Optimizar el acceso a los dispositivos de almacenamiento

Linux puede gestionar una gran variedad de dispositivos de almacenamiento, de tipo local o remoto. Según el tipo de dispositivo y su uso, puede ser necesario modificar la configuración de acceso al dispositivo para optimizar su rendimiento. Existen muchas herramientas y comandos que pueden ayudar al administrador a configurar y monitorear los recursos del almacenamiento de los sistemas.

1. Gestión de los discos duros locales

El administrador debe poder determinar los periféricos locales de almacenamiento reconocidos por el sistema, listar sus características y, eventualmente, modificar su configuración para mejorar el rendimiento de las entradas/salidas o corregir conflictos. Debe poder, también, determinar cuáles son los tipos de periféricos de almacenamiento adaptados al sistema, en función de los pilotos instalados y, si fuera necesario, saber instalar nuevos pilotos.

a. Determinación de los archivos especiales

En Linux, cada periférico de almacenamiento está gestionado por un archivo especial en modo de bloque, asociado al piloto cargado para ese tipo de periférico. La detección de los periféricos de almacenamiento y la creación de los archivos especiales correspondientes se hace, generalmente, de manera automática y durante el arranque del sistema (excepto para los dispositivos extraíbles, detectados automáticamente durante la actividad del sistema).

Esos archivos especiales se encuentran en el directorio `/dev`. Su nombre está determinado por el núcleo, según algunas convenciones de nombres, en función del tipo de dispositivo y del orden de detección del dispositivo con respecto a su tipo.

Un archivo especial Linux no es un archivo de disco, no contiene ningún dato almacenado. En realidad se trata de un punto de comunicación con el piloto del dispositivo, gestionado por el núcleo. Cuando una aplicación «escribe» en el archivo especial, el núcleo transmite

los bytes emitidos por la aplicación al piloto del dispositivo en cuestión. A la inversa, cuando una aplicación «lee» el archivo especial, el núcleo le transmite los bytes emitidos por el piloto de dispositivo en cuestión.

Cada archivo especial Linux posee un número mayor (*major*) y un número menor (*minor*). El núcleo hace la conexión entre el archivo especial y el piloto del dispositivo gracias al número mayor del archivo. El número menor del archivo es un dato transmitido por el núcleo al piloto y su significado depende del tipo de piloto.

Ejemplo

En un sistema Linux con un solo disco:

```
ls -l /dev/sd*
brw-rw---- 1 root disk 8, 0 marzo  12 13:14 /dev/sda
brw-rw---- 1 root disk 8, 1 marzo  12 13:14 /dev/sda1
brw-rw---- 1 root disk 8, 2 marzo  12 13:14 /dev/sda2
brw-rw---- 1 root disk 8, 5 marzo  12 13:14 /dev/sda5
```

Se encuentran cuatro archivos especiales en modo de bloque (primera letra de la línea: `b`), que corresponden al mismo disco (`/dev/sda`) y a sus tres particiones (`/dev/sda1`, `sda2` y `sda5`). El mayor de cada archivo es `8`, todos son gestionados por el mismo piloto. El menor hace que el piloto puede distinguir los objetos, `0` para el disco entero de número `0`, `0+1`, `0+2` y `0+5` para las particiones del disco 0.

b. El servicio udev

El problema de los archivos especiales asociados a los dispositivos por el núcleo es que no son perennes. De hecho, el nombre del archivo especial depende del orden de detección de los dispositivos hecha por el núcleo. Por ejemplo, el nombre del archivo especial asociado a un dispositivo USB puede cambiar en función de los dispositivos USB presentes en el momento de la detección por el núcleo.

El servicio `udev` se encarga de interactuar con el núcleo y de crear dinámicamente enlaces simbólicos en el directorio `/dev` hacia los archivos especiales de los dispositivos conectados al sistema. El objetivo es poder acceder a los dispositivos gracias a nombres más explícitos y, sobre todo, perennes,

El servico `udev` usa un conjunto de archivos de configuración, por defecto en el directorio `/etc/udev/rules.d`, donde están especificadas las reglas que permiten definir el nombre asociado a cada dispositivo. El administrador puede modificar los archivos existentes para definir sus propias reglas.

La sintaxis de declaración de las reglas es compleja y debe ser manipulada con prudencia.

c. Interactuar con el servicio udev

El servicio `udev` está asegurado por uno de los procesos de systemd, `/usr/lib/systemd/systemd-udevd`. Se puede interactuar con ese proceso gracias al comando `udevadm`.

Para obtener datos detallados de un dispositivo, se puede usar el subcomando `info` del comando `udevadm`.

Sintaxis

```
udevadm info [ -q | --query=TipoInfo ] ArchivoEspecial
```

Parámetros principales

`-q | --query=TipoInfo`	Tipo de información.
`ArchivoEspecial`	Archivo especial del dispositivo.

Descripción

El comando ofrece diferentes características en el periférico asociado al archivo especial `ArchivoEspecial`, incluyendo los enlaces simbólicos creados para él en los sub

directorios de `/dev/disk`.

El parámetro `TipoInfo` puede tener como valor:

`name`	Nombre del nodo.
`symlink`	Enlace simbólico hacia el nodo.
`path`	Camino de acceso del dispositivo en `/sys`.
`property`	Propiedades del periférico.
`all`	Toda la información.

Ejemplo

Información del disco duro asociado al archivo especial de bloque `/dev/sda`:

```
devadm info --query=all /dev/sda
P: /devices/pci0000:00/0000:00:1f.2/ata1/host1/target1:0:0/1:0:0:0/block/sda
N: sda
S: disk/by-id/ata-ST320LT020-9YG142_W043X12J
S: disk/by-id/wwn-0x5000c500493da068
S: disk/by-path/pci-0000:00:1f.2-ata-1
E: DEVLINKS=/dev/disk/by-id/wwn-0x5000c500493da068 /dev/disk/by-path/
pci-0000:00:1f.2-ata-1 /dev/disk/by-id/ata-ST320LT020-9YG142_W043X12J
E: DEVNAME=/dev/sda
E: DEVPATH=/devices/pci0000:00/0000:00:1f.2/ata1/host1/target1:0:0/1:0:0:0/block/sda
E: DEVTYPE=disk
E: ID_ATA=1
E: ID_ATA_DOWNLOAD_MICROCODE=1
E: ID_ATA_FEATURE_SET_APM=1
E: ID_ATA_FEATURE_SET_APM_CURRENT_VALUE=128
E: ID_ATA_FEATURE_SET_APM_ENABLED=1
E: ID_ATA_FEATURE_SET_PM=1
```

```
E: ID_ATA_FEATURE_SET_PM_ENABLED=1
E: ID_ATA_FEATURE_SET_SECURITY=1
E: ID_ATA_FEATURE_SET_SECURITY_ENABLED=0
E: ID_ATA_FEATURE_SET_SECURITY_ENHANCED_ERASE_UNIT_MIN=78
E: ID_ATA_FEATURE_SET_SECURITY_ERASE_UNIT_MIN=78
E: ID_ATA_FEATURE_SET_SECURITY_FROZEN=1
E: ID_ATA_FEATURE_SET_SMART=1
E: ID_ATA_FEATURE_SET_SMART_ENABLED=1
E: ID_ATA_ROTATION_RATE_RPM=5400
E: ID_ATA_SATA=1
E: ID_ATA_SATA_SIGNAL_RATE_GEN1=1
E: ID_ATA_SATA_SIGNAL_RATE_GEN2=1
E: ID_ATA_WRITE_CACHE=1
E: ID_ATA_WRITE_CACHE_ENABLED=1
E: ID_BUS=ata
E: ID_MODEL=ST320LT020-9YG142
E: ID_MODEL_ENC=ST320LT020-9YG142\x20\x20\x20\x20\x20\x20\x20\x20\x20\x20\x20\x20\x20\
x20\x20\x20\x20\x20\x20\x20\x20\x20\x20\x20
E: ID_PART_TABLE_TYPE=dos
E: ID_PART_TABLE_UUID=12deb3a0
E: ID_PATH=pci-0000:00:1f.2-ata-1
E: ID_PATH_TAG=pci-0000_00_1f_2-ata-1
E: ID_REVISION=0002HPM1
E: ID_SERIAL=ST320LT020-9YG142_W043X12J
E: ID_SERIAL_SHORT=W043X12J
E: ID_TYPE=disk
E: ID_WWN=0x5000c500493da068
E: ID_WWN_WITH_EXTENSION=0x5000c500493da068
E: MAJOR=8
E: MINOR=0
E: SUBSYSTEM=block
E: TAGS=:systemd:
E: USEC_INITIALIZED=23759782
```

Se trata de un disco SATA, en el bus PCI, mayor `8` y menor `0`.

También se puede seguir la actividad del servicio `udev`, usando el subcomando `monitor` del comando `udevadm`. El comando muestra sobre la marcha los eventos detectados por el servicio.

En las antiguas versiones de distribuciones, el comando se llamaba `udevmonitor`.

Ejemplo

Se ejecuta el comando `udevadm monitor`, y después conectamos una memoria USB:

```
udevadm monitor
monitor will print the received events for:
UDEV - the event which udev sends out after rule processing
KERNEL - the kernel uevent

KERNEL[2225.947554] add      /devices/pci0000:00/0000:00:1a.0/usb1/1-1/1-1.3 (usb)
KERNEL[2225.953573] add      /devices/pci0000:00/0000:00:1a.0/usb1/1-1/1-1.3/
1-1.3:1.0 (usb)
KERNEL[2225.953860] bind     /devices/pci0000:00/0000:00:1a.0/usb1/1-1/1-1.3 (usb)
UDEV  [2226.418775] add      /devices/pci0000:00/0000:00:1a.0/usb1/1-1/1-1.3 (usb)
KERNEL[2226.492426] add      /module/usb_storage (module)
KERNEL[2226.493561] add      /devices/pci0000:00/0000:00:1a.0/usb1/1-1/1-1.3/
1-1.3:1.0/host6 (scsi)
KERNEL[2226.493600] add      /devices/pci0000:00/0000:00:1a.0/usb1/1-1/1-1.3/
1-1.3:1.0/host6/scsi_host/host6 (scsi_host)
KERNEL[2226.493639] bind     /devices/pci0000:00/0000:00:1a.0/usb1/1-1/1-1.3/
1-1.3:1.0 (usb)
KERNEL[2226.493657] add      /bus/usb/pilotos/usb-storage (pilotos)
UDEV  [2226.494931] add      /module/usb_storage (module)
UDEV  [2226.496511] add      /bus/usb/pilotos/usb-storage (pilotos)
KERNEL[2226.506905] add      /module/uas (module)
KERNEL[2226.506948] add      /bus/usb/pilotos/uas (pilotos)
[...]
_generic)
KERNEL[2227.863795] add      /devices/virtual/bdi/8:16 (bdi)
UDEV  [2227.866095] add      /devices/virtual/bdi/8:16 (bdi)
KERNEL[2227.872788] add      /devices/pci0000:00/0000:00:1a.0/usb1/1-1/1-1.3/
1-1.3:1.0/host6/target6:0:0/6:0:0:0/block/sdb (block)
UDEV  [2227.964430] add      /devices/pci0000:00/0000:00:1a.0/usb1/1-1/1-1.3/
1-1.3:1.0/host6/target6:0:0/6:0:0:0/block/sdb (block)
```

CTRL/C

Se vuelve a lanzar el comando y se desconecta la memoria USB:

```
udevadm monitor
monitor will print the received events for:
UDEV - the event which udev sends out after rule processing
KERNEL - the kernel uevent

KERNEL[2452.259891] remove   /devices/pci0000:00/0000:00:1a.0/usb1/1-1/1-1.3/
1-1.3:1.0/host6/target6:0:0/6:0:0:0/bsg/6:0:0:0 (bsg)
KERNEL[2452.262870] remove   /devices/pci0000:00/0000:00:1a.0/usb1/1-1/1-1.3/
1-1.3:1.0/host6/target6:0:0/6:0:0:0/scsi_generic/sg2 (scsi_generic)
[...]
UDEV  [2452.289223] remove   /devices/pci0000:00/0000:00:1a.0/usb1/1-1/1-1.3/
1-1.3:1.0 (usb)
UDEV  [2452.291043] unbind   /devices/pci0000:00/0000:00:1a.0/usb1/1-1/1-1.3 (usb)
UDEV  [2452.292691] remove   /devices/pci0000:00/0000:00:1a.0/usb1/1-1/1-1.3 (usb)
```

udev ha detectado correctamente la desconexión de la memoria USB.

d. Los enlaces simbólicos en /dev/disk

El servicio `udev` crea toda una serie de enlaces simbólicos en los subdirectorios del repertorio `/dev/disk`. Estos enlaces permiten identificar y acceder a los dispositivos de almacenamiento disponibles, discos o particiones, en función de diferentes criterios. Cada criterio corresponde a uno de los subdirectorios de `/dev/disk`:

`by-id`	Por los identificadores del dispositivo (tipo, número de serie, etc.).
`by-label`	Por etiqueta.
`by-partuuid`	Por el UUID de la partición.
`by-path`	Por su camino de acceso a su bus.
`by-uuid`	Por el UUID del dispositivo.

En el interior de esos subdirectorios se encuentran enlaces simbólicos hacia el archivo especial de bloques del disco o de la partición. De esta manera se les puede identificar de manera perenne, independientemente del orden de detección de los dispositivos.

Ejemplo

Los diferentes enlaces creados hacia el lector CD-ROM, archivo especial de bloque `/dev/sr0`, de una máquina CentOS 8:

```
ls -l /dev/disk/*/* | grep sr0
lrwxrwxrwx. 1 root root  9 12 marzo  10:10 /dev/disk/by-id/ata-hp_DVDRAM_
GT50N_M8JC2TK5225 -> ../../sr0
lrwxrwxrwx. 1 root root  9 12 marzo  10:10 /dev/disk/by-id/
wwn-0x5001480000000000 -> ../../sr0
lrwxrwxrwx. 1 root root  9 12 marzo  10:10 /dev/disk/by-label/
CentOS-8-1-1911-x86_64-dvd -> ../../sr0
lrwxrwxrwx. 1 root root  9 12 marzo  10:10 /dev/disk/by-path/pci-
0000:00:1f.2-ata-2 -> ../../sr0
lrwxrwxrwx. 1 root root  9 12 marzo  10:10 /dev/disk/by-uuid/
2020-01-03-21-30-07-00 -> ../../sr0
```

e. El sistema de archivos virtuales sysfs

Este sistema de archivos virtuales permite obtener datos de todos los dispositivos del sistema. Algunos pseudo archivos están accesibles en escritura y permiten modificar dinámicamente parámetros de los dispositivos (las modificaciones se hacen en memoria viva y se pierden cuando el sistema se para).

Para los dispositivos de almacenamiento gestionados en modo de bloque, los datos se presentan bajo la forma de archivos, en los subdirectorios de `dev/block/M:m`, donde `M` es el número mayor y `m` el número menor que identifican el dispositivo.

Ejemplo

Datos del disco duro asociado al archivo especial `/dev/sda`, mayor `8`, menor `0`:

```
ls -l /dev/sda
brw-rw---- 1 root disk 8, 0 marzo  12 13:14 /dev/sda
ls dev/block/8:0
alignment_offset  discard_alignment  hidden    power     sda1    stat
bdi               events             holders   queue     sda2    subsystem
capability        events_async       inflight  range     sda5    trace
dev               events_poll_msecs  integrity  removable  size    uevent
device            ext_range          mq        ro        slaves
```

Tamaño del disco en bloques:

```
cat dev/block/8:0/size
312581808
```

¿Es extraíble?

```
cat dev/block/8:0/removable
0
```

No.

f. El comando dmesg

El comando `dmesg` muestra el contenido del buffer de anillo (*ring buffer*) en el que se almacenan los mensajes enviados por el núcleo desde su arranque. Ahí se pueden encontrar los mensajes correspondientes a la detección e inicialización de los dispositivos.

El buffer se llama de anillo porque su contenido es sobrescrito cuando está lleno, los mensajes más antiguos pueden desaparecer, por lo tanto, rápidamente.

Ejemplo

Mensajes relacionados con el disco duro asociado a `/dev/sda` :

```
dmesg | grep sda | more
[    6.598141] sd 3:0:0:0: [sda] 625142448 512-byte logical blocks: (320 GB/298 GiB)
[    6.598143] sd 3:0:0:0: [sda] 4096-byte physical blocks
[    6.598154] sd 3:0:0:0: [sda] Write Protect is off
[    6.598156] sd 3:0:0:0: [sda] Mode Sense: 00 3a 00 00
[    6.598173] sd 3:0:0:0: [sda] Write cache: enabled, read cache: enabled, doesn't
support DPO or FUA
[    6.648134]  sda: sda1 sda2 sda3 sda4
[    6.648682] sd 3:0:0:0: [sda] Attached SCSI disk
[   42.463470] EXT4-fs (sda1): mounted filesystem with ordered data mode. Opts: (null)
```

g. Los comandos ls*

Distintos comandos bajo la forma `ls*` permiten mostrar la lista más o menos detallada de los dispositivos de este o aquel tipo. Entre ellos podemos destacar, con respecto a los dispositivos de almacenamiento:

lsdev

Instalado con el paquete `procinfo`, es el comando más genérico. Utiliza distintos pseudo archivos del sistema de archivos `proc` y muestra las características de hardware de los diferentes dispositivos: DMA: interrupciones y direcciones de entrada/salida.

lsscsi [-v]
lspci [-v]
lsusb [-v]

Estos comandos muestran los datos de los elementos conectados a los buses SCSI, PCI, o de los dispositivos USB.

El paquete `procinfo` ya no está presente en los depósitos estándar de CentOS ni de Red Hat para la versión 8.

Ejemplo

`lsdev` *en una distribución Debian:*

```
lsdev
Device            DMA   IRQ  I/O Ports
------------------------------------------------
0000:00:01.1                3000-303f   3040-307f   3080-30bf
0000:00:06.0                0170-0177   01f0-01f7   0376-0376   03f6-03f6   3
0c0-30cf
0000:00:09.0                30d0-30df   30e0-30e3   30e4-30e7   30e8-30ef   3
0f0-30f7
0000:00:0a.0                30f8-30ff
ACPI                      1000-1003    1004-1005    1008-100b    1010-
1015    1020-1027    1484-1484
acpi                   9
```

```
ahci                    30d0-30df    30e0-30e3    30e4-30e7    30e8-
30ef    30f0-30f7
ahci[0000:00:09.0]       23
cascade          4
dma                     0080-008f
dma1                     0000-001f
dma2                     00c0-00df
EC                      0062-0062    0066-0066
ehci_hcd:usb2          22
enp0s10              24
firewire_ohci          5
forcedeth                30f8-30ff
fpu                     00f0-00ff
i8042              1 12
keyboard                 0060-0060   0064-0064
mmc0                7
nForce2_smbus              3000-303f    3040-307f
nvkm               16
ohci_hcd:usb4          18
pata_amd          14 15    0170-0177    01f0-01f7    0376-0376    03f6-
03f6    30c0-30cf
PCI                     0000-0cf7 0cf8-0cff 0d00-ffff   4000-4fff
pic1                     0020-0021
pic2                     00a0-00a1
pnp                      0360-0361   04d0-04d1   1000-107f   1080-10ff   1
400-147f   1480-14ff   1800-187f   1880-18ff
PNP0800:00               0061-0061
PNP0C04:00                00f0-00f1
PNP0C09:00               0062-0062   0066-0066
rtc0               8   0070-0071
snd_hda_intel:card0       21
timer              0
timer0                   0040-0043
timer1                   0050-0053
vga+                     03c0-03df
```


2. Gestión de bajo nivel de los dispositivos de almacenamiento

Distintos comandos, según el tipo de dispositivo de almacenamiento, permiten visualizar, e incluso modificar los parámetros de estos periféricos.

Estos comandos trabajan en un nivel bajo, tienen que ser usados con precaución en modo modificación, ya que existe el riesgo de disminuir el rendimiento del dispositivo o incluso estropearlo.

a. El comando hdparm

Este comando permite interactuar con diferentes pilotos de dispositivos de almacenamiento de tipo SATA, IDE, algunos tipos de SCSI y USB…

Soporta numerosas opciones, puede mostrar los parámetros de configuración de los dispositivos y modificar algunos de ellos.

Sintaxis

```
hdparm -opciones ArchivoEspecial
```

Parámetros principales

`-opciones [ valorParÆmetro ]`	Parámetro para leer, si no se especifica ningún `valorParÆmetro`, o para posicionar en `valorParÆmetro`.
`ArchivoEspecial`	Archivo especial del dispositivo.

Descripción

El comando funciona en modo lectura o modificación de los parámetros. Si se utiliza una opción sin argumento, el comando retornará el valor del parámetro correspondiente. Si se utiliza con un argumentó, el comando posicionará este valor en el parámetro.

Algunas opciones son específicas para cierto tipo de discos.

Ejemplo

```
hdparm -v /dev/sda
/dev/sda:
multcount    = 16 (on)
IO_support   =  1 (32-bit)
readonly     =  0 (off)
readahead    = 8192 (on)
geometry     = 38913/255/63, sectors = 625142448, start = 0
```

b. El comando sdparm

Este comando permite interactuar con diferentes pilotos de dispositivos de almacenamiento de tipo SCSI y similares.

Admite muchas opciones, puede mostrar los parámetros de configuración de los dispositivos y modificar algunos de ellos.

Sintaxis

```
sdparm -opciones ArchivoEspecial
```

Parámetros principales

`-opciones [ valorParÆmetro ]`	Parámetro para leer, si no se especifica ningún `valorParÆmetro`, o para posicionar en `valorParÆmetro`.
`ArchivoEspecial`	Archivo especial del dispositivo.

Descripción

Como `hdparm`, el comando funciona en modo lectura o modificación de algunos parámetros.

También permite enviar comandos SCSI al controlador de dispositivo.

Ejemplo

Parámetros de un disco duro:

```
sdparm /dev/sda
  /dev/sda: ATA     ST9160821AS     D
Read write error recovery mode page:
 AWRE         1  [cha: n, def:  1]
 ARRE         0  [cha: n, def:  0]
 PER          0  [cha: n, def:  0]
Caching (SBC) mode page:
 IC           0  [cha: n, def:  0]
 WCE          1  [cha: y, def:  1]
 RCD          0  [cha: n, def:  0]
Control mode page:
 TST          0  [cha: n, def:  0]
 SWP          0  [cha: n, def:  0]
```

Utilización con un lector de CD-ROM:

```
sdparm /dev/sr0
  /dev/sr0: PIONEER   DVDRW   DVR-K17   1.05  [cd/dvd]
```

```
Read write error recovery mode page:
 AWRE         1  [cha: y, def:  1, sav:  1]
 ARRE         0  [cha: n, def:  0, sav:  0]
 PER          0  [cha: y, def:  0, sav:  0]
Write parameters (MMC) mode page:
 BUFE         1  [cha: y, def:  1, sav:  1]
 WR_T         0  [cha: y, def:  0, sav:  0]
 MULTI_S      3  [cha: y, def:  3, sav:  3]
Caching (SBC) mode page:
 IC           0  [cha: n, def:  0, sav:  0]
Power condition mode page:
 PM_BG        0  [cha: n, def:  0, sav:  0]
Power consumption mode page:
 ACT_LEV      0  [cha: y, def:  0, sav:  0]
SAT ATA Power condition mode page:
 APMP         0  [cha: y, def:  0, sav:  0]
Timeout and protect (MMC) mode page:
 WORMM        0  [cha: n, def:  0, sav:  0]
CD/DVD (MM) capabilities and mechanical status (MMC) mode page:
 D_RAM_R      1  [cha: n, def:  1, sav:  1]
```

Se puede utilizar el comando para expulsar el CD del lector:

```
sdparm --command=unlock /dev/sr0
  /dev/sr0: PIONEER   DVDRW   DVR-K17   1.05  [cd/dvd]
sdparm --command=eject /dev/sr0
  /dev/sr0: PIONEER   DVDRW   DVR-K17   1.05  [cd/dvd]
```

c. Gestión de los discos SSD

Los discos SSD (*Solid State Drive*) se utilizan cada vez más, por su rendimiento excepcional en tiempo de acceso, particularmente en lectura.

Tienen una duración de vida limitada en función del número de escrituras.

En Linux, los discos SSD, conectados a un bus PCI express, son gestionados por el piloto NVMe (*Non-Volatile Memory express*, nombre de la especificación de la gestión de los dispositivos SSD), integrado en el núcleo.

Los periféricos SSD gestionados por ese piloto se asocian a archivos especiales cuyo nombre tiene la forma de `/dev/nvme*`.

La especificación NVMe concierne a los discos SSD conectados a un bus PCI express, método más específico pero que muestra mejor rendimiento comparado con el uso de un bus más clásico, de tipo AHCI (Advanced Host Controller Interface), peor adaptado a las particularidades de los discos SSD.

El paquete `nvme-cli` instala el comando `nvme`, que permite interactuar con el piloto que gestiona los discos SSD PCI express:

Sintaxis

```
nvme Comando ArchivoEspecial [ Args ]
```

Parámetros principales

`Comando`	Subcomando que se ejecuta.
`ArchivoEspecial`	Archivo especial del dispositivo.
`Args`	Argumentos del subcomando.

Descripción

El comando funciona en modo lectura o modificación de los parámetros del dispositivo SSD especificado. También puede enviar comandos al controlador del dispositivo (lectura de bloques, escritura de bloques).

Ejemplo

Lista de los discos SSD:

```
nvme list
Node          SN                Model                Namespace
Usage                Format         FW Rev
----------------- -------------------- ---------------------------------
------------------------- ---------------- --------
/dev/nvme0n1     YZ38451820          SAMSUNG MZVPV512HDGL   1
5.3  GB / 512.11  GB    512   B +  0 B   CD24ZRT
```

Características del controlador:

```
nvme id-ctrl -H /dev/nvme0
NVME Identify Controller:
vid    : 0x144d
ssvid  : 0x144d
sn     : YZ38451820
mn     : SAMSUNG MZVPV512HDGL
fr     : CD24ZRT
rab    : 2
ieee   : 002538
cmic   : 0
 [2:2]: 0 PCI
 [1:1]: 0 Single Controller
 [0:0]: 0 Single Port
mdts   : 5
cntlid : 1
ver    : 0
rtd3r  : 0
[...]
```

Los pilotos SSD implementan técnicas elaboradas para optimizar el uso de las células de escritura y aumentar la vida del dispositivo. Para ello, las versiones recientes implementan generalmente de manera automática una operación de *trimming*.

Si fuera necesario, esta operación podría ser ejecutada manualmente en un sistema de archivos de un disco SSD montado, usando el comando siguiente:

```
fstrim PuntoMontaje
```

d. Gestión de los bloques defectuosos

El comando `badblocks` hace referencia a los bloques físicamente defectuosos en un disco o en una partición.

Sintaxis

```
badblocks [ Opciones ] ArchivoEspecial
```

Parámetros principales

`Opciones`	Opciones.
`ArchivoEspecial`	Archivo especial del dispositivo.

Descripción

Por defecto, el comando comprueba los sectores del periférico especificado y muestra la lista de los bloques defectuosos. Esta lista puede almacenarse en un archivo, que podrá ser utilizado después con otros comandos de comprobación de los sistemas de archivos (de tipo `e2fsck`).

Ejemplo

Detección de los bloques en error del disco `/dev/sda` (la ejecución del comando puede tomar bastante tiempo):

```
badblocks /dev/sda
```

e. Identificador SCSI

El comando `scsi_id` permite visualizar el identificador SCSI único de un disco SCSI.

Sintaxis

```
/lib/udev/scsi_id -g ArchivoEspecial
```

Parámetros principales

`-g`	Fuerza el modo whitelisted.
`ArchivoEspecial`	Archivo especial del dispositivo.

Descripción

El comando con la opción `-g` permite obtener el identificador único del disco especificado. Este identificador sirve en algunas aplicaciones para garantizar la identidad del disco que se debe tratar.

Ejemplo

En una distribución CentOS 8:

```
/lib/udev/scsi_id -g /dev/sda
35000c500493da068
```

3. Las redes de almacenamiento SAN

Una red de almacenamiento SAN (*Storage Area Network*) permite acceder a los recursos de almacenamiento a través de una red. Los recursos se ven como espacios de almacenamiento locales, discos o volúmenes físicos LVM, en los que se pueden crear particiones, volúmenes lógicos LVM y sistemas de archivos.

La comunicación con los recursos de almacenamiento remotos se hace gracias a los comandos de bajo nivel, de tipo SCSI o ATA, encapsulados en un protocolo que puede ser de nivel de enlace de datos (Ethernet) o de red (IP). Los protocolos que permiten esta comunicación son, respectivamente, iSCSI (*Internet SCSI*) y AoE (*ATA over Ethernet*).

También se pueden utilizar redes de fibra óptica (*Fibre Channel*) o encapsular el protocolo en Ethernet para poder acceder a las bahías de almacenamiento de fibra óptica a través de una red Ethernet (protocolo FCoE, *Fibre Channel over Ethernet*).

En el marco de la certificación, se estudiará la implementación de iSCSI en Linux.

4. Gestión de los discos iSCSI

SCSI (*Small Computer System Interface*) es una familia de protocolos de comunicación destinados a los dispositivos de entrada/salida. Se trata de un protocolo cliente-servidor, que permite al cliente (el sistema) enviar solicitudes al servidor (el controlador SCSI), que es el encargado de gestionar las unidades de almacenamiento (discos, lectores de bandas, etc.).

iSCSI (*Internet SCSI*) es una extensión de red del protocolo SCSI que permite enviar comandos SCSI a través de una red TCP/IP. De esta manera se pueden administrar a distancia accesos a discos de bajo nivel, normalmente en una topología de tipo Ethernet. Este protocolo estándar está definido en la RFC 3720.

iSCSI se utiliza en el marco de redes de almacenamiento (*Storage Area Network*) en TCP/IP, solución menos onerosa que las redes SAN en fibra óptica.

Open-iSCSI es una implementación de software libre de este protocolo, que funciona en Linux. Este software permite a un sistema Linux utilizar este protocolo como cliente (acceder a los servidores de almacenamiento de discos iSCSI).

Linux también puede hacer el rol de servidor de almacenamiento iSCSI; en este caso la máquina administrada por Linux pondrá algunos de sus discos a disposición exclusiva de los clientes iSCSI remotos.

a. Terminología

Un **cliente** iSCSI se llama iniciador iSCSI (*iSCSI Initiator*). Está definido por un identificador único, con formato WWN (*World Wide Name*).

Un **objetivo** iSCSI (*iSCSI target*) corresponde a una interfaz de un servidor de almacenamiento. Está identificada por un identificador único, en formato WWN. Un objetivo puede estar asociado a distintas direcciones IP y, a la inversa, una dirección IP puede corresponder a distintos objetivos.

Un objetivo iSCSI administra **unidades lógicas** de almacenamiento, identificadas por un número de unidad lógica LUN (*Logical Unit Number*). Cada LUN es visto por el sistema cliente como un dispositivo físico de almacenamiento (disco duro, cartucho).

Un **portal de red** (*Network Portal*) es un elemento iSCSI asociable a una conexión TCP/IP. Un servidor de portal de red escucha en una dirección IP y en un puerto TCP. El puerto bien conocido reservado a iSCSI es el número 3260.

Los nombres iSCSI utilizando un formato de nombre de tipo WWN (*World Wide Name*), de manera que se asegure su unicidad. Este formato es el siguiente:

```
iqn.AAAA-MM.Dominio:cadena_ident
```

donde:

`iqn.`	Prefijo que especifica el espacio de nombres (*iSCSI qualified name*).
`AAAA-MM.`	Fecha (Año-Mes) que debe corresponder al primer mes de atribución del nombre de dominio utilizado por la autoridad que gestiona este WWN.
`Dominio`	Nombre de dominio, en orden inverso, de la autoridad que gestiona este WWN.
`:cadena_ident`	Opcionalmente, identificador del elemento designado por éste nombre. La unicidad de este identificador es responsabilidad de la autoridad que gestiona WWN.

Ejemplo

```
iqn.2020-01.no.net:cloud1.target1
```

Objetivo `target1` *del servidor iSCSI* `cloud1` *del dominio* `no.net`*, declarado desde junio de 2020.*

b. Paquetes de software iSCSI

Los componentes de software iSCSI no se suministran generalmente por defecto con las distribuciones, habrá que instalarlos por separado. Los paquetes son distintos para el cliente o para el servidor.

Paquetes para los clientes iSCSI

Para las distribuciones de tipo Red Hat, el paquete se llama `iscsi-initiator-utils` .

Para las distribuciones de tipo Debian, el paquete se llama `open-iscsi` .

Ejemplo

Instalación del paquete en una distribución CentOS:

```
yum install iscsi-initiator-utils
```

Instalación del paquete en una distribución Debian:

```
apt-get install open-iscsi
```

Paquetes de servidores iSCSI

Para las distribuciones de tipo Red Hat versión 7, el paquete se llama `scsi-target-utils` y es necesario que el repositorio EPEL (*Extra Packages for Enterprise Linux*) esté activado.

Para las distribuciones de tipo Red Hat versión 8, el paquete se llama `targetcli` e implementa un nuevo servidor iSCSI.

Para las distribuciones de tipo Debian, el paquete se llama `tgt`.

Ejemplo

Instalación del paquete en una distribución CentOS 8:

```
yum install targetcli
```

Instalación del paquete en una distribución Debian:

```
apt-get installtgt
```

La mayoría de las veces, Linux se utiliza como un cliente iSCSI, ya que utilizará discos remotos proporcionados por un equipo especializado que hace el rol de servidor iSCSI.

c. Linux como cliente iSCSI

La implementación de un cliente iSCSI en Linux, para que el sistema pueda usar discos remotos a través de una red TCP/IP, se hace con tres conjuntos de elementos:

- Un módulo del núcleo que administra las interfaces de red TCP/IP.
- Un módulo de tipo servidor que asegura la comunicación con los servidores de almacenamiento destinatarios.
- Comandos de gestión que permiten administrar las unidades lógicas de almacenamiento puestas a disposición por los servidores destinatarios iSCSI.

El término servidor es confuso en el caso de `iscsid`. De hecho, aunque sea un "servidor" desde el punto de vista del sistema, se comporta como cliente (iniciador) de un servidor de almacenamiento iSCSI, enviando solicitudes de acceso a un dispositivo de almacenamiento.

Arranque del servidor iscsid

El archivo de configuración por defecto del servidor `iscsid` está en `/etc/iscsi/iscsid.conf`. Para un uso simple, no es necesario modificar su contenido por defecto.

Si los recursos remotos tienen el acceso protegido, habrá que configurar en ese archivo la cuenta del usuario y la contraseña que tendrá que enviarse al servidor remoto.

Por defecto, el programa servidor `iscsid` arranca automáticamente en cuanto el

comando le hace una llamada.

Se puede arrancar el servidor en modo manual usando el comando:

```
systemctl iscsid start
```

Ejemplo

```
systemctl start iscsid
ps -ef | grep iscsi
root     2862    2  0 18:43?     00:00:00 [iscsi_eh]
root     3000    1  0 18:44?     00:00:00 /sbin/iscsid
root     3001    1  0 18:44?     00:00:00 /sbin/iscsid
```

Acceso a recursos de disco remotos

Para acceder la primera vez a un dispositivo gestionado por un servidor iSCSI remoto, primero hay que solicitar que `iscsid` haga un descubrimiento (*discover*) de este servidor.

Después, habrá que dar los elementos para conectarse a uno de los recursos remotos propuestos por el servidor iSCSI:

- WWN del recurso.
- Opcionalmente, el nombre de usuario y la contraseña, si el acceso del recurso está protegido.

Una vez que esta conexión haya sido efectuada, el dispositivo iSCSI podrá ser utilizado como un dispositivo de disco local: será incluido en el grupo de volúmenes LVM o creado directamente en un sistema de archivos.

Para automatizar estas conexiones y que perduren en el tiempo, se declaran (automáticamente) en la base de datos del servidor iSCSI, y éste será configurado para ser arrancado durante el inicio del sistema.

Descubrimiento de un servidor iSCSI

El descubrimiento de un servidor iSCSI se hace a través de su portal de red, identificado por una dirección IP (y opcionalmente un puerto, si el puerto bien conocido no es usado).

La opción `-m discovery` del comando `iscsiadm` permite descubrir los objetivos iSCSI expuestos por un servidor de almacenamiento iSCSI remoto.

Ejemplo

```
iscsiadm -m discovery -t sendtargets -p 192.168.0.39
192.168.0.39:3260,1 iqn.2008-09.com.example:server.target1
```

En este ejemplo, se interroga al servidor `192.168.0.39` enviándole los identificadores WWN de clientes locales (Initiator WWN). El servidor remoto responde indicando el WWN de un objetivo accesible `iqn.2008-09.com.example:server.target1` .

Conexión a un objetivo iSCSI

Después de haber hecho el descubrimiento, hay que conectarse al objetivo deseado. Si esta conexión funciona, se guardará en la base de datos del servidor `icsid` y será posible volver a conectarse automáticamente.

Las opciones `-m nodo` , `-T objetivo` , `-p portal` y `--login` del comando `iscsiadm` permiten solicitar la conexión a un objetivo.

Ejemplo

```
iscsiadm -m node -T iqn.2008-09.com.example:server.target1 -p 192.168.0.39 --login
Logging in to [iface: default, target: iqn.2008-09.com.example:server.target1, portal:
192.168.0.39,3260]
Login to [iface: default, target: iqn.2008-09.com.example:server.target1, portal:
192.168.0.39,3260] successful.
```

Si la conexión ha sido aceptada, las unidades de almacenamiento remotas gestionadas por el objetivo serán asociadas a archivos especiales locales en modo de bloque, como si se tratara de dispositivos locales.

Se puede comprobar con el comando `fdisk` o consultando los mensajes de `iscsid` en el archivo del registro general (`/var/log/messages`).

Ejemplo

```
tail /var/log/messages
Mar 15 19:26:41 beta kernel: scsi 8:0:0:0: RAID              IET
Controller       0001 PQ: 0 ANSI: 5
Mar 15 19:26:41 beta kernel: scsi 8:0:0:0: Attached scsi generic sg4 type 12
Mar 15 19:26:41 beta kernel: scsi 8:0:0:1: Direct-Access     IET
VIRTUAL-DISK     0001 PQ: 0 ANSI: 5
Mar 15 19:26:41 beta kernel: sd 8:0:0:1: Attached scsi generic sg5 type 0
Mar 15 19:26:41 beta kernel: sd 8:0:0:1: Power-on or device reset occurred
Mar 15 19:26:41 beta kernel: sd 8:0:0:1: [sdd] 30322688 512-byte logical
blocks: (15.5 GB/14.5 GiB)
Mar 15 19:26:41 beta kernel: sd 8:0:0:1: [sdd] Write Protect is off
Mar 15 19:26:41 beta kernel: sd 8:0:0:1: [sdd] Write cache: enabled, read
cache: enabled, supports DPO and FUA
Mar 15 19:26:41 beta kernel: sdd: sdd1
Mar 15 19:26:41 beta kernel: sd 8:0:0:1: [sdd] Attached SCSI disk
```

El archivo especial `/dev/sdd` ahora puede utilizarse como disco local.

```
fdisk -l /dev/sdd
Disco /dev/sdd: 14,5 GiB, 15525216256 bytes, 30322688 sectores
Unidades: sectores 1 × 512 = 512 bytes
Tamaño de sector (lógico/físico): 512 bytes / 512 bytes
Tamaño de E/S (mínimo/óptimo): 512 bytes / 512 bytes
Tipo de etiqueta de disco: dos
Identificador de disco: 0xcb2f47a6
Disposit.  Inicio Comienzo    Final Sectores Tamaño Id Tipo
/dev/sdd1            8064 30322687 30314624  14,5G  c W95 FAT32 (LBA)
```

Uso del dispositivo iSCSI

El dispositivo ya se puede declarar como disco físico de un grupo de volúmenes o se puede usar directamente para almacenar uno o distintos sistemas de archivos.

Se puede usar como un disco entero o particionarlo.

Ejemplo

Uso del disco remoto `/dev/sdd`:

```
mkfs -t ext4 /dev/sdd
mke2fs 1.44.5 (15-Dec-2018)
Found a dos partition table in /dev/sdd
Proceed anyway? (y,N) y
Creating filesystem with 3790336 4k blocks and 948416 inodes
Filesystem UUID: 09ea13b3-3181-459e-b931-0124b5b27878
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208

Allocating group tables: done
Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done

mkdir /var/bigdatos
mount /dev/sdd /var/bigdatos
df /var/bigdatos
S.ficheros     bloques de 1K  Usados Disponibles Uso% Montado en
/dev/sdd          14857656   40984   14042224   1% /var/bigdatos
```

En este ejemplo, se ha usado el disco entero como un sistema de archivos ext4, y después se ha montado en el directorio `/var/bigdatos`. Tiene una capacidad aproximada de 16 GB.

Perdurabilidad de la conexión

Generalmente es necesario hacer que el sistema de archivos sea accesible en cada inicio del sistema, siendo activado por `systemd`:

```
systemctl enable iscsid.service
```

Después, se puede configurar en montaje automático en el archivo `/etc/fstab`, añadiendo esta línea:

```
/dev/sdd /var/bigdatos ext4 _netdev 0 0
```

La opcion `_netdev` especifica el modo de acceso de red del sistema de archivos.

Ejemplo

Declaramos el sistema de archivos remoto en `/etc/fstab` :

```
/dev/sdd /var/bigdatos ext4 _netdev 0 0
```

Se desmonta:

```
umount /dev/sdd
```

Se monta usando el archivo `/etc/fstab` :

```
mount /var/bigdatos
```

Comprobamos su montaje:

```
mount | grep /var/bigdatos
/dev/sde on /var/bigdatos type ext4 (rw,_netdev)
```

Después de haber reiniciado el sistema:

```
mount | grep /var/bigdatos
/dev/sdd on /var/bigdatos type ext4 (rw,relatime,seclabel,_netdev)
```

Comprobamos que el servicio `iscsid` ha sido iniciado automáticamente:

```
systemctl status iscsid
```

```
 iscsid.service - Open-iSCSI
 Loaded: loaded (/usr/lib/systemd/system/iscsid.service; enabled; vendor
preset: disabled)
 Active: active (running) since Sun 2020-03-15 11:52:21 GMT; 8h ago
[...]
marzo 15 19:26:41 beta iscsid[1298]: iscsid: Connection1:0 to [target:
iqn.2008-09.com.example:server.target>
[...]
ps -ef | grep scsid
root     1298    1  0 11:52?       00:00:00 /usr/sbin/iscsid -f
root     8544  7866  0 20:08 pts/1    00:00:00 grep --color=auto scsid
```

El sistema de archivos remoto es operativo:

```
cp /etc/hosts /var/bigdatos/archivo1
ls -l /var/bigdatos/archivo1
-rw-r--r--. 1 root root 216 15 marzo  20:08 /var/bigdatos/archivo1
```

Desconexión de un recurso iSCSI

En el caso en que un dispositivo remoto iSCSI tenga que dejar de ser utilizado por el sistema, es posible solicitar a `iscsid` que lo desconecte, usando la opción `--logout` del comando `iscsiadm`, después de asegurarse de que no estaba montado.

Ejemplo

```
iscsiadm -m node -T iqn.2008-09.com.example:server.target1 -p
192.168.0.39 --logout
Logging out of session [sid: 1, target: iqn.2008-09.com.example:server.target1,
portal: 192.168.0.39,3260]
Logout of [sid: 1, target: iqn.2008-09.com.example:server.target1, portal:
192.168.0.39,3260] successful.
```

d. Linux como servidor iSCSI

Esta funcionalidad se utiliza de manera menos frecuente, pero es posible poner a disposición de los clientes iSCSI remotos dispositivos de almacenamiento administrados por el sistema Linux local. Estos periféricos pueden ser discos o particiones físicas, o incluso volúmenes lógicos (LVM). También podrían ser lectores de bandas.

Los accesos remotos son accesos de bajo nivel, por lo tanto es necesario que estos discos no se utilicen localmente y que, salvo casos excepcionales, un disco solamente sea utilizado por un cliente.

Principio

Una vez que el programa servidor iSCSI haya sido iniciado como servicio, se pondrá a la escucha de solicitudes de los clientes iSCSI (los iniciadores), por defecto en el número de puerto bien conocido 3260. Este administrará uno o distintos objetivos y, por cada objetivo, una o distintas unidades lógicas (LUN). Respondiendo a un comando iSCSI de un cliente, transmite la solicitud al controlador del dispositivo local y devuelve la respuesta al cliente.

La configuración es diferente para las distribuciones de tipo Debian y Red Hat 8, pero el principio es el mismo.

Ejemplo con el servidor `tgt`

En las distribuciones de tipo Debian, el servicio servidor iSCSI se llama `tgt.service` y lo administra `systemd`.

Se empieza por parar el servicio antes de modificar su configuración:

```
systemctl stop tgt
```

Se añade un archivo de configuración `mytarget.conf`, en el directorio `/etc/tgt/conf.d` para declarar un objetivo local:

```
vi /etc/tgt/conf.d/mytarget.conf:
<target iqn.2008-09.com.example:server.target1>
  backing-store /dev/sdb
</target>
```

Este archivo asocia el disco local `/dev/sdb` al objetivo, cuyo WWN es:

```
iqn.2008-09.com.example:server.target1
```

El dispositivo no debe estar montado en el momento de la activación del objetivo iSCSI.

Se reinicia el servicio, y después se configura en arranque automático.

```
systemctl start tgt
```

Se hace perdurable (arranque automático en modo multiusuario):

```
systemctl enable tgt
```

Para seguir el estado del servicio del servidor iSCSI e interactuar con él, se usa el comando `tgtadm`. El comando siguiente muestra la lista de los objetivos administrados por el servidor:

```
tgtadm --mode target --op show
Target 1: iqn.2008-09.com.example:server.target1
  System information:
    Piloto: iscsi
    State: ready
  I_T nexus information:
  LUN information:
    LUN: 0
      Type: controller
      SCSI ID: IET     00010000
      SCSI SN: beaf10
```

```
        Size: 0 MB, Block size: 1
        Online: Yes
        Removable media: No
        Prevent removal: No
        Readonly: No
        SWP: No
        Thin-provisioning: No
        Backing store type: null
        Backing store path: None
        Backing store flags:
    LUN: 1
        Type: disk
        SCSI ID: IET     00010001
        SCSI SN: beaf11
        Size: 15525 MB, Block size: 512
        Online: Yes
        Removable media: No
        Prevent removal: No
        Readonly: No
        SWP: No
        Thin-provisioning: No
        Backing store type: rdwr
        Backing store path: /dev/sdb
        Backing store flags:
Account information:
ACL information:
    ALL
```

La LUN 0 corresponde al controlador, la LUN 1 al disco `/dev/sdb`.

Desde una máquina donde el cliente iSCSI ha sido instalado (incluyendo la máquina local), se pide hacer un descubrimiento hacia el portal de redes del servidor iSCSI, para asegurarse de que esté bien accesible y de que proponga los objetivos configurados.

```
iscsiadm -m discovery -t sendtargets -p 192.168.0.39
192.168.0.39:3260,1 iqn.2008-09.com.example:server.target1
```

El servidor que tiene la dirección IP 192.168.0.39 propone efectivamente el objetivo

configurado.

Preste atención al firewall del sistema, hay que configurarlo para que acepte las solicitudes de conexiones entrantes hacia el puerto asociado al servidor iSCSI (3260 por defecto). Para los tests, se puede desactivar el firewall temporalmente con el comando `systemctl stop firewalld` .

Ejemplo con el servidor `target`

En las distribuciones de tipo CentOS 8, el servicio servidor iSCSI se llama `target.service` y es gestionado por `systemd`. Usa achivos de configuración en el formato JSON, que se pueden crear en modo interactivo con el comando `targetcli` .

Se empieza por parar el servicio, antes de modificar la configuración:

```
systemctl stop target
```

Se usa el comando interactivo `targetcli` para configurar el objetivo iSCSI:

```
targetcli
Warning: Could not load preferences file /root/.targetcli/prefs.bin.
targetcli shell version 2.1.fb49
Copyright 2011-2013 by Datera, Inc and others.
For help on commands, type 'help'.
/>
```

Se entra en el directorio de los dispositivos `backstores` en modo de bloque:

```
cd /backstores/block
/backstores/block>
```

Se crea un objeto `LUN0`, asociado al disco `/dev/sdb`:

```
create name=LUN0 dev=/dev/sdb
Created block storage object LUN0 using /dev/sdb.
```

Entramos en el directorio `/iscsi` para declarar un objetivo local:

```
cd /iscsi
/iscsi> create iqn.2020-03.com.example:server
Created target iqn.2020-03.com.example:server.
Created TPG 1.
Global pref auto_add_default_portal=true
Created default portal listening on all IPs (0.0.0.0), port 3260.
```

Se entra en el directorio del objetivo creado:

```
cd iqn.2020-03.com.example:server
/iscsi/iqn.20...xample:server>
```

Se lista el directorio:

```
ls
o- iqn.2020-03.com.example:server ..............[TPGs: 1]
 o- tpg1 ...................................[no-gen-acls, no-auth]
   o- acls ..................................... [ACLs: 0]
   o- luns ..................................... [LUNs: 0]
   o- portals ................................... [Portals: 1]
     o- 0.0.0.0:3260 ..................... [OK]
```

Entramos en el directorio de declaración de los LUN del objetivo:

```
cd tpg1/luns
/iscsi/iqn.20...ver/tpg1/luns>
```

Se crea el LUN asociándolo al backstore `LUN0` creado:

```
create /backstores/block/LUN0
Created LUN 0.
```

Se vuelve a la raíz y se guarda la configuración:

```
cd /
saveconfig
Configuration saved to /etc/target/saveconfig.json
```

Salimos:

```
exit
Global pref auto_save_on_exit=true
Last 10 configs saved in /etc/target/backup/.
Configuration saved to /etc/target/saveconfig.json
```

El archivo de configuración, que se puede modificar, se encuentra en: `/etc/target/saveconfig.json`.

Se inicia el servicio:

```
systemctl start target
```

Se pregunta al servidor local:

```
iscsiadm -m discovery -t sendtargets -p 127.0.0.1
127.0.0.1:3260,1 iqn.2020-03.com.example:server
```

El servidor local propone correctamente el objetivo configurado.

Desde una máquina remota:

```
iscsiadm -m discovery -t sendtargets -p 192.168.0.4
192.168.0.4:3260,1 iqn.2020-03.com.example:server
```