Web Exclusive

Implementing a Simple Spectrum Scale/GPFS Single Node Cluster

October 2017 | by Jaqui Lynch

In April 2014, we looked at implementing a three node GPFS (IBM General Parallel File System) cluster on GPFS 3.5. Since then GPFS has been through many changes including a name change to IBM Spectrum Scale. I will refer to it as SCALE throughout this article. SCALE has now added support for additional protocols such as AD (active directory) and samba. In this article we will cover installing SCALE 4.3.2 on a single node on AIX 7.2.1. Installing and configuring samba within that node to work with AD and SCALE will be the subject of another article. In general, SCALE is setup as at least a 2 node and preferably a multi-node cluster. In this case we are doing a single node to get around many of the JFS2 restrictions on data size, queuing and performance.

Cluster Description

For this installation there will be one AIX node that acts as the SCALE system as well as the samba server. Although this is a cluster of one, it can be easily extended to add additional servers and clients. The intent is to migrate several 20TB JFS2 filesystems into SCALE to provide a single high-performing namespace and the allow users to update the files using samba from their desktops. This makes the changeover transparent to the users. For this test we will be using hdisk4 through hdisk40. The LPAR name or node being used is gpfsnode1.mydomain.com and this is a bare metal system with no VIO servers. This could be implemented in a virtualized environment very easily.

Implementation

The first step is to install AIX and to make sure we have a clean AIX installation. AIX was installed at 7.2 tl01 sp2 – running ‘oslevel -s’ shows: 7200-01-02-1717 Additionally, lppchk -v and various other commands were run to make sure there are no missing filesets.

The fiber adapter settings were changed to ensure there is no queuing on those adapters:

```
chdev -l fcs0 -a max_xfer_size=0x200000 -a num_cmd_elems=2048 -P
chdev -l fcs1 -a max_xfer_size=0x200000 -a num_cmd_elems=2048 -P
```

You should also check that the hdisks have their queue_depth and reserve_policy set and that they have a PVID on them, otherwise you need to set the values using chdev.

```
chdev -l hdisk4 -a pv=yes
chdev -l hdisk4 -a queue_depth=64 -a reserve_policy=no_reserve -P
```

Perform this on every hdisk that will be in the cluster (in our case hdisk4 through hdisk40). The changes to settings on the fibre adapters and the hdisks are dependent on what your storage supplier can support so you may need to check with them.

In a multiple node system, occasionally the hdisks show up in a different order so it may be helpful to rename them on the nodes so that they all match. You can do this using rendev. Don't use rendev to rename the hdisks to anything other than hdisk – SCALE does not recognize the disk type when you run mmcrnsd. In our case we only have one system so out of order disks is not an issue.

You should also check the following settings:

```
/etc/security/limits
        Look at fsize and nofiles – I usually set them to fsize=-1
and nofiles=20000 or nofiles=-1
        This allows for large file sizes and lots of files
/etc/environment
        Add /usr/lpp/mmfs/bin to end of PATH
Add WCOLL=/usr/local/etc/gpfs-nodes.txt
```

Then: I normally have a /usr/local/ filesystem that I use for customizations. There is an etc directory in it that I use for configuration files so I put the SCALE configuration files in there. On gpfsnode1 create a file called /usr/local/etc/gpfs-nodes.txt and put in it a list (one per line) of the nodes in the cluster. At this point, you must decide whether or not you'll use fully qualified names. I used fully qualified names, which I also used for the hostname. The IP and hostname should be in /etc/hosts and should be resolvable before you start.

vi /usr/local/etc/gpfs-nodes.txt
gpfsnode1.mydomain.com

The above is the fully qualified name of the node. If you are not using fully qualified names then you would just put gpfsnode1 or whatever your node name is.

At this point, reboot the SCALE LPAR. After the reboot, check that you see all the disks and that they have the correct PVIDs, etc.

Installing the Software

The next step is to install the SCALE software and any fixes – in my case, SCALE 4.2.3.0 and then an upgrade to 4.2.3.4. The installation files were delivered as two .tgz files that I downloaded. They were uploaded to /usr/local/soft and unzipped and untarred (you will need gzip to be able to unzip them). The files provided were:

```
Spectrum_Scale_Dat_Management-4.2.3.0-ppc64-AIX-install.tgz
Spectrum_Scale_Dat_Management-4.2.3.4-ppc64-AIX-update.tgz
Each was untarred into a different directory
/usr/local/soft/spectrumscale/scale4230
/usr/local/soft/spectrumscale/scale4234
```

The install was then very simple:

```
cd /usr/local/soft/spectrumscale/scale4230
smitty install and install everything
cd /usr/local/soft/spectrumscale/scale4234
smitty update_all
```

If you have not already installed dsh you should also install it as this will let you talk to other nodes in the system if you add them later. Go to the AIX 7.2 base and install the dsm.dsh fileset. It will also install the dsm.core and Java7_64.jre 7.0.0.370 filesets. You should then go to your update directory and update them to the same level as the operating system (TL and SP). I would also recommend using flrtvc to find the latest security ifixes and efixes along with the latest java, openssl and openssh levels and they should be downloaded and installed so the system has all known security patches installed.

You can now check your levels:

```
lslpp -l | grep ava
  Java7_64.jre             7.0.0.610  COMMITTED  Java SDK 64-bit Java Runtime
  Java7_64.jre             7.0.0.610  COMMITTED  Java SDK 64-bit Java Runtime

lslpp -l | grep dsm
  dsm.core                   7.2.1.1  COMMITTED  Distributed Systems Management
  dsm.dsh                    7.2.0.0  COMMITTED  Distributed Systems Management
  dsm.core                   7.2.1.0  COMMITTED  Distributed Systems Management

lslpp -l | grep gpfs
  gpfs.adv                   4.2.3.4  COMMITTED  GPFS Advanced Features
  gpfs.base                  4.2.3.4  COMMITTED  GPFS File Manager
  gpfs.crypto                4.2.3.4  COMMITTED  GPFS Cryptographic Subsystem
  gpfs.ext                   4.2.3.4  COMMITTED  GPFS Extended Features
  gpfs.gskit               8.0.50.75  COMMITTED  GPFS GSKit Cryptography
  gpfs.license.dm            4.2.3.0  COMMITTED  GPFS Data Management Edition
  gpfs.msg.en_US             4.2.3.3  COMMITTED  GPFS Server Messages - U.S.
  gpfs.base                  4.2.3.4  COMMITTED  GPFS File Manager
  gpfs.docs.data             4.2.3.4  COMMITTED  GPFS Server Manpages and

oslevel -s
7200-01-02-1717
```

If ‘lppchk -v’ comes back clean and ‘instfix -I | grep ML’ shows no missing filesets then I normally run bosboot, set the bootlist and reboot. After that I take a backup prior to installing the cluster.

CLUSTER INSTALLATION

Create /usr/local/etc/gpfsnodes.txt - it should contain: gpfsnode1.mydomain.com This is separate to the gpfs-nodes.txt file you already created – use whatever node name you put in that file.

```
Also on gpfsnode1 create /usr/local/etc/gpfsdisks.txt - it should contain the names
of all the disks to be used by SCALE:
hdisk4
hdisk5
………
hdisk40
```

On LPAR gpfsnode1, create a file /usr/local/etc/gpfs-nodesinit.txt that contains: gpfsnode1.mydomain.com:quorum-manager

Create the NSD stanza to use for disks /usr/local/etc/gpfs-nsdstanza.txt %nsd: nsd=nsdhdisk4 device=/dev/hdisk4 usage=dataAndMetadata pool=system Do this for all of hdisk4 through hdisk40

```
Check for sufficient swap/page space
lsps -a
Page Space      Physical Volume   Volume Group          Size
%Used   Active    Auto    Type   Chksum
hd6                             hdisk0
```

```
rootvg                  20480MB     0       yes         yes      lv       0
```

Now You Can Set Up SCALE:

First create the cluster with gpfsnode1 as primary and no secondary.

```
mmcrcluster -C CLGPFS1 -p gpfsnode1.mydomain.com -r
/usr/bin/ssh -R /usr/bin/scp -N /usr/local/etc/gpfs-nodesinit.txt -A
```

mmcrcluster: Performing preliminary node verification ... lots of messages seen here mmcrcluster: Finalizing the cluster data structures ... mmcrcluster: Command successfully completed mmcrcluster: Warning: Not all nodes have proper GPFS license designations. Use the mmchlicense command to designate licenses as needed.

Now you must accept the licenses.

mmchlicense server --accept -N gpfsnode1.mydomain.com

```
You should now see:
The following nodes will be designated as possessing server licenses:
        gpfsnode1.mydomain.com
mmchlicense: Command successfully completed
```

I normally check the cluster at this point using mmlscluster and mmlsconfig.

```
#mmlscluster

GPFS cluster information
========================
  GPFS cluster name:         CLGPFS1.mydomain.com
  GPFS cluster id:           2513740808193740764
  GPFS UID domain:           CLGPFS1.mydomain.com
  Remote shell command:      /usr/bin/ssh
  Remote file copy command:  /usr/bin/scp
  Repository type:           CCR

 Node  Daemon node name        IP address   Admin node name         Designation
--------------------------------------------------------------------------------
1       gpfsnode1.mydomain.com  10.0.117.57  gpfsnode1.mydomain.com  quorum-manager

#mmlsconfig
```

Configuration data for cluster CLGPFS1.mydomain.com:

```
----------------------------------------------------
clusterName CLGPFS1.mydomain.com
clusterId 2513740808193740764
autoload yes
dmapiFileHandleSize 32
minReleaseLevel 4.2.3.0
ccrEnabled yes
cipherList AUTHONLY
```

```
adminMode central
```

File systems in cluster CLGPFS1.mydomain.com: -- (none)

Now create the NSDs

Run the following command to create the NSDs

```
mmcrnsd -F /usr/local/etc/gpfs-nsdstanza.txt
```

You will see a number of lines similar to:

```
mmcrnsd: Processing disk hdisk4
….. to hdisk40
```

This defines the NSDs and names them. I made the NSD names match the hdisk names. We set the stanza up above but each disk stanza will look something like:

```
%nsd:  nsd=nsdhdisk4 device=/dev/hdisk4 usage=dataAndMetadata pool=system
```

You can now run lspv and mmlspv to see how the disks are mapped. At this point, you can start SCALE on gpfsnode1 using mmstartup –a. mmlsnsd will show you the NSD mappings.

```
# lspv
hdisk4         00f660a7bf0cf27b                    nsdhdisk4
hdisk5         00f660a7bf0ef316                    nsdhdisk5
……. to hdisk40

# mmlspv
hdisk4 nsdhdisk4
hdisk5 nsdhdisk5
……. to hdisk40

#mmlsnsd
File system   Disk name    NSD servers
---------------------------------------------------------------------------
 (free disk)   nsdhdisk4   (directly attached)
 (free disk)   nsdhdisk5   (directly attached)
```

We want to change the disks so they can be shared later over the network in case we want to add clients who may be network attached rather than fibre attached to the disks:

```
mmchnsd "nsdhdisk4:gpfsnode1"
mmchnsd: Processing disk nsdhdisk4
Do hdisk4-40
```

Now mmlsnsd will look more like:

```
# mmlsnsd


 File system   Disk name    NSD servers
```

```
---------------------------------------------------------------------------
```

```
Now you can startup SCALE #mmstartup Tue Sep 26 12:47:58 EDT 2017: mmstartup:
Starting GPFS ... /tmp/mmfs has been created successfully.
```

Final Steps

Final steps include creating the filesystem.

Our filesystem will be gpfs0 with a client name of /gpfsfiles We are using the default blocksize of 512 with no replication -R2 says max of 2 replicas for data, -R2 says max of 2 data replicas We are only doing 1 replica (-r 1 and –m1)

```
mmcrfs gpfs0 -F /usr/local/etc/gpfs-nsdstanza.txt -B 512K -m1 -M2 -r 1 -R 2 -T /gpfsfiles
```

```
You should see a number of lines similar to:
The following disks of gpfs0 will be formatted on node gpfsnode1.mydomain.com:
    nsdhdisk4: size 256000 MB
    nsdhdisk5: size 256000 MB
…….
    nsdhdisk40: size 256000 MB
Formatting file system ...
Disks up to size 8.5 TB can be added to storage pool system.
Creating Inode File
  80 % complete on Tue Sep 26 12:53:12 2017
100 % complete on Tue Sep 26 12:53:14 2017
Creating Allocation Maps
Creating Log Files
Clearing Inode Allocation Map
Clearing Block Allocation Map
Formatting Allocation Map for storage pool system
Completed creation of file system /dev/gpfs0.
```

Now you can mount the filesystem using mmount

```
#mmmount all
Tue Sep 26 12:54:14 CDT 2017: mmmount: Mounting file systems ...
```

```
#df -g /gpfsfiles
Filesystem    GB blocks      Free %Used    Iused %Iused Mounted on
/dev/gpfs0      6000.00   5996.32    1%     4038     1% /gpfsfiles
```

```
#mmdf gpfs0
disk                disk size  failure holds    holds              free KB             free KB
name                    in KB    group metadata data        in full blocks        in fragments
--------------- ------------- -------- -------- ----- -------------------- -------------------
Disks in storage pool: system (Maximum disk size allowed is 8.5 TB)
nsdhdisk6           262144000       -1 yes      yes       261980672 (100%)          1264 ( 0%)
nsdhdisk7           262144000       -1 yes      yes       261982720 (100%)          1248 ( 0%)
……
nsdhdisk40           262144000       -1 yes      yes       261982720 (100%)          1248 ( 0%)
```

You will also see a pool total that shows the total storage for that filesystem along with a section that shows details on the inodes in use, free, allocated and the maximum inodes you can have.

Next steps:

```
At this point, you’re ready to test the cluster by adding data to
the filesystem and testing access. You can also use the following
commands to document your cluster:
mmlsconfig
mmgetstate -aLs
mmlsnsd
mmlscluster
mmdf gpfs0
```

At this point, your cluster is ready to go. You can add additional nodes or just use it as a single node cluster, depending on your needs.

A Simple Alternative

This is a fairly simple implementation for a specific use but it can be used as the foundation for your SCALE environment and allows a scale up solution for a user who has huge filesystems and who needs the latency reduction you get with SCALE. If it becomes necessary to add additional nodes in the future that is easy to do. The next steps in our case are to add the samba and AD integration and to update some of the SCALE tunables, but as of right now we have a fully functional, well performing Spectrum Scale cluster. If you are having issues around JFS2 performance or scalabilty with respect to the size or number of files, or if you need to server out files to multiple systems while maintaining performance, then I would recommend getting a trial of Spectrum Scale. IBM offers the ability to use an Intel VM they provide or to trial it on your own systems. Spectrum Scale is supported on multiple operating systems including Windows, Linux, Linux on Power, Linux on Z and AIX. All of these can be in the cluster at the same time as long as they meet the required levels which can be found in the FAQ (frequently asked questions) document from IBM. The FAQ also provides documentation on the architectural limits on Spectrum Scale which are significantly higher than JFS2 filesystems.

RESOURCES VIDEO SOLUTIONS EDITION BLOGS WEBINARS SUBSCRIBE ABOUT US

Systems

Connect With Us:

Magazine Archives

IBM i | LINUX ON POWER | MAINFRAME | POWER

Search

AIX ADMINISTRATOR TRENDS CASE STUDIES TIPS & TECHNIQUES STORAGE PRODUCT NEWS

For more information:

< Return to main article

Print Email

1. Implementing a three node redundant GPFS Cluster
2. Building a two-node IBM GPFS cluster on IBM AIX
3. Spectrum Scale Documentation Home Page - links to Admin, etc., guides
4. Spectrum Scale 4.2.3 Commands
5. Spectrum Scale mmbackup
6. Spectrum Scale FAQ
7. Links to redbooks and papers on Spectrum Scale
8. FLRTVC ï Fix Level Recovery Tool Vulnerability Checker

< Return to main article

READ THE CURRENT ISSUE: DIGITAL | ONLINE | eNEWSLETTER

AIX | IBM i | LINUX ON POWER | MAINFRAME | POWER |

Connect With Us:

Homepage About Us Contact Us Subscriptions Editorial Calendar

Advertise With Us Reprints Privacy Policy Terms of Service Sitemap