CMST SI(1) 75-85 (2006)

DOI:10.12921/cmst.2006.SI.01.75-85

COMPUTATIONAL METHODS IN SCIENCE AND TECHNOLOGY **Special Issue 2006**, 75-85

Global Block Services for the Grid. A New Architecture from SGI and YotaYota

Silicon Graphics, Inc. and YotaYota

(Rec. November 2005)

Abstract: SGI and YottaYotta have developed a new architecture for data grids that leverages globally-shared file systems, automated multi-site data locality management, and multi-site authentication and access control. This document presents results of studies comparing performance for the new architecture to that of conventional data grid architectures.

Key words: data grids, globally-shared filesystems, NFS, CIFS, CXFS, WAN, failover, backup, DMF

1. THE CHALLENGE OF DATA LOCALITY

To ensure optimal use of grid resources, data must be both "local" and "global" at the same time. Users must ensure that their data is "local" so that processing at any given site can progress without the delays of remote I/O or file transfer. At the same time, the data must be "global" so that it remains consistent between sites. This dilemma presents one of the great un-solved problems in leveraging both compute and storage resources across geography and one of the major obstacles to widespread deployment of data grids.

1.1. The File Copy Paradigm

Most data grids attempt to address data locality through some variant of the File Transfer Protocol (FTP). With this approach, data is "shared" through duplication. In order to duplicate a file with FTP, data processing operations on the file must be suspended. The user then logs in to a remote file system, initiates an FTP session, waits for the file transfer to complete from source to target disk, and then terminates the session. This process results in significant delay before data access can begin.

More importantly, no real "sharing" of data occurs at all. Once new copies have been generated, FTP takes no responsibility to ensure the ongoing consistency between copies. If the user changes a local copy of a "shared" file, it is his/her responsibility to re-transmit new copies to other sites. Even though only a few blocks of the file may have changed, the entire file must be re-transmitted to all potentially interested parties. The process of updating and rebroadcasting files through FTP can consume considerable WAN bandwidth. It can also generate multiple non-current versions of a file that must be stored, managed, and archived in multiple locations.

In addition to the above challenges, there is the issue of performance. Most standard FTP servers do not perform well over large distances. For example, even if we ignore the delays required to initiate and manage the transfer session, a standard FTP server requires more than four hours to transmit a 10 GB file over 5000 km on a 1 GbE link (see Section 2). In this case, transmission time is constrained not by bandwidth but by round-trip latency – the time to transfer the file does not decrease significantly regardless of how much wide area bandwidth is available.

While specialized FTP variants such as GridFTP can yield much better performance if properly tuned, these variants are not aware of the layout of files on physical disk, and, hence, are often constrained by I/O to storage. Further, all versions of FTP suffer from the inherent weaknesses of the file copy paradigm: excess data duplication, excess WAN transfer, and increased challenges and risks associated with version management. In many cases, the interruption required to copy data back and forth between sites is greater than the time required to simply process the entire job at a single site.

1.2. The Network File System Paradigm

Recognizing the shortcomings of FTP, some organizations have explored using variants of network file systems such as NFS and CIFS for inter-site data sharing. With the network file system model, all servers have access to a globally consistent file system, so there is no need to copy files back and forth.

However, while network file systems enable true data sharing, typically they do this by routing all access to a given file through a single master server. Unfortunately, this strategy can lead to serious performance penalties, particularly when the file sharing is conducted over significant distance. Even when the processor of the master server itself is not a bottleneck to data access, insisting that data be accessed through a single channel can seriously constrain network routing and limit overall network efficiency. Further, when deployed over long distance, network file

systems are notoriously inefficient due to heavy reliance on remote procedure calls (RPCs) between server and client.

1.3. The Clustered File System Paradigm

Clustered file systems overcome some of the weaknesses of network file systems by reducing dependence on RPCs and allowing multiple servers to access the same files using independent paths to data storage. Since all sites share direct access to a single data image, replication is unnecessary. Further, since each node within the cluster can perform direct I/O through an independent channel, all servers in the cluster can perform concurrent I/O to the shared image and achieve greater performance scalability. Global data consistency is guaranteed by a meta-data server that controls file access.

While a clustered file system can provide more scalable access to shared data than a network file system, inter-site transport latency can still have a devastating impact on its performance in WAN deployments. With conventional clustered file systems, all access to a given logical disk is through a unique storage controller in a single site. A theoretical upper bound on steadystate throughput for single-threaded data exchange between a server and a remote disk is given by:

$$T \le \frac{B}{1 + R \times B / D} \quad , \qquad (1)$$

where T is steady state throughput, D is the active data window for each exchange, R is the round trip time, and B is the available bandwidth. Note that as $R \to D/B$, transport latency becomes the dominant constraint on throughput.

Thus, even though all servers mounting a conventional clustered file system have logical access to shared data, the problem of managing the physical locality of the data and providing acceptable application performance remains. This is the problem that the SGI/YottaYotta global file sharing solution is designed to address.

2. SGI/YOTTAYOTTA DATA GRID

The SGI/YottaYotta data grid complements SGI's clustered file system, CXFS, with YottaYotta's distributed block system, (Y2DBS), to offer a new standard in data grid service. Y2DBS is mounted on an array of YottaYotta's GSX 3000 NetStorage Control Nodes known as a "NetStorager System" (NS)[1]. The combined SGI/YottaYotta solution provides access to a globally consistent name space at near-local performance. It also offers a number of data protection services that ensure data consistency and continuous data access even in the event of site outage or network failure.

To understand the difference between the SGI/YottaYotta data grid and other models for global file sharing, consider Fig. 1. Note that with either NFS or a conventional clustered file system, all file server nodes perform Small Computer System Interface (SCSI) exchanges directly with a target Logical Unit Number (LUN). By contrast, with the SGI/YottaYotta data grid, each server interacts with a local distributed block system (DBS) node that acts as a SCSI proxy for the intended LUN. The distributed block system then maintains consistency through block-level locking and coherence mechanisms that may involve meta-data exchange with other block server nodes.

2.1. Global Block Services

The SGI/YottaYotta solution offers a suite of Global Block Services that enable a new paradigm for Global Data Sharing and Fully-Active Continuity of Operations (COOP). These include Data Localization, WAN Optimization, and Fully-Active COOP services. Data Localization services provide improved distributed data processing performance, more efficient utilization of WAN resources, reduced data duplication, and reduced time to solution for data-intensive workflows.

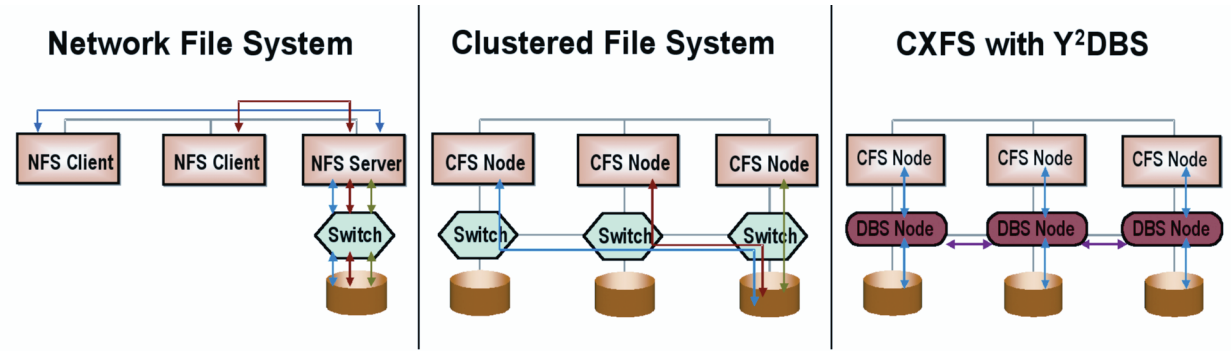

Fig. 1. Data sharing *via* NFS, *via* CXFS with serial block system, and *via* CXFS with YottaYotta's Y2DBS

[1] More information on the NetStorage Control Node can be obtained at www.yottayotta.com.

2.2. Data Localization Services

Geo-cache

Each GSX 3000 NetStorage Control Node provides up to 32 GB of cache memory. Cache resources from individual nodes aggregate to form a globally coherent cache pool. By serving I/O requests from local RAM or disk cache, the SGI/YottaYotta data grid improves performance and dramatically reduces data transfer over theWAN.

Site-Aware Geographic RAID

The SGI/YottaYotta data grid provides geographic RAID services. Unlike traditional data replication, which can offer only a passive copy of the data at a remote site, the SGI/YottaYotta data grid can offer fully active, coherent copies at multiple locations, allowing sites to perform both read and write operations to a single distributed data image. Read operations are automatically routed to the nearest copy of the data image. Write operations only transmit updated blocks between sites. This enables many sites to interact with a shared data image while reducing WAN traffic.

Distributed Meta-Data Directories

The NetStorage Control Node maintains global data consistency through a distributed directory service. Directory segments migrate between nodes according to data access patterns. This minimizes meta-data exchanges, reduces WAN traffic, and enhances performance.

Access-Sensitive Data Migration

In order to localize access to shared data, the SGI/YottaYotta data grid leverages the SGI Data Migration Facility (DMF) and Y2DBS to automatically migrate data between persistent storage in different sites according to usage and need. Administrators can set policies governing the movement of data between sites and storage media.

2.3. WAN Optimization Services

While strategies to store and/or cache data where it is needed can minimize inter-site data exchange and improve performance, ultimately, there are many applications for which real-time data transfer across the WAN is unavoidable. As a result, highspeed WAN data transfer is a critical element of the SGI/YottaYotta data grid solution. The SGI/YottaYotta solution leverages a number of key innovations to accelerate data transmission and improve utilization of available WAN bandwidth over larger distances.

RDMA

NetStorage Control Nodes can perform direct memory exchanges through RDMA. This way, data exchanges bypass the OS kernel and the CPU memory bus of the file servers as well as the overhead of Fibre Channel re-encapsulation.

TCP Forwarding

In most cases, congestion and packet loss occur in the "last mile" of transport. Long-distance TCP sessions are particularly vulnerable to packet loss because they require very large congestion windows to overcome the performance penalties of transport latency. Consequently, LAN congestion can devastate WAN data transfer. To address this problem, each NetStorage Control Node provides a TCP forwarding service by which a single long distance TCP session is decomposed into distinct sessions over the LAN and over the WAN. A benefit is that users end up with the best of both network domains: resilience to packet loss in the LAN and high-speed performance over the WAN.

Message Gathering

Acknowledgement-based flow control and reliability services become less efficient as transport latency increases. To improve efficiency, the SGI/YottaYotta data grid solution employs message gathering to increase the active data window associated with each acknowledgment. Pre-fetch and writebehind policies optimize the active data window for a given I/O profile and round trip latency.

Parallel TCP sessions

Another key innovation of the SGI/YottaYotta solution is to introduce a transparent TCP session management layer that launches multiple concurrent TCP sessions for a given I/O operation. By increasing concurrency, this strategy both reduces the penalty of round trip latency on steady-state throughput and better protects WAN traffic from the penalties of packet loss.

Parallel data movers

Applications that require very high-speed data transfer over long distance can benefit from parallelizing data movement across an array of data movers within the SGI/YottaYotta data grid. Users can achieve sustained transfer rates in excess of 10 Gbps over 5000 km without modifying their applications.

2.4. Fully-Active COOP Services

Protecting data and data access is one of the great challenges and, potentially, one of the great benefits of multi-site storage deployments. On the one hand, distributed systems that guarantee global coherence can be vulnerable to loss of network connectivity. At the same time, if network partition and data consistency policies are properly managed, a multi-site data system can provide on-going data access even in the face of site disasters.

Multi-Site Clustered Replication (Synchronous, Asynchronous, Semi-synchronous)

In the SGI/YottaYotta data grid, all sites perform I/O to mirrored data through local NetStorage Control Nodes. These nodes route read requests to the nearest mirror and, on write operations, transmit only updated blocks to remote mirrors. Propagation of write updates to remote sites may be synchronous, asynchronous, or semi-synchronous. However, regardless of which update mode is chosen, all sites always access the current data image. If updated data has not yet been transmitted to remote sites, read requests from these sites will automatically re-route to the location with the latest updates.

Automatic Site Fail-over

In the event of a site failure, local I/O fails-over to the nearest surviving mirror. Meanwhile, all NetStorage Control Nodes within the multi-site cluster can work in parallel to rebuild the failed storage site. When the restoration is complete, I/O automatically fails-back to the reconstructed mirror.

Network Partition Management and Incremental Re-synch

To protect against data corruption due to network partition, the SGI/YottaYotta data grid provides fencing and inter-site incremental re-synch services. Distinct network partition management and incremental re-synch policies can be configured for each multi-site file system. These policies ensure continued operation and data consistency despite network or site failures.

Upon network recovery, all sites can immediately access the current data image even though updates may not yet have propagated through to physical disk in each site. This is achieved through temporary I/O redirection within the NetStorage Control Node. Meanwhile, updates that occurred during the outage are propagated at high-speed to each site provisioned with a physical mirror. This way, all sites can benefit from improved data locality soon after the network recovers.

Distributed PiT

While inter-site mirroring protects against data loss due to failures, it does not protect against accidental data corruption due to human error and it does not guarantee that mirrors will be in an application-layer consistent state at the time of failure. For these reasons, it is necessary to take periodic snapshots of the multi-site file system. This way, administrators always have the option to roll back to one or more known consistent states.

The SGI/YottaYotta data grid can provide up to four concurrent multi-site snapshots and guarantees multi-site global consistency of each snapshot. If any site fails-over active I/O to a recent snapshot during a network outage or a remote site failure, remote mirrors of the snapshot will be incrementally updated at other sites once the network and storage recovers.

To minimize storage consumption, the snapshots are logical (*e.g.*, copy-on-write) rather than physical. And, to minimize WAN traffic, all copy-on-write exchanges occur within sites rather than between them. To improve performance after a fail-over to a logical snapshot, the SGI/YottaYotta grid can perform dynamic volume migrations. This means that the logical image is promoted to a physical one during active I/O in a fashion that is completely transparent to end-users.

Centralized Backup and High-speed Restore

When deployed with YottaYotta's distributed block system, SGI's InfiniteStorage Data Migration Facility (DMF) provides a unique capability to migrate data transparently not just between different storage media, but also between different locations. Geographic caching, LUN virtualization, and high-speed data transfer services within the NS enable DMF to perform highspeed backup and restore operations even over long distance. An important benefit is that a single centralized archive facility can serve many "satellite" locations. Another benefit is that satellite locations can "thin-provision" their local storage. In other words, they can reduce the amount of storage they provision locally while still exporting the entire data image stored at the central archive. Files automatically migrate back and forth between satellite locations and the central archive in response to administrative policy and user need.

3. EVALUATING THE SGI/YOTTAYOTTA DATA GRID

In this section, we evaluate performance of the SGI/-YottaYotta data grid for three applications:

- WAN Data exchange
- Distributed Data Processing
- Centralized Archiving

3.1. WAN Data Exchange

As noted above, the SGI/YottaYotta data grid is designed to cache or mirror data where it is needed and, in this fashion, reduce WAN data exchanges. However, some transfer of data across the WAN is inescapable. Several applications require real-time, high-speed WAN transfer to function effectively in geographic contexts.

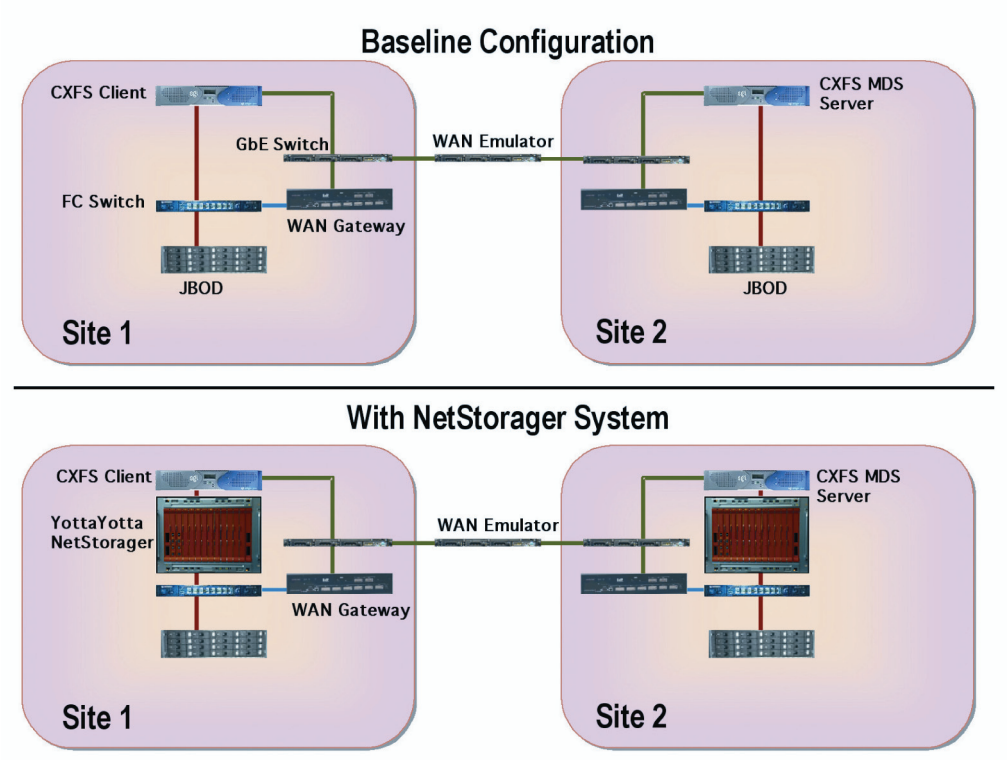

Fig. 2. Two-site distributed system used for WAN data transfer testing. Testing was performed on a two-site CXFS cluster with and without YottaYotta

This section provides performance results for common WAN data transfer applications with YottaYotta's GSX 2400 NetStorager System. To better represent the user's experience, we focus on the actual time to complete various data transfer operations with and without the NS in the data path. In each case, both NS cache and host cache is invalidated before launching the application and 100% of data is physically transferred over the WAN during the course of the measurement.

Figure 2 shows the distributed system configurations used for benchmarking WAN data transfer performance. The baseline configuration is a two-node CXFS stretch cluster with default mount options. The test configuration is altered from baseline only by including a two-blade NS in the data path. Only one blade was deployed in each NetStorager System (the minimum configuration). The server cluster and the NS were each provisioned with a 1 GbE interconnect. The two interconnect networks were then switched onto a single WAN 1GbE link. A DummyNet WAN Emulator was used to emulate distance and packet loss due to congestion on this link. A Xyratex1600 FC JBOD was deployed for storage in each site. Source and target LUNs were configured as RAID-0, 8 disks. All data transfer was from Site 2 to Site 1.

3.1.1. Reads from Remote Disk

In many use models for the SGI/YottaYotta data grid, reads from remote disk will be a frequent cause for real-time data transfer across the WAN. Reads from remote disk occur when a site requires access to data that has not already been cached, mirrored, or temporarily migrated by the NS.

Write operations, by contrast, do not normally require real-time transmission over long-distance. Since NS metadata exchanges are synchronous and guarantee global consistency, writes to remote disk can be delayed, aggregated, and performed at near wire-speed. If a remote site reads updated data before that data has propagated to physical disk, the NS serves the read request from its distributed cache pool – an operation that always requires less time than a read from remote disk.

By benchmarking performance for read operations from remote disk, we obtain an upper bound on response time for all read operations. Figure 3 shows time to read a 2 GB file from disk across a range of distances. In each case, measured times are for single-thread, sequential reads with the CXFS default block size of 64 KB.

Access times for the baseline configuration increase linearly for larger distances. Actual sustained WAN trans-

fer rates for the baseline configuration agree closely with the theoretical upper bound given by Eq. (1). For instance, at 4800 km, read performance drops to just 1.36 MB/s for the baseline configuration. With the SGI/YottaYotta data grid, the time to read the file does not increase significantly with distance. Steady state WAN transfer rate for a single threaded read operation remains constant at about 62 MB/s even over 4800 km. More detailed analysis reveals that approximately 60% of the performance enhancement for the SGI/YottaYotta solution is due to message gathering. The rest of the enhancement comes from launching multiple concurrent TCP sessions for each data transfer (see Fig. 4).

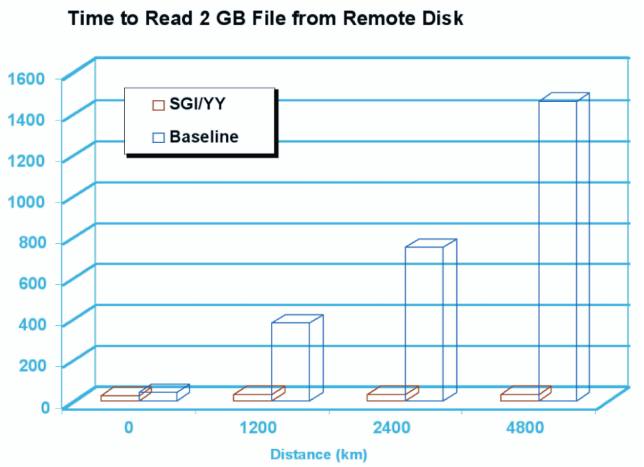

Fig. 3. Time to read a 2 GB file from disk across a range of distances. When CXFS is used with a NS, the time to complete the read operation stays relatively constant regardless of distance

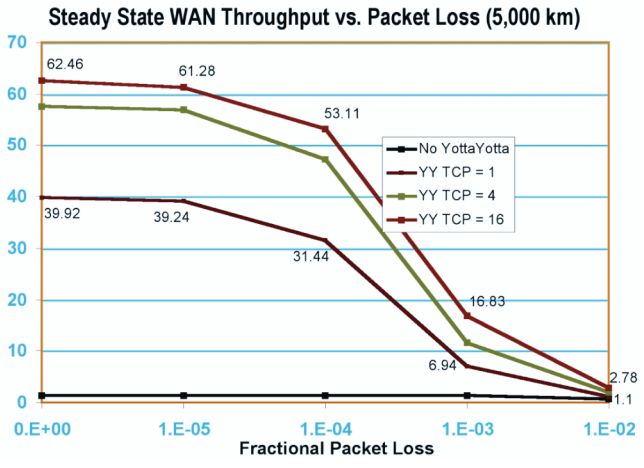

Fig. 4. Steady state WAN throughput for single-threaded sequential read operations across 5000 km. Increasing the number of concurrent TCP sessions improves performance and relative resistance to packet loss

3.1.2. Impact of Congestion and Packet Loss

The NetStorage Control Node allows storage administrators to configure TCP concurrency on a per LUN basis. For example, with TCP concurrency set at 16, a single threaded remote operation will launch up to 16 concurrent TCP sessions across the WAN. By increasing the number of concurrent TCP sessions, administrators can increase both WAN throughput (particularly for OLTP workloads) and relative resilience to packet loss.

Figure 4 shows steady-state WAN throughput for singlethreaded read operations across 5000 km. Note that with no packet loss, increasing the number of TCP sessions only increases performance by 56%. As fractional packet loss increases, the relative performance benefit increases. When the loss rates increases to 1 out of every 100 packets, the performance enhancement of TCP concurrency increases to 152%.

3.1.3. File Replication: FTP and File System Copy

Next to remote I/O operations, the most frequent application for real-time WAN transfer is likely to be file replication through FTP or CXFS file copy. Even though, the SGI/YottaYotta data grid provides multi-site consistent access to shared files, there will continue to be a need for users to create local copies of files for various purposes. When the NetStorage Control Node does not have access to a given file through its local cache or a local disk mirror, file replication will trigger real-time data transfer across the WAN.

Figure 5 shows times to replicate a 2 GB file using the FTP client/server software installed with IRIX 6.5.23.

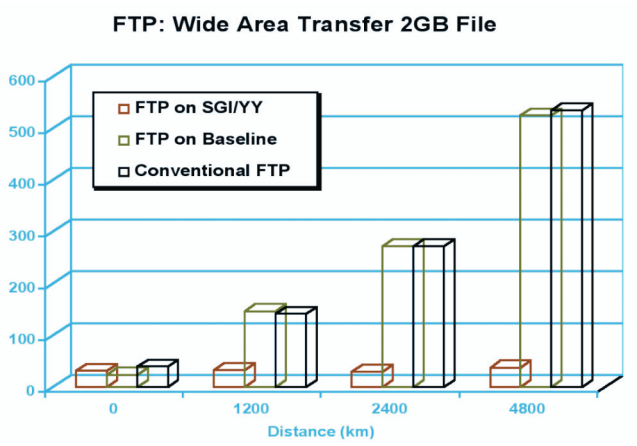

Fig. 5. Times to complete FTP of a 2 GB file across distances ranging to 4800 km

The blue columns show transfer times for a conventional FTP transfer. In this case, the FTP server is mounted in Site 2 while the FTP client is in Site 1. Both the server and client perform local I/O while conducting an FTP ex-

change over the WAN. Note that transfer time increases linearly with distance between the source and target volumes.

The green columns show transfer times when the FTP exchange occurs over the baseline CXFS configuration. In this case, the FTP server and client are both in Site 1. The server reads the file over the WAN from a source volume in Site 2 and performs a local FTP exchange with a client that writes the file locally to a volume in Site 1. Note that transfer times increase linearly with distance and are comparable to the transfer times for conventional FTP.

The orange columns show transfer times for FTP on the SGI/ YottaYotta data grid. The only change from the baseline configuration is the introduction of YottaYotta into the data path. Note that the time to complete the transfer increases only marginally as distance increases to 5000 km.

Figure 6 shows times to replicate a 2 GB file across distance using the CXFS file copy command. For the baseline configuration, times to replicate increase linearly with distance and are ~ 40% greater than with FTP. When the NetStorager System is added, replication times do not change significantly with distance and are virtually the same as for FTP on the same configuration.

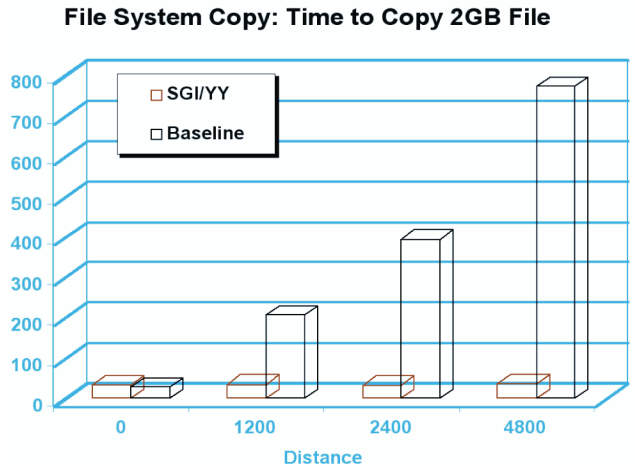

Fig. 6. Times to complete a 2GB CXFS file copy operation across distances ranging to 4800 km

Figure 7 shows times to replicate a 2 GB file across distance using the NFS file copy command. The blue columns show file copy times for conventional NFS when the NFS server and client are in different sites. The red columns show NFS copy times when the NFS server is mounted on a CXFS client node. Note that while copy times increase linearly with distance, there is a significant performance enhancement achieved by mounting NFS on CXFS. The orange columns show NFS copy times when the NetStorager System is added to the data path. Note that copy times do not change with the introduction of distance.

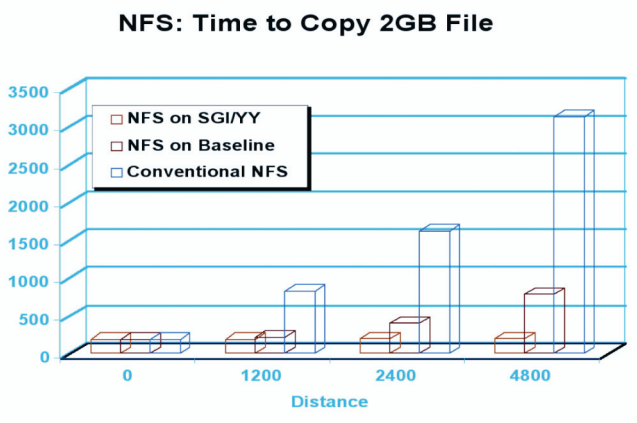

Fig. 7. Times to complete a 2GB NFS file copy operation across distances ranging to 4800 km

In fact, steady-state transfer is limited by the local exchange between the NFS client and NFS server to ~10.8 MB/s.

3.2. Multi-Site Distributed Processing: Performance Scalability

One of the principle benefits of the SGI/YottaYotta data grid is improved distributed processing performance. In this section, we examine aggregate throughput to storage when servers in a three-site CXFS cluster perform I/O to a single file system. We initiate distributed workloads with Platform LSF MultiCluster a distributed resource scheduler developed by Platform Computing.

Figure 8 shows the system configuration for three-site distributed processing tests. We conduct performance benchmarking both with and without a NetStorager System in the data path. Again, each NetStorager System has only a single blade and a single 1GbE link provides connectivity between the sites. DummyNet WAN emulators introduce round-trip latency equivalent to inter-site distances of 1000 km, 5000 km and 6000 km. In all cases, we use the default settings for the clustered file system and for the volume manager. In particular, the block size for all I/O is fixed at 64 KB. While the system can support transparent data localization through mirroring and migration, we focus on the case in which all data is stored at a single site – the worst scenario from a performance perspective. The file system mounts on a RAID 0, 2-disk LUN exported by the Clariion FC4700.

3.2.1. Single Cluster, Small File Performance

Figure 9 shows sequential read/write performance for access to:

- local storage (*i.e.*, compute servers and storage are at the same site),
- remote storage *(i.e.*, compute servers and storage are separated by 5000 km)

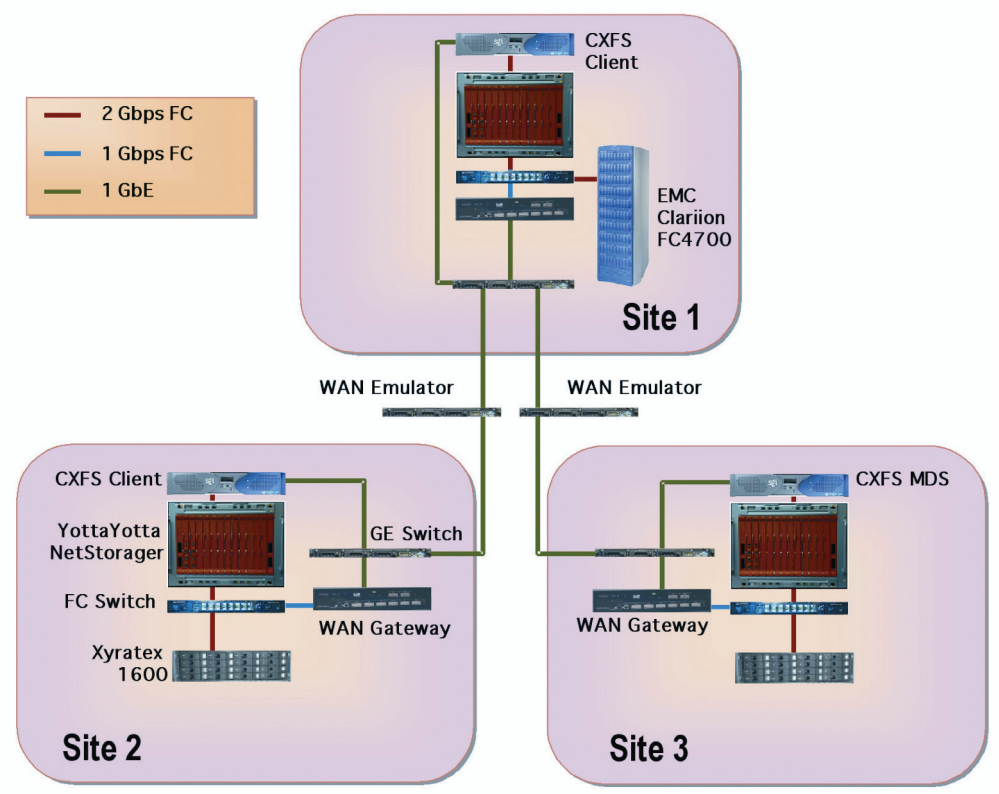

Fig. 8. System configuration for multi-site distributed processing testing. Performance benchmarking was performed both with and without YottaYotta in the data path

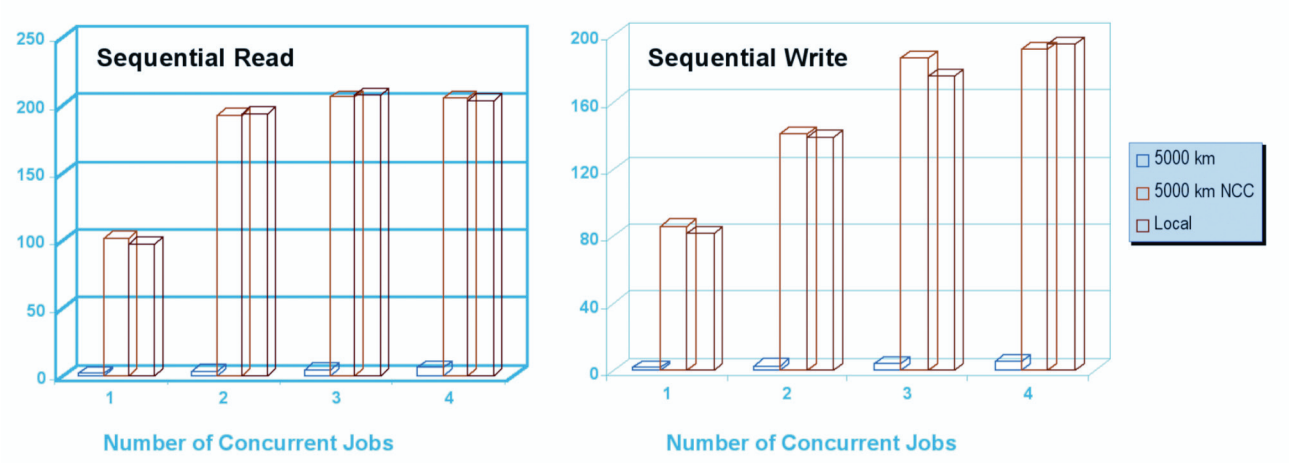

Fig. 9. Multi-job small file concurrent read and write performance to local storage, across 5000 km, and across 5000 km with the NS in the data path

- remote storage with the NetStorager System in the data-path.

In each case, we use LSF MultiCluster to accept job submissions every 40 seconds. Each job initiates a sequential read or write of a distinct 500 MB file.

Note that for both read and write workloads, when the NS is not in the data path, throughput to the remote server cluster throttles down to less than 3% of local throughput. For these cases, throughput is constrained by transport latency and, to an excellent approximation, is predicted by Equation (1).

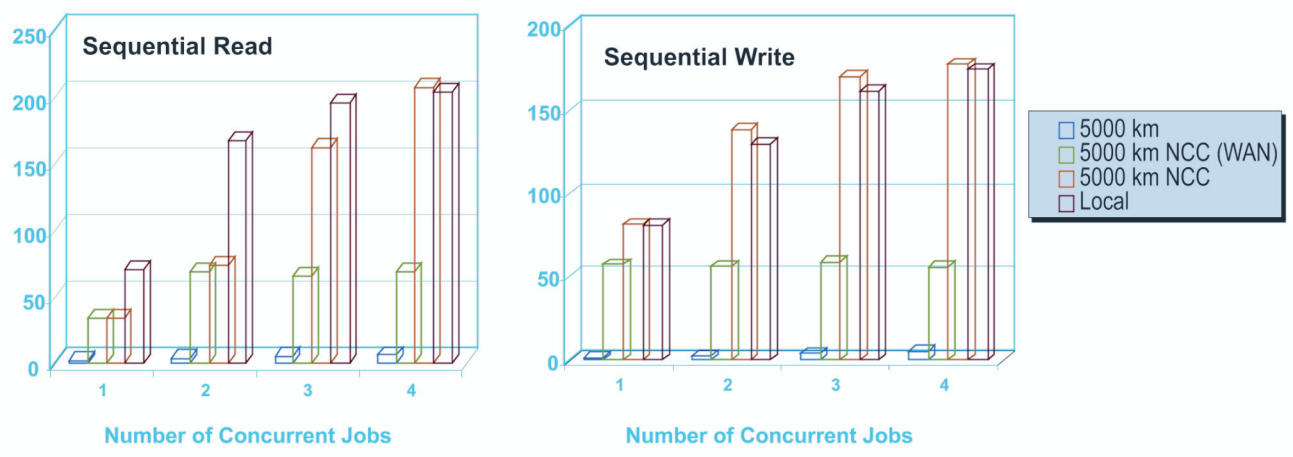

Fig. 10. Multi-job large file concurrent read and write performance to local storage, across 5000 km, and across 5000 km with the NS in the data path

By contrast, when the NS is in the data path, remote throughput is comparable to local. The root cause for the dramatic improvement is the presence of the NS distributed cache pool. The NetStorage Control Blade in each site was configured with 4 GB of local cache (the minimum configuration). Meanwhile, the total workload prescribed for this experiment is only 4 × 500 MB = 2 GB, and the amount of meta-data traffic required by the NS to maintain cache coherence is negligible (~10 kbps). Consequently, the prescribed workload can be served entirely from the cache of the single NetStorage Control Blade deployed in each site, and remote performance closely approximates local performance.

3.2.2. Single Cluster, Large File Performance

To study performance for workloads that benefit less from distributed cache, we increase the amount of data accessed by each job. Specifically, we use LSF MultiCluster to accept a new job every 40 seconds where each job performs a sequential read or write of a 2 GB file. Now, the total data accessed by four jobs is double the cache resources of the NetStorage Control Blade in each site.

Figure 10 shows the results for this test. The red column shows local performance, while the blue column shows performance over 5000 km. Again, remote performance is a small fraction of local performance due to the effects of transport latency captured in equation (1). The orange column shows performance over 5000 km when the NS is in the data path. Local NS cache serves a portion of this I/O and transfers the remainder (shown in the green column) across the WAN. Note that even for this workload, the local NS cache serves much of the I/O. This localization of I/O reduces by 50-75% the traffic that must go across the WAN. Also note that when the NetStorager System is in the datapath, WAN throughput improves by more than a factor of ten. The reasons for this performance enhancement are discussed above in Section 2.1.

3.2.3. Two-Cluster Performance

To study performance for workloads that involve distributed processing of centralized data, we use LSF MultiCluster to accept a new job every 40 seconds, with the first four jobs allocated to the local cluster (*i.e.*, the cluster co-located with the storage) and the next four jobs allocated to the remote cluster (*i.e.*, 5000 km from the storage). Each job initiates a sustained sequential read or write of a 500 MB file. Figure 11 shows the results of this experiment with and without the NS in the data path.

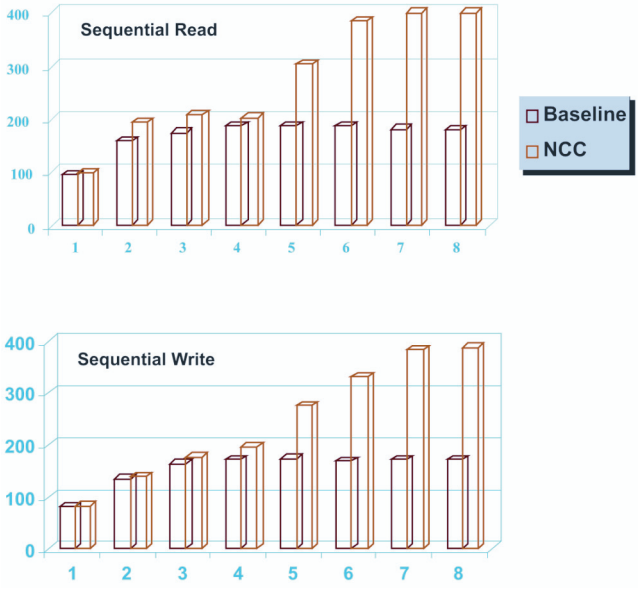

Fig. 11. Aggregate read and write performance for 2-cluster distributed processing with and without the NS in the data path

Note that without the NS in the data path (red columns), the first four jobs saturate the available 200 MB/s of bandwidth of the local cluster, but the next four jobs, which are allocated to the remote cluster, do not increase aggregate I/O throughput. In fact, aggregate performance actually degrades somewhat when jobs are allocated to the remote cluster. The reason is that the overhead for the clustered file system to maintain filelevel locking and coherence across the two clusters is greater than the benefit derived from the processing resources of the remote cluster. For such cases, attempts to leverage remote processing resources are actually counter-productive.

On the other hand, when the NS is in the data path (orange columns), the remote cluster achieves I/O throughput comparable to the local cluster and the aggregate I/O for the eight jobs is roughly double that for four. Each cluster saturates its local I/O bandwidth of ~200 MB/s and the aggregate throughput approaches ~400 MB/s.

3.2.4. Three-Site Performance

To study scalability of distributed processing performance when the NS is in the data path, we use LSF MultiCluster to load-balance jobs across the three sites according to the number of jobs executing at each site. In particular, LSF MultiCluster performs the load balancing with no awareness of data locality. We choose a workload so that the first eight jobs initiate only sustained sequential read operations, while the next eight jobs initiate sequential write operations. Figure 12 shows the results of the experiment.

Note the near-linear increase in aggregate I/O as additional jobs are scheduled until the limit of the bandwidth to

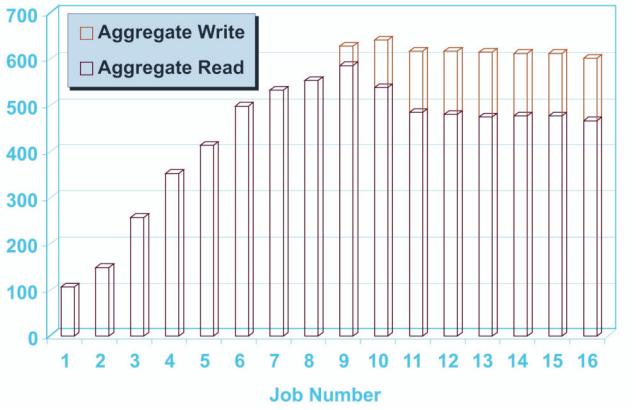

Fig. 12. Aggregate throughput as three sites initiate concurrent I/O to a single shared file system with the NS in the data path. LSF MultiCluster load balances all jobs with no awareness of data locality. The first eight jobs are read-only; the next eight involve concurrent reads and writes

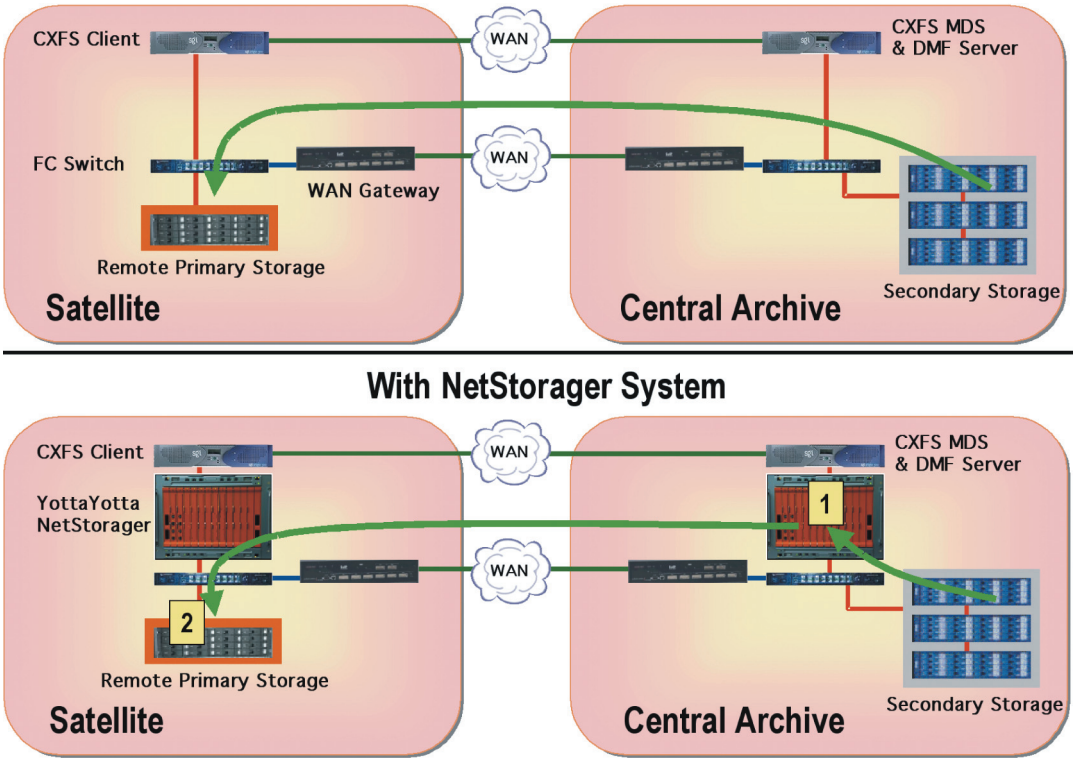

Fig. 13. System configuration for testing central archiving with high-speed file restore

storage from all three sites (*i.e.*, ~600 MB/s) is reached. As additional jobs are scheduled, aggregate read/write performance remains close to the theoretical ~600 MB/s limit.

3.3. Centralized Archiving and Geo-HSM

By complementing SGI's Data Migration Facility (DMF) with the NS, the SGI/YottaYotta data grid allows storage administrators to set policies to govern automatic migration of data both between storage sites as well as storage media. This way, only the most frequently used portion of a data image is physically resident at a given location and yet users can still address the entire data image at the central archive as though it were local. Files migrate to and from a central archive depending on frequency of use at a given branch office. This allows storage administrators to "thin provision" storage at branch offices while centralizing backup and HSM operations in a fashion that is completely transparent to end users. This can yield considerable cost savings on storage while ensuring high service levels.

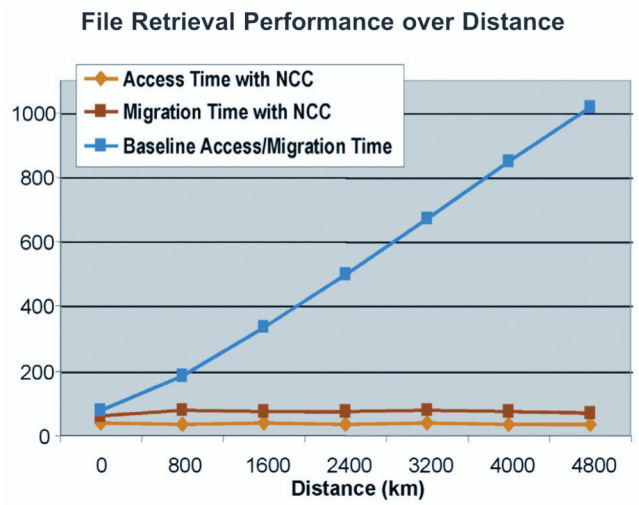

Fig. 14. Times to access and restore a file from a central archive to primary storage at a satellite location. File Retrieval Performance over Distance

Figure 13 shows the configuration for performance testing of the Geo-HSM and centralized archive capabilities of the SGI/ YottaYotta data grid. In such a configuration, two measurements are relevant to end-users. First, is the time until users at a satellite location can access a file that has been archived at the central site. This time is determined by the time required to restore the file from secondary storage to the local cache of the NS at the central archive (see Fig. 13). Second, is the time before this file is fully restored to primary storage at the satellite location. This time is determined by available WAN bandwidth and the NS's ability to perform high-speed transfer between sites.

Figure 14 shows times to access and fully migrate ten 2 GB files from secondary storage (*e.g.*, the central archive) to primary storage over various distances. Note that without the NS in the data path, the time to restore the files increases linearly with distance. By contrast, when the NS is in the data path, the time required to access and fully restore the files remain fairly constant even out to 5000 km.

4. CONCLUSION

We have presented a new architecture for the data grid that leverages both SGI and YottaYotta products to deliver a suite of data sharing, transfer, and protection services over wide geographic regions. We have evaluated this architecture for three applications: WAN data exchange, distributed data processing, and centralized archiving. Performance benchmarks reveal that for many applications, the SGI/YottaYotta data grid can deliver access to a globally consistent data image at near-local performance.