ONTAP LoadShare Mirroring - part1

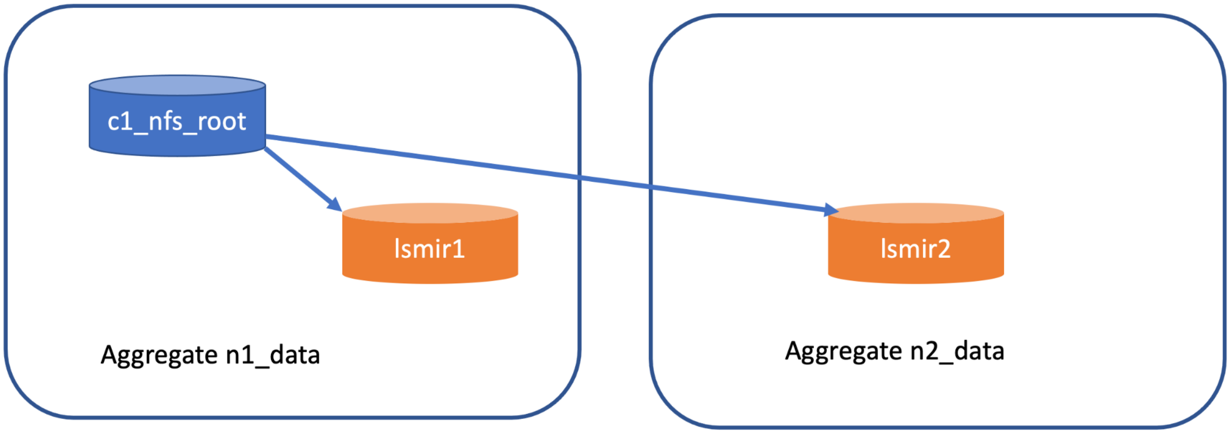
This lab consist of two parts.

Requirements:
There is a running SVM called c1_nfs
The default export policy has ro, rw and superuser for any

This is what you will do:

1. Create two destination volumes in the c1_nfs SVM, using type DP
2. Create the loadshare relationships to both destination volumes using type LS
3. Initialize the loadshare relationships
4. List the snapmirror relationships
5. Create and mount a volume to be used by the NFS client
6. Make sure there is an rule in the default export policy of the c1_nfs SVM default policy
7. From the linux client list the exported volumes from c1_nfs
8. On linux, create a mountpoint and mount the volume
9. Unmount the playvol volume on cluster1
10. From the linux client check wether the volume still seams to be exports and access it
11. On cluster1 run a snapmirror update-ls-set of the root-volume of c1_nfs
12. On the linux client check whether the cluster still shows the junction-path

(see next page for the commands)



Snapmirror Loadshare Relationship
SVM Rootvolume protection

Commands

# 1. Create two destination volumes in the c1_nfs SVM, using type DP

```
cluster1::>
vol create -vserver c1_nfs -volume lsmir1 -type dp -aggregate n1_data
vol create -vserver c1_nfs -volume lsmir2 -type dp -aggregate n2_data
```

# 2. Create the loadshare relationships to both destination volumes using type LS

```
cluster1::>
snapmirror create -source-path c1_nfs:c1_nfs_root -destination-path c1_nfs:lsmir1 -type LS
snapmirror create -source-path c1_nfs:c1_nfs_root -destination-path c1_nfs:lsmir2 -type LS
```

# 3. Initialize the loadshare relationships

```
cluster1::>
snapmirror initialize-ls-set -source-path c1_nfs:c1_nfs_root
```

# 4. List the snapmirror relationships

```
cluster1::>
rows 0
snapmirror show
cluster1://c1_nfs/c1_nfs_root LS cluster1://c1_nfs/lsmir1 Snapmirrored Idle - true -
                                 cluster1://c1_nfs/lsmir2 Snapmirrored Idle - true -
```

# 5. Create and mount a volume to be used by the NFS client

```
cluster1::>
vol create -vserver c1_nfs -volume playvol -aggregate n1_data -size 1g -junction-path /playvol
```

# 6. Make sure there is an rule in the default export policy of the c1_nfs SVM default policy
And enable NFS

```
cluster1::>
export-policy rule show -vserver c1_nfs -policyname default
nfs on -vserver c1_nfs
```

# 7. From the linux client list the exported volumes from c1_nfs

```
Linux:
showmount -e 192.168.4.209
Export list for 192.168.4.209:
/        (everyone)
/playvol (everyone)
```

# 8. On linux, create a mountpoint and mount the volume

```
Linux:
```

```
mkdir /mnt/playvol
mount 192.168.4.209:/playvol /mnt/playvol
```

```
# 9. Unmount the playvol volume on cluster1
cluster1::>
vol unmount -volume playvol -vserver c1_nfs

Notice: Volume playvol has been unmounted from volume c1_nfs_root.  The load sharing (LS)
mirrors of volume c1_nfs_root have no replication schedule.
        Volume playvol will remain visible in the global namespace until the LS mirrors of volume
c1_nfs_root have been updated.
```

```
# 10. From the linux client check wether the volume still seams to be exports and  access it
Linux:
showmount -e 192.168.4.209
Export list for 192.168.4.209:
/        (everyone)
/playvol (everyone)

ls /mnt/playvol
ls: cannot open directory '/mnt/playvol': Stale file handle
```

NOTE: The reason for this is that, for reading, the loadshare destination volumes are used and never the actual rootvolume. So you will have to either schedule the updates or better even - run an update manually after every vol mount and vol unmount action.

```
# 11. On cluster1 run a snapmirror update-ls-set of the root-volume of c1_nfs
cluster1::>
snapmirror update-ls-set -source-path c1_nfs:c1_nfs_root
```

```
# 12. On the linux client check whether the cluster still shows the junction-path
Linux:
showmount -e 192.168.4.209
Export list for 192.168.4.209:
/       (everyone)
```