ISCSI Target

With **iSCSI** you can access storage over an IP-based network.

The exported storage entity is the **target** and the importing entity is the **initiator**. There are different modules available to set up the target:

Related articles

iSCSI Initiator

iSCSI Boot

- The **SCSI Target Framework (STGT/TGT) (http://stgt.berlios.de/)** was the standard before linux 2.6.38.
- The current standard is the **LIO target (http://linux-iscsi.org/)**.
- The **iSCSI Enterprise Target (IET) (http://iscsitarget.sourceforge.net/)** is an old implementation and **SCST Target Subsystem (SCST) (http://scst.sourceforge.net/)** is the successor of IET and was a possible candidate for kernel inclusion before the decision fell for LIO.

Contents

Setup with LIO Target

LIO target is included in the kernel since 2.6.38. However, the iSCSI target fabric is included since linux 3.1.

The important kernel modules are *target_core_mod* and *iscsi_target_mod*, which should be in the kernel and loaded automatically.

It is highly recommended to use the free branch versions of the packages: **targetcli-fb (https://aur.archlinux.org/packages/targetcli-fb/)**[AUR], **python-rtslib-fb (https://aur.archlinux.org/packages/python-rtslib-fb/)**[AUR] and **python-configshell-fb (https://aur.archlinux.org/packages/python-configshell-fb/)**[AUR]. The original **targetcli (https://aur.archlinux.org/packages/targetcli/)**[AUR][**broken link**: archived in **aur-mirror (https://github.com/felixonmars/aur3-mirror/tree/master/targetcli)**] is also available but has a different way of saving the configuration using the deprecated *lio-utils* and depends on *epydoc*.

A systemd `target.service` is included in **python-rtslib-fb (https://aur.archlinux.org/packages/python-rtslib-fb/)**[AUR] when you use the free branch and a `/etc/rc.d/target` in **lio-utils (https://aur.archlinux.org/packages/lio-utils/)**[AUR][**broken link**: archived in **aur-mirror (https://github.com/felixonmars/aur3-mirror/tree/master/lio-utils)**] when you use the original *targetcli* or *lio-utils* directly.

You start LIO target with

```
# systemctl start target
```

This will load necessary modules, mount the configfs and load previously saved iscsi target configuration.

With

```
# targetcli status
```

you can show some information about the running configuration (only with the free branch). You might want to enable the lio target on boot with

```
# systemctl enable target
```

You can use **targetcli** to create the whole configuration or you can alternatively use the **lio utils** tcm_* and lio_* directly (deprecated).

Using targetcli

The external manual is only available in the *free branch*. **targetd (https://github.com/agrover/targetd)** is not in AUR yet, but this depends on the free branch.

The config shell creates most names and numbers for you automatically, but you can also provide your own settings. At any point in the shell you can type `help` in order to see what commands you can issue here.

Tip: You can use tab-completion in this shell

Tip: You can type `cd` in this shell to view & select paths

After starting the target (see above) you enter the configuration shell with

```
# targetcli
```

In this shell you include a block device (here: `/dev/disk/by-id/md-name-nas:iscsi`) to use with

```
/> cd backstores/block
/backstores/block> create md_block0 /dev/disk/by-id/md-name-nas:iscsi
```

Note: You can use any block device, also raid and lvm devices. You can also use files when you go to fileio instead of block.

You then create an iSCSI Qualified Name (iqn) and a target portal group (tpg) with

```
...> cd /iscsi
/iscsi> create
```

Note: With appending an iqn of your choice to `create` you can keep targetcli from automatically creating an iqn

In order to tell LIO that your block device should get used as *backstore* for the target you issue

Note: Remember that you can type `cd` to select the path of your <iqn>/tpg1

```
.../tpg1> cd luns
.../tpg1/luns> create /backstores/block/md_block0
```

Then you need to create a *portal*, making a daemon listen for incoming connections:

```
.../luns/lun0> cd ../../portals
.../portals> create
```

Targetcli will tell you the IP and port where LIO is listening for incoming connections (defaults to 0.0.0.0 (all)). You will need at least the IP for the clients. The port should be the standard port 3260.

In order for a client/**initiator** to connect you need to include the iqn of the initiator in the target configuration:

```
...> cd ../../acls
.../acls> create iqn.2005-03.org.open-iscsi:SERIAL
```

Instead of `iqn.2005-03.org.open-iscsi:SERIAL` you use the iqn of an initiator. It can normally be found in `/etc/iscsi/initiatorname.iscsi`. You have to do this for every initiator that needs to connect. Targetcli will automatically map the created lun to the newly created acl.

Note: You can change the mapped luns and whether the access should be rw or ro. See `help create` at this point in the targetcli shell.

The last thing you have to do in targetcli when everything works is saving the configuration with:

```
...> cd /
/> saveconfig
```

The will the configuration in `/etc/target/saveconfig.json`. You can now safely start and stop `target.service` without losing your configuration.

Tip: You can give a filename as a parameter to `saveconfig` and also clear a configuration with `clearconfig`

Authentication

Authentication per CHAP is enabled per default for your targets. You can either setup passwords or disable this authentication.

Disable Authentication

Navigate targetcli to your target (i.e. /iscsi/iqn.../tpg1) and

```
.../tpg1> set attribute authentication=0
```

Warning: With this setting everybody that knows the iqn of one of your clients (initiators) can access the target. This is for testing or home purposes only.

Set Credentials

Navigate to a certain acl of your target (i.e. /iscsi/iqn.../tpg1/acls/iqn.../) and

```
...> get auth
```

will show you the current authentication credentials.

```
...> set auth userid=<username in target>
...> set auth password=<password in target>
...> set auth mutual_userid=<username in initiator>  (optional)
...> set auth mutual_password=<password in initiator>  (optional)
```

The first two fields are the username and password of the target. The initiator will use this to log into the target. The last two fields (prefixed with "mutual_") are the username and password of the initiators (note that all initiators will have the same username and password). These two are optional parameters and it ensures that initiators will only accept connections from permitted targets.

Using (plain) LIO utils

You have to install **lio-utils (https://aur.archlinux.org/packages/lio-utils/)**AUR[**broken link**: archived in **aur-mirror (https://github.com/felixonmars/aur3-mirror/tree/master/lio-utils)**] from **AUR** and the dependencies (python2).

Tips & Tricks

- With `targetcli sessions` you can list the current open sessions. This command is included in the **targetcli-fb (https://aur.archlinux.org/packages/targetcli-fb/)**[AUR] package, but not in *lio-utils* or the original *targetcli*.

Upstream Documentation

- **targetcli (http://www.linux-iscsi.org/wiki/Targetcli)**
- **LIO utils (http://www.linux-iscsi.org/wiki/Lio-utils_HOWTO)**
- You can also use `man targetcli` when you installed the *free branch* version **targetcli-fb (https://aur.archlinux.org/packages/targetcli-fb/)**[AUR].

Setup with SCSI Target Framework (STGT/TGT)

You will need the Package **tgt (https://aur.archlinux.org/packages/tgt/)**[AUR] from **AUR**.

See: **TGT iSCSI Target**

Setup with iSCSI Enterprise Target (IET)

You will need **iscsitarget-kernel (https://aur.archlinux.org/packages/iscsitarget-kernel/)**AUR[**broken link**: archived in **aur-mirror (https://github.com/felixonmars/aur3-mirror/tree/master/iscsitarget-kernel)**] and **iscsitarget-usr (https://aur.archlinux.org/packages/iscsitarget-usr/)**AUR[**broken link**: archived in **aur-mirror (https://github.com/felixonmars/aur3-mirror/tree/master/iscsitarget-usr)**] from **AUR**.

Create the Target

Modify /etc/iet/ietd.conf accordingly

Hard Drive Target

```
Target iqn.2010-06.ServerName:desc
Lun 0 Path=/dev/sdX,Type=blockio
```

File based Target

Use "dd" to create a file of the required size, this example is 10GB.

```
dd if=/dev/zero of=/root/os.img bs=1G count=10
```

```
Target iqn.2010-06.ServerName:desc
Lun 0 Path=/root/os.img,Type=fileio
```

Start server services

```
rc.d start iscsi-target
```

Also you can "iscsi-target" to DAEMONS in /etc/rc.conf so that it starts up during boot.

See also

- **iSCSI Boot** Booting Arch Linux with / on an iSCSI target.
- **Persistent block device naming** in order to use the correct block device for a target