Sanjeevi
sanjeevim@yahoo.com
+91-9611131569

Fundamental of iSCSI Protocol

What is iSCSI?

iSCSI stands for "**Internet Small Computer System Interface**". It is a TCP/IP based protocol for emulating a SCSI high-performance local storage bus over IP networks, providing data transfer and management to remote block storage devices.

How iSCSI works?

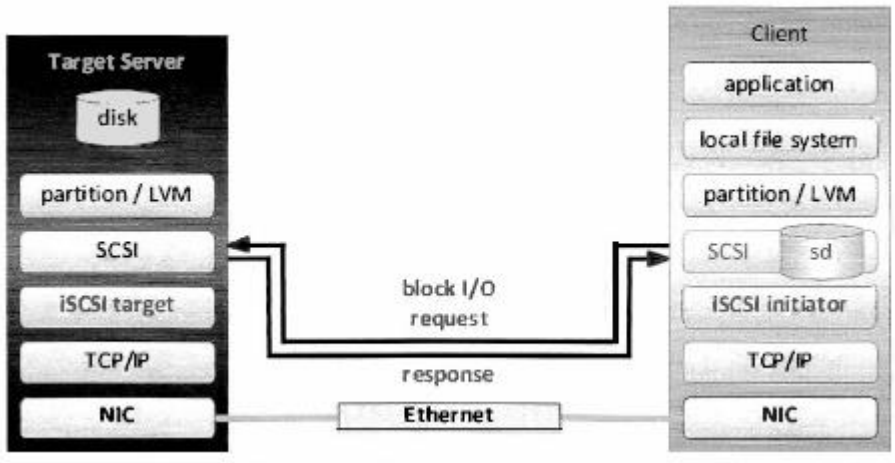
iSCSI works by transporting block-level data between an iSCSI initiator on a server and an iSCSI target on a storage device.

The iSCSI protocol encapsulates SCSI commands and assembles the data in packets for the TCP/IP layer. Packets are sent over the network using a point-to-point connection.

iSCSI Fundamentals:

The iSCSI protocol works in a similar fashion like client-server configuration. Client systems configure initiator software to send SCSI commands to remote server storage targets. Accessed iSCSI targets appear on the client system as local, unformatted SCSI block devices, identical to devices connected with SCSI cabling, FC direct attached, or FC switched fabric.

- iSCSI uses ACLs to perform LUN masking, managing the accessibility of appropriate targets and LUNs to initiators. Access to target may also be limited with CHAP authentication. iSCSI ACLs are similar to FC's use of devices World Wide Numbers (WWNs) for soft zoning management restrictions. Although FC switch-level compulsory port restriction (hard zoning) has no comparable iSCSI mechanism, Ethernet VLANs could provide similar isolation security.
- Unlike local block devices, iSCSI network-accessed block devices are discoverable from many remote initiators. Typical local file systems (e.g., ext4, XFS, btrfs) do not support concurrent multi system mounting, which can result in significant file system corruption. Clustered systems resolve multiple system access by use of the Global File System (GFS2), designed to provide distributed file locking and concurrent multinode file system mounting.
- An attached iSCSI block device appears as a local SCSI block device (sdx) for use underneath a local file system, swap space or a raw database installation as shown below



Target Setup

Red Hat Enterprise Linux 7 uses the targetcli shell as a front end for viewing, editing, and saving the configuration of the Linux-IO Target without the need to manipulate the kernel target's configuration files directly. The targetcli tool is a command-line interface that allows an administrator to export local storage resources, which are backed by either files, volumes, local SCSI

Sanjeevi
sanjeevim@yahoo.com
+91-9611131569

devices, or RAM disks, to remote systems. The targetcli tool has a tree-based layout, includes built-in tab completion, and provides full auto-complete support and inline documentation.
The hierarchy of targetcli does not always match the kernel interface exactly because targetcli is simplified where possible.
To ensure that the changes made in `targetcli` are persistent, start and enable the target service:

```
# systemctl start target
# systemctl enable target
```

Installing and Running targetcli

To install targetcli, use:

```
# yum install targetcli
```

Start the target service:

```
# systemctl start target
```

Configure target to start at boot time:

```
# systemctl enable target
```

Open port `3260` in the firewall and reload the firewall configuration:

```
# firewall-cmd --permanent --add-port=3260/tcp
Success
# firewall-cmd --reload
Success
```

Use the `targetcli` command, and then use the `ls` command for the layout of the tree interface:

```
# targetcli
:
/> ls
o- /.....................................[...]
  o- backstores..........................[...]
  | o- block..............[Storage Objects: 0]
  | o- fileio.............[Storage Objects: 0]
  | o- pscsi..............[Storage Objects: 0]
  | o- ramdisk............[Storage Ojbects: 0]
  o- iscsi........................[Targets: 0]
  o- loopback.....................[Targets: 0]
```

Note

In Red Hat Enterprise Linux 7.0, using the targetcli command from Bash, for example, targetcli iscsi/ create, does not work and does not return an error. Starting with Red Hat

Creating a Backstore

Backstores enable support for different methods of storing an exported LUN's data on the local machine. Creating a storage object defines the resources the backstore uses.

Note

In Red Hat Enterprise Linux 6, the term 'backing-store' is used to refer to the mappings created. However, to avoid confusion between the various ways 'backstores' can be used, in Red Hat Enterprise Linux 7 the term 'storage objects' refers to the mappings created and 'backstores' is used to describe the different types of backing devices.
The backstore devices that LIO supports are:

Sanjeevi
sanjeevim@yahoo.com
+91-9611131569

FILEIO (Linux file-backed storage)

FILEIO storage objects can support either write_back or write_thru operation. The write_back enables the local file system cache. This improves performance but increases the risk of data loss. It is recommended to use write_back=false to disable write_back in favor of write_thru.
To create a fileio storage object, run the command /backstores/fileio create *file_name file_location file_size* write_back=false. For example:

```
/> /backstores/fileio create file1 /tmp/disk1.img 200M write_back=false
Created fileio file1 with size 209715200
```

BLOCK (Linux BLOCK devices)

The block driver allows the use of any block device that appears in the /sys/block to be used with LIO. This includes physical devices (for example, HDDs, SSDs, CDs, DVDs) and logical devices (for example, software or hardware RAID volumes, or LVM volumes).
Note
BLOCK backstores usually provide the best performance.
To create a BLOCK backstore using any block device, use the following command:

Sanjeevi
sanjeevim@yahoo.com
+91-9611131569

```
# fdisk /dev/vdb
Welcome to fdisk (util-linux 2.23.2).

Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table
Building a new DOS disklabel with disk identifier 0x39dc48fb.

Command (m for help): n
Partition type:
   p   primary (0 primary, 0 extended, 4 free)
   e   extended
Select (default p): *Enter*
Using default response p
Partition number (1-4, default 1): *Enter*
First sector (2048-2097151, default 2048): *Enter*
Using default value 2048
Last sector, +sectors or +size{K,M,G} (2048-2097151, default 2097151): +250M
Partition 1 of type Linux and of size 250 MiB is set

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
```

Note
You can also create a BLOCK backstore on a logical volume.

Creating an iSCSI target

1. Run `targetcli`.
2. Move into the iSCSI configuration path:

```
/> iscsi/
```

Note

The cd command is also accepted to change directories, as well as simply listing the path to move into.
3.Create an iSCSI target using a default target name.

```
/iscsi> create
Created target
iqn.2003-01.org.linux-iscsi.hostname.x8664:sn.78b473f296ff
Created TPG1
```

Sanjeevi
sanjeevim@yahoo.com
+91-9611131569

Or create an iSCSI target using a specified name.

```
/iscsi > create iqn.2006-04.com.example:444
Created target iqn.2006-04.com.example:444
Created TPG1
```

4. Verify that the newly created target is visible when targets are listed with ls.

```
/iscsi > ls
o- iscsi.....................................[1 Target]
    o- iqn.2006-04.com.example:444.............[1 TPG]
        o- tpg1.........................[enabled, auth]
            o- acls.............................[0 ACL]
            o- luns.............................[0 LUN]
            o- portals.....................[0 Portal]
```

Note

As of Red Hat Enterprise Linux 7.1, whenever a target is created, a default portal is also created.

Configuring an iSCSI Portal

To configure an iSCSI portal, an iSCSI target must first be created and associated with a TPG.

Note

As of Red Hat Enterprise Linux 7.1 when an iSCSI target is created, a default portal is created as well. This portal is set to listen on all IP addresses with the default port number (that is, 0.0.0.0:3260). To remove this and add only specified portals, use /iscsi/iqn-name/tpg1/portals delete ip_address=0.0.0.0 ip_port=3260 then create a new portal with the required information.

Creating an iSCSI Portal

1. Move into the TPG.

   ```
   /iscsi> iqn.2006-04.example:444/tpg1/
   ```

2. There are two ways to create a portal: create a default portal, or create a portal specifying what IP address to listen to.

   Creating a default portal uses the default iSCSI port 3260 and allows the target to listen on all IP addresses on that port.

```
/iscsi/iqn.20...mple:444/tpg1> portals/ create
Using default IP port 3260
Binding to INADDR_Any (0.0.0.0)
Created network portal 0.0.0.0:3260
```

To create a portal specifying what IP address to listen to, use the following command.

Sanjeevi
sanjeevim@yahoo.com
+91-9611131569

```
/iscsi/iqn.20...mple:444/tpg1> portals/ create 192.168.122.137
Using default IP port 3260
Created network portal 192.168.122.137:3260
```

3.Verify that the newly created portal is visible with the `ls` command.

```
/iscsi/iqn.20...mple:444/tpg1> ls
o- tpg.................................. [enambled, auth]
    o- acls ....................................[0 ACL]
    o- luns ....................................[0 LUN]
    o- portals .................................[1 Portal]
        o- 192.168.122.137:3260.....................[OK]
```

Configuring LUNs

1. Create LUNs of already created storage objects.

```
/iscsi/iqn.20...mple:444/tpg1> luns/ create /backstores/ramdisk/rd_backend
Created LUN 0.

/iscsi/iqn.20...mple:444/tpg1> luns/ create /backstores/block/block_backend
Created LUN 1.

/iscsi/iqn.20...mple:444/tpg1> luns/ create /backstores/fileio/file1
Created LUN 2.
```

2. Show the changes.

```
/iscsi/iqn.20...mple:444/tpg1> ls
o- tpg.................................. [enambled, auth]
    o- acls ....................................[0 ACL]
    o- luns ....................................[3 LUNs]
    |  o- lun0.........................[ramdisk/ramdisk1]
    |  o- lun1.................[block/block1 (/dev/vdb1)]
    |  o- lun2...................[fileio/file1 (/foo.img)]
    o- portals .................................[1 Portal]
        o- 192.168.122.137:3260.....................[OK]
```

Note

Be aware that the default LUN name starts at 0, as opposed to 1 as was the case when using `tgtd` in Red Hat Enterprise Linux 6.

Configuring ACLs

Sanjeevi
sanjeevim@yahoo.com
+91-9611131569

1. Move into the acls directory.

```
/iscsi/iqn.20...mple:444/tpg1> acls/
```

2. Create an ACL. Either use the initiator name found in `/etc/iscsi/initiatorname.iscsi` on the initiator, or if using a name that is easier to remember

For example:

```
/iscsi/iqn.20...444/tpg1/acls> create iqn.2006-04.com.example.foo:888
Created Node ACL for iqn.2006-04.com.example.foo:888
Created mapped LUN 2.
Created mapped LUN 1.
Created mapped LUN 0.
```

Note

The given example's behavior depends on the setting used. In this case, the global setting `auto_add_mapped_luns` is used. This automatically maps LUNs to any created ACL.

You can set user-created ACLs within the TPG node on the target server:

```
/iscsi/iqn.20...scsi:444/tpg1> set attribute generate_node_acls=1
```

3. Show the changes.

```
/iscsi/iqn.20...444/tpg1/acls> ls
o- acls ..................................................[1 ACL]
    o- iqn.2006-04.com.example.foo:888 ....[3 Mapped LUNs, auth]
        o- mapped_lun0 .............[lun0 ramdisk/ramdisk1 (rw)]
        o- mapped_lun1 ..................[lun1 block/block1 (rw)]
        o- mapped_lun2 ..................[lun2 fileio/file1 (rw)]
```

Removing Objects with `targetcli`

To remove an backstore use the command:

```
/> /backstores/backstore-type/backstore-name
```

To remove parts of an iSCSI target, such as an ACL, use the following command:

```
/> /iscsi/iqn-name/tpg/acls/ delete iqn-name
```

To remove the entire target, including all ACLs, LUNs, and portals, use the following command:

```
/> /iscsi delete iqn-name
```

Creating an iSCSI Initiator

Sanjeevi
sanjeevim@yahoo.com
+91-9611131569

1. Install iscsi-initiator-utils:

```
# yum install iscsi-initiator-utils -y
```

2. modify the /etc/iscsi/initiatorname.iscsi file accordingly. For example:

```
# cat /etc/iscsi/initiatorname.iscsi

InitiatorName=iqn.2006-04.com.example.node1

# iscsiadm -m discovery -t st -p target-ip-address

10.64.24.179:3260,1 iqn.2006-04.com.example:3260
```

3. Log in to the target with the target IQN you discovered in above step

```
# iscsiadm -m node -T iqn.2006-04.com.example:3260 -l

Logging in to [iface: default, target: iqn.2006-04.com.example:3260, portal:
10.64.24.179,3260] (multiple)

Login to [iface: default, target: iqn.2006-04.com.example:3260, portal:
10.64.24.179,3260] successful.
```

This procedure can be followed for any number of initiators connected to the same LUN so long as their specific initiator names are added to the ACL as described in

4. Find the iSCSI disk name and create a file system on this iSCSI disk:

```
# grep "Attached SCSI" /var/log/messages

# mkfs.ext4 /dev/disk_name
```

Replace *disk_name* with the iSCSI disk name displayed in /var/log/messages.

5. Mount the file system:

```
# mkdir /mount/point

# mount /dev/disk_name /mount/point
```

Sanjeevi
sanjeevim@yahoo.com
+91-9611131569

Replace /mount/point with the mount point of the partition.

6. Edit the /etc/fstab to mount the file system automatically when the system boots:

```
# vim /etc/fstab


/dev/disk_name /mount/point ext4 _netdev 0 0
```

Replace disk_name with the iSCSI disk name.

7. Log off from the target:

```
# iscsiadm -m node -T iqn.2006-04.com.example:3260 -u
```

Note:

We must use **_netdev** parameter while adding entry in **/etc/fstab,** Which is used to prevent the system from attempting to mount the file systems until the network has been enabled on the system.