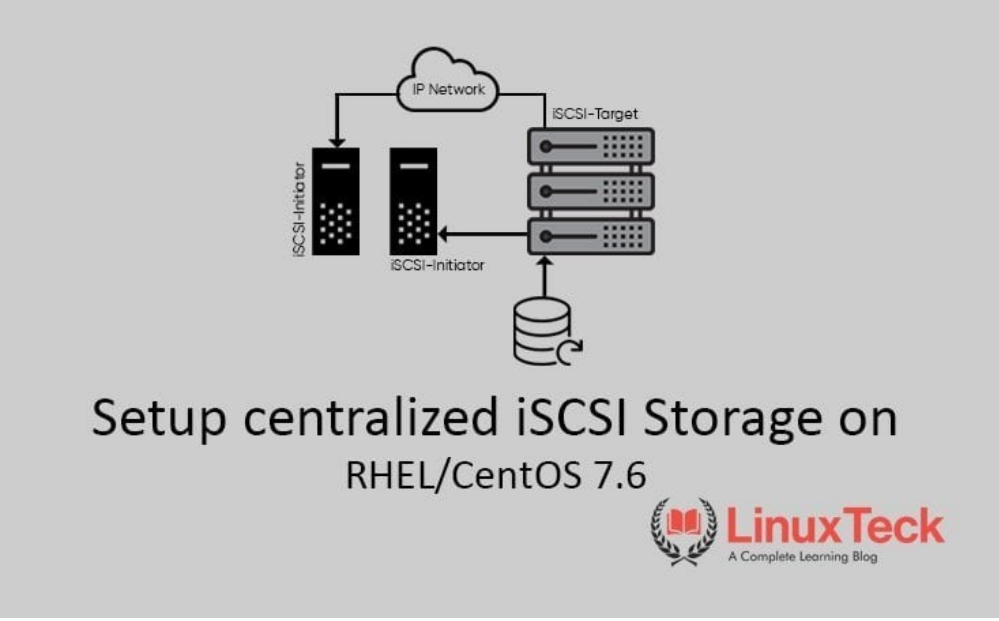
LinuxTeck
A Complete Learning Blog

RHEL-Centos-7

How to configure iSCSI target & initiator on RHEL/CentOS 7.6

Last updated on June 8th, 2021 - by LinuxTeck - Leave a Comment



This article will help you learn how to setup/configure iSCSI on Linux/Unix based systems. iSCSI stands for (Internet Small Computer System Interface), which is

an Industry standard protocol which is mainly used to share the storage device over the TCP/IP layer. Unlike Samba or NFS, which work at the file system level whereas iSCSI works only on the block-level device. Most block-level storage devices have the capability of built-in work to share data across volume.

Advantages of using block-level storage?

Lowest possible latency
High performance IOPS
Highly redundant

iSCSI handles client-server architecture. It uses iSCSI components to communicate with each other. For Client, it uses "initiators" and for Server, it uses “targets”.

iSCSI Target: This is also known as the iSCSI Server which is responsible for exporting disk/block devices to the iSCSI initiator/client. We can create multiple targets in the iSCSI Server and each target has its own unique ID. We can also attach one or more LUNs for each target. Each LUN has its back device attached to it and that device is the actual file that is exported and will be visible to the iSCSI initiator. It could be either a disk or a disk partition, LVM, Raid or a file. It uses the iscsi protocol to interact with the clients using the IP network. It provides security features like Authentication, Network, etc.

iSCSI Initiator: It is also known as iSCSI Client which is responsible for importing the LUNs which are exported by the iSCSI target/server. Once the client application is connected to the server, then it will send the ISCSI commands to the iscsi server over the IP network.

This step-by-step guide will help you on how to configure the iSCSI target and initiator on RHEL/CentOS 7.6 Linux. This guide is fair enough to work for all the versions of RHEL/CentOS/Fedora with a few minimal changes in commands.

Table of Contents [show]

Prerequisites :

Operating System: CentOS Linux 7
IP Address : For Server and For Client
Package : targetcli (Server) and iscsi-initiator-utils (Client)
Port : 3260
Block device : /dev/sdb or /dev/sdc ..

My Lab Setup :

For the lab setup, I am using 2 centos machines. One for iSCSI Server/ Target and the other for iSCSI initiator /Client

iSCSI-Server:-

Operating System: CentOS Linux 7 (Core)
hostname: iscsi-server.local
IP Address: 192.168.3.101
Block device : /dev/sdb

iSCSI-Client:-

Operating System: CentOS Linux 7 (Core)
hostname: iscsi-client.local
IP Address: 192.168.3.102

Follow the steps to configure/setup iSCSI Target on RHEL / Centos-7.

Before starting the process, we need to confirm the available disk to use as an iSCSI Target Server.

lsblk or # sfdisk -s (Use either one of the command)

Output:

```
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda 8:0 0 20G 0 disk
├─sda1 8:1 0 500M 0 part /boot
├─sda2 8:2 0 3.9G 0 part [SWAP]
└─sda3 8:3 0 15.6G 0 part /
sdb 8:16 0 1G 0 disk <-------
sr0 11:0 1 4.1G 0 rom /run/media/john/CentOS 7 x86_64
```

Note:

The above command will list the details of all available or particular block devices in a tree format. The output shows that now the iSCSI Target Server has 2 block-devices "/dev/sda and /dev/sdb". The /dev/sda is already being used for server OS and other applications and /dev/sdb is available to use as shared storage for iscsi.

Step1: Setup iSCSI Target/Server

First, let's update the latest current version of the package.

```
# yum update -y
```

Install the target utility package using the following command:

```
# yum install -y targetcli
```

Once you have successfully installed the package, then enter the following command to get the interactive shell of the iSCSI target. There you can edit, list and save the configuration file. It is a ".jason" format. The targetcli layout of the filesystems is in a hierarchical structure.

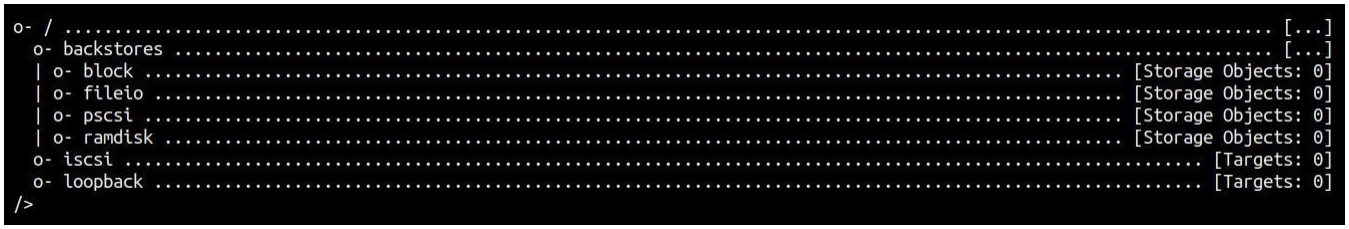
```
# targetcli

targetcli shell version 2.1.fb49
Copyright 2011-2013 by Datera, Inc and others.
For help on commands, type 'help'.
/>
```

Note:

In the above interactive shell you can hit the 'help' command to list all the available options. Now we use the 'ls' command to list out the available configuration objects.

```
/> ls
```



In the above output, you can see there are 3 options under '/' which are "backstores, iscsi, and loopback". Under "backstores" there are 4 sub-options which are "block, fileio, pscsi, and ramdisk".

A short explanation is given below of the above "options and sub-options":

backstores It is a kind of local storage or a partition that you need to access as storage object which defines the backstore use. It could be either a block device or LVM or a file.

- **block** It is a kind of local storage or a partition that you need to access as storage object which defines the backstore use. It could be either a block device or LVM or a file.
- **fileio** It is a kind of local storage or a partition that you need to access as storage object which defines the backstore

		use. It could be either a block device or LVM or a file.
	pscsi	It is a kind of local storage or a partition that you need to access as storage object which defines the backstore use. It could be either a block device or LVM or a file.
	ramdisk	It is a kind of local storage or a partition that you need to access as storage object which defines the backstore use. It could be either a block device or LVM or a file.
iscsi		It is mainly used for all the configurations like who can access this LUN and how many LUNs can be attached and their ACLs (Access Control List).
loopback		It is nothing but a fabric module..

Note:

In backstores we generally only use two sub-options i.e., block or fileio. The rest is not so frequent. For our LAB exercise, I will use the first two options only.

In the interactive shell, you can use the TAB completion to fill in partially typed commands or object path.

Let's start with the configuration part:

(a) Create a backstore block device:

Option 1:

```
/> /backstores/block create ltecklun1 /dev/sdb
```

Output:
Created block storage object ltecklun1 using /dev/sdb.

--------> OR <---------

Option 2:

/> backstores/fileio/ create shareddata /opt/ltecklun1.img 1024M

Output:
Created fileio shareddata with size 1073741824

Note:

Using the 1st option, we can create a block type backing-store for the storage-object named as "ltecklun1" with an attached block device "/dev/sdb". Use the 'ls' command to see the list below:

/> ls

Note:

As the above output shows, you have created a LUN named "ltecklun1" with the backed device of "/dev/sdb" in a hierarchy format.

(b) Create iSCSI for IQN target:

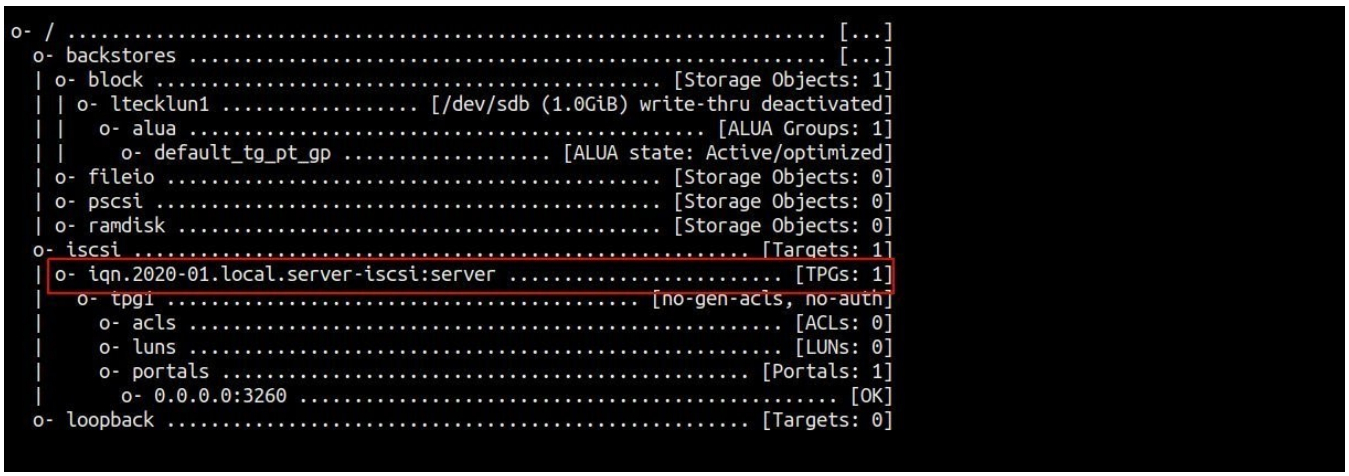
/> /iscsi create iqn.2020-01.local.server-iscsi:server

Note:

Using the above command we can create a custom made 'iqn' for the iSCSI target. Alternatively, we can just execute "create command" i.e. (/> /iscsi create) without any arguments and it will create a default iSCSI iqn and target name. Use the 'ls' command

below to see the list:

/> ls



Note:

As the above output shows, 3 objects have been defined under TPG1 which are "acls, luns, and portals". Under the portals, you can see the IPv4 interface with TCP port of '0.0.0.0:3260' as created automatically.

From RHEL 7.2 version onwards the default portal configuration is done automatically, whereas the previous version of RHELs like "7.0 and 7.1" we need to manually configure the default portal settings to listen to the TCP port of "3260 on 0.0.0.0" after the target configuration has been set.

Here is a short explanation of iSCSI component terminology:":

IQN	(iSCSI Qualified Name: Worldwide unique identification name for both targets and initiators. The naming format of IQN is "iqn.yyyy-mm.naming-authority:unique-name").
yyyy-mm	year and month. You can use either the current date or establishment date of your naming authority.

naming-authority	Meaning your organization authority name/internet domain name. The syntax is used in the form of reverse order eg: com.domainname.
unique-name	we can use any name, it is just for identification.
ACLs	(Access Control Lists: to provide access permissions using iQN).
LUNs	(Logical Unit Number: it is a block device to attach. One or more LUNs can be attached in a single target).
Portals	(Mainly used to configure the IP address and TCP port on either a target of initiator to establish the connections).

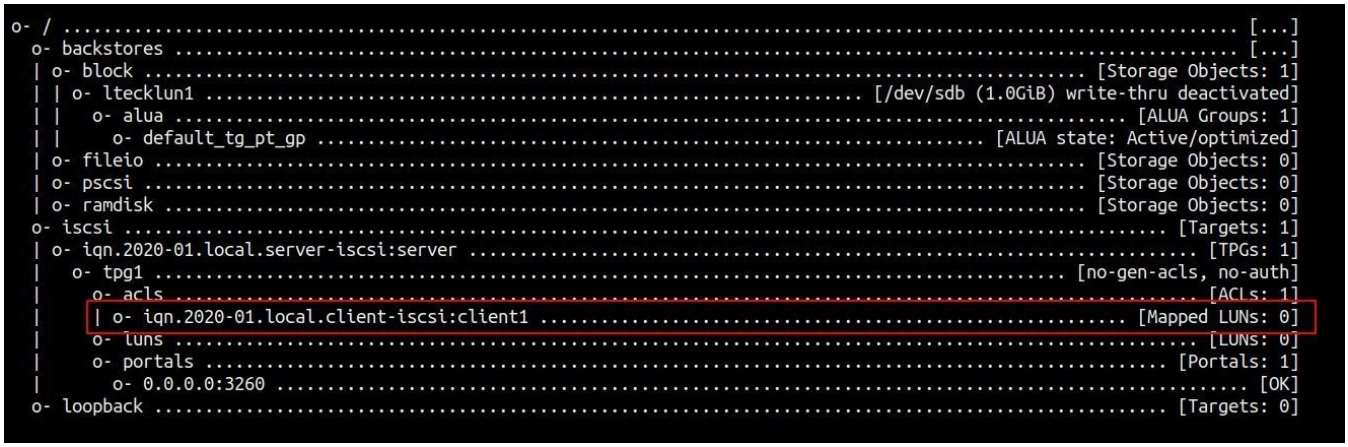
(c) Create ACLs:

```
/>  /iscsi/iqn.2020-01.local.server-iscsi:server/tpg1/acls  create
iqn.2020-01.local.client-iscsi:client1
```

Note:

The above command would permit the client to access the LUN which is based on the client's iQN. Don't forget to use the same iQN name to configure the initiator/client name. If there is any mismatch of the iQN, then the client won't able to communicate to the Server. Use the 'ls' command to see the list below:

```
/> ls
```



(d) Create LUNs under the ISCSI target:

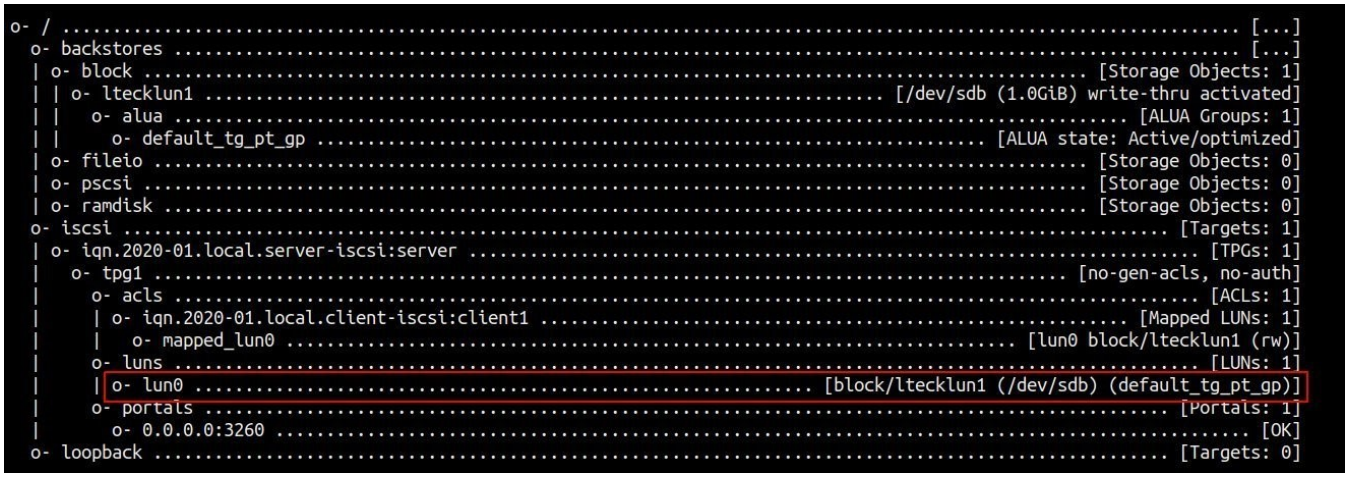
/> /iscsi/iqn.2020-01.local.server-iscsi:server/tpg1/luns create /backstores /block/ltecklun1

Output:
Created LUN 0.
Created LUN 0->0 mapping in node ACL iqn.2020-01.local.client-iscsi:client1

Note:

Using the above command we can associate the block device with a particular Target Portal Group (TPG). In our case, we will be using the backing storage object which has been created above "/backstores/block/ltecklun1" to present to the LUN directory under TPG1. Use the 'ls' command below to see the list:

/> ls



(d) Two optional choices:

Note:

CHAP Authentication and use the specific IP address on the Portal. It is completely at the discretion of your choice. In this article, we will use both optional services as additional security.

Option 1: CHAPP Authentication

This is an additional layer of security to exchange communication between the server and the client. It is widely supported authentication terminology to exchange passwords between Server-Client communications. Follow the steps to configure :

⇒ Change the directory to tpg1

```
/> cd /iscsi/iqn.2020-01.local.server-iscsi:server/tpg1
```

⇒ Change the value of attribute authentication to '1', meaning it will disable all the corresponding endpoints, else it will grant open access to all the initiators.

```
/iscsi/iqn.20...i:server/tpg1> set attribute authentication=1
```

⇒ To cross-check the attribute auth status.

```
/iscsi/iqn.20...i:server/tpg1> get attribute authentication

Output:
authentication=1
```

⇒ Change the directory to iSCSI Node ACL

```
/iscsi/iqn.20...i:server/tpg1> cd acls/iqn.2020-01.local.client-iscsi:client1
```

⇒ Execute user-id command to setup for CHAP authentication for iscsi initiator

```
/iscsi/iqn.20...iscsi:client1> set auth userid=linuxteck
```

⇒ Execute the command to create a password for the above user-id

```
/iscsi/iqn.20...iscsi:client1> set auth password=password@123
```

⇒ To get verify the above-executed credentials

```
/iscsi/iqn.20...iscsi:client1> get auth

Output:
AUTH CONFIG GROUP
==================
mutual_password=
----------------
The mutual_password auth parameter.
mutual_userid=
--------------
The mutual_userid auth parameter.
password=password@123
---------------------
```

The password auth parameter.
userid=linuxteck

The user-id auth parameter.

Note:

After verifying the above credentials, use 'ls' command to confirm whether this LUN is marked as "1-way auth" with the following output:

```
/iscsi/iqn.20...iscsi:client1> ls
o- iqn.2020-01.local.client-iscsi:client1 ..... [1-way auth, Mapped LUNs: 1]
o- mapped_lun0 ..... [lun0 block/ltecklun1 (rw)]
```

⇒ Now move back to the root directory and execute saveconfig and exit

```
/iscsi/iqn.20...iscsi:client1> cd /

/> saveconfig

/> exit
```

⇒ Restart the target service and check the status

```
# systemctl restart target.service

# systemctl status target.service
```

Option 2: Create a specific IP address on Portal

```
/> cd /iscsi/iqn.2020-01.local.server-iscsi:server/tpg1/portals/
```

/iscsi/iqn.20.../tpg1/portals> create 192.168.3.102

Output:
Using default IP port 3260
Could not create NetworkPortal in configFS

WARNING:

You will be notified by the error mentioned above "Could not create NetworkPortal in configFS". The reason being, if any default portal (0.0.0.0:3260) exists, then it will not allow you to create any new portals, hence you need to delete the default/old portal and create a new portal using the following commands:

(i) Delete the default portal interface which is 0.0.0.0:3260

/> cd /iscsi/iqn.2020-01.local.server-iscsi:server/tpg1/portals/
/iscsi/iqn.20.../tpg1/portals> delete 0.0.0.0 ip_port=3260

Output:
Deleted network portal 0.0.0.0:3260

(2) Create a new portal with a specific IP address of iSCSI target

/iscsi/iqn.20.../tpg1/portals> create 192.168.3.101

Output:
Using default IP port 3260
Created network portal 192.168.3.102:3260.

(3) Use the 'ls' command to see the list below:

/iscsi/iqn.20.../tpg1/portals> ls

```
o- portals ..... [Portals: 1]
o- 192.168.3.102:3260 ..... [OK]
```

(4) Now go back to the root directory of targetcli shell 'cd /' and execute the 'saveconfig' command to save all the target configuration. This file will be saved in a ".json format" and exit.

```
/> saveconfig
Output:
Configuration saved to /etc/target/saveconfig.json

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

Use the following command to check whether the TCP port 3260 is open or not

```
# ss -na | grep 3260
tcp LISTEN 0 256 192.168.3.102:3260 *:*

# netstat -tnlp | grep 3260
tcp 0 0 192.168.3.102:3260 0.0.0.0:* LISTEN -
```

Note:

The 'ss' utility is similar to netstat, but it can print more information than other tools. It is confirmed now the port 3260 is opened with our given IP address.

Use the following command to open Firewall ports on iSCSI target (3260) services OR simply disable the Firewall.

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

# firewall-cmd --reload

# firewall-cmd --list-all
```

OR

```
# systemctl disable firewalld.service

# systemctl stop firewalld.service
```

Disable the SELinux or click here to configure SELinux for the iSCSI target.

Finally, start the iSCSI target and enable it for every reboot.

```
# systemctl enable target.service

# systemctl restart target.service

# systemctl status target.service
```

Step2: Setup iSCSI Initiator/Client

(a) First, let's update the latest current version of the package.

```
# yum update -y
```

(b) Install the iscsi-initiator package using the following command

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

Note:

Once you have successfully installed the package, then edit the iscsi-initiator configuration file "/etc/iscsi/initiatorname.iscsi" and add the name of the initiator. In our case, we have already created during the Target configuration, use the same iqn here (iqn.2020-01.local.client-iscsi:client1) and restart service.

```
# vi /etc/iscsi/initiatorname.iscsi

InitiatorName=iqn.2020-01.local.client-iscsi:client1
```

Save the file and restart the iscsid service.

```
# systemctl restart iscsid.service

# systemctl enable iscsid.service

# systemctl status iscsid.service
```

(c) Configure CHAP Authentication

Edit the "iscsid" configuration file and enabled/uncomment line number 57 for "CHAP" authmethod and uncomment following lines also 61 and 62 for ("node.session.auth.username = username and node.session.auth.password = password") in Centos-7.

```
# vi /etc/iscsi/iscsid.conf

node.session.auth.authmethod = CHAP
node.session.auth.username = linuxteck
node.session.auth.password = password@123
```

Save and Exit:

(d) Discover iSCSI Targets/Server (LUNs):

Follow the 'iscsiadm' command to perform discovery and login to iSCSI Targets. Basically, it comes with a combination of three different arguments which are "discover, portal, send target type".

(i) We need to discover the target by executing the following command. Here is the IP address that is for your target server:

```
# iscsiadm --mode discoverydb --type sendtargets --portal 192.168.3.101 --discover
```

Output:
192.168.3.101:3260,1 iqn.2020-01.local.server-iscsi:server

Note:

It will show you the available iQNs of the target server. Next, you can attempt to log-in. If you face any errors related to authentication, then you can check your CHAP settings for your iscsi initatior.

(ii) # iscsiadm -m node --login

Output:
Logging in to [iface: default, target: iqn.2020-01.local.server-iscsi:server, portal: 192.168.3.101,3260] (multiple)
Login to [iface: default, target: iqn.2020-01.local.server-iscsi:server, portal: 192.168.3.101,3260] successful.

Note:

Once you have logged in successfully without any error. Next, check the session and the node.

(iii) # iscsiadm -m session -o show
Output:
tcp: [1] 192.168.3.101:3260,1 iqn.2020-01.local.server-iscsi:server (non-flash)

iscsiadm --mode node -P 1

Output:
Target: iqn.2020-01.local.server-iscsi:server
Portal: 192.168.3.101:3260,1
Iface Name: default

(iv) Use the following command to check the newly added disk into iscsi initatior:

lsblk or # sfdisk -s

Output:
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda 8:0 0 20G 0 disk
├─sda1 8:1 0 500M 0 part /boot
├─sda2 8:2 0 3.9G 0 part [SWAP]
└─sda3 8:3 0 15.6G 0 part /
sdb 8:16 0 1G 0 disk <-------

Note:

The above output lists the newly added disk drive "sdb" with 1GB volume which is shared from the target server.

(e) Mount the Disk Drive (/dev/sdb)

(i) First, use the following command to create a filesystem on newly added iscsi device '/dev/sdb'.

mkfs.xfs /dev/sdb

```
Output:

      meta-data=/dev/sdb               isize=512    agcount=4, agsize=65536 blks
               =                       sectsz=512   attr=2, projid32bit=1
               =                       crc=1        finobt=0, sparse=0
      data     =                       bsize=4096   blocks=262144, 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
```

Note:

The "xfs" filesystem has been created successfully. ".xfs" format is a highly scalable and high-performance file system in centos/rhel7.

(ii) Create a new Mount Point for '/dev/sdb' block device

mount /dev/sdb /mnt

df -HT

```
Output:

      Filesystem     Type      Size  Used Avail Use% Mounted on
      /dev/sda3      xfs        17G  5.0G   12G  30% /
      devtmpfs       devtmpfs  942M     0  942M   0% /dev
      tmpfs          tmpfs     957M   87k  956M   1% /dev/shm
      tmpfs          tmpfs     957M  9.4M  947M   1% /run
      tmpfs          tmpfs     957M     0  957M   0% /sys/fs/cgroup
      /dev/sda1      xfs       521M  169M  352M  33% /boot
      tmpfs          tmpfs     192M   17k  192M   1% /run/user/42
      tmpfs          tmpfs     192M     0  192M   0% /run/user/0
      /dev/sdb       xfs       1.1G   34M  1.1G   4% /mnt
```

Note:

Currently the iscsi disk can be mounted on the client's machine, but the mount point will last only on a temporary basis, which means once you have rebooted the client machine the mount point will disappear. To avoid this issue, we need to make it an automount so that it will be mounted permanently for every reboot.

To do so, add an entry into '/etc/fstab', it is a system configuration file. it will handle all the mount points and related options.

(iii) Use the following command to create a new mount point and get the UUID of iSCSI disk

```
# mkdir /mnt/ltecklun1
# blkid /dev/sdb

Output:
/dev/sdb: UUID="f89efb1b-b02b-4d52-8d54-38567c61efc9" TYPE="xfs"
```

(iv) Now, add the above UUID entry into '/etc/fstab':

```
# vi /etc/fstab
UUID=f89efb1b-b02b-4d52-8d54-38567c61efc9 /mnt/ltecklun1 xfs defaults
0 0
```

save and exit

(v) Mount the iSCSI drive:

```
# mount /mnt/ltecklun1          OR          # mount -a          (It will
activate all the mount points)
```

(vi) check the mount points :

```
# df -HT
```

```
Output:

      Filesystem     Type      Size  Used Avail Use% Mounted on
       devtmpfs       devtmpfs  938M     0  938M   0% /dev
       tmpfs          tmpfs     954M     0  954M   0% /dev/shm
       tmpfs          tmpfs     954M   12M  943M   2% /run
       tmpfs          tmpfs     954M     0  954M   0% /sys/fs/cgroup
       /dev/sda3      xfs        17G  6.2G   11G  38% /
       /dev/sda1      xfs       521M  226M  296M  44% /boot
       tmpfs          tmpfs     191M  8.2k  191M   1% /run/user/42
       tmpfs          tmpfs     191M   33k  191M   1% /run/user/1000
       /dev/sr0       iso9660   4.4G  4.4G     0 100% /run/media/john/CentOS 7 x86_64
       tmpfs          tmpfs     191M     0  191M   0% /run/user/0
       /dev/sdb       xfs       1.1G   34M  1.1G   4% /mnt/ltecklun1
```

Note:

You can see above "/dev/sdb" is mounted now with the mount point of /mnt/ltecklun1.

(f) How to detach iSCSI storage

Use the following steps in case you want to remove the iSCSI storage from the client:

(i) # unmount /mnt/ltecklun1

(ii) # iscsiadm --mode node --targetname iqn.2020-01.local.server-iscsi:server --portal 192.168.3.101:3260 --logout

Output:
Logging out of session [sid: 1, target: iqn.2020-01.local.server-iscsi:server, portal: 192.168.3.101,3260]
Logout of [sid: 1, target: iqn.2020-01.local.server-iscsi:server, portal: 192.168.3.101,3260] successful.

Note:

You have now successfully logout from the client machine.

Congratulations, you have successfully configured an iSCSI target (server) and shared a block device with to an iSCSI initiator (client) on RHEL/CentOS 7.6. If

you have any difficulties in configuring the same, just let us know through the comment box.

I hope this article will help you to understand a few things about the 'iSCSI target/initiator'. Drop me your feedback/comments. If you like this article, kindly share it and it may help others as well.

Thank you!

RELATED POST

15 useful YUM commands for Beginners

June 08,2021

PREVIOUS ARTICLE
How to install Magento-2.2.3 on RHEL / CentOS 7.6

NEXT ARTICLE
How to configure Two Node High Availability Cluster On RHEL/CentOS 7.6

10 useful steps to install and secure SSH server in Linux

pgbpete September 25, 2020 at 11:00 pm

December 31,2021

Great article. Thank you for posting it. An unsolicited suggestion or two: 1) you may want to add

Thank you for your excellent suggestions and commands.

15 basic useful firewall-cmd commands in Linux
REPLY
June 08,2021

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment *

Name *

Email *

Website

Save my name, email, and website in this browser for the next time I comment.

POST COMMENT

TRENDING

How to configure Two Node High Availability Cluster On RHEL/CentOS 7.6

March 15, 2020

15 basic useful firewall-cmd commands in Linux

April 21, 2020

10 basic and most useful 'ssh' client commands in Linux

April 7, 2020

12 useful 'sed' commands in Linux

July 14, 2021

Popular | Comments | Tags

How to configure Two Node High Availability Cluster On RHEL/CentOS 7.6

Mar 15, 2020

15 basic useful firewall-cmd commands in Linux

Apr 20, 2020

First name or full name

Email

I want to receive a newsletters