→ Convenciones:

```
# En todos los nodos como 'sudo su'.
[root@srv1 ~]# Solo en servidor 'srv1'→ como 'sudo su'.
[root@srv2 ~]# Solo en servidor 'srv2'→ como 'sudo su'.
```

## 362.2 Cluster Storage Access (weight: 3)

| | |
|---|---|
| **Weight** | 3 |
| **Description** | Candidates should be able to connect a Linux node to remote block storage. This includes understanding common SAN technology and architectures, including management of iSCSI, as well as configuring multipathing for high availability and using LVM on a clustered storage. |

**Key Knowledge Areas:**

- Understand the concepts of Storage Area Networks
- Understand the concepts of Fibre Channel, including Fibre Channel Toplogies
- Understand and manage iSCSI targets and initiators
- Understand and configure Device Mapper Multipath I/O (DM-MPIO)
- Understand the concept of a Distributed Lock Manager (DLM)
- Understand and manage clustered LVM
- Manage DLM and LVM with Pacemaker

**Partial list of the used files, terms and utilities:**

- tgtadm
- targets.conf
- iscsiadm
- iscsid.conf
- /etc/multipath.conf
- multipath
- kpartx
- pvmove
- vgchange
- lvchange

## →→ Conceptos preliminares Teóricos.

### Arquitecturas de Almacenamiento:

Empezaremos con los términos DAS ("Direct Attached Storage"), NAS ("Network Attached Storage"), y SAN ("Storage Area Network"), sus ventajas e inconvenientes, diferencias y similitudes, así como su relación con otros conceptos y tecnologías (SCSI, "Fibre Channel", RAID, NFS ó "Network File System", CIFS ó "Common Internet File System", iSCSI, MPIO,

"SecurePath", "LUN Masking", "Zoning", etc).

**DAS ("Direct Attached Storage"):**

Se trata de dispositivos de almacenamiento directamente conectados a la máquina, como es el caso de discos duros internos, cabinas de disco o unidades de cinta para "backup".

Suelen basarse en tecnologías SCSI ("Small Computers System Interface") y FC ("Fiber Channel"). Esta arquitectura de almacenamiento, se relacionaba principalmente con la época de los "Mainframe" de IBM. Sin embargo, hoy en día, los PC's de sobremesa utilizan arquitectura de almacenamiento DAS, mientras que en los servidores de las empresas, empieza a caer en desuso, utilizándose únicamente para el almacenamiento del sistema operativo.

La arquitectura de almacenamiento DAS ("Direct Attached Storage"), presenta muchos inconvenientes, como es la dispersión del almacenamiento, que implica una dificultad en la gestión de los "Backups", así como una baja tolerancia a fallos (sólo posible a través de soluciones RAID), y un alto TCO ("Total Cost of Ownership")debido a las dificultades de mantenimiento.

**NAS (Network Attached Storage):**

Con la introducción de las redes locales (LAN), se empezaron a utilizar servidores de almacenamiento conectados a la red, a los cuales se podía acceder directamente a través de la propia infraestructura mediante protocolos específicos como NFS ("Network File System") en entornos UNIX y CIFS ("Common Internet File System") en entornos Microsoft (antes conocido como SMB, protocolo original de IBM que fue mejorado por Microsoft en CIFS) o incluso mediante FTP, HTTP, etc.

Actualmente las soluciones NAS se basan en TCP/IP, con protocolos NFS o CIFS por encima. En consecuencia, un dispositivo NAS será una máquina dedicada con una o varias direcciones IP y además estará dotado de una conexión de alta velocidad a la red LAN. De esta forma, los equipos clientes en una arquitectura de almacenamiento NAS, delegan la gestión del sistema de ficheros al propio dispositivo NAS, que se limita a montar las unidades de red exportadas o compartidas, de tal modo que los usuarios y aplicaciones utilizan estos sistemas de ficheros como si fueran locales, aunque para el sistema operativo se trate claramente de sistemas de ficheros remotos.

El problema de esta arquitectura de almacenamiento, es que la red LAN puede actuar de cuello de botella "Bottle neck". Actualmente, siguen utilizándose masivamente las arquitecturas NAS (Carpetas Compartidas o "Shared Folder", las cuales se usan en las empresas para el almacenamiento de ficheros).

Los principales beneficios de las Arquitecturas de Almacenamiento NAS son que proporcionan un mejor TCO ("Total Cost of Ownship"), resultando fácilmente escalable y capaces de ofrecer una alta disponibilidad.

**SAN (Storage Area Network):**

Esta arquitectura implica disponer de una infraestructura de red de alta velocidad dedicada sólo para almacenamiento y “Backup”, optimizada para mover grandes cantidades de datos y consistente en múltiples recursos de almacenamiento geográficamente distribuidos o no, además de otros elementos (cables, “switches” de fibra FC, “routers”, adaptadores HBA, etc).

Las redes de almacenamiento SAN han facilitado enormemente la creación de Centros de Procesos de Datos (CDP) distribuidos, Clusters Geográficos, creación de centros de respaldo (BDC), etc.

La utilización de una arquitectura de almacenamiento SAN implica la existencia y mantenimiento de al menos dos redes: la red LAN y la red SAN. En la práctica, las redes de almacenamiento SAN suelen basarse en la tecnología FC (“Fibre Channel”), aunque también pueden basarse en “Gigabit Ethernet” (iSCSI).

Cuando se habla de redes conmutadas en “Fibre Channel”, suele utilizarse el término “Switch Fabric”. Se emplean múltiples “switches” y múltiples puertos para ofrecer alta disponibilidad basada en la existencia de un gran número de caminos “Paths”, apoyándose para ello en soluciones y protocolos como MPIO (“Multipath Input Output”) y “SecurePath” (solución propietaria de HP).

Además de la alta disponibilidad relativa a la redundancia de caminos, también se utilizan soluciones de alta disponibilidad del almacenamiento (“Mirroring” o RAID1, RAID5, RAID10, etc).

Los beneficios o ventajas de las redes de almacenamiento SAN, son evidentes: mayor velocidad de acceso a datos, menor tiempo de recuperación ante desastres (los tiempos de “Backup” y “Restore” se minimizan), escalabilidad (siempre es posible añadir más discos, o incluso, más cabinas de discos y “Switches”), y sobre todo, una gestión centralizada, compartida y concurrente del almacenamiento (indiferentemente de la plataforma y sistema operativo de los “Hosts”).

Como inconveniente, las redes de almacenamiento SAN tienen un elevado coste. Una de la principales alternativas es la utilización de soluciones de almacenamiento SAN basadas en iSCSI, que funcionan con tarjetas Ethernet, no haciendo falta HBA (Hot Bus Adapter), que no es más que una tarjeta hardware que se instala en el equipo (normalmente en el puerto SCSI) y que permite conectar mediante “fibre cannel” (canal de fibra) el equipo con un sistema de almacenamiento en fibra (SAN).

Las diferencias entre NAS y SAN son principalmente que un “Host” o Servidor accede a un disco NAS a través de la red, siendo el sistema operativo consciente de que se está accediendo a un recurso remoto. Sin embargo, un “Host” o Servidor accede a un disco SAN como si fuera un disco local, de forma transparente para el sistema operativo, siendo las tarjetas HBA y sus drivers quienes se encargan de que dicho acceso a la SAN sea de esta forma.

**¿Qué es Fiber Channel (FC)?**

Es una tecnología de red Gigabit utilizada principalmente para redes de almacenamiento SAN y

para la conexión de cabinas de discos DAS, capaz de funcionar sobre cables de fibra óptica y sobre cables de cobre de par trenzado (“twisted pair copper wire”), aunque la gran mayoría de las veces se suele usar cableado de fibra óptica.

“Fibre Channel Protocol” (FCP) es un protocolo de transporte para la transmisión de comandos SCSI sobre redes “Fibre Channel”. Es el más usado en redes de almacenamiento basadas en conexiones de fibra óptica.

La tecnología “Fibre Channel” (FC) ofrece tres posibles topologías:

**FC-P2P, Point-to-Point:**

Se utiliza en soluciones de almacenamiento DAS, en las cuales, se conecta una cabina de almacenamiento o un robot de cintas directamente a las tarjetas HBA del servidor.

**FC-AL:**

Permite conectar hasta 126 dispositivos en anillo, compartiendo el ancho de banda, de forma análoga a las redes Token Ring. No tiene mucho uso.

**FC-SW, Switch Fabric:**

Aprovecha la utilización de conmutadores o “switches” de “Fibre Channel” (FC) para la conexión de múltiples dispositivos, sin compartir el ancho de banda. Se utiliza habitualmente en arquitecturas de almacenamiento SAN.

Suelen utilizarse múltiples “switches” y múltiples puertos tanto en las cabinas de almacenamiento “Storage” y librería de cintas, como en los equipos cliente (en este caso servidores), de tal modo, que puedan definirse múltiples caminos entre ellos. Esto proporciona un mecanismo de alta disponibilidad, vital en las actuales infraestructuras de almacenamiento SAN corporativas. Los protocolos más usados para la gestión de caminos múltiples al almacenamiento, son MPIO (“Multi Path Input Output”) y “SecurePath” (protocolo propietario de HP).

Comentar que “Fibre Channel” es un Protocolo Multicapa:

- **FC0, la capa física (“physical layer”).** Cables, conectores, etc.
- **FC1, la capa de enlace (“data link layer”).** Realiza la codificación-decodificación.
- **FC2, la capa de red (“network layer”).** Es el corazón de “Fibre Channel”. Define los principales protocolos.
- **FC3, la capa de servicios comunes (“common services layer”).** Puede implementar funcionalidades como encriptación y RAID.
- **FC4, la capa de mapeo de protocolos (“protocol mapping layer”).** Es la capa, en la que otros protocolos son encapsulados para su entrega a FC2.

## Componentes de la Arquitectura de Almacenamiento SAN:

### Dispositivos Cliente:

Los Servidores realizarán un acceso transparente al almacenamiento SAN, como si se tratase de discos locales DAS, siendo esta la principal diferencia entre SAN y NAS (ya comentada). Para poder conectarse a la red de almacenamiento SAN, necesitarán de tarjetas HBA (Host Bus Adapter). Las tarjetas HBA, son dispositivos de conexión “Fibre Channel” (FC), que permitirán la conexión de estos equipos clientes a los “switches” de la red de almacenamiento SAN mediante cables de fibra.

Múltiples puertos “Fibre Channel”, facilitarán disponer de alta disponibilidad a través de un mayor número de caminos, consiguiendo así ser menos vulnerables a la caída de un “Switch de Fibra” o de un puerto de la cabina de almacenamiento, todo ello gracias a la utilización de protocolos como de MPIO (“Multi Path Input Output”) y “SecurePath” (protocolo propietario de HP).

En el caso de iSCSI, existen tarjetas HBA específicas, pero también pueden utilizarse tarjetas de red tradicionales, a poder ser de alto rendimiento (Gigabit Ethernet o más, pues actualmente ya existe tecnología Ethernet de 10Gbps), pudiendo del mismo modo disfrutar de alta disponibilidad.

### Equipos de Almacenamiento:

Las cabinas de almacenamiento suelen disponer de múltiples puertos para ofrecer alta disponibilidad, como ya hemos comentado. Del mismo modo suelen utilizar tecnologías RAID como RAID1 y RAID5 para ofrecer redundancia en el almacenamiento. Esto conlleva, que la pérdida de un disco, no ocasione pérdida de datos. Actualmente, en las cabinas de almacenamiento se suele configurar un RAID y sobre este crear las LUN (“Logical Unit Number”).

¿Qué es una LUN? Una LUN (“Logical Unit Number”) es una parte de este RAID, el cual, se presentará o asignará a un servidor para su utilización. Es decir, una LUN es un disco lógico, desde el punto de vista de la cabina de almacenamiento (Storage).

El servidor la verá como un disco más, aunque realmente se almacene físicamente repartido entre 45 discos físicos de la cabina de almacenamiento. Una LUN (“Logical Unit Number”) es una dirección que identifica dicha parte del RAID o disco lógico.

El término LUN (“Logical Unit Number”) es originario del protocolo SCSI. De hecho, una red de almacenamiento SAN utiliza el protocolo SCSI, trasportándolo a través de “Fibre Channel” (FC).

### Dispositivos de interconexión:

Los “switches” “Fiber Channel” (FC), son una de las partes más importantes de una red de almacenamiento SAN, al igual que ocurre en las redes Ethernet. Estos dispositivos son los que permitirán interconectar al resto de equipos de la red de almacenamiento SAN, como los “Hosts” o servidores, las cabinas de almacenamiento y las librerías de cintas. Sobre estos “switches” se realiza la configuración de “Zoning”. Evidentemente, en el caso de redes de almacenamiento SAN basadas

en iSCSI, hablaremos de “Switches Ethernet” (utilicen fibra o cobre, como medio de transporte), y en vez de “Zoning” utilizaremos VLAN.

## ¿Qué es el “World Wide Name” (WWN)?

**“World Wide Name” (WWN):**

Un “World Wide Name” (WWN) o “World Wide Identifier” (WWID) es un identificador único dentro de una red de almacenamiento SAN, es decir, es una dirección de 64 bits para identificar elementos en una red “Fibre Channel” (FC), similar a una dirección MAC.

Con este WWN hay que introducir dos conceptos “World Wide Node Name” (WWNN), y “World Wide Port Name” (WWPN).

**“World Wide Port Name” (WWPN):**

Un “World Wide Port Name” (WWPN) es un “World Wide Name” (WWN) asignado a un puerto de fibra en una red de almacenamiento SAN, similar a lo que es una dirección MAC en una red Ethernet.

**“World Wide Node Name” (WWNN):**

Un “World Wide Node Name” (WWNN) es un “World Wide Name” (WWN) asignado a un Nodo o Dispositivo “Fibre Channel” (FC) de una red de almacenamiento SAN.

Como consecuencia de lo anterior, un servidor habitualmente tendrá varios puertos de fibra, cada uno con su WWPN además del propio “World Wide Node Name”. Lo mismo ocurre con el resto de Dispositivos “Fibre Channel” (FC) de la red de almacenamiento SAN, como es el caso de las cabinas de almacenamiento y las librerías de cintas.

## “Switch Zoning” y “LUN Masking”

**“Zoning” o “Switch Zoning”:**

Los conmutadores o “switches” “Fibre Channel” (FC), permiten segmentar sus puertos en diferentes zonas, de forma similar a como ocurre con las VLAN en los conmutadores o “switches” Ethernet, de tal modo, que cada dispositivo sólo podrá comunicarse con el resto de dispositivos de su zona. Es importante tener en cuenta que cada puerto de fibra puede ser miembro de múltiples zonas.

Existen principalmente dos métodos de “Zoning”, por “Hardware” y por “Software”. Así, en el “Zoning” por “Software” (“Soft Zoning”) se restringe el acceso por nombre, sin embargo, cualquier servidor podrá acceder a cualquier dispositivo por su dirección de red. Por el contrario, el “Zoning” por “Hardware” (“Hard Zoning”) restringe las comunicaciones en los “switches” a través de filtrado de tramas, resultando mucho más seguro que el “Zoning” por “Software”.

Del mismo modo, suelen emplearse con dos tipos de atributos, el Puerto y el “World Wide Name”

(WWN). Así, el “Zoning” por Puerto (“Port Zoning”) permite restringir a un puerto de un “switch” con que otros puertos puede comunicarse. Por el contrario, el “Zoning” por Nombre (“Name Zoning”) restringe el acceso entre dispositivos en función del “World Wide Name” (WWN), resultando mucho más flexible aunque más inseguro que el “Zoning” por Puerto (“Port Zoning”).

Habitualmente, las redes de almacenamiento SAN suelen particionarse en múltiples zonas por motivos de seguridad para evitar interferencias que podrían generar problemas en el almacenamiento y así facilitar la gestión.

Todo dispositivo conectado a una red de almacenamiento SAN, debe poder acceder sólo a los puertos y dispositivos que necesita utilizar, para lo cual se configurarán las correspondientes zonas en los “switches” “Fibre Channel” (“Switching Zoning”).

En consecuencia, es posible que múltiples Servidores sean mapeados al mismo puerto de la cabina de almacenamiento, del mismo modo, que es posible que un único servidor sea mapeado a múltiples puertos de las cabina de almacenamiento.

**“LUN Masking”:**

Permite restringir que servidores o “Hosts”, pueden acceder a una determinada LUN (disco virtual) en una cabina de almacenamiento. En consecuencia, podemos ver el “LUN Masking”, como una configuración o proceso de autorización y seguridad dentro de la red de almacenamiento SAN.

Así, si una LUN no es asignada o presentada por la cabina de almacenamiento a un “Host” o Servidor específico, este no podrá acceder a dicha LUN (no tendrá visibilidad). En el caso de servidores en “Cluster”, es necesario que todos los servidores miembros del mismo “Cluster” tengan visibilidad sobre todas las LUN.

“LUN Masking” nos va a permitir que una LUN (o un grupo de LUN’s) en un puerto de la cabina de almacenamiento, sea mapeada a un WWN (o a un grupo de WWN, en el caso de un “Cluster”). “LUN Masking” se implementa de diferentes formas, en función del fabricante.

En la práctica, “Switch Zoning” y “LUN Masking” suelen utilizarse en conjunto. En consecuencia, para que un “Host” o Servidor pueda acceder a una LUN (disco virtual) de una cabina de almacenamiento, ambos deben pertenecer a la misma Zona (“Switch Zoning”), y además la cabina de almacenamiento debe permitir el acceso de dicho “Host” o Servidor a dicha LUN (“LUN Masking”).

## FCIP, iFCP e iSCSI:

### FCIP (“Fibre Channel over IP”):

También conocido como “Fibre Channel Tunneling” o “Storage Tunneling”, se trata de una tecnología basada en IP y desarrollada por el IETF (“Internet Engineering Task Force”), que permite la transmisión de tramas “Fibre Channel” (FC) a través de túneles IP, con el objetivo de

facilitar la extensión geográfica de las redes de almacenamiento SAN. Para ello se utilizan equipos conversores (“Edge Devices” ó “FCIP Gateways”) situados en la periferia de cada una de las redes SAN que se desean intercomunicar, de tal modo que dichos dispositivos, se limitarán a encapsular y reenviar las tramas FC (“Fibre Channel”) vía TCP/IP.

Gracias a la combinación de redes IP y redes SAN, es posible interconectar múltiples redes de almacenamiento SAN a través de distancias mucho mayores dando lugar a las redes de almacenamiento “SAN-to-SAN”.

**iFCP (“Internet Fibre Channel Protocol”):**

Se trata de una tecnología basada en IP ratificada por el IETF, que permite la transmisión de las capas superiores de tramas “Fibre Channel” (FC) a través de una red IP.

Una característica especial de iFCP, es que mapea direcciones IP a dispositivos “Fibre Channel” (FC) específicos de la red de almacenamiento SAN. Además, TCP es el responsable de gestionar la congestión, detección de errores y recuperación ante fallos.

FCIP (“Fibre Channel over IP”) e iFCP (“Internet Fibre Channel Protocol”) son protocolos muy parecidos, cuyo objetivo es la extensión de las redes de almacenamiento SAN, y cuya principal diferencia radica en el método elegido para cumplir su objetivo:

FCIP utiliza “Tunneling” mientras iFCP utiliza “Routing”.

**iSCSI (“Internet Small Computer System Interface”):**

Se trata de un standard de almacenamiento basado en IP y desarrollado por el IETF (“Internet Engineering Task Force”), que permite el envío de comandos SCSI a través de redes IP. Habitualmente iSCSI utiliza los puertos TCP-860 y TCP-3260.

En consecuencia, iSCSI permite implementar redes de almacenamiento SAN basadas en TCP/IP (sin utilizar “Fibre Channel” FC), de tal modo que se minimizan los costes, ya que es posible reutilizar los dispositivos de red de la LAN, como “switches” y “routers”, funcionando así sobre la infraestructura de red existente.

El protocolo iSCSI permite que los clientes (“initiators”) envíen comandos SCSI a los dispositivos de almacenamiento (“targets”) a través de IP.

Los clientes iSCSI (“iSCSI initiators”) pueden utilizar tarjetas de red Ethernet convencionales. Sin embargo, es conveniente que dichas tarjetas soporten TOE (“TCP Offload Engine”).

¿Qué es TOE (“TCP Offload Engine”)?, TOE es una característica de las tarjetas de red ethernet que libera a la CPU del equipo cliente de la realización de ciertas tareas propias del protocolo TCP/IP.

En iSCSI, en vez de utilizar WWN (como se hace sobre “Fibre Channel” FC), se utilizan direcciones IP y nombres iSCSI (“iSCSI Qualified Name” ó IQN). Los nombres IQN tiene el formato “iqn.yyyy-mm.{reverse domain name}”.

La principal diferencia entre iSCSI y “Fibre Channel”, es que con iSCSI se está trabajando directamente con TCP/IP, hecho que facilita la extensión geográfica de la red de almacenamiento SAN, evitando tener que utilizar tecnologías como FCIP e iFCP. Además, con iSCSI se puede aprovechar la electrónica de red ya establecida, evitando así la inversión en una nueva infraestructura de red basada en fibra.

→ **Topología**

fedora34

| [iSCSI Target Server] | | [iSCSI Initiator - multipath] |
|---|---|---|
| target.cadilinea.lan | <====> | multipath.cadilinea.lan |
| 192.168.10.160 | | 192.168.10.165 |
| 192.168.10.170 | | |

**# vim /etc/hosts**

```
192.168.10.160  target.cadilinea.lan        target
192.168.10.170  target.cadilinea.lan        target
192.168.10.165  multipath.cadilinea.lan     multipath
```

→ **Sincronización horaria** → **chronyd.service**

**target ~ # lsblk /dev/vd[b-z]**

```
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
vdb  251:16   0  20G  0 disk
vdc  251:32   0  20G  0 disk
vdd  251:48   0  20G  0 disk
```

**target ~ # dnf install targetcli device-mapper-multipath scsi-target-utils**

**target ~ # firewall-cmd --permanent --add-port=3260/tcp**

**target ~ # firewall-cmd --permanent --add-service=iscsi-target**

**target ~ # firewall-cmd --reload**

**target ~ # systemctl enable --now tgtd.service**

**==> Creación del target**

**1→ Creamos el target con iqn=iqn.2021-06.lan.cadilinea:target asignando un tid=1**

**target ~ # tgtadm --lld iscsi --mode target --op new --tid=1 --targetname iqn.2021-**

**06.lan.cadilinea:target**

**target ~ # tgtadm -lld iscsi --mode target --op show**

```
Target 1: iqn.2021-06.lan.cadilinea:target
    System information:
        Driver: 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:
    Account information:
    ACL information:
```

**target ~ # pvcreate /dev/vdb**
**target ~ # vgcreate vg_target /dev/vdb**
**target ~ # lvcreate -n lv_target -L 5G vg_target**

==> **Configuración** → **VOLÁTIL**

**2→ Añadimos un LUN en el target creado, y para backing-store → /dev/vg_target/lv_target**

(El LUN=0 está reservado e identifica al propio controlador del target. Se empieza la asignación por tanto desde 1).

**target ~ # tgtadm --lld iscsi --mode logicalunit --op new --tid 1 --lun 1 --backing-store /dev/vg_target/lv_target**

**target ~ # tgtadm --lld iscsi --mode target --op show**

```
Target 1: iqn.2021-06.lan.cadilinea:target
    System information:
        Driver: 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: 5369 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/vg_target/lv_target

Backing store flags:

Account information:

ACL information:

**3→ Control de acceso de los initiators → multipath → 192.168.10.165**

**target ~ # tgtadm --lld iscsi --mode target --op bind --tid 1 --initiator-address 192.168.10.165**

Con la opción → **--op unbind** deshabilitamos el acceso para esa dirección.

Con la opción → **--initiator-address ALL** permitimos cualquier dirección.

**4→ Definimos usuario carlos con el password 123456 para el control de acceso CHAP.**

En iSCSI se puede usar un mecanismo de autenticación mutua (initiator ante target [initiator authentication] y, opcionalmente, de target ante initiator [target authentication]) basado en el protocolo CHAP Challenge-Handshake Authentication Protocol.

Es necesario crear el par usuario-contraseña y vincular el usuario al target (y opcionalmente al initiator en autenticación inversa).

Initiator authetication: el initiator es autenticado por el target.

**target ~ # tgtadm --lld iscsi --mode account --op new --user carlos --password 123456**

**target ~ # tgtadm --lld iscsi --mode account --op bind --tid 1 --user carlos**

Nota: En los **initiators** debe habilitarse la autenticación CHAP e incluirse el par **carlos/123456** en el fichero de configuración /etc/iscsi/iscsid.conf (parámetros node.session.auth.{username,password}).

**→ Verificamos:**

**target ~ # tgtadm --lld iscsi --mode target --op show**

…

```
        LUN: 1
            Type: disk
            SCSI ID: IET     00010001
            SCSI SN: beaf11
            Size: 5369 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/vg_target/lv_target
            Backing store flags:
    Account information:
        carlos
    ACL information:
        192.168.10.165
```

**→ Eliminar LUNs y target (y en ese orden).**

**# tgtadm --lld iscsi --mode logicalunit --op delete --tid 1 --lun 1**

**# tgtadm --lld iscsi --mode target --op delete --tid 1**

**→ → La configuración realizada no es superstite después de un reboot.**

**==> Configuración → PERMANENTE**

**target ~ # cp /etc/tgt/conf.d/sample.conf /etc/tgt/conf.d/sample.conf.original**

**target ~ # cp /etc/tgt/conf.d/sample.conf.original /etc/tgt/conf.d/cadilinea_scsi.conf**

**target ~ # vim/etc/tgt/conf.d/cadilinea_scsi.conf**

**→ Debemos configurar ahora los parámetros indicados en la configuración volátil, y a saber:**

**1 →**

**target ~ # tgtadm --lld iscsi --mode target --op new --tid=1 --targetname iqn.2021-06.lan.cadilinea:target**

**2 →**

**target ~ # tgtadm  --lld iscsi --mode logicalunit --op new --tid 1 --lun 1 --backing-store /dev/vg_target/lv_target**

**3 →**

**target ~ # tgtadm --lld iscsi --mode target --op bind --tid 1 --initiator-address 192.168.10.165**

**4 →**

**target ~ # tgtadm --lld iscsi --mode account --op new --user carlos --password 123456**

**target ~ # tgtadm --lld iscsi --mode account --op bind --tid 1 --user carlos**

**5 →==> Comprobamos.**

**target ~ # tgtadm --lld iscsi --mode target --op show**

**6 →==> Editamos configuración.**

**target ~ # vim /etc/tgt/conf.d/cadilinea_scsi.conf**

```
<target iqn.2021-06.lan.cadilinea:target:lun1>
    backing-store /dev/vg_target/lv_target
    scsi_sn 2021-06-01
    vendor_id CADILINEA.COM
    incominguser carlos 123456
    outgoinguser carlos 123456
    initiator-address 192.168.10.165
</target>
```

**target ~ # systemctl status tgtd.service**

● tgtd.service - tgtd iSCSI target daemon

Loaded: loaded (/usr/lib/systemd/system/tgtd.service; enabled; vendor preset: disabled)

Active: active (running) since Fri 2021-05-14 14:30:05 CEST; 3h 12min ago

Process: 808 ExecStartPost=/bin/sleep 5 (code=exited, status=0/SUCCESS)

Process: 952 ExecStartPost=/usr/sbin/tgtadm --op update --mode sys --name State -v offline (code=exited, status=0/SUCCESS)

Process: 954 ExecStartPost=/usr/sbin/tgt-admin -e -c $TGTD_CONFIG (code=exited, status=0/SUCCESS)

Process: 985 ExecStartPost=/usr/sbin/tgtadm --op update --mode sys --name State -v ready (code=exited, status=0/SUCCESS)

Main PID: 807 (tgtd)

Tasks: 17 (limit: 3455)

Memory: 6.9M

CPU: 1.031s

CGroup: /system.slice/tgtd.service

└─807 /usr/sbin/tgtd -f

may 14 14:29:54 target.cadilinea.lan systemd[1]: Starting tgtd iSCSI target daemon...

may 14 14:30:05 target.cadilinea.lan tgtd[807]: tgtd: device_mgmt(246) sz:30 params:path=/dev/vg_target/lv_target

may 14 14:30:05 target.cadilinea.lan tgtd[807]: tgtd: bs_thread_open(409) 16

may 14 14:30:05 target.cadilinea.lan systemd[1]: Started tgtd iSCSI target daemon.

**target ~ # tgtadm --lld iscsi --mode target --op show**

…

LUN: 1

Type: disk

SCSI ID: IET 00010001

SCSI SN: 2021-06-01

Size: 5369 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/vg_target/lv_target
        Backing store flags:
    Account information:
        carlos
        carlos (outgoing)
    ACL information:
        192.168.10.165
target ~ # tgtadm --lld iscsi --mode target --op show --tid 1
nop_interval=0
nop_count=0
MaxRecvDataSegmentLength=8192
HeaderDigest=None
DataDigest=None
InitialR2T=Yes
MaxOutstandingR2T=1
ImmediateData=Yes
FirstBurstLength=65536
MaxBurstLength=262144
DataPDUInOrder=Yes
DataSequenceInOrder=Yes
ErrorRecoveryLevel=0
IFMarker=No
OFMarker=No
DefaultTime2Wait=2
```

DefaultTime2Retain=20

OFMarkInt=Reject

IFMarkInt=Reject

MaxConnections=1

RDMAExtensions=Yes

TargetRecvDataSegmentLength=262144

InitiatorRecvDataSegmentLength=262144

MaxOutstandingUnexpectedPDUs=0

MaxXmitDataSegmentLength=8192

MaxQueueCmd=128

**==> Creación del initiator → multipath**

**multipath ~ # lsblk**

```
NAME                      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sr0                        11:0    1 1024M  0 rom
zram0                     251:0    0  2,8G  0 disk [SWAP]
vda                       252:0    0   25G  0 disk
├─vda1                    252:1    0    1G  0 part /boot
└─vda2                    252:2    0   24G  0 part
  ├─fedora_fedora-pool00_tmeta   253:0    0   20M  0 lvm
  │ └─fedora_fedora-pool00-tpool 253:2    0 19,1G  0 lvm
  │   ├─fedora_fedora-root       253:3    0   15G  0 lvm  /
  │   └─fedora_fedora-pool00     253:4    0 19,1G  1 lvm
  └─fedora_fedora-pool00_tdata   253:1    0 19,1G  0 lvm
    └─fedora_fedora-pool00-tpool 253:2    0 19,1G  0 lvm
      ├─fedora_fedora-root       253:3    0   15G  0 lvm  /
      └─fedora_fedora-pool00     253:4    0 19,1G  1 lvm
```

**multipath ~ # dnf install iscsi-initiator-utils**

**multipath ~ # systemctl enable --now iscsid.service**

**multipath ~ # systemctl status iscsid.service**

● iscsid.service - Open-iSCSI

Loaded: loaded (/usr/lib/systemd/system/iscsid.service; enabled; vendor preset: disabled)

Active: active (running) since Fri 2021-05-14 18:11:51 CEST; 20s ago

TriggeredBy: ● iscsid.socket

Docs: man:iscsid(8)

man:iscsiuio(8)

man:iscsiadm(8)

Main PID: 1085 (iscsid)

Status: "Ready to process requests"

Tasks: 1 (limit: 3455)

Memory: 2.8M

CPU: 21ms

CGroup: /system.slice/iscsid.service

└─1085 /usr/sbin/iscsid -f

may 14 18:11:51 multipath.cadilinea.lan systemd[1]: Starting Open-iSCSI...

may 14 18:11:51 multipath.cadilinea.lan systemd[1]: Started Open-iSCSI.

**multipath ~ # iscsiadm -m discovery -t st -p 192.168.10.160**

192.168.10.160:3260,1 iqn.2021-06.lan.cadilinea:target:lun1

**multipath ~ # vim /etc/iscsi/initiatorname.iscsi**

#InitiatorName=iqn.1994-05.com.redhat:9255f1aa827a

InitiatorName=iqn.2021-06.lan.cadilinea:target:lun1

**multipath ~ # cat /etc/iscsi/iscsid.conf | grep -v '^#' | grep ^[^#]]**

iscsid.startup = /bin/systemctl start iscsid.socket iscsiuio.socket

node.startup = automatic

node.leading_login = No

node.session.timeo.replacement_timeout = 120

node.conn[0].timeo.login_timeout = 15

node.conn[0].timeo.logout_timeout = 15

node.conn[0].timeo.noop_out_interval = 5

node.conn[0].timeo.noop_out_timeout = 5

node.session.err_timeo.abort_timeout = 15

node.session.err_timeo.lu_reset_timeout = 30

node.session.err_timeo.tgt_reset_timeout = 30

node.session.initial_login_retry_max = 8

node.session.cmds_max = 128

node.session.queue_depth = 32

node.session.xmit_thread_priority = -20

node.session.iscsi.InitialR2T = No

node.session.iscsi.ImmediateData = Yes

node.session.iscsi.FirstBurstLength = 262144

node.session.iscsi.MaxBurstLength = 16776192

node.conn[0].iscsi.MaxRecvDataSegmentLength = 262144

node.conn[0].iscsi.MaxXmitDataSegmentLength = 0

discovery.sendtargets.iscsi.MaxRecvDataSegmentLength = 32768

node.conn[0].iscsi.HeaderDigest = None

node.session.nr_sessions = 1

node.session.reopen_max = 0

node.session.iscsi.FastAbort = Yes

node.session.scan = auto

node.session.auth.authmethod = CHAP

node.session.auth.username = carlos

node.session.auth.password = 123456

**multipath ~ # systemctl restart iscsid.service**

**multipath ~ # iscsiadm -m discovery -t st -p 192.168.10.160**

192.168.10.160:3260,1 iqn.2021-06.lan.cadilinea:target:lun1

**multipath ~ # iscsiadm -m discovery -t st -p 192.168.10.170**

192.168.10.170:3260,1 iqn.2021-06.lan.cadilinea:target:lun1

**multipath ~ # iscsiadm -m node**

192.168.10.160:3260,1 iqn.2021-06.lan.cadilinea:target:lun1

192.168.10.170:3260,1 iqn.2021-06.lan.cadilinea:target:lun1

**multipath ~ # iscsiadm --mode node --targetname iqn.2021-06.lan.cadilinea:target:lun1 --portal 192.168.10.160 --login**

→ (De forma abreviada → **iscsiadm --mode node -T iqn.2021-06.lan.cadilinea:target:lun1 -P 192.168.10.160 -l**)

Logging in to [iface: default, target: iqn.2021-06.lan.cadilinea:target:lun1, portal: 192.168.10.160,3260]

Login to [iface: default, target: iqn.2021-06.lan.cadilinea:target:lun1, portal: 192.168.10.160,3260] successful.

**multipath ~ # lsblk /dev/sda**

NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT

sda 8:0 0 5G 0 disk

→ **Actualizar información con la opción → --op=update**

```
# iscsiadm --mode node --targetname "iqn.2018-09.net.cda.discos:pruebas" \
        --portal "192.168.100.11:3260" \
        --op=update --name node.session.auth.authmethod --value=CHAP

# iscsiadm --mode node --targetname "iqn.2018-09.net.cda.discos:pruebas" \
        --portal "192.168.100.11:3260" \
        --op=update --name node.session.auth.username --value=someuser

# iscsiadm --mode node --targetname "iqn.2018-09.net.cda.discos:pruebas" \
        --portal "192.168.100.11:3260" \
        --op=update --name node.session.auth.password --value=secret
```

→ **Desconexión/Conexión:**

**multipath ~ # iscsiadm --mode node -T iqn.2021-06.lan.cadilinea:target:lun1 -P 192.168.10.160 -u**

Logging out of session [sid: 3, target: iqn.2021-06.lan.cadilinea:target:lun1, portal:

192.168.10.160,3260]

Logout of [sid: 3, target: iqn.2021-06.lan.cadilinea:target:lun1, portal: 192.168.10.160,3260] successful.

**multipath ~ # iscsiadm --mode node -T iqn.2021-06.lan.cadilinea:target:lun1 -P 192.168.10.160 -l**

**→ Creación del FS|xfs**

**multipath ~ # lsblk /dev/sda**

NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT

sda 8:0 0 5G 0 disk

**multipath ~ # mkfs.xfs /dev/sda**

meta-data=/dev/sda isize=512 agcount=4, agsize=327680 blks

= sectsz=512 attr=2, projid32bit=1

= crc=1 finobt=1, sparse=1, rmapbt=0

= reflink=1 bigtime=0

data = bsize=4096 blocks=1310720, imaxpct=25

= sunit=0 swidth=0 blks

naming =version 2 bsize=4096 ascii-ci=0, ftype=1

log =internal log bsize=4096 blocks=2560, version=2

= sectsz=512 sunit=0 blks, lazy-count=1

realtime =none extsz=4096 blocks=0, rtextents=0

**multipath ~ # mkdir /mnt/iscsi**

**multipath ~ # vim /etc/fstab**

…

**/dev/sda /mnt/iscsi xfs _netdev,defaults 0 0**

**multipath ~ # mount -a**

**multipath ~ # lsblk /dev/sda**

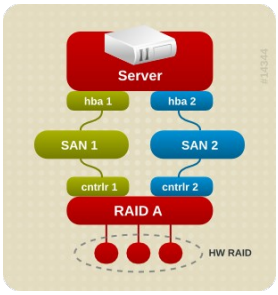
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT

sda 8:0 0 5G 0 disk /mnt/iscsi

→ **Instalación** → **multipath**

→ **MULTIPATH CONFIGURATIONS**

1→ **Active/Passive** multipath configuration with one RAID device




In this configuration, there are two Host Bus Adapters (HBAs) on the server, two SAN switches, and two RAID controllers. Following are the possible failure in this configuration:

→ HBA failure

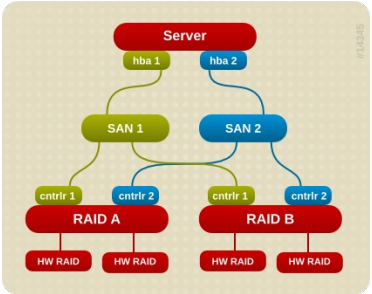
→ Fibre Channel cable failure

→ SAN switch failure

→ Array controller port failure

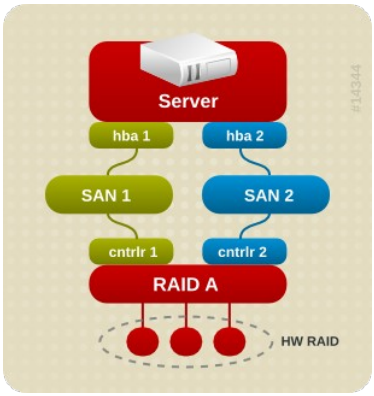
With DM Multipath configured, a failure at any of these points causes DM Multipath to switch to the alternate I/O path. Here, there is one I/O path that goes through hba1, SAN1, and cntrlr1 and a second I/O path that goes through hba2, SAN2, and cntrlr2.

2→ **Active/Passive** multipath configuration with two RAID devices




In this configuration, there are two HBAs on the server, two SAN switches, and two RAID devices with two RAID controllers each. With DM Multipath configured, a failure at any of the points of the I/O path to either of the RAID devices causes DM Multipath to switch to the alternate I/O path for that device. Here, there are two I/O paths to each RAID device.

### 3→ **Active/Active** multipath configuration with one RAID device




In this configuration, there are two HBAs on the server, two SAN switch, and two RAID controllers. Here, I/O can be spread among these two paths.

→ **DM Multipath components**

| | |
|---|---|
| **dm_multipath** kernel module | Reroutes I/O and supports failover for paths and path groups. |
| **mpathconf** utility | Configures and enables device mapper multipathing. |
| **multipath** command | Lists and configures the multipath devices. It is also executed by udev whenever a block device is added, to determine if the device should be part of a multipath device or not. |
| **multipathd** daemon | Automatically creates and removes multipath devices and monitors paths; as paths fail and come back, it may update the multipath device. Allows interactive changes to multipath devices. Reload the service if there are any changes to the /etc/multipath.conf file. |
| **kpartx** command | Creates device mapper devices for the partitions on a device. This command is automatically executed by udev when multipath devices are created to create partition devices on top of them. The kpartx command is provided in its own package, but the device-mapper-multipath package depends on it. |
| **mpathpersist** | Sets up SCSI-3 persistent reservations on multipath devices. This command works similarly to the way sg_persist works for SCSI devices that are not multipathed, but it handles setting persistent reservations on all paths of a multipath device. It coordinates with multipathd to ensure that the |

reservations are set up correctly on paths that are added later. To use this functionality, the `reservation_key` attribute must be defined in the `/etc/multipath.conf` file. Otherwise the `multipathd` daemon will not check for persistent reservations for newly discovered paths or reinstated paths.

**→ → multipath → Es una piedra angular para un despliegue LUN, y donde se buscan rutas alternativas mas fáciles de interpretar.**

**multipath ~ # dnf install device-mapper-multipath-***

**multipath ~ # systemctl enable --now multipathd.service**

**multipath ~ # modprobe dm_multipath**

**multipath ~ # lsmod | grep -i dm_multipath**

dm_multipath 40960 0

**multipath ~ # multipath -ll**

Apr 07 19:54:12 | /etc/multipath.conf does not exist, blacklisting all devices.

Apr 07 19:54:12 | You can run "/sbin/mpathconf --enable" to create

Apr 07 19:54:12 | /etc/multipath.conf. See man mpathconf(8) for more details

**multipath ~ # apropos multipath**

mpathconf (8) - A tool for configuring device-mapper-multipath

mpathpersist (8) - Manages SCSI persistent reservations on dm multipath devices.

multipath (8) - Device mapper target autoconfig.

multipath.conf (5) - multipath daemon configuration file.

multipathd (8) - Multipath daemon.

**multipath ~ # tree /usr/share/doc/device-mapper-multipath**

/usr/share/doc/device-mapper-multipath

├── multipath.conf

├── README

└── README.alua

**multipath ~ # cp /usr/share/doc/device-mapper-multipath/multipath.conf /etc/multipath.conf**

**multipath ~ # multipath -ll**

**multipath ~ # systemctl restart multipathd.service**

**multipath ~ # systemctl status multipathd.service**

**→ Una vez logueados obtenemos todos los bloques → mpath**

**multipath ~ # iscsiadm -m node -T iqn.2021-04.lan.cadilinea:target -p 192.168.10.160 -l**

**multipath ~ # lsblk /dev/sd[a-z]**

...

sda  xfs  6a06c1cd-5718-4534-9a1a-3d8a0cef1b32

**multipath ~ # cat /etc/multipath.conf | grep -v '^#' | grep ^[^#]]**

```
defaults {
        user_friendly_names yes
        find_multipaths yes
}
```

**multipath ~ # mpathconf --enable --with_multipathd y**

**multipath ~ # mpathconf --enable --user_friendly_names n**

**multipath ~ # mpathconf --enable --find_multipaths n**

**multipath ~ # cat /etc/multipath.conf | grep -v '^#' | grep ^[^#]**

```
defaults {
        user_friendly_names no
        find_multipaths no
}
blacklist {
}
```

**multipath ~ # systemctl restart multipathd.service**

**multipath ~ # multipath -ll**

**360000000000000000e00000000010001** dm-5 CADILINE,VIRTUAL-DISK

```
size=5.0G features='0' hwhandler='0' wp=rw
`-+- policy='service-time 0' prio=1 status=active
  `- 7:0:0:1 sda 8:0 active ready running
```

**multipath ~ # ls /dev/mapper/360000000000000000e00000000010001 -al**

lrwxrwxrwx. 1 root root 7 may 16 08:47 /dev/mapper/360000000000000000e00000000010001 -> ../dm-5

**multipath ~ # cat /etc/multipath.conf | grep -v '^#' | grep ^[^#]]**

```
defaults {
        user_friendly_names no
        find_multipaths no
}
multipaths {
        multipath {
                wwid                    360000000000000000e00000000010001
                alias                   aurora
                path_grouping_policy    multibus
                path_selector           "round-robin 0"
                failback                manual
                rr_weight               priorities
                no_path_retry           5
        }
}
blacklist {
}
```

**multipath ~ # systemctl restart multipathd.service**

**multipath ~ # systemctl status multipathd.service**

● multipathd.service - Device-Mapper Multipath Device Controller

Loaded: loaded (/usr/lib/systemd/system/multipathd.service; enabled; vendor preset: enabled)

Active: active (running) since Sun 2021-05-16 09:12:28 CEST; 6s ago

TriggeredBy: ● multipathd.socket

Process: 2319 ExecStartPre=/sbin/modprobe -a scsi_dh_alua scsi_dh_emc scsi_dh_rdac dm-multipath (code=exited, status=0/SUCCESS)

Process: 2320 ExecStartPre=/sbin/multipath -A (code=exited, status=0/SUCCESS)

Main PID: 2321 (multipathd)

Status: "up"

Tasks: 7

Memory: 18.8M

CPU: 26ms

CGroup: /system.slice/multipathd.service

└─2321 /sbin/multipathd -d -s

may 16 09:12:28 multipath.cadilinea.lan systemd[1]: Starting Device-Mapper Multipath Device Controller...

may 16 09:12:28 multipath.cadilinea.lan multipathd[2321]: --------start up--------

may 16 09:12:28 multipath.cadilinea.lan multipathd[2321]: read /etc/multipath.conf

may 16 09:12:28 multipath.cadilinea.lan multipathd[2321]: failed to increase buffer size

may 16 09:12:28 multipath.cadilinea.lan multipathd[2321]: path checkers start up

may 16 09:12:28 multipath.cadilinea.lan multipathd[2321]: aurora: reload [0 10485760 multipath 1 queue_if_no_path 0 1 1 round-robin 0 1>

may 16 09:12:28 multipath.cadilinea.lan systemd[1]: Started Device-Mapper Multipath Device Controller.

**multipath ~ # tree /dev/mapper/**

/dev/mapper/

├── aurora -> ../dm-5

├── control

├── fedora_fedora-pool00 -> ../dm-4

├── fedora_fedora-pool00_tdata -> ../dm-1

├── fedora_fedora-pool00_tmeta -> ../dm-0

├── fedora_fedora-pool00-tpool -> ../dm-2

└── fedora_fedora-root -> ../dm-3

0 directories, 7 files

**multipath ~ # vim /etc/fstab**

…

/dev/mapper/aurora /mnt/iscsi xfs _netdev,defaults 0 0

**multipath ~ # dmsetup ls**

**aurora(253:5)**

...

**multipath ~ # multipathd show maps format "%n %w %d %s"**

name uuid sysfs vend/prod/rev

**aurora** 360000000000000000e00000000010001 dm-5 CADILINE,VIRTUAL-DISK

**multipath ~ # multipathd show maps raw format "%n %w %d %s"**

aurora 360000000000000000e00000000010001 dm-5 CADILINE,VIRTUAL-DISK

**→ Modo consola**

**multipath ~ # multipathd -k**

**multipathd> show paths**

hcil dev dev_t pri dm_st chk_st dev_st next_check

7:0:0:1 sda 8:0 1 active ready running X..... 2/20

**<CTRL-D>**

**→ → Replanteamiento del multipath con targetcli → fedora34**

**(No exigido en la certificación).**

**==> → Restauramos las máquinas a su estado inicial.**

**→Configuración → targetcli**

**target ~ # lsblk /dev/vd[b-z]**

NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT

vdb 252:16 0 20G 0 disk

vdc 252:32 0 20G 0 disk

vdd 252:48 0 20G 0 disk

**target ~ # pvcreate /dev/vdc**

**target ~ # vgcreate vg_target-02 /dev/vdc**

**target ~ # lvcreate -n lv_target-02 -L 5G vg_target-02**

**target ~ # lsblk /dev/vdc**

```
NAME                      MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
vdc                       252:32   0  20G  0 disk
└─vg_target--02-lv_target--02 253:6    0   5G  0 lvm
```

**==> El target o target's administrarán los dispositivos y tipos de dispositivos objeto.**

**→ Instalar 'target' → target**

**target ~ # dnf install targetcli  scsi-target-utils device-mapper-multipath**

**target ~ # firewall-cmd --permanent --add-port=3260/tcp**

**target ~ # firewall-cmd --reload**

**→ Refresco de comandos básicos.**

**target ~ # apropos tgt**

```
tgt-admin (8)        - Linux SCSI Target Configuration Tool
tgt-setup-lun (8)    - Helper script that creates a target, adds a device to ...
tgtadm (8)           - Linux SCSI Target Administration Utility
```

**target ~ # apropos targetcli**

```
targetcli (8)        - administration shell for storage targets
targetclid (8)       - daemon component for targetcli
```

**target ~ # apropos iscsiadm**

```
iscsiadm (8)         - open-iscsi administration utility
```

**target ~ # apropos multipath**

```
mpathconf (8)        - A tool for configuring device-mapper-multipath
mpathpersist (8)     - Manages SCSI persistent reservations on dm multipath d...
multipath (8)        - Device mapper target autoconfig.
multipath.conf (5)   - multipath daemon configuration file.
```

multipathd (8)  - Multipath daemon.

**target ~ # tree /etc/tgt**

/etc/tgt

├── conf.d

│ └── sample.conf

├── targets.conf

└── tgtd.conf

**target ~ # tree /etc/target/**

**/etc/target/**

├── backup

│ └── saveconfig-20210403-09:49:00-json.gz

└── saveconfig.json

**→ Creación de los target's**

**target ~ # targetcli**

targetcli shell version 2.1.53

Copyright 2011-2013 by Datera, Inc and others.

For help on commands, type 'help'.

**/> ls**

o- / [...]

o- backstores [...]

| o- block [Storage Objects: 0]

| o- fileio [Storage Objects: 0]

| o- pscsi [Storage Objects: 0]

| o- ramdisk [Storage Objects: 0]

o- iscsi [Targets: 0]

o- loopback [Targets: 0]

o- vhost [Targets: 0]

**1 => No ejecutar cd despues de ejecutar comandos**

**/> set global auto_cd_after_create=false**

**/> cd backstores/block**

**2 => Crear backstore**

**/backstores/block> create dev=/dev/mapper/vg_target--02-lv_target--02 name=target-02**

Created block storage object target-02 using /dev/mapper/vg_target--02-lv_target--02.

**3 => Instanciar target**

**/backstores/block> cd /iscsi**

**/iscsi> create iqn.2021-06.lan.cadilinea:target-02**

Created target iqn.2021-06.lan.cadilinea:target-02.

Created TPG 1.

Global pref auto_add_default_portal=true

Created default portal listening on all IPs (0.0.0.0), port 3260.

**4 => Crear ACL**

**/iscsi> cd iqn.2021-06.lan.cadilinea:target-02/tpg1/acls**

**/iscsi/iqn.20...-02/tpg1/acls> create iqn.2021-06.lan.cadilinea:target-02:initiator-02**

Created Node ACL for iqn.2021-06.lan.cadilinea:target-02:initiator-02

**5 => Crear TPG1 → CHAP**

**/iscsi/iqn.20...-02/tpg1/acls> cd /iscsi/iqn.2021-06.lan.cadilinea:target-02/tpg1/**

**/iscsi/iqn.20...arget-02/tpg1> set attribute generate_node_acls=1**

**/iscsi/iqn.20...arget-02/tpg1> set attribute demo_mode_write_protect=0**

**/iscsi/iqn.20...arget-02/tpg1> set attribute authentication=1**

**/iscsi/iqn.20...arget-02/tpg1> set auth userid=carlos**

**/iscsi/iqn.20...arget-02/tpg1> set auth password=123456**

**6 => Exportar luns**

**/iscsi/iqn.20...arget-02/tpg1> cd /iscsi/iqn.2021-06.lan.cadilinea:target-02/tpg1/luns**

**/iscsi/iqn.20...-02/tpg1/luns> create /backstores/block/target-02**

Created LUN 0.

Created LUN 0->0 mapping in node ACL iqn.2021-06.lan.cadilinea:target-02:initiator-02

### 7 => Salvamos configuración

**/iscsi/iqn.20...:initiator-02> cd /**

**/> saveconfig**

Last 10 configs saved in /etc/target/backup/.

Configuration saved to /etc/target/saveconfig.json

**/> ls**

```
o- / ..... [...]
  o- backstores ..... [...]
  | o- block ..... [Storage Objects: 1]
  | | o- target-02 ..... [/dev/mapper/vg_target--02-lv_target--02 (5.0GiB) write-thru activated]
  | |   o- alua ..... [ALUA Groups: 1]
  | |     o- default_tg_pt_gp ..... [ALUA state: Active/optimized]
  | o- fileio ..... [Storage Objects: 0]
  | o- pscsi ..... [Storage Objects: 0]
  | o- ramdisk ..... [Storage Objects: 0]
  o- iscsi ..... [Targets: 1]
  | o- iqn.2021-06.lan.cadilinea:target-02 ..... [TPGs: 1]
  |   o- tpg1 ..... [gen-acls, tpg-auth, 1-way auth]
  |     o- acls ..... [ACLs: 1]
  |     | o- iqn.2021-06.lan.cadilinea:target-02:initiator-02 ..... [auth via tpg, Mapped LUNs: 1]
  |     |   o- mapped_lun0 ..... [lun0 block/target-02 (rw)]
  |     o- luns ..... [LUNs: 1]
  |     | o- lun0 ..... [block/target-02 (/dev/mapper/vg_target--02-lv_target--02) (default_tg_pt_gp)]
  |     o- portals ..... [Portals: 1]
  |       o- 0.0.0.0:3260 ..... [OK]
  o- loopback ..... [Targets: 0]
  o- vhost ..... [Targets: 0]
```

### 7 => Salvar/Salir

**/> exit**

Global pref auto_save_on_exit=true

Last 10 configs saved in /etc/target/backup/.

Configuration saved to /etc/target/saveconfig.json

**target ~ # systemctl enable --now target.service**

**target ~ # ss -nltp**

State Recv-Q Send-Q Local Address:Port Peer Address:Port Process

LISTEN 0 128 0.0.0.0:22 0.0.0.0:* users:(("sshd",pid=804,fd=4))

LISTEN 0 4096 0.0.0.0:3260 0.0.0.0:* users:(("tgtd",pid=807,fd=6))

…

→ Desde **multipath**

**target ~ # ssh multipath**

**multipath ~ # systemctl enable --now iscsid.service**

**multipath ~ # vim /etc/iscsi/initiatorname.iscsi**

#InitiatorName=iqn.1994-05.com.redhat:33641a34376

InitiatorName=iqn.2021-06.lan.cadilinea:target-02

**multipath ~ # vim /etc/iscsi/iscsid.conf**

…

**node.session.auth.authmethod = CHAP**

**node.session.auth.username = carlos**

**node.session.auth.password = 123456**

**multipath ~ # systemctl restart iscsid.service**

→ Descubrimos y nos logueamos en el nodo

**multipath ~ # iscsiadm --mode discovery --type sendtargets --portal 192.168.10.160:3260**

192.168.10.160:3260,1 iqn.2021-06.lan.cadilinea:target-02

**multipath ~ # iscsiadm --mode node -T iqn.2021-06.lan.cadilinea:target-02 --portal 192.168.10.160:3260 --login**

Logging in to [iface: default, target: iqn.2021-06.lan.cadilinea:target-02, portal: 192.168.10.160,3260]

Login to [iface: default, target: iqn.2021-06.lan.cadilinea:target-02, portal: 192.168.10.160,3260]

successful.

**multipath ~ # fdisk -l | grep sd**

Disco /dev/sda: 5 GiB, 5368709120 bytes, 10485760 sectores

**multipath ~ # mkfs.xfs /dev/sda**

```
meta-data=/dev/sda               isize=512    agcount=4, agsize=327680 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=1, sparse=1, rmapbt=0
         =                       reflink=1    bigtime=0
data     =                       bsize=4096   blocks=1310720, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096   ascii-ci=0, ftype=1
log      =internal log           bsize=4096   blocks=2560, version=2
         =                       sectsz=512   sunit=0 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
```

**multipath ~ # tree /var/lib/iscsi/**

```
/var/lib/iscsi/
├── ifaces
├── isns
├── nodes
│   └── iqn.2021-06.lan.cadilinea:target-02
│       └── 192.168.10.160,3260,1
│           └── default
├── send_targets
│   └── 192.168.10.160,3260
│       ├── iqn.2021-06.lan.cadilinea:target-02,192.168.10.160,3260,1,default -> /var/lib/iscsi/nodes/iqn.2021-06.lan.cadilinea:target-02/192.168.10.160,3260,1
│       └── st_config
├── slp
```

└── static

**multipath ~ # mkdir /mnt/iscsi**

**multipath ~ # mount /dev/sda /mnt/iscsi/ -o _netdev,defaults,rw**

**multipath ~ # touch /mnt/iscsi/Prueba-multipath.txt**

**→ multipath es un concepto quizás algo abstracto, el cual permite identificar objetivos de bloque o similares, de distinta índole y de una forma amigable, y permitir su acceso y en alta disponibilidad, a través de diferentes rutas → → consistencia del principio HA.**

**→ Instalar 'multipath'.**

**→ Nuestro objetivo's multipath's son equipos sin recursos. Sus único/s discos son para su propio sistema operativo y poco mas, …**

**multipath ~ # dnf install device-mapper-multipath-***

**multipath ~ # systemctl enable --now multipathd.service**

**multipath ~ # modprobe dm_multipathh**

**multipath ~ # lsmod | grep -i dm_multipathh**

dm_multipath 40960 0

**multipath ~ # multipath -ll**

417.370116 | /etc/multipath.conf does not exist, blacklisting all devices.

417.370277 | You can run "/sbin/mpathconf --enable" to create

417.370445 | /etc/multipath.conf. See man mpathconf(8) for more details

**multipath ~ # cp /usr/share/doc/device-mapper-multipath/multipath.conf /etc/multipath.conf**

**multipath ~ # systemctl restart multipathd.service**

**multipath ~ # systemctl status multipathd.service**

**multipath ~ # mpathconf --enable --with_multipathd y**

**multipath ~ # mpathconf --enable --user_friendly_names n**

**multipath ~ # mpathconf --enable --find_multipaths n**

**multipath ~ # cat /etc/multipath.conf | grep -v '^#' | grep ^[^#]**

```
defaults {
        user_friendly_names no
        find_multipaths no
}
blacklist {
}
```

**multipath ~ # systemctl restart multipathd.service**

**multipath ~ # multipath -ll**

**multipath ~ # iscsiadm -m discovery -t st -p 192.168.10.1600**

192.168.10.160:3260,1 iqn.2021-06.lan.cadilinea:target-02

**multipath ~ # multipath -ll**

**multipath ~ # iscsiadm -m node -T iqn.2021-06.lan.cadilinea:target-02 -u**

Logging out of session [sid: 1, target: iqn.2021-06.lan.cadilinea:target-02, portal: 192.168.10.160,3260]

Logout of [sid: 1, target: iqn.2021-06.lan.cadilinea:target-02, portal: 192.168.10.160,3260] successful.

**multipath ~ # iscsiadm -m node -T iqn.2021-06.lan.cadilinea:target-02 -l**

Logging in to [iface: default, target: iqn.2021-06.lan.cadilinea:target-02, portal: 192.168.10.160,3260]

Login to [iface: default, target: iqn.2021-06.lan.cadilinea:target-02, portal: 192.168.10.160,3260] successful.

**multipath ~ # multipath -ll**

```
36001405522045a021e84716bd56ecf62 dm-5 LIO-ORG,target-02
size=5.0G features='0' hwhandler='1 alua' wp=rw
`-+- policy='service-time 0' prio=50 status=active
```

`- 7:0:0:0 **sda 8:16 active ready running**

**multipath ~ # cat /etc/multipath.conf | grep -v '^#' | grep ^[^#]]**

```
defaults {
        user_friendly_names no
        find_multipaths no
}
multipaths {
        multipath {
                wwid                    36001405522045a021e84716bd56ecf62
                alias                   margarita
                path_grouping_policymultibus
                path_selector           "round-robin 0"
                failback                manual
                rr_weight               priorities
                no_path_retry           5
        }
}
blacklist {
}
```

**multipath ~ # lsblk /dev/sda**

```
NAME                                MAJ:MIN RM SIZE RO TYPE  MOUNTPOINT
sda                                   8:16   0   5G  0 disk
└─36001405522045a021e84716bd56ecf62 253:5    0   5G  0 mpath
```

**multipath ~ # systemctl restart multipathd.service**

**multipath ~ # lsblk /dev/sdb -f**

```
NAME       FSTYPE      FSVER LABEL UUID                                 FSAVAIL FSUSE%
```

MOUNTPOINT

sda mpath_member bafbd8df-68b1-44ee-b787-ce3d0bab5fcc

└─**margarita xfs bafbd8df-68b1-44ee-b787-ce3d0bab5fcc**

**multipath ~ # mount /dev/mapper/margarita /mnt/iscsi/ -o _netdev,defaults,rw**

**multipath ~ # ls /mnt/iscsi/**

Prueba-multipath.txt

→ → **iscsiadm incorpora una panoplia de comandos sencillos para manejar el target**

→ **iscsiadm opciones principales:**

**-m, --mode op**

specify the mode. op must be one of discovery, discoverydb, node, fw, host iface or session.

If no other options are specified: for discovery, discoverydb and node, all of their respective records are displayed;

for session, all active sessions and connections are displayed; for fw, all boot firmware values are displayed; for

host, all iSCSI hosts are displayed; and for iface, all ifaces setup in /var/lib/iscsi/ifaces are displayed.

**-t, --type=type**

type must be **sendtargets** (or abbreviated as **st**), slp, isns or fw. Currently only sendtargets, fw, and iSNS is sup-

ported, see the DISCOVERY TYPES section.

This option is only valid for discovery mode.

**-p, --portal=ip[:port]**

Use target portal with ip-address ip and port. If port is not passed in the default port value is 3260.

IPv6 addresses can be specified as [ddd.ddd.ddd.ddd]:port or ddd.ddd.ddd.ddd.

Hostnames can also be used for the ip argument.

This option is only valid for discovery, or for node operations with the new operator.

This should be used along with --target in node mode, to specify what the open-iscsi docs

refer to as a node or node

record. Note: open-iscsi's use of the word node, does not match the iSCSI RFC's iSCSI Node term.

**-T, --targetname=targetname**

Use target targetname.

This should be used along with --portal in node mode, to specify what the open-iscsi docs refer to as a node or node

record. Note: open-iscsi's use of the word node, does not match the iSCSI RFC's iSCSI Node term.

**-l, --login**

For node and fw mode, login to a specified record. For discovery mode, login to all discovered targets.

This option is only valid for discovery and node modes.

**-u, --logout**

logout for a specified record.

This option is only valid for node and session mode.

**-P, --print=printlevel**

If in node mode print nodes in tree format. If in session mode print sessions in tree format. If in discovery mode

print the nodes in tree format.

**-o, --op=op**

Specifies a database operator op. op must be one of **new**, **delete**, **update**, **show** or **nonpersistent**.

…

→ **Comandos relevantes** → **tgadm.**

**target ~ # tgtadm --help**

Linux SCSI Target administration utility, version 1.0.79

Usage: tgtadm [OPTION]

→ **<driver> ==> iscsi**

**--lld <driver> --mode target --op new --tid <id> --targetname <name>**

add a new target with <id> and <name>. <id> must not be zero.

**--lld <driver> --mode target --op delete [--force] --tid <id>**

delete the specific target with <id>.

With force option, the specific target is deleted

even if there is an activity.

**--lld <driver> --mode target --op show**

show all the targets.

**--lld <driver> --mode target --op show --tid <id>**

show the specific target's parameters.

--lld <driver> --mode target --op update --tid <id> --name <param> --value <value>

change the target parameters of the target with <id>.

--lld <driver> --mode target --op bind --tid <id> --initiator-address <address>

--lld <driver> --mode target --op bind --tid <id> --initiator-name <name>

enable the target to accept the specific initiators.

--lld <driver> --mode target --op unbind --tid <id> --initiator-address <address>

--lld <driver> --mode target --op unbind --tid <id> --initiator-name <name>

disable the specific permitted initiators.

--lld <driver> --mode logicalunit --op new --tid <id> --lun <lun>

--backing-store <path> --bstype <type> --bsopts <bs options> --bsoflags <options>

add a new logical unit with <lun> to the specific

target with <id>. The logical unit is offered

to the initiators. <path> must be block device files

(including LVM and RAID devices) or regular files.

bstype option is optional.

bsopts are specific to the bstype.

bsoflags supported options are sync and direct

(sync:direct for both).

--lld <driver> --mode logicalunit --op delete --tid <id> --lun <lun>

delete the specific logical unit with <lun> that

the target with <id> has.

--lld <driver> --mode account --op new --user <name> --password <pass>

add a new account with <name> and <pass>.

--lld <driver> --mode account --op delete --user <name>

delete the specific account having <name>.

--lld <driver> --mode account --op bind --tid <id> --user <name> [--outgoing]

add the specific account having <name> to

the specific target with <id>.

<user> could be <IncomingUser> or <OutgoingUser>.

If you use --outgoing option, the account will

be added as an outgoing account.

--lld <driver> --mode account --op unbind --tid <id> --user <name> [--outgoing]

delete the specific account having <name> from specific

target. The --outgoing option must be added if you

delete an outgoing account.

--lld <driver> --mode lld --op start

Start the specified lld without restarting the tgtd process.

--control-port <port> use control port <port>

--help

display this help and exit

**BIBLIOGRAFIA:**

https://community.mellanox.com/s/article/howto-configure-tgt-enabled-with-iser-transport-for-rhel

https://www.hiroom2.com/2017/07/12/centos-7-scsi-target-utils-en/

https://www.thegeekdiary.com/how-to-configure-iscsi-initiator-client-in-centos-rhel-7/

https://docs.oracle.com/en/operating-systems/oracle-linux/7/stordev/chap-iscsi.html#ol7-s17-storage

https://kb.wisc.edu/15607

https://fedoraproject.org/wiki/Scsi-target-utils_Quickstart_Guide

https://www.server-world.info/en/note?os=Fedora_31&p=iscsi&f=3

https://www.server-world.info/en/note?os=CentOS_8&p=iscsi&f=1

http://whatis.techtarget.com/glossary/Storage-area-network-SAN

https://www.programmersought.com/article/48433291777/

https://www.youtube.com/watch?v=q-ncOSGtMkQ

https://ccia.esei.uvigo.es/docencia/CDA/1819/practicas/ejercicio-iscsi/

https://es.linux-console.net/?p=385

https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/configuring_device_mapper_multipath/index

http://linux-iscsi.org/wiki/ISCSI