Paper 1559-2014

SAS® Grid Manager I/O: Optimizing SAS® Application Data Availability for the Grid

Gregg Rohaly, IBM Corporation – STG Platform Computing; Harry Seifert, IBM Corporation – STG Senior Certified IT Specialist

ABSTRACT

As organizations deploy SAS® applications to produce the analytical results that are critical for solid decision making, they are turning to distributed grid computing operated by SAS® Grid Manager. SAS Grid Manager provides a flexible, centrally managed computing environment for processing large volumes of data for analytical applications. Exceptional storage performance is one of the most critical components of implementing SAS in a distributed grid environment. When the storage subsystem is not designed properly or implemented correctly, SAS applications do not perform well, thereby reducing a key advantage of moving to grid computing. Therefore, a well-architected SAS environment with a high-performance storage environment is integral to clients getting the most out of their investment. This paper introduces concepts from software storage virtualization in the cloud for the generalized SAS Grid Manager architecture, highlights platform and enterprise architecture considerations, and uses the most popularly selected distributed file system, IBM GPFS, as an example. File system scalability considerations, configuration details, and tuning suggestions are provided in a manner that can be applied to a client's own environment. A summary checklist of important factors to consider when architecting and deploying a shared, distributed file system is provided.

INTRODUCTION

SAS® Grid Manager enables solutions to use a centrally managed grid infrastructure to provide workload balancing, high availability and parallel processing for business analytics jobs. For example,

- SAS® Data Integration Studio and SAS® Enterprise Miner™ are automatically customized for parallel processing in a grid computing environment. These programs detect the SAS Grid Manager environment at the time of execution.
- SAS® Enterprise Guide® programs can be sent over a pool of server resources with a set of pre-defined SAS Grid-enabled statements for load balancing.
- SAS® Web Report Studio, SAS® Marketing Automation jobs can be scheduled to run or directed to the most appropriate resource in the grid.
- SAS® Risk Dimensions® has an iterative workflow of executing the same analysis against different subsets of data whereby each iteration of the analysis can be submitted to the grid using SAS Grid Manager to provide load balancing and efficient resource allocation.

The objective of this paper is to describe the storage infrastructure and file system capability necessary for optimal SAS Grid Manager performance. A shared file system is a required and integral component of all SAS Grid Manager deployments, Enterprise Business Intelligence deployments with load balanced servers on multiple systems, and certain other types of distributed SAS applications. The important characteristics of shared file systems with respect to SAS performance are:

- The ability of file system data to be retained in a well-managed local file cache (in memory),
- high performance handling of file system metadata, and
- matching data throughput to the underlying physical resources.

Software-defined storage (SDS) is an approach to data storage in which the programming that controls storage-related tasks, such as Service Level Agreement (SLA) tiering, are decoupled from the physical storage hardware. This storage-based incorporation of networked disks, disk arrays, flash memory arrays, etc. toward meeting SLAs and deriving the best use of pooled physical assets is often referred to as storage virtualization and is now considered a large component of an overall enterprise sized storage strategy.

Software-defined storage puts the emphasis on storage services such as deduplication or replication, instead of storage hardware. Without the constraints of a physical system, a storage resource can be used more efficiently and its administration can be simplified through automated policy-based management. For example, a storage administrator can use service levels when deciding how to provision storage. Storage can, in effect, become a shared pool that runs on commodity hardware.

Software-defined storage is part of a larger industry trend that includes software-defined networking (SDN) and software-defined data centers (SDDC). As is the case with SDN, software-defined storage enables flexible management at a much more granular level through programming capability.

THE SAS® GRID MANAGER REQUIREMENT FOR A SHARED FILE SYSTEM

From the Server-side with respect to a SAS Grid Manager environment, there are certain services that need to always be available and accessible. These services are vital to the running applications and their ability to process SAS jobs. Example SAS Grid Manager services include:

- SAS® Metadata Server
- SAS® Object Spawner
- IBM Platform Computing Process Manager
- IBM Platform Computing Grid Management Service
- Web Application Tier Components

High Availability for grid services can be achieved using the IBM Platform Computing Enterprise Grid Orchestration (EGO) capabilities provided by the Platform Suite for SAS® which is a third party component included with the SAS Grid Manager product.

In order for SAS Grid Manager to provide high availability for SAS services, the following aspects must be addressed:

- Service Resource Redundancy – Provide alternate resources for execution of essential services. Allowing critical functions to execute on multiple physical or virtual nodes eliminates a single point of failure for the grid.
- Service Resiliency – Provide a mechanism to monitor the availability of services on the grid and to automatically restart a failed service on the same resource or on an alternate when necessary.
- Service Relocation – Provide a method for allowing access to services running on the grid without requiring clients to have any knowledge of the physical location of the service.

From the Storage-side with respect to a SAS Grid Manager environment, the location of services (or applications) binaries must be seen across the server resources. This, across the SAS Grid Manager view, is what permits the fail over of these services and the checkpoint – restart of SAS user jobs upon server failure. It is the use of a shared and/or distributed file system that adds both the capability and performance required for a job to restarts via its managed context that resides within the IBM Platform Computing Load Sharing Facility (LSF) component of SAS Grid Manager. The initial condition data for a SAS workload would still reside in the users workspace (often termed /sasdata) if the SAS jobs server resources became unavailable – due to equipment failure or resource reallocation to higher priority work. If a job can be check pointed, then upon similarly defined server resource considerations, the SAS job could be restarted at the point it last check pointed (often in /saswork), but on another available resource. The shared file system resource enables these capabilities.

Given the requirements listed for SAS Grid Manager file system resiliency and storage capacity for large SAS I/O data files, the performance capability of such a shared file system is of primary focus.

ENTERPRISE-LEVEL DISTRIBUTED/SHARED STORAGE OPTIONS

COMMON INTERNET FILE SYSTEM (CIFS)

Common Internet File System (CIFS) is a file-sharing protocol that provides an open and cross-platform mechanism for requesting network server files and services. CIFS is based on the enhanced version of Microsoft's Server Message Block (SMB) protocol for Internet and intranet file sharing. It uses TCP for data access and delivery. It is predominant for file shares in a Windows environment. This option is largely utilized for Departmental file sharing and print services.

NETWORK FILE SYSTEM (NFS)

A network file system (NFS) is a type of file system mechanism that enables the storage and retrieval of data from multiple disks and directories across a shared network. NFS is derived from the distributed file system mechanism. It is generally implemented in computing environments where the centralized management of data and resources is critical. Network file system works on all IP-based networks. It uses TCP and UDP for data access and delivery, depending on the version in use. It is predominant for file shares in a Unix/Linux environment. This option is largely utilized for Departmental or light Enterprise-level file sharing.

DISTRIBUTED FILE SYSTEMS (DFS)

A shared (distributed) file system uses a storage-area network (SAN) to provide direct disk access from multiple computers at the block level. Access control and translation from file-level operations that applications use to block-

level operations used by the SAN must take place on the client node. The most common type of clustered file system is the shared-disk file system, which—by adding mechanisms for concurrency control—provides a consistent and serialized view of the file system, avoiding corruption and unintended data loss even when multiple clients try to access the same files at the same time. It is a common practice for shared-disk file systems to employ some sort of a fencing mechanism which will prevent data corruption in the case of node failures. An unfenced device can cause data corruption if it loses communication with its sister nodes; and then tries to access the same information other nodes are accessing. The underlying storage area network may use any number of block-level protocols, including SCSI, iSCSI, HyperSCSI, ATA over Ethernet (AoE), Fibre Channel, network block device, and Infiniband.

Distributed file systems may aim for "transparency" in a number of aspects. That is, the aim is to be transparent to client programs, which view a single pool of storage appearing as a local file system. Behind the scenes, the chosen distributed/shared file system should handle locating files, transporting data, and potentially provide these other features listed:

- Access transparency is that clients are unaware that files are distributed and can access them in the same way as local files are accessed.
- Location transparency; a consistent name space exists encompassing local as well as remote files. The name of a file does not give its location.
- Concurrency transparency; all clients have the same view of the state of the file system. This means that if one process is modifying a file, any other processes on the same system or remote systems that are accessing the files will see the modifications in a coherent manner.
- Failure transparency; the client and client programs should operate correctly after a server failure.
- Heterogeneity; file service should be provided across different hardware and operating system platforms.
- Scalability; the file system should work well in small environments (1 machine, a dozen machines) and also scale gracefully to huge ones (hundreds through tens of thousands of systems).
- Replication transparency; to support scalability, we may wish to replicate files across multiple servers. Clients should be unaware of this.
- Migration transparency; files should be able to move around without the client's knowledge.

These software-based entities are usually purchased from a vendor and implementations can vary greatly depending upon the features required.

ENTERPRISE DATACENTER CONSIDERATIONS FOR STORAGE AND STORAGE VIRTUALIZATION

STORAGE CLOUD FOR EVERYONE - CONCEPTS

Free cloud storage is easier to come by these days – almost anyone can provide it and can give out a lot of it. However, the best cloud storage providers give you more than just storage. For example, they offer availability, multi-platform support, security, and application integration. Because of the popularity of backing up mobile device data – most people's first foray into the storage cloud and its concepts - you probably already know what Dropbox is. Support for virtually every operating system, both desktop and mobile, experimental and beta builds that add tons of useful features, and a vast third-party developer community which take advantage of its open APIs to build applications on top of it make Dropbox a powerful cloud storage service. Whether you use Dropbox for your files thereby syncing with the desktop clients, or you have another favorite app that uses Dropbox to keep your files synced across devices, the capability that Dropbox provides has been welcomed.

Another example is brought to you by the thoughtful people at Google. Google Drive has only been around under two years (at the time of this writing), but the fact that it combines the tools - formerly known as Google Docs (Docs, Spreadsheets, Presentations, and Drawing), all of your files created with those tools, and 15GB of storage for anything else you want is what makes it a popular Cloud Storage provider. After all, if you're using those tools in your day-to-day work, it makes sense to use Google Drive for file storage as well. Plus, the fact that it's available in OS X, Windows, iOS, and Android makes it useful at your desk or in today's mobile environments.

THE ENTERPRISE NEED FOR STORAGE VIRTUALIZATION

The average person using a mobile device or desktop is now becoming familiar with Cloud Storage concepts and using those services - even if just to store personal digital images or backing up a 'drive'. Most companies will not or cannot use a storage cloud service like Google Drive and Dropbox – free, cheap, or otherwise; however, the corporate need for pools of storage and easy use exists in the same generalized sense as for the mobile device user. In fact, SAS has many applications that generate plots, reports, etc. that get pushed to mobile devices and tablets as their use has become more pervasive. Yet, today's Enterprise requires pooled storage with a typical SLAs structure, security, compliance, governance, and the cost concerns that get reduced with removing storage silos and turning

storage into a shared pool. Even though SAS Grid Manager requires a shared file system, it is not alone in having Enterprise-level expectations with respect to Corporate IT demands on costs, functionality, and governance.

THE MODERN ENTERPRISE DEMANDS SNAPS, CLONES, MULT-TIER DATA MOVEMENT FOR MEETING CORPORATE WIDE SERVICE LEVEL AGREEMENTS

Storage virtualization software, which is required to provide a shared pool of storage beyond block arrays, should provide high availability tools to continuously monitor the health of the file system components. When failures are detected, appropriate recovery action must be taken automatically. Journaling and recovery capabilities should be provided which maintain metadata consistency when a node holding locks or performing administrative services fails. For an additional level of data availability and protection, synchronous data replication should be available for file system metadata and data. Storage software should provide a very flexible replication model that allows you to replicate a file, set of files, or an entire file system.

Synchronous replication should be location aware which allows you to optimize data access when the replicas are separated across a WAN. Ideally, the storage software has knowledge of which copy of the data is “local” so read-heavy applications can get local data read performance even when data is replicated over a WAN. Snapshots can be used to protect the file system's contents against a user error by preserving a point in time version of the file system or a sub-tree of a file system called a fileset. Storage virtualization software can help you to achieve data lifecycle management efficiencies through policy-driven automation and tiered storage management.

Storage pools are used to manage groups of disks within a file system. With the usage of storage pools you can create tiers of storage by grouping disks based on performance, locality or reliability characteristics. For example, one pool could contain high performance solid state disk (SSD) disks and another more economical 7,200 RPM disk storage. These types of storage pools are called internal storage pools. When data is placed in or moved between internal storage pools all of the data management is done by GPFS. In addition to internal storage pools GPFS supports external storage pools. Some vendors offer tight integration with backup and recovery. For example with IBM GPFS, external storage pools are used to interact with an external storage management application including IBM Tivoli Storage Manager (TSM) and High Performance Storage System (HPSS). When moving data to an external pool GPFS handles all of the metadata processing and then hands the data to the external application for storage on alternate media, like tape for example. When using TSM or HPSS, data can be retrieved from the external storage pool on demand as a result of an application opening a file or the data can be retrieved in a batch operation using a command or GPFS policy (Fadden, 2012).

A “fileset” is a sub-tree of the file system namespace and provides a way to partition the namespace into smaller, more manageable units. Filesets provide an administrative boundary that can be used to set quotas, take snapshots, define data tiering relationships, and be used in user defined policies to control initial data placement or data migration. Where the file data resides and how it is managed once it is created would be based on a set of rules in a user defined policy.

CIFS, NFS, AND DISTRIBUTED ARCHITECTURES FOR USE WITH SAS GRID MANAGER

COMMON INTERNET FILE SYSTEM (CIFS)

Common Internet File System (CIFS) is the native shared file system often used with the Windows operating systems. With improvements over time, CIFS can be used for workloads with moderate levels of concurrency and works best for SAS workloads that get limited benefit from the local file cache. The recommended configuration is to place /saswork on a non-CIFS file system and use CIFS to manage shared permanent files even including /sasdata. Running /saswork on a file system built off of drives internal to the Windows server would meet this need well. However, the /saswork is not shared and the SAS Grid Manager implementation would lack checkpoint restart ability. This also results in a more complex configuration and additional server costs due to increased internal storage.

SAS has compared CIFS to other shared file systems measured on the Windows platform. IBM GPFS, for example, showed much better throughput, local file cache retention and metadata management compared to CIFS. These performance-related attributes translated to better scalability whereby GPFS can serve a larger number of client systems compared to CIFS (Walters et al., 2013).

Windows Implementation

CIFS or SMB works through a client-server approach, where a client makes specific requests and the server responds accordingly. One section of the SMB protocol specifically deals with access to file systems, such that clients may make requests to a file server; but some other sections of the SMB protocol specialize in inter-process communication (IPC). The Inter-Process Communication (IPC) share, or ipc$, is a network share on computers running Microsoft Windows. This virtual share is used to facilitate communication between processes and computers over SMB, often to exchange data between computers that have been authenticated.

Developers have optimized the SMB protocol for local subnet usage, but users have also put SMB to work to access different subnets across the Internet—exploits involving file-sharing or print-sharing in MS Windows environments usually focus on such usage.

SMB servers make their file systems and other resources available to clients on the network. Client computers may want access to the shared file systems and printers on the server, and in this primary functionality SMB has become best-known and most heavily used. However, the SMB file-server aspect would count for little without the NT domains suite of protocols, which provide NT-style domain-based authentication at the very least. Almost all implementations of SMB servers use NT Domain authentication to validate user-access to resources.

Samba Implementation - Overview

Andrew Tridgell started the development of Samba, a free-software re-implementation of the CIFS/SMB networking protocol for Unix-like systems, in 1991. As of version 3 (2003), Samba provides file and print services for Microsoft Windows clients and can integrate with a Windows NT 4.0 server domain, either as a Primary Domain Controller (PDC) or as a domain member. Samba4 installations can act as an Active Directory domain controller or member server, at Windows 2008 domain and forest functional levels.

NETWORK FILE SYSTEM (NFS)

NFS client and server implementations show a wide variety of behavior that affect performance. For that reason, the specific client and server should be measured to ensure performance goals can be met. The NFS client maintains a cache of file and directory attributes. The default settings will not ensure that files created or modified on one system will be visible on another system within a minute of file creation or modification. The default settings may cause software to malfunction if multiple computer systems are accessing data that is created or modified on other computer systems.

In order to ensure file data consistency, when a NFS client detects a change in a file system attribute, any data in the local file cache is invalidated. The next time the file is accessed, its data will be retrieved from the NFS server. This means that retention in the file cache may have much different behavior with a NFS file system compared to other file systems. The file system storage devices and network must be provisioned to handle a larger demand compared to either a local file system or a shared file system that uses a different strategy for cache coherency (Walters et al, 2013).

NFS Implementation

The high-level steps mentioned assume a Unix-style scenario in which one machine (the client) requires access to data stored on another machine (the NFS server).

1. The server implements NFS daemon processes (running by default as nfsd) in order to make its data generically available to clients.
2. The server administrator determines what to make available, exporting the names and parameters of directories (typically using the /etc/exports configuration file and the exportfs command).
3. The server security-administration ensures that it can recognize and approve validated clients.
4. The server network configuration ensures that appropriate clients can negotiate with it through any firewall system.
5. The client machine requests access to exported data, typically by issuing a mount command. (The client asks the server (rpcbind) which port the NFS server is using, the client connects to the NFS server (nfsd), nfsd passes the request to mountd)
6. Users on the client machine can then view and interact with mounted file systems on the server within the parameters permitted.

Note: That automation of the NFS mounting process may take place — perhaps using /etc/fstab and/or automounting facilities.

DISTRIBUTED FILE SYSTEMS (DFS)

Current enterprise data center SLA requirements generally dictate fault tolerance and configuration for continued access to data even if individual nodes or storage systems fail. Distributed/shared file systems can accomplish these goals through robust clustering features and support for synchronous and asynchronous data replication. This option is largely utilized for Enterprise-level data sharing.

Typically, this level of functionality and support requires a vendor product to be purchased. Storage clustering software should include the infrastructure to handle data consistency and availability. This implies that a storage clustering product should not rely on external applications for cluster operations like node failover. The clustering support should go beyond who owns the data or who has access to the disks. In an Enterprise-level cluster, all nodes see all of the data and all cluster operations can be done by any node in the cluster with proper licensing. All server-related nodes would be capable of performing all cluster-related tasks. For example, as a part of the built-in

availability tools, the IBM GPFS clustering continuously monitors the health of the file system components. When failures are detected appropriate recovery action is taken automatically. Extensive journaling and recovery capabilities must be provided which maintain metadata consistency when a node holding locks or performing administrative services fails. Metadata consistency is of utmost importance for local and remote operations given the level of caching and locks involved for this degree of data availability.

For example, IBM GPFS has been tested at SAS in multiple use cases. The IBM General Parallel File System (GPFS) performed well on both Red Hat Enterprise Linux (RHEL) and Microsoft Windows operating systems. “Both /sasdata and /saswork files were managed by GPFS with excellent throughput and low overhead for file system metadata management. GPFS requires dedicated system memory and the amount of memory is defined by the file system option PAGEPOOL=. The client systems used in the benchmarks have 96 GB of memory and 32 GB of it was dedicated to the page pool. It is likely that client computer systems that utilize both a non-shared file system and GPFS will require additional memory to allow both file systems to perform adequately.”

“For both the SAN and NAS configurations, GPFS was able to transmit data at a rate limited only by the throughput rate of the physical device. This rate was sustained during the large sort workload and achieved periodically for the calibration workload. The calibration workload ran 100 to 250 concurrent SAS processes spread over six 12-core systems. Maximum throughput was achieved and sustained and file system metadata overhead stayed at a manageable and responsive level.” More details are available in the cited publication from SAS (Walters et al., 2013).

As one would expect, the heavy I/O associated with most SAS workloads requires an enterprise-level file system that has been architected to perform and scale based upon its technical computing roots. SAS has tested other similar vendor file system products as well.

PERFORMANCE, SCALABILITY, AND CONFIGURATION CONSIDERATIONS

PERFORMANCE

Similar to other software vendors, SAS has specific system requirements that are tied to hardware configuration and capability. For SAS Grid Manager clients, SAS has tested and recommends that 100 MBps/core is a solid expectation to keep the SAS Grid environment from being I/O starved. Because each core is essentially a job slot in the grid, one needs to have the I/O bandwidth to keep that degree of computational workload progressing and not on I/O waits. It is true that every client has the potential for a unique workload profile; however, SAS has amassed a great deal of experience which has resolved to this throughput number. The generalized configuration tweaks that provide the best performance will not be repeated in this paper. However, SAS and their technology partners have done many studies to find best practices for various vendor hardware, OSes, SAS software, etc. which are available on the SAS support website (http://support.sas.com/). Typically, expected tunables regarding I/O firmware, I/O queue depth, file system caching, storage layout to name a few are great areas to use the recommended settings for your infrastructure and see performance gains.

SCALABILITY

The scalability for enterprise storage largely consists of adding disks to a storage frame or adding frames thus increasing capacity and resulting in the same performance characteristics. Adding storage capacity is sometimes challenging given the responsibility is often owned by a Client’s core IT department. Therefore, it is recommended to bring the IT Department into the discussion early when selecting a suitable architecture for SAS Grid Manager deployment. There are many benefits to having a single shared copy of all SAS binaries, one /saswork and one /sasdata all available from a single file system namespace. Some Clients face multiple data center locations for data and compute; therefore, being able to use storage virtually in the software defined storage framework aids in administration by seamlessly adding storage to the existing pool of storage. Often technology refresh adds complexity with respect to storage because of data migration aspects. Modern storage software capability enables full uptime and transparent data movement so that the underlying physical storage devices can be swapped out. This can be done with storage appliances that virtualize the arrays using a single array (or more) as the front end that manages and provides a single view of the storage capacity ‘hanging off’ of the primary array. However, this can cause port limitation choke points when dealing with medium to large data center environments. Hence, array-based virtualization approach could have a scalability issue down the road.

Being able to transparently insert extra storage capacity into the existing file system framework is an important capability. For example, when creating an IBM GPFS file system one provides a list of raw devices or LUNS and they are assigned to GPFS as Network Shared Disks (NSD). Once a NSD is defined, all of the nodes in the GPFS cluster can access the disk, using local disk connection, or by using the GPFS NSD network protocol for shipping data over a TCP/IP or Infiniband connection. If you buy another array for added capacity, then you can make more NSDs and add them into the existing file systems which are built by NSDs. The file system can then be rebalanced in the background so that the stripping of data is properly laid out for peak performance. Care, of course, needs to be given when adding storage to match like devices so that performance effects for different speed drives, for example, does

not impact base performance. Additionally, many current enterprise arrays need their own rebalancing to be done when adding disks to the existing pool of physical disks.

CONFIGURATION

The complexity of solution configuration has been steadily growing as we move from hardware only to having multiple layers of hardware and software comprising solutions in the datacenter. Then, by adding a complex software suite like SAS into the mix forces diligence in understanding how to tune the whole solution not just a single component. SAS has completed and documented testing with other vendor solutions which includes CIFS and NFS (Walters et al., 2013).

For example, the following IBM GPFS configuration parameters are from work done as part of the SAS-IBM Alliance enablement efforts which have resulted in some best practices, white papers, and performance documents. The tunable parameters shown below are from a specific, detailed effort (Pattipati, 2012) which is further defined in the Section “SAS GRID MANAGER ARCHITECTURE UTILIZING IBM GPFS – EXAMPLE”. We include them here for completeness around our focus on IBM GPFS as an example for this paper.

Important! IBM, through its work with SAS personnel, has found that due to certain bugs fixes and performance improvements SAS-based deployments benefit from being at IBM GPFS version 3.5.0.15 or later. We advise installing at or moving to that release level or higher when possible.

Block size: 1 MB

The IBM XIV system uses 1 MB as stripe size, by default. Hence, creating file systems with 1 MB block size gave better performance.

Block allocation type: Cluster

The cluster allocation method is the default for GPFS clusters with eight or fewer nodes and for file systems with eight or fewer disks. Cluster block allocation type proved to be a better option for the workload run during the benchmark activity.

pagepool: At least 4 GB

The GPFS pagepool is used to cache user file data and file system metadata. pagepool is the pinned memory for each node. GPFS pagepool of at least 4GB of gave better performance for the workload used in the benchmark.

seqDiscardThreshold: 1 MB (default)

This parameter affects what happens when GPFS detects a sequential read access pattern. If a file is being re-read and its size is greater than 1 MB, then the file will not be cached in GPFS. However, in SAS analytics, many files are re-read and files are usually of size greater than 1 MB. Hence, increase this value based on the workload characteristics and size of the input files.

prefetchPct: 20 (default)

GPFS uses this as a guideline to limit how much pagepool space will be used for pre-fetch or write-behind buffers in the case of active sequential streams. If the workload does mostly sequential I/O, increasing it might benefit. SAS workloads predominantly do sequential I/O. Hence, increasing it to 40% might help performance.

Note: Changing just one tunable might not help in performance. It is important to understand how the different tunable work with each other and find out the right combination by running tests. For the workload used in benchmark activity, setting seqDiscardThreshold=1GB, pagepool=8GB and prefetchPct=40 gave slightly better performance.

maxMBpS: 5000

The maxMBpS value should be adjusted for the nodes to match the I/O throughput that the system is expected to support. A good rule of thumb is to set the maxMBpS value to twice the I/O throughput required of a system. For example, if a system has two 4 Gb host bus adapters (HBAs) (400 MBps per HBA) maxMBpS should be set to 1600.

GPFS mount options: rw, mtime, atime, dev

Other GPFS cluster tunable used:

Autoload: yes

dmapiFileHandleSize: 32

maxFilesToCache: 20000

prefetchThreads: 72

worker1Threads: 48

SAS® GRID MANAGER STORAGE – IBM GPFS USE CASE EXAMPLE

IBM GPFS supports a variety of cluster configurations independent of which file system features you use. Cluster configuration options can be characterized into three basic categories:

- Shared disk
- Network block I/O
- Multi-cluster Synchronously sharing data between clusters
- Multi-cluster Asynchronously sharing data between clusters.

A high-level discussion for the shared disk and network block I/O architectures will be provided. The multi-site cluster architectures will be briefly commented upon, but for the sake of this paper will be considered beyond the scope of the average SAS Grid Manager deployment, at least at the time of this writing.

SHARED DISK CLUSTER

A shared disk cluster is the most basic environment. In this configuration, the storage is directly attached to all SAS Grid Manager nodes in the cluster as shown in Figure 1. The direct connection means that each shared block device is available concurrently to all of the nodes in the GPFS cluster. Direct access means that the storage is accessible using a SCSI or other block level protocol using a SAN, Infiniband, iSCSI, Virtual IO interface or other block level IO connection technology.

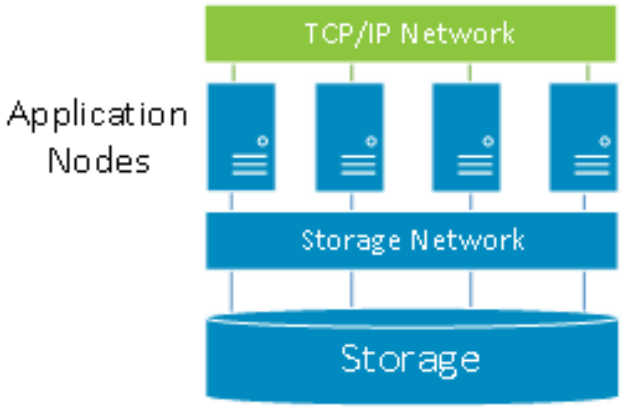

Figure 1. Shared disk architecture with all IBM GPFS servers connected to the storage through the Fibre Channel.

Figure 1 illustrates a GPFS cluster where all nodes are connected to a common fibre channel SAN. The nodes are connected to the storage using the SAN and to each other using a LAN. Data used by applications running on the GPFS nodes flows over the SAN and GPFS control information flows among the GPFS instances in the cluster over the LAN. This configuration is optimal when all nodes in the cluster need the highest performance access to the data Fadden, 2012).

NETWORK BLOCK I/O CLUSTER

As data storage requirements increase and new storage and connection technologies are released a single SAN may not be a sufficient or appropriate choice of storage connection technology. For environments where every node in the cluster is not attached to a single SAN, GPFS makes use of an integrated network block device capability. GPFS provides a block level interface over TCP/IP networks called the Network Shared Disk (NSD) protocol. Whether using the NSD protocol or a direct attachment to the SAN, the mounted file system looks the same to the application while GPFS transparently handles I/O requests.

GPFS clusters can use the NSD protocol to provide high speed data access to applications running on LAN-attached nodes. Data is served to these client nodes from one or more NSD servers. In this configuration, disks are attached only to the NSD servers. Each NSD server is attached to all or a portion of the disk collection. With GPFS you can define up to eight NSD servers per disk and it is recommended that at least two NSD servers are defined for each disk to avoid a single point of failure.

GPFS uses the NSD protocol over any TCP/IP capable network fabric. On Linux GPFS can use the VERBS RDMA protocol on compatible fabrics (such as Infiniband) to transfer data to NSD clients. The network fabric does not need to be dedicated to GPFS; but should provide sufficient bandwidth to meet your GPFS performance expectations and for applications which share the bandwidth. GPFS has the ability to define a preferred network subnet topology. For example, an architecture can designate separate IP subnets for intra-cluster communication and the public network.

This provides for a clearly defined separation of communication traffic and allows you to increase the throughput and possibly the number of nodes in a GPFS cluster. Allowing access to the same disk from multiple subnets means that all of the NSD clients do not have to be on a single physical network. For example, you can place groups of clients onto separate subnets that access a common set of disks through different NSD servers so not all NSD servers need to serve all clients. This can reduce the networking hardware costs and simplify the topology reducing support costs, providing greater scalability and greater overall performance (Fadden, 2012).

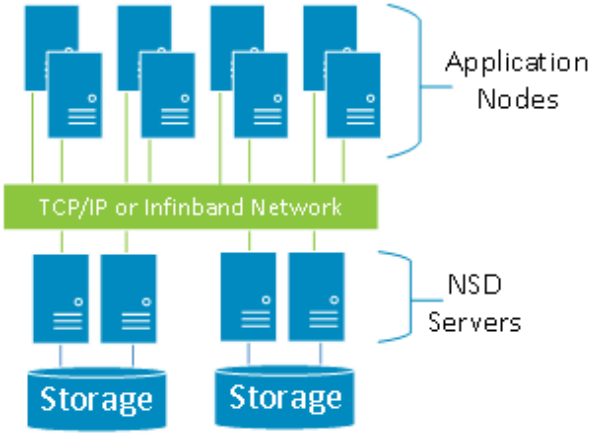

Figure 2. Network block I/O architecture with application (NSD server) nodes communicating across TCP/IP or Infiniband Network.

In this configuration, a subset of the total node population is defined as NSD server nodes. The NSD Server is responsible for the abstraction of disk data blocks across a TCP/IP or Infiniband VERBS (Linux only) based network. The fact that the disks are remote is transparent to the application. Figure 2 shows an example of a configuration where a set of compute nodes are connected to a set of NSD servers using a high-speed interconnect or an IP-based network such as Ethernet. In this example, data to the NSD servers flows over the SAN and both data and control information to the clients flow across the LAN. Since the NSD servers are serving data blocks from one or more devices, data access is similar to a SAN attached environment in that data flows from all servers simultaneously to each client. This parallel data access provides the best possible throughput to all clients. In addition, it provides the ability to scale up the throughput even to a common data set or a single file. The choice of how many nodes to configure as NSD servers is based on performance requirements, the network architecture, and the capabilities of the storage subsystems. High bandwidth LAN connections should be used for clusters requiring significant data transfer rates. This can include 1Gbit or 10 Gbit Ethernet. For additional performance or reliability, you can use link aggregation (EtherChannel or bonding), networking technologies like source based routing or higher performance networks such as Infiniband. The choice between SAN attachment and network block I/O is a performance and economic one. In general, using a SAN provides the highest performance; but the cost and management complexity of SANs for large clusters is often prohibitive. In these cases network block I/O provides an option (Fadden, 2012).

GPFS MULTICLUSTER AND ADVANCED FILE MANAGEMENT (AFM) OPTIONS

There are two methods available to share data across GPFS clusters: GPFS multi-cluster and a new feature called Active File Management (AFM).

GPFS Multi-cluster allows you to utilize the native GPFS protocol to share data across clusters. With the use of this feature you can allow other clusters to access one or more of your file systems and you can mount file systems that belong to other GPFS clusters for which you have been authorized. A multi-cluster environment allows the administrator to permit access to specific file systems from another GPFS cluster. This feature is intended to allow clusters to share data at higher performance levels than file sharing technologies like NFS or CIFS. It is not intended to replace such file sharing technologies which are optimized for desktop access or for access across unreliable network links.

Multi-cluster capability is useful for sharing across multiple clusters within a physical location or across multiple locations. Clusters are most often attached using a LAN; however, the cluster connection could also include a SAN. Multi-cluster environments are well suited to sharing data across clusters belonging to different organizations for collaborative computing, grouping sets of clients for administrative purposes, or implementing a global namespace across separate locations.

Active File Management (AFM) allows you to create associations between GPFS clusters. Now the location and flow of file data between GPFS clusters can be automated. Relationships between GPFS clusters using AFM are defined at the fileset level. A fileset in a file system can be created as a “cache” that provides a view to a file system in another GPFS cluster called the “home.” File data is moved into a cache fileset on demand. When a file is read, the

“in the cache fileset” file data is copied from the home into the cache fileset. Data consistency and file movement into and out of the cache is managed automatically by GPFS.

Cache filesets can be read-only or writeable. Cached data is locally read or written. On read, if the data is not in the “cache”, then GPFS automatically creates a copy of the data. When data is written into the cache the write operation completes locally then GPFS asynchronously pushes the changes back to the home location. You can define multiple cache filesets for each home data source. The number of cache relationships for each home is limited only by the bandwidth available at the home location. Placing a quota on a cache fileset causes the data to be cleaned (evicted) out of the cache automatically based on the space available. If you do not set a quota, a copy of the file data remains in the cache until manually evicted or deleted (Fadden, 2012).

SAS® GRID MANAGER ARCHITECTURE UTILIZING IBM GPFS - EXAMPLE

This section describes the deployment architecture for deployment of SAS Grid Manager on IBM Power 780 server with GPFS and XIV Storage System (Pattipati, 2012). The Power 780 server is configured with four Logical Partitions (LPARs) and each LPAR acts as a node in the grid. The LPARs are dedicated, which means that the processor resources are dedicated to an LPAR and they are not shared. VIOS with N-Port ID Virtualization (NPIV) was used for sharing physical Fibre Channel (FC) adapters among LPARs. FC adapters are mapped to the VIOS and virtual FC adapters are connected to the client LPARs. The SAS GPFS-based file systems are created on the LUNs mapped from XIV storage system and they are shared across the four logical grid nodes. All five physical FC adapters (10 FC ports) are directly mapped to the VIOS. On the VIOS, twenty virtual server FC adapters are created, with each physical FC port mapping to two of the twenty virtual server FC adapters. On the VIOS, five virtual server FC adapters are mapped to each of the four grid nodes. On each grid node, five client FC adapters are created and mapped to 5 of the virtual server FC adapters created on the VIOS. This effectively virtualizes the I/O through VIOS. This architecture as deployed is shown in Figure 3.

SOFTWARE

- SAS® 9.3 64-bit software
- Base SAS® 9.3
- SAS/STAT® 9.3
- SAS/CONNECT® 9.3
- SAS® Grid Manager 9.3
- IBM AIX OS 7.1.0.0
- IBM Virtual I/O Server (VIOS) 2.2.1.3
- IBM PowerVM Enterprise Edition
- IBM GPFS 3.4.0.7

HARDWARE

IBM Power 780 Server

- Architecture – IBM POWER7
- Cores - 16 (2 sockets)
- Processor clock speed: 3864 MHz Note: Power 780 server supports turbo core mode at 4.14 GHz. However, this mode is not used during the benchmark activity.
- Simultaneous multithreading (SMT) 4 enabled
- Memory: 256GB Note: Used 80GB total memory for all the grid nodes during the benchmark activity.
- Internal drives: Eighteen 300 GB (5.4TB) Note: Used for booting logical partitions (LPARs) and VIOS, not used for SAS data. SAS data is on the XIV system.

IBM XIV Storage System

From the XIV GUI, a storage pool was created and 16 volumes (LUNs), each with a size of 395 GB, were created from the pool. The volumes are then mapped to the grid cluster, consisting of the four grid nodes. From the grid nodes, the LUNs are mapped as logical disks (hdisks). /saswork and /sasdata GPFS file systems were created with eight LUNs each and mounted on all the grid nodes. Note: The XIV system uses the stripe size of 1 MB by default; hence, the file systems were created with 1 MB block size for optimizing the throughput.

- XIV system version: 10.2.4.c-1
- System ID: 1300316 (316)
- Machine model: A14 / 2810 (Gen2)
- Drives: 180 SATA drives each with 1 TB capacity and 7200 rpm speed

- Usable space: 79 TB
- Modules: 15 with 12 drives each
- Memory / Cache: 120 GB
- Stripe size: 1 MB (Default)
- Six 4 Gb dual-port Fibre Channel (FC) adapters (12 ports) connected to storage area network (SAN)

SAN connectivity

Two redundant IBM System Storage SAN24B-4 Express Brocade Fiber Switch switches are connected in the SAN fabric with NPIV enabled. They connect the XIV system to the Power 780 server. The IBM Power 780 server is connected to the SAN Switches by five 8 Gb dual-port FC adapters (10 ports). The XIV system is connected to the SAN Switches by six 4 Gb dual-port FC adapters (12 ports). The ports connecting the Power 780 server to the SAN Switch are configured to have port speeds N8 (Auto-negotiate) and the ports connecting the switches to the XIV system are configured to have port speeds N4 (Auto-negotiate). Appropriate zoning is done on both the SAN Switches. Note: The zoning should include the worldwide port names (WWPNs) from the client virtual FC adapters created on the grid nodes.

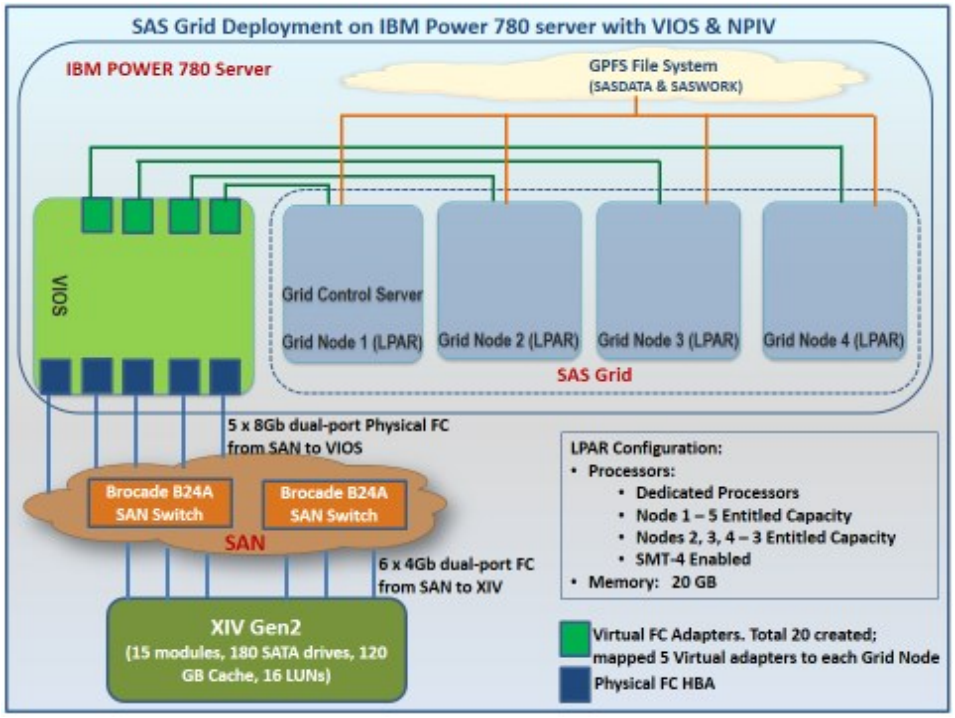

Figure 3. SAS® Grid Manager deployment architecture on an IBM Power 780 server with VIOS and NPIV.

The objective of this benchmark testing was to document the best practice tuning guidelines for achieving optimal performance of SAS Grid Manager on Power / AIX with GPFS and XIV Storage System. With optimal tuning of the entire stack, the benchmark results show how the architecture supports the computing and I/O requirements of the SAS workloads running on the grid.

The 80 session simulation benchmark was run on the grid to simulate the workload of a typical Foundation SAS customer that uses SAS Grid Manager computing. The goal was to evaluate the multiuser performance of the SAS software on the IBM Power platform. The workload has an emphasis on I/O, which SAS software uses heavily while running a typical Foundation SAS program execution.

Summary of the benchmark results (Pattipati, 2012):

- The deployment architecture delivered a peak I/O throughput of 1.5 GBps and sustained throughput of 1.45 GBps for the 80 session workload.
- More than 100 MB per core per sec sustained SAS I/O throughput, considering that the grid has only 14 cores assigned to the nodes in dedicated mode.
- The processor usage of the grid during the workload is 61%, processor wait is 11%, and processor idle is 28%.

SAS GRID MANAGER STORAGE DEPLOYMENT CHECKLIST

The overall goal of deploying a SAS Grid Manager-based solution is to aid the Business in working with data in an analytical fashion to impact business objectives tied to revenue, costs, customer satisfaction, and more. However, the 'actual' SAS Grid Manager solution deployment should be a project-based merge of business needs and IT infrastructure considerations to achieve optimal success. Numerous times we have seen much focus on the software and/or server aspects and lessened focus on the storage aspects. The table constructed as a 'deployment checklist', although still high-level, is meant to add focus on the overall planning and deployment around the storage requirements feedback loop through to actual storage provisioning for file system creation. Every SAS Grid Manager Client solution is somewhat unique in these regards, but this blueprint should help draw attention to the storage aspects.

Planning/Deployment Steps	Comments
Define SAS Grid Manager and SAS Software stack needed	Work with SAS to define the overall software architecture for the grid. (Includes software products, tiers, number of SAS sessions, etc.).
Provide input on server hardware – if a preference exists	Define existing Client server infrastructure – if standardized across Client IT or if vendor preferences exist. If not, work with SAS to review enablement papers to select server architecture.
Refine the Physical Implementation Reference Diagram (PIRD)	Work with SAS to refine and understand how the SAS software will reside on the server infrastructure – number of cores, processors, memory requirements, OS(es), etc.
Use final PIRD assumptions to drive IP and FC network connections including file system requirements	Work with SAS to understand the expected storage and networking requirements to meet expected performance goals. This includes definition of file systems (location and sizes), IP network bandwidth, FC network bandwidth. Note: FC is not necessarily needed if NAS storage is part of the solution.
Procure and/or repurpose IT assets for SAS environment	Ensure servers, networks, storage, shared file system, OSes, etc. are in place for software deployment.
Deploy servers	Install, configure, and test servers and operating systems to meet SAS best practices as defined in white papers off of SAS Support site.
Deploy storage and shared file system(s)	Install, configure, and test storage and especially shared files system components to ensure the expected MBps/core bandwidth is achieved.
Install SAS depot	Ensure all purchased/required SAS software is in the depot in order for installation.
Install Platform Suite for SAS components	Follow instructions in the install guide and test as shown in the guide to ensure all grid nodes can run 'jobs'
Install SAS including SAS Grid Manager	Install SAS components and test as instructed and as your organization requires.
Allow access to Power Users to test and begin production use per organizational policy	This step differs widely due to organizational requirements, size, compliance, security, etc.

Table 1. Typical planning/deployment steps for a shared file system to meet SAS® Grid Manager requirements. The overall steps were included for a whole deployment for completeness.

CONCLUSION

As we enjoy "living in the moment" embedded in the flow of Information Technology time, we often have to recognize and realize that transformative moments are on the horizon. Many solutions, which had been considered their own infrastructure, like SAS, Oracle, and SAP have had to find their way to peacefully coexist with the modern data center. However, the basic needs and characteristics of how these types of applications operate within the context of IT infrastructure has not changed as much. It is true that SAS now leverages some aspects of Hadoop, for example, but much SAS workload, where SAS Grid Manager workloads fall, is still more traditional. Therefore, the heavy focus on sequential I/O has not changed. The requirements for the storage environment only become steeper to keep modern server technology CPUs and buses full and busy churning SAS cycles. This paper focuses on introducing the

transformation that corporate IT is undergoing in storage and how it is now more often viewed as a commodity base for utility IT. A core driver is the Cloud view of IT as a utility via a menu of service levels with defined costs. SAS, and these other applications, now must fit in these data center SLA-centric models while still enjoying a cost effective peak I/O.

As we mentioned, Google Drive and Dropbox are technologies that the “average Joe or Joelle” uses now as a Cloud storage offering. Behind the scenes, all of that storage is being virtualized in an enterprise data center, which SAS is now seeing as a more typical deployment scenario. Servers and storage are oriented as utility components with defined service levels; yet the need for optimal SAS performance still exists. Furthermore, we discussed the requirements for scalability and optimal configuration of the software and hardware environment. The two authors could be called “old school”, but still recall using CIFS and NFS for print, file, and general file system serving, like /home/user1. These two IP-based technologies struggle at true technical computing which is where SAS sits. It is the I/O performance that has most architectures using a software-based shared file system solution. We used IBM GPFS as a proxy to look at features and functionality to meet SAS needs (and even those potentially imposed by a Client’s core IT). As storage changes over time, the ease of adding storage, migrating data, and deploying cost efficient storage pools in tiers is best handled with these vendor developed and supported solutions. To remind, architectural changes over time has not removed the need for high performance I/O with SAS.

Finally, the authors show a real world architecture and its details which meets these I/O requirements (100 MBps/core). A blueprint for approaching a SAS Grid Manager definition, planning, and deployment is provided to help keep focus on the not so sexy yet still very important storage and file system components. Every Client will likely have their own unique needs on how SAS is configured: due to the SAS products being purchased, whether there is a separation of dev, test, or prod, and NAS or SAN storage is the underlying storage pool technology. However, there is no substitute for planning and testing. We encourage SAS users and especially SAS Grid Manager users to do plenty of both (planning and testing) through the conceptual, procurement, and deployment processes. This should help ensure that expected SAS I/O storage performance is met!

REFERENCES

Various IBM Corporation contributors. “GPFS Frequently Asked Questions and Answers.” 2014 but updated frequently. Available at http://publib.boulder.ibm.com/infocenter/clresctr/vxrx/index.jsp?topic=%2Fcom.ibm.cluster.gpfs.doc%2Fgpfs_faqs%2Fgpfsclustersfaq.html

Barbara Walters, Ken Gahagan, Leigh Ihnen, and Vicki Jones. 2013. “A Survey of Shared File Systems: Determining the Best Choice for Your Distributed Applications.” Proceedings of the SAS® Global Forum 2013 Conference. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/resources/papers/proceedings13/484-2013.pdf .

Scott Fadden. August 2012. “An Introduction to GPFS Version 3.5 - Technologies that enable the management of big data.” IBM Corporation White Paper. IBM Corporation. Available at http://www-03.ibm.com/systems/resources/introduction-to-gpfs-3-5.pdf .

Narayana Pattipati. September 2012. “SAS 9.3 grid deployment on IBM Power servers with IBM XIV Storage System and IBM GPFS.” IBM Systems and Technology Group ISV Enablement White Paper. IBM Corporation. Available at http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP102192 .

ACKNOWLEDGMENTS

The authors would like to gratefully acknowledge the IBM-SAS Alliances team most notably Leamon Hall Jr.. Frank Battaglia, Justin Jones, Narayana Pattipati, and Kevin Go. We also would like to acknowledge SAS personnel - Margaret Crevar, Ken Gahagan, Tony Brown, and Rich Pletcher - for their time and feedback regarding Client solutions and SAS testing. It is a pleasure to be able to work with such talented and devoted people.

RECOMMENDED READING

- *SAS® Grid Computing For Dummies®*
- *IBM® GPFS For Dummies®*

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Name: Gregg Rohaly
Organization: IBM Platform Computing
Address: 100 SAS Campus Drive, C4223

City, State ZIP: Cary, NC 27513
Work Phone: 919.930.7180 (M) or 919.531.3471 (D)
Email: Gregg.Rohaly@sas.com or grohaly@us.ibm.com

Name: Harry Seifert
Organization: IBM Sales and Distribution
Address:
City, State ZIP:
Work Phone: 270.207.8274 (M) or 720.396.7015 (D)
Email: seifert@us.ibm.com