Redpaper

Michael Hennecke
David Lebutsch
Stefan Schleipen

GPFS in the Cloud: Storage Virtualization with NPIV on IBM System p and IBM System Storage DS5300

The IBM® General Parallel Filesystem (GPFS™) is known for its performance, scalability, availability, and manageability features. It is the file system used in many High Performance Computing (HPC) solutions, scaling to thousands of servers and serving multi-petabyte file systems. GPFS is also used in many non-HPC areas, such as database solutions, but typically has been deployed at a much smaller scale in those environments both in number of servers and size of the underlying storage subsystems.

This IBM Redpaper™ publication focuses on the use of GPFS in conjunction with server virtualization, where logical partitions (LPARs) access the external storage through a relatively new Fibre Channel storage virtualization technique called N_Port ID Virtualization (NPIV). From the GPFS perspective, this case is positioned between the two above extremes: Virtualization might increase the number of GPFS nodes to the scales typically seen in HPC clusters while the number of physical servers is still relatively small (as in other non-HPC environments).

Our goal is to use GPFS and NPIV as the underlying storage infrastructure for Enterprise Content Management (ECM) solutions, which always include storage as a central component. An excellent overview on ECM and storage in general can be found in the IBM Redbooks® publication *IBM Enterprise Content Management and System Storage Solutions: Working Together*, SG24-7558.

With the ever-growing amount of unstructured data, the size of the storage hardware infrastructure required for ECM solutions is now becoming comparable to storage deployments in HPC environments, so the typical HPC "scale-out" approach that uses midrange storage subsystems such as the IBM System Storage® DS5300 is beneficial for ECM as well. On the other hand, deploying ECM solutions in a cloud environment naturally introduces server virtualization, and NPIV is an effective means to provide access to shared storage with GPFS as the file system and manageability layer.

This paper describes a proof of concept in which we have put these individual components together to implement a scalable GPFS infrastructure in a virtualized environment. It is organized as follows:

- "Architectural decisions for GPFS and storage connectivity" on page 2 discusses the advantages, disadvantages and constraints of various connectivity options.
- "Hardware components for the proof of concept" on page 5 describes the hardware used for this proof of concept.
- "Storage virtualization planning" on page 7 focuses on the importance of planning in virtualization environments and recommends some best practices.
- "Configuring the infrastructure" on page 13 is the main part of this paper. In this section, we walk through the initial configuration of all the components. This includes the SAN switches, the DS5300 storage subsystems, IBM System p® partitioning using the hardware management console (HMC), NPIV setup (on the VIO servers, the client LPARs, and the related DS5300 configuration steps), and the GPFS configuration.
- "Provisioning scenarios" on page 46 outlines which of the configuration steps are required when provisioning more resources. We also discuss online *removal* of resources, an action that might be necessary when old storage subsystems are replaced by newer technology (while keeping the data online). This capability is an important manageability feature of GPFS.

Our primary goal is to set up the infrastructure to use GPFS in Enterprise Content Management cloud solutions that make use of server virtualization. However, this work is equally applicable to other solution areas where GPFS is deployed in a cloud environment. In "Summary and outlook" on page 47, we give a short outlook of how we plan to use the advanced GPFS features to build ECM appliances in cloud environments.

Architectural decisions for GPFS and storage connectivity

Two main architectural decisions govern the overall system design of a solution based on GPFS and DS5300 midrange storage subsystems: the choice of GPFS cluster connectivity options, and the Fibre Channel connectivity and multipathing between the DS5300 storage subsystems and the GPFS nodes.

GPFS cluster connectivity

GPFS requires shared disk access from all nodes in the GPFS cluster to all disks used by the GPFS file systems. This can be achieved by two methods: either by direct disk access through a SCSI or other block-level protocol over a SAN, or through network-based block I/O using the GPFS Network Shared Disk (NSD) layer over a TCP/IP network.

- SAN attached storage model

  This is the most basic setup, where all GPFS nodes are on a shared SAN with the storage controllers and can directly and concurrently access all disks (LUNs). Figure 1 on page 3 shows this setup on the left. This model is often used for small clusters (less than 100 nodes). It provides excellent bandwidth to the disk storage from all nodes, and high availability by using Fibre Channel multipathing (for example, 2 * 8 Gbps for two FC connections per node).

Note: The SAN attached storage model usually requires that all nodes have the same operating system, for example, all of them are AIX® nodes or all Linux® nodes. This limitation is imposed by the disk storage subsystems, not by GPFS itself, but it still is an important aspect to consider when choosing the GPFS cluster connectivity.

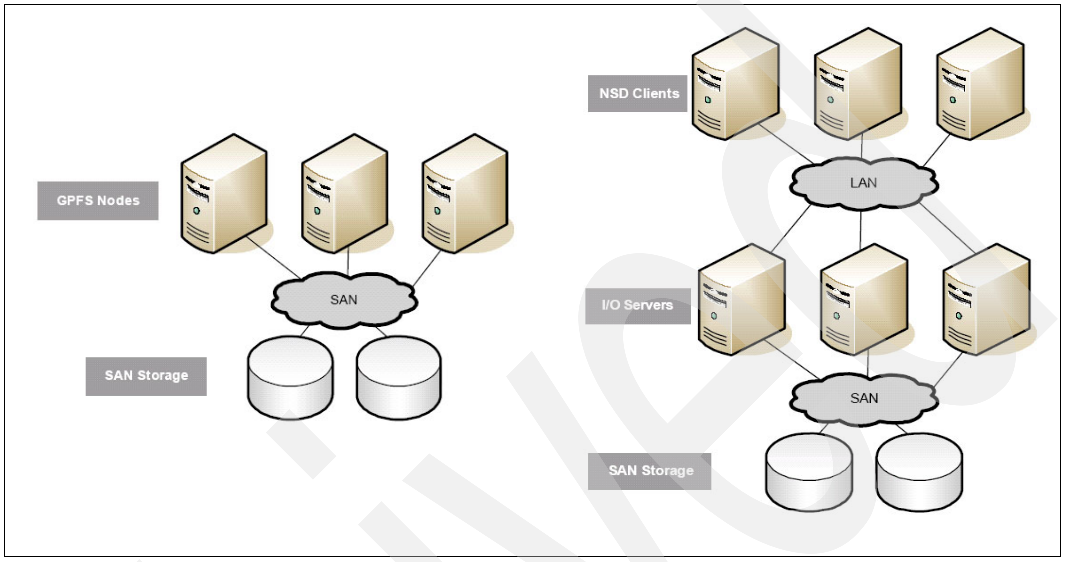


Figure 1 GPFS tiers (nodes and NSD clients)

- NSD server model (network-based block I/O)

  For GPFS clusters with hundreds or thousands of nodes, direct SAN connectivity is generally undesirable for cost and manageability/complexity reasons.

  To provide shared disk access in such environments, GPFS provides the Network Shared Disk (NSD) layer, which transparently provides shared access to the disks from all GPFS nodes: A subset of the GPFS nodes is directly attached or SAN attached to the storage, and those nodes act as NSD servers. All other GPFS nodes are NSD clients and access the disks through the NSD servers, using a TCP/IP network. Figure 1 shows this setup on the right.

  The NSD server model provides excellent scalability, and high availability is achieved by defining multiple NSD servers for the disks. Performance depends on the type of the deployed TCP/IP network: For 1 GbE Ethernet networks, performance will be generally inferior to direct SAN attachment, while for 10 GbE Ethernet or InfiniBand networks, it will be comparable to direct SAN attachment. In addition, link aggregation / channel bonding can be used to utilize multiple network links.

  In the High Performance Computing (HPC) segment, clusters normally have a high-speed interconnect for application level communication, which can be shared for the GPFS/NSD traffic, providing a cost-efficient and high performance solution.

Note: As GPFS supports a mix of the SAN-attached storage model and the NSD server model, it is still possible to add nodes with different operating systems into an otherwise homogeneous-OS environment with direct SAN attachment, by designating some of the homogeneous-OS nodes as NSD servers.

In a cloud environment, the nodes are (potentially many) logical partitions (LPARs), virtualized on a few physical servers. Ethernet and Fibre Channel resources can be virtualized, so in general the per LPAR cost for both connectivity modes is reduced because the physical resources are shared by many LPARs.

The following factors should be considered when selecting the GPFS cluster connectivity for a cloud environment:

- Factors in favor of the SAN attached storage model:
  - The NSD server LPARs needs significant CPU (and memory) resources, as they move all the GPFS traffic between the storage subsystems and the (physical or virtual) Ethernet layer. For that reason, they are usually dedicated partitions, which are not used for applications, adding to the overall CPU and memory requirements.
  - TCP/IP over System p Integrated Virtual Ethernet (IVE) or Shared Ethernet Adapters (SEA) is usually less efficient than Fibre Channel virtualization, consuming more resources on the VIO servers (SEA) and IBM POWER® hypervisor (IVE and SEA).
  - With multiple physical servers in the GPFS cluster, external connectivity with adequate performance is needed *between* the servers, in addition to the virtualization layer *within* the physical servers. It is usually more cost effective to provide this connectivity at high performance using a SAN, rather than through Ethernet, where it may be necessary to deploy expensive 10 GbE infrastructure to reach comparable per-node performance.
- Factors in favor of the NSD server model:
  - Ethernet Virtualization (both IVE and SEA) is more mature than Fibre Channel Virtualization using N_Port ID Virtualization (NPIV).
  - The NSD server model is proven to scale to a much higher number of nodes than the shared SAN access model.
  - In the SAN attached storage model, when many LPARs directly access many LUNs on a storage subsystem, the limited *command queue* of the storage subsystem might become a bottleneck. In the NSD server model, only a few NSD server LPARs will issue SCSI commands to the storage subsystem, so this is generally not a concern.
  - Current disk subsystems usually do not support concurrent access to the same LUN from different operating systems (for example, AIX and Linux), so heterogeneous GPFS clusters require NSD servers.
  - Client LPAR deployment in the NSD server model only involves TCP/IP setup, not Fibre Channel setup. For the SAN attachment model, both TCP/IP and Fibre Channel setup is needed.

A number of resource-related factors favor the SAN attached storage model. In contrast, many of the factors in favor of the NSD server model are actually new technology risks of the SAN attached storage model, caused by the lack of experience with this technology in a GPFS and cloud environment. The primary goal of this paper is to close this gap and to explore N_Port ID Virtualization (NPIV) on System p servers for the GPFS SAN attached storage model.

DS5300 Fibre Channel connectivity and multipathing

The DS5300 midrange storage subsystems are an excellent choice for cost-effective, high performance “scale-out” storage solutions with GPFS. As storage needs grow beyond the capabilities of a single midrange storage system, more storage subsystems are added to the solution rather than adding more components to a high-end storage subsystem, such as the IBM System Storage DS8000® series.

When comparing a DS5300 based solution to high-end storage subsystems, two aspects of the midrange series are important, because they govern the connectivity between the storage subsystems and the System p LPARs, which access the LUNs on the storage subsystems:

- There is a maximum of *two* DS5300 partitions per AIX host.

  While the DS5300 supports a large number of storage partitions, a single AIX host may only be a member of a maximum of two DS5300 partitions. In GPFS configurations, we normally use storage partitions to increase the performance: By assigning half of the LUNs to one storage partition and the other half of the LUNs to a second storage partition, we can utilize two Fibre Channel connections between the DS5300 and the host, effectively doubling the performance.

- There is a maximum of *two* paths per disk.

  The multipath device driver used with DS5300 is the default AIX MPIO driver. This driver supports only two paths per disk, and uses an active-passive mode of operation. For GPFS configurations, this is not a limiting factor: As the access patterns to the disks are regular (I/O is striped across all LUNs in the file system), load balancing across all FC paths can be achieved by balancing primary ownership of the LUNs across the “A” and “B” controllers of the DS5300 storage subsystems.

  Note: In principle, other multipath drivers such as the SDDPCM driver used with the high-end storage subsystems are supported with the DS5300. However, SDDPCM for DS5300 is currently not supported in VIOS environments, and, in our opinion, in a GPFS environment, its benefits (more than two paths, better load balancing, and higher resilience against path failures) are not significant enough to offset the substantial new technology risk that the use of this driver with DS5300 would introduce.

These two factors combined imply that our Fibre Channel layout includes *four* FC paths between a single DS5300 and an AIX LPAR. For perfect balance across all four FC paths, LUNs are configured into GPFS file systems in groups of four: Two LUNs will be mapped into each of the two storage partitions, and one LUN in each storage partition has the “A” controller as its primary controller while the other LUN in that storage partition has the “B” controller as its primary controller.

Note that we are using SAN boot disks for rootvg. In the proof of concept, these are served by the same single DS5300 that also holds the GPFS LUNs. As we want to separate GPFS LUNs from the boot LUNs on the FC layer, the limitation of two storage partitions per host forces us to use VSCSI (disk) virtualization for the boot LUNs rather than NPIV (FC adapter) virtualization, as we cannot use a third storage partition for the boot LUNs on the client LPARs. By using VSCSI for boot LUNs and NPIV for GPFS LUNs, the VIO servers only see the boot LUNs (and their associated third storage partition), while the client LPARs only see the GPFS LUNs (and their associated first and second storage partition). In larger production setups, the boot LUNs may be served by a separate storage controller, and then NPIV could also be used for the boot LUNs (which is generally preferable over VSCSI).

Hardware components for the proof of concept

The minimum hardware required to build a solution is composed of one DS5300 storage subsystem with disk storage, one NPIV-capable SAN switch, one System p server with NPIV-capable FCS adapters (two ports to attach to the DS5300 “A” and “B” controllers, four ports to utilize two DS5300 storage partitions per AIX LPAR), and an accompanying HMC for partitioning the server and virtualizing the physical FCS adapter ports. In a production environment, redundant SAN switches would be configured for additional resilience against

FC path failures. Typically, one SAN switch hosts the ports connected to the DS5300 “A” controller, while the second SAN switch hosts the “B” ports. Multiple servers may be configured to have the ability to move services across physical servers. Servers and storage subsystems are added as requirements grow.

In our proof of concept (PoC), we use the following hardware configuration (Figure 2 on page 7) with physical wiring and connections in the virtualization layer:

- One DS5300 with eight host ports (four used for GPFS, two used for SAN boot, and two unused)
- Three EXP5000 storage expansions (two with 16 SATA disks and one with 16 FC disks)
- One SAN24B SAN switch with 24 ports (14 ports used)

  Note: A single SAN switch hosts the port groups for both the “A” and the “B” FC paths. While this makes the SAN switch a single point of failure, this is acceptable in a test environment and does not differ functionally from a fully redundant configuration.

- One System p Model 570 server, fully configured with four central processor complexes (CPCs):
  - One disk enclosure with local SAS disks per CPC
  - One 4-port IVE Ethernet adapter per CPC
  - One NPIV-capable, dual-port 8 Gbps FC adapter per CPC
- One HMC
- Ethernet ports are provided on the data center’s network as required.

The System p server used in the PoC is described in *IBM System p 570 Technical Overview and Introduction*, REDP-4405. References for the other hardware components can be found in the appropriate subsections of “Configuring the infrastructure” on page 13.

For planning and configuration purposes, it is important to document the physical location of the hardware components, including serial numbers, adapter and port locations, Fibre Channel WWPN numbers, and other information that will be needed later. Example 1 shows our documentation of the Fibre Channel ports in our IBM POWER6® server, including the physical locations (serial numbers of the four CPCs/drawers of our System p Model 570 server, PCI adapter slot, and port ID on the dual-port adapters), the FC WWNs of the ports, and their intended usage (VIOS name and fcs*X* device name, as well as the intended SAN zone).

Example 1   Fibre Channel port documentation for the PoC hardware

```
WWN                CPC      SLOT  ID/PORT  LPAR    DEVICE  ZONE

00:00:C9:95:AE:1A  DQDOD4H  C1    0  T1    p01v01  fcs0    VIOSA
00:00:C9:95:AE:1B  DQDOD4H  C1    1  T2    p01v01  fcs1    VIOSB

00:00:C9:94:32:A2  DQDOD3X  C1    0  T1    p01v01  fcs2    NPIV1A
00:00:C9:94:32:A3  DQDOD3X  C1    1  T2    p01v01  fcs3    NPIV2A

00:00:C9:95:AD:86  DQDOD3Y  C1    0  T1    p01v02  fcs0    VIOSA
00:00:C9:95:AD:87  DQDOD3Y  C1    1  T2    p01v02  fcs1    VIOSB

00:00:C9:95:B0:9A  DQDOD3W  C1    0  T1    p01v02  fcs2    NPIV1B
00:00:C9:95:B0:9B  DQDOD3W  C1    1  T2    p01v02  fcs3    NPIV2B
```

Storage virtualization planning

In this section, we describe the storage connectivity (both physical and virtual) that we are going to use to implement GPFS on the hardware described in "Hardware components for the proof of concept" on page 5. Figure 2 shows the overall architecture, including the three main hardware components:

1. The DS5300 storage subsystem at the bottom, labeled "DS01"
2. The SAN switch in the middle, labeled "SAN01"
3. The System p server in the dotted box at the top, which in turn shows three LPARs:
   - Two VIO servers, labeled "P01V01" and "P01V02"
   - One of the (potentially many) client LPARs hosted on this server, labeled "LS01"

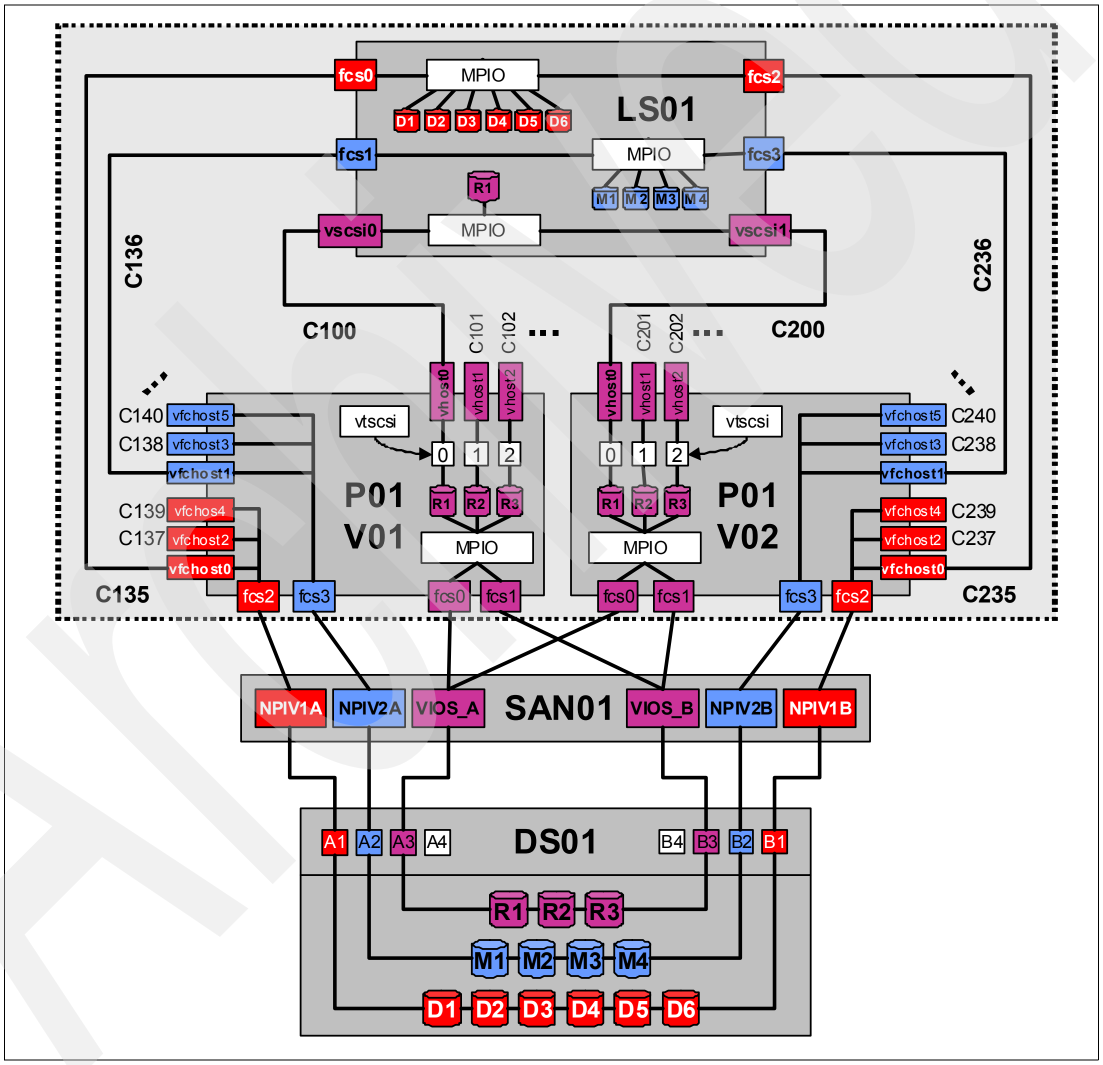


Figure 2 Storage connectivity, virtualization, and multipathing

We have color-coded the figure to indicate the three different usage areas:

- Red and blue color is used to indicate two groups of NPIV connections to the client LPARs. The DS5300 supports a maximum of two host groups per host, and we want to fully use this capacity to maximize the number of FC connections in an LPAR to achieve maximum performance. Load is balanced across these two groups of FC connections by statically distributing the LUNs on the DS5300 across the two host groups.

Note: In our PoC, we are using separate LUNs for GPFS *data* and GPFS *metadata*. We have distributed the GPFS LUNs to the two NPIV host groups so that all GPFS *data* disks are in the NPIV1 group and all GPFS *metadata* disks are in the NPIV2 group. Depending on your workload, you may want to distribute the LUNs differently. For example, you may put data and metadata on each disk, or put 50% of the data disks *and* 50% of the metadata disks into the NPIV1 group, and the remaining disks into the NPIV2 group.

- Violet color is used to indicate a third group of FC connections, which are used to provide SAN boot volumes to the client LPARs through the VIO servers.

The FC connectivity within these three groups is described in the following sections.

GPFS data disk connectivity (red)

Red color is used to indicate the GPFS data disks. Their connectivity is set up so that all "A" paths go through the VIO server P01V01 and all "B" paths go through VIO server P01V02:

- We use six LUNs on the DS5300, labeled "D1" to "D6" for *data*.
- Those LUNs are mapped into NPIV1 hostgroups, which use the DS5300 host ports "A1" and "B1"; for load balancing, we typically assign the "A" controller as the primary owner of LUNs D1, D3, and D5, and assign the "B" controller as the primary owner of LUNs D2, D4, and D6.
- There is a *physical* FC connection from the DS5300 host port "A1" to the FC adapter port "fcs3" on the VIO server P01V01, through a SAN zone "NPIV1A" on the SAN switch.
- From the VIO server P01V01, there is a *virtual* FC connection to the client LPAR LS01 (and to all other LPARs), using N_Port ID Virtualization (NPIV). This includes several components:
  - The POWER hypervisor (PHYP) creates a virtual server slot for the VIO server and a virtual client slot for the client LPAR. Those slots are numbered C*xxx*, and we strongly recommend that you plan your virtualization environment so that the same slot number on a VIO server and on a client LPAR is identifying the same virtual FC connection. In our example, we are using slot C136 for the connection between P01V01 and LS01.
  - A virtual server device vfchost1 is created on P01V01, and is associated with the physical FC adapter port fcs3.
  - The connection between the virtual server slot and the virtual client slot is made.
  - The client LPAR LS01 sees an FC adapter port fcs0.
- There is a *physical* FC connection from the DS5300 host port "B1" to the FC adapter port "fcs3" on the VIO server P01V02, through a SAN zone "NPIV1B" on the SAN switch.
- From the VIO server P01V02, there is a *virtual* FC connection to the client LPAR LS01, using N_Port ID Virtualization (NPIV). This is set up in the same way as for the "A" path described above, using slot number C236 to connect vfchost1 on VIOS P01V02 with fcs2 on LPAR LS01.

- On the client LPAR LS01, the AIX MPIO multipath driver sees two paths (through fcs0 and fcs2) to the LUNs D1 through D6, and presents those to the upper layers as six AIX hdisks.
- Note that the VIO servers themselves *do not* see the LUNs "D1" to "D6".

GPFS metadata disk connectivity (blue)

Blue color is used to indicate the GPFS metadata disks. Their connectivity is equivalent to the connectivity of the GPFS data disks. We use four LUNs labeled "M1" to "M4", which are connected through the DS5300 host ports "A2" and "B2" to the two VIO servers' "fcs2" ports, and from there through virtual slots C135 (for the "A" path) and C235 (for the "B" path) to the client LPAR. The client LPAR LS01 again sees two paths to those LUNs (through its fcs1 and fcs3 FC adapter ports), and presents four AIX hdisks to the operating system.

Note: As mentioned above, we have distributed the GPFS LUNs across these two identical NPIV groups simply to get more aggregate performance between the DS5300 and the client LPAR. The same setup also works if all LUNs are mapped into only one DS5300 host group, and only two FC connections per client LPARs are set up.

SAN boot disk connectivity (violet)

Violet color is used to indicate the SAN boot volumes. We do not cover details about how these volumes are set up, but include them in the overall architecture to illustrate the difference between NPIV storage virtualization and the traditional virtual SCSI (VSCSI) storage virtualization that we are using for the SAN boot.

Note that the connectivity of the SAN boot LUNs is different from the NPIV setup: When using VSCSI virtualization, the VIO servers *do* see the LUNs, so we have added another layer of AIX MPIO multipathing by connecting both VIO servers to both the "A" and the "B" port of the DS5300:

- We are using three LUNs on the DS5300 (labeled "R1" to "R3" for *rootvg*), which would be used to SAN-boot three LPARs.
- Those LUNs are connected through the DS5300 host ports "A3" and "B3" to the SAN switch zones VIOS_A and VIOS_B.
- As mentioned above, both VIO servers connect to both the "A" and the "B" zone, using the FC adapter ports fcs0 and fcs1 on both VIO servers.
- The VIO servers' AIX MPIO layer sees two paths to the LUNs "R1" to "R3", and presents them to the VIOS operating system as three AIX hdisks.
- On the VIO server P01V01, the hdisk that is used to SAN-boot LS01 is exported as vtscsi0, through a virtual connection between the server device vhost0 and the client device vscsi0. The VIO server P01V01 uses slot C100 for this connection.
- On the VIO server P01V02, the hdisk that is used to SAN-boot LS01 is exported as vtscsi0, through a virtual connection between the server device vhost0 and the client device vscsi1. The VIO server P01V02 uses slot C200 for this connection.
- The client LPAR's AIX MPIO driver sees two paths to the hdisk (through vscsi0 and vscsi1), and presents this disk as a single AIX rootvg disk.

It is also possible to use NPIV to virtualize the SAN boot LUNs. In our case, we did not use this approach, because we want to demonstrate the difference between the two methods, separate GPFS traffic from SAN boot traffic, and use the maximum of two DS5300 host groups per node for GPFS traffic. Using VSCSI for the SAN boot conveniently circumvents the hostgroup limitation, as the SAN boot LUNs are mapped to the VIO servers, not to the client LPARs.

Note: In a production environment, the SAN boot LUNs could be served by a different storage controller, which would enable us to separate SAN boot LUNs from GPFS LUNs on the SAN fabric *and* use NPIV for the SAN boot LUNs and for GPFS.

The value of NPIV for GPFS

It is important to note that in the VSCSI case, all the LUNs are visible at the VIO servers and need to be individually "virtualized" and made available to the client LPARs. This setup is manageable for the SAN boot LUNs, where there is a small total number of LUNs and each client LPAR needs access to exactly one of those LUNs.

In contrast, the LUNs virtualized through NPIV are *not* visible at the VIO servers: NPIV virtualizes the whole FC adapter, and the LUNs are mapped to the WWN of the virtual client FC adapter rather than to the WWN of the physical FC adapter in the VIO server that is associated with these virtual FC adapters.

As GPFS requires all its LUNs to be visible at all nodes in the cluster, and there may be hundreds or thousands of LUNs in a GPFS file system, this is a huge manageability advantage. Table 1 shows the amount of resources needed for a VSCSI solution and for an equivalent NPIV solution (devices on the VIO servers, devices on the client LPARs, and the virtual slots on the PHYP hypervisor to link the two). While the number of resources for a VSCSI solution would be (#LPARs * #LUNs * #VIOSs), for NPIV, the number of required resources is only (#LPARs * #VIOSs), which is a huge difference for large GPFS solutions with many LUNs.

Table 1 Resource requirements for NPIV virtualization versus VSCSI virtualization (two VIO servers)

Number of LPARs	**Number of GPFS LUNs**	**NPIV resources (VIOS vfchost, LPAR fcs, PHYP C*xxx*)**	**VSCSI resources (VIOS vtscsi,vhost, LPAR vscsi, PHYP C*xxx*)**
3	10	6	60
3	48	6	288
3	240	6	1.440
3	480	6	2.880
32	10	64	640
32	48	64	3.072
32	240	64	15.360
32	480	64	30.720

Here we use three LPARs and 10 LUNs, as this is the number of LUNs shown in Figure 2 on page 7. A fully populated DS5300 with 480 SATA disks configured as 8+2P RAID 6 arrays will serve 48 LUNs if the GPFS best practice of one LUN per array is followed, or multiples thereof if more than one LUN per array is used. Therefore, five DS5300 subsystems with one LUN per array equal 240 LUNs, or five DS5300 subsystems with two LUNs per array equal 480 LUNs. All of these are reasonable configurations to expect in larger deployments.

It is obvious that the number of resources required for a VSCSI based solution will quickly exhaust the capabilities of the virtualization layer. For example, the PowerVM™ Wiki at `http://www.ibm.com/developerworks/wikis/display/virtualization/VIOS+Detail` states these limits:

> "However, the virtual I/O server partition supports a maximum of 65535 virtual I/O slots. A maximum of 256 virtual I/O slots can be assigned to a single partition."

In addition to the significantly reduced requirements for virtualization resources, a second benefit of NPIV virtualization is that LUN provisioning is controlled in a single place: LUNs are mapped to their appropriate host group on the DS5300, and this is transparently handled both by the SAN switch (which uses port-based zoning, not WWN-based zoning, as described in "SAN switch setup" on page 13) and by the VIO servers, which have virtualized the FC adapter and have no knowledge of the disk devices served over the virtual FC connection. In the VSCSI case, as individual SCSI devices are virtualized, provisioning a LUN requires additional configuration on the VIO servers to virtualize the LUN and make it accessible to the client LPARs.

Planning for the virtual server and client slots

Before starting the configuration of the System p partitions and the storage virtualization, it is helpful to do some planning. Our first recommendation is to make the hardware setup as symmetric as possible:

- If DS5300 host port "A1" is used for the host group NPIV1, use host port "B1" for the alternate path, not "B2" or any of the other "B" host ports.
- On the SAN switch, use some systematic way of grouping the ports. In our case, we are using odd-numbered ports for "A" paths and even-numbered ports for "B" paths (in the PoC, we need to host both the "A" and "B" paths on a single SAN switch, which is not recommended for production environments), and we are locating ports in the same SAN zone close to each other.
- If the VIO servers are placed into different CPCs, place the FC adapters in the two VIO server partitions/CPCs into identical PCI slots so that they are configured in the same order and get the same device IDs.
- If using dual-port FC adapters, use the same port on both VIO servers for corresponding functions.
- If the VIO servers each consist of several CPCs, try to be symmetric with respect to the order of the CPCs within the VIO servers. In our example, each of the two VIO servers is using PCI slots from two CPCs. Therefore, if the first VIO server uses a PCI slot in the upper CPC for NPIV, the second VIO server also has the corresponding PCI slot in the upper CPC assigned for the corresponding NPIV connection.

While all of this is not required technically, it makes administration much easier. It may not be possible to adhere to these guidelines when expanding an existing infrastructure, but when starting with a new installation, it pays off to make the hardware setup symmetric.

Even more important than the physical layout is the virtualization layout. We strongly recommend using a consistent numbering scheme for the PHYP slots C*xxx*, which are needed to link the AIX devices on the VIO server side and the client LPAR side:

- It is extremely helpful (for planning, deployment, and troubleshooting) to use the *same* slot number for the server slot and corresponding client slot.
- We reserve slot numbers C1*xx* for VIO server one and C2*xx* for VIO server two.
- We reserve slot numbers below C100 for special use (for example, vsa0 is always C0).

- We use numbers C100 to C131 on P01V01 (and C200 to C231 on P01V02) for the boot LUNs of our 32 LPARs (vhost0..vhost31).
- We use slot numbers C135 and above on P01V01 (and C235 and above on P01V02) for the NPIV virtual FC connections (vfchost0 and above).

As our PoC does not use more than 32 LPARs, and we need two virtual FC connections per client LPAR, this layout just fits into the C1*xx* range on P01V01 and into the C2*xx* range on P01V02. For production environments, it is usually advisable to include some space for growth in the initial planning. Figure 2 on page 7 shows some virtual slot numbers that follow our above numbering scheme.

When planning for the VIO servers and client LPARs, we recommend using a machine readable format to document the virtual slot assignments. Doing so makes it easier to automate the configuration of these slots on the HMC, especially when a larger number of slots needs to be configured. Example 2 shows our text file with the necessary information to create the virtual FC slots for the partition labeled “LS01”. To create the virtual FC slots, we need the following data:

- The name of the managed system (We specify this even though we are only using a single physical server to make the configuration consistent when more servers are added at a later time.)
- The name of the VIO server
- The name of the client LPAR
- The slot number that is going to be used for a specific virtual FC connection (Remember that we are using the *same* number on the VIO server and on the client LPAR.)

Example 2 FC virtual slot planning: vfchost-input.txt

```
# managed-system vios-name lpar-name slot
...
ivt6-p01  ivt6-p01v01  ivt6-bcrs-ls01  135
ivt6-p01  ivt6-p01v01  ivt6-bcrs-ls01  136
ivt6-p01  ivt6-p01v02  ivt6-bcrs-ls01  235
ivt6-p01  ivt6-p01v02  ivt6-bcrs-ls01  236
...
```

We could alternatively use system and LPAR IDs instead of names. Here we are using names as they are more descriptive. Note that we do not include slots C100 and C200, which are used for the SAN boot volume, as we only discuss the NPIV setup.

In “NPIV setup” on page 23, we use this file to script the invocation of the HMC command **chsyscfg**, which adds these slots to the LPAR profiles. As each slot needs to be added on both ends of the virtual FC connection, the four lines of data in Example 2 will be used to issue a total of eight **chsyscfg** commands to set up NPIV for a single client LPAR.

Configuring the infrastructure

After the storage virtualization planning has been completed, we now discuss the initial configuration of all the infrastructure components to set up the GPFS cluster with NPIV.

SAN switch setup

In a production environment, we use two separate SAN switches and connect all DS5300 “A” host ports to the first SAN switch and all DS5300 “B” host ports to the second SAN switch to provide increased resilience against FC path failures. In the proof of concept, a single SAN24B switch is used to hold the “A” zones and the “B” zones. Details about this series of SAN switches can be found in *Implementing an IBM/Brocade SAN with 8 Gbps Directors and Switches*, SG24-6116.

To support the PoC configuration, eight SAN zones are defined:

- Four zones (NPIV1A/1B and NPIV2A/2B) for the two storage partitions per client LPAR, which will provide access to the GPFS Data and Metadata LUNs

> **Note:** These four zones are for the *physical* connections. See “Creating DS5300 hosts and NPIV host ports for the client LPARs” on page 31 for details about setting up multiple *virtual* connections (with their own host groups on the DS5300) over these physical connections.

- Two zones (VIOSA/VIOSB) for the Rootvg LUNs passed to the VIOS servers
- Two zones (TESTA/TESTB) for miscellaneous testing (not covered here)

As the switch is only used locally to interconnect the System p server and DS5300, zones can be defined just by port numbers; no additional security, such as restricting zones to specific WWNs, is required. This has the big advantage that when new client LPARs are defined and their NPIV WWNs become visible on the SAN fabric, no actions are needed on the SAN switch. The mapping of the GPFS LUNs to the new client LPARs can be done completely on the DS5300.

Figure 3 shows the front of the SAN24B switch. Note that the ports are numbered with white and black boxes above the top row of ports. For convenience, we use the “white” ports (in the top row) for “A” ports, and the “black” ports (in the bottom row) for “B” ports. For the PoC, we use the 14 ports to the left, in six port groups. Red color indicates the NPIV1A/1B ports, blue color indicates the NPIV2A/2B ports, and violet indicates VIOSA/VIOSB.

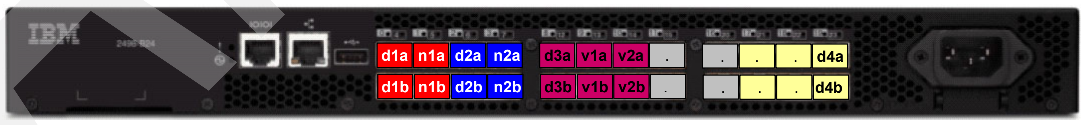


Figure 3   SAN24B switch with SAN zones indicated by color

The setup of the SAN switch is straightforward. The first step is to verify that the SAN switch has a current firmware level, and if necessary upgrade to the most recent level. You can do this in the GUI, or by logging in through SSH and using the **`version`** command:

```
ssh admin@9.152.35.10
admin@9.152.35.10's password:
-----------------------------------------------------------------
IVT6-SAN01:admin> version
```

```
Kernel:     2.6.14.2
Fabric OS:  v6.3.0c
Made on:    Thu Dec 17 19:03:20 2009
Flash:      Wed Mar 31 13:30:27 2010
BootProm:   1.0.9
```

It is also worth checking that the switch is capable to handle NPIV, by checking that the "NPIV capability" column of the **portcfgshow** command reports "ON" (some output omitted):

```
IVT6-SAN01:admin> portcfgshow
Ports of Slot 0    0  1  2  3    4  5  6  7    8  9 10 11   12 13 14 15
-----------------+--+--+--+--+----+--+--+--+----+--+--+--+----+--+--+--
Speed             8G 8G 8G 8G   8G 8G 8G 8G   8G 8G 8G 8G   8G 8G 8G 8G
Fill Word          0  1  0  1    0  1  0  1    0  1  1  1    0  1  1  1
...
NPIV capability   ON ON ON ON   ON ON ON ON   ON ON ON ON   ON ON ON ON
...

Ports of Slot 0   16 17 18 19   20 21 22 23
-----------------+--+--+--+--+----+--+--+--
Speed             8G 8G 8G 8G   8G 8G 8G 8G
Fill Word          1  1  1  0    1  1  1  0
...
NPIV capability   ON ON ON ON   ON ON ON ON
...

where AE:QoSAutoEnable, AN:AutoNegotiate, ..:OFF, NA:NotApplicable, ??:INVALID,
```

Note that there are two blocks of output, as there are more ports than what would fit on a single line. Be sure to check all ports.

Attention: Our testing with the SAN24B at Fabric OS V6.3.0c revealed that with the default settings, the SAN24B and the 8 Gbps host ports of the DS5300 only randomly synchronized their FC8 ports. Setting the SAN switch ports connected to the DS5300 to 4 Gbps fixes this problem. This issue will be addressed in a future firmware release.

A workaround to enable 8 Gbps is described in RETAIN® tip H196488, available at the following address:

http://www-947.ibm.com/systems/support/supportsite.wss/docdisplay?brandind=5000008&lndocid=MIGR-5083089

Manually setting the fillword to 0 (IDLE) for the ports connected to the DS5300 allows the ports to function properly at 8 Gbps:

```
IVT6-SAN01:admin> portcfgfillword  0 0
IVT6-SAN01:admin> portcfgfillword  2 0
IVT6-SAN01:admin> portcfgfillword  4 0
IVT6-SAN01:admin> portcfgfillword  6 0
IVT6-SAN01:admin> portcfgfillword  8 0
IVT6-SAN01:admin> portcfgfillword 12 0
IVT6-SAN01:admin> portcfgfillword 19 0
IVT6-SAN01:admin> portcfgfillword 23 0
```

The above **portcfgshow** output shows the settings in the "Fill Word" column after these changes have been made for the 8 ports that attach to DS5300 host ports (0 means IDLE and 1 means ARB).

We have created the SAN zones using the GUI (see *Implementing an IBM/Brocade SAN with 8 Gbps Directors and Switches*, SG24-6116 for details). Example 3 shows the resulting zone setup that displays when you run the **zoneshow** command.

Example 3 SAN zone configuration

```
IVT6-SAN01:admin> zoneshow
Defined configuration:
 cfg:   IVT6    NPIV1A; NPIV1B; NPIV2A; NPIV2B; TESTA; TESTB; VIOSA; VIOSB
 zone:  NPIV1A  1,0; 1,1
 zone:  NPIV1B  1,4; 1,5
 zone:  NPIV2A  1,2; 1,3
 zone:  NPIV2B  1,6; 1,7
 zone:  TESTA   1,17; 1,18; 1,19
 zone:  TESTB   1,21; 1,22; 1,23
 zone:  VIOSA  1,8; 1,9; 1,10
 zone:  VIOSB  1,12; 1,13; 1,14

Effective configuration:
 cfg:   IVT6
... same info as above for "defined" configuration IVT6 ...
```

This completes the SAN switch setup. When additional servers or storage controllers are added, then the SAN zones need to be expanded accordingly.

DS5300 setup

The DS5300 storage subsystem and its configuration is described in great detail in *IBM Midrange System Storage Hardware Guide*, SG24-7676 and *IBM Midrange System Storage Implementation and Best Practices Guide*, SG24-6363.

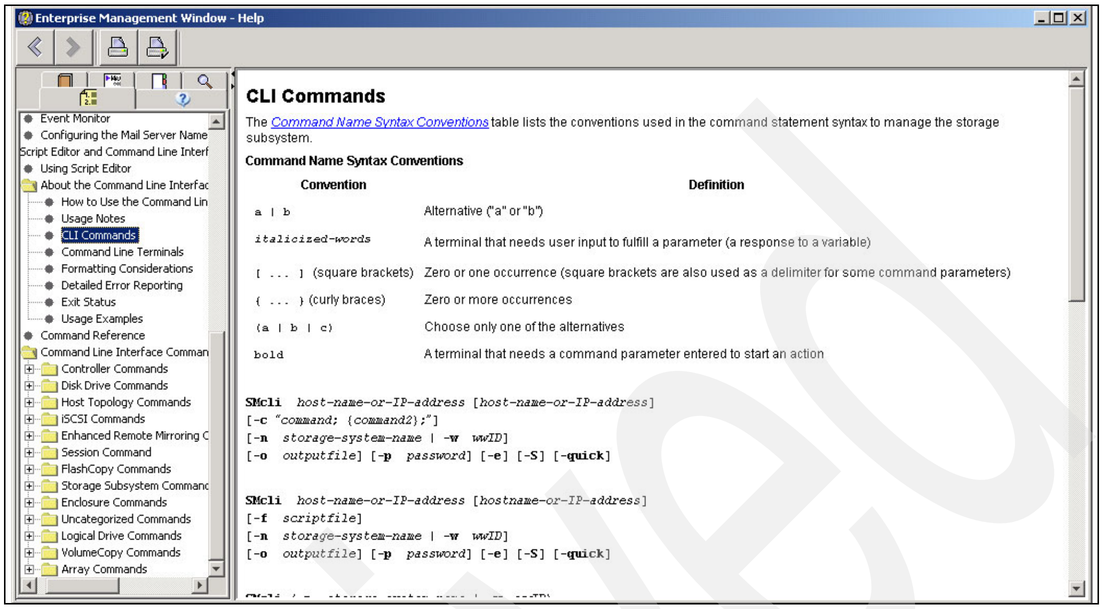
The DS5300 setup can be subdivided into three steps:

1. The creation of the disk arrays and LUNs, and their distribution across host groups. This can be done without the knowledge of the WWN numbers of the physical FCS adapters in the VIO servers or the virtual WWN numbers assigned by the NPIV layer.
2. "VIOS hosts and host ports mapping" on page 21 describes the setup for the VIOS hosts and their FC host ports. This requires the knowledge of the WWN numbers of the physical FCS adapters in the VIO servers. As those WWNs are typically printed on the physical adapter cards, this step can be done without having the VIO servers running.
3. "Creating DS5300 hosts and NPIV host ports for the client LPARs" on page 31 describes the setup for the client LPAR hosts and their NPIV ports. This requires the knowledge of the NPIV WWNs that have been assigned to the client LPARs.

The System p hypervisor (PHYP) assigns the NPIV WWNs, so there is no way to complete step 3 without actually performing the NPIV setup. Therefore, step 3 has to be deferred to the NPIV section, but all other steps for the DS5300 setup are discussed here.

Using the DS5300 command-line interface

We prefer to configure the DS5300 through the command-line interface (CLI) using the **SMcli** command. The CLI and the available scripting commands are documented in *IBM System Storage DS3000, DS4000, and DS5000 Command Line Interface and Script Commands Programming Guide*, GC52-1275. Online help is also available from the GUI, which is started by running the **SMcli** command. In the main DS Storage Manager window (*not* in the windows displaying a specific storage subsystem), selecting **Help** → **Contents** opens the help window (Figure 4), which includes the CLI help and CLI commands reference.



CLI Commands

The *Command Name Syntax Conventions* table lists the conventions used in the command statement syntax to manage the storage subsystem.

Command Name Syntax Conventions

Convention	Definition
`a	b`	Alternative ("a" or "b")
`italicized-words`	A terminal that needs user input to fulfill a parameter (a response to a variable)
`[ ... ]` (square brackets)	Zero or one occurrence (square brackets are also used as a delimiter for some command parameters)
`{ ... }` (curly braces)	Zero or more occurrences
`(a | b | c)`	Choose only one of the alternatives
`bold`	A terminal that needs a command parameter entered to start an action

```
SMcli  host-name-or-IP-address [host-name-or-IP-address]
[-c "command; {command2};"]
[-n  storage-system-name | -w  wwID]
[-o  outputfile] [-p  password] [-e] [-S] [-quick]

SMcli  host-name-or-IP-address [hostname-or-IP-address]
[-f  scriptfile]
[-n  storage-system-name | -w  wwID]
[-o  outputfile] [-p  password] [-e] [-S] [-quick]
```

Figure 4   DS Storage Manager help: CLI commands

Using the CLI, we typically generate a text file, such as `scriptfile.txt`, with the DS5300 commands to perform a specific function, and then run these command on the storage subsystem using the **`SMcli`** command. The first two arguments to that command are the TCP/IP names of the “A” and “B” controllers (in our case, d01a and ds01b). The -f option points to the script, and most activities require the DS5300’s controller password to be provided after the -p option:

```
SMcli ds01a ds01b -f scriptfile.txt [-o scriptfile.log] -p password
```

The following sections will only show the contents of the script files that need to be run to perform the various tasks. Configuring the DS5300 this way eases both planning and documentation of the configuration and has proven to be valuable in large setups.

Array and LUN creation

Creation of the DS5300 arrays and LUNs is performed as for any other DS5300 deployment, with no special considerations regarding the use of NPIV. Array and LUN layout is guided by the application requirements. Here we are using three types of arrays:

- RAID 1 arrays on FC disks for GPFS *metadata*, named DS01M*xx*
- RAID 5 arrays on SATA disks for GPFS *data*, named DS01D*yy*

> **Note:** RAID 6 is often a better choice for today’s high capacity SATA disks, as it protects against double failures, but with the limited amount of hardware in the PoC, we prefer more (4+P) RAID 5 arrays over few (8+2P) RAID 6 arrays.

- A RAID 1 array on SATA disks to hold the *rootvg* LUNs for SAN boot of the client LPARs, named DS01R*zz*

We strongly recommend using a naming convention that is unique across *all* DS5300 storage subsystems, arrays, and LUNs in the GPFS cluster. Such unique names can then also be used as GPFS Network Shared Disk (NSD) names without naming conflicts (see "NSD creation" on page 43). This is convenient for configuration and troubleshooting. On our single DS5300 named DS01, we name the arrays DS01M*xx*, DS01D*yy*, and DS01R*zz*. The LUNs on those arrays will be named by adding suffixes L*xx* to the array names, even if there is only a single LUN L01. This scheme ensures unique names and is easily expandable to more storage subsystems, more arrays, and more LUNs per array. Example 4 shows the array creation.

Example 4 DS5300 create array commands

```
show "creating GPFS (M)etadata arrays RAID1 1+P...";
create array diskDrives=(11,1  11,2  ) raidLevel=1 userLabel="DS01M01";
create array diskDrives=(11,3  11,4  ) raidLevel=1 userLabel="DS01M02";
create array diskDrives=(11,5  11,6  ) raidLevel=1 userLabel="DS01M03";
create array diskDrives=(11,7  11,8  ) raidLevel=1 userLabel="DS01M04";
create array diskDrives=(11,9  11,10 ) raidLevel=1 userLabel="DS01M05";
create array diskDrives=(11,11 11,12 ) raidLevel=1 userLabel="DS01M06";
create array diskDrives=(11,13 11,14 ) raidLevel=1 userLabel="DS01M07";
create array diskDrives=(11,15 11,16 ) raidLevel=1 userLabel="DS01M08";

show "creating GPFS (D)ata arrays RAID5 4+P...";
create array diskDrives=(21,1  21,2  21,3  21,4  21,5 ) raidLevel=5 userLabel="DS01D01";
create array diskDrives=(21,6  21,7  21,8  21,9  21,10) raidLevel=5 userLabel="DS01D02";
create array diskDrives=(21,11 21,12 21,13 21,14 21,15) raidLevel=5 userLabel="DS01D03";
create array diskDrives=(31,1  31,2  31,3  31,4  31,5 ) raidLevel=5 userLabel="DS01D04";
create array diskDrives=(31,6  31,7  31,8  31,9  31,10) raidLevel=5 userLabel="DS01D05";
create array diskDrives=(31,11 31,12 31,13 31,14 31,15) raidLevel=5 userLabel="DS01D06";

show "creating (R)ootvg arrays RAID1 1+P...";
create array diskDrives=(21,16 31,16) raidLevel=1 userLabel="DS01R01";
```

Note: There are no spare drives defined. While this setup can be used in a PoC, spare drives should be used in a production environment to protect the arrays.

Example 5 on page 18 shows the LUN creation on the arrays that were created in Example 4. You can do both steps with a single command, but we prefer to configure arrays and LUNs separately to better control the setup.

- We are configuring ten LUNs on each of the GPFS *data* and *metadata* arrays (five for each of two subprojects). This does not following the usual GPFS best practices: GPFS assumes that different LUNs are on different hardware and stripes across all LUNs. In our case, this is a conscious design choice: For two test projects, we are creating five different file systems (one on each of the L*xx* LUN numbers), each spanning all arrays, to be able to benefit from the aggregate performance of all arrays in each of the file systems. This is only going to work well when the file systems are not active at the same time. This is the case for the ECM archiving solution that motivated this study, in which one or two of the file systems are active while the other file systems are rarely accessed.
- Only odd-numbered GPFS data and metadata *LUNs* are shown. Two testing projects are sharing the arrays, one using the odd LUNs and the other using the even LUNs. This works for testing, but again would not be done in production environments.
- All LUNs on odd-numbered *arrays* are owned by controller "A", and all LUNs on even-numbered *arrays* are owned by controller "B". This setup helps evenly balance the load on the FC links using the AIX MPIO multipath driver.

- Note the different *capacity* of the GPFS metadata and data LUNs to accommodate different file system capacities and number of inodes/files in the different GPFS file systems. This is governed by the application layer, in our case, the sizing of the file systems used within the ECM solution.
- We create 32 *rootvg* LUNs to be able to support the SAN boot of 32 LPARs. In production, one would probably not put all of those LUNs onto a single SATA array.
- Only the *rootvg* LUNs have cacheReadPrefetch enabled. For the GPFS LUNs, this is disabled, as GPFS has its own prefetching logic that would conflict with DS5300 prefetching.

We do not cover any other performance tuning of the LUN settings, as this PoC is focusing on the functional verification of the NPIV and storage design, not on performance.

Example 5 DS5300 create logicalDrive commands (only odd-numbered LUNs for “BCRS” shown)

```
show "creating GPFS (M)etadata logical drives...";
// only showing odd-numbered LUNs for BCRS subproject...
create logicalDrive array="DS01M01" userLabel="DS01M01L01" capacity=128 GB owner=a
cacheReadPrefetch=FALSE segmentSize=32;
create logicalDrive array="DS01M01" userLabel="DS01M01L03" capacity=16  GB owner=a
cacheReadPrefetch=FALSE segmentSize=32;
create logicalDrive array="DS01M01" userLabel="DS01M01L05" capacity=16  GB owner=a
cacheReadPrefetch=FALSE segmentSize=32;
create logicalDrive array="DS01M01" userLabel="DS01M01L07" capacity=4   GB owner=a
cacheReadPrefetch=FALSE segmentSize=32;
create logicalDrive array="DS01M01" userLabel="DS01M01L09" capacity=4   GB owner=a
cacheReadPrefetch=FALSE segmentSize=32;
...
create logicalDrive array="DS01M08" userLabel="DS01M08L09" capacity=4   GB owner=b
cacheReadPrefetch=FALSE segmentSize=32;

show "creating GPFS (D)ata logical drives...";
// only showing odd-numbered LUNs for BCRS subproject...
create logicalDrive array="DS01D01" userLabel="DS01D01L01" capacity=1024 GB owner=a
cacheReadPrefetch=FALSE segmentSize=256;
create logicalDrive array="DS01D01" userLabel="DS01D01L03" capacity=256 GB  owner=a
cacheReadPrefetch=FALSE segmentSize=256;
create logicalDrive array="DS01D01" userLabel="DS01D01L05" capacity=128 GB  owner=a
cacheReadPrefetch=FALSE segmentSize=256;
create logicalDrive array="DS01D01" userLabel="DS01D01L07" capacity=32 GB   owner=a
cacheReadPrefetch=FALSE segmentSize=256;
create logicalDrive array="DS01D01" userLabel="DS01D01L09" capacity=32 GB   owner=a
cacheReadPrefetch=FALSE segmentSize=256;
...
create logicalDrive array="DS01D06" userLabel="DS01D06L09" capacity=32 GB   owner=b
cacheReadPrefetch=FALSE segmentSize=256;

show "creating (R)ootvg logical drives...";
create logicalDrive array="DS01R01" userLabel="DS01R01L01" capacity=28 GB owner=a
cacheReadPrefetch=TRUE segmentSize=256;
...
create logicalDrive array="DS01R01" userLabel="DS01R01L32" capacity=28 GB owner=a
cacheReadPrefetch=TRUE segmentSize=256;
```

Host group creation

On the DS5300, we create host groups that correspond to the SAN zones that have been created on the SAN switch. On the *physical* layer, there are two SAN zones per DS5300 host group (one for the "A" path and one for the "B" path), as a DS5300 host group always includes both controller paths.

One big advantage of NPIV is the ability to implement multiple *virtual* SAN zones over a single physical SAN connection: We have physically implemented only two pairs of FC connections between the DS5300 and the System p server's VIOS partitions (one for GPFS metadata LUNs and one for GPFS data LUN), with four corresponding SAN zones on the SAN switch. However, with NPIV, we can create multiple host groups on the DS5300 and map different groups of client LPARs with their NPIV WWNs into those different host groups. Those connections will share the same physical FC connections, yet will be completely isolated from each other. In the PoC, we use this technique to host two subprojects, BCRS and IVT6, on the same hardware.

The following host groups are created on the DS5300 that correspond to the SAN zones (note that the DS5300 host groups always include "A" and "B" ports, so there are twice as many SAN zones than DS5300 host groups):

- One host group "VIOS" for the VIO servers and the SAN boot LUNs.
- Two host groups for NPIV and GPFS would correspond to the four zones NPIV1A/NPIV1B and NPIV2A/NPIV2B on the SAN switch. As mentioned above, we are hosting two independent subprojects and so generate two sets of host groups:
  - Host groups BCRS_NPIV1 and BCRS_NPIV2 for subproject BCRS
  - Host groups IVT6_NPIV1 and IVT6_NPIV2 for subproject IVT6

  In "NPIV setup" on page 23, the two groups of client LPARs of those two subprojects are mapped into those two pairs of NPIV host groups, separating access to the two groups of LUNs.
- One host group "TEST" for miscellaneous testing, not discussed here.

Note: We recommend defining a host group VIOS_NOMAP, which will only be used to hold the WWNs of the VIOS physical FCS adapters that will carry the NPIV traffic. No LUNs will be mapped to this host group: All GPFS LUNs are mapped to the NPIV1 and NPIV2 host groups, which contain the NPIV WWNs of the client LPARs (as opposed to the WWNs of the physical adapters in the VIOS servers over which those NPIV clients connect). By putting those FCS adapters' WWNs into this dummy host groups, we remove them from the list of "unknown" host ports, so all WWNs in the "unknown" state should be NPIV WWNs that need to be mapped into the NPIV1/NPIV2 host groups.

Example 6 shows the DS5300 commands used to create the host groups.

Example 6 DS5300 create hostGroup commands

```
show "creating host groups...";
create hostGroup userLabel="VIOS";
create hostGroup userLabel="VIOS_NOMAP";
create hostGroup userLabel="BCRS_NPIV1";
create hostGroup userLabel="BCRS_NPIV2";
create hostGroup userLabel="IVT6_NPIV1";
create hostGroup userLabel="IVT6_NPIV2";
create hostGroup userLabel="TEST";
```

Mapping the LUNs to host groups

After creating the host groups, the LUNs can be mapped into their appropriate host groups. No hosts or host ports need to be defined to map the LUNs. The hosts and host ports are needed eventually to make the LUNs accessible, but they can be defined at a later time after mapping on the host group level. Example 7 shows the DS5300 commands to map the LUNs. Note that we are mapping the odd-numbered LUNs into one pair of NPIV host groups (for the BCRS subproject), and are mapping the even-numbered LUNs into the other pair of NPIV host groups (for the IVT6 subproject).

Note: The SCSI IDs ("LUN numbers") are assigned at this time. They must be unique within each host group, but not across host groups (as different host groups connect to hosts using different FCS adapters).

Example 7   DS5300 set logicalDrive hostgroup command

```
show "mapping (R)ootvg logical drives into VIOS host group...";
set logicalDrive ["DS01R01L01"] logicalUnitNumber=1  hostGroup="VIOS";
set logicalDrive ["DS01R01L02"] logicalUnitNumber=2  hostGroup="VIOS";
set logicalDrive ["DS01R01L03"] logicalUnitNumber=3  hostGroup="VIOS";
...
set logicalDrive ["DS01R01L30"] logicalUnitNumber=30 hostGroup="VIOS";
set logicalDrive ["DS01R01L31"] logicalUnitNumber=31 hostGroup="VIOS";
set logicalDrive ["DS01R01L32"] logicalUnitNumber=32 hostGroup="VIOS";


show "mapping GPFS (D)ata logical drives into BCRS_NPIV1 host group...";
set logicalDrive ["DS01D01L01"] logicalUnitNumber=1  hostGroup="BCRS_NPIV1";
set logicalDrive ["DS01D01L03"] logicalUnitNumber=3  hostGroup="BCRS_NPIV1";
...
set logicalDrive ["DS01D06L07"] logicalUnitNumber=57 hostGroup="BCRS_NPIV1";
set logicalDrive ["DS01D06L09"] logicalUnitNumber=59 hostGroup="BCRS_NPIV1";

show "mapping GPFS (M)etadata logical drives into BCRS_NPIV2 host group...";
set logicalDrive ["DS01M01L01"] logicalUnitNumber=1  hostGroup="BCRS_NPIV2";
set logicalDrive ["DS01M01L03"] logicalUnitNumber=3  hostGroup="BCRS_NPIV2";
...
set logicalDrive ["DS01M08L07"] logicalUnitNumber=77 hostGroup="BCRS_NPIV2";
set logicalDrive ["DS01M08L09"] logicalUnitNumber=79 hostGroup="BCRS_NPIV2";


show "mapping GPFS (D)ata logical drives into IVT6_NPIV1 host group...";
set logicalDrive ["DS01D01L02"] logicalUnitNumber=2  hostGroup="IVT6_NPIV1";
set logicalDrive ["DS01D01L04"] logicalUnitNumber=4  hostGroup="IVT6_NPIV1";
...
set logicalDrive ["DS01D06L08"] logicalUnitNumber=58 hostGroup="IVT6_NPIV1";
set logicalDrive ["DS01D06L10"] logicalUnitNumber=60 hostGroup="IVT6_NPIV1";

show "mapping GPFS (M)etadata logical drives into IVT6_NPIV2 host group...";
set logicalDrive ["DS01M01L02"] logicalUnitNumber=2  hostGroup="IVT6_NPIV2";
set logicalDrive ["DS01M01L04"] logicalUnitNumber=4  hostGroup="IVT6_NPIV2";
...
set logicalDrive ["DS01M08L08"] logicalUnitNumber=78 hostGroup="IVT6_NPIV2";
set logicalDrive ["DS01M08L10"] logicalUnitNumber=80 hostGroup="IVT6_NPIV2";
```

VIOS hosts and host ports mapping

Now we define the VIOS hosts and their host ports. We use a single System p server with two VIO servers, named P01V01 and P01V02. Each of the two VIO servers has two dual-port FC adapters assigned to it, for a total of four FC ports:

- One of the two adapters is used for *rootvg* LUNs for SAN booting the client LPARs. Its two adapter ports are mapped into the "VIOS" host group to make the SAN boot LUNs visible on the VIO servers.
- The second adapter is used for NPIV. The LUNs served to the client LPARs via NPIV are *not* be visible on the VIO servers themselves, so no LUNs are mapped to the WWNs of those adapter ports. This adapter's two ports are mapped to the dummy host group "VIOS_NOMAP" simply for ease of management: when mapped, they do not appear in the DS5300's list of "unknown" ports.

By associating a host with a host group, it automatically gains access to the LUNs mapped into that host group (through the host ports that are configured for that host). Example 8 shows the creation of the two hosts "P01V01" and "P01V02" (together with the two dummy hosts for the "VIOS_NOMAP" host group), followed by the creation of the eight FC host ports.

Example 8 DS5300 create host and create hostPort commands for VIO servers physical FC ports

```
show "creating VIOS and VIOS_NOMAP hosts...";
create host       userLabel="P01V01" hostGroup="VIOS"        hostType="AIX";
create host       userLabel="P01V02" hostGroup="VIOS"        hostType="AIX";
create host userLabel="P01V01_NOMAP" hostGroup="VIOS_NOMAP"  hostType="AIX";
create host userLabel="P01V02_NOMAP" hostGroup="VIOS_NOMAP"  hostType="AIX";

show "creating host ports for VIOS servers (FCS for rootvg LUNs)...";
create hostPort identifier="10000000c995ae1a" userLabel="P01-DQD0D4H-P1-C1-T1"
host="P01V01" interfaceType=FC;
create hostPort identifier="10000000c995ae1b" userLabel="P01-DQD0D4H-P1-C1-T2"
host="P01V01" interfaceType=FC;
create hostPort identifier="10000000c995ad86" userLabel="P01-DQD0D3Y-P1-C1-T1"
host="P01V02" interfaceType=FC;
create hostPort identifier="10000000c995ad87" userLabel="P01-DQD0D3Y-P1-C1-T2"
host="P01V02" interfaceType=FC;

show "creating host ports for VIOS_NOMAP (FCS for NPIV, not used on VIOS)...";
create hostPort identifier="10000000c99432a2" userLabel="P01-DQD0D3X-P1-C1-T1"
host="P01V01_NOMAP" interfaceType=FC;
create hostPort identifier="10000000c99432a3" userLabel="P01-DQD0D3X-P1-C1-T2"
host="P01V01_NOMAP" interfaceType=FC;
create hostPort identifier="10000000c995b09a" userLabel="P01-DQD0D3W-P1-C1-T1"
host="P01V02_NOMAP" interfaceType=FC;
create hostPort identifier="10000000c995b09b" userLabel="P01-DQD0D3W-P1-C1-T2"
host="P01V02_NOMAP" interfaceType=FC;
```

Note: We strongly recommend assigning meaningful and unique labels to the host ports. Here we store (most of) the physical location code of the adapter port (as reported by the **`lsslot -c pci`** AIX command or the **`lsnports`** VIOS command) in the DS5300 userLabel field, so we can easily correlate the DS5300 host port with the corresponding end on the System p side. (We have replaced the AIX location code prefix "U789D.001." with the more mnemonic "P01-" to identify our server P01 by name.)

After the DS5300 setup is complete, scanning the FCS adapters on the VIO servers (by invoking the **cfgmgr** AIX command) should discover the SAN boot LUNs on the appropriate FCS ports. None of the GPFS data and metadata LUNs should be visible at the VIO servers, as they will only be mapped to the NPIV1/NPIV2 host groups. This mapping is done as part of the NPIV setup in “Creating DS5300 hosts and NPIV host ports for the client LPARs” on page 31.

VIO server setup and basic LPAR setup

In this paper, we are not discussing the general steps for creating System p logical partitions (LPARs) or setting up a virtual I/O server (VIOS). *PowerVM Virtualization on IBM System p: Introduction and Configuration Fourth Edition*, SG24-7940 and *IBM PowerVM Virtualization Managing and Monitoring*, SG24-7590 cover these topics.

We also do not discuss here how to establish TCP/IP connectivity for the client LPARs. Even though GPFS accesses its disks directly through Fibre Channel connections, there still needs to be TCP/IP connectivity between all the nodes in a GPFS cluster for control traffic. The two books mentioned above also discuss how virtual Ethernet can be set up, using the VIO servers. Complementing this setup, the System p Integrated Virtual Ethernet (IVE) option allows external network connectivity for LPARs using dedicated ports without requiring a Virtual I/O Server. This setup has been supported since POWER6. Details about IVE can be found in *Integrated Virtual Ethernet Adapter Technical Overview and Introduction*, REDP-4340. In our PoC setup, we are using IVE to establish TCP/IP connectivity for the VIO servers and LPARs.

One aspect that we *do* want to focus on is automation by using the HMC’s command-line interface (CLI). Refer to *Hardware Management Console V7 Handbook*, SG24-7491 for a description of the HMC, including the GUI and CLI. The HMC command man pages can be found at the following address:

http://www14.software.ibm.com/webapp/set2/sas/f/hmc/power6/related/hmc_man_7310.pd
f

To display the current status of your environment, the **lssyscfg -r sys** HMC command lists all managed systems controlled by this HMC (in our case, only the server ivt6-p01):

```
hscroot@ivt6-hmc01:~> lssyscfg -r sys -F name
ivt6-p01
```

The **lssyscfg -r lpar** HMC command lists the LPAR names and LPAR IDs of all LPARs on a specific managed system:

```
hscroot@ivt6-hmc01:~> lssyscfg -r lpar -F name,lpar_id -m ivt6-p01 | sort
ivt6-bcrs-ls01,11
ivt6-bcrs-srv01,5
ivt6-p01v01,1
ivt6-p01v02,2
...
```

For the remainder of this paper, we assume that the client LPARs have already been set up, with CPU and memory resources appropriately sized. We also assume that TCP/IP network connectivity has been established for administrative use and for GPFS use. The NPIV setup will be discussed in “NPIV setup” on page 23, using HMC commands for those configuration steps that need to be performed on the HMC.

Note: As a prerequisite to automating the HMC configuration, you should ensure that your user ID on your workstation is authorized to run password-less SSH commands on the HMC (as the hscroot user). This is done by storing your SSH public key in the hscroot user's SSH authorized_keys file on the HMC. You store this key by running the **`mkauthkeys -a`** HMC command.

NPIV setup

After the VIO server partitions and client LPARs have been defined and installed, the following steps are performed to set up the NPIV storage virtualization:

1. The virtual FC *server* slots are added to the VIO server partition profiles by running the **`chsyscfg`** HMC command, and the VIO server partitions are reactivated to pick up the changed partition profile.
2. On the VIO servers, the **`vfcmap`** VIOS command is used to *associate* the virtual FC server slots with physical FCS adapter ports on the VIO servers.
3. The virtual FC *client* slots are added to the client LPAR partition profiles by running the **`chsyscfg`** HMC command; this step creates the NPIV WWNs. The client LPAR partitions are reactivated to pick up the changed partition profile.
4. On the DS5300 storage subsystem, the GPFS LUNs are made accessible to the client LPARs by *mapping* the client LPARs' NPIV WWNs as host ports into the DS5300 host groups, which have been created in the previous step.
5. On the client LPARs, the NPIV LUNs are *discovered* as AIX hdisks by running the **`cfgmgr`** AIX command.
6. Some AIX device driver *tuning* is performed by running the **`lsattr`** and **`chdev`** AIX commands.

The general NPIV setup is described in the following System p online documentation:

http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/index.jsp?topic=/iphat/iphatvfc.htm

PowerVM Migration from Physical to Virtual Storage, SG24-7825 contains some information about NPIV. In the following section, we exclusively use CLI commands to perform the configuration of NPIV for our GPFS setup. This procedure is particularly useful when configuring large numbers of virtual connections.

Setting up the virtual FC server adapters

As with most System p LPAR operations, there are two ways to perform configuration changes: The persistent method is to change the partition profile with the **`chsyscfg`** HMC command, and then restart the partition. An alternative is to use a Dynamic LPAR (DLPAR) operation, which takes effect immediately, but does not survive a reboot (so the partition profile should be changed accordingly to also preserve the configuration change across partition restarts). In the following section, we only use the persistent method. This approach is perfectly fine for initial configuration when no workload is running on the server. As we are using redundant VIO servers, you can also use this method in production, and restart one VIO server at a time to incorporate any changes.

Initially, the VIO servers do not have any virtual FC *server* adapters defined. This definition can be verified on the HMC by querying the virtual_fc_adapters LPAR property with the **lssyscfg** HMC command:

```
ssh hscroot@ivt6-hmc01
$ lssyscfg -r prof -m ivt6-p01 -F lpar_name virtual_fc_adapters | sort
...
ivt6-p01v01 none
ivt6-p01v02 none
...
```

Should there be some virtual_fc_adapters data that you want to clear out, this can be done by setting the virtual_fc_adapters property to none:

```
$ chsyscfg -r prof -m ivt6-p01 \
  -i 'lpar_name=ivt6-p01v01,name=default,"virtual_fc_adapters=none"'
```

On the two VIO servers, the **lsslot -c slot** AIX command can be used to view the virtual slots (this command is only available in the privileged shell, so you need to issue the **oem_setup_env** VIOS command after logging in to the VIO server as the padmin user):

```
ssh padmin@ivt6-p01v01
$ oem_setup_env
# lsslot -c slot
# Slot                      Description       Device(s)
HEA 1                       Logical I/O Slot  lhea0 ent0
U789D.001.DQD0D4H-P1-T3     Logical I/O Slot  pci3 sissas0
U9117.MMA.107769E-V1-C0     Virtual I/O Slot  vsa0
U9117.MMA.107769E-V1-C2     Virtual I/O Slot  vasi0
U9117.MMA.107769E-V1-C100   Virtual I/O Slot  vhost0
U9117.MMA.107769E-V1-C101   Virtual I/O Slot  vhost1
U9117.MMA.107769E-V1-C102   Virtual I/O Slot  vhost2
...
U9117.MMA.107769E-V1-C130   Virtual I/O Slot  vhost30
# exit
$ exit

ssh padmin@ivt6-p01v02
$ oem_setup_env
# lsslot -c slot
# Slot                      Description       Device(s)
HEA 1                       Logical I/O Slot  lhea0 ent0
U789D.001.DQD0D3Y-P1-T3     Logical I/O Slot  pci5 sissas1
U9117.MMA.107769E-V2-C0     Virtual I/O Slot  vsa0
U9117.MMA.107769E-V2-C2     Virtual I/O Slot  vasi0
U9117.MMA.107769E-V2-C200   Virtual I/O Slot  vhost0
U9117.MMA.107769E-V2-C201   Virtual I/O Slot  vhost1
U9117.MMA.107769E-V2-C202   Virtual I/O Slot  vhost2
...
U9117.MMA.107769E-V2-C230   Virtual I/O Slot  vhost30
# exit
$ exit
```

The virtual slot listing shows `vhost`*XX* devices, which are used for the VSCSI SAN boot, but not any `vfchost`*XX* devices, which would correspond to NPIV.

The VIO servers' partition profiles are now changed to add one virtual FC *server* slot per client LPAR, using the planning information from "Storage virtualization planning" on page 7.

On our workstation, we are using the shell commands in Example 9 to generate the required **chsyscfg** HMC commands, by parsing the `vfchost-input.txt` file, which we have outlined in Example 2 on page 12. For convenience, we create two script files, one for each VIO server, which will contain the **chsyscfg** HMC commands that can then be inspected and finally cut and pasted into an HMC shell to be executed.

Note that we are *adding* virtual FC slots (by using the `virtual_fc_adapters+=` clause), so be sure to clear out any unwanted previous data (as shown above) before running these commands.

Example 9 Generating the commands to set up the virtual FC server slots

```
grep -vE "^#|^[::space:]]*$" vfchost-input.txt | grep p01v01 | while read M V L S
do
  echo chsyscfg -r prof -m $M \
    -i 'lpar_name=$V,name=default,\"virtual_fc_adapters+=\"\"$S/server//$L/$S//0\"\"' "
done > npiv-p01v01-server-slots.sh

grep -vE "^#|^[::space:]]*$" vfchost-input.txt | grep p01v02 | while read M V L S
do
  echo chsyscfg -r prof -m $M \
    -i 'lpar_name=$V,name=default,\"virtual_fc_adapters+=\"\"$S/server//$L/$S//0\"\"' "
done > npiv-p01v02-server-slots.sh

head -2 npiv-p01v01-server-slots.sh # output is two long lines...
chsyscfg -r prof -m ivt6-p01 -i 'lpar_name=ivt6-p01v01,name=default,
  "virtual_fc_adapters+=""135/server//ivt6-bcrs-ls01/135//0""'
chsyscfg -r prof -m ivt6-p01 -i 'lpar_name=ivt6-p01v01,name=default,
  "virtual_fc_adapters+=""136/server//ivt6-bcrs-ls01/136//0""'

head -2 npiv-p01v02-server-slots.sh # output is two long lines...
chsyscfg -r prof -m ivt6-p01 -i 'lpar_name=ivt6-p01v02,name=default,
  "virtual_fc_adapters+=""235/server//ivt6-bcrs-ls01/235//0""'
chsyscfg -r prof -m ivt6-p01 -i 'lpar_name=ivt6-p01v02,name=default,
  "virtual_fc_adapters+=""236/server//ivt6-bcrs-ls01/236//0""'
```

Note: The quoting format of the **chsyscfg** HMC command is non-obvious: We are using single quotes for the whole value following the -i flag of the **chsyscfg** command. Then, double quotes are used to surround the `"virtual_fc_adapters+="` attribute/value, and finally *pairs* of double quotes are used within that to denote the value itself.

In the loop, we are only printing the commands rather than directly executing them, so we have enclosed the complete **chsyscfg** command string in double quotes (as the argument of the **echo** command), and thus have to shell-escape all the double quotes, which are needed in the **chsyscfg** command.

As mentioned earlier, we recommend using the *same* numerical value for the virtual-slot-number and remote-slot-number when setting up the virtual server and client FC adapters, so it is easier to associate matching server and client slots with each other. This is why we can use the slot variable $S twice in the above virtual_fc_adapters clause (once for the server side slot, and once for the client side slot).

When the commands in the resulting shell scripts npiv-p01v01-server-slots.sh and npiv-p01v01-server-slots.sh have been executed on the HMC, listing the partition configuration should now show all virtual FC server slots in the virtual_fc_adapters attribute:

```
ssh hscroot@ivt6-hmc01
$ lssyscfg -r prof -m ivt6-p01 --filter "lpar_names=ivt6-p01v01"\
    -F lpar_name virtual_fc_adapters # output is one very very long line...
ivt6-p01v01  135/server/11/ivt6-bcrs-ls01/135//0,136/server/11/ivt6-bcrs-ls01/136/
/0,137/server/12/ivt6-bcrs-was01/137//0,138/server/12/ivt6-bcrs-was01/138//0,.....
.,193/server/44/ivt6-ivt6-td03/193//0,194/server/44/ivt6-ivt6-td03/194//0,195/serv
er/6/ivt6-ivt6-srv01/195//0,196/server/6/ivt6-ivt6-srv01/196//0
```

After restarting the VIO server partitions, we can log in to the VIO servers and verify that the virtual FC server slots are now visible by running the **lsslot -c slot** AIX command (here we are only displaying the vfchost lines; see above for the other output of the **lsslot** command):

```
ssh padmin@p01v01
$ oem_setup_env
# lsslot -c slot | grep -E "Description|vfchost"
# Slot                     Description       Device(s)
U9117.MMA.107769E-V1-C135  Virtual I/O Slot  vfchost0
U9117.MMA.107769E-V1-C136  Virtual I/O Slot  vfchost1
U9117.MMA.107769E-V1-C137  Virtual I/O Slot  vfchost2
U9117.MMA.107769E-V1-C138  Virtual I/O Slot  vfchost3
...
U9117.MMA.107769E-V1-C196  Virtual I/O Slot  vfchost61
# exit
$ exit

ssh padmin@p01v02
$ oem_setup_env
# lsslot -c slot | grep -E "Description|vfchost"
# Slot                     Description       Device(s)
U9117.MMA.107769E-V2-C235  Virtual I/O Slot  vfchost0
U9117.MMA.107769E-V2-C236  Virtual I/O Slot  vfchost1
U9117.MMA.107769E-V2-C237  Virtual I/O Slot  vfchost2
U9117.MMA.107769E-V2-C238  Virtual I/O Slot  vfchost3
...
U9117.MMA.107769E-V2-C296  Virtual I/O Slot  vfchost61
# exit
$ exit
```

Alternatively, the **lsmap** VIOS command can be used with the -npiv option to display the mapping of the vfchost*XX* slots. The output of this command also includes the LPAR ID of the corresponding client LPAR:

```
ssh padmin@ivt6-p01v01
$ lsmap -all -npiv | grep ^vfchost
vfchost0      U9117.MMA.107769E-V1-C135             11
vfchost1      U9117.MMA.107769E-V1-C136             11
vfchost2      U9117.MMA.107769E-V1-C137             12
vfchost3      U9117.MMA.107769E-V1-C138             12
...
vfchost60     U9117.MMA.107769E-V1-C195              6
vfchost61     U9117.MMA.107769E-V1-C196              6
$ exit

ssh padmin@ivt6-p01v02
$ lsmap -all -npiv | grep ^vfchost
vfchost0      U9117.MMA.107769E-V2-C235             11
vfchost1      U9117.MMA.107769E-V2-C236             11
vfchost2      U9117.MMA.107769E-V2-C237             12
vfchost3      U9117.MMA.107769E-V2-C238             12
```

```
...
vfchost60     U9117.MMA.107769E-V2-C295              6
vfchost61     U9117.MMA.107769E-V2-C296              6
$ exit
```

Note that at this time that the virtual FC *server* slots are not yet associated with *physical* FC ports on the VIO servers. The virtual FC *client* slots that constitute the "other end" of the NPIV connections have not been created either.

Associating VIOS virtual server adapters with physical FC ports

After the vfchost devices are created, they need to be associated with NPIV-capable physical FC ports on the VIO servers. In our PoC setup, we use the `fcs2` and `fcs3` AIX devices on both VIO servers for NPIV. We can use the **`lsslot -c pci`** AIX command to list the physical (PCI) slots (this command needs the privileged shell, which can be accessed by running **`oem_setup_env`**), or we can use the **`lsnports`** VIOS command, which also lists the number of supported and available WWNs on the fcs*X* devices:

```
ssh padmin@ivt6-p01v01
$ lsnports
name            physloc                        fabric tports aports swwpns  awwpns
fcs0            U789D.001.DQD0D4H-P1-C1-T1          1     64     64   2048    2047
fcs1            U789D.001.DQD0D4H-P1-C1-T2          1     64     64   2048    2047
fcs2            U789D.001.DQD0D3X-P1-C1-T1          1     64     64   2048    2048
fcs3            U789D.001.DQD0D3X-P1-C1-T2          1     64     64   2048    2048
$ exit

ssh padmin@ivt6-p01v02
$ lsnports
name            physloc                        fabric tports aports swwpns  awwpns
fcs0            U789D.001.DQD0D3Y-P1-C1-T1          1     64     64   2048    2047
fcs1            U789D.001.DQD0D3Y-P1-C1-T2          1     64     64   2048    2047
fcs2            U789D.001.DQD0D3W-P1-C1-T1          1     64     64   2048    2048
fcs3            U789D.001.DQD0D3W-P1-C1-T2          1     64     64   2048    2048
$ exit
```

This command output provides the mapping between the AIX device names and the physical locations of the adapter ports. We have already included this information in our FC port documentation in Example 1 on page 6.

On each of the two VIO servers, we are using two consecutive virtual FC slots "C*xxx*" per client (one odd-numbered and one even-numbered), which corresponds to two consecutive vfchost*XX* devices (one even and one odd-numbered). Using the **`vfcmap`** VIOS command, we map the physical FC adapter port fcs2 to all even-numbered vfchostX virtual server slots, and we map fcs3 to all odd-numbered vfchost*XX* virtual server slots. These are the VIOS commands on the first VIO server; the same sequence needs to be run on the second VIO server as well:

```
ssh padmin@ivt6-p01v01 # repeat for ivt6-p01v02

for x in 0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 \
         32 34 36 38 40 42 44 46 48 50 52 54 56 58 60
do
 echo vfcmap -vadapter vfchost$x -fcp fcs2
      vfcmap -vadapter vfchost$x -fcp fcs2
done

for x in 1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 \
         33 35 37 39 41 43 45 47 49 51 53 55 57 59 61
```

```
do
 echo vfcmap -vadapter vfchost$x -fcp fcs3
      vfcmap -vadapter vfchost$x -fcp fcs3
done

exit
```

After the association, we can list the resulting configuration by running the **lsmap** VIOS command:

```
ssh padmin@ivt6-p01v01

$ lsmap -all -npiv | grep -E "^vfchost|^FC name"
vfchost0      U9117.MMA.107769E-V1-C135             11
FC name:fcs2                  FC loc code:U789D.001.DQD0D3X-P1-C1-T1
vfchost1      U9117.MMA.107769E-V1-C136             11
FC name:fcs3                  FC loc code:U789D.001.DQD0D3X-P1-C1-T2
vfchost2      U9117.MMA.107769E-V1-C137             12
FC name:fcs2                  FC loc code:U789D.001.DQD0D3X-P1-C1-T1
vfchost3      U9117.MMA.107769E-V1-C138             12
FC name:fcs3                  FC loc code:U789D.001.DQD0D3X-P1-C1-T2
...
exit

ssh padmin@ivt6-p01v02
$ lsmap -all -npiv | grep -E "^vfchost|^FC name"
vfchost0      U9117.MMA.107769E-V2-C235             11
FC name:fcs2                  FC loc code:U789D.001.DQD0D3W-P1-C1-T1
vfchost1      U9117.MMA.107769E-V2-C236             11
FC name:fcs3                  FC loc code:U789D.001.DQD0D3W-P1-C1-T2
vfchost2      U9117.MMA.107769E-V2-C237             12
FC name:fcs2                  FC loc code:U789D.001.DQD0D3W-P1-C1-T1
vfchost3      U9117.MMA.107769E-V2-C238             12
FC name:fcs3                  FC loc code:U789D.001.DQD0D3W-P1-C1-T2
...
exit
```

Now the server side of the NPIV configuration is complete. What is still missing is the setup of the client side, including the generation of the NPIV WWNs, which can then be used on the storage subsystems to map LUNs to the NPIV endpoints.

Setting up the virtual FC client adapters

After the virtual *server* slots are defined and mapped to physical ports on the VIO servers, we need to complete the NPIV setup by creating matching virtual *client* slots in the client LPAR profiles. So far, the client LPARs do not have any virtual FC *client* adapters defined, which can be verified by running the **lssyscfg** HMC command:

```
ssh hscroot@ivt6-hmc01
$ lssyscfg -r prof -m ivt6-p01 -F lpar_name virtual_fc_adapters | sort
...
ivt6-bcrs-ls01 none
ivt6-bcrs-srv01 none
...
$ exit
```

The commands to generate the virtual FC *client* adapters are very similar to the commands in "Setting up the virtual FC server adapters" on page 23. Example 10 shows how to generate the **chsyscfg** HMC commands to create the virtual FC *client* adapters from the `vfchost-input.txt` file shown in Example 2 on page 12. This is almost identical to Example 9 on page 25; only the usage of $V and $L is reversed and "server" is replaced by "client".

Note that we are *adding* virtual FC slots (by using the `virtual_fc_adapters+=` clause), so be sure to clear out any unwanted previous data (by assigning virtual_fc_adapters=none) before running these commands.

Example 10 Generating the commands to set up the virtual FC client slots

```
grep -vE "^#|^[::space:]]*$" vfchost-input.txt | grep p01v01 | while read M V L S
do
  echo chsyscfg -r prof -m $M \
    -i 'lpar_name=$L,name=default,\"virtual_fc_adapters+=\"\"$S/client//$V/$S//0\"\"' "
done > npiv-p01v01-client-slots.sh

grep -vE "^#|^[::space:]]*$" vfchost-input.txt | grep p01v02 | while read M V L S
do
  echo chsyscfg -r prof -m $M \
    -i 'lpar_name=$L,name=default,\"virtual_fc_adapters+=\"\"$S/client//$V/$S//0\"\"' "
done > npiv-p01v02-client-slots.sh

head -2 npiv-p01v01-client-slots.sh # output is two long lines...
chsyscfg -r prof -m ivt6-p01 -i  'lpar_name=ivt6-bcrs-ls01,name=default,
  "virtual_fc_adapters+=""135/client//ivt6-p01v01/135//0""'
chsyscfg -r prof -m ivt6-p01 -i  'lpar_name=ivt6-bcrs-ls01,name=default,
  "virtual_fc_adapters+=""136/client//ivt6-p01v01/136//0""'

head -2 npiv-p01v02-client-slots.sh # output is two long lines...
chsyscfg -r prof -m ivt6-p01 -i  'lpar_name=ivt6-bcrs-ls01,name=default,
  "virtual_fc_adapters+=""235/client//ivt6-p01v02/235//0""'
chsyscfg -r prof -m ivt6-p01 -i  'lpar_name=ivt6-bcrs-ls01,name=default,
  "virtual_fc_adapters+=""236/client//ivt6-p01v02/236//0""'
```

Note: The quoting format of the **chsyscfg** HMC command is non-obvious: refer to the note following Example 9 on page 25 for an explanation.

When the commands in the resulting shell scripts npiv-p01v01-client-slots.sh and npiv-p01v01-client-slots.sh have been executed on the HMC, a listing of the partition configuration should now contain four virtual FC *client* slots in the virtual_fc_adapters attribute of each client LPAR.

Note: In the following paragraphs, we use the ivt6-bcrs-srv01 LPAR (our GPFS management partition), as this is the first LPAR that is added to the GPFS cluster. Previous examples used a client LPAR named "LS01". The NPIV setup steps are identical for all LPARs.

In contrast to the virtual FC *server* slots, this output also includes *a pair of WWNs* for each NPIV port, which can later be used to map LUNs to these WWNs:

```
ssh hscroot@ivt6-hmc01
$ lssyscfg -r prof -m ivt6-p01 --filter "lpar_names=ivt6-bcrs-srv01" \
    -F lpar_name virtual_fc_adapters # output is one long line...
ivt6-bcrs-srv01  """163/client/1/ivt6-p01v01/163/c050760226b70008,c050760226b70009
/0"",""164/client/1/ivt6-p01v01/164/c050760226b7000a,c050760226b7000b/0"",""263/cl
ient/2/ivt6-p01v02/263/c050760226b7000c,c050760226b7000d/0"",""264/client/2/ivt6-p
01v02/264/c050760226b7000e,c050760226b7000f/0"""
$ exit
```

After a partition restart to incorporate the changes to the LPAR profile, the following output shows the view from the client LPAR using the **lsslot -c slot** AIX command:

```
$ ssh root@ivt6-bcrs-srv01
$ lsslot -c slot
# Slot                     Description      Device(s)
HEA 1                      Logical I/O Slot  lhea0 ent0 ent1 ent2
U9117.MMA.107769E-V5-C0    Virtual I/O Slot  vsa0
U9117.MMA.107769E-V5-C116  Virtual I/O Slot  vscsi0
U9117.MMA.107769E-V5-C163  Virtual I/O Slot  fcs0
U9117.MMA.107769E-V5-C164  Virtual I/O Slot  fcs1
U9117.MMA.107769E-V5-C226  Virtual I/O Slot  vscsi1
U9117.MMA.107769E-V5-C263  Virtual I/O Slot  fcs2
U9117.MMA.107769E-V5-C264  Virtual I/O Slot  fcs3
```

The NPIV virtual slots show up as fcs*X* devices. To view the NPIV WWNs assigned to the fcs*X* devices, run the **lscfg** AIX command on the client LPARs:

```
$ lscfg -vl fcs* | grep -E "fcs|Network Address"
  fcs0    U9117.MMA.107769E-V5-C163-T1  Virtual Fibre Channel Client Adapter
        Network Address.............C050760226B70008
  fcs2    U9117.MMA.107769E-V5-C263-T1  Virtual Fibre Channel Client Adapter
        Network Address.............C050760226B7000C
  fcs1    U9117.MMA.107769E-V5-C164-T1  Virtual Fibre Channel Client Adapter
        Network Address.............C050760226B7000A
  fcs3    U9117.MMA.107769E-V5-C264-T1  Virtual Fibre Channel Client Adapter
        Network Address.............C050760226B7000E
```

Comparing this output to the output of the **lssyscfg** HMC command above shows that AIX always sees the *first* WWN of the pair of WWNs assigned to a virtual FC client slot.

At this time, we do not yet see any disks through the NPIV layer:

```
$ lsdev -Ccdisk
hdisk0 Available  Virtual SCSI Disk Drive
# only the boot disk

$ mpio_get_config -Av
# nothing returned
```

This is the expected behavior, as we have not yet added the NPIV WWNs of the client LPARs to the appropriate host groups on the DS5300.

Creating DS5300 hosts and NPIV host ports for the client LPARs

In Example 6 on page 19, we have already created the DS5300 host groups, including two pairs of host groups for two different subprojects, which both use NPIV to access their LUNs:

- Host groups BCRS_NPIV1 and BCRS_NPIV2 for subproject BCRS
- Host groups IVT6_NPIV1 and IVT6_NPIV2 for subproject IVT6

We define these two subprojects to show that we can support multiple disjoint GPFS clusters (on separate groups of LPARs), all sharing the same physical FC connections between the DS5300 and the System p server. Each of these GPFS clusters will only able to access its own LUNs, even although all are using the same physical FC connections. In Example 7 on page 20, we map half of the GPFS LUNs to each of these two sub-projects.

After the NPIV client and server slots are configured and NPIV WWNs are assigned, the hosts and A/B host ports for the NPIV client LPARs need to be defined on the DS5300 storage subsystem. Note that we use two DS5300 storage partitions per client LPAR, so there are two DS5300 hosts and four host ports per client LPAR.

By associating the hosts with a host group in the **`create host`** DS5300 command, and by associating the host ports with a host in the **`create hostPort`** DS5300 command, the client LPARs automatically gain access to the LUNs that have already been mapped into that host group in "Mapping the LUNs to host groups" on page 20. Example 11 shows the host creation for our NPIV client LPARs.

Example 11 DS5300 create host commands for NPIV client LPARs

```
show "creating Client LPAR hosts for BCRS subproject...";
create host userLabel="BCRS-SRV01-1" hostGroup="BCRS_NPIV1" hostType="AIX";
create host userLabel="BCRS-SRV01-2" hostGroup="BCRS_NPIV2" hostType="AIX";
create host userLabel="BCRS-LS01-1"  hostGroup="BCRS_NPIV1" hostType="AIX";
create host userLabel="BCRS-LS01-2"  hostGroup="BCRS_NPIV2" hostType="AIX";
create host userLabel="BCRS-WAS01-1" hostGroup="BCRS_NPIV1" hostType="AIX";
create host userLabel="BCRS-WAS01-2" hostGroup="BCRS_NPIV2" hostType="AIX";
create host userLabel="BCRS-IHS01-1" hostGroup="BCRS_NPIV1" hostType="AIX";
create host userLabel="BCRS-IHS01-2" hostGroup="BCRS_NPIV2" hostType="AIX";
...

show "creating Client LPAR hosts for IVT6 subproject...";
create host  userLabel="IVT6-SRV01-1" hostGroup="IVT6_NPIV1" hostType="AIX";
create host  userLabel="IVT6-SRV01-2" hostGroup="IVT6_NPIV2" hostType="AIX";
...
```

To generate the host ports, the NPIV WWNs are needed. It is useful to check on the DS5300 if those WWNs are now visible. Use the **`SMcli`** command processor to run the **`show storageSubsystem hostTopology`** (which shows the full DS5300 topology of host groups, hosts, and host ports) or **`show allHostPorts`** (which only shows the host ports without their host or host group association) DS5300 commands. Example 12 shows the output of the **`show allHostPorts`** command after the NPIV WWNs have been generated on the server, but before the corresponding DS5300 host ports have been created:

Example 12 DS5300 show allHostPorts command output before creating the NPIV host ports

```
$ SMcli 9.152.35.11 9.152.35.12 -c "show allHostPorts;"

Performing syntax check...
Syntax check complete.
Executing script...
```

```
HOST PORT IDENTIFIER     HOST PORT NAME           HOST TYPE
10:00:00:00:c9:95:ae:1b  P01-DQD0D4H-P1-C1-T2     AIX
10:00:00:00:c9:95:ae:1a  P01-DQD0D4H-P1-C1-T1     AIX
10:00:00:00:c9:94:32:a2  P01-DQD0D3X-P1-C1-T1     AIX
10:00:00:00:c9:94:32:a3  P01-DQD0D3X-P1-C1-T2     AIX
10:00:00:00:c9:95:b0:9a  P01-DQD0D3W-P1-C1-T1     AIX
10:00:00:00:c9:95:b0:9b  P01-DQD0D3W-P1-C1-T2     AIX
c0:50:76:02:26:b7:00:08  Undefined                Undefined
c0:50:76:02:26:b7:00:0c  Undefined                Undefined
c0:50:76:02:26:b7:00:0a  Undefined                Undefined
c0:50:76:02:26:b7:00:0e  Undefined                Undefined
...

Script execution complete.
SMcli completed successfully.
```

If the NPIV WWNs can be seen by the DS5300, the corresponding DS5300 host ports can be created to complete the LUN mapping. Example 13 shows a short Perl script that runs the **`lssyscfg -r prof`** HMC command to list the virtual_fc_adapter attributes of the LPARs, and then generates suitable **`create hostPort`** DS5300 commands. (The DS5300 commands are only printed to stdout, not executed, so it is safe to invoke this script to test it.)

Note: We strongly recommend assigning meaningful and unique labels to the host ports. We store (most of) the NPIV virtual Fibre Channel client adapter location code in the DS5300 userLabel field, so we can easily correlate the DS5300 host port with the corresponding device on the System p side. (We have replaced the AIX location code prefix “U9117.MMA.” with the more mnemonic “P01-” to identify our server P01 by name.)

Example 13   Perl script to generate DS5300 create hostPort commands from virtual FCS slots

```
#!/usr/bin/perl -w
#
# parse virtual FCS slots of client LPARs (with names starting $PROJ),
# and transform into DS5300 create hostPort commands
#
my $HMC='9.152.35.8';
my $SRV='P01-107769E';
my $PROJ='^ivt6-bcrs'

foreach $_ (`ssh hscroot\@$HMC "lssyscfg -r prof -m ivt6-p01  -F lpar_name lpar_id virtual_fc_adapters
|grep $PROJ|sort"`) {
  my ($lpar_name,$lpar_id,$virtual_fc_adapters) = split(/ /);
  my @virtual_fc_adapter =
    ( $virtual_fc_adapters =~ /"""(.*)"",""(.*)"",""(.*)"",""(.*)"""/ );
  die "expected 4 vfc slots not found" unless ($#virtual_fc_adapter == 3);
  foreach $slot (  @virtual_fc_adapter ) {
    my ($virtual_slot_number, $client_or_server,
        $remote_lpar_ID, $remote_lpar_name, $remote_slot_number,
        $wwpns, $is_required) = split(/\//, $slot);
    my ($wwpn1,$wwpn2) = split (/,/, $wwpns);
    my $i=uc($wwpn1);
    my $u="$SRV-V$lpar_id-C$virtual_slot_number-T1";
    my $h=substr(uc($lpar_name),5) . ($virtual_slot_number % 2 ? '-1' : '-2');
    print
     "create hostPort identifier=\"$i\" userLabel=\"$u\" host=\"$h\" interfaceType=FC;\n";
  }
}
```

Example 14 shows an excerpt of the resulting DS5300 commands:

Example 14 DS5300 create hostPort commands for NPIV client LPARs

```
create hostPort identifier="C050760226B70008" userLabel="P01-107769E-V5-C163-T1"
host="BCRS-SRV01-1" interfaceType=FC;
create hostPort identifier="C050760226B7000C" userLabel="P01-107769E-V5-C263-T1"
host="BCRS-SRV01-1" interfaceType=FC;
create hostPort identifier="C050760226B7000A" userLabel="P01-107769E-V5-C164-T1"
host="BCRS-SRV01-2" interfaceType=FC;
create hostPort identifier="C050760226B7000E" userLabel="P01-107769E-V5-C264-T1"
host="BCRS-SRV01-2" interfaceType=FC;
```

After all the NPIV host ports are created, the DS5300 should no longer see any "unknown" WWNs, which can be verified by running the **show storageSubsystem hostTopology** or **show allHostPorts** DS5300 commands again. Example 15 shows the output of the **show allHostPorts** command after the DS5300 host ports have been created:

Example 15 DS5300 show allHostPorts command output after creating the NPIV host ports

```
$ SMcli 9.152.35.11 9.152.35.12 -c "show allHostPorts;"

Performing syntax check...
Syntax check complete.
Executing script...

HOST PORT IDENTIFIER     HOST PORT NAME          HOST TYPE
10:00:00:00:c9:95:ae:1b  P01-DQD0D4H-P1-C1-T2    AIX
10:00:00:00:c9:95:ae:1a  P01-DQD0D4H-P1-C1-T1    AIX
10:00:00:00:c9:94:32:a2  P01-DQD0D3X-P1-C1-T1    AIX
10:00:00:00:c9:94:32:a3  P01-DQD0D3X-P1-C1-T2    AIX
10:00:00:00:c9:95:b0:9a  P01-DQD0D3W-P1-C1-T1    AIX
10:00:00:00:c9:95:b0:9b  P01-DQD0D3W-P1-C1-T2    AIX
c0:50:76:02:26:b7:00:08  P01-107769E-V5-C163-T1  AIX
c0:50:76:02:26:b7:00:0c  P01-107769E-V5-C263-T1  AIX
c0:50:76:02:26:b7:00:0a  P01-107769E-V5-C164-T1  AIX
c0:50:76:02:26:b7:00:0e  P01-107769E-V5-C264-T1  AIX
...

Script execution complete.
SMcli completed successfully.
```

Discovering the LUNs on the client LPARs

Finally, the disks should appear on the client LPAR after rescanning the FC adapters with the **cfgmgr** AIX command. This device discovery and configuration takes some time, in our case, 100 seconds for the 70 LUNs mapped into one pair of NPIV1/NPIV2 host groups. After running **cfgmgr**, we can use the **lsdev** and **mpio_get_config** AIX commands to verify that the hdisks have been created and that the MPIO multipathing is set up as expected.

Example 16 shows the LUN discovery on the management partition ivt6-bcrs-srv01 for the “BCRS” subproject’s GPFS cluster.

Example 16 LUN discovery for the “BCRS” subproject

```
ssh root@ivt6-bcrs-srv01 # do the same on all client LPARs

$ cfgmgr -v
# runs 100s for our 70 LUNs, verbose output not shown

$ lsdev -Ccdisk
hdisk0  Available          Virtual SCSI Disk Drive
hdisk1  Available 63-T1-01 MPIO DS5100/5300 Disk
hdisk2  Available 63-T1-01 MPIO DS5100/5300 Disk
hdisk3  Available 63-T1-01 MPIO DS5100/5300 Disk
...
hdisk70 Available 64-T1-01 MPIO DS5100/5300 Disk

$ mpio_get_config -Av
Frame id 0:
    Storage Subsystem worldwide name: 60ab8006e51a00004ba1cc77
    Controller count: 2
    Partition count: 2
    Partition 0:
    Storage Subsystem Name = 'ivt6-ds01'
        hdisk      LUN #   Ownership          User Label
        hdisk31        1   A (preferred)      DS01M01L01
        hdisk32        3   A (preferred)      DS01M01L03
        hdisk33        5   A (preferred)      DS01M01L05
        hdisk34        7   A (preferred)      DS01M01L07
        hdisk35        9   A (preferred)      DS01M01L09
        hdisk36       11   B (preferred)      DS01M02L01
        ...
        hdisk70       79   B (preferred)      DS01M08L09
    Partition 1:
    Storage Subsystem Name = 'ivt6-ds01'
        hdisk      LUN #   Ownership          User Label
        hdisk1         1   A (preferred)      DS01D01L01
        hdisk2         3   A (preferred)      DS01D01L03
        hdisk3         5   A (preferred)      DS01D01L05
        hdisk4         7   A (preferred)      DS01D01L07
        hdisk5         9   A (preferred)      DS01D01L09
        hdisk6        11   B (preferred)      DS01D02L01
        ...
        hdisk30       59   B (preferred)      DS01D06L09

$ exit
```

Similarly, Example 17 shows the LUN discovery on the management partition ivt6-ivt6-srv01 for the “IBVT6“subproject’s GPFS cluster.

Example 17 LUN discovery for the “IVT6” subproject

```
ssh root@ivt6-ivt6-srv01

$ cfgmgr -v
```

```
# runs 100s for our 70 LUNs, verbose output not shown

$ lsdev -Ccdisk
hdisk0  Available          Virtual SCSI Disk Drive
hdisk1  Available 95-T1-01 MPIO DS5100/5300 Disk
hdisk2  Available 95-T1-01 MPIO DS5100/5300 Disk
...
hdisk70 Available 96-T1-01 MPIO DS5100/5300 Disk

$ mpio_get_config -Av
Frame id 0:
    Storage Subsystem worldwide name: 60ab8006e51a00004ba1cc77
    Controller count: 2
    Partition count: 2
    Partition 0:
    Storage Subsystem Name = 'ivt6-ds01'
        hdisk      LUN #   Ownership          User Label
        hdisk31        2   A (preferred)      DS01M01L02
        hdisk32        4   A (preferred)      DS01M01L04
        hdisk33        6   A (preferred)      DS01M01L06
        hdisk34        8   A (preferred)      DS01M01L08
        hdisk35       10   A (preferred)      DS01M01L10
        hdisk36       12   B (preferred)      DS01M02L02
        ...
        hdisk70       80   B (preferred)      DS01M08L10
    Partition 1:
    Storage Subsystem Name = 'ivt6-ds01'
        hdisk      LUN #   Ownership          User Label
        hdisk1         2   A (preferred)      DS01D01L02
        hdisk2         4   A (preferred)      DS01D01L04
        hdisk3         6   A (preferred)      DS01D01L06
        hdisk4         8   A (preferred)      DS01D01L08
        hdisk5        10   A (preferred)      DS01D01L10
        hdisk6        12   B (preferred)      DS01D02L02
        ...
        hdisk30       60   B (preferred)      DS01D06L10

$ exit
```

As expected, on the first LPAR we see all the odd-numbered DS5300 LUNs (as visible in the “LUN #” and “User Label” fields of the **mpio_get_config** AIX command output), while on the second LPAR we see all the even-numbered DS5300 LUNs. We have successfully mapped the DS5300 LUNs into two disjoint groups of NPIV client LPARs, sharing the same physical FC connections to the storage.

We recommend that on the first LPAR on which the LUNs are discovered, you should assign AIX physical volume IDs (PVIDs) to the hdisks using the **chdev** AIX command. These IDs are unique, and make it easier to verify on other LPARs if the same hdisk number identifies the same LUN or not.

```
# assign PVIDs to all MPIO DS5k hdisks:
lsdev -Ccdisk|grep "MPIO DS5"|while read HDISK rest_of_line
do
  echo "$HDISK: `chdev -l $HDISK -a pv=yes`"
done
```

```
# display PV IDs:
# (alternatively, use lsattr -El hdiskXX -a pvid)
lspv
```

Note: PVIDs are assigned for the convenience of the administrator. GPFS does not depend on identical hdisk numbers for the same LUN on all GPFS nodes. When creating a Network Shared Disk (NSD) on top of an AIX hdisk, GPFS assigns a unique NSD identifier that is written to the disk so that GPFS can always identify its NSDs regardless of the hdisk numbering.

Tuning the Fibre Channel and disk devices

After the LUNs have been discovered on the client LPARs, a number of AIX device driver settings should be verified, and adjusted if necessary. It is important to perform these steps on *each* client LPAR, as the AIX device information is stored locally in the Object Data Manager (ODM) database on each AIX LPAR.

Setting the hdisk reserve_policy

In GPFS environments with DS5300 storage subsystems and the AIX MPIO multipath driver, it is absolutely critical that the reserve_policy attribute of *all* GPFS hdisks is set to no_reserve. The default setting of single_path may cause the complete GPFS cluster to hang. Example 18 shows how to list the hdisks' reserve_policy attribute using the **lsdev** and **lsattr** AIX commands, and how to change it using the **chdev** AIX command. As the hdisks are already in the "active" state, the changes should be done by using the -P flag of the **chdev** command and then rebooting to activate the change.

Example 18 Displaying and setting the GPFS hdisk reserve_policy

```
ssh padmin@ivt6-bcrs-srv01 # repeat for all other LPARs

# list reserve_policy of all MPIO DS5k hdisks:
lsdev -Ccdisk|grep "MPIO DS5"|while read D rest_of_line
do
  echo "$D: `lsattr -El $D -a reserve_policy`"
done
hdisk1: reserve_policy single_path Reserve Policy True
hdisk2: reserve_policy single_path Reserve Policy True
...
hdisk70: reserve_policy single_path Reserve Policy True

# change reserve_policy of all MPIO DS5k hdisks (deferred):
lsdev -Ccdisk|grep "MPIO DS5"|while read D rest_of_line
do
  echo "$D: `chdev -l $D -a reserve_policy=no_reserve -P`"
done

# reboot to activate the deferred attribute change
# (you may want to do all the remaining tuning first, so only one reboot is needed)
shutdown -Fr
```

Note: Be aware that these steps need to be performed on *each* LPAR in the GPFS cluster; a single LPAR where this has not been done may cause a cluster-wide GPFS hang. As the VIO servers do not see the GPFS hdisks, there is no need to change any GPFS hdisk parameters on the VIO servers.

Tuning the command queue depth

The second important tunable is the SCSI command queue depth. This setting controls how many SCSI commands can be queued by the various components in the FC data path. For performance reasons, it is always advisable to have more than one command in flight for a given LUN, and the service queues need to be tuned to support this case. This case is particularly relevant for workloads with a high number of I/O operations per second (IOPS):

- For the `fcs`*X* devices, the parameter for the service queue is num_cmd_elems. It needs to be set both on the VIO servers (for the physical FC adapter ports), and on every client LPAR (for the virtual NPIV FC adapter ports).
- For the `hdisk`*XX* devices, the parameter for the service queue is `queue_depth`. It needs to be set only at the client LPARs, as the VIO servers do not see the GPFS hdisks.

In a GPFS configuration with the SAN-attached disk model, a large number of LPARs may issue commands against a large number of LUNs. As a storage subsystem only supports a limited command queue, it is possible that the desirable command queue depth in such scenarios is exhausting the capabilities of the storage subsystem. For example, section 3.4, "DS5100 and DS5300 storage subsystems", in *IBM Midrange System Storage Hardware Guide*, SG24-7676 states that the DS5300 supports a *total* of 4096 commands (with a maximum *drive* queue depth of 16).

Overrunning the command queue typically generates errors in the AIX error log, so it is advisable to run the **`errpt`** AIX command from time to time and monitor its output for resource shortages after having changed those tunables.

Tuning the fcs devices on the VIO servers

When planning for command queue depths settings, the first step is to consider if all the physical FC adapter ports connected to a single storage subsystem are carrying roughly the same load (in terms of number of commands), or if there are differences that should be reflected in the tunables. In our case, a GPFS file system is built over LUNs that are served over all four FC paths, and we have roughly the same number of LUNs on each of the four paths. In general, this would imply that we should assign 1/4 of the maximum queue depth to each of the four paths. However, we have set up two host groups: one for metadata LUNs and one for data LUNs. So, even with roughly the same number of LUNs, as metadata operations tend to be more IOPS dominated and data operations typically involve fewer but larger I/Os, it might be advisable to allocate a larger portion of the available total queue depth to the metadata host group. In our PoC setup, the usage pattern is not well-known, so in the end we simply assign a quarter of the total queue depth to each of the four physical FC adapter ports used for GPFS traffic (fcs2 and fcs3 on p01v01 and on p01v02).

Note: In the case of an FC path failure when an "A" or "B" path fails, the remaining FC adapter port serves both the LUNs for which it is the preferred path, *and* the LUNs for which it is the backup path. So, in the failure case, we want to be able to queue *twice* the number of commands against that fcs*X* device. However, it is then not easy to avoid overrunning the storage subsystems in the error-free state, so we keep the queue depth at the level that is appropriate for the error-free case.

Example 19 shows how to list the num_cmd_elems attributes of the physical FC adapter ports on one of the VIO servers. The -E option of the **lsattr** AIX command displays the current setting, and the -R option displays the list or range of allowable values. Note that the maximum supported value displayed with the -R option is 4096. As explained above, we change num_cmd_elems from the default of 500 to 1024 on the two devices fcs2 and fcs3 that are used for NPIV and GPFS.

Example 19 Displaying and setting the VIOS physical FC adapter ports' num_cmd_elems

```
ssh padmin@ivt6-p01v01 # repeat for ivt6-p01v02
$ oem_setup_env

# list num_cmd_elems of physical FC adapter ports used for NPIV and GPFS:
lsattr -El fcs2 -a num_cmd_elems
num_cmd_elems 500 Maximum number of COMMANDS to queue to the adapter True
lsattr -El fcs3 -a num_cmd_elems
num_cmd_elems 500 Maximum number of COMMANDS to queue to the adapter True

# list allowed values for num_cmd_elems:
lsattr -Rl fcs2 -a num_cmd_elems
20...4096 (+1)

# change num_cmd_elems of physical FC adapter ports used for NPIV and GPFS
(deferred):
chdev -l fcs2 -a num_cmd_elems=1024 -P
chdev -l fcs3 -a num_cmd_elems=1024 -P

# reboot to activate the deferred attribute change:
shutdown -Fr
```

Tuning the fcs devices on the NPIV clients

On the client LPAR side, a single LPAR uses all four physical FC adapter ports in the two VIO servers through its four virtual NPIV FC adapter ports fcs0 to fcs3, so those four devices could also be configured to a maximum queue depth of 1024 each. However, as all NPIV clients share the capabilities of the underlying physical FC adapter port, this configuration would likely overrun the physical adapter port in the VIO server when all NPIV clients are active.

The safest setting for the virtual NPIV FC adapter ports would be to divide the physical FC adapter port's maximum by the number of virtual NPIV ports that share this physical port. In our case, this calculation would result in a queue depth of roughly 1024/32=32, which is a very low value, especially if many LUNs are served over a single FC port (which is typically the case for GPFS configurations).

On the other hand, we know that our proof of concept supports two separate subprojects, which likely will not produce burst I/O traffic at exactly the same time. This would allow us to double the queue depth. Additionally, in each of the subprojects, we support multiple customers or tenants, each with a small subset of the LPARs assigned to it. To balance the overall system load, the applications at the higher layers could then schedule I/O intensive operations (such as batch data load or indexing) to happen at different times for different tenants. This usage scenario also justifies increasing the queue depth allocated to individual LPARs. These configurations are, of course, application dependent.

Example 20 how to query the queue depth for the NPIV adapter ports on one of the client LPARs. The default setting for the virtual FC adapter ports is 200, which we do not increase, as it roughly matches our above reasoning.

Example 20   Displaying the client LPAR virtual FC adapter ports' num_cmd_elems

```
ssh root@ivt6-bcrs-srv01 # repeat for all client LPARs

# list num_cmd_elems of virtual FC adapter ports (all used for NPIV and GPFS):
lsattr -El fcs0 -a num_cmd_elems
num_cmd_elems 200 Maximum number of COMMANDS to queue to the adapter True
lsattr -El fcs1 -a num_cmd_elems
num_cmd_elems 200 Maximum number of COMMANDS to queue to the adapter True
lsattr -El fcs2 -a num_cmd_elems
num_cmd_elems 200 Maximum number of COMMANDS to queue to the adapter True
lsattr -El fcs3 -a num_cmd_elems
num_cmd_elems 200 Maximum number of COMMANDS to queue to the adapter True

# list allowed values for num_cmd_elems:
lsattr -Rl fcs0 -a num_cmd_elems
20...2048 (+1)

exit
```

Tuning the hdisk devices on the NPIV clients

After tuning the FC adapter device drivers, the hdisk devices must be checked (and tuned if necessary). As mentioned above, this task only needs to be done on the client LPARs, as the VIO severs do not see the GPFS hdisks.

For the hdisks, the safest setting would be to divide the total number of commands supported by the DS5300 by the number of LUNs, and by the number of LPARs that see those LUNs. In our case, 16 LPARs of one subproject access 70 LUNs, and 16 LPARs of the second subproject access a different set of 70 LUNs, so the resulting hdisk queue depth would be 4096 / (16*70 + 16*70), which is roughly *two*. This is definitely too small to achieve decent performance (and clearly shows that we are oversubscribing our hardware, which is acceptable for our PoC tests but not recommended for a production environment).

Similar to the tuning of the NPIV fcs devices, we can factor in some properties of our PoC setup (which would not be implemented this way in a production environment). We use the same hardware for two subprojects, which typically will not be active at the same time, and we configure multiple GPFS file systems on disjoint subsets of the 70 LUNs, which also are not active at the same time. So in this special setup, it should be safe to increase the hdisk queue depth.

Example 21 shows how to list the hdisk queue_depth on one of the client LPARs. The default is 10, which is a factor of 5 larger than our calculation. Given the above statements about the PoC setup, we do not change the default.

Example 21   Displaying the client LPAR hdisks' queue_depth

```
ssh padmin@ivt6-bcrs-srv01 # repeat for all other client LPARs

# list queue_depth of all MPIO DS5k hdisks:
lsdev -Ccdisk|grep "MPIO DS5"|while read D rest_of_line
do
  echo "$D: `lsattr -El $D -a queue_depth`"
```

```
done
hdisk1: queue_depth 10 Queue DEPTH True
hdisk2: queue_depth 10 Queue DEPTH True
...
hdisk70: queue_depth 10 Queue DEPTH True

# list allowable values of queue_depth:
lsattr -Rl hdisk1 -a queue_depth
1...256 (+1)

exit
```

Tuning the I/O transfer size

Another important tunable is the maximum I/O transfer size. This setting controls how big an I/O request can be without being fragmented by the various device drivers in the data path. Ideally, all elements in the data path should support buffers of the size of the GPFS blocksize. Avoiding I/O buffer fragmentation is particularly relevant for workloads that require high bandwidth (GBps), and in environments that use SATA disks, as those environments perform best with large GPFS block sizes (1 MiB to 4 MiB).

- For the `fcs`*X* devices, the parameter for the I/O transfer size is max_xfer_size. It needs to be set both on the VIO servers (for the physical FC adapter ports), and on every client LPAR (for the virtual NPIV FC adapter ports).
- For the `hdisk`*XX* devices, the parameter for the I/O transfer size is max_transfer. It needs to be set only at the client LPARs, as the VIO servers do not see the GPFS hdisks.

Example 22 shows how to list the max_xfer_size attribute of the physical FC adapter ports on one of the VIO servers. We change it from the default of 0x100000 (1 MiB) to 0x200000 (2 MiB) on fcs2 and fcs3, which are used for NPIV and GPFS.

Example 22 Displaying and setting the VIOS physical FC adapter ports' max_xfer_size

```
ssh padmin@ivt6-p01v01 # repeat for ivt6-p01v02
oem_setup_env

lsattr -El fcs2 -a max_xfer_size
max_xfer_size 0x100000 Maximum Transfer Size True
lsattr -El fcs3 -a max_xfer_size
max_xfer_size 0x100000 Maximum Transfer Size True

$ lsattr -Rl fcs0 -a max_xfer_size | paste -s -
0x100000        0x200000        0x400000        0x800000        0x1000000

chdev -l fcs2 -a max_xfer_size=0x200000 -P
chdev -l fcs3 -a max_xfer_size=0x200000 -P

# reboot to activate the deferred attribute change:
shutdown -Fr
```

Example 23 shows the same commands for one of the client LPARs, again increasing the maximum transfer size from 1 MiB to 2 MiB.

Example 23 Displaying and setting the client LPAR virtual FC adapter ports' max_xfer_size

```
ssh root@ivt6-bcrs-srv01 # repeat for all other client LPARs

lsattr -El fcs0 -a max_xfer_size
max_xfer_size 0x100000 Maximum Transfer Size True
lsattr -El fcs1 -a max_xfer_size
max_xfer_size 0x100000 Maximum Transfer Size True
lsattr -El fcs2 -a max_xfer_size
max_xfer_size 0x100000 Maximum Transfer Size True
lsattr -El fcs3 -a max_xfer_size
max_xfer_size 0x100000 Maximum Transfer Size True

lsattr -Rl fcs0 -a max_xfer_size | paste -s -
0x100000        0x200000        0x400000        0x800000        0x1000000

chdev -l fcs0 -a max_xfer_size=0x200000 -P
chdev -l fcs1 -a max_xfer_size=0x200000 -P
chdev -l fcs2 -a max_xfer_size=0x200000 -P
chdev -l fcs3 -a max_xfer_size=0x200000 -P

# reboot to activate the deferred attribute change:
shutdown -Fr
```

Example 24 shows how to list and change the hdisk maximum transfer size on the client LPARs. The default is 0x40000 (256 kiB). We increase it to 0x200000 (2 MiB), in line with the increased values of the fcs*X* devices.

Example 24 Displaying and setting the client LPAR hdisks' max_transfer

```
ssh root@ivt6-bcrs-srv01 # repeat for all other client LPARs

lsdev -Ccdisk|grep "MPIO DS5"|while read D rest_of_line
do
  echo "$D: `lsattr -El $D -a max_transfer`"
done
hdisk1: max_transfer 0x40000 Maximum TRANSFER Size True
hdisk2: max_transfer 0x40000 Maximum TRANSFER Size True
...
hdisk70: max_transfer 0x40000 Maximum TRANSFER Size True

lsattr -Rl hdisk1 -a max_transfer | paste -s -d' ' -
0x20000 0x40000 0x80000 0x100000 0x200000 0x400000 0x800000 0x1000000

lsdev -Ccdisk|grep "MPIO DS5"|while read D rest_of_line
do
  echo "$D: `chdev -l $D -a max_transfer=0x200000 -P`"
done

# reboot to activate the deferred attribute change:
shutdown -Fr
```

GPFS configuration

With the NPIV setup completed, configuring GPFS is identical to the configuration of any other GPFS cluster with SAN-attached disks. This configuration is described in detail in the GPFS manuals, in particular in *GPFS V3.3 Concepts, Planning, and Installation Guide*, GA76-0413. Here we only outline the high-level steps and some aspects that are specific to our use of GPFS in a virtualization environment.

Setting up GPFS management partitions

GPFS requires password-less root access from one or more GPFS nodes to all other GPFS nodes in the cluster through a remote shell command such as **`rsh`**, and a remote copy command such as **`rcp`**. This is usually implemented through the secure shell (SSH), with its **`ssh`** and **`scp`** commands, using public key authentication. This access is required to distribute GPFS configuration files across the cluster, and for cluster-wide management functions, such as starting up the mmfsd GPFS daemon on remote nodes.

In an HPC cluster, typically one or more of the already existing GPFS nodes are used as the GPFS management node(s). In a virtualization environment, where different LPARs are likely to host services for different tenants or customers, root access across those LPARs is generally not acceptable. We recommend creating two dedicated GPFS management LPARs, which will not host any applications, and will be set up to allow root SSH access to all client LPARs, but not from the client LPARs to the GPFS management LPARs or to other client LPARs.

Note: GPFS V3.2 and earlier required any-to-any remote shell execution, and GPFS management commands could be run from any GPFS node. Since GPFS V3.3, the administration mode described above is also supported. GPFS management commands can then only be executed from one or a few central nodes to all other GPFS nodes. The **`mmchconfig`** GPFS command can be used to set the desired behavior by setting the adminMode parameter to “allToAll” or “central”. We recommend using “central”, which is the default since Version 3.3.

We also designate those two management LPARs as the GPFS primary and secondary cluster configuration servers, so we do not need to perform these special GPFS management functions on one of the client/application LPARs.

GPFS quorum planning

GPFS uses a quorum mechanism to ensure consistency. There are two GPFS quorum mechanisms: *node quorum* and *tiebreaker disks*. Refer to the GPFS documentation for details. Typically, node quorum is used in GPFS clusters with many nodes, and tiebreaker disks are used when the cluster consists of only a few nodes.

In a virtualization environment where LPARs might be frequently added or removed from the configuration (“thin provisioning”), we recommend using the tiebreaker disk mechanism with tiebreaker disks connected to the dedicated GPFS management LPARs. Using tiebreaker disks keeps the GPFS cluster operational regardless of the state of the client LPARs, as long as one of the management LPARs is up and has access to a majority of the tiebreaker disks. While node quorum could be used when a large number of LPARs are configured as GPFS nodes, you need to pay close attention to not lose your quorum when provisioning large numbers of new LPARs.

In the proof of concept, we do not consider quorum and just designate the primary configuration server as the only GPFS quorum node; GPFS will be up if the LPAR is up, and will be shut down otherwise. Using node quorum with only a single quorum node is strongly discouraged for a production environment because that node would become a single point of failure, but it is acceptable for our testing environment, which focuses on other topics.

GPFS prerequisites

A number of prerequisite steps must be performed before setting up GPFS. These steps are described in the GPFS manuals, and do not differ in a virtualization environment:

1. Set up TCP/IP network connectivity, `/etc/hosts`, and DNS name resolution.
2. Configure SSH, using the central or allToAll mode as described above.
3. Establish a consistent time across the GPFS cluster (for example, using NTP).

When using the GPFS SAN attachment model, the SAN connectivity also has to be established and all LUNs intended for GPFS need to be visible on all GPFS nodes before creating the NSDs. Ensuring that all LUNs are visible on all GPFS nodes by using NPIV virtualization has been the main topic of this paper. After installing the GPFS software packages and latest PTFs (using the **`installp`** AIX command), the GPFS cluster can be set up.

GPFS cluster creation

We create the GPFS cluster on our ivt6-bcrs-srv01 management partition using the **`mmcrcluster`** and **`mmchlicense`** GPFS commands to designate the node as a GPFS server under the GPFS licensing model. We also set some configuration parameters using the **`mmchconfig`** GPFS command (you need to adapt these settings to your own environment), and then start GPFS on this LPAR using the **`mmstartup`** GPFS command:

```
ssh root@ivt6-bcrs-srv01

mmcrcluster \
  -N "ivt6-bcrs-srv01g:manager-quorum:ivt6-bcrs-srv01g" \
  -p ivt6-bcrs-srv01g.boeblingen.de.ibm.com \
  -r /usr/bin/ssh -R /usr/bin/scp \
  -C ivt6-bcrs.boeblingen.de.ibm.com \
  -U ivt6-bcrs.boeblingen.de.ibm.com \
  -A
mmchlicense server --accept -N ivt6-32g  # same node as ivt6-bcrs-srv01g
mmchconfig maxblocksize=4M,pagepool=256M,worker1Threads=256
mmstartup
mmgetstate
# Node number  Node name        GPFS state
#------------------------------------------
#       1      ivt6-32g         active
```

GPFS is now up and running, and the next step is to create GPFS *Network Shared Disks* (NSDs) on the LUNs that are intended for GPFS usage.

NSD creation

In “Array and LUN creation” on page 16, we recommend as a best practice to create LUN names on the storage subsystems that are globally unique across the GPFS cluster. If this is the case, we can reuse those LUN names as the names of the GPFS NSDs, which also need to be globally unique. When using DS5300 and the AIX MPIO driver, we can create the GPFS disk descriptor file (which is needed as input to the **`mmcrnsd`** GPFS command) by parsing the output of the **`mpio_get_config`** AIX command and associating hdisk*XX* device names (as the GPFS DiskName) with LUN names (as the GPFS DesiredName). Here we designate all LUNs named DS01D*xxxx* as GPFS *dataOnly* disks and all LUNs named DS01M*xxxx* as

GPFS *metadataOnly* disks. Like the GPFS cluster creation, we create the NSDs on the ivt6-bcrs-srv01 GPFS management partition:

```
# descriptor=DiskName:ServerList::DiskUsage:FailureGroup:DesiredName:StoragePool
#   DiskUsage={dataAndMetadata|dataOnly|metaDataOnly|descOnly}
#   StoragePool={system|<user-defined>} (only dataOnly disks can be user-defined)
rm /tmp/nsdlist-bcrs.txt # start with empty disk descriptor file

mpio_get_config -Av|grep DS01D|while read HDISK L AB P NAME
do
 echo "$HDISK:::dataOnly:0101:$NAME:sata1tb_4p"
done >> /tmp/nsdlist-bcrs.txt

mpio_get_config -Av|grep DS01M|while read HDISK L AB P NAME
do
 echo "$HDISK:::metadataOnly:0101:$NAME:system"
done >> /tmp/nsdlist-bcrs.txt

cat /tmp/nsdlist-bcrs.txt  #  verify that the descriptors are what is intended...

mmcrnsd -F /tmp/nsdlist-bcrs.txt
```

Note: We have placed the dataOnly NSDs into a non-default GPFS *storage pool* that we named “sata1tb_4p” (the LUNs reside on 1 TB SATA disks configured as RAID 5 4+P arrays). This approach is useful, for example, when 2 TB drives or other RAID levels are added later, and we want to control which files go onto which LUNs. We can use the storage pool names with the GPFS policy engine to specify file placement and migration rules. By default, all NSDs would be placed into the *system* pool, which also holds the metadataOnly disks.

You should then check that the NSDs have been created correctly by running the **mmlsnsd** GPFS command. Without any options, this command displays the file systems and NSD servers (if any) for all NSDs:

```
mmlsnsd
# File system   Disk name    NSD servers
#---------------------------------------------------------------------------
# (free disk)   DS01D01L01   (directly attached)
# (free disk)   DS01D01L03   (directly attached)
...
# (free disk)   DS01M08L07   (directly attached)
# (free disk)   DS01M08L09   (directly attached)
```

With the -L or -m option, the NSD volume IDs are also displayed. These IDs are generated by GPFS and are written to the disks, so GPFS can always identify its NSDs regardless of the hdisk*XX* numbering. With the -m option, the local hdisk*XX* device names of the NSDs are also displayed in addition to the NSD volume IDs:

```
mmlsnsd -m
# Disk name    NSD volume ID      Device         Node name                Remarks
#---------------------------------------------------------------------------------
# DS01D01L01   099823264BD869D7   /dev/hdisk1    ivt6-32g
# DS01D01L03   099823264BD869D8   /dev/hdisk2    ivt6-32g
...
# DS01M08L07   099823264BD86A1B   /dev/hdisk69   ivt6-32g
# DS01M08L09   099823264BD86A1C   /dev/hdisk70   ivt6-32g
```

File system creation

GPFS file systems are created on top of the GPFS NSDs. The **mmcrnsd** GPFS command executed in the previous section modified the contents of its input file, so it now contains the information for the created NSDs in a format suitable for use by the **mmcrfs** GPFS command. For our PoC, we create multiple LUNs on each DS5300 disk array, and we generate different file systems on the different LUNs. Here we use all LUNs named DS01*xxx*L01 to generate the first file system, named bcrs-was:

```
grep L01 /tmp/nsdlist-bcrs.txt | tee /tmp/nsdlist-bcrs-L01.txt

mmcrfs /gpfs/bcrs-was bcrs-was -F /tmp/nsdlist-bcrs-L01.txt \
  -A yes -D posix -B 256K -E yes -k posix \
  -m 1 -M 2 -n 33 -N 50M -Q no -r 1 -R 2 -S no -v yes -z no

mmlsfs bcrs-was
mmmount all
mount|grep mmfs
# /dev/bcrs-was  /gpfs/bcrs-was  mmfs  Apr 28 15:57 rw,mtime,atime,dev=bcrs-was
```

Note that the use of a non-default storage pool for the dataOnly NSDs requires that a GPFS file placement policy be installed, even if there is only one storage pool that holds dataOnly NSDs. The default file placement is into the system pool, so without a file placement rule you cannot create files in the file system (0 byte files can still be created, as they do not need data blocks). Policies are installed by putting the policy rules into a text file, and then installing those rule definitions by using the **mmchpolicy** GPFS command. It is always a good idea to test the new rules using the **-I** test option before installing it with the **-I** yes option (capital i), and to list the result with the **mmlspolicy** GPFS command after installing them:

```
echo "RULE 'sata1tb_4p' SET POOL 'sata1tb_4p'\n" > /tmp/bcrs-was.policy

mmchpolicy bcrs-was /tmp/bcrs-was.policy -t "GPFS policies for bcrs-was" -I test
mmchpolicy bcrs-was /tmp/bcrs-was.policy -t "GPFS policies for bcrs-was" -I yes
mmlspolicy bcrs-was -L
mmlspolicy bcrs-was
```

We repeat these steps for each “BCRS” file system, using the LUNs L03, L05, L07, and L09.

Adding a client LPAR to the cluster

The GPFS cluster is now completely set up with the GPFS management partition as the only GPFS node in the GPFS cluster. Because the NPIV setup has already been performed for all client LPARs, adding them as new GPFS nodes is as simple as installing the GPFS software and verifying the GPFS prerequisites, and then adding the node with the **mmaddnode** and **mmchlicense** GPFS commands. GPFS will be started at then next reboot, or can be manually started using the **mmstartup** GPFS command:

```
mmaddnode -N "ivt6-34g:client-nonquorum:ivt6-34g"
mmaddnode -N "ivt6-35g:client-nonquorum:ivt6-35g"
mmaddnode -N "ivt6-36g:client-nonquorum:ivt6-36g"

mmchlicense client --accept -N ivt6-34g,ivt6-35g,ivt6-36g

mmstartup -a
mmgetstate -a
#
# Node number  Node name        GPFS state
#------------------------------------------
#       1      ivt6-32g         active
#       2      ivt6-34g         active
#       3      ivt6-35g         active
#       4      ivt6-36g         active
```

Note that we designate all client LPARs as “nonquorum”, as we plan to use tiebreaker disks for quorum (see “GPFS quorum planning” on page 42). We also designate the client LPARs as “client”, which prevents the GPFS daemon on those nodes from performing server functions, and places all server load on the GPFS management partition that is designated “server”. In a production environment, this configuration will most likely not be appropriate, so depending on the workload, more GPFS nodes may need to be designated as “server”.

Provisioning scenarios

In the previous sections, we discussed the initial setup of the hardware and software components. In this section, we briefly outline the steps needed to add more components at a later time. There are three typical cases where you add new hardware:

- Adding disks to an existing storage subsystem. “Array and LUN creation” on page 16, “Mapping the LUNs to host groups” on page 20, and “Discovering the LUNs on the client LPARs” on page 33 describe how new LUNs on these disks can be made available to the client LPARS, with subsequent device tuning and GPFS setup as described above.
- Adding more DS5300 disk subsystems. Depending on your performance needs, this task can be done with or without scaling the Fibre Channel infrastructure on the server side:
  - Without adding more FC paths on the server side, the host ports of the new DS5300 are connected to the existing SAN zones, and use the same physical and virtual FC connections in the System p server. “DS5300 setup” on page 15 and “Creating DS5300 hosts and NPIV host ports for the client LPARs” on page 31 describe the required DS5300 setup. After the disk subsystem has been integrated, the LUNs need to be discovered on the client LPARs and can then be tuned and used within GPFS.
  - When the server side FC connectivity is scaled up proportionally to the new storage subsystem, additional physical FC adapters would be added to the two VIO servers. These servers would be connected to the new DS5300 through separate SAN zones (so additional SAN switches may be needed to provide more ports). In addition to the DS5300 setup, we then also need to mirror the steps of “NPIV setup” on page 23 to virtualize the new FC paths. All LUNs of the new DS5300 would then be accessed through those new FC paths, scaling up the client-side bandwidth.
- Adding more physical System p servers, with their own VIO servers. Again, this can be done with or without scaling the Fibre Channel infrastructure on the DS5300 side:
  - Without adding more FC host ports on the DS5300 side, the FC adapters of the new server’s VIO servers are connected to the existing SAN zones and DS5300 host groups. After performing the steps in “VIO server setup and basic LPAR setup” on page 22 for the new server, perform the steps in “NPIV setup” on page 23 to enable the new server to access the storage using the already existing DS5300 host ports.
  - When there are still free host ports on the DS5300, you may want to use those ports to provide more aggregate bandwidth from the DS5300 to the SAN switch. In addition to the server and NPIV setup, new SAN zones will be needed on the SAN switch (see “SAN switch setup” on page 13 for configuration details). Those zones contain the newly added server’s physical FCS adapters and the matching new DS5300 host ports. On the DS5300, the new VIO servers become members of the same host groups as the VIO servers of the first server.

When adding more virtual resources (LPARs, or new subprojects with separate DS5300 host groups for GPFS LUN/file system isolation), the steps in “Configuring the infrastructure” on page 13 can be performed as described for the initial setup.

Removing or migrating storage

One important manageability feature of GPFS is its ability to transparently *add* and *remove* disks from a GPFS file system while the file system is active. This function is very important in a dynamic infrastructure where new workload may be added over time, and depreciated hardware needs to be decommissioned and replaced with new hardware. The two primary GPFS commands to add and remove disks from a GPFS file system are **`mmaddisk`** and **`mmdeldisk`**:

- When a disk is added to a file system with **`mmaddisk`**, it is empty initially and will be used for block allocations for newly created files. To rebalance the complete file system (including already existing files), the **`mmrestripefs`** GPFS command can be used. This command is important to achieve optimal bandwidth for large files, which are striped across all active NSDs at the time they are created. Such files need to be restriped to benefit from the additional bandwidth from new NSDs (and there also is a **`mmrestripefile`** command that acts on individual files). For small files that reside on only one or a few NSDs, the file system will be automatically balanced over time.
- When deleting a disk from a live file system by using the **`mmdeldisk`** command, file data residing on this disk needs to be moved to free space on other disks before the disk can be removed, so that sufficient free space needs to be available on other disks for the **`mmdeldisk`** command to be successful. It should also be noted that a previously balanced file system may become unbalanced, so it may be useful to run **`mmrestripefs`** after deleting disks.

There is also a **`mmrpldisk`** command that does a *replacement* of a GPFS disk. This command differs from running **`mmaddisk`** and **`mmdeldisk`** in sequence, in that it avoids any imbalance; data blocks will be copied directly from the disk to be removed to the disk that replaces it, keeping the file striping intact and avoiding the need to rebalance the file system.

Note: In GPFS, there are two sets of commands that deal with disks:

- The **`mm*nsd`** commands deal with the NSD layer, for example, which physical disks are used for an NSD, which *failure groups* they belong to, which NSD servers will serve it, and whether it holds data, metadata, or both.
- The **`mm*disk`** commands use the already established NSDs, and they control the allocation of NSDs to file systems (addition, removal) and their state in the file system.

Summary and outlook

GPFS is at the core of a growing number of IBM cloud storage solutions. Prominent examples are the IBM Scale Out Network Attached Storage (SONAS) appliance, which is described in detail in *IBM Scale Out Network Attached Storage Concepts*, SG24-7874 and *IBM Scale Out Network Attached Storage Architecture and Implementation*, SG24-7875, and the IBM Smart Business Storage Cloud, which is described at `http://www.ibm.com/services/storage`.

In this paper, we showed how to implement GPFS in a cloud environment where NPIV storage virtualization is used to provide SAN access from the LPARs to the LUNs on DS5300 midrange storage subsystems. This work complements the above storage solutions, as it enables us to build cloud solutions that utilize many of the enhanced GPFS features that are not made available externally by a storage appliance like SONAS.

Our motivation is to build tightly integrated ECM cloud solutions, where user interaction is at a higher software layer than the file system layer. GPFS is used within these solutions to provide a high performance and cost effective storage infrastructure, similar to its use in cloud storage solutions. In addition, we use enhanced GPFS features, such as file sets, snapshots, and fine-grain control over LUN layout and file system tuning to achieve a tighter integration with the ECM software stack than would be possible if we simply interface to an NAS filer. ECM cloud solutions include software components like IBM DB2® (see *Best Practices for DB2 on AIX 6.1 for POWER Systems*, SG24-7821), Content Manager (see *Content Manager Implementation and Migration Cookbook*, SG24-7051) and IBM WebSphere® Application Server (see *WebSphere Application Server V7: Concepts, Planning and Design*, SG24-7708). In a highly available cloud environment, all of these components require access to some form of shared storage, but the individual components have vastly different performance and functionality requirements, which can be addressed much better when the native GPFS features are used.

The team who wrote this Redpaper

This Redpaper was produced by a team of specialists from around the world working at the International Technical Support Organization, Poughkeepsie Center.

Michael Hennecke is a The Open Group Master Certified IT Specialist and HPC Systems Architect in the Deep Computing technical sales team at IBM Germany. He has 19 years of experience in High Performance Computing, including 7 years at IBM. His primary focus area is the design of large HPC clusters based on IBM System p and IBM System Blue Gene® technology, which typically include GPFS. Michael has designed and implemented large GPFS solutions since GPFS V1.2, and has recently deployed a large GPFS storage cluster with more than 5 PB of DS5300 based storage and POWER6 servers.

David Lebutsch is a 'The Open Group' Master Certified IT Specialist and Enterprise Content Management Solution Architect with 12 years experience in designing complex Content Management and compliance solutions for large global companies. David is a leader in the ECM Architecture & Solutions team of the IBM Research and Development Lab Germany. He is responsible for the design and implementation of first of their kind and complex ECM solutions. Most recently, he has been focusing on the development of the IBM ECM cloud offerings. Before joining the IBM Germany Research and Development Lab, he worked for IBM SWG Lab Services in the USA and Austria.

Stefan Schleipen works as an IT Specialist for IBM Research and Development in Germany and is a member of the Archive Cloud Development Team. His main working area is infrastructure and platform development with a strong focus on virtualization. He works mainly with IBM PowerVM and RHEL/KVM and develops dynamic deployment systems, thin provisioning systems, and management and monitoring of virtualized systems. Stefan also has experience with IBM Storage Systems and GPFS.

Thanks to the following people for their contributions to this project:

Dino Quintero, Alfred Schwab, Michael Schwartz, Wade Wallace
International Technical Support Organization, Poughkeepsie Center

Cataldo Mega
IBM Boeblingen

Gordon McPheeters
IBM Poughkeepsie

Maureen Largay and Ernie Obrist
IBM USA

Related publications

The publications listed in this section contain more detailed information about the infrastructure components (hardware and software) used in this proof of concept.

IBM Redbooks

You can search for, view, or download Redbooks, IBM Redpapers™ publications, Technotes, draft publications and additional materials, as well as order hardcopy Redbooks publications, at the following address:

http://www.redbooks.ibm.com/

- *Best Practices for DB2 on AIX 6.1 for POWER Systems*, SG24-7821
- *Content Manager Implementation and Migration Cookbook*, SG24-7051
- *Deploying Oracle 10g RAC on AIX V5 with GPFS*, SG24-7541
- *Hardware Management Console V7 Handbook*, SG24-7491
- *IBM Enterprise Content Management and System Storage Solutions: Working Together*, SG24-7558
- *IBM Midrange System Storage Hardware Guide*, SG24-7676
- *IBM Midrange System Storage Implementation and Best Practices Guide*, SG24-6363
- *IBM Power 750 and 755 Technical Overview and Introduction*, REDP-4638
- *IBM Power 770 and 780 Technical Overview and Introduction*, REDP-4639
- *IBM PowerVM Virtualization Managing and Monitoring*, SG24-7590
- *IBM Scale Out Network Attached Storage Architecture and Implementation*, SG24-7875
- *IBM Scale Out Network Attached Storage Concepts*, SG24-7874
- *IBM System p 570 Technical Overview and Introduction*, REDP-4405
- *Implementing an IBM/Brocade SAN with 8 Gbps Directors and Switches*, SG24-6116
- *Integrated Virtual Ethernet Adapter Technical Overview and Introduction*, REDP-4340
- *PowerVM Migration from Physical to Virtual Storage*, SG24-7825
- *PowerVM Virtualization on IBM System p: Introduction and Configuration Fourth Edition*, SG24-7940
- *WebSphere Application Server V7: Concepts, Planning and Design*, SG24-7708

GPFS manuals

The GPFS product manuals can be found in the IBM Cluster Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/topic/com.ibm.cluster.gpfs.doc/gpfsbooks.html

- *GPFS V3.3 Administration and Programming Reference*, SA23-2221
- *GPFS V3.3 Advanced Administration Guide*, SC23-5182

- *GPFS V3.3 Concepts, Planning, and Installation Guide*, GA76-0413
- *GPFS V3.3 Data Management API Guide*, GA76-0414
- *GPFS V3.3 Problem Determination Guide*, GA76-0415

Notices

This document REDP-4682-00 was created or updated on September 28, 2010.

Send us your comments in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:
  ibm.com/redbooks
- Send your comments in an email to:
  redbooks@us.ibm.com
- Mail your comments to:
  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400 U.S.A.

Redpaper™

Trademarks