TOPICS: [GPFS](http://www.aixhealthcheck.com/blog?topic=25)

## [Not properly balanced or replicated](http://www.aixhealthcheck.com/blog?id=59)

If a GPFS filesystem is changed (e.g. a disk error or caused by adding a disk) it is possible that a filesystem becomes no longer properly "balanced" or "replicated" (see the output of the mmlsdisk command). In GPFS 2.2 you could resolve this issue by running the mmrestripefs command. From GPFS 2.3 and onwards, the GPFS filesystems will balance themselves in a period of several weeks when data was changed. The messages about unbalanced file systems have disappeared in GPFS 3.2 alltogether. Because of this, do not run the mmrestripefs -b command on GPFS filesystems in GPFS 2.3. This may cause an unstable situation in which nodes could become unreachable.

A not properly replicated filesystem does need to be restriped however. Do this by running the mmrestripefs -r command.

TOPICS: [GPFS](http://www.aixhealthcheck.com/blog?topic=25)[↑](http://www.aixhealthcheck.com/blog?topic=25#container)

## [Viewing the GPFS disks](http://www.aixhealthcheck.com/blog?id=58)

Use the mmlsdisk command to view the disks in a GPFS filesystem. Use the mmcrlv command to create GPFS volume groups and disks. For example:

# mmlsdisk /dev/slvdata01208

TOPICS: [GPFS](http://www.aixhealthcheck.com/blog?topic=25)[↑](http://www.aixhealthcheck.com/blog?topic=25#container)

## [Viewing and modifying GPFS file systems](http://www.aixhealthcheck.com/blog?id=57)

Use the mmlsfs command to view the attributes and values of a GPFS file system. Use mmchfs to change the attributes of a GPFS file system. For example:

# mmlsfs /dev/slvdata01208

TOPICS: [GPFS](http://www.aixhealthcheck.com/blog?topic=25)[↑](http://www.aixhealthcheck.com/blog?topic=25#container)

## [View the GPFS nodes](http://www.aixhealthcheck.com/blog?id=56)

Use the mmlsnode command to view the nodes in your GPFS nodesets:

# mmlsnode -a

TOPICS: [GPFS](http://www.aixhealthcheck.com/blog?topic=25)[↑](http://www.aixhealthcheck.com/blog?topic=25#container)

## [View the GPFS file systems](http://www.aixhealthcheck.com/blog?id=55)

Use the mmlsconfig command to view which GPFS file systems are defined on your system.

TOPICS: [GPFS](http://www.aixhealthcheck.com/blog?topic=25)[↑](http://www.aixhealthcheck.com/blog?topic=25#container)

## [Where's the config of GPFS stored?](http://www.aixhealthcheck.com/blog?id=54)

In file /var/mmfs/gen/mmsdrfs. In a cluster, one node is primary and another node is secondary cluster data server. For a GPFS change, both nodes need to be active. For starting a GPFS node, a least one of the two need to be active (but this is no problem, since GPFS doesn't work with only 1 node).

TOPICS: [GPFS](http://www.aixhealthcheck.com/blog?topic=25)[↑](http://www.aixhealthcheck.com/blog?topic=25#container)

## [GPFS & FSCK](http://www.aixhealthcheck.com/blog?id=52)

There's a special GPFS command for FSCK of a MMFS filesystem: mmfsck.

It works the same way as a normal fsck: it will only show the lost blocks. For all other checks and repairs, an unmount of the filesystem is necessary.

TOPICS: [GPFS](http://www.aixhealthcheck.com/blog?topic=25)[↑](http://www.aixhealthcheck.com/blog?topic=25#container)

## [GPFS introduction](http://www.aixhealthcheck.com/blog?id=51)

GPFS is a concurrent file system. It is a product of [IBM](http://www.ibm.com/) and is short for General Parallel File System. It is a high-performance shared-disk file system that can provide fast data access from all nodes in a homogenous or heterogenous cluster of IBM UNIX servers running either the AIX or the Linux operating system.

All nodes in a GPFS cluster have the same GPFS journaled filesystem mounted, allowing multiple nodes to be active at the same time on the same data.

A specific use for GPFS is RAC, [Oracle](http://www.oracle.com/)'s Real Application Cluster. In a RAC cluster multiple instances are active (sharing the workload) and provide a near "Allways-On" database operation. The Oracle RAC software relies on [IBM](http://www.ibm.com/)'s HACMP software to achieve high availability for hardware and the operating system AIX. For storage it utilizes the concurrent filesystem called GPFS.

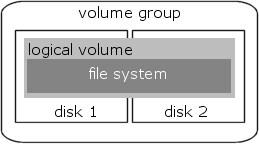
Data availability

GPFS is fault tolerant and can be configured for continued access to data even if cluster nodes or storage systems fail. This is accomplished though robust clustering features and support for data replication. GPFS continuously monitors the health of the file system components. When failures are detected appropriate recovery action is taken automatically. Extensive logging and recovery capabilities are provided which maintain metadata consistency when application nodes holding locks or performing services fail. Data replication is available for journal logs, metadata and data. Replication allows for continuous operation even if a path to a disk or a disk itself fails. GPFS Version 3.2 further enhances clustering robustness with connection retries. If the LAN connection to a node fails GPFS will automatically try and reestablish the connection before making the node unavailable. This provides for better uptime in environments experiencing network issues. Using these features along with a high availability infrastructure ensures a reliable enterprise storage solution.

GPFS interaction with AIX

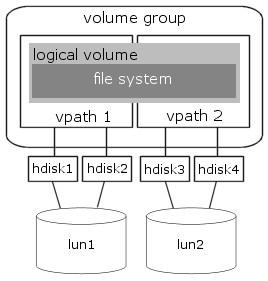
GPFS is a means to provide a journaled filesystem that can be mounted on multiple nodes simultaneously. GPFS stripes the data across all disks that belong to that file system. GPFS has a somewhat different approach of dealing with AIX volume groups and disks as we're used to; also mirroring is done in a different way.

A standard AIX setup has a device relationship that follows the following rules: A volumegroup is created that holds one or more physical disks. A disk contains one or more logical volumes, or a logical volume may span multiple disks. There is a one-to-one relation between a logical volume and the filesystem it contains. With LVM-mirroring each logical partition of a logical volume is placed on two separate disks. This typical setup is shown in the figure below:



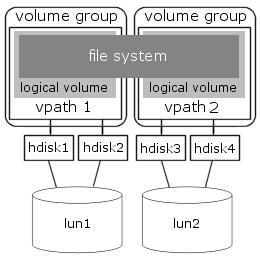
The original AIX filesystem structure.

In a SAN environment, this picture looks like this:



The AIX filesystem structure in a SAN environment.

Each volume group of GPFS contains only 1 (one) physical disk. Each disk containsonly 1 (one) logical volume. Each filesystem contains multiple logical volumes (one for each disk). LVM mirroring is not supported (there is only one disk in a volume group). This translates in the following picture:

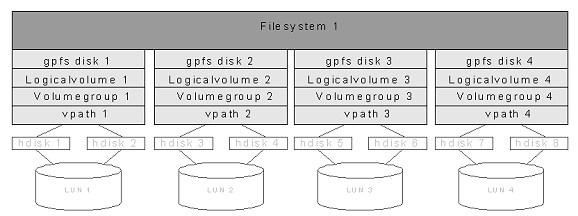


The GPFS filesystem structure in a SAN environment.

In GPFS 2.3 the GPFS volumes are called Network Storage Devices (NSD), that contain each only one physical disk. No volume groups and/or logical volumes are created in this GPFS version. In migrated clusters (from GPFS 2.2 to GPFS 2.3) you will still see volume groups and logical volumes, but only for the "old" disks. New disks and filesystems will be created without them.

We change the picture in a more "stack"-like representation. Here you see one GPFS filesystem that is made up out of four separate disks. AIX multipath-software has created the hdisk and vpath devices.

On the AIX level GPFS creates a separate volume group for each disk, so 4 volume groups in total. GPFS fills each disk with a logical volume, so 4 logical volumes in total. These logical volumes are represented as disk in the GPFS configuration. These GPFS-disks are used in the filesystem. A file stored in the filesystem is striped across the four disks (in 8kb blocks). The command used to create the GPFS disks is mmcrlv.



The stacked GPFS filesystem structure.

Usually, only small LUNs of only 17,5 GB are used instead of big luns (of 400 GB), because of performance.

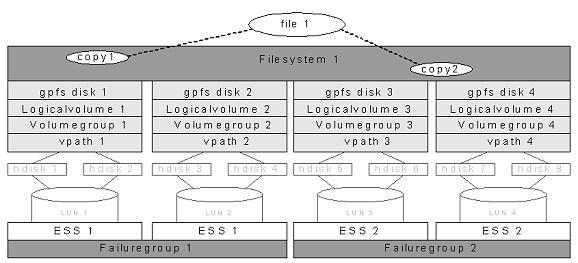
Mirroring versus replication

Traditional AIX mirrorring on the logical volume level can not be done in a typical GPFS device setup. The volume group holds only one disk that is completely filled with one logical volume, so there is no destination possible for the second copy of the lv's logical partitions. GPFS provides replication as the alternative.

GPFS provides a structure called replication that provides a means of surviving a diskfailure. On the file level you can specify how many copies of that file must be present in the filesystem (one or two). When you specify two copies, GPFS will duplicate the file across two "failuregroups". Setting replication on the file level is error-prone, this can easily be forgotten. It is also possible to specify this globally on the filesystem level. Set the "Default number of replicas = 2" and the "Maximum number of replicas = 2" on each GPFS filesystem, so that every file in all the GPFS filesystem are automatically replicated. Keep in mind that replication stores two file copies in the same filesystem. Each file will use twice the amount of space, so the filesystem free space will drop in size twice as fast. An example: The free space in the filesystem is 15 MB. You want to save a file of 10 MB, the result is a FILE SYSTEM FULL ERROR. The reason is that you need at least 20 MB free space to hold both copies of the file!

Failure groups

GPFS groups disks into "failuregroups". A failuregroup is a collection of disks that share a single point of failure (SPOF). In a SAN setup there is usually only one SPOF for the disk: All disks are usually multipath, so a single Host Bay Adapter (HBA) failure is no problem. All systems can be connected to two separate SAN fabrics, so a fabric failure is also no problem. Each disk is hosted by one ESS. When the ESS fails, all disks in that ESS will fail. Unless you have a second ESS, you can prevent this failure by using failuregroups. GPFS uses these failure groups to prevent that both replication copies of a file will fail at the same time. It does this by writing the two copies of a file to disks in separate failuregroups.



Each file copy in a separate failuregroup.

In the example above you see that the file is written twice in the filesystem. One copy is striped over lun 1 + 2 and the other copy is striped across lun 3 + 4. When ESS1 fails the second copy of the file is still completely usable on ESS2.

Striping

Large files in GPFS are divided into equal sized blocks, and consecutive blocks are placed on different disks in a round-robin fashion. To minimize seek overhead, the block size is large (typically 256K). Large blocks have the advantage that they allow a large amount of data to be retrieved in a single I/O from each disk. GPFS stores small files (at the end of large files) in smaller units called sub-blocks, which are as small as 1/32 of the size of a full block. Striping works best when disks have equal size and performance. This is why you should use one disksize for data storage in a filesystem; do not mix and match large and small luns.

GPFS transaction log

Just like JFS, GPFS is a journaled filesystem. GPFS records all metadata updates that affect file system consistency in a journal log. Each node has a separate log for each file system it mounts, stored in that file system. Because this log can be read by all other nodes, any node can perform recovery on behalf of a failed node. It is not necessary to wait for the failed node to come back to life. After a failure, file system consistency is restored quickly by simply re-applying all updates recorded in the failed node's log. Once the updates described by a log record have been written back to disk, the log record is no longer needed and can be discarded. Thus, logs can be fixed size, because space in the log can be freed up at any time by flushing "dirty" metadata back to disk in the background.

GPFS data and metadata

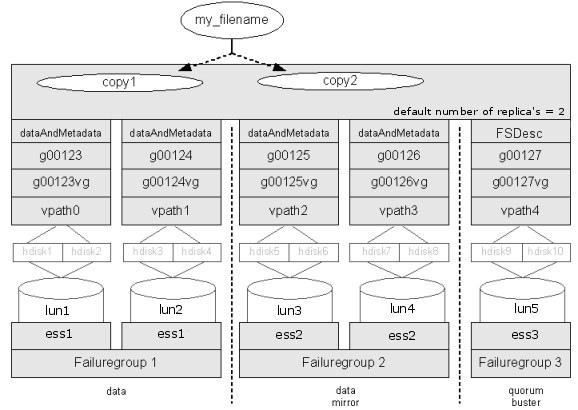
The GPFS filesystem contains two types of data: Data and Metadata. "Data" means the actual files you want to store in the filesystem. This is the usable storage space. "Metadata" refers to all sorts of information used by GPFS internally. For each GPFS disk you can specify what it will contain: DataAndMetadata, MetadataOnly, Dataonly or DescOnly. "DataAndMetadata" is used for normal disks, so most disks in the system will have this designation. "DescOnly" is used for "quorum busters".

GPFS filesystem descriptor quorum

There is a structure in GPFS called the Filesystem Descriptor (FSDesc) that is written originally to every disk in the filesystem, but is updated only on a subset of the disks as changes to the filesystem occur, such as adding or deleting disks. The subset of disks is usually a set of three or five disks, depending on how many disks and failuregroups are in the filesystem. The disks that constitute this subset of disks can be found by reading any one of the FSDesc copies on any disk. The FSDesc may point to other disks where more up-to-date copies of the FSDesc are located.

To determine the correct filesystem configuration, a quorum of the subset of disks must be online so that the most up-to-date FSDesc can be found. If there are three special disks, then two of the three must be available. GPFS distributes the copies of FSDesc across the failure groups. If there are only two failuregroups, one failure group has two copies and the other failure group has one copy. In a scenario that causes one entire failure group to disappear all at once, if half of the disks that are unavailable contain the single FSDesc that is part of the quorum, everything stays up. On the other hand, if the downed failure group contains the majority of the quorum, the FSDesc cannot be updated and the filesystem must be force unmounted. If the disks fail one at a time, the FSDesc is moved to a new subset of disks by updating the other two copies and a new disk copy. However, if two of the three disks fail simultaneously, the FSDesc copies cannot be updated to show the new quorum configuration. In this case, the filesystem must be unmounted to preserve existing data integrity. To survive a single ESS failure in a dual ESS configuration, there must be a third failure group on an independent disk outside both ESSs (the so-called TieBreaker node, which contains one disk per filesystem which contains the third FSDesc).

The final picture will be:



Final GPFS on SAN picture.

Taking all things mentioned above in account, the final solution for a GPFS filesystem is:

All files in the filesystem are replicated across two failuregroups on two nodes (preferably in two sites). This is controlled by the filesystem setting "default number of replicas = 2". The number of disk that hold data is the same at each of the two sites. The number of disk used for data has no practical limit. You will probably create multiple filesystems for other reasons than the disk limit. These disks also hold a copy of the metadata.

There is a third site with one disk used as quorum buster on the TieBreaker node. These disks hold no data or metadata, only a single filesystem descriptor (FSDesc).

GPFS software

For GPFS 2.2 the following filesets are installed on each node of the GPFS cluster:

* mmfs.base.cmds
* mmfs.base.rte
* mmfs.gpfs.rte
* mmfs.gpfsdocs.data
* mmfs.msg.en\_US

For GPFS 2.3 the following filesets are installed on each node of the GPFS cluster:

* gpfs.base
* gpfs.msg.en\_US
* gpfs.docs.data

For Oracle RAC using GPFS 2.3, installation of HACMP 5.2 (and RSCT) is required. This is specifically necessary for Oracle RAC and not for GPFS.

TOPICS: [GPFS](http://www.aixhealthcheck.com/blog?topic=25)[↑](http://www.aixhealthcheck.com/blog?topic=25#container)

## [GPFS links](http://www.aixhealthcheck.com/blog?id=50)

* [GPFS on IBM.com](http://www-1.ibm.com/servers/eserver/clusters/software/gpfs.html)
* [Search www.redbooks.ibm.com on GPFS](http://www.redbooks.ibm.com/cgi-bin/searchsite.cgi?query=gpfs) (Especially for the GPFS design and principles redbook)
* [GPFS FAQ](http://publib.boulder.ibm.com/infocenter/clresctr/index.jsp?topic=/com.ibm.cluster.gpfs.doc/gpfs_faqs/gpfs_faqs.html)

TOPICS: [GPFS](http://www.aixhealthcheck.com/blog?topic=25), [ORACLE](http://www.aixhealthcheck.com/blog?topic=24), [POWERHA / HACMP](http://www.aixhealthcheck.com/blog?topic=17)[↑](http://www.aixhealthcheck.com/blog?topic=25#container)

## [Oracle RAC introduction](http://www.aixhealthcheck.com/blog?id=49)

The traditional method for making an Oracle database capable of 7\*24 operation is by means of creating an HACMP cluster in an Active-Standby configuration. In case of a failure of the Active system, HACMP lets the standby system take over the resources, start Oracle and thus resumes operation. This takeover is done with a downtime period of aprox. 5 to 15 minutes, however the impact on the business applications is more severe. It can lead to interruptions up to one hour in duration.

Another way to achieve high availability of databases, is to use a special version of the Oracle database software called Real Application Cluster, also called RAC. In a RAC cluster multiple systems (instances) are active (sharing the workload) and provide a near always-on database operation. The Oracle RAC software relies on IBM's HACMP software to achieve high availability for hardware and the operating system platform AIX. For storage it utilizes a concurrent filesystem called GPFS (General Parallel File System), a product of IBM. Oracle RAC 9 uses GPFS and HACMP. With RAC 10 you no longer need HACMP and GPFS.

HACMP is used for network down notifications. Put all network adapters of 1 node on a single switch and put every node on a different switch. HACMP only manages the public and private network service adapters. There are no standby, boot or management adapters in a RAC HACMP cluster. It just uses a single hostname; Oracle RAC and GPFS do not support hostname take-over or IPAT (IP Address take-over). There are no disks, volume groups or resource groups defined in an HACMP RAC cluster. In fact, HACMP is only necessary for event handling for Oracle RAC.

Name your HACMP RAC clusters in such away, that you can easily recognize the cluster as a RAC cluster, by using a naming convention that starts with RAC\_.

On every GPFS node of an Oracle RAC cluster a GPFS daemon (mmfs) is active. These daemons need to communicate with each other. This is done via the public network, not via the private network.

Cache Fusion

Via SQL\*Net an Oracle block is read in memory. If a second node in an HACMP RAC cluster requests the same block, it will first check if it already has it stored locally in its own cache. If not, it will use a private dedicated network to ask if another node has the block in cache. If not, the block will be read from disk. This is called Cache Fusion or Oracle RAC interconnect.

This is why on RAC HACMP clusters, each node uses an extra private network adapter to communicate with the other nodes, for Cache Fusion purposes only. All other communication, including the communication between the GPFS daemons on every node and the communication from Oracle clients, is done via the public network adapter. The throughput on the private network adapter can be twice as high as on the public network adapter.

Oracle RAC will use its own private network for Cache Fusion. If this network is not available, or if one node is unable to access the private network, then the private network is no longer used, but the public network will be used instead. If the private network returns to normal operation, then a fallback to the private network will occur. Oracle RAC uses cllsif of HACMP for this purpose.