Seguridad y disponibilidad: RAID por software y backup diferencial con DAR usando GNU/Linux*

Mauro Silvosa Rivera

6 de junio de 2008 (Rev. 1)

Resumen

En un servidor de producción hay que garantizar la seguridad y la disponibilidad de los datos. Pero éstos, para variar, están sujetos a los dos principales peligros permanentes de la informática: los fallos hardware y los usuarios. Para reducir estos riesgos debemos establecer (1) sistemas de almacenamiento redundantes que nos eviten la pérdida de datos ante fallos hardware de discos duros, y (2) sistemas de backup lo suficientemente potentes como para permitir recuperar datos que los usuarios hayan modificado o eliminado por error a partir de copias anteriores. Las soluciones hardware orientadas a resolver estos problemas (RAID por hardware y robots de cintas de backup) son caras o muy caras. Pese a todo, con GNU/Linux dispondremos de los drivers, software y documentación necesarios para afrontar los anteriores problemas de forma eficiente y económica. En esta charla-taller mostraremos cómo montar un sistema RAID completo (centrándonos en los niveles 1 y 5) por software y un sistema de backup diferencial basado en DAR.

Parte I

RAID por software con GNU/Linux

1. Instalación del sistema base

Iniciamos la instalación del sistema base a partir del Network Installation Disk de Debian[1]. Llegado el momento, definimos en el primer disco las particiones que finalmente queramos usar. En este caso, por tratarse de un ejemplo, haremos las siguientes particiones:

Partición	Punto de montaje	Tamaño	Formato	Tipo	MD
/dev/sda1	/	2 GBytes	Ext3	Primary - Linux	md0
/dev/sda2	swap	512 MBytes	swap	Primary - swap	md1
/dev/sda3	/boot	30 MBytes	Ext2	Primary - Linux	md2
/dev/sda5	/home	2 GBytes	ReiserFS	Logical - Linux	md3
/dev/sda6	/tmp	1 GBytes	Ext2	Logical - Linux	md4
/dev/sda7	/dat	2 GBytes	RaiserFS	Logical - Linux	md5

Cuadro 1: Tabla de particiones definidas en el primer disco.

Como BootLoader usaremos LILO, de modo que cuando el software de instalación nos pregunte acerca de configurar GRUB le diremos que no lo haga y que continuamos nosotros indicando el siguiente paso. El siguiente paso será decirle que use LILO como BootLoader instalándolo en el MBR.

Una vez finalizado el proceso de instalación se nos pedirá que retiremos el CD de la unidad y el sistema se reiniciará para arrancar el sistema operativo recién instalado. Posteriormente terminaremos con la configuración mínima del sistema: contraseñas de root, configuración horaria, APT y configuración de los paquetes seleccionados.

Hecho esto, tendremos un servidor con una configuración mínima de Linux.

2. Recompilación del kernel

En el servidor que estamos configurando, todas las particiones serán arrays RAID por software. Esto quiere decir que incluso la partición /boot desde la que arranca el sistema también será un volumen RAID. Incluso el sistema de ficheros raíz es un volumen RAID. Por todo esto, tendremos que hacer que el kernel tenga el soporte para dispositivos MD RAID-1 compilado en el propio kernel y no como módulos ya que de otro modo no podrá montar la partición raíz para cargarlo.

Aprovechando la recompilación del kernel podemos personalizar más el kernel para ajustarlo al hardware del propio servidor. Es fundamental no olvidarse de los sistemas de ficheros usados en el arranque ni de las tarjetas SATA que estemos usando para acceder a los discos de arranque. El soporte para discos SCSI también debe estar presente.

[1] http://www.debian.org/CD/netinst/

Para seleccionar las opciones que queramos instalar (como módulos o como parte del kernel) usamos:

```
$cd /usr/src/linux-source-2.6.15
```

```
$ make menuconfig
```

Y para compilar el kernel y los módulos hacemos:

```
$ make-kpkg kernel-image
```

```
$ cd /usr/src
```

```
$ dpkg -i nombre_paquete.deb
```

Antes de reiniciar el servidor para arrancar con el nuevo kernel es apropiado revisar la configuración de `/etc/lilo.conf` para asegurarse de que tendremos la posibilidad de arrancar tanto con el kernel nuevo como con el anterior (por si falla el nuevo). Hay que prestar especial atención a los enlaces simbólicos `/vmlinuz` y `/vmlinuz.old` y a `/initrd.img.old` y comprobar que están bien definidos.

Cuando todo esté listo reiniciamos el servidor:

```
$reboot
```

Si todo ha ido bien, ahora mismo estaremos ejecutando un sistema operativo a medida de nuestro servidor y con soporte para dispositivos MD RAID-1.

3. Creación de los arrays RAID-1

Ahora es el turno de crear los arrays RAID. Para ello seguiremos los siguientes pasos:

3.1. Copia de la estructura de particiones del primer disco a los demás

Los demás discos implicados en el volumen RAID-1 que vamos a crear deben ser de igual o mayor capacidad que el primero y deben tener la misma estructura. Lo ideal es que sean discos iguales (incluso en marca y modelo) sobretodo para no tener que desperdiciar espacio en el disco de mayor tamaño. No tendría sentido que aprovechásemos ese espacio sobrante en otra partición porque los accesos a esa nueva partición enlentecerían los accesos a las particiones que se usan para construir los volúmenes RAID, que son nuestra prioridad.

La copia de la tabla de particiones de un disco en otro es tan sencilla como:

```
$ sfdisk -d /dev/sda | sfdisk /dev/sdb
```

```
$ sfdisk -d /dev/sda | sfdisk /dev/sdc
```

3.2. Linux RAID autodetect

Dado que queremos que el sistema arranque desde unidades MD RAID-1 tendremos que indicarle de alguna manera que durante el arranque las busque para que pueda crear el volumen y posteriormente

montarlas. Esto se hace indicando en las particiones que forman parte de los arrays RAID que son del tipo `FD` o lo que es lo mismo "Linux RAID Autodetect". Usando la utilidad `fdisk` podremos cambiar el tipo de cada partición. Hay que tener cuidado para no cambiar el tipo de la partición de tipo `Extended` que permite definir las particiones lógicas.

Inicialmente, este cambio de tipos en las particiones de todos los discos excepto el primero, es decir, excepto el que estamos usando actualmente, dado que el primero todavía lo necesitamos tal y como está dado que contiene el sistema con el que actualemente estamos trabajando.

3.3. Creando los arrays RAID-1

Para poder crear los arrays tendremos que tener instaladas las herramientas `mdadm`. Si no las tenemos instaladas las instalamos[2]:

```
$ apt-get install mdadm
```

Si ya las tenemos instaladas creamos los arrays de la siguiente manera:

```
$ mdadm --create /dev/md0 --verbose --level=1 --raid-devices=2
        --spare-devices=1 missing /dev/sdb1 missing

$ mdadm --create /dev/md1 --verbose --level=1 --raid_devices=2
        --spare-devices=1 missing /dev/sdb2 missing

$ mdadm --create /dev/md2 --verbose --level=1 --raid-devices=2
        --spare-devices=1 missing /dev/sdb3 missing

$ mdadm --create /dev/md3 --verbose --level=1 --raid-devices=2
        --spare-devices=1 missing /dev/sdb5 missing

$ mdadm --create /dev/md4 --verbose --level=1 --raid-devices=2
        --spare-devices=1 missing /dev/sdb6 missing

$ mdadm --create /dev/md5 --verbose --level=1 --raid-devices=2
        --spare-devices=1 missing /dev/sdb7 missing
```

Como podemos observar, inicialmente los arrays RAID-1 creados sólo tienen un disco de los tres que lo componen. El disco que falta todavía no lo podemos añadir porque lo estamos usando ahora en el sistema que se está ejecutando actualmente. Esto nos obligará a arrancar desde un volumen RAID compuesto de un disco en lugar de dos. Tampoco podemos añadir aún el disco de spare, porque de hacerlo entraría inmediatamente a cubrir el hueco del disco que falta para completar el RAID-1. Posteriormente, una vez arrancado el sistema sobre los volúmenes RAID, ya podremos añadir en caliente el disco que falta y el spare. En todo momento podemos ver el estado de todos los dispositivos MD con la orden:

```
$ cat /proc/mdstat
```

[2]Durante el proceso de instalación se nos preguntará si deseamos que los volúmenes se monten automáticamente y si queremos activar el demonio de monitorización. En ambos casos responderemos afirmativamente.

3.4. Creación de mdadm.conf

Aunque no es necesario, procederemos a la creación del fichero `/etc/mdadm/mdadm.conf` porque su existencia permite abreviar la longitud de los comandos `mdadm` que ejecutemos en el día a día.

```
$ mdadm --detail -scan > /etc/mdadm/mdadm.conf
```

3.5. Creación de los filesystems en los arrays

Una vez creados los arrays RAID, tendremos que crear en ellos los sistemas de ficheros correspondientes. En este caso:

```
$ mkfs.ext3 /dev/md0
$ mkswap /dev/md1
$ mkfs.ext2 /dev/md2
$ mkfs.ext3 /dev/md3
$ mkfs.ext3 /dev/md4
$ mkfs.ext3 /dev/md5
```

4. Reinicio

Aunque no es necesario reiniciar para proceder con los siguientes pasos, no es mala idea hacerlo para coprobar que lo que hemos hecho hasta ahora es correcto. Si reiniciamos la máquina tendremos que ver como el sistema detecta los dispositivos MD RAID. Tratará de buscar los dos discos por array y sólo verá uno, pero todo eso lo reflejará en los mensajes que aparecen en pantalla.

Como siempre, podremos ver el estado de los arrays con la siguiente orden:

```
$ cat /proc/mdstat
```

El resultado será un listado, que indica que los arrays están definidos y constituidos por un sólo de disco de los dos que deberían ser, parecido al siguiente:

```
Personalities : [raid1]
md1 : active raid1 sdb2[1]
      497920 blocks [1/2] [_U]

md2 : active raid1 sdb3[1]
      7920 blocks [1/2] [_U]

md3 : active raid1 sdb5[1]
      4883648 blocks [1/2] [_U]

md4 : active raid1 sdb6[1]
```

```
      29294400 blocks [1/2] [_U]

md5 : active raid1 sdb7[1]
      33680128 blocks [1/2] [_U]

md0 : active raid1 sdba1[1]
      9767424 blocks [1/2] [_U]
```

Si no podemos ver un listado similar, es que algo ha ido mal anteriormente y no podríamos continuar.

5. Preparación del sistema para arrancar desde los arrays RAID-1

Los últimos pasos de este procedimiento se centran en copiar el contenido de las particiones del primer disco (excepto área de swaping) en los arrays RAID-1 correspondientes. También se le dirá al sistema que arranque montando los arrays en lugar de las paticiones de disco y finalmente se añadirán las particiones del primer disco a los arrays RAID de forma que todo el sistema estará usando los arrays y éstos estarán compuestos por las dos unidades de disco duro en lugar de una. Para conseguirlo seguiremos los siguientes pasos[3]:

5.1. Edición de lilo.conf

Hay que actualizar el campo `root=` del fichero `/etc/lilo.conf` para indicarle que el dispositivo que a partir de ahora montaremos como raíz, ya no es `/dev/sda1` si no `/dev/md0` (el array RAID-1):

```
root=/dev/md0
```

5.2. Edición de fstab

Para que el sistema monte los arrays RAID en lugar de las particiones del primer disco, tenemos que editar el fichero `/etc/fstab` para que tenga un aspecto similar a este:

```
# /etc/fstab: static file system information.
#
# <file system> <mount point>   <type>  <options>                 <dump>  <pass>
proc            /proc           proc    defaults                    0       0
#
/dev/md0        /               ext3    defaults,errors=remount-ro  0       1
/dev/md5        /DATOS          ext3    defaults                    0       2
/dev/md4        /tmp            ext3    defaults                    0       2
```

[3]Es muy importante respetar el orden. De no hacerlo podríamos hacer que los datos copiados en los volúmenes RAID no fueran iguales a los que tienen las particiones del primer disco, el que está usando actualmente el sistema. Es decir, el último paso tiene que ser la replicación de datos sobre el volumen RAID.

```
/dev/md3        /home           ext3    defaults                    0       2
/dev/md2        /boot           ext3    defaults                    0       2
/dev/md1        none            swap    sw                          0       0
#
#/dev/sda1      /               ext3    defaults,errors=remount-ro  0       1
#/dev/sda7      /DATOS          ext3    defaults                    0       2
#/dev/sda6      /tmp            ext3    defaults                    0       2
#/dev/sda5      /home           ext3    defaults                    0       2
#/dev/sda3      /boot           ext3    defaults                    0       2
#/dev/sda2      none            swap    sw                          0       0
#
/dev/hda        /media/cdrom0   iso9660 ro,user,noauto               0       0
```

5.3. Replicando el primer disco en los arrays RAID

Para poder arrancar el sistema desde los arrays RAID tenemos que copiar el contenido de las particiones del primer disco (desde donde arrancamos ahora) en los arrays RAID. Para ello hacemos lo siguiente:

Para la partición raíz /.

```
$ cd /
$ mount /dev/md0 /mnt
$ find . -xdev | cpio -pm /mnt
```

Para /boot

```
$ cd /boot
$ mount /dev/md2 /mnt
$ find . -xdev | cpio -pm /mnt
```

Para /home

```
$ cd /home
$ mount /dev/md3 /mnt
$ find . -xdev | cpio -pm /mnt
```

Para /tmp

```
$ cd /tmp
$ mount /dev/md4 /mnt
$ find . -xdev | cpio -pm /mnt
```

Y para /dat

```
$ cd /dat
```

```
$ mount /dev/md5 /mnt
$ find . -xdev | cpio -pm /mnt
```

Ahora, al sistema le debería dar igual montar los arrays RAID-1 que las particiones del primer disco que estamos usando actualmente.

5.4. Reinicio

Tenemos que reiniciar si queremos que el sistema entero se monte sobre los arrays RAID. Si todo va bien, la partición raíz, la zona de swapping, /boot, /tmp y /var serán montados desde los arrays RAID (que todavía, no lo olvidemos, están compuestos de un sólo disco).

Antes de reiniciar **no podemos olvidarnos** de ejecutar LILO para aplicar los cambios realizados anteriormente:

```
$ lilo
```

6. Completar los arrays RAID-1 con el resto de discos

Para ello seguiremos los siguientes pasos:

6.1. Linux Raid Autodetect

Para que el sistema pueda buscar las particiones del primer disco que son candidatas a formar parte de un volumen RAID-1 tenemos que marcarlas con tipo FD (Linux RAID Autodetect). Para ello seguimos las mismas instrucciones indicadas en la sección 3.2, pero aplicadas al primer disco.

6.2. Añadir el primer disco al volumen RAID-1

De nada vale haber hecho todo esto si al final nuestros arrays RAID sólo están formados por particiones de un sólo disco. Tenemos que añadir las particiones correspondientes del primer disco para disponer de la redundancia y de todas las ventajas e inconvenientes de RAID-1. Para ello:

```
$ mdadm --manage /dev/md0 --add /dev/sda1
$ mdadm --manage /dev/md1 --add /dev/sda2
$ mdadm --manage /dev/md2 --add /dev/sda3
$ mdadm --manage /dev/md3 --add /dev/sda5
$ mdadm --manage /dev/md4 --add /dev/sda6
$ mdadm --manage /dev/md5 --add /dev/sda7
```

Al hacer esto el sistema se pondrá a sincronizar las particiones de los dos discos con la información del volumen RAID[4]. Estas operaciones se harán simultáneamente siempre y cuando las particiones

[4]Se puede dar el caso de que al sincronizar las particiones raíz (/dev/sda1 y /dev/sdb1) en el volumen RAID se pierdan

que intervienen no estén en los mismos discos. En general, tendremos que esperar a que acabe de sincronizarse un array para que empiece el siguiente. Podemos ver el estado de la sincronización y de los arrays con la orden:

```
$ cat /proc/mdstat
```

Al finalizar la sincronización, la orden anterior debería producir un resultado como el siguiente, que indica que todos los arrays están activos y formados por dos discos de un total de dos:

```
Personalities : [raid1]
md1 : active raid1 sdb2[0] sda2[1]
      497920 blocks [2/2] [UU]

md2 : active raid1 sdb3[0] sda3[1]
      7920 blocks [1/2] [UU]

md3 : active raid1 sdb5[0] sda5[1]
      4883648 blocks [2/2] [UU]

md4 : active raid1 sdb6[0] sda6[1]
      29294400 blocks [2/2] [UU]

md5 : active raid1 sdb7[0] sda7[1]
      33680128 blocks [2/2] [UU]

md0 : active raid1 sdb1[0] sda1[1]
      9767424 blocks [2/2] [UU]

unused devices: <none>
```

6.3. Añadir los discos de reserva (spare)

Ahora añadimos los discos de reserva al volumen RAID-1[5]:

```
$ mdadm --manage /dev/md0 --add /dev/sdc1
$ mdadm --manage /dev/md1 --add /dev/sdc2
$ mdadm --manage /dev/md2 --add /dev/sdc3
```

los cambios realizados en las secciones 5.2 y 5.1 si se ha cambiado el orden de alguno de los pasos indicados. En este caso habría que volver a hacer esos cambios, pero eso puede no ser tan inmediato como parece (hay que tener en cuenta que ya no tenemos montadas esas particiones como raíz) y es muy probable que tengamos que acceder a esos sistemas de ficheros desde una versión live (o desde el CD de instalación en red de Debian o quizá, habría que probarlo, desde el kernel anterior que sólo montaba la partición raíz desde el primer disco) para poder hacer esos cambios directamente en las particiones sin pasar por el array RAID. Otro fallo frecuente es olvidarse de ejecutar LILO después del cambio realizado en 5.1. En este caso tendremos que cargar un Linux desde una versión live, montar el sistema de ficheros raíz, hacer un chroot y ejecutar LILO desde ahí para que se hagan los cambios en ese disco. Todo un lío, por lo que es mejor hacer las cosas bien a la primera.

[5]Este paso se podría haber realizado en la misma instrucción en la que añadimos el primer disco al array: mdadm –manage /dev/mdX –add /dev/sdaX /dev/sdcX, siendo /dev/sdc el disco de spare.

```
$ mdadm --manage /dev/md3 --add /dev/sdc5
$ mdadm --manage /dev/md4 --add /dev/sdc6
$ mdadm --manage /dev/md5 --add /dev/sdc7
```

7. Puesta en marcha

Para terminar necesitamos duplicar el MBR. De este modo, aunque falle uno de los dos discos seguiremos teniendo la capacidad de reiniciar la máquina.

7.1. Duplicación del MBR

Hasta ahora tenemos duplicados los datos en dos unidades de disco físicos en los que creamos varios volúmenes RAID-1 lógicos. Si un disco se para podremos seguir leyendo y escribiendo los datos como si nada hubiese pasado. ¿Pero qué ocurre si reiniciamos el ordenador y se da la circunstancia de que el disco estropeado era que que se encargaba de arrancar el sistema? En ese caso habríamos perdido el sector de arranque (MBR)[6]. Por tanto, también deberíamos tener duplicado el sector de arranque para que siempre podamos arrancar el sistema independientemente del disco estropeado.

Para solucionar esto simplemente duplicaremos el sector de arranque haciendo lo siguiente[7]:

- Editamos /etc/lilo.conf para que refleje: `boot=/dev/md2` y `raid-extra-boot=mbr`
- Ejecutamos LILO para aplicar los cambios: `$ lilo -v`

7.2. Reinicio

Para asegurarnos de que todo ha ido bien podemos reiniciar de nuevo. No es necesario, pero no está de más. Posteriormente deberíamos probar incluso que todo funciona bien incluso cuando quitamos un disco. También deberíamos probar a quitar uno y añadir otro sobre la marcha para probar que funciona la resincronización (mdadm –manage /dev/mdX –re-add /dev/sdYX), etc, etc.

8. Probando el sistema

Una vez finalizado el proceso de configuración del array RAID-1 deberemos probar que funciona correctamente. Como ejemplo mostramos unas cuantas pruebas que se pueden hacer, pero hay más. Incluso se pueden hacer paradas e insercciones de discos en caliente (**NO con discos IDE**) para ver que todo va bien.

[6] http://en.wikipedia.org/wiki/MBR

[7] En caso de emergencia siempre podemos recurrir a copiar el MBR de un disco al otro usando $ dd if=/dev/sda of=/dev/sdb bs=512 count=1, aunque esto puede hacer que LILO se queje en el arranque con un error "Duplicate ID"que queda ahí, pero que no impide el arranque.

8.1. Primera prueba

Consistente en:

- Shutdown
- Retirar el primer disco
- Arrancar de nuevo

Si todo va bien, el sistema no debería haber perdido la capacidad de arrancar. Además, como teníamos un disco de spare, éste será usado para ocupar el "hueco" dejado por el disco que hemos quitado. El sistema comenzará a sincronizarse (esto puede hacer que el arranque sea muy lento) y cuando termine volveremos a tener dos discos en el volumen RAID-1 (evidentemente no tendremos disco de spare).

8.2. Segunda prueba

Consistente en:

- Shutdow
- Volvemoa a añadir el primer disco
- Arrancar el sistema
- Añadir de nuevo el disco de spare y resincronizar el primer disco:

  ```
  $ mdadm --manage /dev/mdX --re-add /dev/sdaY /dev/sdcY
  ```

8.3. Tercera prueba

Consistente en:

- Marcar un disco (no spare) como `faulty`

  ```
  $ mdadm --manage /dev/md4 --set-faulty /dev/sdb6
  ```

- Quitar el disco marcado como `faulty` del volumen RAID-1

  ```
  $ mdadm --manage /dev/md4 --remove /dev/sdb6
  ```

- Marcar de nuevo un disco como `faulty`

  ```
  $ mdadm --manage /dev/md4 --set-faulty /dev/sdc6
  ```

- Quitar este nuevo disco marcado como `faulty`

  ```
  $ mdadm --manage /dev/md4 --remove /dev/sdc6
  ```

- Volver a añadir al volumen los dos discos que hemos quitado con una sóla operación

```
$ mdadm --manage /dev/md4 --re-add /dev/sdb6 --add /dev/sdc6
```

Nada más marcar como faulty el disco, el sistema debería detectarlo al cabo de un rato y debería ocupar el hueco dejado por el disco que ha fallado con el disco de spare. Al marcar el segundo disco como faulty, el sistema siguió funcionando, pero con un sólo disco en el volumen RAID-1. De haberse producido otro fallo hubiésemos perdido información, por eso es importante añadir de nuevo los discos tan pronto podamos.

9. Ejemplo. Creación de un array RAID-5

Para finalizar, mostramos lo sencillo que es crear un array RAID-5 con disco de spare. Para facilitar un poco las cosas y no tener que tocar las tablas de particiones de los discos existentes haremos lo siguiente: copiaremos la estructura de los discos en el cuarto disco y convertiremos el actual volumen RAID-1 correspondiente al disco /dat en un volumen RAID-5.

9.1. Crear las particiones del disco de spare

Del mismo modo que particionamos los discos para el volumen RAID-1, lo hacemos ahora para el cuarto disco, que será el disco de spare del RAID-5:

```
$ sfdisk -d /dev/sda | sfdisk /dev/sdd
```

9.2. Eliminar el volumen RAID-1 de /dat

Como dijimos, vamos a convertir el volumen RAID-1 de /dat en un volumen RAID-5. Para ello debemos desmontar el volumen de su punto de montaje en el sistema y posteriormente deshacer el volumen RAID-1 para que libere los recursos que empleaba:

```
$ umount /dat
$ mdadm --manage /dev/md5 --stop
```

9.3. Creación del nuevo volumen RAID-5

De un sólo paso podemos crear el volumen RAID-5 indicándole cuántos discos debe usar tanto para el volumen como para los discos de spare:

```
$ mdadm --create /dev/md5 --verbose --level=5 --raid-devices=3
        --spare-devices=1 /dev/sda7 /dev/sdb7 /dev/sdc7 /dev/sdd7
```

El resultado de esta operación puede verse inmediatamente consultando /proc/mdstat:

```
md5 : active raid5 sdc7[2] sdd7[3](S) sdb7[1] sda7[0]
      7807360 blocks level 5, 64k chunk, algorithm 2 [3/3] [UUU]
```

9.4. Crear el sistema de ficheros y montar el nuevo array en /dat

Para finalizar, creamos el sistema de ficheros en el volumen RAID-5 y posteriormente lo montamos en el punto de montaje del sistema correspondiente:

```
$ mkfs.ext3 /dev/md5
```

```
$ mount /dat
```

Parte II

Backup con GNU/Linux usando DAR

10. Introducción

10.1. ¿Qué es DAR?

DAR[8] es un programa de línea de comando creado por Denis Corbin que permite realizar backup de directorios y/o ficheros. Está probado en múltiples plataformas y posee una API que deja abiertas las puertas a desarrollos de terceros (como Kdar, una aplicación gráfica para gestionar backups DAR). Se distribuye bajo licencia GPL (GNU General Public License) y lo podemos encontrar empaquetado en múltiples distribuciones, entre ellas en Debian GNU/Linux. Su instalación se reduce a:

```
$ apt-get install dar dar-docs
```

10.2. Uso básico de DAR

La forma más sencilla de usar DAR es la de generar un archivo de backup para una serie de ficheros y/o directorios:

```
$ dar -c nombre_fichero_backup -g dir_1 ... -g dir_n -g file_1 ... -g file_n
```

Esto genera un archivo llamado nombre_fichero_backup.1.dar que contiene los ficheros que le hemos indicado. En cualquier momento podemos ver qué ficheros han sido guardado consultando el catálogo del backup. Esto se hace de la siguiente manera:

```
$ dar -l nombre_fichero_backup
```

Nótese que pese a que DAR indica con un número el orden de los ficheros que componen el backup, nosotros sólo usamos el nombre del backup, no de los ficheros. No indicamos ni el número de fichero ni la extensión. El motivo de que aparezca un número en el nombre se debe a que DAR nos permite fijar un tamaño máximo a cada fichero (por ejemplo 600MB para que quepa en un CD-ROM), por tanto, en el caso de que el backup no quepa en un sólo fichero (slice en terminología de DAR) habrá que marcarlos de alguna manera para saber en qué orden se generaron.

Para recuperar los datos a partir de la copia de seguridad, simplemente tenemos que hacer lo siguiente:

```
$ cd ~

$ mkdir tmp

$ cd tmp

$ dar -x ../nombre_fichero_backup
```

Esto dejará en en tmp/ los ficheros y directorios contenidos en el backup tal y como estaban en el

[8] http://dar.sourceforge.net

momento en que se realizó la copia.

11. Backup diferencial usando dar

11.1. ¿Qué es un backup diferencial?

Además de realizar el backup sencillote explicado anteriormente, DAR permite hacer un nuevo backup de los datos de una serie de ficheros y/o directorios considerando sólo aquellos que hayan cambiado en relación a otro backup tomado como referencia.

Esta estrategia de backup se conoce como backup diferencial (o incremental) y tiene dos grandes ventajas: 1) ahorro de espacio en disco y 2) velocidad. Con esta estrategia ocupamos poco espacio porque sólo copiamos las diferencias de los datos, haciendo que sólo uno de los backups (el de referencia) sea realmente grande. Y es rápida porque lleva muchísimo menos tiempo hacer copia de las diferencias de los datos que de todos los datos.

11.2. Planificación del backup

Antes de proceder a realizar un backup debemos preguntarnos lo siguiente:

- ¿Qué datos queremos copiar?
- ¿Con qué frecuencia y a qué horas?
- ¿Cuánto tiempo tengo que conservar las copias?
- ¿Tengo espacio para almacenar esas copias?

11.3. Ejecución del backup

Generalmente, las líneas de comando que generamos para llamar a DAR son bastante largas, llenas de parámetros de configuración y de carpetas a incluir o excluir del backup, etc. Para facilitar esta tarea es aconsejable escribir unos scripts que se encarguen de:

- crear un backup de referencia o diferencial según corresponda
- con los parámetros de configuración de DAR oportunos
- copiar en él los ficheros y/o directorios que nos interesan
- chequear que los backups se han llevado a cabo sin problemas

Podemos encontrar buenos ejemplos y documentación en:

- La documentación del proyecto DAR (apt-get install dar-docs)

- http://dar.sourceforge.net/doc/samples/index.html
- http://gradha.sdf-eu.org/textos/dar-differential-backup-mini-howto.es.html

Para este ejemplo usaremos un par de scripts muy simples: `dar_master.sh` y `dar_diff.sh`, que se encargarán respectivamente de realizar la copia maestra y las de diferencias cuando correspondan[9].

- `dar_master.sh`

```
#!/bin/sh

# SOURCE = Esto es el directorio del que queremos hacer backup.
SOURCE=/DATOS/SHARES/dsd_mauro
# DIR = Es el directorio en donde guardamos los ficheros del backup.
DIR=/DATOS/BACKUP
SLICE_NAME=${DIR}/backup_`/bin/date -I`_master
SLICE_SIZE=600M
#
mv $DIR/ANT $DIR/ANT.TMP
mkdir $DIR/ANT
mv $DIR/*.dar $DIR/ANT

# Añadir "-y" para que comprima.
/usr/bin/dar -m 256 -s $SLICE_SIZE -D -R $SOURCE -c $SLICE_NAME -Z "*.gz" \
        -Z "*.bz2" -Z "*.zip" -Z "*.png" -Z "*.jpg" -Z "*.rar" -Z "*.tgz"

/usr/bin/dar -t $SLICE_NAME
/usr/bin/find $DIR -type f -exec chmod 440 \{\} \;
/usr/bin/find $DIR -type d -exec chmod 550 \{\} \;

rm -rf $DIR/ANT.TMP
```

- `dar_diff.sh`

```
#!/bin/bash

# SOURCE = Esto es el directorio del que queremos hacer backup.
SOURCE=/DATOS/SHARES/dsd_mauro
# DIR = Es el directorio en donde guardamos los ficheros del backup.
DIR=/DATOS/BACKUP
```

[9]Importante. Este sistema tiene un inconveniente: si ocurre algo en el servidor que impida que se ejecute dar_master.sh el día en que está planeado, habremos perdido la capacidad de hacer copias diferenciales por parte de dar_diff.sh, ya que no tendrán una copia de referencia. Este problema se soluciona haciendo que dar_diff.sh compruebe que existe la copia master antes de realizar las de diferencias. En http://dar.sourceforge.net/doc/samples/cluster_digital_backups.sh podemos encontrar una solución más elegante a este problema.

```
PREV=‘/bin/ls $DIR/*.dar|/usr/bin/tail -n 1|/usr/bin/awk -F ’.’ ’{print $1;}’‘

SLICE_NAME=${DIR}/backup_‘/bin/date -I‘_diff
SLICE_SIZE=600M

# Añadir "-y" para que comprima.
/usr/bin/dar -m 256 -s $SLICE_SIZE -D -R $SOURCE -c $SLICE_NAME -Z "*.gz" \
        -Z "*.bz2" -Z "*.zip" -Z "*.png" -Z "*.jpg" -Z "*.rar" -Z "*.tgz" \
        -A $PREV

/usr/bin/dar -t $SLICE_NAME
/usr/bin/find $DIR -type f -exec chmod 440 \{\} \;
/usr/bin/find $DIR -type d -exec chmod 550 \{\} \;
```

Y para que esto tenga sentido y no sea algo de lo que estar preocupándonos, es necesario que el proceso de realización de backups sea completamente automático, por tanto tendremos que configurar cron para que se encargue de llamar a los scripts según corresponda para alcanzar los objetivos marcados en la planificación del backup. En este sentido, si le decimos a cron que haga la copia master con mucha frecuencia estaremos necesitando más capacidad de almacenamiento (salvo que eliminemos las copias anteriores). También con cron marcamos la frecuencia con la que queremos realizar copias diferenciales y a qué horas se realizan (no deben coincidir con las horas en las que nuestros usuarios usan la máquina). Una configuración habitual es la siguiente:

```
SHELL=/bin/bash
PATH=/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:/usr/games
MAILTO="mauro"

30 4 2-31 * * /root/scripts/backup/dar_diff.sh
30 4 1    * * /root/scripts/backup/dar_master.sh
```

Que significa que el día 1 de cada mes generamos una copia master que usaremos como referencia para los backups diferenciales del resto de días del mes.

Si por ejemplo, configuramos dar_master.sh para mantener una copia del backup del mes anterior, podremos recuperar, en todo momento, datos del mes anterior más datos del mes en curso hasta el día actual.

11.4. Recuperación de datos

Diferentes situaciones van a necesitar diferentes soluciones. Las situaciones que se nos van a poder presentar son las siguientes:

1. Hemos perdido todo y queremos recuperar todos los datos que teníamos en el backup en el momento de la pérdida. Esto es una recuperación completa. No tenemos nada, salvo las copias de

seguridad y queremos restaurarlas. Para ello tendremos que ir "desempaquetando" cada copia de seguridad una a una y en orden. Así, primero extraeremos los ficheros y directorios del backup de referencia y luego de cada uno de los backups diferenciales. En todo momento debemos seguir el orden de creación y debemos pasarle a DAR la opción -w que indica que sobreescriba los cambios en cada fichero y directorio que tenga que actualizar.

Con el siguiente script se puede automatizar esta tarea:

```
#!/bin/sh

if [ -n "$3" ]; then
   INPUT="$1_master"
   FS_ROOT="$2"
   dar -x "$INPUT" -w -R "$FS_ROOT"
   for file in ${INPUT:0:8}*_diff*; do
      dar -x "${file:0:15}" -w -R "$FS_ROOT"
   done
   echo "All done."
else
   echo "Not enough parameters.

   Usage: script master_full_backup destination_dir

   Where master_full_backup is a date in the format 'YYYY-MM-DD', and
   destination_dir is the place where you want to put the restored data."
fi
```

2. Hemos perdido o un usuario ha hecho cambios que ahora no quiere en una serie de datos (ficheros, o directorios) y queremos recuperar esos datos tal y como estaban en una determinada fecha anterior a la actual.

En este caso tenemos que realizar la tediosa operación de buscar en los diferenes ficheros del backup cuál es el que tiene una copia de ese fichero con la fecha que más se acerque a la fecha en la que queremos recuperar el fichero.

Esto se hace consultando el catálogo de cada copia de seguridad hasta encontrar lo que buscamos.

```
$ dar -l nombre_copia_seguridad
```

Una vez en contrado sólo hay que extraer el fichero o directorio en cuestión.

```
$ dar -x nombre_copia_seguridad -g nombre_fichero_a_recuperar
```

Parte III

Referencias

12. Enlaces de interés y bibliografía

- man mdadm
- man dar
- http://neil.brown.name/blog/mdadm
- http://cgi.cse.unsw.edu.au/~neilb/SoftRaid
- http://www.cse.unsw.edu.au/~neilb/source/mdadm/
- http://dar.sourceforge.net/
- http://www.tldp.org/HOWTO/Software-RAID-HOWTO.html
- http://unthought.net/Software-RAID.HOWTO/Software-RAID.HOWTO.spanish.html
- http://wiki.clug.org.za/wiki/RAID-1_in_a_hurry_with_grub_and_mdadm
- http://gentoo-wiki.com/Talk:HOWTO_Gentoo_Install_on_Software_RAID
- http://gradha.sdf-eu.org/textos/dar-differential-backup-mini-howto.en.html
- http://en.wikipedia.org/wiki/RAID