Setting up an iSCSI SAN

iSCSI Initiator and Target

- Traditional SCSI uses a long cable and the SCSI **Command Descriptor Block** command (CDB) to attach to SCSI devices
- iSCSI uses the same **CDB** command, but encapsulated in IP packets over a network
- SCSI devices are emulated by using a storage backend and presenting it on the network using iSCSI targets
- Ethernet infrastructures can be as fast as FC structures, which makes iSCSI an enterprise ready choice to create a SAN

SAN - iSCSI target is a process, the bridge to the storage backend (disk, file, volume, partition).
External machine runs an iSCSI initiator, which does a discover, and can log in to the SAN
The external machine will then find these in /proc/partitions
Ths lsscsi tool isn't installed normally but you can go grab it

IQN qualified name to identify target and initiator
Initiator is client, target is the process on iscsi server
The target also runs ACLs to give access to specific node IQNs.
"Portal" is synonymous with node- ip address and port
Discovery is when initiators finds targets configured on a portal
LUN - for block devices shared through target
login - initiator authentication
TPG - Target Portal Group - collection of IP addresses and TCP ports a iscsi target will listen

Initiator will see added block devices as local (/dev/sdb etc)
If a LUN is exposed to multiple clients, they will all have access after connecting to the LUN
    Normally need clustering to provide access to device, to prevent simultaneous writes
    XFS and EXT aren't designed for that
    Shared file system like GFS2 is an alternative to clustering (is it really??)
    GFS2 shared cache between nodes (?)
    Neither GFS or clustering are in the RHCE

Setting up and provisioning an iSCSI Target

On the backend, two devices, which can be any physical volumes in LVM are going to be grouped into a VG, then split into LVsan1 and LVsan2, then their LUN's shared through the iscsi target out the network as Lun1 and Lun2
This example (below) is a little disingenuous since it didn't make a VG of two PVs

To start with, cat proc/partitions then consider sdb

```
[root@server1 ~] /vgcreate vgsan /dev/sdb
[root@server1 ~] pvs
PV             VG      Fmt   Attr   PSize   PFree
/dev/sda2      centos  lvm2  a--    19.51g  0
/dev/sdb       vgsan   lvm2  a--    1020.00m        1020.00m
[root@server1 ~] vgs
VG     #PV  #LV  #SN  Attr   VSize   VFree
centos 1    2    0    wz--n- 19.51g  0
vgsan  1    0    0--  wz--n- 1020.00m        1020.00m
[root@server1 ~] lvcreate -L 500M -n lvsan1 vgsan
  Logical volume "lgsan1" created
[root@server1 ~] lvcreate -L 500M -n lvsan2 vgsan
  Logical volume "lgsan2" created
```

Run lvs and you will see all your LVs. mnemonic for these commands as LVShow, VGShow
Remember that LVM create commands are parameters/name, then source

Install iscsi software
yum -y install targetcli
Running targetcli gives you a shell - you can do ls, cd, other simple commands
cd and select from the TUI "backstores" Type ls
Contents are block, fileio, pscsi, ramdisk - for your backend storage devices
Type "block/ create block1 /dev/vgsan/lvsan1" to create the block
"Created block storage object block1..."
Type "block/ create block2 /dev/vgsan/lvsan2" to create the block2
Type "fileio/ create file1 /root/diskfile1 1G" to create the fileio object
it will create the file for you! (it doesn't have to pre-exist)
This is what you end up with:

```
  o- backstores ..................................................... [...]
  | o- block ......................................... [Storage Objects: 2]
  | | o- block1 ..... [/dev/vgsan/lvsan1 (500.0MiB) write-thru deactivated]
  | | o- block2 ..... [/dev/vgsan/lvsan2 (500.0MiB) write-thru deactivated]
  | o- fileio ........................................ [Storage Objects: 1]
  | | o- file1 ............... [/root/diskfile1 (1.0GiB) write-back deactivated]
  | o- pscsi ......................................... [Storage Objects: 0]
  | o- ramdisk ....................................... [Storage Objects: 0]
  o- iscsi ................................................... [Targets: 0]
  o- loopback ................................................ [Targets: 0]
/> cd iscsi
/iscsi> ls
o- iscsi ................................................... [Targets: 0]
/iscsi> create iqn.2014-09.com.example:target1
Created target iqn.2014-09.com.example:target1.
Created TPG 1.
/iscsi> ls
o- iscsi ................................................... [Targets: 1]
  o- iqn.2014-09.com.example:target1 ............................ [TPGs: 1]
    o- tpg1 ....................................... [no-gen-acls, no-auth]
      o- acls ..................................................... [ACLs: 0]
      o- luns ..................................................... [LUNs: 0]
      o- portals ................................................. [Portals: 0]
/iscsi>
```

So moving around and going into iscsi, here is the first Target Portal Group created that is empty.
The syntax is "create iqn.2014-09.com.example:target1"
/iscsi> cd iqn.2014-09.com.example:target1
/iscsi/iqn.2014-09.com.example:target1/> cd tpg1
/iscsi/iqn...target1/tpg1/> acls/ create iqn.2014-09.com.example:server2
Created Node ACL for iqn iqn.2014-09.com.example:server2
/iscsi/iqn...target1/tpg1/>
So now when you ls, it will show server2 listed right under ACLs. Next we need to add LUNs from backstores in under server2's iqn inside ACLs
/iscsi/iqn...target1/tpg1/> luns/ create /backstores/block/block1
/iscsi/iqn...target1/tpg1/> luns/ create /backstores/block/block2
/iscsi/iqn...target1/tpg1/> luns/ create /backstores/fileio/file1

You get what you see next:

```
  o- luns ..................................................................... [LUNs: 0]
  o- portals ............................................................... [Portals: 0]
/iscsi/iqn.20...:target1/tpg1> luns/ create /backstores/block/block1
Created LUN 0.
Created LUN 0->0 mapping in node ACL iqn.2014-09.com.example:server2
/iscsi/iqn.20...:target1/tpg1> luns/ create /backstores/block/block2
Created LUN 1.
Created LUN 1->1 mapping in node ACL iqn.2014-09.com.example:server2
/iscsi/iqn.20...:target1/tpg1> luns/ create /backstores/fileio/file1
Created LUN 2.
Created LUN 2->2 mapping in node ACL iqn.2014-09.com.example:server2
/iscsi/iqn.20...:target1/tpg1> ls
o- tpg1 ..................................................... [no-gen-acls, no-auth]
  o- acls ................................................................... [ACLs: 1]
  | o- iqn.2014-09.com.example:server2 ..................... [Mapped LUNs: 3]
  |   o- mapped_lun0 ................................ [lun0 block/block1 (rw)]
  |   o- mapped_lun1 ................................ [lun1 block/block2 (rw)]
  |   o- mapped_lun2 ................................ [lun2 fileio/file1 (rw)]
  o- luns ................................................................... [LUNs: 3]
  | o- lun0 ................................ [block/block1 (/dev/vgsan/lvsan1)]
  | o- lun1 ................................ [block/block2 (/dev/vgsan/lvsan2)]
  | o- lun2 ................................ [fileio/file1 (/root/diskfile1)]
  o- portals ............................................................... [Portals: 0]
/iscsi/iqn.20...:target1/tpg1>
```

Finally, going down the list, portals is next. This is how we finally advertise it out the target
/iscsi/iqn...target1/tpg1/> portals/ create 192.168.1.100
/iscsi/iqn...target1/tpg1/> cd /
/iscsi/iqn...target1/tpg1/> ls

```
o- backstores ..................................................................... [...]
| o- block ......................................... [Storage Objects: 2]
| | o- block1 ........... [/dev/vgsan/lvsan1 (500.0MiB) write-thru activated]
| | o- block2 ........... [/dev/vgsan/lvsan2 (500.0MiB) write-thru activated]
| o- fileio ........................................ [Storage Objects: 1]
| | o- file1 ............... [/root/diskfile1 (1.0GiB) write-back activated]
| o- pscsi ......................................... [Storage Objects: 0]
| o- ramdisk ....................................... [Storage Objects: 0]
o- iscsi ................................................... [Targets: 1]
| o- iqn.2014-09.com.example:target1 ........................ [TPGs: 1]
|   o- tpg1 ..................................... [no-gen-acls, no-auth]
|     o- acls ................................................... [ACLs: 1]
|     | o- iqn.2014-09.com.example:server2 ................ [Mapped LUNs: 3]
|     |   o- mapped_lun0 ........................ [lun0 block/block1 (rw)]
|     |   o- mapped_lun1 ........................ [lun1 block/block2 (rw)]
|     |   o- mapped_lun2 ........................ [lun2 fileio/file1 (rw)]
|     o- luns ................................................... [LUNs: 3]
|     | o- lun0 ........................ [block/block1 (/dev/vgsan/lvsan1)]
|     | o- lun1 ........................ [block/block2 (/dev/vgsan/lvsan2)]
|     | o- lun2 ........................ [fileio/file1 (/root/diskfile1)]
|     o- portals ............................................... [Portals: 1]
|       o- 192.168.4.101:3260 ..................................... [OK]
o- loopback ................................................ [Targets: 0]
```

And that's it type exit and it saves.
Global pref auto_save_on_exit=true
Last 10 configs saved in /etc/target/backup
Configuration saved to /etc/target/saveconfig.json

Now you just have to configure firewalld so it is ready for it.

firewall-cmd --add-port=3260/tcp --permanent
firewall-cmd --reload
systemctl enable target

systemctl start target

man targetcli has TONS of examples and some quickstart stuff

-- Connecting the SAN up
On one of the other machines, set up the initiator:
yum -y install iscsi-initiator-utils

vim /etc/iscsi/initiatorname.iscsi
InitiatorName= iqn.2014-09.com.example:server2

cat /etc/iscsi/iscsid.conf -contains all of the settings you need to optimize the iscsi configuration

iscsiadm is the program to set up discovery, log in, and log out (also list etc- anything you'd need)
The commands are long and luckily the man page has plenty of examples since, as Vugt says- unless you do this ever day you are unlikely to remember them.

Here are the examples in the man page:

Discover targets at a given IP address:
    iscsiadm --mode discoverydb --type sendtargets --portal 192.168.1.10 --discover
Login, must use a node record id found by the discovery:
    iscsiadm --mode node --targetname iqn.2001-05.com.doe:test --portal 192.168.1.1:3260 --login
Logout:
    iscsiadm --mode node --targetname iqn.2001-05.com.doe:test --portal 192.168.1.1:3260 --logout
List node records:
    iscsiadm --mode node
Display all data for a given node record:
iscsiadm --mode node --targetname iqn.2001-05.com.doe:test --portal 192.168.1.1:3260

So we do a discovery on our IP address, get the iqn and tell it to log into it.
iscsiadm --mode discoverydb --type sendtargets --portal 192.168.1.100 --discover
    iqn.2014-09.com.example:target1
iscsiadm --mode node --targetname iqn.2014-09.com.example:target1 --portal 192.168.1.100 --login
    Logging in... successful.

yum -y install lsscisi
lists all scsi connections. Here is more info:
ls /var lib/iscsi/nodes
    192.168.1.100,3260,1
cat /var lib/iscsi/nodes/192.168.1.100,3260,1
    tons of info

The link should work until you log out, even after restart
You can delete the link entirely with this:
iscsiadm -m node -T iqn.2014-09.com.example:target1 -o delete
(or delete the directory)

Verifying the connection
iscsiadm -m session -P 0 or 1, 2, or 3 (the higher number gives more info)
iscsiadm -m node -P 1
iscsiadm -m discovery -P 1

On the server, just run targetcli

Exercise 2

Use server1 as your iSCSI Target server. Use server2 as the iSCSI initiator server.
Configure the iSCSI target server with two LUNS. Both LUNS needs to be based on a 1 GB LVM logical volume as storage backend. Configure everything to make this LUN operational. Configure LUN0 to be accessible by server2, configure LUN1 to be accessible by server1 **and** server2.
Configure the iSCSI Initiator on both server1 as server2. (It does not make sense to run an iSCSI initiator and target on the same server, but it helps you do this exercise without installing another server). Verify that server2 has access to both LUNS, where server1 has access to LUN1 only.
On server2, create an XFS file system on LUN1 and mount it permanently through fstab.

In the explanation, he mentions that XFS volume won't come up automatically on reboot because the system (fstab) knows the network setup isnt complete. Look in the man page for net dev mount options

Also says that you should know this well, because on the exam, there might be (a hypothetical example- it could be something else) a NFS share that depends on iscsi, and if this or that isn't done right, the later stuff might not work at all so you're screwed.