# Solution M3: Distributed and Shared Storage (CentOS)

# Task 1

**Challenge:** Create a **Samba** group share (one folder, two users, one group, accessible only by the group)

**Solution:**

Let’s install the required packages

**sudo dnf install samba samba-client samba-common policycoreutils-python-utils**

Let’s start and enable them to start on boot

**sudo systemctl enable --now smb nmb**

Add firewall exception for **Samba**

**sudo firewall-cmd --add-service samba --permanent**

And reload the configuration

**sudo firewall-cmd --reload**

Create a group, for example **sales**

**sudo groupadd sales**

And two users (**john** and **jane**) as members and without login shells

**sudo useradd -s /sbin/nologin -G sales john**

**sudo useradd -s /sbin/nologin -G sales jane**

Add them to the **SMB** user database and set their passwords

**sudo smbpasswd -a john**

**sudo smbpasswd -a jane**

Prepare the folder to be shared

**sudo mkdir -p /storage/sales**

Set the ownership and the permissions

**sudo chgrp sales /storage/sales**

**sudo chmod 2770 /storage/sales**

Adjust the **SELinux** context

**sudo chcon -R -t samba\_share\_t /storage/sales**

Set the default **SELinux** security context

**sudo semanage fcontext -at samba\_share\_t "/storage/sales(/.\*)?"**

Now, open the main configuration file for **Samba**

**sudo vi /etc/samba/smb.conf**

And add the following at the end

**[sales]**

**comment = Sales share**

**path = /storage/sales**

**writable = yes**

**valid users = @sales**

Save and close the file

Execute the following to test the configuration changes

**sudo testparm**

Reload the **Samba** services

**sudo systemctl reload smb nmb**

Let’s check for the available shares locally

**sudo smbclient -L //localhost**

Done. We are ready to use the share on a client machine

# Task 2

**Challenge:** Create an **NFS** share with different access (read-write and read-only) for two stations

**Solution:**

Install the required packages

**sudo dnf install nfs-utils**

Enable and start the service

**sudo systemctl enable --now nfs-server**

Open the firewall

**sudo firewall-cmd --add-service nfs --permanent**

And reload the rules

**sudo firewall-cmd --reload**

Create the folder to be exported

**sudo mkdir -p /storage/nfs**

In order to share our folder, we must edit the **/etc/exports** file

**sudo vi /etc/exports**

Enter the following to share the folder with everyone with read and write permissions

**/storage/nfs <ip-station-1>(rw) <ip-station-2>(ro)**

We can execute the following to apply the changes

**sudo exportfs -rav**

And if we want more information about the options of the share, we can execute just

**sudo exportfs -v**

Done. We are ready to use the exported folder on a client machine

# Task 3

**Challenge:** Create an **iSCSI** disk-based target

**Solution:**

Assume that we have an extra disk (for example 1 GB in size) attached to the machine as **sdb**

Install the required package

**sudo dnf install targetcli**

Start the administration tool

**sudo targetcli**

Switch to the **block** backend

**cd backstores/block**

Create an **iSCSI** disk based on a block device (existing hard disk)

**create D1 /dev/sdb**

Switch to the **iscsi** functions

**cd /iscsi**

Define a new target

**create iqn.2021-09.lab.lsaa:m1.tgt1**

Enter the target

**cd iqn.2021-09.lab.lsaa:m1.tgt1/tpg1/luns**

Create a **LUN** using the disk created earlier

**create /backstores/block/D1**

Adjust the access to the resource

**cd ../acls**

Register the initiator

**create iqn.2021-09.lab.lsaa:m2.init1**

Enter the record

**cd iqn.2021-09.lab.lsaa:m2.init1/**

Set user and password

**set auth userid=demo**

**set auth password=demo**

Exit the administrative tool

**exit**

Adjust the firewall

**sudo firewall-cmd --add-service iscsi-target --permanent**

**sudo firewall-cmd --reload**

Enable and start the **target** service

**sudo systemctl enable --now target**

Done. We are ready to use the device on a client machine

# Task 4

**Challenge:** Create a **GlusterFS** dispersed volume

**Solution:**

Information on the topic:

<https://docs.gluster.org/en/latest/Administrator-Guide/Setting-Up-Volumes/#creating-dispersed-volumes>

Assume that we have three machines for this set up – **M1**, **M2**, and **M3**

### Preparation

Log in to **M1**

Install the missing repository

**sudo dnf install centos-release-gluster7**

Enable **GlusterFS** and **PowerTools** repositories

**sudo dnf config-manager --set-enabled centos-gluster7**

**sudo dnf config-manager --set-enabled powertools**

Install the required packages

**sudo dnf install glusterfs glusterfs-server**

Enable and start the service

**sudo systemctl enable --now glusterd**

Adjust the firewall settings

**sudo firewall-cmd --add-service=glusterfs --permanent**

**sudo firewall-cmd --reload**

Create the folder

**sudo mkdir -p /storage/glusterfs**

Repeat this procedure on the second node (**M2**) and the third node (**M3**) as well

### Configuration

Return on **M1**

Test if there is a communication with the second and third node

**sudo gluster peer probe m2**

**sudo gluster peer probe m3**

Confirm the status

**sudo gluster peer status**

Create a volume

**sudo gluster volume create vol01 disperse-data 2 redundancy 1 transport tcp m1:/storage/glusterfs m2:/storage/glusterfs m3:/storage/glusterfs force**

As we are not using a separate partition, we must add **force** at the end of the above command

Get information about the volume

**sudo gluster volume info vol01**

If the volume is not started, you can do it with

**sudo gluster volume start vol01**

We can check the status of the volume

**sudo gluster volume status vol01**

Done. We are ready to use the volume on a client machine