VIOS shared storage pool and thin provisioning

VIOS clustering

Karthikeyan Kannan July 03, 2013

I wondered why IBM® Power Systems™ does not have the snapshot and thin-provisioning features that can make the life of a system administrator much easier when there is a need to upgrade or maintain efficiency. Finally, I found it in Virtual I/O Server (VIOS) shared storage pool, which I have detailed in this article.

Introduction to VIOS shared storage pool

I love Power Systems and always wondered why Power Systems doesn't have snapshot and thin-provisioning features. Finally, I found that these are enabled in IBM Power Systems too with the introduction to the shared storage pool concept.

Shared storage pool, as the name states is basically to share storage resources (SAN disks) across a group of IBM VIOS instances. Not just as a physical disk, but slicing them like a logical volume inside the shared storage pool, which is denoted as a logical unit (LU). A LU is basically file-backed storage present in the clustered storage pool.

The VIOS needs to be at a minimum of version 2.2.0.11. I tested the functionality in VIOS 2.2.1.4. With the present release of VIOS 2.2.2.1, you can have 16 VIOS nodes in a cluster and can support up to 200 clients per VIOS node.

The shared storage pool concept takes advantage of the Cluster Aware AIX (CAA) feature in the IBM AIX® operating system to form a cluster of VIOS. Using the CAA feature, the cluster can monitor the peers in the cluster. Refer to Chris Gibson's blog for more information about CAA.

In this article, I am using two VIOS instances hosted on two different physical systems. We will see details about the following tasks as you navigate through this article.

- Creating a cluster and a shared storage pool
- Verifying the status of the cluster
- Listing the share storage pool attributes
- Creating a logical unit
- Assigning a logical unit to a client

- Modifying a cluster

Features include:

- Thick provisioning
- Thin provisioning
- Snapshot feature
  - Create
  - Rollback
  - Delete

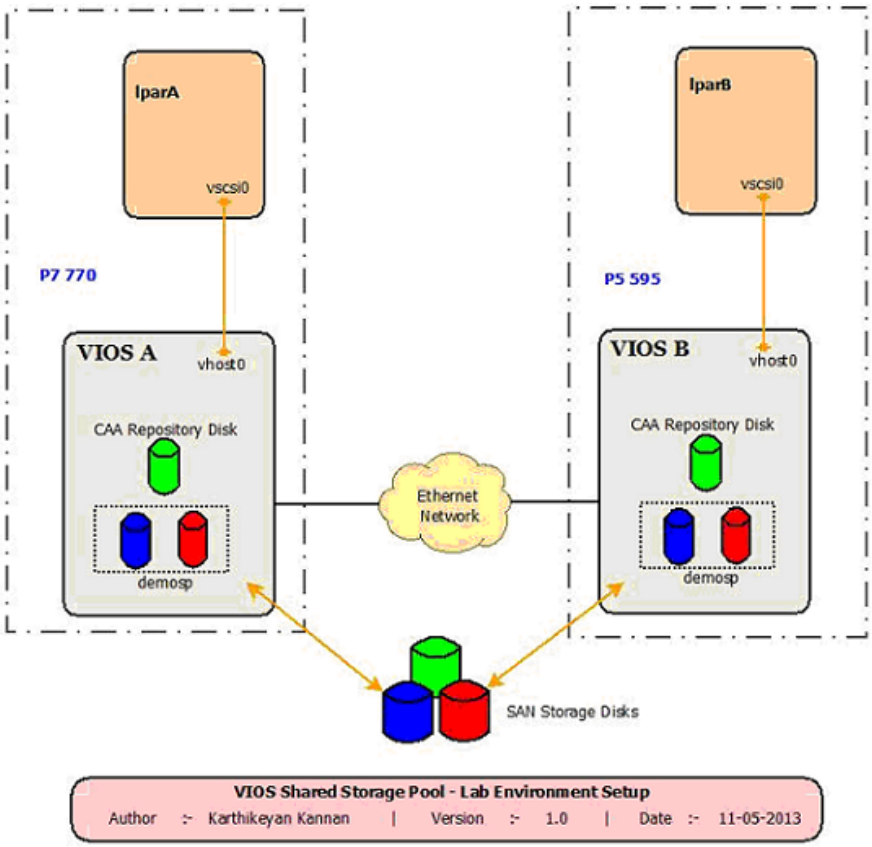
Requirements

- IBM PowerVM® Standard Edition
- VIOS version 2.2.0.11, Fix Pack 24, Service Pack 1 and later
- Minimum two disks: One for the CAA repository and the other for the storage pool.

Lab setup

Figure 1 shows the lab setup that I have used to illustrate this feature throughout the article.

Figure 1. Lab environment setup



We will also log in to both the VIOS and verify the configuration.

Listing 1. On VIOS A

```
$ hostname
VIOSA
$ ioslevel
2.2.1.4
$ lspv
NAME            PVID                                VG              STATUS
hdisk0          00c858a2bde1979e                    rootvg          active
hdisk1          00c858a2cbd45f6b                    None
hdisk2          00c858a2cca2a81d                    None
hdisk3          00c858a210d30593                    None
hdisk4          00c858a210d32cfd                    None
$ lsvg
rootvg
$ lssp
Pool             Size(mb)   Free(mb)  Alloc Size(mb)    BDs Type
rootvg             102272      77824             128      0 LVPOOL
$
$ cluster -list
$
```

Listing 2. On VIOS B

```
$ hostname
VIOSB
$ ioslevel
2.2.1.4
$ lspv
NAME            PVID                                VG              STATUS
hdisk0          00c858a2bde1979e                    rootvg          active
hdisk1          00c9095f0f795c20                    None
hdisk2          00c858a2cca2a81d                    None
hdisk3          00c858a210d30593                    None
hdisk4          00c858a210d32cfd                    None
$ lsvg
rootvg
$ lssp
Pool             Size(mb)   Free(mb)  Alloc Size(mb)    BDs Type
rootvg             102272      77824             128      0 LVPOOL
$
$ cluster -list
$
```

I have five disks in my VIOS systems, hdisk0 and hdisk1 are used by the VIOS and client logical partition or LPAR (rootvg) respectively in both the VIOS. The disks that we are going to use are hdisk2, hdisk3, and hdisk4. Look at the physical volume ID (PVID) for all of them; they are same on both the VIOS, which confirms that the same set of physical disks is shared between both the VIOS instances. The order of their naming does not need to be the same, as it is the PVID that matters.

You also need to ensure that the VIOS nodes in a cluster are reachable in the IP network. You should be able to resolve their hostnames either by using */etc/hosts* or by DNS.

Creating a shared storage pool

Now that our playground is ready, let's start the game by creating a VIOS cluster and a shared storage pool. This should be performed using the *cluster* command that initializes the cluster process and creates a shared storage pool.

For our demo cluster, I am using hdisk2 for the CAA repository disk that holds all the vital data about the cluster and hdisk3 and hdisk4 for shared storage pool.

Listing 3. On VIOS A

```
$ cluster -create -clustername demo1 -repopvs hdisk2 -spname demosp -sppvs
hdisk3 hdisk4 -hostname viosa
Cluster demo1 has been created successfully.

$
```

As soon as the command completes successfully, we can verify the status of the cluster and its attributes using the `–list` and `-status` flags of the cluster command.

Listing 4. On VIOS A

```
$ cluster -list
CLUSTER_NAME:    demo1
CLUSTER_ID:      36618f14582411e2b6ea5cf3fceba66d
$

$ cluster -status -clustername demo1
Cluster Name         State
demo1                    OK

    Node Name        MTM                       Partition Num  State  Pool State
    VIOSA            9117-MMC0206858A2           39              OK       OK

$
```

The above code ran on VIOS A tells us that there is a cluster with the name *demo1* created with the Cluster ID *36618f14582411e2b6ea5cf3fceba66d*. This cluster ID is a unique identifier for each cluster that is created. The command cluster status indicates the status of the cluster denoting whether the cluster is in the operating state or do we have any problems in it. It also gives useful information (such as model type, serial number, and the partition ID of the hosting VIOS) about the physical system.

We can also use the CAA commands, such as `lscluster` to view the status of the cluster to ensure that it is operational.

Listing 5. On VIOS A

```
$ lscluster -m
Calling node query for all nodes
Node query number of nodes examined: 1
        Node name: VIOSA
        Cluster shorthand id for node: 1
        uuid for node: 365731ea-5824-11e2-b6ea-5cf3fceba66d
        State of node:  UP  NODE_LOCAL
        Smoothed rtt to node: 0
        Mean Deviation in network rtt to node: 0
        Number of clusters node is a member in: 1
        CLUSTER NAME       TYPE  SHID   UUID

        demo1              local        36618f14-5824-11e2-b6ea-5cf3fceba66d

        Number of points_of_contact for node: 0
        Point-of-contact interface & contact state
         n/a

$
```

So far, we been verifying only the cluster, where did the storage pool go? The cluster command will neither show you the shared storage pool created using the cluster command nor the `lssp` command.

To view the shared storage pool, we need to use the legacy `lssp` command of VIOS, which is used to list the storage pools, but with a special flag *–clustername*.

The command format to list the shared storage pool available within the cluster is `lssp -clustername <NAME>`.

Listing 6. On VIOS A

```
$ lspv
NAME        PVID                 VG         STATUS
hdisk0       00c858a2bde1979e            rootvg       active
hdisk1       00c858a2cbd45f6b            None
hdisk2       00c858a2cca2a81d            caavg_private   active
hdisk3       00c858a210d30593            None
hdisk4       00c858a210d32cfd            None

$ lssp
Pool        Size(mb)   Free(mb)  Alloc Size(mb)   BDs Type
rootvg        102272    77824        128    0 LVPOOL

$ lsvg
rootvg
caavg_private

$ lssp -clustername demo1
POOL_NAME:       demosp
POOL_SIZE:       102144
FREE_SPACE:      100391
TOTAL_LU_SIZE:   0
TOTAL_LUS:       0
POOL_TYPE:       CLPOOL
POOL_ID:         000000000979382300000000050E9B08C

$
```

In the above output, you can see that the name of the shared storage pool is **demosp**, the total size of the shared storage pool is 100 GB, and the free space is 100391 MB. You can also see the fields pointing to the number of LUs and the total size of the LUs is 0, as we do not have any LUs created so far. Along with a unique cluster ID, the shared storage pool also gets a unique identifier.

You may also need to note that there is a new volume group (VG) named, `caavg_private`, created along with the shared storage pool. This VG is CAA-specific and the disks that are part of this VG stores the vital data to keep the cluster alive and running. You should not use this VG for any other purposes.

Logical unit

As stated in the start of the article, a *logical unit (LU)* is a file-backed storage device that can be presented to a VIOS client as a virtual SCSI (VSCSI) disk-backing device.

Now, we need to create a LU on top of the shared storage pool. In VIOS A, we already have a vhost0 connection created, through which the lparA gets a physical hard disk for its rootvg.

Listing 7. On VIOS A

```
$ lsmap -all
SVSA            Physloc                                        Client Partition ID
--------------- ---------------------------------------------- ------------------
vhost0          U9117.MMC.06858A2-V39-C3                       0x0000000f

VTD                   LPARA_RVG
Status                Available
LUN                   0x8100000000000000
Backing device        hdisk1
Physloc               U78C0.001.DBJ0379-P2-C3-T1-W500507680120D9ED-L1000000000000
Mirrored              false

$
```

Now all the setup for us to have the LU created for the client LPAR A from the shared storage pool. The LU is also going to be same as a LV or file-backed backing device on top of a storage pool. The command we use to create it is the same VIOS command *mkbdsp*, with some additional flags.

Listing 8. On VIOS A

```
$ mkbdsp -clustername demo1 -sp demosp 20G -bd lparA_lu1
Lu Name:lparA_lu1
Lu Udid:2f4adc720f570eddac5dce00a142de89

$
```

In the above output, I used the `mkbdsp` command to create the LU first. I have created a LU of size 20 GB on demosp (which is not mapped to any client yet). To map it, you need to again use the `mkbdsp` command as shown in Listing 9.

Listing 9. On VIOS A

```
$ mkbdsp -clustername demo1 -sp demosp -bd lparA_lu1 -vadapter vhost0 -tn lparA_datavg
Assigning file "lparA_lu1" as a backing device.
VTD:lparA_datavg

$
```

Note that I have not mentioned the size here because the LU already exists. This command will map the LU `lparA_lu1` to vhost0 with the VTD name, **lparA_datavg**.

Instead of going with two steps, one for creating and one for assigning to a client, we can perform both of these operations in a single command as depicted in the following output. Before that, I will have to delete the VTD *lparA_datavg* backed by the LU *lparA_lu1* which we just mapped. We can use the usual *rmvdev* for the VTD and *rmbdsp* for the LU.

Listing 10. On VIOS A

```
$ rmvdev -vtd lparA_datavg
lparA_datavg deleted
$ rmbdsp -clustername demo1 -sp demosp -bd lparA_lu1
Logical unit lparA_lu1 with udid "a053cd56ca85e1e8c2d98d00f0ab0a0b" is removed.
$
```

Now, I will create and map the LU in a single command as shown in the following output.

Listing 11. On VIOS A

```
$ mkbdsp -clustername demo1 -sp demosp 20G -bd lparA_lu1 -vadapter vhost0 -tn lparA_datavg
Lu Name:lparA_lu1
Lu Udid:c0dfb007a9afe5f432b365fa9744ab0b

Assigning file "lparA_lu1" as a backing device.
VTD:lparA_datavg

$
```

As the LU is created and mapped, the client should be able to see the LU as a disk. We will verify the LU and the mapping in VIOS A once and move over to the client lparA which is a client of VIOS A.

The `lssp` command can be used to list the backing devices in the shared storage pool.

Listing 12. on VIOS A

```
$ lssp -clustername demo1 -sp demosp -bd
Lu Name          Size(mb)    ProvisionType    Lu Udid
lparA_lu1        20480       THIN             c0dfb007a9afe5f432b365fa9744ab0b
$
```

The following output in Listing 13 (in bold) shows that the lparA_lu1 LU is mapped to the lparA client on vhost0.

Listing 13. On VIOS A

```
$ lsmap -all
SVSA            Physloc                                     Client Partition ID
--------------- ------------------------------------------- ------------------
vhost0          U9117.MMC.06858A2-V39-C3                    0x0000000f

VTD                   LPARA_RVG
Status                Available
LUN                   0x8100000000000000
Backing device        hdisk1
Physloc               U78C0.001.DBJ0379-P2-C3-T1-W500507680120D9ED-L1000000000000
Mirrored              false

VTD                      lparA_datavgStatus                   AvailableLUN
                      0x8200000000000000Backing device    lparA_lu1.c0dfb007a9afe5f432b365fa9744ab0bPhyslocMirrored
A

$
```

On the client machine lparA, we already have one physical volume that is used by rootvg. Now, the new disk should be available for the client to use. We will attempt to configure the LU provided to the client.

Listing 14. On lparA

```
lparA#hostname
lparA
lparA#lspv
hdisk0          00c858a2cbd45f6b                    rootvg          active
lparA#cfgmgr
lparA#lspv
hdisk0          00c858a2cbd45f6b                    rootvg          active
hdisk1          none                                None
# lscfg -vpl hdisk1
  hdisk1           U9117.MMC.06858A2-V15-C2-T1-L8200000000000000  Virtual SCSI Disk Drive

  PLATFORM SPECIFIC

  Name:  disk
    Node:  disk
    Device Type:  block
#
```

We made it through! The LU is now available to the client as a virtual SCSI disk drive.

Snapshot and restore

So far, we have set up a shared storage pool with a single VIOS, created a logical unit, and assigned the LU to the lparA client.

Now let's concentrate on some VG operations on the client side to explore the snapshot feature of shared storage pool. Using the new LU provided to lparA, I am going to create a volume group (datavgA) and a file system, named */datafsA*, on top of it.

Listing 15. On lparA

```
lparA#mkvg -y datavgA hdisk1
0516-1254 mkvg: Changing the PVID in the ODM.
datavgA
lparA#crfs -v jfs2 -m /datafsA -g datavgA -a size=2G
File system created successfully.
2096884 kilobytes total disk space.
New File System size is 4194304
lparA#
lparA#mount /datafsA
lparA#cd /datafsA
lparA#touch file_lparA
lparA#ls
file_lparA  lost+found
lparA#
```

Now, we will create two files. One named *before_snap* after which I will take a snapshot and then one more named *after_snap*. We will also restore the snapshot for demonstration.

Listing 16. On lparA

```
lparA#touch before_snap
lparA#ls
before_snap  file_lparA   lost+found
lparA#pwd
/datafsA
lparA#
```

On VIOS A, we will take a snapshot now.

The command to capture a snapshot is:

snapshot -clustername <Clustername> -spname <Shared_Pool_Name> -luudid <ID> -create SNAP_NAME

Listing 17. On VIOS A

```
$ snapshot -clustername demo1 -create lparA_lu1_SNAP1 -spname demosp -lu lparA_lu1
lparA_lu1_SNAP1
$
$ lssp -clustername demo1 -sp demosp -bd
Lu Name          Size(mb)    ProvisionType    Lu Udid
lparA_lu1        20480       THIN             687f8420bbeee7a5264ce2c6e83d3e66
Snapshot
lparA_lu1_SNAP1

$
```

The `lssp` command in the above code listing indicates that there is a snapshot named *lparA_lu1_SNAP1* associated with the *lparA_lu1* LU.

Now, we will create one more file named *after_snap* in the lparA client.

Listing 18. On lparA

```
lparA#pwd
/datafsA
lparA#touch after_snap
lparA#ls
after_snap   before_snap  file_lparA   lost+found
lparA#cd
lparA#umount /datafsA
lparA#varyoffvg datavgA
lparA#
```

I have varied off the volume group now with data on it. It is always recommended to take the resources offline incase you want to restore some data. You should be familiar with it on your experience.

Let's try to restore the *lparA_lu1_SNAP1* snapshot and see what data is present in the volume group on the client side.

Listing 19. On VIOS A

```
$ snapshot -clustername demo1 -rollback lparA_lu1_SNAP1 -spname demosp -lu lparA_lu1
$
```

Listing 20. On lparA

```
lparA#varyonvg datavgA
lparA#mount /datafsA
Replaying log for /dev/fslv00.
lparA#ls -l /datafsA
total 0
-rw-r--r--    1 root     system            0 Jan  6 23:27 before_snap
-rw-r--r--    1 root     system            0 Jan  6 23:24 file_lparA
drwxr-xr-x    2 root     system          256 Jan  6 23:23 lost+found
lparA#
```

After the snapshot was restored and the volume group brought back online, there is no file named *after_snap*. This is because the file was created after the snapshot. Now that we rolled back the snapshot, it does not exist.

If you want to delete the snapshot, you can use the `snapshot` command, as shown in the following listing.

Listing 21. On VIOS A

```
$ snapshot -clustername demo1 -delete lparA_lu1_SNAP1 -spname demosp -lu lparA_lu1
$
```

Modifying the cluster

So far, whatever we performed is only on VIOS A and lparA. The cluster what we have created is also a single-node cluster. You can ask me, what is a single node cluster and how can it be? Well that is what the CAA feature of AIX dictates. A cluster can be created with a single node too.

Let's expand our cluster, *demo1*, by adding the second VIOS instance VIOS B on the other CEC.

Listing 22. On VIOS A

```
$ cluster -addnode -clustername demo1 -hostname viosb
Partition VIOSB has been added to the demo1 cluster.

$
```

The above command has added VIOS B to the cluster. Let's verify it with the *cluster -status* command.

Listing 23. On VIOS A

```
$ cluster -status -clustername demo1
Cluster Name         State
demo1                OK

    Node Name        MTM           Partition Num  State  Pool State
    VIOSA            9117-MMC0206858A2        39  OK     OK
    VIOSB            9119-59502839095F         3  OK     OK
$
```

If you see the above output, it clearly states that VIOS A and VIOS B are hosted on two different physical systems, and now both are part of the VIOS cluster, *demo1.*

Now, we can move to VIOS B and check if the entire configuration that we did on VIOS A has really reflected in VIOS B.

Listing 24. On VIOS B

```
$ hostname
VIOSB
$ cluster -list
CLUSTER_NAME:    demo1
CLUSTER_ID:      36618f14582411e2b6ea5cf3fceba66d

$ lscluster -m
Calling node query for all nodes
Node query number of nodes examined: 2


        Node name: VIOSA
        Cluster shorthand id for node: 1
        uuid for node: 365731ea-5824-11e2-b6ea-5cf3fceba66d
        State of node:  UP
        Smoothed rtt to node: 7
        Mean Deviation in network rtt to node: 3
        Number of clusters node is a member in: 1
        CLUSTER NAME       TYPE  SHID   UUID
        demo1              local        36618f14-5824-11e2-b6ea-5cf3fceba66d

        Number of points_of_contact for node: 2
        Point-of-contact interface & contact state
         dpcom  UP  RESTRICTED
         en3  UP

------------------------------

        Node name: VIOSB
```

```
        Cluster shorthand id for node: 2
        uuid for node: a9d1aeee-582d-11e2-bda1-5cf3fceba66d
        State of node:  UP  NODE_LOCAL
        Smoothed rtt to node: 0
        Mean Deviation in network rtt to node: 0
        Number of clusters node is a member in: 1
        CLUSTER NAME       TYPE  SHID   UUID
        demo1              local        36618f14-5824-11e2-b6ea-5cf3fceba66d

        Number of points_of_contact for node: 0
        Point-of-contact interface & contact state
         n/a
$

$ lssp -clustername demo1
POOL_NAME:       demosp
POOL_SIZE:       102144
FREE_SPACE:      100353
TOTAL_LU_SIZE:   20480
TOTAL_LUS:       1
POOL_TYPE:       CLPOOL
POOL_ID:         000000000979382300000000050E9B08C

$ lssp -clustername demo1 -sp demosp -bd
Lu Name          Size(mb)    ProvisionType    Lu Udid
lparA_lu1        20480       THIN             687f8420bbeee7a5264ce2c6e83d3e66
Snapshot
lparA_lu1_SNAP1
$
```

We have verified from the above command output that VIOS B is also connected to the cluster and the shared storage pool, *demosp*, is available on VIOS B. I tried to map the LU lparA_lu1 to the client lparB, which is connected to VIOS B.

Listing 25. On VIOS B

```
$ lsmap -all
SVSA            Physloc                                      Client Partition ID
--------------- -------------------------------------------- ------------------
vhost0          U9119.595.839095F-V3-C2                      0x00000004

VTD                   lparB_RVG
Status                Available
LUN                   0x8100000000000000
Backing device        hdisk1
Physloc               U5791.001.99B0PA1-P2-C02-T1-W500507680110D9E3-L1000000000000
Mirrored              false

$


$ mkbdsp -clustername demo1 -sp demosp -bd lparA_lu1 -vadapter vhost0 -tn lparB_datavgA
Assigning file "lparA_lu1" as a backing device.
VTD:datavgA
$

 lsmap -all
SVSA            Physloc                                      Client Partition ID
--------------- -------------------------------------------- ------------------
vhost0          U9119.595.839095F-V3-C2                      0x00000004

VTD                   lparB_RVG
Status                Available
LUN                   0x8100000000000000
Backing device        hdisk1
```

```
Physloc                 U5791.001.99B0PA1-P2-C02-T1-W500507680110D9E3-L1000000000000
Mirrored                false

VTD                     lparB_datavgA
Status                  Available
LUN                     0x8200000000000000
Backing device          lparA_lu1.687f8420bbeee7a5264ce2c6e83d3e66
Physloc
Mirrored                N/A

$
```

Yes, I am able to successfully map the same LU to lparA and lparB at the same time. I can now log in to lparB and see if it got the disk visible for the client OS.

Listing 26. On LPAR B

```
lparB#lspv
hdisk0          00c9095f0f795c20                    rootvg          active
hdisk1          00c858a210fdef5e                    None
lparB#
```

Listing 27. On LPAR A

```
lparA#lspv
hdisk0          00c858a2cbd45f6b                    rootvg          active
hdisk1          00c858a210fdef5e                    datavgA         active
lparA#
```

Looking at the above output, I can confirm that we are able to share the same LU for two clients at the same time. Notice that the PVID is similar in both LPARs. Now, you can use the functionality to access the disk on both the clients. Beware of data corruption and use the right technology to access the disk, be it Logical Volume Manager (LVM) with concurrent or enhanced concurrent VG.

We have seen how to expand the cluster. We will also see how to shrink the cluster, that is, remove a VIOS node from the cluster.

Before removing a VIOS from the cluster, ensure that there is no LU provided to any clients from the specific VIOS that you intend to remove. In our case, we will remove VIOSB from the cluster.

Listing 28. On VIOS A

```
$ cluster -rmnode -clustername demo1 -hostname viosb
PARTITION HAS MAPPINGS
VIOSB

Command did not complete.

$
```

Oops!!! The command failed.

This is because we have not removed the mapping of the *lparA_lu1* LU that was provided to lparB through VIOSB. We can delete VTD mapping and rerun the command or use the *-f* flag. I'm using the *–f* flag because I know that there is only one LU mapped. Using the *-f* flag will remove all the

VTD devices created using the LUs from that specific cluster. If you have multiple mapping, you need to verify and then proceed.

Listing 29. On VIOS A

```
$ cluster -rmnode -f -clustername demo1 -hostname viosb
Partition VIOSB has been removed from the demo1 cluster

$
```

In case you need to add additional disks to the shared storage pool, you can use the following command format. I have not run it as I do not have an additional disk.

chsp -add -clustername <cluster_name> -sp <ssp_name> hdiskn

Thin and thick provisioning

We have not touched up on one thing yet, which is *thin provisioning*. You do not need to perform or set up anything exclusively on a shared storage pool for using *thin provisioning*. If you would have seen all the output of the `lssp` commands, you can see the header named "ProvisionType" and throughout our demonstration, all the LUs were thin provisioned. This is because in a shared storage pool, the default behavior is thin provisioning.

If you want to thick provision a LU, you need to specifically mention it using a `-thick` flag with the `mkbdsp` command.

Listing 30. On VIOS A

```
$ lssp -clustername demo1 -sp demosp -bd
Lu Name          Size(mb)    ProvisionType    Lu Udid
lparA_lu1        20480       THIN             687f8420bbeee7a5264ce2c6e83d3e66
$
```

We will try creating a *thick provisioned* LU for demonstration.

Listing 31. On VIOS A

```
$ lssp -clustername demo1 -sp demosp -bd
Lu Name          Size(mb)    ProvisionType    Lu Udid
lparA_lu1        20480       THIN             687f8420bbeee7a5264ce2c6e83d3e66

$
$ mkbdsp -clustername demo1 -sp demosp 50G -bd lparA_lu2 -vadapter
vhost0 -tn lparA_datavg_D2 -thick
Lu Name:lparA_lu2
Lu Udid:0ceaf03105d97f45ef4c595968f61cf7

Assigning file "lparA_lu2" as a backing device.
VTD:lparA_datavg_D2

$
$ lssp -clustername demo1 -sp demosp -bd
Lu Name          Size(mb)    ProvisionType    Lu Udid
lparA_lu1        20480       THIN             687f8420bbeee7a5264ce2c6e83d3e66
lparA_lu2        51200       THICK            0ceaf03105d97f45ef4c595968f61cf7
$
```

```
$ lsmap -all
SVSA            Physloc                                        Client Partition ID
--------------- ---------------------------------------------- ------------------
vhost0          U9117.MMC.06858A2-V39-C3                       0x0000000f

VTD                   LPARA_RVG
Status                Available
LUN                   0x8100000000000000
Backing device        hdisk1
Physloc               U78C0.001.DBJ0379-P2-C3-T1-W500507680120D9ED-L1000000000000
Mirrored              false

VTD                   lparA_datavg
Status                Available
LUN                   0x8200000000000000
Backing device        lparA_lu1.687f8420bbeee7a5264ce2c6e83d3e66
Physloc
Mirrored              N/A

VTD                   lparA_datavg_D2
Status                Available
LUN                   0x8300000000000000
Backing device        lparA_lu2.0ceaf03105d97f45ef4c595968f61cf7
Physloc
Mirrored              N/A

$
```

Now, take a look at the above output. The new LU that we have created is a thick provisioned LU and it has been also mapped to the client lparA.

Thin provisioning helps you to over commit the storage resources available. For example, consider we have a 20 GB LU and a 50 GB LU in our shared storage pool. Now let's say we have a requirement for a client for 50 GB of space. We cannot fulfill this request in a normal VG scenario or if we have used thick provisioning of LUs in a shared storage pool. Now, as we have used thin provisioning for *lparA_lu1*, the unused space by the client is available for use. You can also see the output of the `lssp` command in the following listing, which tells that there is 49 GB of free space in the shared storage pool.

Listing 32. On VIOS A

```
$ lssp -clustername demo1 -sp demosp -bd
Lu Name          Size(mb)    ProvisionType    Lu Udid
lparA_lu1        20480       THIN             687f8420bbeee7a5264ce2c6e83d3e66
lparA_lu2        51200       THICK            0ceaf03105d97f45ef4c595968f61cf7
$
$ lssp -clustername demo1
POOL_NAME:       demosp
POOL_SIZE:       102144
FREE_SPACE:      49150
TOTAL_LU_SIZE:   71680
TOTAL_LUS:       2
POOL_TYPE:       CLPOOL
POOL_ID:         000000000979382300000000050E9B08C
$ mkbdsp -clustername demo1 -sp demosp 50G -bd testlu1 -thick
Storage Pool subsystem operation, unable to create LU.
Storage Pool subsystem operation, not enough space in the pool.

$ mkbdsp -clustername demo1 -sp demosp 50G -bd testlu1
Lu Name:testlu1
Lu Udid:9e75b355e376eb81914df20bfb6c07f1
```

```
$
```

I tried to create a thick provisioned LU of 50 GB, but it failed due to insufficient space, whereas the command without the *–thick* flag was successful as it is thin provisioned.

Using thin provisioning also puts a risk of over-committing your storage resources. Though it is an advantage of virtualization, it also brings you a risk if you do not have control over the usage limit. Assume a scenario where all your clients started occupying whatever is allocated to them. In this case, you will be ending up with a problem of LVM write errors in the clients if the LUs are thin provisioned as there are no real blocks available to support when you have overcommitted your shared storage pool.

To overcome this, you can use the *alert* functionality of the shared storage pool to let the system administrator know in case the hard usage of the shared storage pool crosses the threshold limit.

Listing 33. On VIOS A

```
$ alert -set -clustername demo1 -spname demosp -type threshold -value 75
$ Pool freespace is 47 percent.
$ alert -list -clustername demo1 -spname demosp -type threshold
PoolName:      demosp
PoolID:       0000000009793823000000050E9B08C
ThresholdPercent: 75
$ alert -unset -clustername demo1 -spname demosp -type threshold
$
```

After looking into the Listing 31 with two LUs mapped to a client, there might be a question in mind on how to take a snapshot at the same time when multiple LU's are provided to a client. In storage, we refer to this as a *consistency group* where snapshots are created for a group of volumes at the same time to maintain consistency. This is also possible in shared storage pools.

To explain this, I am creating a single snap of the two LUs allocated to lparA on VIOS A.

Listing 34. On VIOS A

```
$ snapshot -clustername demo1 -create datavgA_snap -spname demosp -lu lparA_lu1 lparA_lu2
datavgA_snap
$

$ lssp -clustername demo1 -sp demosp -bd
Lu Name          Size(mb)    ProvisionType    Lu Udid
lparA_lu1        20480       THIN             687f8420bbeee7a5264ce2c6e83d3e66
Snapshot
datavgA_snap

lparA_lu2        51200       THICK            0ceaf03105d97f45ef4c595968f61cf7
Snapshot
datavgA_snap

$
```

This way, we can ensure consistency across multiple disks by creating snapshots at the same time for consistency.

Hints

- A LU, either thick or thin provisioned, cannot be resized. You should only add new LUs to the client. May be the future releases might have an option, but IBM support confirmed that it is not supported.
- Disks that are part of the shared storage pools can still be listed as not part of any VG (none) in the *lspv* output. (Refer to Listing 6.)
- You can also use the VIOS Shared Storage Pool with a dual-VIOS setup to have redundancy for your clients with default AIX MPIO. It is an advantage over the traditional file-backed or LV-backed storage offering redundancy.
- You can use only VSCSI with shared storage pools and NPIV cannot be used as of now. No idea if it will be supported in the future.
- You can use the LU ID instead of the LU name as duplication of LU names is allowed in a shared storage pool.
- I have not fully tested the *alert* functionality, but it will be a good feature if it works properly.

Resources

- Power System information center

Summary

In this article, I have detailed the features of the shared storage pool and how best you can use it in your infrastructure. I felt that using a shared storage pool makes life easier on environments where you do not have heavy data workload systems. I have not performed any benchmarks for it. Apply it wherever it best suits your environment. This article is not intended to replace any official document, but can act as a quick-start guide for system administrators who would like to explore or test the shared storage pool concept.