-4772-8E79-3D69D8477DE4
RAID partition	A19D880F-05FC-4D3B-A006-743F0F84911E
Swap partition	0657FD6D-A4AB-43C4-84E5-0933C84B4F4F
LVM partition	E6D6D379-F507-44C2-A23C-238F2A3DF928
ESP	C12A7328-F81F-11D2-BA4B-00A0C93EC93B

g. UEFI Boot manager

UEFI reemplaza la BIOS, y está almacenado en el firmware del ordenador (de su tarjeta madre). Su configuración se sitúa en dos emplazamientos:

- La **NVRAM**, *Non Volatile RAM*, equivalente al CMOS de las antiguas BIOS, con la diferencia de que su contenido está normalizado,
- La partición ESP, descrita en la sección siguiente.

Como ya no existe el MBR, UEFI dispone de otro medo para poder identificar los soportes y los sistemas bootables, el equivalente del menú de arranque. Dependiendo de su material, y pulsando las teclas específicas, un menú (en modo texto o gráfico) le permitirá seleccionar el sistema con el que quiera arrancar. La NVRAM contiene una lista de variables, cuyas opciones de boot son las siguientes:

- Qué sistema es el que se está ejecutando actualmente
- El orden de las entradas en el boot
- El timeout antes de arrancar en la entrada por defecto (dependiendo del orden anterior)
- Las listas de las entradas: nombre, soporte (disco), archivo que se tiene que arrancar desde la partición ESP

El comando **efibootmgr** permite modificar el contenido de la variable de la NVRAM. No obstante preste atención, un error podría impedir que el sistema arranque. He aquí un ejemplo:

```
# efibootmgr -v
BootCurrent: 0002
Timeout: 3 seconds
BootOrder: 0001,0002,0000
Boot0000* CD/DVD Drive  BIOS(3,0,00)
Boot0001* Fedora  HD(1,800,61800,6d98f360-cb3e-4727-8fed-5ce0c040365d)
File(\EFI\fedora\grubx64.efi)
Boot0002* SUSE HD(1,800,61800,6d98f360-cb3e-4727-8fed-5ce0c040365d)
File(\EFI\suse\grubx64.efi)
```

h. La partición sistema EFI

Como lo muestra el ejemplo anterior, UEFI va a ejecutar el archivo indicado en las opciones de boot. Estos archivos se almacenan en una partición espacial llamada **ESP**, *EFI System Partition*. Es de tipo 0xEF, dispone de un sistema de archivos FAT, y posee un tamaño mínimo de 100 Mo, aunque podría ser más grande, si esto fuera necesario.

La partición ESP contendrá los datos necesarios para el arranque de uno o varios sistemas operativos, y, posiblemente, la configuración de un cargador de arranque o incluso el mismo cargador de arranque como GRUB. Esta partición se compone de una arborescencia de directorios:

```
+-- EFI
  +-- BOOT
  ¦  +-- BOOTX64.efi
  +-- fedora
  ¦  +-- grubx64.efi
  +-- suse
  ¦  +-- grubx64.efi
  ¦  +-- shimx64.efi
  +-- CLOVER
  ¦  +-- CLOVERX64.efi
```

En el caso de que no exista una configuración específica en la NVRAM, especialmente para los soportes externos (CD, DVD, discos USB y pendrives, etc.), el soporte UEFI ejecutará el archivo BOOTX64.EFI o cualquier otro (único) que se encuentre presente en la carpeta BOOT del periférico ejecutable.

Si la entrada de la variable NVRAM contiene el camino específico de un archivo, UEFI lo ejecutará, como por ejemplo grubx64.efi, que no es otra cosa que el cargador de arranque GRUB2.

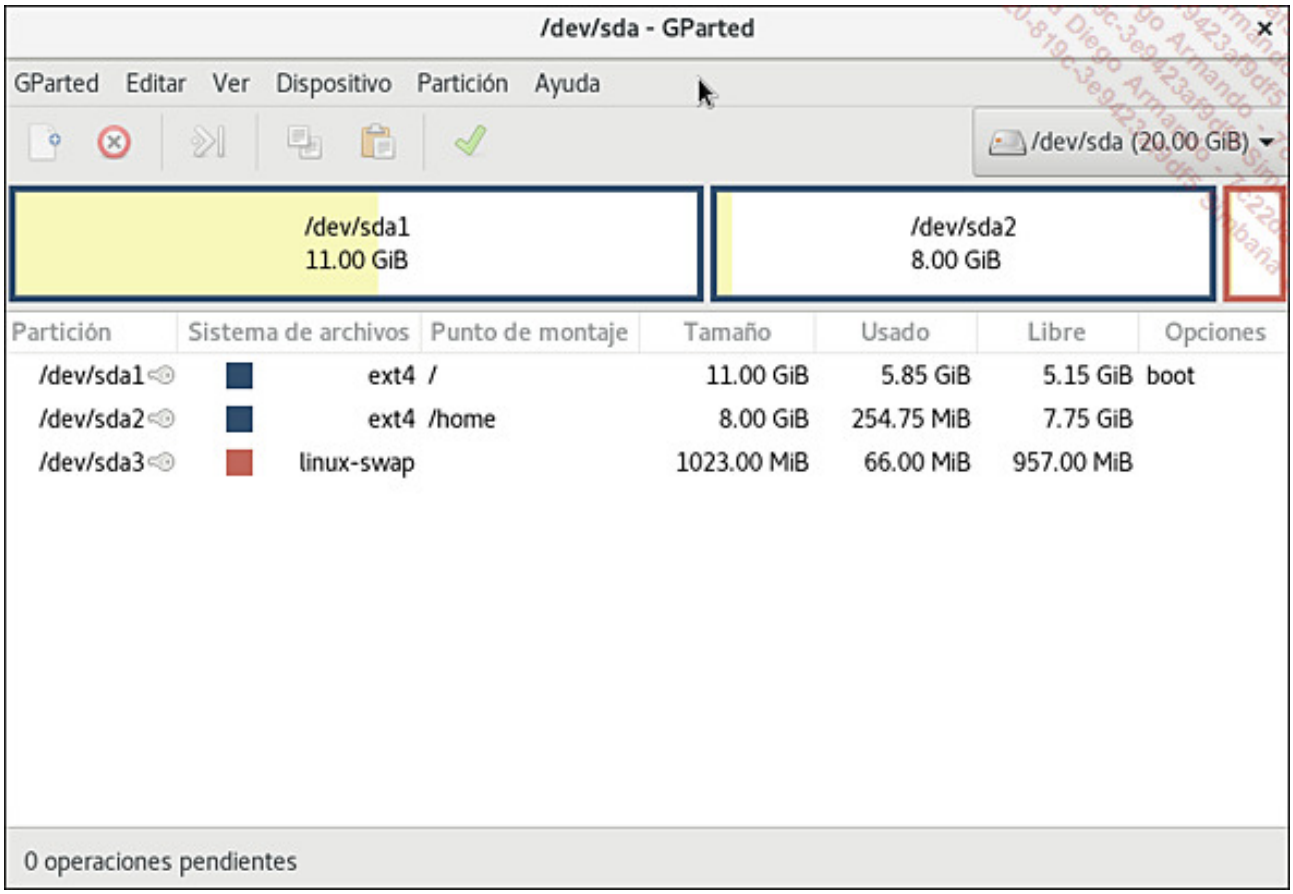
4. Manejar las particiones

a. Herramientas de gestión de particiones

Las herramientas **fdisk**, **cfdisk**, **sfdisk**, **parted** o también **gdisk**, sin contar con las herramientas gráficas disponibles durante la instalación o en los paneles de configuración, permiten manejar las particiones.

- **fdisk** es la más antigua y más utilizada de las herramientas de particionado. No tiene relación con el fdisk de Microsoft. Se basa en menús y atajos textuales.
- **cfdisk** es un poco más «visual» y se utiliza con las flechas direccionales. Permite las mismas operaciones que fdisk y es de fácil manejo.
- **sfdisk** funciona, opcionalmente, de forma interactiva. Es bastante complicada, pero más precisa.
- **parted** permite operaciones muy avanzadas en las particiones, como, por ejemplo, su redimensionamiento. Presenta una interfaz interactiva (intérprete de comandos) que atiende a scripts. Es compatible con GPT. Pero, además, hay en el mercado interfaces gráficas de parted, como qtparted o gparted.
- **gdisk** es equivalente a fdisk para GPT.

En la captura siguiente puede ver a **gparted** en acción.



gparted, un editor de particiones gráfico

La siguiente sección describe las operaciones de particiones de tipo MBR. La sección Manipular las particiones GPT ofrece las diferencias entre GPT mediante el uso de gdisk.

b. Manipular las particiones MBR

Listar

Los administradores e ingenieros de sistemas suelen utilizar la herramienta **fdisk**, que es a la vez la más antigua y más estándar. Hay que tener privilegios de root para poder usar fdisk.

```
fdisk [-l] [disco]
```

Cada parámetro es opcional. Iniciado tal cual, **fdisk** se sitúa en el primer disco del sistema. El parámetro

`-l` permite listar las particiones del disco dado o de todos los discos. La información obtenida es la misma que en modo interactivo con la entrada p (print) del menú.

```
# fdisk -l /dev/sda

Disco /dev/sda: 164,6 Gb, 164696555520 bytes
255 heads, 63 sectors/track, 20023 cylinders
Units = cilindros of 16065 * 512 = 8225280 bytes
Disk identifier: 0x000c02ae

Periférico Arranque  Principio   Fin     Bloques   Id  Sistema
/dev/sda1            1       5222    41945683+   7  HPFS/NTFS
/dev/sda2   *     5223       5288      530145   83  Linux
/dev/sda3         5289      10510    41945715   83  Linux
/dev/sda4        10511      20023    76413172+   f  W95 Extendido (LBA)
/dev/sda5        10511      10772     2104483+  82  Linux swap / Solaris
/dev/sda6        10773      20023    74308626   83  Linux
```

Los campos hablan por sí mismos. Fíjese en que la partición sda4 es la extendida. Tiene como tipo 0x0f, pero cualquier tipo extendido hubiese funcionado: se consideran los tipos 0x05 y 0x85 como idénticos. Sin embargo, es posible que Windows no reconozca estos tipos ni, por lo tanto, las particiones lógicas contenidas en ellos, en particular las de un disco LBA. Por eso, la mayoría de las herramientas de particionado de las distribuciones Linux prefieren el tipo asociado a Windows.

Sólo se tienen en cuenta las operaciones efectuadas con **fdisk** al final, una vez que haya guardado usted sus modificaciones, y no al mismo tiempo. Si piensa que se ha equivocado, no dude en salir sin guardar o en hacer un [Ctrl] **C**. Se perderán sus modificaciones, pero habrá salvado sus particiones.

Lista de particiones

El ejemplo siguiente se basa en un disco reconocido por el sistema como /dev/sdb que no contiene ninguna partición. El objetivo es crear tres particiones: una primaria, una extendida y una lógica.

➔ Ejecute **fdisk** con el disco como argumento. No tenga en cuenta las primeras líneas visualizadas, excepto si indican un error.

```
# fdisk /dev/sdb
...
Comando (m para la ayuda):
```

➔ Verifique primero la existencia de particiones con la tecla p (print), luego [Entrar].

```
Comando (m para la ayuda): p

Disco /dev/sdb: 4026 Mb, 4026531840 bytes
64 heads, 62 sectors/track, 1981 cylinders
Units = cilindros of 3968 * 512 = 2031616 bytes
Disk identifier: 0x0003ed63

Periférico  Arranque   Principio   Fin    Bloques   Id  Sistema
/dev/sdb1   *        1         1981    3930273   c   W95 FAT32 (LBA)
```

Suprimir

Para suprimir una partición, utilice la tecla d (delete); luego, si hay varias particiones, el número de partición (sdbX, siendo X el número). Si hay una única partición, se coge por defecto.

```
Comando (m para la ayuda): d
Partición seleccionada 1
```

Crear

Para crear una partición, utilice la tecla n (new). Luego debe elegir el tipo de partición: primaria o extendida.

```
Comando (m para la ayuda): n
Acción de comando
  e   extendida
  p   partición primaria (1-4)
```

- Para esta primera partición, seleccione una partición primaria con la tecla p (que significa “primary” esta vez).
- Como el MBR contiene cuatro entradas, puede elegir el número de partición que crear. Es perfectamente posible crear la partición sdb2 antes de la sdb1. Aquí, teclee **1**.
- El primer cilindro corresponde a la posición de principio de su partición. Por defecto, fdisk se coloca en el primer cilindro disponible desde el principio del disco. Es perfectamente posible crear particiones primarias en la mitad de un disco. Seleccione aquí el valor por defecto (1) pulsando [Entrar].
- Finalmente, elija el tamaño de la partición. Es preferible utilizar una unidad legible, como los KB o MB. Por ejemplo, para una partición de 1 GB, es decir, de 1024 MB, use **+1024M** y pulse [Entrar]. Ahora se ha definido la partición.

```
Comando (m para la ayuda): n
Acción de comando
  e   extendida
  p   partición primaria (1-4)
p
Número de partición (1-4): 1
Primer cilindro (1-1981, por defecto 1):
Utilización del valor por defecto 1
Último cilindro o + tamaño o +tamañoM o +tamañoK (1-1981, por
defecto 1981): +1024M
```

- Compruebe el estado de la partición (p).

```
Comando (m para la ayuda): p
...
Periférico Arranque   Principio   Fin     Bloques    Id  Sistema
```

```
/dev/sdb1          1       505    1001889    83  Linux
```

Guardar

Salga de fdisk guardando su tabla de las particiones con la tecla w (write). Fdisk escribe la nueva tabla de las particiones en el MBR o los EBR. Puede que el sistema le muestre alertas que nosotros aquí le indicamos en negrita.

```
Comando (m para la ayuda): w
¡Se alteró la tabla de particiones!

Llamada de ioctl() para volver a leer la tabla de particiones.

ADVERTENCIA: la relectura de la tabla de particiones fracasó con
el error 16: Periférico o recurso ocupado.
El kernel va a seguir utilizando la antigua tabla.
Se utilizará la nueva tabla durante el próximo reinicio.
Sincronización de los discos.
```

Este mensaje significa que, como el disco está en uso, Linux no puede volcar la nueva tabla ni, por lo tanto, crear las nuevas particiones. El comando siguiente lo puede confirmar. Observe que la última línea debería mostrarle su nueva partición. Pero no es así.

```
# cat /proc/particiones | grep sdb
8   16   3932160 sdb
```

Forzar la sincronización

Para corregir este último problema y forzar al núcleo a leer de nuevo la tabla de las particiones, tiene a su disposición dos comandos. El primero es **blockdev** con el parámetro `--rereadpt` (re-read partition table).

```
# blockdev --rereadpt /dev/sdb
```

El segundo es **partprobe**, disponible solamente si parted está instalado. Puede probarlo si blockdev no funciona. Por defecto, vuelve a leer las tablas de todas las particiones, pero le puede especificar como argumento un disco en concreto.

```
# partprobe /dev/sdb
```

Compruebe si se reconoce la partición.

```
# cat /proc/particiones | grep sdb
8   16   3932160 sdb
8   17   1001889 sdb1
```

Ahora puede utilizar su nueva partición y añadirle un sistema de archivos. Cree ahora una partición extendida, luego una partición lógica. Observe que se crea una partición extendida o lógica de la misma manera que las otras. Una partición extendida no tiene por qué ser la última de las primarias. También puede ser la segunda, por ejemplo. Si no ha utilizado todo el tamaño para crearla, puede completar más adelante la creación de particiones primarias.

Se pueden usar los comandos de parted como **mkpart**, **rm** o **resize** para trabajar con las particiones. El comando **partx**, y su derivado **kpartx**, permiten notificar en el núcleo las modificaciones realizadas a las tablas de partición.

Modificar el tipo

Como la modificación del tipo de una partición no implica la modificación de su tamaño, puede hacerlo en cualquier momento. Le presentamos a continuación el modo de proceder para pasar del tipo por defecto Linux al tipo FAT para que Windows reconozca la partición.

→ Ejecute **fdisk** y pulse t (tipo).

➔ Seleccione la partición, la 5 para la primera partición lógica.

Puede visualizar todos los tipos pulsando L, que le facilita la misma lista mencionada más arriba. Utilice el tipo c W95 FAT32 (LBA) para estar tranquilo.

➔ Guarde con w.

```
Comando (m para la ayuda): t
Número de partición (1-5): 5
Código Hex (teclear L para listar los códigos): c
Tipo de partición del sistema modificado de 5 a c (W95 FAT32 (LBA))

Comando (m para la ayuda): p

Disco /dev/sdb: 4026 Mb, 4026531840 bytes
64 heads, 62 sectors/track, 1981 cylinders
Units = cilindros of 3968 * 512 = 2031616 bytes
Disk identifier: 0x0003ed63

Periférico Arranque  Principio   Fin    Bloques   Id  Sistema
/dev/sdb1            1       505   1001889    83  Linux
/dev/sdb2          506      1981   2928384     5  Extended
/dev/sdb5          506      1981   2928353     c  W95 FAT32 (LBA)

Comando (m para la ayuda): w
```

Los tipos de particiones más utilizados para Linux son los siguientes:

- **83**: Partición de tipo Linux (datos)
- **82**: Partición de tipo swap
- **fd**: Partición de tipo RAID
- **8e**: Partición de tipo LVM

c. Manipular las particiones GPT

El comando **gdisk**, GPT fdisk es su verdadero nombre, es el equivalente de fdisk pero para un disco GPT. Su uso es cercano al de fdisk, y también su sintáxis. Veamos qué pasa con un disco virgen:

```
# gdisk /dev/sdb
GPT fdisk (gdisk) version 0.8.10
Partition table scan:
 MBR: not present
 BSD: not present
 APM: not present
 GPT: not present

Creating new GPT entries.

Command (? for help):
```

Ya no existen las nociones de particiones primarias, extendidas y lógicas. Las particiones se numeran de 1 a 128. El proceso de creación es idéntico al particionado con fdisk. La única diferencia notable es la lista de códigos y los tipos de particiones. Parece evidente que no va a introducir el GUID, por lo que los tipos propuestos están codificados con cuatro caracteres hexadecimales, y para los más utilizados, son los mismos que en fdisk, completados por dos ceros:

- **8300** : partición de tipo Linux (datos)
- **8200** : partición de tipo swap
- **fd00** : partición de tipo RAID
- **8e00** : partición de tipo LVM

Puede constatar la presencia de los códigos 8301 para una partición prevista para mantener un /home y los códigos ef02 para un tipo de partición boot BIOS, que volveremos a ver cuando abordemos la gestión del arranque.

No se efectúa ninguna modificación hasta que no escriba [w]. La tecla [c] permite cambiar el nombre de la partición.

Puede obtener información detallada de una partición con [i]:

```
Command (? for help): i
Using 1
Partition GUID code: 0FC63DAF-8483-4772-8E79-3D69D8477DE4 (Linux filesystem)
```

Partition unique GUID: 81CDF84E-D87B-48C5-A0C8-13EE07134D56
First sector: 2048 (at 1024.0 KiB)
Last sector: 10487807 (at 5.0 GiB)
Partition size: 10485760 sectors (5.0 GiB)
Attribute flags: 0000000000000000
Partition name: 'root'

Observe: el uso de gdisk en un disco particionado en el método MBR puede ser peligroso: al guardar el disco completo éste se convierte al format GPT, Protective MBR incluido.