Issue #6 April 2005

RED HAT MAGAZINE

DATA SHARING WITH A GFS STORAGE CLUSTER

by Mark Hlawatschek and Marc Grimme
ATIX, Munich, Germany

Introduction

Linux server clustering has become an important technique to provide scalable performance and high availability for IT services. These services quite often require that data be shared between servers. In addition, even small companies often have many computers, including desktops and servers, that must share data. Hence, data sharing is a requirement for both small and large companies.

Some services have static data that can easily be split between servers. Using duplication, each server in the cluster hosts a copy of all the data. However, other services utilize dynamic data that is rapidly changing, and it is much more difficult to duplicate data. For example, databases and file services (based on protocols like SQL, NFS or CIFS) would have to distribute the new information synchronously to all other servers after each write. This would lead to very long response times and an extremely high network load. Another disadvantage is the higher costs for maintaining duplicate copies and for the associated increase in system management complexity. What these applications really need is access to a single data store that can be read from and written to by all servers simultaneously. The use of a file server (network attached storage server) supporting the NFS and CIFS protocols is the traditional approach for this kind of shared data. Linux, of course, offers these popular data sharing protocols, and this solution is suitable for some applications, but a single file server often becomes a performance bottleneck and single point of failure (SPoF) in the complete system.

To solve the difficulty of using traditional approaches for scalable and simplified data sharing, every server in the cluster should have direct access to the storage device, and each server should be able to read and write to the data store simultaneously. The Red Hat Global File System (GFS) represents the heart of such a solution. It combines Linux servers and a storage network (SAN) to create a data sharing cluster through a single shared file system base.

GFS internals

The Global File System was created as a 64-bit cluster file system. It enables several servers connected to a storage area network (SAN) to access a common, shared file store at the same time with standard UNIX/POSIX file system semantics.

The development of GFS began in 1995 at the University of Minnesota. At that time, large-scale computing clusters in use at the University were generating huge data sets, which had to be written effectively to a central storage pool. To solve this problem, Matthew O'Keefe, a professor at the University of Minnesota at that time, started to develop GFS with a group of students. Since then, Matthew has become Red Hat's Director of Storage Strategy. These efforts over time resulted in the Red Hat GFS cluster file system, which is released under the GPL. At the moment GFS is only available for Linux.

GFS is a journaling file system. Each cluster node is allocated its own journal. Changes to the file system metadata are written in a journal and then on the file system like other journaling file systems. In case of a node failure, file system consistency can be recovered by replaying the metadata operations. Optionally, both data and metadata can be journaled.

GFS saves its file system descriptors in inodes that are allocated dynamically (referred to as *dynamic nodes* or dinodes). They are placed in a whole file system block (4096 bytes is the standard file system block size in Linux kernels). In a cluster file system, multiple servers access the file system at the same time; hence, the pooling of multiple dinodes in one block would lead to more competitive block accesses and false contention. For space efficiency and reduced disk accesses, file data is saved (*stuffed*) the dinode itself if the file is small enough to fit completely inside the dinode. In this case, only one block access is necessary to access smaller files. If the files are bigger, GFS uses a "flat file" structure. All pointers in a dinode have the same depth. There are only direct, indirect, or double indirect pointers. The tree height grows as much as necessary to store the file data as shown in Figure 1.

"Extendible hashing" (ExHash) is used to save the index structure for directories. For every filename a multi-bit hash is saved as an index in the hash table. The corresponding pointer in the table points at a "leaf node." Every leaf node can be referenced by multiple pointers. If a hash table leaf node becomes too small to save the directory entries, the size of the whole hash table is doubled. If one leaf node is too small, it splits up into two leaf nodes of the same size. If there are only a few directory entries, the directory information is saved within the dinode block, just like file data. This data structure lets each directory search be performed in a number of disk accesses proportional to the depth of the extendible hashing tree structure, which is very flat. For very large directories with thousands or millions of files, only a small number of disk accesses are required to find the directory entry.

The latest version GFS 6.0 offers new features including file access control lists (ACLs), quota support, direct I/O (to accelerate database performance), and dynamic on-line enlargement of the file system.

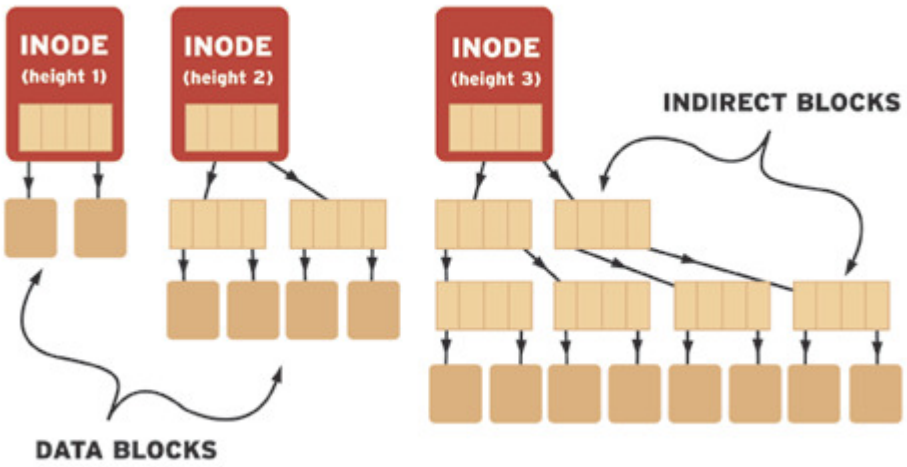

Figure 1: GFS metadata structure

Structure

Figure 2 shows the structure of a typical GFS storage cluster. The GFS file system is mapped onto a pool volume, which is constructed from one or more independent storage units. The servers are connected with a storage network (SAN) over one or more data paths to the pool volume. The individual cluster servers are also connected via one or more data paths to the network. Thus every server can directly access the storage arrays onto which the pool volume is mapped, greatly increasing I/O system performance and providing scalability far beyond what can be achieved with a single NAS server.

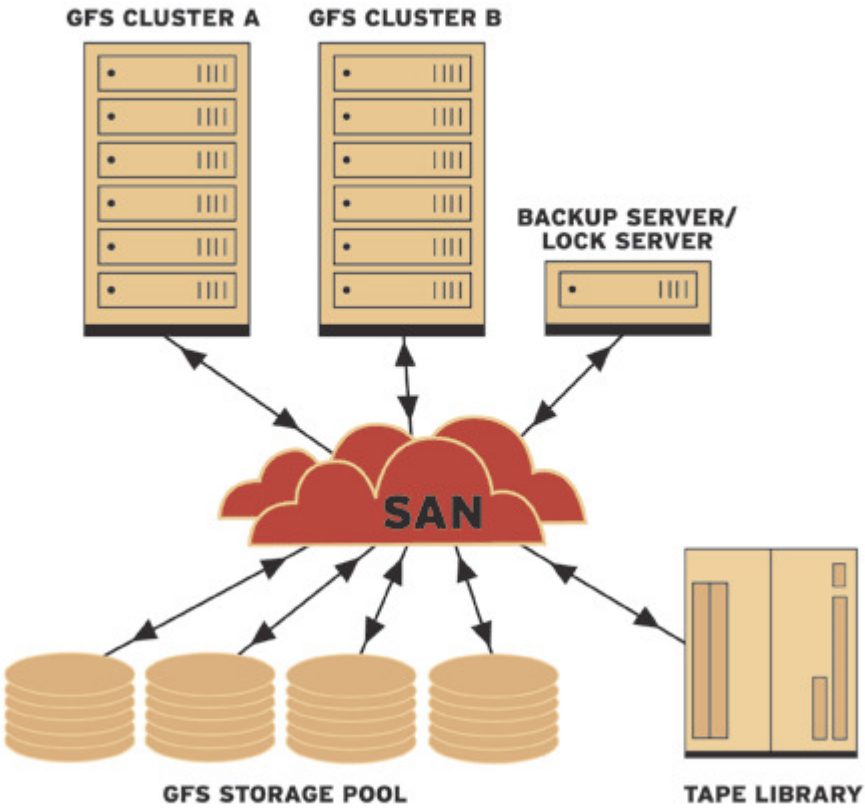

Figure 2: GFS storage cluster

The servers in the GFS storage cluster use Linux as the operating system. A simple cluster volume manager, the GFS pool layer virtualizes the storage units (/dev/sda) and aggregates them into a single logical pool volume (dev/pool/foo). Multiple devices can be combined by striping or by concatenation. Changes in the pool configuration are visible in all cluster servers. The pool volume manager allows pool volumes to be resized online and provides I/O multi-pathing, so that single failures in the SAN path can be tolerated. However, the pool volume manager does not provide volume mirroring or snapshots. These capabilities will be provided in the future via CLVM, the Cluster Logical Volume Manager, a LVM2-based cluster volume manager which allows multiple servers to share access to a storage volume on a SAN.

The lock server coordinates the multiple servers which access the same physical file system blocks in a GFS storage cluster. It assures the file system's data consistency. From the beginning, GFS has been provided with a modular locking layer. In early GFS versions lock information was exchanged over the SCSI protocol (DLOCK, DMEP). Since GFS version 5, a redundant, IP-based user space locking service (RLM), which runs on all nodes ,has been used. Red Hat is working to integrate its distributed lock manager (DLM) into GFS 6.1, which will be released in the summer of 2005 Each server in the GFS cluster must heartbeat the lock server on a regular basis. If the server is unable to heartbeat, it will be selected by the lock manager to be removed from the cluster, an operation called fencing. GFS supports several fencing mechanisms including different network power switches and HP's ILO interface.

Scalability

A classic IT system consists of services and applications that run on individual servers and that are generally limited to running on a particular server. If the hardware to which a particular application is limited is no longer sufficient, the application generally cannot exploit the additional memory, processing power, or storage capacity contained in the rest of the cluster. In contrast, applications that can run in parallel on a storage cluster are much easier to scale. In case of a capability shortage, new components (server, storage) can be easily integrated in the system until the required capacity can be achieved. The common use of a storage pool not only removes the need for the laborious duplication of data to multiple servers but also offers elegant scaling possibilities. With growing storage requirements, the common storage pool can be expanded and will be immediately available for all servers.

Availability

The availability of the complete system is an important aspect for providing IT services. To achieve Class 3 availability (99% to 99.9%), it is necessary to eliminate every single point of failure (SPOF). For Class 4 availability (99.9% to 99.99% uptime), it is necessary to have a high availability cluster, mirrored data, and a second data center for disaster recovery. The services must have the possibility to run on multiple servers at different locations. The breakdown of one server or the whole data center must not avert the accessibility of the services for more than a short time. A GFS cluster can be connected to the central storage system via the SAN through redundant I/O paths to overcome the breakdown of individual infrastructure components like switches, host bus adapters, and cables. I/O multi-pathing can be implemented either by the fibre channel driver for the host bus adapter or by the GFS pool. Unfortunately, the GFS storage cluster is not yet able to mirror file blocks redundantly to multiple storage devices from the host servers but can of course take advantage of the hardware redundancy available on good RAID storage arrays. Host-based mirroring in a GFS cluster arrives later in 2005 with the Cluster Logical Volume Manager (CLVM).

The lock server, which is essential for GFS, is available in two versions. There is a simple version (Single Lock Manager, or SLM), which is a SPOF for the complete system, and the redundant version (Redundant Lock Manager, or RLM). It is possible to define multiple lock servers with the RLM, which can transparently overtake the role of an active lock server in case of a failure. In addition, Red Hat Cluster Suite can be used to provide application fail-over in GFS clusters.

LAN-free backup

A data backup is normally done from backup client machines (which are usually production application servers) either over the local area network (LAN) to a dedicated backup server (via products like Legato Networker or Veritas Netbackup), or LAN-free from the application server directly to the backup device. Because every connected server using a cluster file system has access to all data and file systems, it is possible to convert a server to a backup server. The backup server is able to accomplish a backup during ongoing operations without affecting the application server. It is also very useful to generate snapshots or clones of GFS volumes using the hardware snapshot capabilities of many storage products. These snapshot volumes can be mounted and backed up by a GFS backup server. To enable this capability, GFS includes a file system quiesce capability to ensure a consistent data state. To quiesce means that all accesses to the file system are halted after a file system sync operation which insures that all metadata and data is written to the storage unit in a consistent state before the snapshot is taken.

Diskless shared root clustering

As all servers in a GFS storage cluster access their data through a shared storage area network, additional servers can be added to easily scale the server capacity. Hence, each server can be viewed as just another resource in the pool of available servers. The system data and operating system images are on the shared storage, and therefore server and storage can be seen as effectively independent of each other. The result is a diskless shared root cluster where no server needs a local hard disk; each server can instead boot directly from the SAN. Both application data and the operating system images are shared, which means the root (/) partition for all cluster nodes is the same. As a consequence the management is simplified. Changes have to be done only one time and they are immediately valid for all servers. Constructing shared root disk clusters with GFS is quite hardware and kernel-version-specific, and this feature should only be deployed with the help of Red Hat professional services or Red Hat partners like ATIX GmbH.

Case study: IP Tech AG

The deployment of GFS at IP Tech, one of the biggest Internet hosting and service providers in Switzerland, demonstrates how effectively Red Hat cluster technologies are already used in enterprises. Since the beginning of 2002, a Red Hat GFS cluster with 14 nodes has been in production at IP Tech. This cluster supports database (MySQL), email (Qmail) and web serving (Apache) applications in special configurations. Over 1500 MySQL databases, 10,000 mail domains, and 28,000 web domains are hosted for mostly Swiss companies at IP Tech. Current daily operations at IP Tech support about 5-7 million web accesses, 3.0-3.5 million pop3 (email server) connections, 1.0-1.5 million SMTP connections (email relay), and 3.5-4.0 million MySQL connections every day over this GFS-based infrastructure. Over 100,000 individual email users are supported.

In addition to the classic web, database, and email services, IP Tech recently introduced the hosting of virtual machines and the dedicated allocation of servers in a GFS storage cluster. Customers can now dynamically allocate GFS servers on-the-fly and run multiple virtual servers on a single GFS node. This is an excellent way to improve system utilization and squeeze the most out of the GFS data sharing cluster infrastructure.

Recently, IP Tech migrated its services to a centralized blade-based infrastructure with two terabytes of redundant shared root storage and about 22 diskless blade servers. All applications except for the virtual machines run on GFS. This configuration minimizes hardware repair times by simply replacing blades in case of failure and pointing them to boot off the shared root boot image. Server and storage scalability can be achieved during ongoing operation. Additionally each night file system data is replicated on a second storage system via a LAN-free backup using GFS and the SAN. Figure 3 illustrates the IP Tech infrastructure.

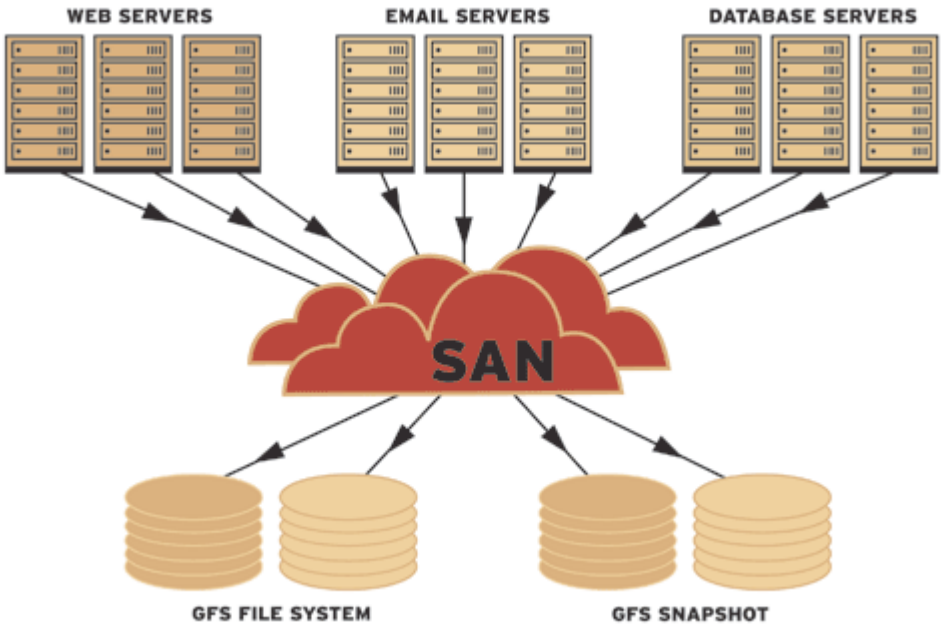

Figure 3: IP Tech infrastructure

IP Tech initially used NFS for the data sharing requirements of their IP hosting environment, but they had significant problems maintaining it because it was unstable under heavy load. File services and mounts would come and go without warning, stopping operations at critical times, and quite often during high load periods where down time would have the maximum negative impact. Two years ago, IP Tech migrated to Red Hat GFS and, in contrast to their NFS-based storage infrastructure, the cluster running Red Hat GFS "ran by itself."

By using a Red Hat Enterprise Linux cluster with Red Hat GFS, IP Tech could achieve both high availability and performance. If any of the servers crashed or if an application (e.g., http or qmail) hung, this server could be re-booted quickly and then brough back into the cluster infrastructure without disrupting the services being provided by the other servers. In addition, IP Tech also uses the shared GFS root disk feature which simplifies the process of updating software and performing static application service load-balancing in the cluster. For example, when IP Tech has a spam attack they can quickly change some webservers to mailservers and keep up the mail service without problems and detect and counter the spam attack, while all cluster services continue to run. They can also scale the infrastructure within minutes using new servers and storage arrays. Finally, IP Tech performs regular backups on a point-in-time snapshot of the GFS volume using a separate server in the cluster. This approach allows a GFS file system to be backed up with almost no impact to other system operations.

In summary, the key benefits IP Tech found in using Red Hat GFS were:

1. Very high performance and scalability that was unattainable with NFS.
2. Reduced complexity via data sharing and shared root disk images.
3. On-the-fly load-balancing of services across the cluster to provide on-demand computing and focused performance for applications that must have it.
4. IP Tech exploits the ability of Red Hat GFS to allow snapshots of existing file system and database volumes on storage hardware. These volumes are then mounted read-only and backed-up in parallel with regular file system operation such that the latter is minimally impacted.

Summary

By using Red Hat Enterprise Linux and GFS, IP Tech has achieved excellent performance and

scalability, reduced complexity, scalability, and availability that was unattainable with NFS. Data sharing via GFS has allowed IP Tech to reduce management complexity and scale performance to meet customer demand while using a low-cost, Linux-based hardware and software infrastructure. Red Hat GFS can also accelerate other storage-intensive applications including Oracle database clusters, software development (builds and compiles), and high performance computing. Learn more about Red Hat GFS at redhat.com.

About the author

Marc Grimme, academically qualified computer scientist, and Mark Hlawatschek, academically qualified engineer, are two of the three founders of the ATIX GmbH. Their main focus lies within the development and the implementation of tailor-made enterprise storage solutions based on SAN/NAS. Additionally, they are responsible for the realisation of highly scalable IT infrastructures for enterprise applications on Linux base.