Discos y Sistemas de Ficheros

Operaciones de bajo nivel

1. Información

El comando **hdparm** permite efectuar un gran número de operaciones directamente en los discos duros gestionados por la librería libata, o sea todos los discos SATA, ATA (IDE) y SAS. El comando **sdparm** puede hacer más o menos lo mismo para los discos SCSI. Observe que, a pesar de que los nombres de periféricos de la libata sean idénticos a los del SCSI, es más que probable que muchas opciones de configuración de hdparm no funcionen en discos SCSI. Lo mismo vale para sdparm con los discos SATA o IDE. Los ejemplos que damos a continuación se basan en hdparm.

Para obtener información completa relativa a un disco, utilice los parámetros -i o -I. El primero recupera la información, desde el núcleo, que se obtiene en el momento del arranque. El segundo interroga directamente al disco. Es preferible -I porque da una información muy detallada.

```
# hdparm -I /dev/sda

/dev/sda:

ATA device, with non-removable media
    Model Number:       VBOX HARDDISK
    Serial Number:      VB91a2e953-933cdc65
    Firmware Revision:  1.0
Standards:
    Used: ATA/ATAPI-6 published, ANSI INCITS 361-2002
    Supported: 6 5 4
Configuration:
    Logical        max    current
    cylinders    16383    16383
    heads        16    16
    sectors/track    63    63
    --
    CHS current addressable sectors:   16514064
    LBA    user addressable sectors:   63152320
    LBA48  user addressable sectors:   63152320
    Logical/Physical Sector size:           512 bytes
    device size with M = 1024*1024:       30836 MBytes
    device size with M = 1000*1000:       32333 MBytes (32 GB)
    cache/buffer size  = 256 KBytes (type=DualPortCache)
Capabilities:
    LBA, IORDY(cannot be disabled)
    Queue depth: 32
    Standby timer values: spec’d by Vendor, no device specific minimum
    R/W multiple sector transfer: Max = 128        Current = 128
    DMA: mdma0 mdma1 mdma2 udma0 udma1 udma2 udma3 udma4 udma5 *udma6
         Cycle time: min=120ns recommended=120ns
    PIO: pio0 pio1 pio2 pio3 pio4
         Cycle time: no flow control=120ns  IORDY flow control=120ns
Commands/features:
    Enabled    Supported:
       *    Power Management feature set
       *    Write cache
       *    Look-ahead
       *    48-bit Address feature set
```

```
       *    Mandatory FLUSH_CACHE
       *    FLUSH_CACHE_EXT
       *    Gen2 signaling speed (3.0Gb/s)
       *    Native Command Queueing (NCQ)
Checksum: correct
```

2. Modificación de los valores

Se puede modificar varios parámetros de los discos. Sin embargo, ¡cuidado! Algunas opciones de **hdparm** pueden resultar peligrosas tanto para los datos contenidos en el disco como para el propio disco. La mayoría de los parámetros son de lectura y escritura. Si no se especifica ningún valor, **hdparm** muestra el estado del disco (o del bus) para este comando. A continuación le presentamos algunos ejemplos de opciones interesantes.

- -c: anchura del bus de transferencia EIDE en 16 o 32 bits. 0=16, 1=32, 3=32 compatible.
- -d: utilización del DMA. 0=no DMA, 1=DMA activado.
- -X: modifica el modo DMA (mdma0 mdma1 mdma2 udma0 udma1 udma2 udma3 udma4 udma5). Puede utilizar cualquiera de los modos anteriores o valores numéricos: 32+n para los modos mdma (n varía de 0 a 2) y 64+n para los modos udma.
- -C: modo de ahorro de energía en el disco (unknown, active/idle, standby, sleeping). Se puede modificar el estado con -S, -y, -Y y -Z.
- -g: muestra la geometría del disco.
- -M: indica o modifica el estado del Automatic Acoustic Management (AAM). 0=off, 128=quiet y 254=fast. No todos los discos lo soportan.
- -r: pasa el disco en sólo lectura.
- -T: bench de lectura de la caché del disco, ideal para probar la eficacia de transferencia entre Linux y la caché del disco. Hay que volver a ejecutar el comando dos o tres veces.
- -t: bench de lectura del disco, fuera de la caché. Mismas observaciones que para la opción anterior.

Así, el comando siguiente pasa el bus de transferencia a 32 bits, activa el modo DMA en modo Ultra DMA 5 para el disco sda:

```
# hdparm -c1 -d3 -X udma5 /dev/sda
```

Le mostramos a continuación otros ejemplos:

```
# hdparm -c /dev/sda

/dev/sda:
 IO_support    =  0 (default 16-bit)

# hdparm -C /dev/sda

/dev/sda:
 drive state is:  active/idle

# hdparm -g /dev/sda
```

```
/dev/sda:
 geometry       = 3931/255/63, sectors = 63152320, start = 0

# hdparm -T /dev/sda

/dev/sda:
 Timing cached reads:   23868 MB in  2.00 seconds = 11950.45 MB/sec
# hdparm -t /dev/sda

/dev/sda:
 Timing buffered disk reads: 308 MB in  3.02 seconds = 101.87 MB/sec
```

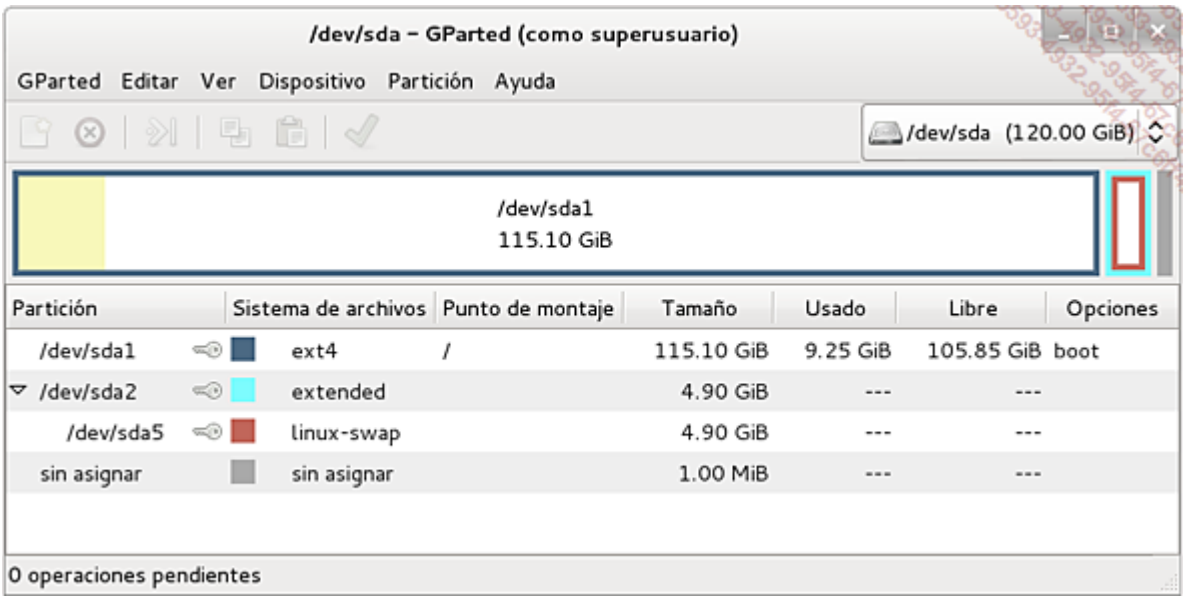
Manejar las particiones

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.

Controlar el sistema de archivos

1. Estadísticas de ocupación

a. Por sistema de archivos

El comando **df** permite obtener estadísticas de ocupación de cada sistema de archivos montado. Sin argumento, **df** facilita información sobre todos los sistemas de archivos. Puede pasar como argumento un periférico montado o un punto de montaje. Si pasa un directorio cualquiera, df da la información del sistema de archivos que contenga este directorio.

```
# df
Sis. de fich.      1K-bloques   Ocupado Disponible Capacidad Montado en
/dev/sda3            41286828   6482952  32706592  17%       /
udev                  1031240       124   1031116   1%       /dev
/dev/sda2              521780     27092    468184   6%       /boot
/dev/sdb1           153834852  49189572  96830864  34%       /home
/dev/sda6            73142560  19150372  50276760  28%       /public
/dev/sdc1           292890560 175894672 116995888  61%       /media/EXTERNO
```

El resultado es explícito. La unidad por defecto es el KB (idéntico al parámetro -k) aunque la norma POSIX define una unidad de bloque en 512 bytes. Puede modificar los parámetros para pedir el resultado en MB (-m).

```
# df -m /home
Sis. de fich.      1M-bloques  Ocupado  Disponible Capacidad Montado en
/dev/sdb1              150230   48043       94557    34%      /home
```

Para que sea más legible, añada el parámetro -h (Human readable).

```
# df -h /home
Sis. de fich.      Tam.   Oc.  Disp. %Oc.  Montado en
/dev/sdb1          147G   47G   93G  34%   /home
```

No confunda este último parámetro con -H, que visualiza el resultado en unidades **SI** (Sistema Internacional).

```
# df -H /home
Sis. de fich.      Tam.   Oc.  Disp.  %Oc.  Montado en
/dev/sdb1          158G   51G  100G   34%   /home
```

Las unidades del SI que definen las unidades de peso y medida se basan en potencias de 10. Es sencillo memorizar que 1 Kg es igual a 103 gramos, o sea, 1000 gramos. Por lo tanto, 1 KB equivale a 103 bytes, o sea 1000 bytes... ¿No está de acuerdo? Un ordenador no trabaja con

potencias de 10, sino de 2. A nivel físico, 1 KB equivale 210 bytes, o sea 1024 bytes; 1 MB vale 220 bytes, y así sucesivamente. Este método se llama método tradicional. El sistema internacional prefiere emplear los términos kibibyte (kilo Binario, Kib), mebibytes (Meb) y gibibytes (Gib) para las representaciones binarias, y KB, MB y GB para las potencias de 10, como en el caso de los metros y los gramos. Ahora entiende por qué un disco de 160 GB se corresponde en realidad a 152,5 Gib. En cierto modo, nos dejamos engañar de manera legal y oficial.

La -T añade la visualización del tipo de sistema de archivos.

```
# df -T /home
Sis. de fich. Tipo   1K-bloques  Ocupado Disponible  Capacidad  Montado en
/dev/sdb1     ext4   153834852  49197688  96822748   34%        /home
```

El comando **df** permite también facilitar estadísticas para el uso de los inodos. Puede combinar los parámetros -i y -h.

```
# df -i /home
Sis. de fich.         Inodos   IUtil.  ILib.   %IUti. Montado en
/dev/sdb1            19546112  86016  19460096    1%  /home
# df -ih /home
Sis. de fich.         Inodos   IUtil.  ILib.   %IUti. Montado en
/dev/sdb1               19M     84K    19M       1%  /home
```

b. Por estructura

El comando **du** (disk usage) facilita la información relativa al espacio ocupado por una estructura (un directorio y todo su contenido). Si no se especifica nada, se utiliza el directorio corriente. Los parámetros -k (Kb) y -m (MB) determinan la unidad. Se facilita el tamaño para cada elemento (incluso redondeado). El tamaño total de la estructura está en la última línea.

```
# du -m LIBRO_ALGO
1       LIBRO_ALGO/BACKUP/capítulo7/codigo_java
2       LIBRO_ALGO/BACKUP/capítulo7
1       LIBRO_ALGO/BACKUP/Introducción
1       LIBRO_ALGO/BACKUP/capítulo4/ilustraciones
...
42      LIBRO_ALGO/
```

Para obtener el total, y no todos los detalles, utilice -s.

```
# du -ks LIBRO_ALGO
42696 LIBRO_ALGO/
```

Observe que no se limita **du** a un único sistema de archivos y sigue calculando si encuentra un punto de montaje en la estructura que analiza. Si quiere limitar el cálculo al sistema de archivos corriente sin entrar en los puntos de montaje presentes en la estructura, especifique -x.

```
# du -msx /
1064 /
```

2. Comprobar, ajustar y arreglar

a. fsck

El comando **fsck** permite comprobar y arreglar un sistema de archivos.

```
fsck -t typefs periférico
```

El sistema de archivos que se quiere comprobar o arreglar no debería estar montado, o, como mucho, montado en modo de sólo lectura.

De la misma forma que **mkfs**, **fsck** invoca a otro comando teniendo en cuenta el tipo del sistema de archivos para comprobar: esos otros comandos más especializados son fsck.ext2, fsck.ext3, etc. Cada uno puede presentar opciones particulares. Si **fsck** no reconoce la opción que se le proporciona, la transmite al programa correspondiente. Si no indica un tipo, **fsck** intenta determinarlo por sí mismo.

Para este ejemplo, se pasa el parámetro -f a fsck para forzar la comprobación (no ha sido posible producir una corrupción), así como el parámetro -V para facilitar todos los detalles.

```
# fsck -fV /dev/sda2
fsck 1.40.2 (12-Jul-2007)
e2fsck 1.40.2 (12-Jul-2007)
Paso 1: verificación de los i-nodos, de los bloques y de los tamaños
Paso 2: verificación de la estructura de los directorios
Paso 3: verificación de la conectividad de los directorios
Paso 4: verificación de los contadores de referencia
Paso 5: verificación de la información del sumario del grupo

     42 inodes used (0.06%)
      1 non-contiguous inode (2.4%)
        # of inodes with ind/dind/tind blocks: 10/1/0
   8864 blocks used (6.69%)
      0 bad blocks
      1 large file

     27 regular files
      3 directories
      0 character device files
      0 block device files
      0 fifos
      0 links
      3 symbolic links (3 fast symbolic links)
      0 sockets
--------
     33 files
```

Cuando el sistema de archivos está dañado, **fsck** inicia una batería de preguntas por cada acción necesaria. Puede pasar el parámetro -p para intentar una reparación automática, o también -y para forzar las respuestas a sí.

Durante el inicio del sistema, éste comprueba desde hace cuánto tiempo, o después de cuántos montajes, no se ha comprobado el sistema de archivos. Si el intervalo de tiempo es demasiado grande, ejecutará un **fsck** en el sistema de archivos correspondientes. Se pueden modificar los intervalos mediante el comando **tune2fs**.

Una partición de tipo BTRFS no necesita fsck, emplea **btrfsck** o **btrfs check**. El comando fsck.btrfs sólo existe para indicar el uso de los dos primeros.

Los permisos de acceso

1. Los permisos básicos

a. Permisos y usuarios

El papel de un sistema operativo es también el de asegurar la integridad y el acceso a los datos, lo que es posible gracias a un sistema de permisos. A cada archivo o directorio se le asignan unos privilegios que le son propios, así como autorizaciones de acceso individuales. Al intentar acceder, el sistema comprueba si está autorizado.

Cuando el administrador crea un usuario, le asigna un **UID** (User Identification) único. Los usuarios quedan definidos en el archivo /etc/passwd. Del mismo modo, cada usuario se integra en, al menos, un grupo (grupo primario). Todos éstos tienen un identificador único, el **GID** (Group Identification) y están definidos en el archivo /etc/group.

El comando **id** permite obtener esta información. A nivel interno, el sistema trabaja únicamente con los UID y GID, y no con los propios nombres.

```
$ id
uid=1000(seb) gid=100(users) grupos=7(lp),16(dialout),33(video),
100(users)
```

Se asocian un UID y un GID a cada archivo (inodo) que define su propietario y su grupo con privilegios. Usted asigna permisos al propietario, al grupo con privilegios y al resto de la gente. Se distinguen tres casos:

- UID del usuario idéntico al UID definido para el archivo. Este usuario es propietario del archivo.
- Los UID son diferentes: el sistema comprueba si el GID del usuario es idéntico al GID del archivo. Si es el caso, el usuario pertenece al grupo con privilegios del archivo.
- En los otros casos (ninguna correspondencia): se trata del resto de la gente (others), ni es el propietario, ni un miembro del grupo con privilegios.

d	rwxr-xr-x	29	seb	users	4096	Mar 15 22:13	Documentos

En esta línea de la tabla, el directorio Documentos pertenece al usuario seb y al grupo users, y posee los permisos rwxr-xr-x.

b. Significado

Permiso	Significado
General	
r	Readable (lectura).
w	Writable (escritura).
x	Executable (ejecutable como programa).
Archivo normal	
r	Se puede leer el contenido del archivo, cargarlo en memoria, listarlo y copiarlo.
w	Se puede modificar el contenido del archivo. Se puede escribir dentro. Modificar el contenido no significa poder eliminar el archivo (ver permisos en directorio).
x	Se puede ejecutar el archivo desde la línea de comandos si se trata de un programa binario (compilado) o de un script (shell, perl...).
Directorio	
r	Se pueden listar (leer) los elementos del directorio (catálogo). Sin esta autorización, ls y los criterios de filtro en el directorio y su contenido no serían posibles. No obstante, puede seguir accediendo a un archivo si conoce su ruta de acceso.
w	Se pueden modificar los elementos del directorio (catálogo), y es posible crear, volver a nombrar y suprimir archivos en este directorio.**Es este permiso el que controla el permiso de eliminación de un archivo**.
x	Se puede acceder al catálogo por CD y se puede listar. Sin esta autorización, es imposible acceder al directorio y actuar en su contenido, que pasa a estar cerrado.

Así, para un archivo:

rwx	r-x	r--
Permisos para el propietario de lectura, escritura y ejecución.	Permiso para los miembros del grupo de lectura y ejecución.	Permisos para el resto del mundo de lectura únicamente.

2. Modificación de los permisos

Cuando se crea, un archivo o un directorio dispone de permisos por defecto. Utilice el comando **chmod** (change mode) para modificar los permisos en un archivo o un directorio. Existen dos métodos para modificar estos permisos: mediante símbolos o mediante un sistema octal de representación de permisos. Sólo el propietario de un archivo puede modificar sus permisos (además del administrador del sistema). El parámetro -R cambia los permisos de manera recursiva.

a. Mediante símbolos

La sintaxis es la siguiente:

```
chmod modificaciones Fic1 [Fic2...]
```

Si hay que modificar los permisos del propietario, utilice el carácter **u**; para los permisos del grupo con permisos, el carácter **g**; para el resto, el carácter **o**, y para todos, el carácter **a**.

Para añadir permisos, se utiliza el carácter +; para retirarlos, el carácter -, y para no tener en cuenta los parámetros anteriores, el carácter =.

Finalmente, ponga el permiso cuyos símbolos son: **r**, **w** o **x**.

Puede separar las modificaciones con comas y acumular varios permisos en un mismo comando.

```
$ ls -l
total 0
-rw-r--r-- 1 seb users 0 mar 21 22:03 fic1
-rw-r--r-- 1 seb users 0 mar 21 22:03 fic2
-rw-r--r-- 1 seb users 0 mar 21 22:03 fic3
$ chmod g+w fic1
$ ls -l fic1
-rw-rw-r-- 1 seb users 0 mar 21 22:03 fic1
$ chmod u=rwx,g=x,o=rw fic2
$ ls -l fic2
-rwx--xrw- 1 seb users 0 mar 21 22:03 fic2
$ chmod o-r fic3
$ ls -l fic3
-rw-r----- 1 seb users 0 mar 21 22:03 fic3
```

Si quiere suprimir todos los permisos, no especifique nada después del signo =:

```
$chmod o=fic2
$ ls -l fic2
-rwx--x--- 1 seb users 0 mar 21 22:03 fic2
```

b. Sistema octal

La sintaxis es idéntica a la de los símbolos. A cada permiso le corresponde un valor octal, posicional y acumulable. Para codificar tres permisos rwx, hacen falta tres bits: cada uno tomaría el valor 0 o 1 según la presencia o ausencia del permiso. 23= 8, de ahí la notación octal.

- r vale 4.
- w vale 2.

- x vale 1.

La tabla siguiente servirá de ayuda:

Propietario			Grupo			Resto de la gente		
r	**w**	**x**	**r**	**w**	**x**	**r**	**w**	**x**
400	200	100	40	20	10	4	2	1

Para obtener el permiso final, basta sumar los valores. Por ejemplo, si quiere rwxrw-rw-, entonces obtiene 400+200+100+40+10+4+1=755, y para rw-r--r-- 400+200+40+4=644.

```
$ chmod 755 fic1
$ chmod 644 fic2
$ ls -l fic1 fic2
-rwxr-xr-x 1 seb users 0 mar 21 22:03 fic1
-rw-r--r-- 1 seb users 0 mar 21 22:03 fic2
```

La notación octal de los permisos no es sutil y no permite modificar un solo permiso. Es la totalidad de los permisos lo que se ha modificado de una sola vez.

3. Máscara de permisos

a. Restringir permisos de manera automática

En el momento de la creación de un archivo o de un directorio, se les asignan unos permisos automáticamente. Suele ser rw-r--r-- (644) para un archivo y rwxr-xr-x (755) para un directorio. Una máscara de permisos controla estos valores. Se puede modificar con el comando **umask**. El comando **umask** coge como parámetro un valor octal cuyo permiso individual se suprimirá de los permisos de acceso máximo del archivo o del directorio.

- Por defecto, se crean todos los archivos con los permisos 666 (rw-rw-rw-).
- Por defecto, se crean todos los directorios con los permisos 777 (rwxrwxrwx).
- Luego se aplica la máscara.
- La máscara es la misma para el conjunto de los archivos.
- Una máscara no modifica los permisos de los archivos existentes, sino solamente los de los archivos creados a partir de este momento.

Los permisos por defecto (máximo) de los archivos y de los directorios no son idénticos. Es lógico: como el permiso x permite entrar en un directorio, es normal que éste disponga de él por defecto. Este mismo permiso es inútil por defecto en los archivos: sólo una pequeña minoría de los archivos son scripts o binarios.

La máscara por defecto es 022, o sea ----w--w-. Para obtener este valor, inserte **umask** sin parámetro.

```
$ umask
0022
```

b. Cálculo de máscara

Para un archivo

```
Predeterminado rw-rw-rw- (666)
Retirar        ----w--w- (022)
Resta          rw-r--r-- (644)
```

Para un directorio

```
Predeterminado rwxrwxrwx (777)
Retirar        ----w--w- (022)
Resta          rwxr-xr-x (755)
```

Observe que aplicar una máscara no es sustraer, sino suprimir permisos de los establecidos por defecto, permiso a permiso. Por ejemplo:

```
Predeterminado rw-rw-rw- (666)
Retirar        ----wxrwx (037)
Resta          rw-r----- (640)
```

Y no 629, lo que es imposible en sistema octal...

4. Cambiar de propietario y de grupo

Es posible cambiar el propietario y el grupo de un archivo gracias a los comandos **chown** (change owner) y **chgrp** (change group). El parámetro -R cambia la propiedad de manera recursiva.

```
chown usuario fic1 [Fic2...]
chgrp grupo fic1 [Fic2...]
```

Al especificar el nombre de usuario (o de grupo), el sistema comprueba primero su existencia. Usted puede especificar un UID o un GID. En este caso, el sistema no efectuará comprobación alguna.

Para los dos comandos, no se modifican los permisos anteriores ni la ubicación del archivo. Con un solo comando se puede modificar el propietario y el grupo a la vez.

```
chown usuario[:grupo] fic1 [fic2...]
chown usuario[.grupo] fic1 [fic2...]
```

Sólo root tiene permiso para cambiar el propietario de un archivo. Pero un usuario puede cambiar el grupo de un archivo si forma parte del nuevo grupo.

```
$ chgrp video fic1
$ ls -l fic1
-rwxr-xr-x 1 seb video 0 mar 21 22:03 fic1
```

5. Permisos de acceso extendidos

a. SUID y SGID

Es posible establecer **permisos de acceso especiales** para archivos ejecutables. Estos permisos de acceso extendidos aplicados a un comando permiten sustituir los permisos otrogados al usuario que lo inició por los permisos del propietario o del grupo a los que pertenece el comando.

El ejemplo más sencillo es el programa **passwd**, que permite cambiar la contraseña. Si se ejecutara el comando con los permisos de un usuario clásico, **passwd** no podría abrir y modificar los archivos /etc/passwd y /etc/shadow:

```
$ ls -l /etc/passwd
-rw-r--r-- 1 root root 1440 feb 24 10:35 /etc/passwd
```

Puede observar que este archivo pertenece a root, y que sólo root puede escribir en él. Un usuario normal no puede leer su contenido sin interactuar. El comando **passwd** no debería, por lo tanto, poder modificar los archivos. Vea los permisos del comando **passwd** (/bin/passwd o /usr/bin/passwd):

```
> ls -l /usr/bin/passwd
-rwsr-xr-x 1 root shadow 78208 sep 21 23:06 /usr/bin/passwd
```

Lleva asociado un nuevo permiso: **s** para los permisos del usuario root. Este nuevo atributo permite la ejecución del comando con permisos de acceso extendidos. Durante el tratamiento, se ejecuta el programa con los permisos del propietario del archivo o del grupo al que pertenece. En el caso de passwd, se inicia con los permisos de root y no del usuario que lo lanzó.

El permiso s sobre el usuario se llama **SUID-Bit** (Set User ID Bit), y sobre el grupo, **GUID-Bit** (Set Group ID Bit).

El comando **chmod** permite ubicar SUID-Bit y GUID-Bit.

```
chmod u+s comando
chmod g+s comando
```

Los valores octales son 4000 para SUID-Bit y 2000 para GUID-Bit.

```
chmod 4755 comando
chmod 2755 comando
```

Sólo el propietario o el administrador puede activar esta propiedad. Posicionar SUID-bit o SGID-Bit tiene sentido únicamente si se han establecido los permisos de ejecución previamente (atributo x en el propietario o el grupo). Si éstos no están presentes; se sustituye la **s** por una **S**.

b. Real / efectivo

En los datos de identificación del proceso, ha podido observar la presencia de **UID** y **GID reales y efectivos**. Cuando se inicia un comando con un SUID-Bit o un SGID-Bit posicionado, los permisos se modifican. El sistema conserva los UID y GID de origen del usuario que inició el comando (UID y GID reales) transmitidos por el padre, los números UID y GID efectivos son los del propietario o del grupo de pertenencia del programa.

P. ej.: pepito (UID=100, GID=100) envía passwd, que pertenece a root (UID=1, GID=1) con SUID-Bit activado.

```
UID real: 100
GID real: 100
UID efectivo: 1
GID efectivo: 100
```

Si se posiciona sólo SGID-Bit:

```
UID real: 100
GID real: 100
UID efectivo: 100
GID efectivo: 1
```

Hay que subrayar que no se transmiten los SUID-Bit y SGID-bit a los hijos de un proceso. En este caso, se ejecutarán los hijos con los permisos del usuario que inició el comando básico, los UID reales.

c. Sticky bit

El **sticky bit** (bit pegajoso) permite asignar un criterio protector contra el borrado del contenido de un directorio. Imagine un directorio /tmp donde todos los usuarios tienen permiso para leer y escribir archivos.

```
$ ls -ld /tmp
drwxrwxrwx 6 root system 16384 Ago 14 13:22 tmp
```

En este directorio todo el mundo puede suprimir archivos, incluidos los que no le pertenecen (permiso w presente en todas partes y para todos). Si el usuario pepito crea un archivo, el usuario titi puede suprimirlo incluso aunque no le pertenezca.

El sticky bit aplicado a un directorio, aquí /tmp, impide esta operación. Sí, pepito aún puede visualizar y modificar el archivo, pero sólo su propietario (o el administrador) podrá suprimirlo.

```
$ chmod u+t /tmp
ls -ld /tmp
drwxrwxrwt 35 root root 77824 mar 21 22:30 /tmp
```

En octal, se utilizará el valor 1000 (chmod 1777 /tmp).

Aunque aplicado al usuario, el sticky bit, representado por una **t**, aparece en el grupo de permisos de "others".

d. Permisos y directorios

Si da el permiso s al grupo en un directorio, todos los archivos creados dentro de este directorio serán del mismo grupo que este directorio, sea cual sea el grupo de la persona que crea este archivo.

```
$ mkdir dir
$ chmod 770 dir
$ ls -ld dir
drwxrwx--- 2 seb users 4096 mar 21 22:36 dir
$ chgrp video dir
$ chmod g+s dir
$ ls -ld dir
drwxrws--- 2 seb video 4096 mar 21 22:37 dir
$ cd dir
$ touch pepito
$ ls -l pepito
-rw-r--r-- 1 seb video 0 mar 21 22:37 pepito
```

SERVICIOS

a. Equivalencia con init System V

A pesar de ser un atajo simple, la noción de objetivo es la de acercarse al nivel de ejecución clásico, ya que un objetivo es un estado a obtener. Un archivo objetivo lleva el sufijo **.target**. La ventaja de systemd es que a conseguido harmonizar los diferentes objetivos entre las distribuciones. Aquí tenemos una tabla comparativa de los objetivos entre System V y systemd:

Runlevel	Target Unit	Descripción
0	runlevel0.target, poweroff.target	Extinción de la máquina
1	runlevel1.target, rescue.target	Boot en modo single
2	runlevel2.target, multi-user.target	Consola, multi-usuario
3	runlevel3.target, multi-user.target	Consola, multi-usuario
4	runlevel4.target, multi-user.target	Consola, multi-usuario
5	runlevel5.target, graphical.target	Gráfico, multi-usuario
6	runlevel6.target, reboot.target	Reboot del equipo

b. Conocer el objetivo por defecto

El comando siguiente proporcionar el objetivo por defecto, es decir el estado en el cual el sistema alcanza al arrancar:

```
$ systemctl get-default
multi-user.target
```

El objetivo por defecto es un simple enlace simbólico llamado default.target, al archivo objetivo seleccionado.

```
$ cd l /etc/systemd/system/
$ ls -l default.target
 lrwxrwxrwx 1 root root 37 dic. 28 13:08
default.target ->
/lib/systemd/system/multi-user.target
```

c. Cambiar el objetivo por defecto

Para cambiar el objetivo por defecto, utilice el comando **systemctl** como sigue:

```
$ sudo systemctl set-default multi-user.target
```

Este comando tiene por efecto la modificación del objetivo del enlace simbólico. El hecho de cambiar el objetivo por defecto no modifica el estado actual del sistema.

d. Pasar de un objetivo a otro

El equivalente de un telinit es el comando **systemctl isolate**:

```
# systemctl isolate graphical.target
```

e. Modo seguro y emergencia

Dos objetivos particulares tienen un acceso directo desde el comando **systemctl**: rescue y emergency.

```
# systemctl rescue
# systemctl emergency
```

En los dos casos, se trata de alias de los comandos **isolate** vers les cibles respectives (rescue.target y emergency.target). El modo rescue es el modo mono usuario. El modo emergency se dedica a los casos críticos; proporciona el modo de funcionamiento más pequeño posible: el sistema de archivos raiz se monta en modo solo lectura, no se puede acceder a ningún otro sistema de archivos ni a la red.

f. Objetivos activos y dependencias

Un objetivo puede estar constituido por varios objetivos. Definir objetivos « reducidos » permite agruparles de forma más sencilla y reutilizarlos. Para listar los objetivos actualmente activos:

```
$ systemctl list-units --type target
UNIT                LOAD   ACTIVE SUB    DESCRIPTION
basic.target        loaded active active Basic System
```

```
cryptsetup.target    loaded active active Encrypted Volumes
getty.target         loaded active active Login Prompts
local-fs-pre.target  loaded active active Local File Systems (Pre)
local-fs.target      loaded active active Local File Systems
multi-user.target    loaded active active Multi-User System
network.target       loaded active active Network
paths.target         loaded active active Paths
remote-fs-pre.target loaded active active Remote File Systems (Pre)
remote-fs.target     loaded active active Remote File Systems
slices.target        loaded active active Slices
sockets.target       loaded active active Sockets
sound.target         loaded active active Sound Card
swap.target          loaded active active Swap
sysinit.target       loaded active active System Initialization
time-sync.target     loaded active active System Time Synchronized
timers.target        loaded active active Timers
```

Encontramos por supuesto el objetivo por defecto, multi-user.target, pero no está solo. Esto significa que la activación de un objetivo puede provocar la activación de otros. Se emplea también un mecanismo de dependencias. Aquí vemos el contenido del objetivo multi-user.target:

```
$ cat multi-user.target
[Unit]
Description=Multi-User System
Documentation=man:systemd.special(7)
Requires=basic.target
Conflicts=rescue.service rescue.target
After=basic.target rescue.service rescue.target
AllowIsolate=yes
```

Este archivo contiene:

- La línea **Requires=** proporciona la lista de las unidades dependientes obligatorias del objetivo.
- La línea **After=** indica el orden en que las unidades deben ser cargadas (pero estas no son obligatorias).

También encontramos un directorio multi-user.target.wants, que contiene una lista de las unidades, con frecuencia enlaces simbólicos, que deben ser cargados durante la activación del objetivo.

```
 ls -1 multi-user.target.wants/
abrt-ccpp.service
abrtd.service
abrt-oops.service
abrt-vmcore.service
abrt-xorg.service
atd.service
auditd.service
avahi-daemon.service
chronyd.service
crond.service
cups.path
cups.service
ipmievd.service
irqbalance.service
kdump.service
ksm.service
...
```

g. Listar todos los objetivos

Agregando --all al comando de listado, obtenemos todos los objetivos disponibles en el sistema.

```
$ systemctl list-units --type target --all
```

5. Servicios

Los servicios terminan con el sufijo .service. Al ser unidades como las otras, se controlan con el mismo comando **systemctl**. Tienen el mismo rol que los servicios init System V, y pocas cosas más, se utilizan de manera idéntica. Sin embargo, donde los servicios System V son con frecuencia simples scripts shell (bash ou sh), las unidades de servicios son archivos de texto estandarizados que describen las reglas y comandos vinculados a ese servicio.

Las herramientas como service o chkconfig son inútiles, y son sustituidas por el comando **systemctl**.

a. Acciones

servicio	systemctl	Descripción
service name start	systemctl start name	Arranca el servicio.
service name stop	systemctl stop name	Detiene el servicio.
service name restart	systemctl restart name	Reinicia el servicio.
service name condrestart	systemctl try-restart name	Reinicia el servicio solo si está arrancado.
service name reload	systemctl reload name	Vuelve a cargar la configuración.
service name status	systemctl status name systemctl is-active name	Indica si el servicio está arrancado.
service --status-all	systemctl list-units --type service -all	Muestra el estado de todos los servicios.

El comando **systemctl** accepta las acciones habituales, como **start**, **stop** o **restart**. A título comparativo, aquí tenemos una tabla que presenta las acciones posibles y sus equivalentes con el comando service (o el servicio System V mismo);

Obtendrá, por ejemplo, una lista de los servicios activos como ésta:

```
$ systemctl list-units --type=service
  UNIT                    LOAD   ACTIVE SUB
DESCRIPTION
  abrt-ccpp.service       loaded active exited
 Install ABRT coredump hook
  abrt-oops.service       loaded active running
ABRT kernel log watcher
  abrtd.service           loaded active running
ABRT Automated Bug Reporting Tool
  atd.service             loaded active running
Job spooling tools
  auditd.service          loaded active running
Security Auditing Service
  avahi-daemon.service    loaded active running
Avahi mDNS/DNS-SD Stack
  chronyd.service         loaded active running
NTP client/server
  crond.service           loaded active running
Command Scheduler
  cups.service            loaded active running
CUPS Printing Service
  dbus.service            loaded active running D-
Bus System Message Bus
...
```

Notará que, si la lista sobrepasa el tamaño de su consola, se espera una tecla para pasar a la siguiente. Para evitar este funcionamiento, utilice el siguiente parámetro:

```
$ systemctl list-units --type=service --no-pager
```

Sólo los servicios activos (vinculados al objetivo actual o arrancados de forma específica) se mostrarán. Para tener una lista más completa, agregue --all:

```
$ systemctl list-units --type=service --no-pager --all
```

Para obtener la lista completa de los servicios, sea cual sea su estado:

```
$ systemctl list-unit-files --type service
```

b. Estado

El estado proporcionado por systemctl es por lo general más completo que el provisto por System V, donde son los mismos scripts y syslog quienes permiten obtener las trazas. Así, constatará más abajo que una gran cantidad de información se proporciona y en particular las trazas del servicio. En el caso siguiente ssh, lo que puede ser muy util en caso de que el servicio no arranque.

```
# systemctl status sshd
● sshd.service - OpenSSH server daemon
   Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; vendor
preset: enabled)
   Active: active (running) since mer. 2016-12-28 13:22:32 CET; 2h 7min ago
     Docs: man:sshd(8)
           man:sshd_config(5)
  Process: 1102 ExecStart=/usr/sbin/sshd $OPTIONS (code=exited,
status=0/SUCCESS)
 Main PID: 1114 (sshd)
   CGroup: /system.slice/sshd.service
           └─1114 /usr/sbin/sshd

déc. 28 13:22:31 client systemd[1]: Starting OpenSSH server daemon...
déc. 28 13:22:32 client systemd[1]: PID file /var/run/sshd.pid not readable
(yet?) after start.
déc. 28 13:22:32 client sshd[1114]: Server listening on 0.0.0.0 port 22.
déc. 28 13:22:32 client sshd[1114]: Server listening on :: port 22.
déc. 28 13:22:32 client systemd[1]: Started OpenSSH server daemon.
déc. 28 13:22:41 client sshd[2364]: Accepted publickey for root from
192.168.99.1 port 53960 ssh2: RSA
54:04:40:a5:ba:60:5d:2a:ed:f6:c7:21:c2:6d:54:e6
```

Con respecto a estas trazas, puede consultar **journalctl** para una descripción más detallada.

c. Activación

La activación o desactivación de un servicio permite especificar su estado deseado en la carga del objetivo. Sin embargo, estos cambios no tendrán efecto inmediato en el servicio (iniciar o detener), esto requerirá la ejecución de un start/stop.

Al igual que para el comando **service**, aquí tenemos una tabla de correspondencia para el comando **chkconfig**:

chkconfig	systemctl	Descripción.
chkconfig name on	systemctl enable name	Activar un servicio.
chkconfig name off	systemctl disable name	Desactivar un servicio.
chkconfig --list name	systemctl status name systemctl is-enabled name	Indica el estado del servicio.
chkconfig --list	systemctl list-unit-files --type=service systemctl list-dependencies --before / --after name	Lista de los servicios, con sus dependencias.

La activación de un servicio solo añade un enlace simbólico en la unidad concernente en el directorio « wants » correspondiente al objetivo actual, como muestran los comandos siguientes:

```
# systemctl disable sshd
Removed symlink /etc/systemd/system/multi-user.target.wants/sshd.service.
# systemctl enable sshd
Created symlink from /etc/systemd/system/multi-
user.target.wants/sshd.service to /usr/lib/systemd/system/sshd.service.
```

d. Ocultación

Para o desactivar un servico no impide que sea ejecutado de forma manual o como dependencia. Podemos también constatar que un servicio que pensabamos completamente detenido se encuentra de nuevo en curso de ejecución. La solución es ocultarlo con mask :

```
# systemctl mask sshd
```

Una secuencia completa para deshacerse de forma permanente de un servicio es:

```
# systemctl stop sshd
# systemctl disable sshd
# systemctl mask sshd
```

El servicio puede volver a hacerse visible con unmask.

e. Dependencias

Al igual que los objetivos, los servicios tienen dependencias. Estas son facilmente visibles con list-dependencies. Observe que el sentido de after/before ne debe ser comprendido como « antes de haber arrancado sshd », si no « sshd arranca después de estas unidades ». Un árbol de dependencias se construye, he indica todo lo necesario para el arranque del servicio sshd.

```
# systemctl list-dependencies --after sshd
sshd.service
● ├─sshd-keygen.service
```

```
● ├─system.slice
● ├─systemd-journald.socket
● ├─basic.target
● │ ├─rhel-import-state.service
● │ ├─systemd-ask-password-plymouth.path
● │ ├─paths.target
● │ │ ├─brandbot.path
● │ │ ├─cups.path
● │ │ ├─systemd-ask-password-console.path
● │ │ └─systemd-ask-password-wall.path
● │ ├─slices.target
● │ │ ├─-.slice
● │ │ ├─system.slice
● │ │ └─user.slice
● │ ├─sockets.target
● │ │ ├─avahi-daemon.socket
```

Estas dependencias se definen, de manera recursiva, en cada uno de los archivos de la unidad, como es el caso de los objetivos. Las líneas Wants, After, y WantedBy definen las dependencias.

```
# cat /usr/lib/systemd/system/sshd.service
[Unit]
Description=OpenSSH server daemon
Documentation=man:sshd(8) man:sshd_config(5)
After=network.target sshd-keygen.service
Wants=sshd-keygen.service

[Service]
Type=forking
PIDFile=/var/run/sshd.pid
EnvironmentFile=/etc/sysconfig/sshd
ExecStart=/usr/sbin/sshd $OPTIONS
ExecReload=/bin/kill -HUP $MAINPID
KillMode=process
Restart=on-failure
RestartSec=42s

[Install]
WantedBy=multi-user.target
```

6. Compatibilidad con System V

Muchos scripts de arranque son genéricos, bien por los diferentes sistemas init o por los muchos sistemas Unix diferentes. Igualmente, una actualización, por ejemplo al pasar de Ubuntu 14.04 (upstart) a Ubuntu 16.04 (systemd) no debe dañar el arranque de las aplicaciones. También systemd debe ser capaz de arrancar los servicios System V.

Esta compatibilidad pasa por varias posibilidades. Por ejemplo, un servicio rc.service permitirá leer y ejecutar el contenido de /etc/rcX.d. O bien, un programa llamado systemd-sysv-generator hará exactamente lo mismo. Sin embargo, es en todo caso preferible que convierta sus scripts de arranque en unidades systemd.

7. Acciones de sistema

Al igual que init (via halt, reboot o telinit) permite reiniciar o detener un sistema, systemd hace lo mismo. Aquí tenemos una nueva tabla de comparación:

Comando	Systemd	Descripción
halt	systemctl halt	Detiene el sistema sin apagar el equipo.
poweroff, halt -p	systemctl poweroff	Detiene el sistema y apaga el equipo.
reboot	systemctl reboot	Reinicia el equipo.
pm-suspend	systemctl suspend	Suspende la ejecución del sistema (ahorro de energia).
pm-hibernate	systemctl hibernate	Hiberna el sistema.

Los comandos **shutdown** todavía están disponibles, al igual que **halt**, **reboot** o **poweroff** que son aliases a los comandos **systemd** asociados.

Administración de los usuarios

1. Fundamentos

a. Identificación y autenticación

La **identificación** consiste en saber quién es quién para determinar los permisos de la persona que se conecta. Se identifica un usuario mediante un login.

La **autenticación** consiste en aportar la prueba de quiénes somos mediante, por ejemplo, un secreto compartido entre el usuario y el sistema, y que sólo conocen ellos. Se autentifica al usuario con una contraseña.

b. Los usuarios

Un usuario es la asociación de un nombre de conexión, el login, con un UID y al menos un GID.

- **UID**: User ID.
- **GID**: Group ID.

Los UID y los GID suelen ser únicos. El login es único. Sin embargo, se puede considerar la asociación de varios logins al mismo UID, sabiendo que el sistema trabaja a veces con el login.

El UID identifica el usuario (o la cuenta asociada) a lo largo de su conexión. Se utiliza para el control de sus derechos y de los de los procesos que ha iniciado. Lo que se almacena en la tabla de los inodos, dentro de la tabla de los procesos, son los UID y GID, y no los logins.

El usuario dispone de los atributos básicos siguientes:

- un nombre de inicio de sesión llamado login;
- una contraseña;

- un UID;
- un GID correspondiente a su grupo principal;
- una descripción;
- un directorio de inicio de sesión;
- un comando de inicio de sesión.

Hay otros atributos disponibles mediante la utilización de la seguridad de las contraseñas en el archivo shadow (ver apartado correspondiente).

Se suelen asociar los UID de un valor inferior a 100 a cuentas especiales con derechos extendidos. Así el UID de root, el administrador, es 0. Según las distribuciones, a partir de 100, 500 o 1000, y hasta 65.535 (216-1) aproximadamente son los UID de los usuarios sin privilegios particulares. Estos parámetros pueden modificarse en etc/login.defs.

Un login tiene en principio un tamaño de 8 caracteres. En realidad, Linux y otros sistemas aceptan un tamaño mayor, pero con la mayoría de los comandos la visualización de los logins, incluso su gestión, está limitada a 8 caracteres.

Un login acepta la mayoría de caracteres. No debe empezar por una cifra. Es posible modificar la lista de los caracteres autorizados y forzar la longitud y la complejidad mediante los mecanismos de autenticación PAM y el archivo /etc/login.defs.

c. Los grupos

Cada usuario forma parte de al menos un grupo. Un grupo agrupa usuarios. Como para los logins, el GID del grupo siempre acompaña al usuario para controlar sus privilegios. Un usuario puede formar parte de varios grupos. En este caso, hay que distinguir su grupo primario de los grupos secundarios. Sin embargo, un grupo no puede formar parte de otro grupo.

Los grupos son también números (GID). Existen grupos específicos para la gestión de algunas propiedades del sistema y, en particular, el acceso a ciertos periféricos.

Cada vez que un usuario crea un archivo, éste recibe como valor de grupo con privilegios el primario del creador. Si el usuario seb tiene como grupo primario **users**, entonces los archivos creados por seb tendrán como grupo con privilegios **users**.

Un usuario dispone de todos los derechos asociados a sus grupos secundarios. Si seb tiene como grupo secundario **vídeo** y un archivo dispone de los derechos de escritura para este grupo, entonces seb tendrá derecho a modificar su contenido.

El comando **id** permite conocer la información esencial sobre un usuario: uid, gid, grupos secundarios.

```
$ id seb
uid=1000(seb) gid=100(users) grupos=100(users),16(dialout),3(sys),33(video)
```

Seb ha creado un archivo. Su propietario es seb y su grupo es el grupo primario de seb: users.

```
$ touch test
$ ls -l test
-rw-r--r-- 1 seb users 0 abr 10 14:30 test
```

Para conocer los tipos de codificación autorizados en su distribución, consulte el manual de la función crypt (man crypt).

d. Las contraseñas

Las contraseñas permiten autenticar a los usuarios. Deben ser lo bastante complejas como para que no se pueden descubrir fácilmente, pero lo bastante intuitivas como para que se puedan recordar. Las contraseñas están cifradas (SHA-256, SHA-512, Blowfish, por ejemplo) y no son directamente legibles bajo su forma cifrada por el usuario para que nadie pueda intentar descifrarlas.

Un usuario debería cambiar regularmente su contraseña, no escribirla nunca en ninguna parte ni llevarla encima. Luego veremos cómo se puede obligar a un usuario a aplicar reglas de creación y duración de contraseñas.

Veamos por ejemplo el resultado encriptado por SHA-512 (reconocible por el 6 que empieza la cadena) de una contraseña:

```
contraseña
$6$nrZeMJZT$GCCJz86zIiyBH3O88bQ8YwypCsQ153Wi/
tgjgv5EbaVCL.N1efycL6wj45GwhXWSyXOjuX0CIbVCjSDfmUQOh.
```

Para conocer los tipos de codificación autorizados en su distribución, consulte el manual de la función crypt (man crypt).

2. Los archivos

a. /etc/passwd

El archivo /etc/passwd contiene la lista de los usuarios del sistema local. Cualquier usuario puede leerlo. La información que contiene es pública y útil tanto para el sistema como para los usuarios. Cada línea representa un usuario y se compone de siete campos.

```
Login:password:UID:GID:comment:homedir:shell
```

- Campo 1: el login o nombre de usuario.
- Campo 2: en las antiguas versiones, la contraseña cifrada. Si hay una x, se coloca la contraseña en /etc/shadow. Si es un signo de exclamación, se bloquea la cuenta.
- Campo 3: el User ID.
- Campo 4: el GID, o sea, el grupo principal.
- Campo 5: un comentario o descripción. Es un campo de información.
- Campo 6: el directorio de trabajo, personal, del usuario. Es el directorio al que llega cuando se conecta.
- Campo 7: el shell por defecto del usuario. Pero puede ser cualquier otro comando, incluso un comando que prohíbe la conexión.

b. /etc/group

El archivo /etc/group contiene la definición de los grupos de usuarios y en cada uno, la lista de los usuarios de los cuales es el grupo secundario. Cada línea se compone de cuatro campos:

```
Group:password:GID:user1,user2...
```

- Campo 1: el nombre del grupo.
- Campo 2: la contraseña asociada. Veremos la explicación justo a continuación.
- Campo 3: el Group Id.
- Campo 4: la lista de usuarios que forman parte de este grupo.

Es inútil colocar en el cuarto campo a los usuarios que tienen este grupo como grupo principal; lo maneja el sistema.

Le puede sorprender la presencia de un campo de contraseña para los grupos. En la práctica se utiliza pocas veces. Como por supuesto es imposible iniciar sesión como grupo, la explicación está en otra parte. Un usuario tiene derecho a cambiar de grupo para coger, de manera temporal al menos, un grupo secundario como grupo primario con el comando **newgrp**.

En este caso, el administrador puede establecer una contraseña para el grupo para proteger el acceso a este grupo como grupo principal.

c. /etc/shadow

El archivo /etc/shadow acompaña al archivo /etc/passwd. Ahí se almacenan, entre otras cosas, las contraseñas cifradas de los usuarios. Para ser más precisos, contiene toda la información sobre las contraseñas y su validez en el tiempo. Cada línea se compone de 9 campos separados por ":":

```
bean:$2a$10$AjADxPEfE5iUJcltzYA4wOZO.f2UZ0qP/8EnOFY.P.m10HifS7J8i:13913
:0:99999:7:::
```

- Campo 1: login.
- Campo 2: contraseña cifrada. El xx inicial indica el tipo de cifrado.
- Campo 3: número de días desde el 1º de enero de 1970 hasta el último cambio de contraseña.
- Campo 4: número de días sin poder cambiar la contraseña (0: se puede cambiar en cualquier momento).
- Campo 5: número de días a partir de los cuales se debe cambiar la contraseña.
- Campo 6: número de días antes del vencimiento de la contraseña durante los cuales se debe avisar al usuario.
- Campo 7: número de días después del vencimiento de la contraseña tras los cuales se desactiva la cuenta.
- Campo 8: número de días desde el 1º de enero de 1970 hasta el momento en el cual se desactivó la cuenta.
- Campo 9: reservado.

En el ejemplo de la línea bean, se ha cambiado la contraseña 13913 días después del 01/01/1970. La contraseña se debe cambiar antes de 0 días, pero seguirá siendo válida, ya que el campo siguiente indica que hay que cambiarla al cabo de 99999 días (273 años) y el campo 5 está vacío (sin obligación de cambio de contraseña). La cuenta se desactiva tras 7 días: no hay riesgo de que ocurra...

Los valores actuales para el cifrado de contraseñas son los siguientes:

- **1**: MD5
- **$2a$**: Blowfish
- **$5**: SHA-256
- **$6**: SHA-512
- Otro: DES

Para conocer la fecha en función del 01/01/1970, utilice el comando **date** como a continuación, añada el número de días deseado:

```
# date --date "1 ene 1970 +17194days"
sáb ene 28 00:00:00 CET 2017
```

d. /etc/gshadow

El archivo /etc/gshadow es el equivalente del archivo anterior, pero para los grupos. Sin embargo, algunas de las anteriores distribuciones de Linux no lo soportan por defecto. Se colocan las contraseñas de los grupos en el segundo campo de /etc/group.

```
test:$6$FBuQQ/oCSnY/PLbx$MD…rtYrADveathy6xO/8ZYfIpAMF72OZKHUGwJMBox/:root:seb
```

El formato de gshadow es el siguiente:

- Campo 1: nombre del grupo.
- Campo 2: contraseña cifrada. El xx inicial indica el tipo de cifrado, SHA-512 en este caso.
- Campo 3: administradores del grupo (los que pueden modificar la contraseña o los miembros del grupo).
- Campo 4 : miembros del grupo: pueden acceder sin contraseña. Es la misma lista que en /etc/group.

3. Gestión de los usuarios

a. Creación

La creación de un usuario podría ser totalmente manual, ya que Linux (y los demás Unix) se apoya en una serie de comandos que "sólo" modifican archivos planos ya existentes y que crean y vuelven a copiar archivos y carpetas en el sitio correcto con los privilegios correctos.

La creación de un usuario consiste en:

- añadir una línea en /etc/passwd,
- añadir una línea en /etc/shadow,
- añadir una información si es preciso en /etc/group,
- crear el directorio personal y colocar el contenido de la carpeta /etc/skel,
- cambiar los permisos y el propietario del directorio personal,
- cambiar la contraseña (cifrada).

Se puede crear directamente una cuenta editando los archivos con un editor, aunque no es nada aconsejable. Si desea hacerlo de todas maneras, utilice el comando **vipw**, que actualizará las diversas cachés asociadas a la gestión de las cuentas.

El comando **vipw** admite cuatro argumentos:

- -p: edición de /etc/passwd.
- -g: edición de /etc/group.
- -s: edición de /etc/shadow.
- -gs: edición de /etc/gshadow.

Evidentemente, en la práctica hay que evitar editar estos archivos a mano, ya que hay otros muchos mecanismos creados a partir del uso de comandos específicos para la creación o modificación de cuentas, como por ejemplo, la comprobación de la complejidad de la contraseña.

Cabe destacar que existe una restricción asociada al formato del login. Responde a una norma POSIX: sólo puede empezar por un carácter alfabético, seguido de caracteres portables (letras, cifras, guión, guión bajo, etc.).

Todo esto se puede llevar a cabo con el comando **useradd**. Añade una nueva cuenta y efectúa las principales operaciones:

- creación del usuario y edición de los archivos;
- creación de un grupo privado del usuario (con el mismo nombre que éste);
- creación del directorio personal, edición y modificación de los derechos.

```
# useradd <options> login
```

Si no se ha especificado ninguna opción, se recuperarán los valores por defecto dentro del archivo /etc/default/useradd. Se aceptan las opciones principales siguientes:

Opción	Función
-m	Crea también el directorio personal. Se incluye a veces por defecto, pero es mejor comprobar que el directorio personal esté presente después de la utilización del comando si no utiliza esta opción.
-u	Especifica el UDI numérico del usuario, para forzarlo. Dicho de otro modo, se calcula el UID según las reglas del archivo login.defs y los UID existentes.
-g	Especifica el grupo primario del usuario, por GID o por su nombre (variable GROUP).
-G	Especifica los grupos adicionales (secundarios, del usuario) separados por comas (variable GROUPS).
-d	Ruta del directorio personal. En general /home/<login>, pero no se puede especificar cualquier ruta (variable HOME/<login>).
-c	Un comentario asociado a la cuenta. Puede ser cualquiera, pero a veces algunos comandos como **finger** lo pueden utilizar. El usuario puede modificar su contenido con el comando **chfn**.
-k	Ruta del directorio que contiene el esqueleto del árbol del directorio del usuario. Suele ser /etc/skel (variable SKEL).
-s	Shell (intérprete de comandos) por defecto del usuario (variable SHELL). El usuario puede cambiarlo mediante el comando **chsh**.
-p	La contraseña del usuario. ¡Cuidado! ¡La contraseña debe estar ya cifrada! A no ser que se vuelva a copiar la contraseña de una cuenta genérica, lo mejor es usar el comando **passwd**.

El comando siguiente crea la cuenta "roberto" con la mayoría de las opciones básicas especificadas. Es sólo un ejemplo, excepto a veces la -m, ya que si no se especifica nada, son las opciones por defecto en relación con las especificadas en el archivo /etc/defaults/useradd.

```
# useradd -m -u 1010 -g users -G video,dialout,lp -s /bin/bash -d
/home/roberto -c "Cuenta de Roberto" roberto
# grep roberto /etc/passwd
robert:x:1010:100:Cuenta de Roberto:/home/roberto:/bin/bash
```

El comando no crea la contraseña. Hay que hacerlo a mano con el comando **passwd**.

```
# passwd roberto
Changing password for roberto.
Nueva contraseña:
Vuelva a introducir la nueva contraseña:
Contraseña cambiada.
```

Si desea generar usted mismo una contraseña ya codificada (para useradd o usermod), puede utilizar la línea de comando siguiente. El primer parámetro es la contraseña en texto claro, la segunda un ”salt” que debera ser muy largo y aleatorio:

```
$ echo $(perl -e’print crypt("foobar", "\$6\$salttogenerate\$")’)
$6$salttogenerate$2y5GK3joHstsKC0CKulp19OtPVn2mQpC1hWhinU1vUu/sFFcQdFa/
RaqZ32wiGtHpl3dyBvrXSYpNj4GqcWTI0
```

b. Seguridad de las contraseñas

Cambiar la contraseña

El comando **passwd** permite gestionar las contraseñas, pero también las autorizaciones de inicio de sesión, así como la mayoría de los campos presentes en /etc/shadow.

Cualquier usuario tiene derecho a cambiar su contraseña en el plazo especificado por el campo 4 de /etc/shadow. La acción por defecto consiste en cambiar la contraseña del usuario actual. Se requiere la antigua contraseña por seguridad (en particular para evitar que una persona mal intencionada modifique su contraseña a sus espaldas). La inserción está enmascarada.

```
$ id
uid=1000(seb) gid=100(users) ...
$ passwd
Changing password for seb.
Antigua contraseña:
Nueva contraseña:
Vuelva a introducir la nueva contraseña:
Contraseña cambiada.
```

Los módulos **PAM** (Pluggable Authentication Module) pueden imponer exigencias más o menos estrictas para el cambio de contraseña: en cuanto a la longitud, que no se base en una palabra del diccionario, etc. Veamos lo que ocurre cuando se quiere utilizar pepe (demasiado corto), qwerty (demasiado largo) y María (diccionario):

```
$ passwd
Changing password for seb.
Antigua contraseña:
Nueva contraseña:
Contraseña incorrecta: demasiado corta
Nueva contraseña:
Contraseña incorrecta: demasiado simple
Nueva contraseña:
Contraseña incorrecta: basada en una palabra del diccionario
passwd: Número máximo de intentos agotado para el servicio
```

El usuario root tiene derecho a modificar las contraseñas de todos los usuarios del sistema, sin que sea preciso que conozca la contraseña anterior. Aún mejor: puede forzar el uso de una contraseña incluso aunque no haya sido validada por PAM:

```
# passwd seb
Changing password for seb.
Nueva contraseña:
Contraseña incorrecta: basada en una palabra del diccionario
Vuelva a introducir la nueva contraseña:
Contraseña cambiada.
```

Gestionar la validez

Se pueden modificar todos los campos de /etc/shadow con el comando **passwd**. Veamos algunas de las opciones disponibles.

Opción	Función
-l	Lock: bloquea una cuenta al añadir un ! delante de la contraseña cifrada.
-u	Unlock: desbloquea la cuenta. No se puede activar una cuenta que no tenga contraseña. Hay que utilizar -f para ello.
-d	(root) Suprime la contraseña de la cuenta.
-n <j>	(root) Duración de vida mínima en días de la contraseña.
-x <j>	(root) Duración de vida máxima en días de la contraseña.
-w <j>	(root) Número de días antes de un aviso.
-i <j>	(root) Período de gracia antes de la desactivación si ha vencido la contraseña.
-S	(root) Estatus de la cuenta.

En el ejemplo siguiente, se ha modificado la cuenta bean de esta manera:

- Debe esperar 5 días tras insertar una nueva contraseña para poder cambiarla.
- Su contraseña es válida 45 días.
- Se le avisa 7 días antes de que deba cambiar la contraseña.
- Si no cambia la contraseña tras 45 días, dispone aún de 5 días antes de que sea bloqueada.

```
# passwd -n 5 -x 45 -w 7 -i 5 bean
Password expiry information changed.
```

Veamos la línea de /etc/shadow asociada.

```
bean:$6$fdg4ZnHM$IeMrObWcKlXj.K9ev4ePYplRXzpXsQa/
30GIeBr8xjQAmJg7I..Uz018pG2McPdTWg5OiPJTqlm22to6uIORl0:17194:5:45:7:5::
```

El comando **chage** permite hacer más o menos lo mismo. Es un comando del root. Iniciado sin otro argumento que el login del usuario, es interactivo. Observe al final la posibilidad de modificar la fecha del último cambio de la contraseña y una fecha fija de expiración de la contraseña (campo 8):

```
# chage bean
Changing the aging information for bean
Enter the new value, or press ENTER for the default

      Minimum Password Age [5]:
      Maximum Password Age [45]:
      Last Password Change (YYYY-MM-DD) [2017-01-28]:
      Password Expiration Warning [7]:
      Password Inactive [5]:
      Account Expiration Date (YYYY-MM-DD) [-1]: 2017-02-28
```

Veamos la línea /etc/shadow resultante:

```
bean:$6$fdg4ZnHM$IeMrObWcKlXj.K9ev4ePYplRXzpXsQa/
30GIeBr8xjQAmJg7I..Uz018pG2McPdTWg5OiPJTqlm22to6uIORl0:17194:5:45:7:5:17225:
```

Se aceptan los parámetros siguientes:

Opción	Función
-m	Mindays: equivale a passwd -n.
-M	Maxdays: equivale a passwd -x.
-d	Fecha de última modificación de la contraseña (desde el 01/01/1970).
-E	Fecha de vencimiento de la contraseña (desde el 01/01/1970).
-I	Inactive: equivale a passwd -i.
-W	Warndays: equivale a passwd -w.
-l	List: muestra todos los detalles.

Los detalles son mucho más legibles con **chage** que con **passwd**:

```
# passwd -S bean
bean P 01/28/2017 5 45 7 5

# chage -l bean
Last password change                                    : ene 28, 2017
Password expires                                        : mar 14, 2017
Password inactive                                       : mar 19, 2017
Account expires                                         : feb 28, 2017
Minimum number of days between password change          : 5
Maximum number of days between password change          : 45
Number of days of warning before password expires       : 7
```

Un usuario cualquiera puede visualizar sus propios detalles, pero se le podría pedir su contraseña.

c. Modificación

Utilice el comando **usermod** para modificar una cuenta. Utiliza la misma sintaxis y las mismas opciones que useradd, pero dispone también de una sintaxis complementaria que necesita algunas precisiones.

Opción	Función
-L	Bloquear la cuenta, como passwd -l.
-U	Desbloquar la cuenta, como passwd -u.
-e <n>	Vencimiento: la contraseña expira n días después del 01/01/1970.
-u <UID>	Modificar el UID asociado al login. Se modifica en consecuencia el propietario de los archivos que pertenecen al antiguo UID dentro del directorio personal.
-l <login>	Modificar el nombre de login.
-m	Move: implica la presencia de -d para especificar un nuevo directorio personal. Se mueve el contenido del antiguo directorio al nuevo.

d. Eliminación

Suprime un usuario con el comando **userdel**. Por defecto no se suprime el directorio personal. Para ello, debe pasar la opción -r.

```
# userdel -r bean
```

El comando no comprueba la totalidad del sistema de archivos, los archivos que estén fuera de su directorio personal no se borrarán. Es el administrador quien debe buscarlos y borrarlos. Para ello, puede usar el comando **find**.

4. Gestión de los grupos

a. Creación

Puede crear un grupo directamente en el archivo /etc/group o bien utilizar los comandos asociados. Si edita el archivo manualmente, utilice el comando **vigr** (o vipw -g).

El comando **groupadd** permite crear un grupo. Su sintaxis sencilla acepta el argumento -g para especificar un GID preciso.

```
# grep amigos /etc/group
amigos:!:1234:
```

b. Modificación

El comando **groupmod** permite modificar un grupo. Sus parámetros son los siguientes:

Opción	Función
-n <nombre>	Renombra el grupo.
-g <GID>	Modifica el GID. Cuidado: no se modifica el grupo al que pertenecen los archivos correspondientes

```
# groupadd grp1
# groupmod -n grp2 grp1
# grep grp /etc/group
grp2:x:1003:
```

c. Eliminación

El comando **groupdel** permite suprimir un grupo. Primero el comando comprueba si el grupo que desea suprimir es el grupo primario de un usuario. En este caso, no se permite suprimir el grupo.

Pero si finalmente se puede, no se efectúa más acción que la de suprimir la línea correspondiente en /etc/group. Le corresponde a usted comprobar el sistema de archivos (y la configuración de las aplicaciones si es necesario) para suprimir cualquier traza de este grupo.

```
# groupdel amigos
```

d. Contraseña

El comando **gpasswd** permite asignar una contraseña a un grupo. Esta contraseña se almacena en /etc/group, o, en el caso más frecuente, en /etc/gshadow. Modifique la contraseña del grupo test según este modelo:

```
gpasswd test
```

Agregue un usuario al grupo:

```
@ gpasswd -a seb test
```

Defina la lista de todos los miembros del grupo:

```
# gpasswd -M seb, steph test
```

Defina la lista de administradores del grupo, los que podrán gestionar los usuarios y sus contraseñas:

```
# gpasswd -A admin test
```

Si proporciona una contraseña a un grupo, todos los usuarios que conozcan esta contraseña, incluidos los que no forman parte del grupo, podrán usarla. El grupo se convierte en su grupo principal.

```
$ id -gn
uid=1001(steph) gid=1002(steph) grupos=1002(steph)
$ newgrp test
Contraseña:
$ id
uid=1001(steph) gid=1001(test) grupos=1002(steph),1001(test)
```

El acceso al grupo puede restringirse limitando a sus miembros con el parámetro -R. Es decir, los miembros del grupo no tendrán que introducir la contraseña.

Archivado y backup

1. Las herramientas de copia de seguridad

La copia de seguridad constituye un trabajo importante del administrador, ya que, en caso de problema grave, se suele restaurar el sistema desde una copia de seguridad, o una imagen del sistema cuando éste estaba todavía bien (funcionamiento correcto, sin corrupción). Se entrega cada Unix con sus propios comandos y procedimientos de copia de seguridad; sin embargo, existen algunas herramientas comunes.

a. Comandos, planes, scripts

- Para la copia de seguridad de los archivos y estructuras, utilice los comandos **tar** y **cpio**. Estos comandos guardan una estructura, y no un sistema de archivos. Se puede hacer coincidir los dos.
- Para la copia de seguridad física de discos y sistemas de archivos (los dumps), utilice el comando **dd**.

Una copia de seguridad incremental consiste en salvaguardar una primera vez la totalidad de los datos y luego únicamente los archivos modificados. También se pueden encontrar en forma de programas libres o en el mercado soluciones más avanzadas de copia de seguridad (Networker, por ejemplo).

A veces el administrador tendrá que definir scripts de copia de seguridad y restauración adaptados a cada caso concreto (partición del sistema, datos aplicativos...) y automatizar cuando sea posible la ejecución de estos scripts en función de la fecha, hora o carga de la máquina.

También será importante definir un plan de copia de seguridad y, para ello, hay que hacerse las preguntas correctas:

- ¿Qué se debe guardar?
- ¿Con qué frecuencia?
- ¿Cuánto tiempo se conservarán las copias de seguridad, dónde, cuántos ejemplares?
- ¿Dónde se almacenará el historial de las copias de seguridad?

- ¿Cuál es el soporte más apropiado?
- ¿Cuánta capacidad necesita el soporte de copia de seguridad?
- ¿Cuánto tiempo se prevé para guardar un archivo, un sistema de archivos?, y ¿es razonable?
- ¿La copia de seguridad debe ser automática o manual?
- ¿Cuál es el método más apropiado para la copia de seguridad?

Como cada caso es único, este libro no puede contestar a todas estas preguntas. Las respuestas dependen del entorno de destino (producción, integración, tests, etc.). Sin embargo, no se olvide de hacer una copia de seguridad del sistema (raíz, /opt, /usr, /var, /boot, etc.) después de una instalación y antes de una modificación importante, por si acaso fuera necesario volver atrás.

b. Otros comandos

- **mt**: control de una cinta magnética.
- **touch**: pone la fecha de última modificación a la hora actual, para forzar una copia de seguridad incremental.
- **find**: selecciona los archivos que hay que salvaguardar.
- **compress** y **uncompress**: compresión y descompresión de los archivos.
- **gzip**, **gunzip**, **zcat**: compresión y descompresión en formato GnuZip.
- **xz**, **unxz xzcat**: compresión y descompresión del formato .xz (derivado de lzma).

2. tar

El comando **tar** es simple y eficaz. Crea archivos de los archivos, incluida la estructura de archivos, en cualquier tipo de soporte, incluido en otro archivo (archivo con la extensión .tar). El archivo así creado puede extenderse a varios volúmenes: cuando la cinta o el disquete está lleno, le corresponde al usuario insertar uno nuevo. El proceso de copia/restauración continúa.

a. Archivadores

La sintaxis es la siguiente:

```
tar cvf nombre_archivo archivo(s)
```

Por ejemplo, para colocar en un archivo **tar** el directorio Desktop:

```
$ tar cvf desktop.tar Desktop/
Desktop/
Desktop/fusion-icon.desktop
Desktop/konsole.desktop
Desktop/Support.desktop
Desktop/Office.desktop
Desktop/Terminal.desktop
Desktop/MozillaFirefox.desktop
Desktop/Printer.desktop
Desktop/.directory
Desktop/myComputer.desktop
Desktop/trash.desktop
Desktop/SuSE.desktop
```

```
Desktop/Windows.desktop
```

Los parámetros son los siguientes:

- c: creación de archivo;
- V: modo verboso, **tar** muestra lo que hace;
- f: el parámetro siguiente es el nombre del archivo.

b. Listar

La sintaxis es:

```
tar tvf nombre_archivo
```

Para listar el contenido del archivo anterior:

```
$ tar tvf desktop.tar
drwx------ seb/users       0 2008-04-17 09:44 Desktop/
-rw-r--r-- seb/users     191 2007-10-20 20:10 Desktop/fusion-icon.desktop
-rw-r--r-- seb/users    4786 2007-09-26 00:43 Desktop/konsole.desktop
-rw-r--r-- seb/users     665 2008-04-08 15:14 Desktop/Support.desktop
-rw-r--r-- seb/users    1051 2007-10-05 10:16 Desktop/Office.desktop
-rw-r--r-- seb/users    4586 2007-12-05 11:37 Desktop/Terminal.desktop
-rw-r--r-- seb/users     829 2007-10-17 12:12 Desktop/MozillaFirefox.desktop
-rw-r--r-- seb/users    3952 2007-10-05 10:16 Desktop/Printer.desktop
-rw-r--r-- seb/users    2053 2007-10-05 10:16 Desktop/.directory
-rw-r--r-- seb/users     450 2007-10-23 11:58 Desktop/myComputer.desktop
-rw-r--r-- seb/users     218 2008-02-22 08:43 Desktop/trash.desktop
-rw-r--r-- seb/users     328 2008-04-08 15:14 Desktop/SuSE.desktop
-rw-r--r-- seb/users     472 2008-04-17 09:44 Desktop/Windows.desktop
```

El parámetro t lista el contenido del archivo.

c. Restauración

Para restaurar el contenido de un archivo, la sintaxis es:

```
tar xvf nombre_archivo archivos
```

Para restaurar el archivo anterior:

```
tar xvf desktop.tar
Desktop/
Desktop/fusion-icon.desktop
Desktop/konsole.desktop
Desktop/Support.desktop
Desktop/Office.desktop
Desktop/Terminal.desktop
Desktop/MozillaFirefox.desktop
Desktop/Printer.desktop
Desktop/.directory
Desktop/myComputer.desktop
Desktop/trash.desktop
Desktop/SuSE.desktop
Desktop/Windows.desktop
```

El parámetro x permite la extracción del conjunto de los ficheros del archivo, o de los ficheros especificados después del nombre del archivo.

d. Otros parámetros

El comando **tar** de gnu permite gestionar los formatos de compresión directamente:

- z: se comprime el archivo al formato gzip.
- Z: se comprime el archivo al formato compress.
- j: se comprime el archivo al formato bzip2.
- J: se comprime el archivo en el formato xz.

Así, los comandos anteriores para el formato de compresión **gzip** se convierten en:

```
$ tar cvzf desktop.tar.gz Desktop/
Desktop/
Desktop/fusion-icon.desktop
Desktop/konsole.desktop
Desktop/Support.desktop
Desktop/Office.desktop
Desktop/Terminal.desktop
Desktop/MozillaFirefox.desktop
Desktop/Printer.desktop
Desktop/.directory
Desktop/myComputer.desktop
Desktop/trash.desktop
Desktop/SuSE.desktop
Desktop/Windows.desktop
$ ls -l desktop.tar*
-rw-r--r-- 1 seb users 30720 may  9 11:16 desktop.tar
-rw-r--r-- 1 seb users  7556 may  9 11:22 desktop.tar.gz
```

Observe la diferencia de tamaño. Se pueden utilizar las opciones de compresión con c, t y x. Observe que se comprime el archivo final, y no los archivos de manera individual. Puede ser preferible no especificar una opción de compresión si está haciendo una copia en un soporte que gestiona por sí mismo la compresión.

Si comprime su archivo para destinarlo a otros sistemas, o desea guardar una compatibilidad con los parámetros por defecto de tar, puede proceder así:

```
$ gzip -cd desktop.tar.gz | tar xvf -
Desktop/
Desktop/fusion-icon.desktop
Desktop/konsole.desktop
Desktop/Support.desktop
Desktop/Office.desktop
Desktop/Terminal.desktop
Desktop/MozillaFirefox.desktop
Desktop/Printer.desktop
Desktop/.directory
Desktop/myComputer.desktop
Desktop/trash.desktop
Desktop/SuSE.desktop
Desktop/Windows.desktop
```

El parámetro -d precisa a **gzip** que debe descomprimir el archivo, mientras que -c pasa el resultado por la salida estándar. El - final indica a tar que debe recuperar el flujo por la entrada estándar.

Observe que las versiones recientes de tar reconocen directamente el formato de compresión, si se emplea sin pasar las opciones específicas de estos formatos.

```
# file test.tar.xz
test.tar.xz: XZ compressed data
# tar xf test.tar.xz
# echo $?
0
```

rsync. Como alternativa a cp

rsync puede usarse como sustituto avanzado del comando `cp`. especialmente en la copia de ficheros grandes:

```
$ rsync -P origen destino
```

La opción `-P` es igual que `--partial --progress`, que mantiene los archivos parcialmente transferidos y muestra una barra de progreso durante la transferencia.

Quizás quiera usar la opción `-r --recursive` para recorrer recursivamente los directorios, o la opción `-R` para usar nombres de rutas relativos (recreando la jerarquía completa de carpetas en la carpeta de destino).

Como herramienta para copias de seguridad

El protocolo rsync puede usarse para hacer copias de seguridad, transfiriendo solo aquellos ficheros que hayan cambiado desde la última copia. Esta sección describe un sencillo script programdo para hacer copias de seguridad usando rsync, generalmente usado para hacer copias a dispositivos extraibles. Para ver un ejemplo más detallado y con **opciones adicionales para preservar determinados archivos del sistema**, véase Full system backup with rsync.

Automated backup

A modo de ejemplo, el script es creado en el directorio `/etc/cron.daily` , y se ejecutará a diario si un demonio cron se instala y configura adecuadamente. Configurar y usar cron no entra en el ámbito de este artículo.

Primero, cree un script que contenga las opciones de comandos apropiadas:

`/etc/cron.daily/backup`

```
#!/bin/bash
rsync -a --delete /carpeta/a/respaldar  /carpeta/de/copia/de/seguridad &>
/dev/null
```

`-a`
: indica que los ficheros deberían ser archivados, lo cual significa que se preservarán casi todos las características (**no** así las ACLS, los enlaces duros o atributos extendidos como capacidades)

`--delete`
: indica que los ficheros borrados en origen también serán borrados en destino.

Aquí, `/carpeta/a/resguardar` debería cambiarse a aquella que se quiera respaldar (`/home`, por ejemplo) y `/carpeta/de/copia/de/seguridad` es donde la copia de seguridad debería salvarse (`/media/disk`, por ejemplo).

Finalmente, el script debe tener permisos de ejecución:

```
# chmod +x /etc/cron.daily/rsync.backup
```

Respaldo automático con SSH

Si se respalda a una máquina remota usando SSH, use este script en su lugar:

`/etc/cron.daily/backup`

```
#!/bin/bash
rsync -a --delete -e ssh /carpeta/a/respaldar
usuarioremoto@maquinaremota:/carpeta/de/copia/de/seguridad &> /dev/null
```

`-e ssh`
: indica a rsync que use SSH

`remoteuser`
: es el usuario en la `maquinaremota`

`-a`
: agrupa todas estas opciones `-rlptgoD` (recursivo, enlaces, permisos, tiempos, grupos, propietario, dispositivos)

Configuración de Redes. Casos particulares

a. NetworkManager

NetworkManager es un servicio que permite la configuración y administración dinámica de interfaces de red y de los protocolos asociados. Se acompaña en particular de un conjunto de comandos (**nmcli**, **nmtui**…) que permiten gestionar de forma centralizada la red de su equipo.

Se trata de un desarrollo orientado al servidor; una de las primeras cosas que realizan tanto los administradores como algunas soluciones es la eliminación de NetworkManager (o por lo menos su desactivación) y volver a un funcionamiento basado en archivos de configuración estáticos. Después de todo, una vez instalada, la configuración de la pila de red de un servidor raramente evoluciona.

En una distribución Red Hat, para pasar de NetworkManager al servicio de red clásico, siga los pasos a continuación:

```
systemctl stop NetworkManager
systemctl mask NetworkManager
systemctl disable NetworkManager
systemctl enable network
systemctl unmask network
systemctl start network
```

Para Ubuntu y Debian, sustituya network por **networking**.

b. Nomenclatura de las interfaces

Parece que muchas distribuciones han modificado las reglas de nomenclatura de las interfaces de red. La nomenclatura genérica ha sido remplazada por un nombre que representa la numeración de las interfaces por el bios o por su enumeración en el bus de hardware. Así, podemos encontrar

nombres de interfaces como p2p1, enp0s2… Los nombres de tipo ethX han sido de este modo reemplazados por otros.

Si esto le molesta y quiere volver a la antigua regla de nomenclatura con eth, añada las opciones net.ifnames=0 y biosdevname=0 al arranque del núcleo para evitar la nomenclatura a través de BIOS. Deberá, para esto, modificar los parámetros por defecto de GRUB y regenerar su configuración, como se explica en el capítulo Inicio de Linux, servicios, núcleo y periféricos.

4. Configuración

a. Caso general

La configuración básica de una interfaz de red se hace con la ayuda del comando **ifconfig**. Sin embargo, la mayoría de las distribuciones utilizan scripts de administración y archivos de configuración que simplifican mucho las cosas, ya que configuran al mismo tiempo la interfaz de red y las rutas.

Configuración de eth0 para la dirección de clase C 192.168.1.2

```
ifconfig eth0 inet 192.168.1.2 netmask 255.255.255.0
```

o

```
ifconfig eth0 192.168.1.2
```

Activación de la interfaz de red eth0:

```
ifconfig eth0 up
```

Parada de la interfaz de red eth0:

```
ifconfig eth0 down
```

Visualización de información de eth0:

```
# ifconfig eth0
eth0      Vínculo encap:Ethernet  HWaddr 00:XX:XX:XX:XX:XX
          inet adr:192.168.1.60  Bcast:192.168.1.255  Máscara:255.255.255.0
          adr inet6: fe80::21b:fcff:fec9:f81d/64 Scope:Vínculo
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:16522 errors:0 dropped:0 overruns:0 frame:0
          TX packets:13631 errors:0 dropped:0 overruns:0 carrier:2
          collisions:0 lg file transmission:1000
          RX bytes:17732221 (16.9 MB)  TX bytes:1648879 (1.5 MB)
```

Observe que la dirección IPv6 aparece en el resultado en la línea inet6.

Configuración de una dirección IPv6

```
# ifconfig eth0 inet6 add fe80::21b:fcff:fec9:f81d/64
```

Visualización de todas las interfaces de red activadas:

```
ifconfig
```

Visualización de todas las interfaces de red, activadas o no:

```
ifconfig -a
```

b. Caso de las distribuciones de tipo Red Hat

Red Hat propuso, antes de la llegada de NetworkManager, herramientas para configurar la red básica sin pasar por la manipulación de los archivos de configuración. Estas herramientas desaparecieron después de la versión 7. Como información, encontrará más adelante los comandos para las versiones anteriores.

El comando **netconfig** funciona en modo texto y permite la configuración básica de TCP/IP (IP estática o dinámica, router, nombre de host, servidor DNS).

```
# netconfig --device eth0
```

El comando gráfico **system-config-network** inicia una interfaz muy completa.

La distribución SuSE (o SLES) toma el mismo principio expuesto anteriormente, pero la sintaxis y la ubicación de los archivos pueden variar. Debe consultar la documentación de su distribución para más detalles.

Lo mejor sigue siendo la utilización de los scripts **ifup** e **ifdown**. Se basan en los archivos presentes en /etc/sysconfig/network-scripts/. Estos archivos de configuración de interfaz se llaman ifcfg-xxx, donde xxx es el nombre de la interfaz de red, como eth0:

```
DEVICE=eth0
IPADDR=192.168.1.2
NETMASK=255.255.255.0
NETWORK=192.168.1.0
BROADCAST=192.168.1.255
ONBOOT=yes
BOOTPROTO=static
```

Los parámetros hablan por sí mismos. Los valores **NETWORK** y **BROADCAST** son opcionales si se notifican **IPADDR** y **NETMASK** (en este caso, el cálculo es automático) o si se utiliza **DHCP**. **BOOTPROTO** indica cómo montar la interfaz, o bien **static**, o bien **dhcp**. Se puede utilizar el valor **bootp**. En el caso de DHCP, el archivo se puede parecer a esto:

```
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=dhcp
```

En el caso de una configuración estática, **IPADDR** y **NETMASK** son obligatorios:

```
DEVICE=eth0
IPADDR=192.168.1.2
NETMASK=255.255.255.0
ONBOOT=yes
BOOTPROTO=static
```

ONBOOT determina si se debe activar automáticamente la interfaz en el arranque de la máquina.

Para IPv6, debe utilizar las siguientes variables:

```
IPV6INIT=yes
IPV6ADDR=<IPv6-IP-Address>
```

```
IPV6_DEFAULTGW=<IPv6-IP-Gateway-Address>
```

IPV6INIT indica si IPv6 está activo o no en esta interfaz. Las dos variables siguientes indican la dirección IP y su puerta de enlace.

Una vez notificado correctamente el archivo, se utilizan los comandos **ifup/ifdown**:

Activación de la interfaz eth0:

```
ifup eth0
```

Parada de la interfaz eth0:

```
ifdown eth0
```

Parámetros generales

El archivo /etc/sysconfig/network contiene los parámetros generales de la red.

```
NETWORKING=yes
HOSTNAME=puesto1.mired.org # nombre completo
GATEWAY=0.0.0.0 # pasarela por defecto
NISDOMAIN= # nombre del dominio NIS
NETWORKING_IPV6=yes
```

- **NETWORKING**: activación o no de la red.
- **HOSTNAME**: nombre de dominio completo FQDN.
- **GATEWAY**: dirección IP de la pasarela.
- **GATEWAYDEV**: interfaz de red que permite acceder a la pasarela.
- **NISDOMAIN**: en caso de un dominio NIS.
- **NETWORKING_IPV6**: activación o no del soporte de IPv6.

c. Máquinas de tipo Debian

El archivo de configuración de las interfaces de redes en Debian (y Ubuntu) se sitúa en /etc/network/interfaces. No tiene el mismo formato que en Red Hat.

```
# cat interfaces
auto lo eth0 eth1
iface lo inet loopback

iface eth0 inet static
        address 192.161.1.60
        netmask 255.255.255.0
        broadcast 192.168.1.255
        gateway 192.168.1.1

iface eth1 inet dhcp
```

Este ejemplo muestra tres tipos de interfaces:

- la interfaz lo de loopback,
- la interfaz eth1 en dhcp, que no necesita una configuración más avanzada,
- la interfaz eth0 configurada de manera estática.

La sintaxis general de una declaración es la siguiente:

```
interfaz nombre tipo modo
```

Con una configuración estática, especifique los diferentes parámetros con las palabras claves siguientes:

- **address**: la dirección IP.
- **netmask**: la máscara de subred.
- **broadcast**: la dirección de broadcast.
- **gateway**: la pasarela por defecto.

La línea **auto** indica las interfaces que se activarán automáticamente en el arranque.

Para IPv6 la configuración es idéntica: basta con reemplazar inet con inet6.

A partir de 2012 podemos emplear una directiva **include** que permite incluir un archivo externo a la configuración.

El archivo /etc/hostname contiene el nombre de la máquina:

```
# cat /etc/hostname
slyserver
```

d. Encaminamiento

Con la utilización de los archivos y comandos anteriores, no hace falta crear un encaminamiento específico, puesto que la pasarela por defecto está ya presente (vea Parámetros generales) e **ifup** implementa automáticamente las rutas para las interfaces. Sin embargo, se puede utilizar el comando **route**.

Visualiza las rutas actuales:

```
route
netstat -nr
```

En el ejemplo siguiente, las interfaces vmnet1 y vmnet8 son interfaces virtuales procedentes de la configuración de red de VMWare.

```
# netstat -rn
Tabla de encaminamiento IP del núcleo
Destino         Pasarela      Genmask         Indic   MSS Ventana irtt
Iface
192.168.211.0   0.0.0.0       255.255.255.0   U         0 0          0
vmnet8
192.168.1.0     0.0.0.0       255.255.255.0   U         0 0          0
eth0
172.16.248.0    0.0.0.0       255.255.255.0   U         0 0          0
vmnet1
169.254.0.0     0.0.0.0       255.255.0.0     U         0 0          0
eth0
127.0.0.0       0.0.0.0       255.0.0.0       U         0 0          0
lo
0.0.0.0         192.168.1.1   0.0.0.0         UG        0 0          0
eth0
```

Creación de la entrada loopback:

```
route add -net 127.0.0.0
```

Crea la ruta hacia la red 192.168.1.0, que pasa por eth0. Se puede omitir netmask.

```
route add -net 192.168.1.0 netmask 255.255.255.0 eth0
```

Crea la pasarela por defecto hacia el enrutador:

```
route add default gw 192.168.1.254
```

Suprime la ruta hacia la red 172.16.0.0:

```
route del -net 172.16.0.0 eth0
```

La gestión de rutas IPv6 es idéntica, pero indicando el tipo de dirección usado en el comando con la opción -A:

```
route -A inet6 add <ipv6> gw <ipv6>
```

e. iproute2

Los comandos **ifconfig** y **route** no son los únicos que permiten la asignación, la modificación o la eliminación de las direcciones IP y las rutas. Los comandos **iproute2** hacen eso y mucho más, en particular aquellos que ofrecen opciones de más bajo nivel y permiten operaciones más complejas. El comando principal se llama **ip**. Solo abordaremos las nociones básicas.

Los tres objetos, entre otros, que ip puede manipular son:

- **link**, gestiona los atributos de la interfaz de red.
- **addr**, gestiona la dirección IP de la interfaz de red.
- **route**, gestiona las rutas a las redes y hosts.

Para mostrar el estado de las interfaces de red a nivel de hardware (o virtual), utilice **link show**, y si es preciso la interfaz. Si aparece **NO CARRIER**, es que la interfaz no recibe nada. Si se muestra **state DOWN**, la interfaz no está conectada:

```
$ ip link show eth0
2: eth0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT qlen
1000
    link/ether 08:00:27:6b:d5:77 brd ff:ff:ff:ff:ff:ff
```

Para mostrar la configuración IP de una interfaz, utilice **addr show**; obtendrá la información de IPv4 e IPv6:

```
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
qlen 1000
    link/ether 08:00:27:a7:26:48 brd ff:ff:ff:ff:ff:ff
    inet 192.168.99.13/24 brd 192.168.99.255 scope global eth1
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fea7:2648/64 scope link
       valid_lft forever preferred_lft forever
```

Para cambiar la IP, utilice **addr add** según sigue;

```
# ip addr add 192.168.0.100/24 dev eth0
# ip addr show dev eth0
2: eth0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN qlen 1000
    link/ether 08:00:27:6b:d5:77 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.100/24 scope global eth0
       valid_lft forever preferred_lft forever
```

Para listar las rutas, utilice **route show**:

```
# ip route show
default via 192.168.0.254 dev eth0
169.254.0.0/16 dev eth0  scope link  metric 1002
169.254.0.0/16 dev eth1  scope link  metric 1003
192.168.0.0/24 dev eth0  proto kernel  scope link  src 192.168.0.19
192.168.99.0/24 dev eth1  proto kernel  scope link  src 192.168.99.13
```

Por último, para modificar la puerta de enlace por defecto, utilice **route change**:

```
# ip route change default via 192.168.99.1
# ip route show
default via 192.168.99.1 dev eth1
```