Available on CMS information server

CMS CR -2009/141

The Compact Muon Solenoid Experiment

Conference Report

Mailing address: CMS CERN, CH-1211 GENEVA 23, Switzerland

21 May 2009

Wide Area Network Access to CMS Data Using the Lustre Filesystem

J. L. Rodriguez, T. Brody

Department of Physics, Florida International University, Miami, FL 33199, USA

C. Prescott

High Performance Computing Center University of Florida, Gainesville, FL 32611, USA

Y. Wu, B. Kim, Y. Fu, D. Bourilkov, P. Avery

Department of Physics, University of Florida, Gainesville, FL 32611, USA

Abstract

In this paper, we explore the use of the Lustre cluster filesystem over the wide area network to access Compact Muon Solenoid (CMS) data stored on physical devices located hundreds of kilometres away. We describe the experimental testbed and report on the I/O performance of applications writing and reading data on the distributed Lustre filesystem established across the WAN. We compare the I/O performance of a CMS application to the performance obtained with IOzone, a standard benchmark tool. We then examine the I/O performance of the CMS application running multiple processes on a single server. And compare the Lustre results to results obtained on data stored on local filesystems. Our measurements reveal that the IOzone benchmark tool, accessing data sequentially, can saturate the Gbps network link that connects our Lustre client in Miami Florida to the Lustre storage located in Gainesville, Florida. We also find that the I/O rates of the CMS application is significantly less than what can be obtained with IOzone for sequential access to data.

Presented at $17^{th} InternationalConferenceonComputinginHighEnergyandNuclearPhysics, March21, 2009, Prague, Czech$

1. Introduction

The CMS experiment will generate petabytes of data per year, data that will be processed, moved and stored in large computing facilities at locations all over the globe. The CMS distributed computing consists of a set of facilities organized into a tiered architecture where different aspects of the data processing, Monte Carlo data production and data analysis allocated to the tier with the corresponding capability, agreed upon commitment, and to a lesser degree the proximity to the experimenters, see Figure 1. In the globally distributed system, the Tier3 facilities, the lowest rung of the tiers, typically provide local computing resources to experimenters for data analysis. These sites are to large extent on their own for maintenance and operations. The Tier0 sites, the Tier1s and Tier2, all part of a CMS centrally, managed effort, deploy somewhat complex and sophisticated hardware and software and require a significant amount of effort, provided by dedicated personnel with specific expertise. In addition, the usual methods for remotely accessing data from any of the sites rely on grid interfaces that utilize a batch-job methodology. This batch-job approach, while powerful, significantly increase the amount of procedural overhead and can impede a user's ability to access individual data elements sometimes necessary in the development of code for data analysis. We believe that a remote user's data analysis experience can be significantly improved by providing direct access to data that includes a POSIX interface and access to single data elements.

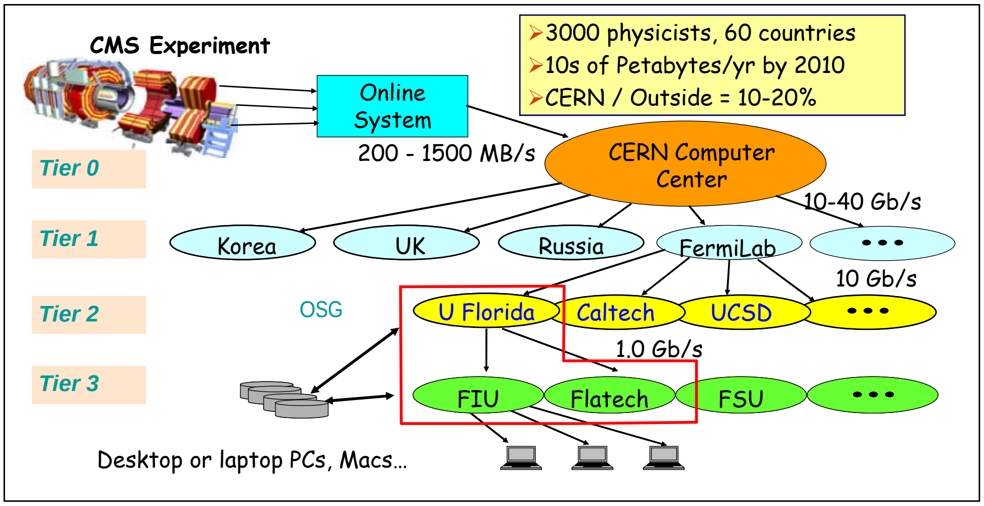

Figure 1. The CMS tiered computing model. The Tier0 based at CERN and co-located with the experiment is where the raw data filtered, processed, calibrated and warehoused. The Tier1 facilities, one per member nation seven for CMS, are also responsible for raw data storage and a majority of the processing and reconstruction. The Tier2 sites, dedicated regional Centers, 40 in all, are primarily responsible for Monte Carlo data production and support of regional data analysis groups. The Tier3 sites, loosely defined in terms of scope, are primarily used for analysis of data and code development in support of their local user base.

In this paper, we explore the use of the Lustre$^{\text{TM}}$ cluster filesystem over the wide area network to access CMS data stored on physical devices located hundreds of kilometres away. Because the Lustre filesystem can be deployed on a variety of network protocols, one of them being TCP over ethernet, a client located on the network can directly mount a filesystem that physically resides at a different location regardless of its geographical location. By direct access here, we mean that to the user the data appears as if it were mounted on a local filesystem on regular mount point. To the system's administrator the process by which remote data is made available must be easy to deploy and be as reliable and maintenance free as any other standard UNIX service. Performance is also an important consideration and in this paper, we focus our attention on I/O performance of the application used.

We will also describe the Florida Lustre testbed formed by two CMS Tier3 sites; one located in Miami, Florida and the other in Daytona Beach and the Lustre server storage system located at the University of Florida's, High Performance Computing (HPC) Center co-located with the Florida Tier2 facility in Gainesville, Florida. We include details on the hardware used, kernel modifications and tunings, report on network bandwidth, system I/O performance, compare benchmarks with actual CMS application runs, and compare I/O performance between Lustre over the WAN to the performance on locally mounted data. Finally, we also explore some of the issues concerning remote user access with Lustre, and touch upon security concerns.

2. The Lustre Cluster Filesystem

Lustre^{TM} [1] is a POSIX compliant, network aware, highly scalable, robust and reliable distributed storage architecture for clusters. The product was originally developed at Carnegie Mellon University and is now maintained and developed by Sun Microsystems Inc. The system can be run over different types of networking infrastructure and currently supports ethernet, Infiniband, Myrinet and others network protocols. It can also be deployed in heterogeneous network configurations where clients operating on different components can connect over different network protocols. Redundancy for every Lustre component is also supported. In the implementation at the U. Florida HPC no Lustre service redundancy is used. Instead redundancy is provided by the hardware elements used to run the services.

Lustre's flexibility and performance characteristic has made it the choice in distributed network filesystems for many computing facilities. It is currently deployed on large, small, commercial and public organizations including some of the largest super computing centers in the world today. Lustre has been tested with 10,000's of client systems, providing petabytes of storage and can move data at 100's of GBps [2]. The system also employs state-of-the-art security features and in the most recent development version includes support for GSS and Kerberos based security. The system is available as Public Open Source under the GNU General Public License.

A typical Lustre installation centers on the Lustre file system, see figure 2. The Lustre architecture consists of three kinds of systems: (1) File system clients, used to access the file system, (2) Object storage servers (OSS) that provide file I/O services and (3) Metadata servers (MDS), which manage the filesystem namespace. Typically the OSS run on servers that front RAID capable storage hardware typically high end fiber channel attached disk arrays. The storage attached to the servers is partitioned, organized and formatted as standard UNIX file system volumes. These storage components are known as Object Storage Targets (OSTs) several of which can be managed from a single OSS. The MDS service runs on a dedicated system with sufficient hardware resources to operate the meta-data database. The database is typically stored on a redundant storage element such as a set of mirrored drives.

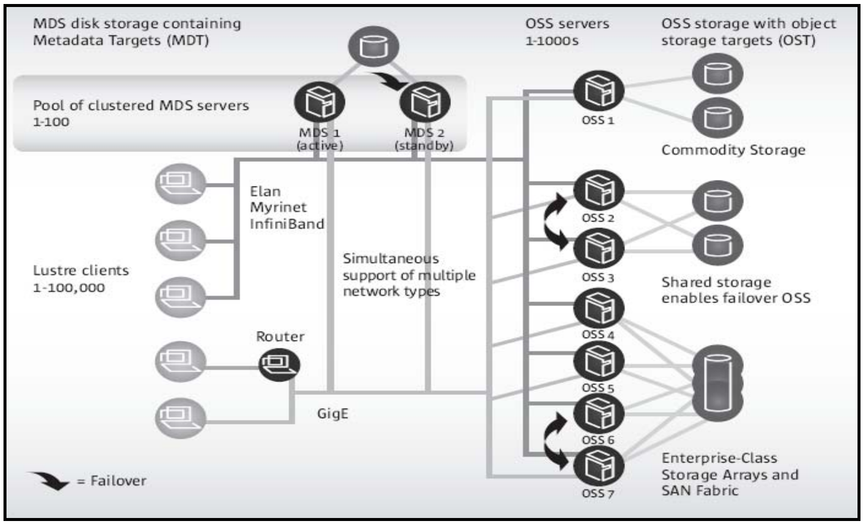

Figure 2. The figure shows the Lustre Cluster Filesystem Architecture deployed with optional redundant components. Indicated on the figure are some of the networking protocols Lustre supports, GigE, Infiniband, Elan and Myrinet. The protocols can coexist on a single filesystem. At the U. Florida HPC Center Inifiband and GigE, networks are used. In this report, we focus on the filesystem deployed using GigE over the wide area network.

3. The Florida State Wide Lustre Testbed

A testbed consisting of three sites, two Florida Tier3 Centers and the High Performance Computing Center at the University of Florida were linked to form a distributed Lustre filesystem. We created the Lustre testbed to examine the feasibility of accessing data over the wide area network.

The physical storage hardware and the majority of the server-side Lustre applications are deployed at the U. Florida HPC Center located in Gainesville, Florida a few hundred kilometers away from either of the Tier3 sites. The Tier3 sites were configured only with Lustre clients. Users at the client sites accessed data through Lustre filesystems mounted on systems running a modified LINUX kernel compiled with Lustre modules (Miami Tier3) or the stock LINUX kernel and with Lustre runtime modules. The results reported on in this paper were from tests performed between the Miami Tier3 site and the HPC in Gainesville, Florida. Tests conducted between the HPC site and the Tier3 in Melbourne and the HPC site in Gainesville reveal similar performance although tests between the two sites were very limited.

3.1. Hardware at the Florida HPC Center

The hardware employed for the Lustre system at the HPC Center [3] consists of high performance fiber-channel attached RAID storage arrays connected to servers running Lustre OSSes. There are six RAID Inc. Falcon III storage enclosures each equipped with 24×750 GB SATA II disks and a single RAID controller with redundant dual 4 Gbit/s cards with 2 GB of cache. Two Penguin Altus 2650SA servers with dual AMD 2350 (2.0GHz) dual core CPUS and 16 GB of RAM front the six storage enclosures. Each server is equipped with three Qlogic 2462 Fiber Channel 4 Gbps HCAs a Mellanox Infiniband card and 10 GigE Chelsio NIC. Another Penguin Altus server, a 1650 with dual dual core AMD 2212 HE (2.0 GHz) processors and 8 GB of RAM was used as the metadata server. Four 750 GB SATA II drives configured into two mirrored RAID arrays were used to store the metadata database. Currently a single Lustre metadata MDS server is deployed with redundancy provided by a set of mirrored disks installed on the server itself. Redundancy at the OST level is provided through the RAID5 configured arrays. The combined raw storage capacity of the six arrays is 104 TBs. The array subsystem and the frontend servers together have achieved an I/O performance of 1.8 GBps sustainable with random access I/O [3].

Network connectivity to the WAN at the HPC is provided through dedicated campus research network routes to dual 10 GigE links. The 10 GigE network at UF connects the campus network to the Florida Lambda Rail, the 10 GigE network backbone that connects universities in the state of the Florida to the National Lambda Rail.

3.2. Hardware at the Miami Tier3 Center

Lustre clients were deployed at the two Tier3 sites but in this paper we report only on measurements performed from the Tier3 in Miami. Below is a brief description of the hardware deployed and used at the Miami Tier3 site.

The Miami Tier3 site consists of a handful of servers three of which were used in the I/O performance tests reported on in this paper. The system included the following:

1. A CMS analysis server, “medianoche.phys.fiu.edu” with dual quad core Xeon X5355 with 16 GB of RAM. The system is also configured with two 750GB SATAII drives and an onboard dual GigE NIC. The system runs LINUX, CentOS 4.5.
2. A NFS fileserver “fs1.local” with dual dual core Xeon 5148 also with 16 GB RAM and an onboard dual port GigE NIC. The 5U system is configured with a single 3Ware 9000 series RAID controller with 16 ports. Each port is connected to a 1.0 TB SATAII drive. RAID 5 arrays are configured on the system and LVM is used to manage the volumes. The server is deployed with the XFS filesystem which is exported to all of the servers at the site through NFS version 3. The system also runs CentOS 4.5 with the stock kernel.
3. A development node, “dgt.hep.fiu.edu” with dual dual core Xeon 2.66 GHz processors with 2 GB of RAM. This system was configured with Scientific Linux 5.2 and was used primarily to check the reliability of the results on a differently configured system. It was also used to successfully test NAT and Luster interoperability.

Both the analysis machine and the development node were installed with recompiled SMP kernels with built-in Lustre support. The kernels version used were 2.6.9-67.0.22.EL_lustre for the CentOS 4.5 system and 2.6.18-92.1.10 for the SL 5.2 system. The kernels together with all of the Lustre clients and tools were nicely packaged into RPMs provided by Sun Microsystems. The RPMs can be downloaded for free from the Sun website after completing their on-line registration. The RPMs take care of the complete client side installation including replacement of the stock kernel. We found that to avoid performance and reliability problems, particularly evident with CMSSW [4], we needed the latest production version of the Lustre client application; version 1.6.7. The Lustre server ran an earlier version 1.6.6 while these tests were performed.

Network connectivity from the FIU campus to the Florida Lambda Rail is available through a dedicated 1 GigE link that is shared with the entire campus. The network path from the FIU Tier3 however bypasses the commodity network traffic out of FIU. Before we began our I/O bandwidth measurements with Lustre we performed tests using the iperf application to establish the network bandwidth baseline. We found that we could obtain a maximum transfer rate of about 750 Mbps. As expected the rates did vary throughout the day as campus traffic patterns changed. Also, the rates depend on the TCP buffer size and it is a well known fact that the default setting in the Linux kernel are way too small [5]. For these tests and all subsequent tests reported in this paper all servers were configured with TCP buffer sizes set to 13 MB. This TCP window size is much larger than what one would need for a 1 Gbps link with round trip time of 16 ms. The value used was in fact, optimized for a completely different set of experiments between the HPC Center which is in Gainesville, FL and a server located at the UIUC in Chicago, IL over a 10 Gbps link. The parameter was not changed since the recommend values from the ESnet website are even larger than the 13 MB used [5].

3.3. Site Configuration and Security

Access to files on the Lustre filesystem located at the UF HPC Center is controlled by the usual UNIX file properties that map onto a users' UNIX account. Users accessing these files remotely through the Lustre filesystem thus see these files with a given set of GUIDs which were assigned them on creation. In order for the GUID to be consistent across the domains all three sites shared a common GUID namespace for the unprivileged accounts enabled across the sites. For our testbed this level of synchronization was the easiest and most practical way of providing access given the nature of the testbed [4].

In addition, only rudimentary security measures were taken with the testbed given that the experiments would be performed for a limited amount of time and systems were not allowed to remain connected for long periods. For example, neither ACLs nor root_squash security measures were employed. In addition, mount access was limited to specific IP's through firewalls rules implemented on the HPC switch. Enabling root_squash is not expected to impact the performance of the filesystem over the WAN. The I/O performance impact of using Kerberos or GSS authentication will be explored when these features become available in a production release.

4. I/O Performance Results

4.1. I/O Performance Baseline with IOzone

The first series of experiments conducted were done with IOzone to set a baseline for comparison with the I/O rates of the CMSSW application. We used the IOzone [6] filesystem benchmark tool to measure the maximum possible throughput between the FIU Tier3 cluster and the U. Florida HPC Center. IOzone is a popular open source I/O benchmark commonly used to measure filesystem performance. It can be configured in a variety of ways to test various aspects of I/O performance. Also, its output is easy to read and interpret.

The IOzone tests were conducted in a systematic way examining the rates for sequential and random reads and writes. We also measured the rates as a function of record lengths, record lengths typically found in files stored on modern filesystems including those used by CMS data. Because of the large amount of RAM on our servers, we made sure that the sizes of the files IOzone wrote and read were larger than twice the amount RAM available to system cores. This prevents caching of files in main memory, which would bias the results. Unfortunately, our analysis server is configured with 16 GB of RAM making it necessary to use very large 32 GB file sizes. The large file sizes slowed the IOzone tests taking up to several hours to complete a single run. To speed things up some tests were performed with alternative boot time parameter constraining the physical RAM to 2 GB. The system was rebooted in its normal configuration for subsequent tests using CMSSW.

We performed a set of a consistency checks to ensure that the baseline results reported below were robust and reliable. First, we compared the I/O rates by running with the constrained RAM and with the server in the full RAM configuration. We then ran a subset of IOzone measurements on a different server, the development server described earlier. We then ran IOzone in multi-processor mode, whereby IOzone writes and reads using up to eight multiple streams simultaneously. All of results from these additional checks were consistent our baseline measurements. The largest discrepancy was between multithreaded runs and single threaded runs. They results in a difference of 16.8% for sequential reads and 15.6% for sequential writes.

[4] This level of synchronization is not a general practical solution and will not scale; however for our limited use testbed it is a fast and simple solution. In version 2.0 of Lustre Kerberos and GSS authentication will be available features and will eliminate the need to synchronize UID/GID across administrative domains.

The IOzone results in MBps verses record lengths are displayed in figure 3. The rates for sequential reads and writes average over all record lengths are 79.2 MBps and 90.9 MBps respectively. This corresponds to about 633 Mbps and 727.2 Mbps consistent with our iperf measurements which maxed out at about 750 Mbps. We also observe a pattern that is consistent with expectations. For example, with sequential reads and writes the rates are flat over a broad range of record lengths. For random reads-writes the I/O rates drop dramatically when the record lengths are below 2 MB. For random reads the rates start to rise when the record lengths are greater than 4 MBs.

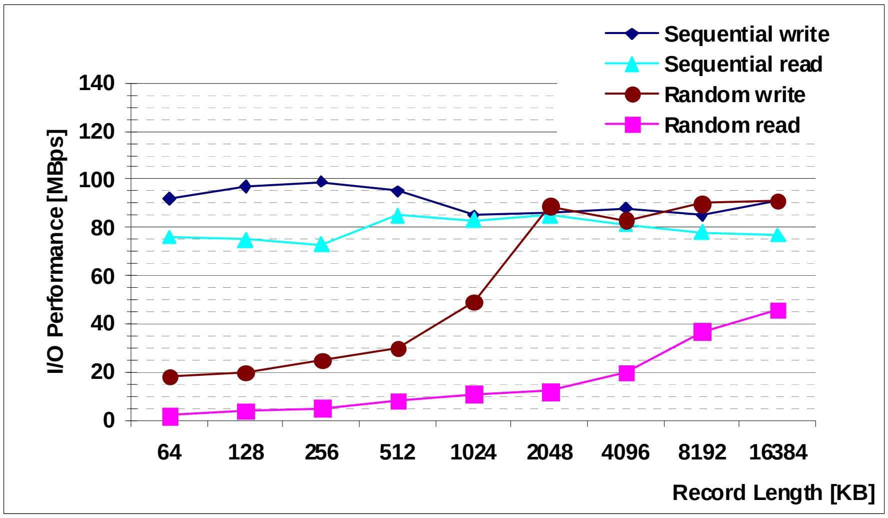

Figure 3. The figure shows the I/O performance of the testbed between FIU and UF. The plot shows sequential and random read/write performance, in MB per second using the IOzone as a function of record length.

Comparing the IOzone results to the network bandwidth measurements performed with iperf, described earlier; show that we can saturate the network bandwidth available to us between FIU and Gainesville for sequential reads and writes with Lustre mounted over the WAN. These results also indicate that the access pattern employed by the application plays a very important part in determining I/O performance. If the application reads data in small block sizes and does so randomly then the I/O performance is significantly compromised.

4.2. I/O Performance with the CMSSW Application

The main goal of this paper is the determination of the feasibility of using Lustre mounted over the WAN to access CMS data stored at a remote site. In this section we report on I/O performance measurements using a CMS application that does little processing on the data it accesses. The CMS application, referred to here as CMSSW is created by linking modules with specific data selection and processing algorithms into the CMSSW framework. The CMSSW software framework is used by members of the CMS collaboration to analyze all data collected or produced by the experiment. The particular configuration and algorithms used were selected to reflect a realistic use case of an I/O bound application. The application module within CMSSW is collectively known as "NtupleDumper" and is available from the CVS repository to everyone in the collaboration. In this study the NtupleDumper was used to read Cosmic Ray data taken during last fall's run sorts through the events and write out the some but not all of the data read. The amount of processing is kept to a minimum. We used a recent version of the CMSSW, version 2_1_10, to build the application. To eliminate possible I/O performance effects from writing to a local disk we directed the root output to /dev/null. All of the I/O measurements were taken by comparing the timing information in the CMSSW log files;

given that instrumentation of I/O performance in the CMSSW framework was not readily available in this version of software. No attempt was made to separate the root I/O performance from that due to the CMSSW framework. Runs were of significant duration such that startup time did not contribute significantly to the performance measurements. The application's runtime configuration was not tweaked, modified or tuned in anyway.

Three sets of experiments were run on the same eight core analysis servers, medianoche.hep.fiu.edu, over the same six data files. All tests were performed using the same CMSSW application. The data files are each approximately 6 GB in length. All six were replicated at three distinct storage locations: 1) a local partition on a single hard drive on the analysis server, 2) on one of the RAID5 volumes on the local NFS FIU fileserver fs1.local, described in section 3.2, and 3) on the 81 TB Lustre filesystem physically located at the U. Florida HPC Center. In all each of the application runs read over approximately 36 GB of data and ran for a few hours depending on the were the data resided and the corresponding I/O performance of the filesystem configuration.

To gauge the impact on running jobs in parallel on a single system, a typical usage pattern with today's multi-core processors, we repeated the measurements with multiple jobs running simultaneously on the same eight core system. We compared results obtained from running a single job or process to the results when running multiple jobs. We tested the performance for two, four and eight simultaneously jobs. We redid these comparisons for each of the data locations: data stored locally on one of the system disks, data stored on a local NFS mounted volume and data located on the Lustre mounted storage.

The histograms in figures figure 4 a) and figure 3 b) summarize the CMSSW I/O performance results obtained in this study. In a) we find that the aggregate rate, the sum of the I/O rates for each process on the system, drops significantly when jobs access data on a local hard drive and the number of jobs increases. This is mostly likely due to the "thrashing" of the disk mechanical components when multiple jobs compete to access different sections of the same hard drive. The decrease in aggregate rate is not observed when data is stored on either the NFS or Lustre volumes since each data file is most likely physically on different pieces of hardware. In the case of the NFS mounted volume the data are distributed across a RAID array with eight drives while on the Lustre filesystem the data is striped across many disks. The difference in aggregate rates between the Lustre mounted filesystem and the local NFS volume shown is likely a result of the latency (12 ms) introduced into Lustre filesystem. This effect is not observed when the Lustre filesystem is deployed on the local area network. We plan to investigate this in future work and will explore if there are optional parameters that can be tweaked to improve the Lustre performance over the WAN. Also in recent versions of the CMSSW framework disk caching parameters may help improve the I/O performance in this particular configuration.

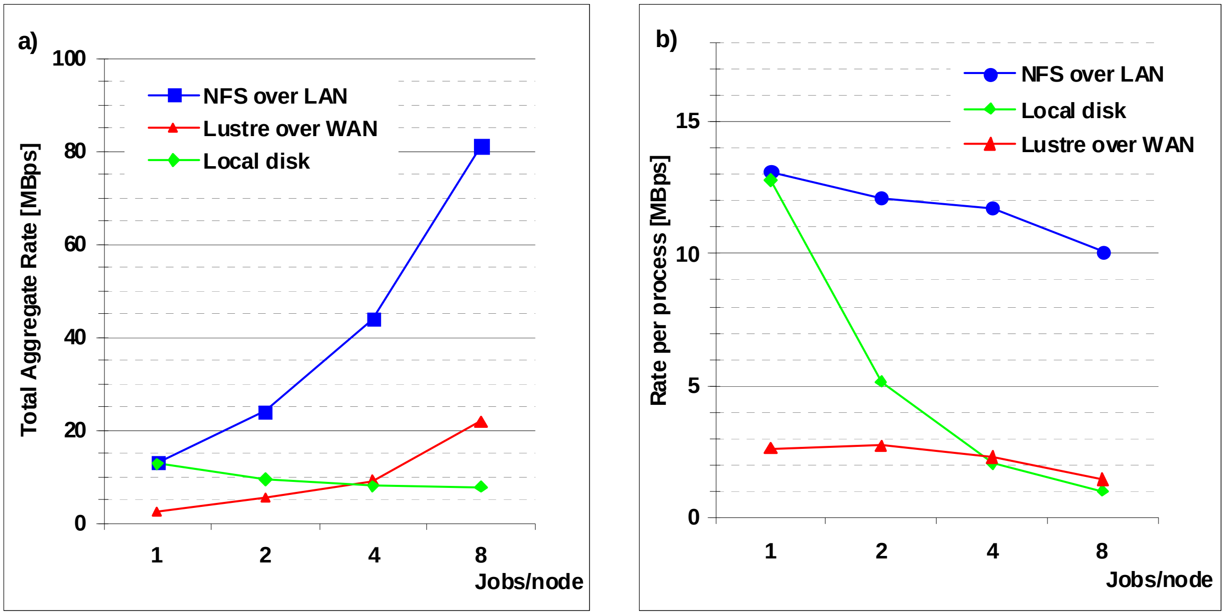

Figure 4. In a) the CMSSW Aggregate I/O Rates verses the number of simultaneous processes. These jobs were all run on a single eight-core node. The plot compares the results when running on data stored on a local disk (green), an NFS mounted RAID5 volume (blue) and on the Lustre filesystem mounted over the WAN from the U. Florida HPC Center (red). In b) the CMSSW the average I/O rate per process verses the number jobs per node. The plot also compares the results when accessing data on local, NFS and Lustre filesystems.

5. Conclusions

In this report, we measured the I/O performance of a Lustre filesystem deployed over a wide area network to access CMS data using the CMS application running a few hundred kilometres away from the physical location of the data files. We compared CMSSW I/O rates with standard benchmarks and to rates obtained by accessing data stored locally. We found that in specific cases where data access patterns are sequential the I/O performance is bound only by our network bandwidth. For the CMSSW application the I/O performance however, was significantly lower than that obtained for sequential access. In addition, in establishing the testbed, we found an extremely simple deployment process for the Lustre client part of the system. The client deployment requires at most the installation of a few RPMs.

The testbed deployed and the measurements performed establish the feasibility of using a wide area network deployment of the Lustre filesystem to access CMS data remotely. We found that even though the performance is lower than the maximum possible a server running CMSSW can access data remotely stored on a distributed Lustre filesystem with reasonable I/O performance that matches what is currently possible with other storage systems used by CMS. The tests so far are on a single system and we intend to examine the performance in a more realistic scenario with multiple servers access data over Lustre on the WAN. Also, given the early stages of our tests, makes it likely that improvements to the application's I/O performance can obtain by modifying data access parameters within the application itself, if available. In addition, a change of workflow, whereby data first gets staged from the Lustre filesystem to a local distributed filesystem, could also result in improved aggregate I/O performance.

Finally, in typical analysis activities having a shared storage space amongst collaborators within a group could significantly improve the collaborative experience. Lustre provides this ability in a simple and reliable way, especially if one of the sites already uses Lustre as part of their storage infrastructure.

References

[1] **LUSTRE^{TM} FILE SYSTEM High Performance Storage Architecture and Scalable Cluster File System White Paper**, Sun Micro Microsystems, December 2007, http://www.docstoc.com/search/Technical-White-Paper-Sun-Luster-File-system-Networking/

[2] **Solving The HPC I/O Bottleneck: SUN^{TM} LUSTRE^{TM} Storage System,** Sean Cochrane et.al., HPC Marketing, Sun BluePrints^{TM} Online, PN 820-7664-10, April 2009. https://www.sun.com/offers/docs/820-7664.pdf

[3] The Florida HPC Center hardware http://wiki.hpc.ufl.edu/index.php/Operating_Environment

[4] CMSSW is the application framework for the CMS experiment. In this paper the term CMSSW is used to generically describe CMS application software or to describe specific analysis algorithms that run within the CMSSW framework.

[5] TCP Tuning Guide, Energy Science Network website, http://fasterdata.es.net/TCP-tuning/linux.html

[6] IOzone, http://www.iozone.org/