# USING AN INTEGRATED RULE-ORIENTED DATA SYSTEM (iRODS) WITH ISILON SCALE OUT NAS

## Guidelines, best practices and functionality when using an Isilon cluster with iRODS


### ABSTRACT

This white paper provides guidelines, best practices and descriptions of enhanced functionality when using an EMC Isilon cluster with iRODS.

June, 2014


EMC WHITE PAPER

EMC²

Part Number H13232

## TABLE OF CONTENTS


EXECUTIVE SUMMARY 4

AUDIENCE 4

ARCHITECTURE 4

AN OVERVIEW OF IRODS 4

OVERVIEW OF ISILON NETWORK ATTACHED STORAGE (NAS) 4

BEST PRACTICES FOR IRODS AND ISILON 5

Initial iRODS install 5

Isilon NFS best practices 9

CONCLUSION 11

REFERENCES: 11

# EXECUTIVE SUMMARY

iRODS is a data management system that has been widely adopted for use with unstructured file data, especially data used in various fields of scientific research such as bioinformatics, climate science, environmental studies, physics and astronomy. In a number of these fields, EMC Isilon clustered NAS is used to a significant degree as the primary data storage platform. This paper explores some of the best ways for using iRODS with Isilon clustered NAS to enhance and extend the overall functionality of the entire solution.

# AUDIENCE

This whitepaper is intended for administrators in institutions that utilize iRODS and Isilon storage to manage large repositories of unstructured file data.

# ARCHITECTURE

The broad description of the architecture referenced in this paper assumes an Isilon cluster as a significant resource for an iRODS zone. The primary data access method for clients is through an iRODS data server, although automatic data placement and a significant amount of bulk transfers could take place out of band from iRODS and be sync'd at a later point. A single iRODS zone will have at least one iRODS server and at least one (primary) instance of an iCAT database to support the zone.

# AN OVERVIEW OF IRODS

iRODS—or the integrated Rule-Oriented Data System—has become a powerful, widely deployed system for managing significant amounts of data that requires extendable metadata. Typical file systems provide only limited functionality for organizing data and few or no options for adding to the metadata associated with the files retained. Additionally, file systems are unable to relate or structure what limited metadata is available and provide only a platform from which to serve unstructured file data. Within several fields, especially scientific research where evolving instrumentation capabilities have vastly expanded the amount and density of unstructured file data, standard file systems can prove to be a limiting factor in the overall use of data.

iRODS provides a complete system for managing unstructured data and adding metadata to all/any files under its control. In addition, structures and relationships that exist solely within the metadata can be manipulated, enhanced and utilized. It has proven to be a powerful system for institutions with petabytes of raw data that require significant organization and analysis. Using iRODS, data can be tracked, cataloged, operated upon in a limited fashion and moved either singly or grouped with its related data, all without the end user interacting with the complexities of the underlying file systems.

# OVERVIEW OF ISILON NETWORK ATTACHED STORAGE (NAS)

Isilon Network Attached Storage (NAS) provides a flexible way to provide widely accessible storage to a large number of servers and users. Through utilization of common network protocols (CIFS/SMB, NFS, HTTP, HDFS) storage can be accessed from any number of machines by any number of users leveraging existing authentication services.

Isilon scale-out NAS provides the performance and capacity that scales linearly and expands to accommodate up to 21 PB of usable space in a single cluster. It utilizes EMC Isilon OneFS®, a high performance clustered file system that presents a global namespace for the entire cluster along with the strongest data protection available in the industry. In addition, Isilon has created many additional services that can run along with OneFS to provide additional functionality. Such as:

- **SmartConnect** provides the ability to effectively balance connections to any number of nodes in an Isilon cluster. It also provides resilient failover support by moving active connections to the cluster to functioning nodes.
- **SmartPools** provides rule-based movement of data through tiers within an Isilon cluster. This functionality is complementary to iRODS and can be used in conjunction with it to make the most efficient use of Isilon storage. Institutions are able to setup rules keeping the higher performing X and S series nodes available for the immediate access to data for computational needs and their NL series used for all other data. This is done while keeping data within the same global namespace, which can be especially useful in a large shared research environment.
- **SmartFail and Auto Balance** ensure that data is protected across the entire cluster. There is no data loss in the event of any failure and no rebuild time necessary. This contrasts favorably with other file systems such Lustre or GPFS as they have significant rebuild times and procedures in the event of failure—with no guarantee of 100% data recovery.

- **SmartQuotas** help control and limit data growth. Evolving data acquisition and analysis modalities coupled with significant movement and turnover of users can lead to significant consumption of space. Institutions without a comprehensive data management plan or practice can rely on SmartQuotas to better manage growth.
- **HDFS**. The advent of frameworks such as Hadoop for the analysis of Big Data has demanded newer file systems to better serve their needs. Isilon OneFS provides native support for HDFS as a protocol. An Isilon cluster with HDFS enabled can replace the direct attached storage (DAS) used by data nodes in a Hadoop cluster, providing a central location for 'in place' analysis of data and eliminate the need for the costly replication.

In addition to the software features available on Isilon, the hardware has such features as:

- **Dual 10GbE interfaces per node.** A cluster of Isilon nodes has enough bandwidth to accommodate instrument traffic as well as HPC analysis and significant data transfers to other tiers of storage or collaborative efforts.
- **SATA For NL And X Nodes**. Using lower cost, high density and reliable SATA drives the overall cost of these nodes is considerably less than equivalent offerings.
- **L1/L2 Cache In Node RAM.** OneFS will adaptively pre-fetch files into the cluster coherent cache, which is space reserved within the system RAM. Increased RAM provides increased cache space and will significantly enhance the performance of applications reading and writing to the cluster.
- **SSD & Global Namespace Acceleration (GNA)**. SSDs used with OneFS GNA can cache file system metadata and dramatically speed up the traversal and sorting of these large directory trees.

# BEST PRACTICES FOR IRODS AND ISILON

## INITIAL IRODS INSTALL

The initial installation of iRODS is a fairly uncomplicated process. There are four primary components to any iRODS setup:

- the iCAT database
- the iRODS server
- iRODS clients
- A storage resource

At least one of each of these elements needs to be present to form a functional iRODS zone. They should all be installed according to the iRODS installation instructions - however, some considerations when being used with Isilon NAS as an iRODS resource follow.

***iCAT***

The recommended location of the iCAT (iRODS metadata catalog) database (PostgresDB by default) is on its own server, separate from the iRODS server instance with at least 8GB of RAM – preferably direct on a machine or at least a VM with direct I/O enabled. The data files for the database should be kept on local direct attached (DAS) storage whenever possible and should not be installed on Isilon. Isilon is optimized for use with unstructured file data, and therefore Isilon is currently not recommended for block data applications such as standard RDBMS databases. Future development for iRODS with Isilon will seek to address this requirement and provide a place for the iCAT DB to reside within the Isilon directly. With these considerations in mind - install the iCAT DB in accordance with the iRODS setup instructions.

***iRODS server***

The initial install of an iRODS server should take place on a machine with at least 4GB of RAM. With systems that span a significant number of resources (storage platforms connected to iRODS and managed as part of a larger repository) or data (file count and/or size) - more memory and or more iRODS servers should be added. It is recommended for performance and scale reasons that the iRODS server be installed on a server independent of iCAT - either directly or using a VM. When configuring iRODS for this type of operation, the server must be built with UnixODBC support so that it can connect to the remote iCAT instance.

There is no change to the number of iRODS data servers recommended when using an Isilon resource. The number deployed will still depend on the requirements of each iRODS Zone and should not be constrained in any way by an Isilon cluster. Even Isilon's lowest performance tier can easily service hundreds of connections via NFS or SMB while maintaining high IOPS.

***Security, Isilon and iRODS***

Isilon NAS is able to use a number of different methods to authenticate users, including LDAP/Kerberos and others (AD, NIS+, local, etc.). For use with iRODS it's recommended that administrators setup LDAP authentication for both Isilon and iRODS. As of version 3.2, iRODS can use Pluggable Authentication Modules (PAM) which supports a plugin for LDAP. To use, set the 'irodsAuthScheme' to 'PAM' in .irodsEnv. It is recommended that both the Isilon cluster and the iRODS installation be set to use the same LDAP server and that iRODS be configured to use the well tested LDAP plugin with PAM. When this is setup - user passwords can be managed by the LDAP server and not from within iRODS itself. The users name (username) will still have to be setup on iRODS despite the same user information existing in LDAP (this is for iRODS own user/group mapping and management). Communication between iRODS servers cannot use LDAP and the username/password setup with 'iinit' will still be required.

Setting up iRODS in this manner will allow for the use of the Isilon cluster either/both as a Direct Access Resource or in the standard resource configuration. A Direct Access Resource in iRODS provides a way for files stored on the resource to be accessible by way of both the local file system and iRODS. This setup is primarily useful when having to access very large files without having to move them and still have iRODS provide meta-data annotation. An example scenario would be to provide direct access from a HPC or Hadoop environment. When using a Direct Access Resource it is critical to maintain a consistent mapping of user names and permissions between iRODS, which will assume the identity of external Unix users and apply ownership and permissions to files as such, and whatever system is accessing the files directly. The best approach is the use of a common external reference such as LDAP.

An Isilon cluster simply needs to be directed to the correct LDAP server. This can be done through the web UI by navigating to 'Cluster Management' -> 'Access Management' -> 'LDAP' from which point a new LDAP server can be setup and additional properties can be configured. From the command line all of the LDAP configuration can be modified using 'isi auth ldap'.

***Isilon as a storage resource***

Once a functional iRODS+iCAT server environment is setup it is ready to have resources attached. At this point, the Isilon clustered NAS can be configured as a resource for use in iRODS. Isilon is best utilized by iRODS when NFS mounted on the server and therefore as the default 'unix file system' resource type. It is recommended that the iRODS server mount the Isilon using NFSv4 for it's more 'stateful' properties and greatly enhanced lock management functionality. NFSv4 support should be enabled on the Isilon first by checking the 'Enable NFSv4' box under….in the UI. Additionally, the Isilon NAS should always be mounted through the SmartConnect address, which will manage both load balancing across the cluster and provide support for failover in the unlikely event of node failure/unavailability.

In a uniform Isilon cluster, choose the iRODS resource class that best represents the data meant to be stored. All Isilon node types can serve adequately as either cache or archive resource classes - though care should be taken to choose the most cost effective combination. For example, S series nodes make poor archive nodes simply because of cost and wasted performance potential. But NL nodes and future offerings in this product line make for good archive resources.

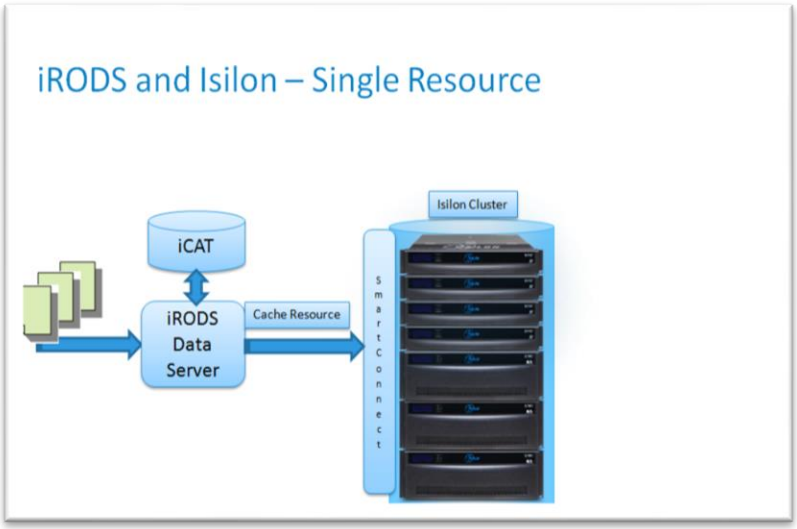


Figure 1. The Isilon clustered NAS can be configured as a resource for use in iRODS

***iRODS resources with SmartPools***

iRODS has two main classes for resources - cache, which is typically for shorter access time/higher performance storage and archive, which is typically for longer access time storage. As Isilon has several different storage tiers available it should be assigned to the correct resource class based on the type of nodes in use. For example, in an Isilon cluster with a tier of X400 nodes and NL nodes the X400s should be assigned to the cache resource class and the NLs should be assigned to the archive class. This is possible with Isilon NAS even if both storage tiers are part of the same cluster and therefore represent the same POSIX namespace. Using SmartPools to divide the different nodes into distinct storage tiers and to specify paths under /ifs which cause data added under them to be placed on the appropriate tier and SmartConnect to ensure the proper connection to each, the "pools" created can be configured as distinct resources in iRODS. The advantage of using SmartPools this way would be to provide a significant performance boost for an iRODS cache resource. By ensuring that the cache resource only connects to Isilon nodes in the higher performance tier, faster access to data can be assured and more efficient utilization of Isilon resources can be achieved.

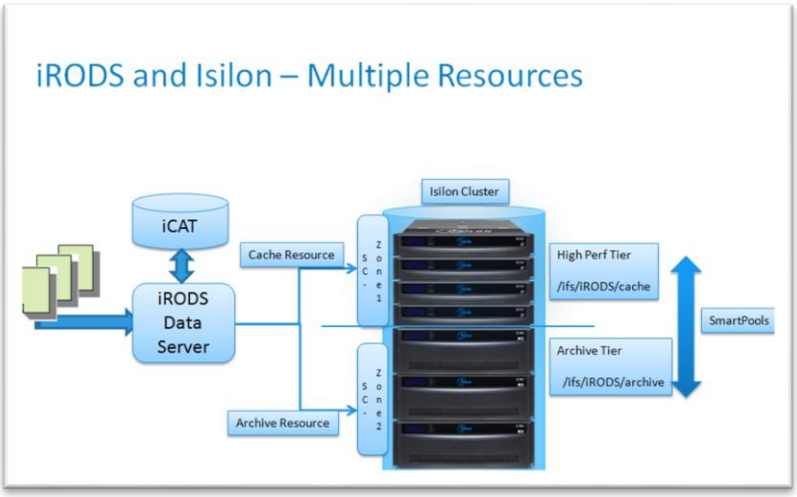


Figure 2. Using SmartPools as distinct resources in iRODS

SmartPool, SmartConnect - Configuration Example

By default on an Isilon cluster, new pools are created whenever nodes of a different type are added to the cluster. However, new pools representing subsets of the cluster can be created at any time and the default pools can be renamed. In the rest of this configuration example a SmartPool 'pool' is referred to as a tier. 'Pools' will instead refer to groups of interfaces on Isilon nodes in a single SmartConnect zone – and this 'zone' is distinct from a 'zone' in iRODS. Each tier created should have at least three member nodes and the nodes should be all of the same type. In the following configuration flow example a SmartConnect zone that contains the pool of interfaces corresponding to the nodes used in the SmartPool tier is setup to ensure that all network traffic reaches the nodes that contain the data for that SmartPool tier.

In the following example, the composition of the Isilon cluster is expected to be at least three X-series nodes and at least 3 NL-series nodes. The same configuration steps can be used when configuring an archive resource on the NL nodes present in the cluster.

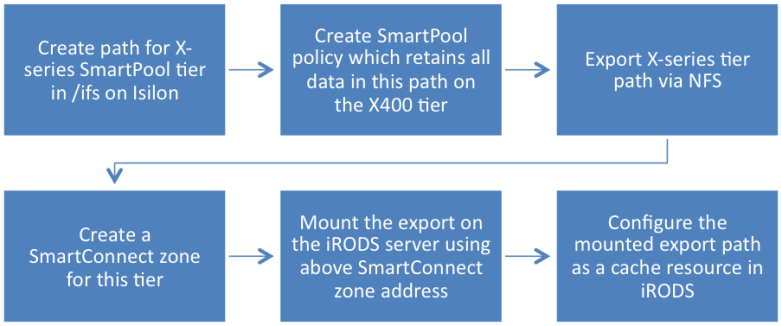


Figure 3. SmartPool, SmartConnect Configuration Example

***iRODS with Isilon SyncIQ***

The SyncIQ functionality on Isilon provides a way to synchronize data at any level (file(s), directories, whole clusters) between two Isilon clusters over a network. This is typically used to provide disaster recovery (DR) ability for sites although it is sometimes also used to replicate commonly accessed data across an institution with multiple Isilon clusters. iRODS is not aware of SyncIQ and it is not a feature that is accommodated by the 'unix file system' resource type. However, use of SyncIQ is recommended when single Isilon clusters are setup as multiple resources within iRODS and there exists a distinct archive tier. This archive tier could then be synchronized with another Isilon cluster offsite and should the primary archive resource become unavailable, be swapped out with the failed archive resource. Due to the potential for high rates of data turnover and input to a cache resource and the time required to synchronize that data with a remote resource this configuration isn't recommended for use with an iRODS cache resource.

***iRODS with Swift on Isilon***

Isilon has developed a Swift interface for use with Isilon storage. Effectively, it implements Swift as a protocol while replacing the features common to both OpenStack Swift and OneFS with their equivalents in OneFS. For example, Swift implements a protection scheme by making multiple redundant, distributed copies of data while OneFS has a faster and more efficient data protection method known as FlexProtect which utilizes Reed-Solomon Coding. The Isilon implementation of the Swift protocol interface informs the Swift proxy of FlexProtect on Isilon that bypasses its usual behavior. Other changes include informing the Swift Proxy of SmartConnect, making sure all partitions end up on the same Isilon cluster and providing 'named' access to files in addition to the standard container-object access that Swift provides.

Currently there are only a few test and proof-of-concept projects for iRODS that enable the use of Swift as a resource. As these projects develop, Isilon will continue to integrate support for Swift into the main releases of OneFS so that clusters can potentially be integrated into existing iRODS zones using Swift and leaving the rest of the Isilon cluster untouched or both new and existing installations can take advantage of the extended functionality Swift provides.

***iRODS and Isilon HDFS support***

As of OneFS 7.0, Isilon supports HDFS as a protocol. The latest version of OneFS supports HDFS 2.x. The primary advantages of using HDFS with an Isilon cluster are both the ability for Hadoop to access files directly, what is commonly referred to as 'in-place' analytics, and the greatly enhanced efficiency. Typical DAS storage setups for HDFS results in several copies of any file placed into the setup to provide a measure of protection. OneFS FlexProtect can achieve this same goal far more efficiently using Reed-Solomon Coding for all data stored on an Isilon cluster. Additionally, given the dramatically larger volume of data used for most map-reduce/Hadoop jobs the cost, both in terms of time taken over a network and the capital cost of building and maintaining a parallel DAS storage environment, of having to transfer such a large volume of data with a staging process into a DAS HDFS setup is very high. Utilizing Hadoop on data that is extant on an Isilon cluster eliminates the wait for long data transfers, makes for a more cost efficient use of existing resources and still provides rapid access to data for Hadoop jobs.

There are at least two different methods for utilizing the HDFS capabilities of Isilon with an iRODS setup. For either of these methods no data transfer to any HDFS DAS is necessary - the data never exits the Isilon cluster while still being managed by iRODS and processed with Hadoop. The illustration below shows the architecture used for either method.

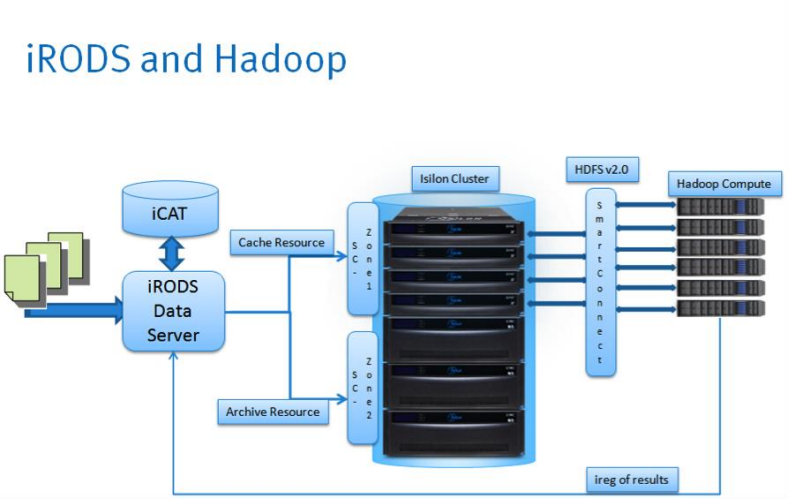


Figure 4. The architecture used for utilizing Hadoop on data that is extant on an Isilon cluster.

Moving data within the resource

Using a rule to periodically move data into a different iRODS directory (logical), the corresponding physical directory would be set as the HDFS root directory on Isilon. That directory would also have an associated SmartPool policy, which moves any of its contents into a higher performance Isilon tier. At any point after iRODS has moved data into the target directory a Hadoop job could be started and the Hadoop compute nodes will access the data via HDFS through SmartConnect – connecting directly to the high performance Isilon tier. Once the Hadoop job is complete and results have been placed back into the HDFS root directory structure on Isilon a final command from the Hadoop compute cluster could be issued to register the results with iRODS. A rule would then move the results back into the appropriate directories (logical, physical or both/either) within the Isilon resource.

Utilizing Direct Access Resources

One disadvantage of the method described above is the potential lack of consistency between iRODS and the Hadoop cluster for usernames, permissions and file ownership. If for example, a Hadoop job is started by a user who doesn't have permissions to the files which appear to be owned by an iRODS administrative user, then there is the potential that the job could fail. To ensure proper consistency between the file owner and the Hadoop job owner the Isilon, either the entire cluster or just one tier could be setup as a Direct Access Resource within iRODS. In this mode, iRODS effectively becomes an overlay for the existing Unix file system to preserve and utilize the existing permission scheme. With an Isilon tier setup as a Direct Access Resource, the data flow and movement described in the first method would remain essentially unchanged but file ownership and permissions would be modified to match what would be expected by a Hadoop instance with more restrictive security settings.

***iRODS with Isilon in the future***

Future development for iRODS with Isilon will include the creation of an Isilon specific compound resource driver based on the Unified Mass Storage System scaffold driver. Once developed, this driver will enable the Isilon resource to be accessed without being NFS mounted and will enable access to the extended functionality that Isilon provides such as SmartLock, a WORM implementation for OneFS, dynamic FlexProtect configuration, the ability to instrument intra-Isilon cluster data movement with SyncIQ and other features. The planned implementation will use HDFS to put and get files allowing for multiple parallel streams to be opened for data transfer with minimal protocol overhead. Finally, there is the possibility that some elements of the iRODS infrastructure such as the data server and iCAT could be placed directly onto the Isilon nodes; allowing the iRODS Zone to scale directly with the Isilon footprint.

## ISILON NFS BEST PRACTICES

In addition to the recommendations described above some recommendations for optimizing the performance of NFS with Isilon follow. Currently, the only way for iRODS data servers to access Isilon as a resource is via NFS and while the following recommendations will work for any NFS application, they have been specifically selected for their impact to an iRODS installation.

***10GbE Network Connections***

Isilon nodes each have two 10GbE (Gigabit Ethernet) connections and there is a minimum of three nodes in a cluster. When iRODS data services are distributed over an entire cluster of machines, each with their own physical network interface, they have the

potential to utilize all of the network bandwidth available to an Isilon cluster. The cluster can easily service all of the requests made of it providing that it is connected as optimally as possible to the machines hosting the iRODS data servers. The best way to utilize the full capability of NFS from the Isilon cluster is to ensure that iRODS data servers are connected to it via 10GbE whenever possible.

***Jumbo Frames***

Jumbo frames, which refers to raising the maximum transfer unit (MTU) from the default of 1500 bytes to 9000 bytes is advised for both client machines and the nodes within the Isilon cluster. This allows the network stack to bundle transfers into larger frames and reduce TCP protocol overhead, which would otherwise result when many smaller frames are used to transfer the same data. The actual value used for any frame can vary depending on the immediate needs of the network session established between the NFS client and server (the Isilon cluster) but raising the limit to 9000 on both the client machines and Isilon cluster will allow the session to take advantage of a wider range of frame sizes.

***NFSv4***

Both NFSv3 and NFSv4 were evaluated. NFSv4 is preferred due to the slight performance advantage it confers when working with large clustered environments accessing a common resource and the extended permissions framework it contains. However, NFSv3 is still frequently used owing to the ease of implementation and wide availability of client and server stacks.

NFSv4 provides several new features as well as improvements on the NFSv3 architecture. When working with a large distributed data management infrastructure, NFSv4 provides four distinct advantages over v3:

- A more 'stateful' implementation,
- Ability to bundle metadata operations,
- An integrated, more functional lock manager, and
- Conditional file delegation.

NFSv4 now requires that all network traffic management (congestion, retransmits, timeouts) be handled by the underlying transport protocol as opposed to the application layer as found in NFSv3. In any environment but especially in high volume, high throughput data management infrastructures – this helps free up the client for additional application specific work on the data.

NFSv4 also has the ability to bundle metadata operations using compound RPCs (Remote Procedure Calls), which reduce the overall number of metadata operations and significantly decrease the overhead required when accessing multiple files. This can be a significant factor for data management frameworks, which often require access to hundreds, even thousands of files to satisfy client requests.

An integrated lock manager provides lock leasing and lock timeouts – a considerable improvement over the previously used NLM in NFSv3, which only provided a limited implementation of these features out of band. This makes for cleaner recovery semantics and processes for failure handling when running HPC applications and distributed data management frameworks such as iRODS.

File delegation is another new feature in NFSv4 in which the server provides a conditional 'exclusive' lock to the client for file operations. This feature allows the client to treat the file as if no other resource is accessing it. If a file were to be accessed then the client would receive an immediate notification and the exclusive lock could be removed. While this delegation is in place the client can cache a greater amount of the data within the file as well as any changes made to it before notifying the server.

There are many other enhancements and new features in NFSv4 – those briefly mentioned above are only those with the most relevance while using Isilon as an iRODS resource. Isilon OneFS currently contains implementations for all of the above NFSv4 functionality with the exception of file delegation which is planned for a future release of OneFS.

Beyond the major enhancements described above, NFSv4 provides improved functionality for authentication and permission management. The ease of use within firewall-managed environments is enhanced with the entire protocol operating from a single port (2049) – there is no need for any client to contact a portmapper or the use of any ephemeral ports. For a more complete description of the protocol refer to the status and documentation available from the NFSv4 working group at http://tools.ietf.org/wg/nfsv4

***Tuning OS Parameters on the iRODS Server***

Linux is one of the most commonly encountered operating systems in scientific computing. When using Linux as a server for iRODS some parameters should be adjusted to maximize performance when mounting the Isilon resource over NFS. The default buffer sizes

for TCP are not adequate for significant data transfer, especially in a high performance computing environment. The following sysctl settings (normally in /etc/sysctl.conf) are recommended as a best practice when mounting NFS file shares:

```
net.core.rmem_max = 524287
net.core.wmem_max = 524287
net.core.rmem_default = 524287
net.core.rmem_default = 524287
net.core.optmem_max = 524287
net.core.netdev_max_backlog = 300000
```

Setting or altering the default values for the following parameters, used in both NFSv3 and v4, will affect the performance of any pipeline for NGS data.

- rsize – represents the maximum size of the RPC read packet used. (max 65536 bytes)
  - Should be set as high as possible when on an un-congested network and when using TCP as a transport protocol. If there is significant network congestion then this value should be reduced to prevent excessive fragmentation of the read packet and improve overall NFS performance.
- wsize – represents the maximum size of the RPC write packet used. (max 65536 bytes)
  - Should be set as high as possible when on an un-congested network and when using TCP as a transport protocol. If there is significant network congestion then this value should be reduced to prevent excessive fragmentation of the write packet and improve overall NFS performance.

***Isilon Cluster Tuning***

The number of threads used by the NFS server running on the Isilon nodes can also be adjusted to help enhance performance. This is especially useful in large, distributed compute environments with many NFS client connections. By adjusting the sysctl variable vfs.nfsrv.rpc.maxthreads from the CLI of any node in the cluster like so:

```
isi_sysctl_cluster vfs.nfsrv.rpc.maxthreads=24
isi_sysctl_cluster vfs.nfsrv.rpc.minthreads=24
```

the number of threads available to the NFS server can be increased. This is recommended when operating in a dedicated NFS environment or when the primary load on the cluster is NFS related. The primary side effect of any increase in the number of NFS server threads can potentially be a lowering of response times to new client requests.

# Conclusion

It is possible to use Isilon as a resource for an iRODS zone while still taking advantage of all the extended functionality provided by OneFS. The additional features offered by Isilon OneFS can complement an existing iRODS installation while also extending its functionality into new areas such as Hadoop to enable analytics on data extant in iRODS without complicated staging tasks.

Isilon clusters provide a scalable, high performance, multi-protocol NAS solution to meet the data needs of institutions working with large data grids managed by iRODS. By making a few adjustments to the way iRODS data servers access the Isilon cluster or to the cluster itself depending on the client load, organizations can realize a significant gain in the performance of their iRODS Zones. EMC Isilon is committed to developing solutions using Isilon hardware and OneFS for many uses in research and data management and is excited to work with iRODS to develop the next generation of data management solutions.

# REFERENCES:


1. iRODS. http://www.irods.org
2. iRODS Wiki. http://wiki.irods.org
3. Rajasekar, Arcot. IRODS Primer: Integrated Rule-Oriented Data System. [San Rafael, Calif.]: Morgan & Claypool Publishers, 2010.
4. EMC Isilon OneFS Operating System: http://www.emc.com/collateral/hardware/white-papers/h8202-isilon-onefs-wp.pdf
5. An Evaluation of NFSv4 Performance with NextGen Sequencing Analysis on Isilon. EMC Publication #H12322. Available on request.