EMC ISILON NFS TUNING AND BEST PRACTICES FOR NEXT GENERATION SEQUENCING

ABSTRACT

This white paper provides guidelines and best practices for tuning an EMC Isilon environment supporting next generation sequencing workflows. It is intended for users and administrators of large compute clusters connected to Isilon storage via NFS working with NGS data.

April 2014

EMC²

Part Number h12712.1

Table of Contents

EXECUTIVE SUMMARY

Next Generation Sequencing (NGS) is a combination of new technologies and methods that have dramatically increased resolution and quality of genetic sequence data. As the cost of acquiring this data has decreased over the last few years, the volume of NGS data has increased considerably. This data is "unstructured," because when it is produced it is normally a series of files containing un-ordered, un-aligned sequence data. Working with NGS data immediately after its acquisition often requires significant compute and storage resources before the data can be used in further analysis. This white paper enumerates best practices for handling NGS data as it is generated and used in downstream analysis. The unique challenges of long term access and management of this unstructured data are addressed in a separate document.

AUDIENCE

This whitepaper is intended for users and administrators of large compute clusters that support NGS applications and are connected to EMC Isilon storage via NFS.

ARCHITECTURE

The broad description of the architecture used in this paper assumes an Isilon cluster as the central file store. The file store accepts input from next generation sequencing instrumentation and acts as a primary data store for a number of compute nodes. An analysis pipeline executed from a compute node will access file data generated from sequencing instruments and the results will be written back to the same Isilon cluster using the same OneFS namespace. Instrumentation is assumed to connect to the Isilon cluster using an SMB/CIFS network that can be shared with the compute nodes. The compute nodes are assumed to connect with NFS and the best practices and tuning parameters presented here apply mostly to those compute nodes.

WORKING WITH NGS DATA

The analysis of next generation sequencing data can be categorized into the three stages: primary, secondary, and tertiary analysis.

PRIMARY ANALYSIS

Primary analysis is defined as the machine-specific steps needed to call base pairs and compute quality scores for those calls. The most common output formats for this data as they arrive from the sequencer is FASTQ. The FASTQ format is ASCII text data containing the sequence and associated quality scores. The sequence data are un-ordered and un-aligned and commonly referred to as "reads". Depending on the type of sequencing instrument used and the run settings, these files can range to a small number of large (> 250 MB) files to an extremely large number of smaller files.

The directory structure in which these files are collected can vary by instrument vendor. As an example, data generated by an Illumina sequencer can be organized by sample and flow-cell identifiers and then in subdirectories by lane ID. Files may also be gathered together by a larger sample request ID that is a common reference to an entry in a laboratory information management system (LIMS).

SECONDARY ANALYSIS

In secondary analysis, the raw reads are aligned and assembled, and variant calling is performed. Assembly and alignment matches reads against a known reference sequence (or genome). Variant calling is the process of identifying single or multiple nucleotide differences in a sequence such as single nucleotide variants (SNV), smaller insertions or deletions (INDELS), or larger structural variants such as transversions, translocations, and copy number variants (CNV).

The access patterns to this data vary based on the analysis algorithms implemented; however some general usage patterns can be established. Until the data is sorted for later use, read access to this data during this stage is typically sequential and often read in large blocks. These sequential, portioned reads can be sporadic as the process completes its work on prior portions. Write access to this data usually follows a similar sequential and sporadic pattern with the only exception being that data is written back in smaller chunks.

Some analysis applications used in NGS, especially those used for sorting, can create a number of temporary files. As a best practice, these files should be placed on direct attached storage (DAS) when feasible, instead of any kind of network file share. However, placing them on an Isilon cluster is still an acceptable approach and will not result in any loss of data or functionality. However, there could be an impact to the overall performance of the analysis pipeline, especially when sorting a significant amount of data.

TERTIARY ANALYSIS

Tertiary analysis diverges into the spectrum of study-specific investigations based on the more manageable set of differences between the sequenced samples and the reference.

This whitepaper focuses on significant contributors to performance and application behavior within primary and secondary analysis. Casava, BWA, BowTie2, SamTools, and BCFTools were collected into three different pipelines and evaluated for performance and throughput when developing the recommendations discussed in this paper.

Table 1

Stage	Operation	Pipeline 1	Pipeline 2	Pipeline 3
Primary	Base Calling	Casava	n/a	n/a
Secondary	Alignment	Casava	bwa	bowtie2
	Sorting	Casava	samtools	samtools
	Variant Calling	Casava	bcftools	bcftools

Pipelines evaluated for performance in this paper.

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) storage can be accessed from any number of machines by any number of users leveraging existing authentication services.

EMC Isilon scale-out NAS is based on the EMC Isilon OneFS operating system. OneFS combines the three layers of traditional storage architectures—file system, volume manager, and data protection—into one unified software layer, creating a single intelligent file system that spans all nodes within a cluster.

The OneFS operating system features:

- A high degree of scalability, with grow- as-you-go flexibility
- Unmatched efficiency to reduce costs
- Multiprotocol support to maximize operational flexibility
- Enterprise data protection and resiliency
- Robust security options

OneFS supports a number of additional services that provide additional functionality:

- SmartConnect provides the ability to effectively balance connections for an HPC cluster particularly when using an automounter. 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. 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. It does all this while keeping data within the same 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 some primary analysis tools such as Crossbow along with more sophisticated libraries such as SeqPig are speeding the adoption and use of Hadoop in the Life Sciences. Isilon supports the HDFS protocol natively. An Isilon cluster running HDFS 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 of data.

In addition, EMC Isilon Hardware provides:

- 10GbE For HPC Clusters. A cluster of Isilon nodes has enough bandwidth to accommodate instrument traffic as well as HPC analysis and data transfers to other tiers of storage or other collaborative efforts.

- SATA For NL And X Nodes. Using lower cost, high density and reliable SATA drives reduces the overall cost of nodes, often making them considerably less costly 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). Many modalities in Life Sciences, such as NGS, create thousands of files that can persist for a significant amount of time. The result is the file system becomes quite complex, and can require a considerable amount of time to traverse. The solid-state disks (SSDs) used with OneFS GNA can cache file system metadata and dramatically speeds up the traversal and sorting of complex file systems.

ESSENTIAL NFS & NAS PRACTICES FOR NGS

USE 10GBE WHENEVER POSSIBLE

Each EMC Isilon node supports two 10GbE (Gigabit Ethernet) connections, and there is a minimum of three nodes in a cluster. As mentioned elsewhere, applications operating on NGS data typically do so by reading as much data in as possible at any given time. As they read sequentially through a large number of files, done in parallel, they will be able to use almost all of the network bandwidth available. When these processes 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. Provided the client machines are connected to the Isilon cluster, the cluster can easily service all of the requests made of it. The best way to utilize the full capability of NFS from the Isilon cluster is to ensure that client systems are connected to it via 10GbE whenever possible.

CONSIDER AGGREGATING CLIENT INTERFACES USING LACP

Link aggregation protocol (LACP or "bonding") is used to bind together multiple physical interfaces into a single logical interface - multiplying the throughput available to the logical interface by the number of physical interfaces incorporated. For example, if a client machine has 4 1GbE interfaces available, aggregating them into a single interface would allow a single TCP session to reach a maximum throughput of approximately 2.8 Gb/s (up to 30% of the available bandwidth can be lost due to TCP protocol overhead) as opposed to any particular session being limited to roughly 700 Mb/s in the single interface configuration. LACP isn't limited to use only on client machines either; nodes within an Isilon cluster can bind together their dual 10 GbE ports into a single 20 GbE logical port on each node. The 1 GbE interfaces can do the same resulting in a single 2 GbE interface. In practice, this should be done only if the second link isn't needed for high availability (redundant) configurations. However, it can provide a significant increase in the throughput available for multiple NFS clients.

When operating a client machine with one or more 1 GbE network interfaces connected to the same network, the best practice would be to aggregate these interfaces together using LACP. If a system has multiple 10 GbE links available there is little benefit to bonding them together as there are almost no client application processes which will maximize the throughput of such a configuration. LACP can be difficult to configure as it requires changes on both the client system whose interfaces are being bound and the switch(es) to which the interfaces are connected. In large environments, and certainly within environments that anticipate rapid growth of their Isilon clustered NAS, the configuration of SmartConnect Advanced should be considered instead. SmartConnect Advanced will balance connections to the Isilon cluster by way of DNS redirection and can do so using a number of different algorithms, throughput/link utilization among them.

USE 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 Isilon storage cluster nodes. Utilizing jumbo frames allows the network stack to bundle transfers into larger frames and reduce TCP protocol overhead. 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.

AUTOMATE DISTRIBUTED PIPELINES WITH SMARTCONNECT

The massively parallel nature of most computation within NGS provides an opportunity for optimizing pipelines to take advantage of distributed resources. Brute force parallelization can be done by distributing jobs driven by shell scripts using tools such as pdsh or

GNU parallel. If more automation and resource management is desired, systems such as SGE/Open Grid Engine, torque/pbs or others can be used.

Pipeline 1, listed above, is driven primarily by make and can be easily accommodated by qmake, which will place the downstream make processes into SGE/Open Grid Engine automatically.

When using distributed pipelines it is best to have each compute resource automount the NFS share from an Isilon cluster using the SmartConnect address. SmartConnect provides a load balancing mechanism for connections, the characteristics of which are configurable by the administrator of an Isilon cluster. It can be configured to assign client connections to Isilon nodes using such algorithms as round-robin and load (connection count or throughput/link utilization). In the event of failure of an Isilon node, SmartConnect and SmartFail can reconnect the client to a new node and the cache consistency of an Isilon cluster helps ensure that the client application processes can continue their I/O operations without interruption. This can be a critical factor when running NGS pipelines, whose overall runtime can be counted in hours and days.

Because both the performance and capacity of the Isilon architecture scales linearly, clusters acting as NFS servers can easily accommodate pipelines run in distributed compute environments. Each Isilon node has a pair of 10 GbE interfaces, a pair of 1 GbE interfaces and its own processor and memory, which allow each node to accommodate hundreds to thousands of simultaneous client connections.

USE NFSV4

Both NFSv3 and NFSv4 were evaluated. NFSv4 is preferred due to the slight performance advantage it provides when working with large clustered environments accessing a common resource, as well as 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 NGS data in a massively parallel compute environment, NFSv4 has four distinct advantages over v3:

- Ability to more thoroughly use TCP,
- 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 NGS analysis, this can lead to a significant savings in overall load on the client—freeing up the client for more direct work on the dataset.

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 in NGS analysis, which often requires a pipeline to access hundreds, even thousands of files during any complete run.

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 such as the computational pipelines found in NGS.

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 direct impact on NGS analysis. 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. It also provides for more complete delegation of some responsibilities to the client, especially with regard to client side file caching. 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

TUNE CLIENT OS PARAMETERS

Linux is one of the most commonly encountered operating systems in scientific computing. When using Linux in a compute infrastructure for NGS data, some parameters should be adjusted to maximize performance. 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.
- Proto – specifies whether TCP or UDP is used for the connection.
    - It defaults to TCP but may need to be specified on some systems. The best practice is to make sure it is set to TCP.
- ac/noac – enables or disables client side caching of file attribute metadata.
    - The noac option should only be considered when operating with files that are accessed by multiple processes on a distributed set of clients. Typical operation of NGS pipelines do not require the sharing of many files between threads/processes aside from a reference and therefore should *not* set the noac option by explicitly setting 'ac'
- mountproto - this is only used by the initial mount request but should be set to TCP whenever possible
- async – delays the sending of file writes/changes until the application directs such writes or until memory pressure forces the data to be written.
- nocto – this setting will disable the NFS 'close-to-open' cache semantics.
    - Disabling cto effectively causes most of the files in the client cache to be invalidated and can save some time by eliminating a few metadata operations but is otherwise not recommended for a general purpose NFS share. Use only with exclusive use, highly multithreaded applications writing to a small common set of files.
- noatime – prevents the update of the atime (access time) for files updated by the client
- nodiratime – prevents the update of the atime for a directory

MODIFY ISILON CLUSTER SYSCTL VARIABLE

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.

OPTIMIZE MOUNT POINT ORGANIZATION

There are essentially two different models when organizing mounts on compute nodes while working with NGS data.

One is monolithic. All input and output data is both read written to the same directory structure under a single mount point. The primary advantage of using such an arrangement is that it can accommodate a wide variety of applications and configurations. The trade-off however is primarily the contention for resources within the shared mount point. There may be contention for the same logical resources such as the metadata and directory structures, the same application resources such as the NFS client thread and cache or even the same physical resources such as the network adaptor. However if maximum flexibility is desired, especially when operating within a widely shared environment, the monolithic mount model provides the easiest way to set everything up with only a modest overall impact to aggregate application performance.

The other common model in use segregates the input and output into different directories on different mount points, thereby eliminating contention for both logical and application resources on the client. To extend this model even further, the input and output mount points could originate from different networks and be accessed through separate network interfaces, which eliminates contention for physical resources on the client. On an Isilon cluster, it's possible to have any of the network interfaces operating on several VLANs or have them separated on different physical networks. When operating in this manner it is possible to apply different settings for each mount point to optimize for the particular workload. For example, changing the 'close-to-open' semantics on the output mount point while maintaining the regular NFS cache semantics on the input mount point to support a specialized application whose threads (forks) are all writing to a small set of common files. This kind of tuning and adjustment is difficult to track and maintain and can result in poor overall aggregate application performance but provide a significant gain to a small subset of highly specialized applications.

CONCLUSION

Using NFSv4 with SmartConnect on Isilon cluster and adjusting a few client system and mount parameters can provide a significant gain for NGS analysis pipelines. Coupling those pipelines with a distributed compute environment can further enhance their performance. An Isilon cluster can accommodate such an environment without any significant performance degradation.

Isilon clusters provide a scalable, high performance, multi-protocol NAS solution to meet the data needs of institutions working with Next Generation Sequencing. By making a few adjustments to the way clients access the Isilon cluster or to the cluster itself, organizations can realize a significant gain in the performance of their NGS analysis workloads. EMC-Isilon is committed to developing solutions using Isilon hardware and OneFS for all modalities in the Life Sciences and excels at providing such solutions for institutions working with NGS.

REFERENCES

CASAVA. http://support.illumina.com/sequencing/sequencing_software/casava.ilmn

BowTie. http://bowtie-bio.sourceforge.net/index.shtml

BWA. Li H. and Durbin R. (2009) Fast and accurate short read alignment with Burrows-Wheeler Transform. Bioinformatics, 25:1754-60

SamTools. Li H.*, Handsaker B., Wysoker A., Fennell T., Ruan J., Homer N., Marth G., Abecasis G., Durbin R. and 1000 Genome Project Data Processing Subgroup (2009) The Sequence alignment/map (SAM) format and SAMtools. Bioinformatics, 25, 2078-9.

An Evaluation of NFSv4 Performance with NextGen Sequencing Analysis on Isilon. EMC Publication #H12322. Available on request.