CITI Technical Report 06-05

Extending NFSv4 for Petascale Data Management

Wm. A. (Andy) Adamson, Dean Hildebrand, Peter Honeyman,
Shawn McKee, and Jiaying Zhang
`{andros, dhildebz, honey, smckee, jiayingz}@umich.edu`

ABSTRACT

Designed with Internet data management in mind, NFSv4 is meeting the needs of widely distributed col-laborations. Anticipating terascale and petascale HPC demands, NFSv4 architects are designing pNFS, a standard extension that supports parallel access to cluster file systems, object stores, and massive SANs. In combination with advances in optical networking, pNFS decouples computing from storage, even for petascale computations.
CITI's GridNFS project integrates NFSv4 into the ecology of Grid middleware with critical virtual or-ganization management: Globus support, name space construction and management, fine-grained access control with foreign user support, and high perform-ance secure file system access for jobs scheduled in an indeterminate future. GridNFS and pNFS combine and integrate standard Internet protocols, promising near-universal compatibility with standards-compliant desktop and enterprise network services.

May 1, 2006

Center for Information Technology Integration
University of Michigan
535 W. William St., Suite 3100
Ann Arbor, MI 48103-4978

Extending NFSv4 for Petascale Data Management

Wm. A. (Andy) Adamson, Dean Hildebrand, Peter Honeyman, Shawn McKee, and Jiaying Zhang
University of Michigan, Ann Arbor

Abstract

Designed with Internet data management in mind, NFSv4 is meeting the needs of widely distributed collaborations. Anticipating terascale and petascale HPC demands, NFSv4 architects are designing pNFS, a standard extension that supports parallel access to cluster file systems, object stores, and massive SANs. In combination with advances in optical networking, pNFS decouples computing from storage, even for petascale computations.

CITI's GridNFS project integrates NFSv4 into the ecology of Grid middleware with critical virtual organization management: Globus support, name space construction and management, fine-grained access control with foreign user support, and high performance secure file system access for jobs scheduled in an indeterminate future. GridNFS and pNFS combine and integrate standard Internet protocols, promising near-universal compatibility with standards-compliant desktop and enterprise network services.

1. Storage requirements for global collaborations

Scientific collaborations that span the globe allow geographically distant teams to share specialized instruments and analyze data collections on massive distributed parallel compute clusters. Emerging optical networks provide a scalable interconnect able to provide efficient massive data transfers on long range networks in the tens or hundreds of gigabits per second (Gbps). These clusters and the scale of data they produce and consume as well as the high speed network capabilities would have been nearly unimaginable only a decade ago. Grid technologies are defined and driven by the needs of science.

It is becoming common for teams of scientists to form *virtual organizations*: geographically distributed, functionally diverse groups linked by electronic communication, relying on lateral, dynamic relationships for coordination [1]. Within the Grid, the need for flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions, and resources presents unique authentication, authorization, resource access, resource discovery, and other challenges [2].

Dynamic virtual organizations sharing massive amounts of data create several new classes of problems unique to inter-institutional collaborations. The GridNFS project addresses three of these problems:

The dynamics of virtual organizations demand agile security mechanisms. Security mechanisms for virtual organizations must be strong enough to protect the integrity of data at all times and to protect the confidentiality of data when necessary. They must be adaptable, to accommodate the varying membership of virtual organizations. They must be able to delineate precise authorization limits for users from outside the virtual organization.

Global collaboration requires a canonical way to name shared data (e.g., file names). This need is driven by the vast amounts of data generated by modern collaborative physics, which must be accessible to a widely dispersed collaborative community.

Wide-area performance. Superior performance is critical for GridNFS to be accepted as a central middleware component. NFSv4 has the essential core to provide excellent performance, but the Grid's special network requirements—long fat pipes, parallel networks, and tiered distribution—demand close attention.

To address these problems, we are developing GridNFS, a middleware solution that melds distributed file system technology with flexible identity management techniques to meet the needs of Grid-based virtual organizations.

The foundation for data sharing in GridNFS is NFS version 4 [3], the IETF standard for distributed file systems that is designed for security, extensibility, performance, and scalability. GridNFS meets the challenges of authentication and authorization with X.509 credentials, which bridge NFSv4 and the Globus Security Infrastructure, allowing GSI identity to control access to files exported by GridNFS servers.

By tying together these middleware technologies, we fill the gap for two vital, missing capabilities: *transparent, scalable, and secure data management* integrated with existing Grid authentication and authorization tools, and *scalable and agile name space* *management* for establishing and controlling identity in virtual organizations and for specifying VO data resources.

In the remainder of this paper, we outline related work, and then detail the middleware technologies that provide the performance, transparency, security, and ease-of-use featured in the scenario.

2. Related work

Many technologies have stretched to meet the performance and scaling requirements of Grid data access.

AFS [8], used extensively in the physics community, features a global name space, secure access control, a rich back-end management system, and a skilled open-source development community. Yet AFS limitations make it unsuitable for many Grid applications: AFS does not meet the scale and performance requirements of cluster computers, relies on a UDP-based network library whose design target is decades old, and uses a coarse-grained security model that complicates access control across autonomous security domains and is not well suited for pre-scheduled access.

NFSv3 [9] is also used extensively on the Grid for data sharing between cluster nodes. Like AFS, NFSv3 was built on UDP, but now uses TCP for good performance across a wide range of network conditions. However, the NFSv3 security and state models preclude its use in untrusted or wide-area environments.

NFSv4, designed with the lessons of AFS and NFSv3 in mind, provides transparent, high-performance access to files and directories; replaces NFSv3's troublesome lock and mount protocols; mandates strong and diverse security mechanisms; and supports scalable and consistent client caching and internationalization.

GridFTP [10], used by many physics applications for transporting data on the Grid, is engineered with Grid applications in mind. GridFTP negotiates TCP options to fill the pipe, transfers data in parallel, uses Grid security, and allows resumption of partial transfers. As an application, GridFTP is easy to install and support on heterogeneous platforms. On the other hand, as an application, GridFTP cannot take advantage of kernel features like zero-copy access, range locks, name space, and consistent sharing.

SRB [11] is mature Grid middleware that solves many data access and management problems, especially life cycle and metadata management for heterogeneous data sets. However, SRB has a potential bottleneck in its single server architecture, and like GridFTP, lives outside the kernel and shares the advantages and disadvantages discussed above.

As we shall see, GridNFS works alongside GridFTP or SRB. In tiered projects such as ATLAS, GridFTP remains a natural choice for long-haul scheduled transfers among the upper tiers, while the file system semantics of GridNFS offers advantages in the lower tiers. Domain scientists can work directly with GridNFS files using conventional names. This promotes effective data management without obviating SRB's life-cycle strengths. GridNFS also offers transparent support for operating system extensions such as RDMA, file replication and migration, extended attributes, and parallel access.

3. Broader impacts

Concurrent with our development of GridNFS, NFSv4 is being deployed worldwide by Sun, Network Appliance, IBM, HP, Hummingbird, EMC, and other vendors. We expect that NFSv4 will silently displace NFSv3, just as NFSv3 displaced NFSv2. We also anticipate that advanced features and vendor support will lead most remaining AFS installations to switch to NFSv4, which stands ready to realize the unmet promises of DCE/DFS for enterprise computing. (To help this transition at the University of Michigan, we are developing tools that support migration of AFS volumes and AFS ACLs to NFSv4 servers and NFSv4 ACLs.)

NFSv4 and the Grid are simultaneously poised for exponential growth in influence. The increasing commercial influence of NFSv4 dovetails with the GridNFS project over its initial three-year span. Cluster computing in problem domains ranging from high-energy physics to entertainment rely increasingly on NFSv4 to tie massively parallel data engines to massively parallel compute servers. Parallel file systems such as Lustre [14], GPFS [15], GFS [16], PolyServe Matrix Server [17], and Panasas [18] use extensions of NFSv4 to position themselves as conventional, standards-compliant components, able to meet spectacular opportunities offered by the advent of petascale computing while continuing to satisfy the needs of enterprise desktops.

4. The GridNFS approach

GridNFS combines Grid and NMI [4] infrastructures with NFSv4 to solve these challenges. Here is a scenario that shows how these technologies interact.

A scientist at an enterprise desktop or laptop uses a Grid client to schedule use of Grid resources. She identifies input and output data objects by global names; in fact, they are the same names that she uses *on her desktop computer to identify the data resources, which lets her provide appropriate access controls over the resources.*

Once the reservation is completely described, a scheduler determines when and where the job will be run and stores appropriate proxy credentials with the scheduled job. Data sets can be pre-staged through a tiered system of data access and automatically replicated onto secure servers in the local neighborhood of the compute engines to be used, or high-speed networks are reserved, enabling parallel WAN access to remote data sets.

At last, the job is ready to be executed. Proxy credentials that reflect the authorizations of the requesting scientist are provided to data and computing resources and used directly to authorize access to the scientist's files.

Finally, the task is complete. Replication nodes for input data are automatically removed from service, while output data is distributed through replication and through conventional tiered mechanisms. The results can be viewed directly by the scientist, whether she is sitting in front of a highly customized visualization workstation in her laboratory or running a conventional application on a commodity laptop while sipping coffee halfway across the world.

5. GridNFS name space for data

NFSv4 uses the familiar hierarchical style of naming common to modern file systems. However, NFSv4 has two useful features that assist in the engineering of name spaces for virtual organizations.

The first feature is the NFSv4 server pseudo-file system. The server presents clients with a *root file handle* that represents the logical root of the file system tree provided by the server. The server constructs a logical image of the file system it wants clients to see by gluing physical file systems under its control under this logical root. Any gaps between the logical root and the physical file systems are filled with pseudo-directories. Clients then mount the logical root constructed by the server.

Because the server constructs the view seen by NFSv4 clients, users and applications in a virtual organization have a common name space relative to that server. To provide a common, organization wide name space, the problem remains to knit the server file systems into a common name space for a virtual organization.

The second NFSv4 feature that we use for constructing a global name space is the FS_LOCATIONS attribute. This lets a server redirect client access requests to another server. When a client encounters a pseudo-node, it retrieves the attribute, whose value is a list of {server, path} pairs. The client picks one from the list and continues with its request at the selected server and path.

Redirection provides essential flexibility in name space construction, allowing the administrators of a virtual organization to dictate the form and shape of that space, especially in the part of the name space close to the root.

One more feature needed for GridNFS name space construction is consensus on the root of the name space. Related matters are under discussion by IETF working groups. We anticipate a solution that builds on the emerging standard that uses SRV records for locating servers [5]; initially, NFSv4 clients simply mount the pseudo-file system root of a virtual organization's GridNFS server under a directory named `/GRIDNFS`.

With these features in place, researchers in the VO can discuss data sets with names like `/GRIDNFS/VO-PROJECT/HOTDATA/2006/11 /24/FILE24` and, with appropriate data management policies, can expect that path name to yield identical results throughout the virtual organization. This simplifies administration of a GridNFS client—it is configured once (and only once) to mount the virtual organization's file hierarchy at `/GRIDNFS`.

Administration of the root of the virtual organization is also easy — as data servers come and go, redirection points are added to or removed from the name hierarchy rooted at `VO-PROJECT/`. All clients immediately see the change.

Finally, because the redirection points refer to data servers in autonomous domains under the control of members of the virtual organization, delegation of policy and control to the members of the virtual organization is consistent with their responsibilities and investment.

6. GridNFS name space for users

GridNFS is able to marry NFSv4 with GSI because both support X.509 distinguished names as a form of user credential: users are identified by names that are bound to credentials. To instantiate a virtual organization in the Grid, member organizations establish a certificate chain by exchanging X.509 certificates or by agreeing on a certification authority. This lets scientists in member organizations use their existing, locally generated credentials for access to resources across organizational boundaries.

The NFSv4 protocol specifies a Unicode string representation for names in access control calls, e.g., **`getacl`** and **`setacl`**, but NFSv4 clients and server platforms represent users with locally controlled numeric identifiers. To map between these names and numbers consistently, CITI's GridNFS prototype stores the maps in LDAP, but many systems use `nsswitch` to configure choices of mapping techniques. The IETF NFSv4 working group is examining extensions to the NSS name-to-ID mapping standard to support ACLs for foreign users.

GSI uses a proxy X.509 certificate for the user's DN. The Globus Gatekeeper uses a local flat file called `gridmap` to map the DN to a local name. Recent versions also support an ad hoc callout interface, which can be used to call maps stored in LDAP, for example.

The NFSv4 protocol mandates the Simple Public Key Mechanism version 3 [6], an X.509-based security mechanism that establishes a secure channel between client and server. Like SSL services in the web, an SPKM-3 uses X.509 credentials but clients may be anonymous or may mutually authenticate with X.509, i.e., GSI, certificates.

7. Wide-area performance

The emerging (first) minor version of the NFSv4 protocol — NFSv4.1 — will address the bottleneck that arises when multiple clients compete for the attention of a single server. The parallel NFS extension to NFSv4 — pNFS — provides for parallel, direct access to storage by separating data and metadata paths. NFSv4 servers continue to process metadata operations, but I/O operations go directly to storage. pNFS metadata servers are NFSv4.1 servers, guaranteeing a consistent view of the name space and strict adherence to NFSv4 semantics. Data servers can be NFSv4 servers, other distributed file systems, object storage, or even block-addressable disks in a SAN. Furthermore, pNFS allows data to be spread across multiple servers, eliminating the "single server bottleneck" so vexing to client/server-based systems. In combination, the elimination of the single server bottleneck and the ability for clients to access data directly from storage results in superior file access performance and scalability [12].

pNFS, designed for the petascale cluster and parallel file system environment, empowers GridNFS to unite enterprise and petabyte scale computing with a single file system protocol.

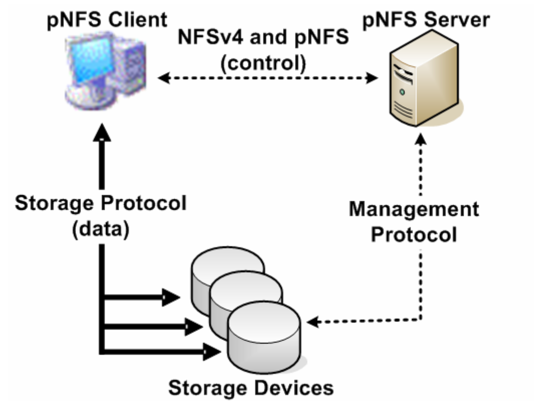

Figure 1. pNFS architecture
pNFS splits the NFSv4 protocol into a control path and a data path. The NFSv4.1 protocol exists along the control path. A storage protocol along the data path provides direct and parallel data access. A management protocol binds metadata servers with storage devices.

Figure 1 displays the general pNFS architecture. The control path contains all NFSv4.1 operations and features. The data path storage protocol is pluggable to allow direct access to a variety of storage devices. In addition to supporting object and block storage protocols, pNFS supports the use of the NFSv4 file protocol as a storage protocol, allowing easy implementation of parallel data access. NFSv4.1 does not specify a management protocol as it may be proprietary to the exported file system

Clients perform direct and parallel I/O by first requesting data location (layout) information from the pNFS server. Clients use the layout information in conjunction with the storage protocol to retrieve data. For example, the NFSv4 storage protocol stripes files across NFSv4 data servers (storage devices); only READ, WRITE, and COMMIT operations are used on the data path.

By selecting backend parameters such as the number of storage devices and stripe size, the pNFS architecture allows a match between the pNFS backend throughput and available network bandwidth. Figure 1 shows a single pNFS client accessing data in parallel on three storage devices. If each storage device can deliver a Gbps or so of data and the pNFS client is on a 10 Gbps network, then the client might approach full network utilization by striping a file across 10 storage servers.

Petascale computing requires inter-site data transfers involving clusters that may have different operating systems and hardware platforms, incompatible or proprietary file systems, or different storage and performance parameters that require differing data layouts. pNFS offers a solution.

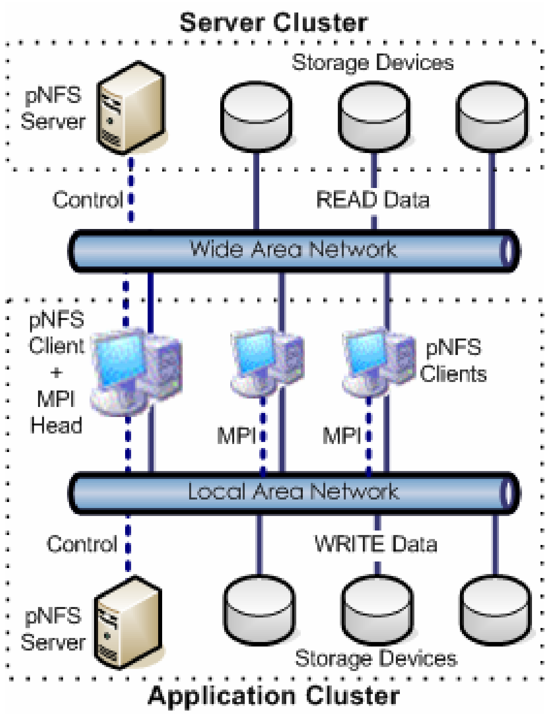

Figure 2. pNFS and inter-cluster data transfers across the WAN

A pNFS cluster retrieves data from a remote storage system, processes the data, and writes to its local storage system. The MPI head node distributes layout information to pNFS clients.

Figure 2 shows two clusters separated by a long range, high-speed WAN. Each cluster has the architecture described in Figure 1. (The management protocol is not shown.)

The *application cluster* is running an MPI [19] application that wants to read a large amount of data from the server cluster and perhaps write to its backend. The MPI head node obtains the data location from the server cluster and distributes portions of the data location information (via MPI) to other application cluster nodes, enabling direct access to server cluster storage devices. The MPI application then reads data in parallel from the server cluster across the WAN, processes the data, and directs output to the application cluster backend.

A natural use case for this architecture is a visualization application processing the results of a scientific MPI code run on the server cluster. Another use case is an MPI application making a local copy of data from the server cluster on the application cluster.

Note that each cluster can use a different storage protocol as long as the pNFS client implements the appropriate pluggable storage protocol.

pNFS not only bridges the gap between proprietary cluster file systems, it also opens cluster file systems to data access from enterprise desktop distributed file systems using common security, file naming, and file ACLs as the basis for data management.

8. Automated replication

Availability and performance are vital for Grid middleware. Replication is the basis of any effective scheme for availability, and also benefits performance and scalability. Replication plays an important role in Grid computing by allowing data to be pre-staged to nearby compute engines.

Grid applications often need access to enormous (read-only) data sets, so they use GridFTP to stage data on cluster computers in advance, then access the data at furious rates when it is needed. Often, much of the pre-staging can be automated, but conflicts arise with security and disk management facilities on the cluster node. In addition, absent protocols for keeping replicas consistent, the pre-staged data may become obsolete without warning if the master copy is updated.

Automated data replication offers a tantalizing alternative to manual or semi-automated pre-staging. CITI's replication and migration prototype for NFSv4 [7] offers per-directory granularity, POSIX-compatible read and write access, and optimal performance for read-only files. We are extending that work to mesh well with Grid security, to automate the creation and destruction of replication sites, and to pre-stage data securely on those sites. With this development in hand, we can pre-stage data to GridNFS servers in a tight neighborhood surrounding a compute server.

A "diamond-shaped" storage profile is characteristic among scientific applications [13]. That is, small inputs are expanded by early stages into large intermediate results, which are often reduced by later stages to small results. Replication can also be used to distribute the results of scientific computations across the Internet, or to protect intermediate results obtained at considerable computational cost from possible loss due to failure in the storage system.

9. Conclusion

To meet the needs of dynamic distributed collaborations, GridNFS combines mature and emerging standards to integrate storage management with Grid protocols seamlessly and universally. Ongoing work to mesh NFSv4 with parallel storage technologies catapults GridNFS capabilities and directly addresses some of the most critical storage requirements of dynamic virtual organizations: seamless, end-to-end access control; agile mechanisms for naming, placing, migrating, and replicating data sets at nearly any granularity; and location-independent parallel access to massive storage elements.

Acknowledgments

This work was partially supported by the NSF Middleware Initiative Grant No. SCI-0438298, by ASC under contract B523296, and by grants from Network Appliance and IBM.

References

[1] G. DeSanctis and P. Monge, "Communication Processes for Virtual Organizations," *J. Computer-Mediated Comm.*, 1998.

[2] I. Foster, C. Kesselman, and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," *Int. J. of HPC Apps.*, 2001.

[3] S. Shepler, B. Callaghan, D. Robinson, R. Thurlow, C. Beame, M. Eisler, and D. Noveck, "Network File System (NFS) version 4 Protocol," *RFC 3530*, 2003.

[4] NSF Middleware Initiative, www.nsf-middleware.org/.

[5] A. Gulbrandsen, P. Vixie, and L. Esibov, "A DNS RR for specifying the location of services," *RFC 2782*, 2000.

[6] M. Eisler, "LIPKEY—A Low Infrastructure Public Key Mechanism Using SPKM," *RFC 2847*, 2000.

[7] J. Zhang and P. Honeyman, "Naming, Migration, and Replication for NFSv4," to appear in *Proc. 5th Intl. Conf. on System Administration and Network Engineering*, Delft (May 2006).

[8] M. Satyanarayanan, J.H. Howard, D.A. Nichols, R.N. Sidebotham, A.Z. Spector, and M.J. West, "The ITC Distributed File System: Principles and Design," *SOSP*, 1985.

[9] B. Callaghan, B. Pawlowski, and P. Staubach, "NFS Version 3 Protocol Specification," *RFC 1813*, 1995.

[10] Globus Project, "GridFTP: Universal Data Transfer for the Grid," White Paper, 2000.

[11] C. Baru, R. Moore, A. Rajasekar, and M. Wan, "The SDSC storage resource broker," CASCON, 1998.

[12] D. Hildebrand and P. Honeyman, "Exporting Storage Systems in a Scalable Manner with pNFS," in *Proc. 22nd IEEE – 13th NASA Goddard Conf. on Mass Storage Systems and Technologies*, Monterey (April 2005).

[13] D. Thain, J. Bent, A. Arpaci-Dusseau, R. Arpaci-Dusseau, and M. Livny, "Pipeline and batch sharing in grid workloads", in *Proc. IEEE HPDC* (Jun. 2003)

[14] Cluster File Systems Inc., "Lustre: A Scalable, High-Performance File System," 2002.

[15] F. Schmuck and R. Haskin, "GPFS: A Shared-Disk File System for Large Computing Clusters," in *Proc. of the USENIX Conference on File and Storage Technologies*, 2002.

[16] Red Hat Software Inc., "Red Hat Global File System," `www.redhat.com/ software/rha/gfs`.

[17] Polyserve Inc., "Matrix Server Architecture," `www.polyserve.com`.

[18] Panasas Inc., "Panasas ActiveScale File System," `www.panasas.com`.

[19] W. Gropp, S. Huss-Lederman, A. Lumsdaine, E. Lusk, B. Nitzberg, W. Saphir, and M. Snir, *MPI: The Complete Reference, volume 2--The MPI-2 Extensions*. Cambridge, MA, 1998.