

DataGrid Authentication via GSI Certificates Within a Very Large Global File System

The 2006 International Conference on Grid Computing & Applications

Phil Andrews¹, Christopher Jordan¹, Patricia Kovatch¹,

¹ San Diego Supercomputer Center, University of California, San Diego,

La Jolla, Ca 92014, USA

Tel: 858-534-5000, Fax: 858-534-5117

{andrews, ctjordan, pkovatch@sdsc.edu, <http://www.sdsc.edu>}

Abstract. Grid computing is moving from the domain of the unrealized research project to the heavily production oriented world of the National Science Foundation's TeraGrid and other organizations. A critical part of such efforts is the ability of users to access their data quickly, efficiently, and independently of physical location. Whereas in small Grid communities it may be feasible to transfer the applicable data from place to place by file transfer utilities such as GridFTP, in large supercomputing communities the Global File System approach seems much more appropriate. This paradigm has been implemented within the TeraGrid and has been in production since October, 2005. The use of a Global File System has necessitated a rethink of the authentication process: unlike a very tightly coupled grid, the UIDs are not unified across sites. To enable the concept of file ownership to survive journeys between sites, it has been necessary to implement a GSI certificate to UID mapping. This has been completed and is in production. The extension of this paradigm to Group Authentication is more complex, and the exact approach is under discussion. In this paper we report on our experiences and intended future work. **Keywords:** Grid, Data, File Systems, Authentication, Security

1. Introduction

At the beginning of the 21st century, Grid Computing[1] was beginning to make inroads on production systems. The general approach was to take the Globus software[2] developed originally within academic environments and move it wholesale to the target systems. The TeraGrid[3] was probably the first really extreme example, with a network backbone capable of 40 Gb/s, compute capability of well over 10 Teraflops, rotating storage of over a Petabyte, and archival capacity in the 10's of Petabytes spread over several sites. It is a testament to the robustness of the original design that much of the translation from relatively constrained academic environments to very large scale production systems was successful. There is one specific area, however, where the original paradigm invited modification for efficient computation in the supercomputing arena. The original mode of operation for Grid Computing was to submit the user's job to the ubiquitous grid, where it would run on the most appropriate computational platform available. Any data required for the computation would be moved to the chosen compute facility's local disk, and output data would be written to the same disk; being saved to the user's permanent storage facility at the end of the job. The normal utility used for the data transfer would be GridFTP[4].

There were several reasons why the normal approach of moving data back and forth did not translate well to a supercomputing grid, mostly relating to the very large size of the data sets used. For example, the National Virtual Observatory (NVO)[5] consists of approximately 50 Terabytes and is used as input by several applications. Some applications write very large amounts of data, e.g., the Southern California Earthquake Center (SCEC)[6] simulations may write close to 250 Terabytes in a single run. Other applications require extremely high I/O rates in order to read and write intermediate results: the Enzo[7] application requires multiple Terabytes per hour be routinely written and read. These sizes and required transfer rates are not conducive to routine migration of wholesale input and output data between grid sites.

The computational system chosen may not be able to guarantee enough room to receive a required dataset, or for the output data, while the necessary transfer rates may not be achievable. Resource discovery[8] can go a long way to help, but it may also prevent several sites from participating in the computational effort. In addition, in many cases the application may treat the very large dataset more as a database, not requiring anywhere near the full amount of data, but instead retrieving individual pieces of very large files. In the case of Enzo, for example, many sites work to interpret and visualize the output data, and moving large files to multiple sites may be both inefficient and restrictive on the participants.

The obvious solution was a Global File System: a file system that could be accessed from each of the TeraGrid sites if the appropriate hardware and software clients are available there. In this paper we shall briefly show how the GFS was initially prototyped and then implemented in a production manner using IBM's GPFS file system. Of course, the usage of a file system originally designed for use within a single machine room, where the concept of a single UID per user still holds, necessitated a rethink of the authentication process. Within the TeraGrid, there is no consistency of the UID mapping to actual users across different sites: a single user could easily have six or more completely different UIDs across the TeraGrid. However, the users still need to access "their" data from different sites, no matter what their distinct UIDs are.

In fact the only "sticky" identification of a user across multiple TeraGrid sites is their GSI certificate. It was thus decided to implement an authentication scheme for file system access with depended upon the users' GSI certificates rather than their specific UID. The mechanics of the implementation depended upon the provision of an Application Programming Interface to the GPFS authentication process by IBM. At SDSC we then developed a GSI certificate to UID mapping system that allowed for a more flexible authentication mechanism. With the individual GSI certificate to UID mapping in place and working satisfactorily, the problem of Group ID mapping was next to be faced. In this case, the problems are not necessarily wholly technical; the concept of Groups across multiple organizations is necessarily more complex than within a single locality.

2. Production Facility: 2005-2006

In 2005, the SDSC, TeraGrid, and NSF personnel agreed to proceed towards production infrastructure using a global file system approach. By this time, the size of datasets had become large enough that only massive installations really held promise for significant facilitation of grid supercomputing. In fact, the size of some datasets was already starting to stress the storage capabilities of several sites. The NVO dataset, for example, was proving particularly useful and multiple sites were committed to providing it to researchers on spinning disk.

At 50 Terabytes per location, this was a noticeable strain on storage resources and if a single, central, site could maintain the dataset this would be extremely helpful to all the sites who could access it in an efficient manner. Of course, updates, data integrity, backups, etc., could also be handled in a much more satisfactory way if copies were not spread around the TeraGrid. Given that NVO at 50 Terabytes was only one such dataset, it was clear that for significant impact, a very large amount of spinning disk (by early 2005 standards) would be required. The onset of Serial ATA (SATA) disk drives made this more of a financial possibility, and in late March, 2005, approximately 0.5 Petabytes of SATA disk was acquired by SDSC.

Experience shows that is essential to design a balanced configuration, particularly with this much disk. It was decided that a design point would be for an eventual maximum theoretical bandwidth within the SDSC machine room of 128 Gb/s. This number should be easily sufficient to fill the SDSC path to the TeraGrid backbone, and is an exact match to the maximum I/O rate of our IBM Blue Gene/L system, "Intimidata", which is also planned to use the GFS as it native file system. For the

Network Shared Disk (NSD) servers we used 64 two-way IBM IA64 systems with a single GbE interface and Fibre Channel 2 Gb/s Host Bus Adapter in each. The plan is to double both of these in the future: the GbE to increase the maximum aggregate throughput to 128 Gb/s and the HBA to provide a separate path for an archive interface to the disk storage. The disks themselves are 32 IBM FastT100 DS4100 RAID systems, with 67 250 GB drives in each. The total raw storage is thus $32 \times 67 \times 250 \text{ GB} = 536 \text{ TB}$. The RAID sets within the DS4100 are internally connected via two 2 Gb/s Fibre Channel arbitrated loops, with an independent controller for each loop. The drives within each RAID set are Serial ATA disks. The two DS4100 controllers each have a 2 Gb/s FC connection to the outside world so that each DS4100 system can connect to two NSD servers. The disk setup for a single DS4100 contains seven 8+P RAID sets with the remaining unused drives used as hot spares.

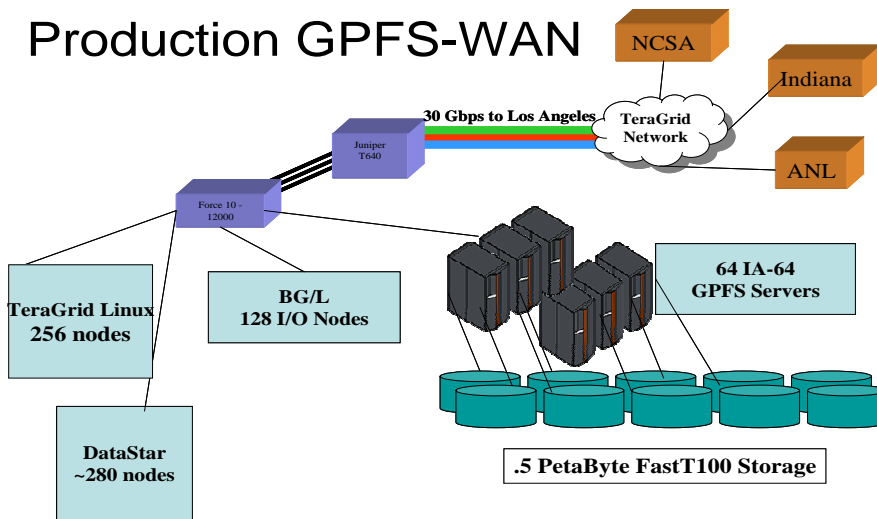


Figure 1. Initial GPFS-WAN file system deployment across TeraGrid

Although the connection between the DS4100 systems and the I/O servers is via Fibre Channel, the outside world normally accesses the file system data via the GbE connection to each I/O server. For maximum usability, the file system needs to be accessible from the systems in the SDSC machine room as well as across the TeraGrid Wide Area Network, and Fig. 1 shows how the 0.5 Petabyte of disk is accessed both (presently) by NCSA and Argonne National Laboratory, and by the major compute resources of SDSC. In October 2005, we began production use of the approximately 0.5 PB of GFS disk, mounting them in a production manner at several sites: all 256 nodes of the TeraGrid Cluster at SDSC, all 32 nodes at Argonne National Laboratory, and over 512 nodes at NCSA.

That this is a true supercomputing resource can be seen from the performance numbers displayed in Fig. 2. Local numbers for read performance can be over 6 GB/s, and network numbers are up to 2.5 GB/s, exemplary for a 30 Gb/s connection.

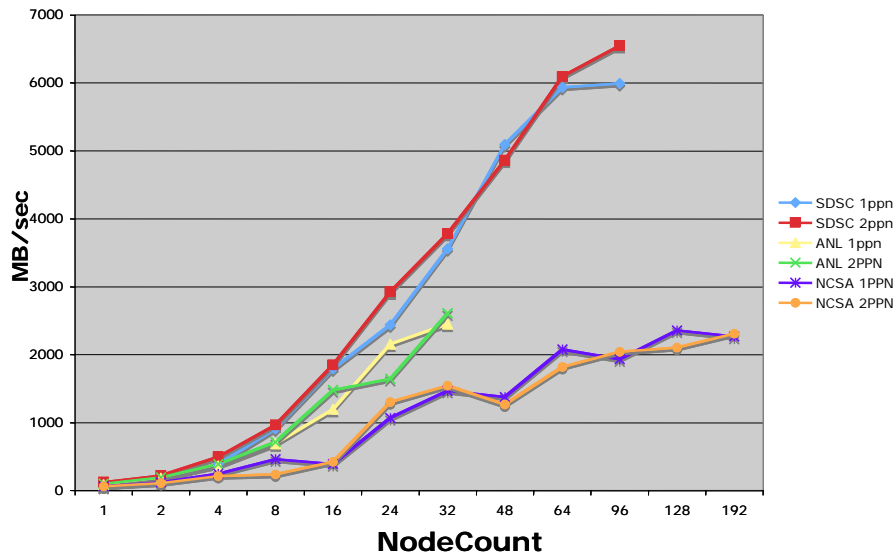


Figure 2. Read Performance scaling with number of remote nodes

3. Authentication

So far in this paper, we have concentrated on working with vendors in the development and early testing of external software, on the assumption that for true production quality performance and support, the development of new software should be minimized. There was one area, however, where the production of new capabilities was unavoidable for the adoption of this approach. That was in the field of authentication. The existing protocol for file system authentication is via the User ID (UID), and secondarily by Group ID (GID). These are normally uniform within an organization, so, e.g., every SDSC user has the same UID on each of the SDSC systems. This greatly simplifies access mechanisms, etc., and is the conventional approach for file systems. The difficulty enters when attempting to extend the concept of file “ownership” across a Grid. Except in very tightly coupled cases, it is very likely that a user will have different UID within the separate organizational entities making up a Grid. E.g., a user will, most likely, have different UIDs at SDSC, NCSA, ANL, etc., across the TeraGrid. However, he will certainly prefer to believe that any data he creates on a centralized Global File System belongs to “him” and not to one of his particular accounts, which probably has no logical relation to his account on any other Grid system. In response to this dilemma, we decided to develop an extended authentication mechanism based on the GSI[9] model, where a single certificate allows a user access to multiple computation platforms. Since we expected each of the TeraGrid users to have access to a GSI certificate, we decide to use this approach for data access, also. GPFS Multi-Cluster UID-mapping is accomplished in two stages, each of

which is performed by a single external application or script. In the first stage, the "mmuid2name" process is invoked to map a local UID to a Globally Unique Name (GUN). This establishes a global identity associated with a UID on the serving cluster. In the second, the "mmname2uid" process is invoked to map this global identity back to a UID on a client cluster. This general mechanism does not determine the actual underlying implementation, nor is it dependent on any particular standard for GUNs.

In the work done by SDSC and IBM, specific implementations of the "mmname2uid" and "mmuid2name" scripts were written to utilize the GSI grid-mapfile to obtain mappings between local UIDs and x.509 certificate Distinguished Names (DNs) - Globus identities. Since the Globus infrastructure already provides for these Distinguished Names to be unique, and requires that mappings exist at each site between the DN and the local UID, we were able to leverage this infrastructure to accomplish global mapping of user identities, and consequently file ownership, across the TeraGrid. Obviously, earlier work has explored this approach[10], but we describe our work in the following subsections.

3.1 GPFS 2.3 Cluster Control

A GPFS cluster is simply a set of nodes which share configuration and local filesystem information, and which can be controlled as a whole using the standard GPFS commands. In a High Performance Computing environment, it is typical for a GPFS cluster to be contiguous with an actual machine cluster, which is treated by other administration tools and scheduling systems as a cluster of nodes. Cluster configuration and filesystem consistency is maintained by a primary configuration server, backed by a secondary in case of node failure, which maintains master copies of all configuration files. The GPFS daemons maintain constant inter-node communication to verify filesystem consistency and communicate metadata information. In some cases, simple configuration queries can be serviced by GPFS daemon network requests to the primary configuration server, and filesystem state information can be gathered from the filesystem manager and other peer daemons; in these cases, the nodes basically act as a predefined peer-to-peer network, allowing connections and sharing information freely amongst themselves.

However, certain configuration changes and queries require the ability to change configuration data and get daemon state information, independent of whether daemons on each cluster node are actually functional. In these cases, the GPFS system must use external remote shell and remote copy commands, which are defined at the time of initial cluster creation. Many GPFS commands are essentially special-purpose versions of a distributed shell command such as dsh(CSM) or psh(XCAT), and there is also a distributed-shell tool called "mmdsh" which is supplied with the GPFS distribution. Any remote shell and copy commands may be used, but they must support rsh/rcp standard syntax, and they must be configured for passwordless authentication as the root user to all nodes in the GPFS cluster. Passwordless authentication is mandatory because the remote shell and copy commands are executed within the GPFS command or can be invoked by the filesystem daemons themselves, and because it is possible to have up to thousands of nodes in a single GPFS cluster. In modern Linux

cluster environments, ssh and scp are typically used as administration tools, with passwordless authentication configured through host-based RSA authentication and host equivalency files. In AIX/CSM environments, the use of rsh and rcp over private, non-routable networks is the default mode of passwordless root access. OpenSSH 3.2 and hostkey-based authentication is used in SDSC's TeraGrid IA64 environment.

It generally does not present an additional security issue to support passwordless SSH within a single cluster – system administrators must already have this kind of functionality in order to administer a large cluster, and users must have it in order to run parallel jobs.

3.2 GPFS 2.3 Multi-Cluster Authentication and Control

In multi-cluster configurations, a GPFS cluster may export its filesystems to other GPFS clusters, allowing nodes in these clusters to mount the filesystems as if they were local GPFS filesystems, using NSD servers in the remote cluster to achieve parallel filesystem performance over any routable TCP/IP network, including wide-area networks. However, in order to use a filesystem controlled by a remote cluster, nodes in the mounting cluster must be able to acquire and maintain the configuration data of the remote cluster, such as the list of primary and secondary NSD servers for a remote filesystem. They must also be authenticated to those servers so filesystem data is not publicly accessible, and there must be a mechanism to map UID and GID information across administrative domains, so that the filesystem doesn't have to rely on sites aligning their UID and GID information.

In the initial implementation of Multi-clustering in GPFS 2.3 development, the use of remote-shell commands was extended to distribute this information to all nodes of all clusters. Since multi-clustering support is particularly useful for sharing filesystems across administrative domains, the requirement of passwordless authentication as the root user was problematic from a security standpoint; many sites do not allow any kind of connection as root from external networks, and even in cases where it is technically allowed, it still represents an obvious security risk. Cluster administrators go to great lengths to isolate their machines such that a security compromise on any one cluster does not automatically translate into a compromise of machines external to the cluster. Multi-clustering is also useful for sharing filesystems across HPC platforms, where preferred remote shell commands may differ; in these cases special system configuration changes must be made to allow the same commands to be used on all nodes in all clusters. This issue is problematic from both an administrative and a security standpoint. In addition, the use of passwordless root access was the primary mechanism for authenticating a client cluster to a serving cluster, and filesystem network traffic was always unencrypted, creating the possibility of data interception through packet-sniffing. Based partly on the experience working with this initial implementation in the SDSC-IBM StorCloud Challenge for SuperComputing 2004, in which multiple SLES8 IA64 and AIX Power4+ clusters at SDSC, NCSA, and at the SuperComputing conference itself were used, the authentication and configuration distribution mechanisms were changed for the GA release of GPFS 2.3. In the new

implementation, there is no requirement for any remote shell access between clusters mounting a filesystem, and there are more control options for authentication and filesystem traffic encryption.

The GPFS 2.3 GA release uses RSA public/private keypairs to securely authenticate each cluster when multi-clustering is used. In this implementation, out-of-band transfer of RSA keys by system administrators is required to establish trust between two clusters, and specific nodes are designated for communicating configuration data when connecting clusters. A new command, "mmauth," is provided to manage the generation and importation of RSA keypairs, along with access control for external clusters and exported filesystems. A new configuration option, "cipherList," is used to require RSA authentication when connecting clusters, and also to enable encryption of all filesystem traffic if desired. For the cluster that will mount an external filesystem, two commands are provided to define and establish trust with external clusters and filesystems, "mmremoteccluster" and "mmremotefs." All of this functionality is supported through by invoking the open source OpenSSL library, thereby creating the possibility of security audits of significant portions of the actual authentication code, leveraging the fact that this library is heavily scrutinized for security flaws, and allowing for easy extension of the specific algorithms used for authentication and authorization. The process of establishing trust is as follows. First, the administrator of each cluster that will participate in a multicluster configuration creates an RSA keypair and enables the configuration option for RSA authentication or authentication plus filesystem traffic encryption. When creating keypairs and changing authentication options, all GPFS daemons in the cluster must be shutdown. The administrators of the clusters then exchange the RSA public key files via an out-of-band mechanism such as e-mail, preferably authenticated through the use of GPG or another signature mechanism. For the cluster that will export the filesystem, the administrator uses the mmauth command to add the remote cluster to the list of external clusters allowed to communicate with the cluster, and the administrator of the importing cluster uses the mmremoteccluster command to define the server cluster, including selecting a set of nodes which will be used for establishing authentication with the remote cluster, and the mmremotefs command to define the remote filesystem. A special device entry is added to each node in the importing cluster to represent the remote filesystem.

Once these steps are completed, the remote filesystem can be mounted using the standard UNIX mount command. When the mount command is issued, the GPFS daemons use the RSA keypairs to securely authenticate to one of the set of designated nodes on the serving cluster, which then distributes the information that the remote cluster has successfully authenticated to all other nodes in the cluster. All standard GPFS filesystem query commands can then be used on the client cluster, for example to list disk status in the remote filesystem, or filesystem configuration statistics.

In the PTF 3 update to GPFS 2.3, an additional per-filesystem access control capability has been added to the multi-cluster configuration options. Using the mmauth command, the administrator of the exporting cluster can control, for each importing cluster, which filesystems can be mounted by the cluster and whether the filesystem can be mounted in read-only or read-write mode. This provides for the sharing of data in situations where users at the remote site should not be given write

access to any part of the filesystem, and prevents the use of any mechanism to defeat UID/GID-level filesystem security. nIt is also important to note that the multi-cluster configuration as described above is not limited to simple client-server relationships between clusters, but is better characterized as a set of client-server connections between peers. A single cluster can export to many other clusters, and in testing at SDSC a single cluster has successfully exported one filesystem to clusters at Argonne National Labs, NCSA, and Purdue over the Teragrid network, as well as an additional GPFS cluster within SDSC. RSA public keys are shared between the various mounting clusters, as nodes in various clusters may need to communicate with each other to negotiate metadata operations. In addition, a cluster which imports a filesystem from another cluster could serve another local filesystem to the same cluster, so all GPFS filesystems within a multi-cluster environment could be shared across all clusters within that environment.

With the GSI certificate to UID mapping in place, we are now working on the more complex “group” mapping problem, and expect to have a solution implemented within the next two months.

4. Acknowledgments

We would like to thank Roger Haskin for coordinating the IBM support during SC’04. We would like to thank the SciNet organization for providing the significant bandwidth necessary to perform these demonstrations. The application groups for Enzo, SCEC, and NVO were also very helpful. A Global File System is of little interest without significant commitment from other sites, and we would like to thank Rob Pennington and his co-workers at NCSA and J.P. Navarro and his at ANL. We would also like to thank NSF personnel for their interest and support in this approach. This work was funded in part by the National Science Foundation.

References

- [1]The Grid: Blueprint for a New Computing Infrastructure by Ian Foster (Editor), Carl Kesselman (Editor)
- [2]Foster, I., Kesselman, C., Nick, J.M. and Tuecke, S. Grid Services for Distributed Systems Integration. IEEE Computer, 35 (6). 37-46. 2002
- [3]Catlett, C. The TeraGrid: A Primer, 2002. www.teragrid.org
- [4]W. Allcock..<http://www-fp.mcs.anl.gov/dsl/GridFTP-Protocol-RFC-Draft.pdf>.
- [5]A.S. Szalay, The National Virtual Observatory, in ASP Conf. Ser., Vol. 238, Astronomical Data Analysis Software and Systems X, 2001.
- [6]T. H. Jordan, C. Kesselman, et al., "The SCEC Community Modeling Environment—An Information Infrastructure for System-Level Earthquake Research. <http://www.scec.org/cme>
- [7]Enzo – AMR Cosmological Simulation Code, cosmos.ucsd.edu/enzo
- [8]Nimrod/G: An Architecture for a Resource Management and Scheduling System in a Global Computational Grid, Rajkumar Buyya, David Abramson, Jonathan Giddy, The Fourth International Conference on High-Performance Computing in the Asia-Pacific Region-Volume 1 , 05 14 - 05, 2000 , Beijing, China
- [9]A Security Architecture for Computational Grids,I Foster, C Kesselman, G Tsudik, S Tuecke ,ACM Conference on Computer and Communications Security, 1998
- [10]Joseph J Tardo, K Alagappan - IEEE Symposium on Security and Privacy, 1991