4-2014

Comparing Remote Data Transfer Rates of Compact Muon Solenoid Jobs with Xrootd and Lustre

Gary H. Kaganas
Florida International University, kaganasg@gmail.com

DOI: 10.25148/etd.FI14071162

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

COMPARING REMOTE DATA TRANSFER RATES OF COMPACT MUON SOLENOID JOBS WITH XROOTD AND LUSTRE

A thesis submitted in partial fulfillment of the

requirements for the degree of

MASTER OF SCIENCE

in

PHYSICS

by

Gary Hal Kaganas

2014

To: Dean Michael R. Heithaus
College of Arts and Sciences

This thesis, written by Gary Hal Kaganas, and entitled Comparing Remote Data Transfer Rates of Compact Muon Solenoid Jobs with Xrootd and Lustre, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this thesis and recommend that it be approved.

_____
Prem Chapagain

_____
Peter Markowitz

_____
Jorge L. Rodriguez, Major Professor

Date of Defense: April 18, 2014

The thesis of Gary Hal Kaganas is approved.

_____
Dean Michael R. Heithaus
College of Arts and Sciences

_____
Dean Lakshmi N. Reddi
University Graduate School

Florida International University, 2014

ABSTRACT OF THE THESIS

COMPARING REMOTE DATA TRANSFER RATES OF COMPACT MUON SOLENOID JOBS WITH XROOTD AND LUSTRE

by

Gary Hal Kaganas

Florida International University, 2014

Miami, Florida

Professor Jorge L. Rodriguez, Major Professor

To explore the feasibility of processing Compact Muon Solenoid (CMS) analysis jobs across the wide area network, the FIU CMS Tier-3 center and the Florida CMS Tier-2 center designed a remote data access strategy. A Kerberized Lustre test bed was installed at the Tier-2 with the design to provide storage resources to private-facing worker nodes at the Tier-3. However, the Kerberos security layer is not capable of authenticating resources behind a private network. As a remedy, an xrootd server on a public-facing node at the Tier-3 was installed to export the file system to the private-facing worker nodes. We report the performance of CMS analysis jobs processed by the Tier-3 worker nodes accessing data from a Kerberized Lustre file. The processing performance of this configuration is benchmarked against a direct connection to the Lustre file system, and separately, where the xrootd server is near the Lustre file system.

TABLE OF CONTENTS

CHAPTER PAGE

LIST OF FIGURES

FIGURE PAGE

Chapter 1 Introduction

The Compact Muon Solenoid (CMS) research centers at the University of Florida (UF) and the Florida International University (FIU) deployed a data sharing strategy whereby CMS users at FIU could access data stored physically at the UF CMS center. The data were streamed across the Wide Area Network (WAN) without an implicit security layer to protect access from unauthorized manipulation. Because of the particular version of the file system used to establish the data link, and the network configuration of the machines responsible for the actual processing of data (hereafter referred to as worker nodes) authenticated access to the storage systems at the UF CMS center could not be established. In the thesis, we report on a novel workaround using an xrootd server, a file server developed by the research collaboration at CERN, as a means to establish a secure data connection between the UF and FIU CMS data centers.

The structure of my thesis will be laid out as follows. For the remainder of the introduction we will outline how the collaboration between the UF and FIU centers was conceived and how the access problem at the FIU center occurred. We will then explain the parameters of the performance benchmark experiments conducted with the xrootd-configured test-bed deployed at the FIU CMS center. Chapter 2 will provide a detailed inventory of the equipment used in the experiment while Chapter 3 will discuss how the xrootd configuration was set up, as well as how the control experiments were run. In Chapter 4, we will explain what measurements were used and how they were obtained. The results are reported in Chapter 5, and a discussion of certain findings will be detailed in Chapter 6.

1.1 The LHC Computing Grid

The CMS particle detector at the Large Hadron Collider (LHC) experiment at CERN exports between 25 to 30 TB of data per day. Early in the design of the CMS computing model, it became apparent that a centralized computing facility would not be the optimal way to manage the enormous storage and processing requirements of the CMS experiment. A distributed approach was instituted to share the sizeable costs of the required resources and to improve the robustness of a system. With no single points of failure and with facilities spanning the time-zone spectrum, a distributed system would nimbly balance its load [1]. The LHC Computing Grid, as it came to be known, forms a hierarchical system of member institutions tasked with various data storage and processing functions demanded by the CMS experiment [2].

The top tier, labeled Tier-0, sits near the detector facility itself at the CERN site. The Tier-0 stores and maintains the original first copy of the data (RAW) generated by the CMS detector. It also reconstructs and processes the raw data into data objects suitable

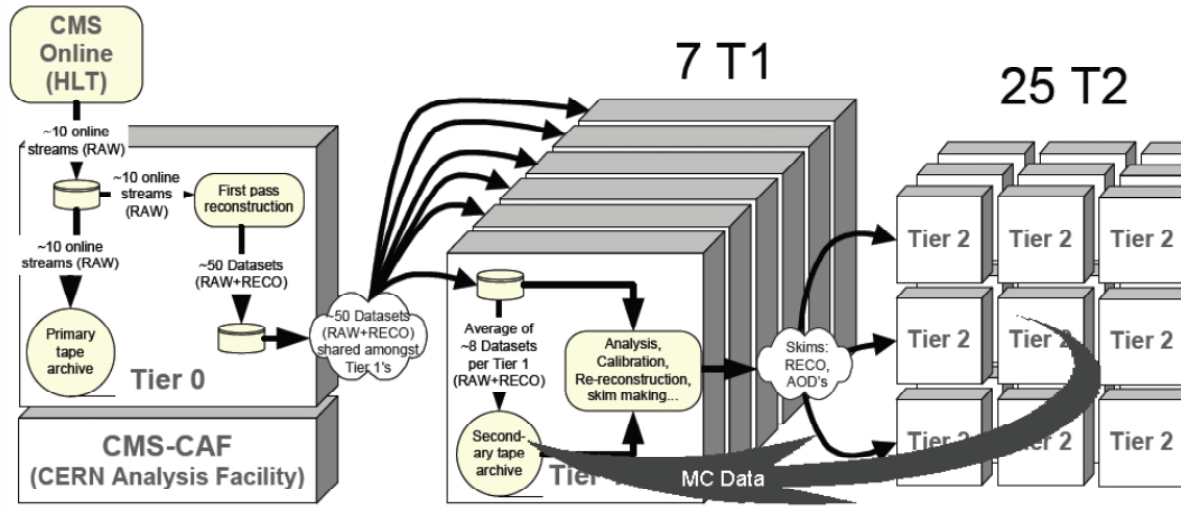

Figure 1-1: A graphical overview of the tiered architecture of the LHC Computing Grid [4].

for physics analysis known as first pass or (RECO) data. When the LHC is down, the Tier-0 reprocesses data sets with updated parameters obtained from calibrations of the previously processed data. Reconstruction of the detector data is scheduled independently of any other CMS or non-CMS events, and therefore the Tier-0 is designed to perform its functions in a self-contained manner. The second major responsibility of the Tier-0 is to distribute the RAW and RECO datasets (together referred to as FEVT data) to the Tier-1 centers [3].

Further processing of the FEVT data is carried out at the Tier-1 centers. Since copies of reconstructed data are maintained at Tier-1 centers they must be made accessible to any CMS user according to priorities set by the CMS collaboration[1]. The FEVT data, as well as other RECO data, can also be transferred, upon request, to Tier-2 and Tier-3 facilities.

Tier-2 facilities have much lower storage requirements than the tiers above, and spend a considerable portion of their resources performing Monte Carlo simulations and data analysis but can also be employed to clean up the data that comes out the detector. The CMS file transfer topology is designed for transfers to flow rather freely between Tier-2s and Tier-3s. is the free flow fo data is a necessary requirement, as Tier-3 facilities act less as a component of the CMS infrastructure and more as a sovereign computing resource serving the computing needs of their local CMS community [3].

1.2 Data access on the Grid

The services offered by the CMS computing system can be roughly divided into two sub-systems: The Workload Management System (WMS) and the CMS Data Management

[1] In practice, most of the user analysis is done at the Tier-2 or at the local Tier-3 facility. [3]

system (DM). Data are accessed using a suite of specialized CMS grid components. The WMS provides the interface between the user and the sites where the data are located. In particular, the WMS locates the computing center hosting the requested data set [4]. It also acts as a matchmaker, implementing site and global priorities that define the way processing cycles are allocated to a given user request [3].

The DM system is a set of services that allow a user to discover, access, and transfer CMS data. The Dataset Bookkeeping System tracks available data. It does not keep track of where the data are located, but holds a description of the event data that are available. To access the data itself, the user will first query Dataset Location System (DLS) that

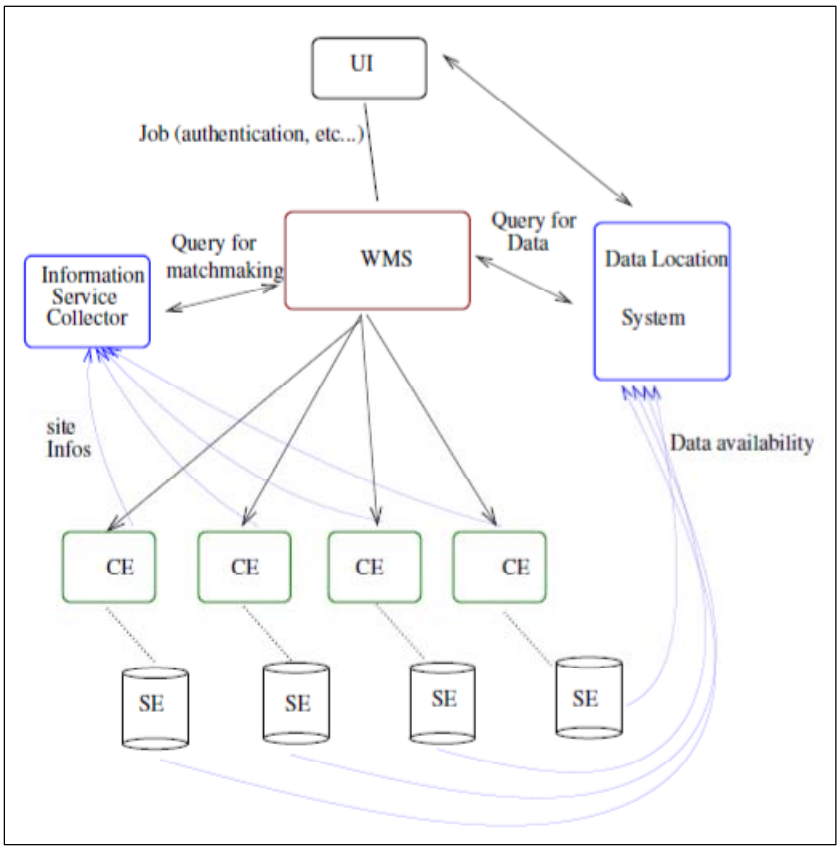

Figure 1-2: A workflow representation of the WMS. Courtesy [3].

keeps a database of sites where replicas of the data are located. The (DLS) does not provide the exact location of the data. At the site where the data are replicated, the Local File Catalogue keeps a record of the exact location of the data set [4]. Aside from providing file location services for the researcher, the DM provides CMS sites with the Data Placement and Transfer System (PhEDEx) to replicate data to their site [3].

The schematic on **Figure 1-1** illustrates how a user would interact with the CMS computing system. At the top level, a user interface (UI) is set up to provide users access to the grid services. Once authenticated through the UI, the researcher would submit a CMS job to the WMS. The WMS would then query the DLS. The DLS would return with information about which storage elements (SEs) contain replicas of the data, thereby defining the list of sites with the available data. The WMS checks the resources available at the data sites as well as the site- and CMS-wide job prioritization policies [4]. Upon finding a data site that matches the requirements of the job submitted by the researcher, the WMS sends the job to the service at the site that manages the analysis itself.

1.3 Using federated storage to manage data transfer

Figure 1-1 makes it clear that the location of the data determines where the analysis may take place[3][3][3][3]³(Collaboration, 2005b). The data transfer scheme was established 10 years ago on the premise that data transfers over the wide area network (WAN) are slow and unreliable. User's jobs are directed to where data live, as opposed to sending data to a site with the most favorable computing environment.

While this scheme is very reliable, it introduces several inefficiencies. For example, if the site with the required data set is currently working at full load, any jobs scheduled for that

data set will be delayed until computing cycles become available. Moreover, users who prefer to use their local processors may have to perform costly storage upgrades to be able to host data at their site. Future storage requirements are projected to require costly capital investments[5].

To resolve the above issues, the CMS collaboration is investigating the use of a federated storage solution. By creating an interconnected network of data sites that share data, members of a storage federation are allowed open access to any data set in the federation. Instead of downloading a data set to a local facility, or waiting for processing cycles at a CMS center that replicates a required data set, under a federated storage strategy the user can simply point his application to a centralized data catalogue.

Such a centralized data catalogue could be implemented in more than one way. For example, the federation could set up a regional redirector, which is nothing more than a server that would negotiate the just-in-time transfer of data between the storage and processing site. The redirector would hold information about where the data are located, and then direct a data stream from the storage point to the facility that would process the data. The data are transmitted to the processing farm at the rate that it is requested, and is discarded as soon as the data are no longer required by the running process.

Another way to implement a data catalogue could be by means of deploying a centralized file system over the wide area network. The remote file system could then be mounted on the machines that would process the data. The user would then simply point the application to the path to the file found below the mount point.

At the time of this writing, the CMS community seems to be gravitating towards the regional redirector solution[5]. However, other CMS institutions have explored the feasibility of implementing a distributed wide area network file system. This thesis deals with a strategy that enables all the machines at a processing facility to mount the file system of a storage resource across the wide area network.

1.4 Exploring a federated data storage strategy between the Florida Tier-2 and Tier-3 centers

An early adopter of the federated storage concept, a collaboration between the CMS Tier-2 center located in the University of Florida High Performance Computing (HPC) Center in Gainesville (the UF CMS center will be hereafter referred to as the Tier-2) and the CMS Tier-3 center at the High Energy Physics Center at the Florida International University in Miami (the FIU CMS center will be hereafter referred to as the Tier-3) began to experiment with a wide area network file system in 2008[6].

1.4.1. ExTENCI Lustre instance at the Tier-2

The collaboration chose the open-source Lustre file system as a platform for these experiments. Lustre is an open source, parallel, distributed file system designed to scale over thousands of compute nodes and is used at six of the ten largest supercomputing sites in the US[7].

The research detailed in the present thesis builds upon infrastructure deployed through resources provided by NSF project “Extending Science Through Enhanced National Cyber Infrastructure” (ExTENCI). ExTENCI was funded, in part, to provide resources to explore the feasibility of building a production level distributed Lustre environment for

CMS data analysis[8]. To fulfill this initiative extensive research on distributed WAN file systems was performed by ExTENCI researchers and in the process the Lustre file system was upgraded to the specifications detailed in Chapter 2.1.

1.4.2. Moving towards a production Lustre environment

In a non-production configuration a Lustre file system would simply be accessed by utilizing the Lustre mount architecture to access storage through client software running on a compute node. A user requesting access to the Lustre file system would simply mount the file system much like any ordinary file system, as for example, an ext3 files system is mounted to a directory or subdirectory on a system running Linux.

To ensure secure data transfer, the Lustre built-in security and authentication layer called Kerberos was configured. The Kerberos authentication system that was developed by MIT authenticates users' credentials when they present certain *tickets* that have been awarded to the user. Because authentication is granted through these tickets, the Kerberos layer prevents security breaches even from cybercriminals that have broken into the private network[9].

The Kerberos authentication layer requires that each machine that connects to a Kerberized domain have an internet address with a fully qualified domain name (FQDN). It is often the case that worker nodes in a cluster environment are placed behind a private network, and thus have no need for an FQDN. All input and output to the machine is handled by an administrator server on the cluster.

While shielding worker nodes from public access by restricting them to the private network is a common reason to keep the nodes behind the firewall, the decision may also be motivated by the growing scarcity of IPv4 IP addresses[2]. Configurations where worker nodes are kept in the private network, while secure and resource inexpensive, effectively prevent the worker nodes from establishing a secure connection to the Kerberized Lustre file system.

1.4.3. An unconventional use of the xrootd file server protocol

Before the Lustre file system was secured using the Kerberos layer, the cluster at the Tier-3 adopted the private-facing-worker-node configuration. To connect to the Lustre file system at UF the Tier-3 was required to either register all of its worker nodes with a FQDN or to find a workaround. Because of the scarcity of IPv4 internet addresses necessary to register a machine on the Tier-3 network the Tier-3 was forced to find a workaround.

The solution the Tier-3 employed consisted of establishing a secure mount of the Kerberized file system[3] to one node in the cluster. This public node on the Tier-3 had a public-facing network interface with a static IP that has a resolvable FQDN. Of course, the Tier-3 public node must also be able to communicate to the worker nodes, and is therefore equipped with a private-facing network interface connected to the same private

[2] Anticipating the exhaustion of all IPv4 internet addresses, a group led by S. Bradner and A. Mankin had submitted a recommendation to move to a next generation IPv6 protocol as early as 1995 [12]. However, at the time of this writing the percentage of users connecting to Google (as a representative measure of the IPv6 population) is a little bit greater than 3.5% [13]. To convert between protocols, software as well as hardware upgrades are often necessary.

[3] A Kerberized file system is fully protected by the Kerberos authentication protocol. Only users that can provide authentic credentials may pass the protection layer and utilize the underlying file system.

network as the worker nodes. Only this public node receives full authentication to the Kerberos layer, and here the Lustre file system is mounted.

The Kerberos realm extends up to the mount point. Users with privileges to access the mount point gain all the rights to the storage their privilege level guarantees. Users with read/write access, for example, can award other users with the same rights, even if those other users are not registered with Kerberos. What the Tier-3 required was a data access mechanism that could negotiate access to and from the Lustre mount for machines outside the Kerberos realm. In other words, the Tier-3 was looking for a way to broadcast the contents of the already authenticated mount point to the worker nodes.

The requirement exactly matched the functionality of the xrootd server.

1.4.4. The xrootd file server

The xrootd architecture is built upon the rootd data server protocol designed at CERN. The rootd data server implements the machinery that allows remote access to detector data files[4]. xrootd extended the rootd file serving mechanism making it highly scalable and adding fail-over functionality. Additionally, xrootd draws from a second technology, a clustering server, called an Open Load Balancing (olb) server. An olb server houses an object-oriented database management system created by Objectivity/DB. An Objectivity/DB database differs from the more commonly known relational database management systems, in that data is not stored into rows and columns, rather, it is organized into objects that can be manipulated by object-oriented languages such as C++

[4] After undergoing several levels of refinement, physics data from the CMS detector is organized into data trees in a file structure called root; hence, the name of the rootd server. While the rootd data server was designed to only manage root file transfers, the xrootd server provides byte-level access to files, which allows it to serve all file types.

and Java. The olbd server is used by xrootd to group multiple data repositories under one common network address and to balance the processing load between them [10].

The advantage of deploying a centralized data scheme is that locating files within a cluster of data repositories becomes a built-in function. File localization is implemented by a specialized server that is placed at the head of an xrootd data cluster, called the file redirector. Under the regional federated storage initiative, a regional redirector would be used to group all of the data repositories within the region. Using a protocol specialized for xrootd file access, a user can query the regional redirector to find his desired data set. The redirector will check to see if the data is found in any of the xrootd servers registered with it. Once the data are found it is streamed to their site as needed [5].

The xrootd server is designed to locate and serve data to remote connections. Thus, it was used to bridge the wide area network Lustre storage mounted on the registered node to all the worker nodes on the cluster.

1.5 The experiment

Unlike the typical use described by the federated data storage initiative above, where the xrootd server is installed on the cluster where the data lives, the Tier-3 workaround relies on having an xrootd server near the client, within the same cluster as the worker nodes. Aside from the workaround it provided for the Tier-3, this configuration may be used, for example, in situations where the data element is not part of the CMS community, but can be patched into the regional redirector via a wide area network xrootd server.

While the majority of applications will strongly prefer to install the xrootd file server near the underlying file system (i.e., in the same local area network), the present thesis investigates a real use-case where the xrootd file server was installed on the local area network where the underlying file system was mounted and data processed. This configuration is referred to as a LAN xrootd file server (since the xrootd file server is on the local area network relative to where the data are processed). To compare the efficiency of this approach, the performance of the more usual case of an xrootd file server in the local area network of the underlying file system was also measured. These configurations are called WAN xrootd configurations, as they are in the wide area network with respect to the worker nodes. Finally, as a control, a default configuration was assembled, where all the worker nodes were given FQDNs and Kerberos registrations. In the WAN xrootd configuration with public worker nodes, each worker node used a Lustre client to mount the Lustre file system. Since an xrootd server was not employed in this last configuration is simply termed the direct Lustre configuration.

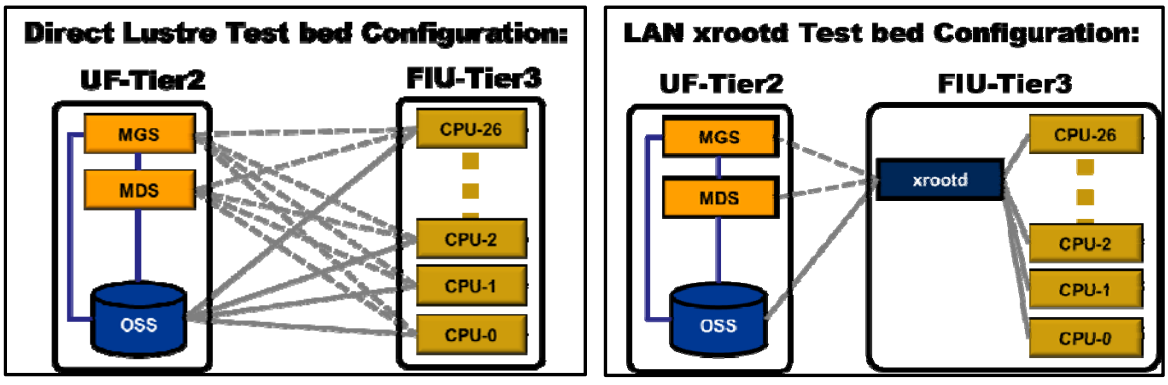

Figure 1-3: In the direct Luster test bed configuration (left), each worker node has its own public interface and FQDN. Kerberized access to Lustre storage is thus possible through Lustre clients running on the nodes. In the LAN xrootd configuration (right), all worker nodes have access only to a private network.

1.5.1. How the xrootd configurations have been be tested

The test and control configuration benchmarks were obtained by running root and CMS SoftWare (CMSSW) analysis jobs. The CMSSW analysis job is decidedly more CPU bound than the root job, as the latter was designed to test I/O performance. The CMSSW analysis application was chosen to be a Higgs to 4-lepton channel analysis to model the behavior of a typical CMSSW application. The root I/O does some minimal processing, i.e., it generates random data and organizes it into proprietary root data containers, and then adds up the number of bytes per container and moves on to the next container. It does not store data to the local storage system.

Both jobs were chosen to better investigate the impact of a low and high intensity I/O processes on the tested configurations.

Chapter 2 UF-FIU Test Bed

A wide area network Lustre file system has been deployed between the University of Florida High Performance Computing (HPC) Center, a CMS Tier-2 site, and the Florida International University Tier-3 computing cluster, located a few hundred kilometers away. Connectivity between the sites was established over the dedicated 10 Gbps research network that runs through AMPATH and the Florida Lambda Rail (FLR). Research networks normally operate in what is referred to as a campus DMZ (demilitarized zone). The DMZ is routed through networking infrastructure that bypasses campus firewall and packet sniffers that protect regular institutional traffic. The trade-off for the enhanced security is the significant gain in effective transfer rates. For example, the total transfer rate through the FIU firewalled pipe rarely breaks 1 Gbps. However, as we discuss in Chapter 2.3.2 the Tier-2 to Tier-3 connection reached steady state transfer rates of over 7 Gbps.

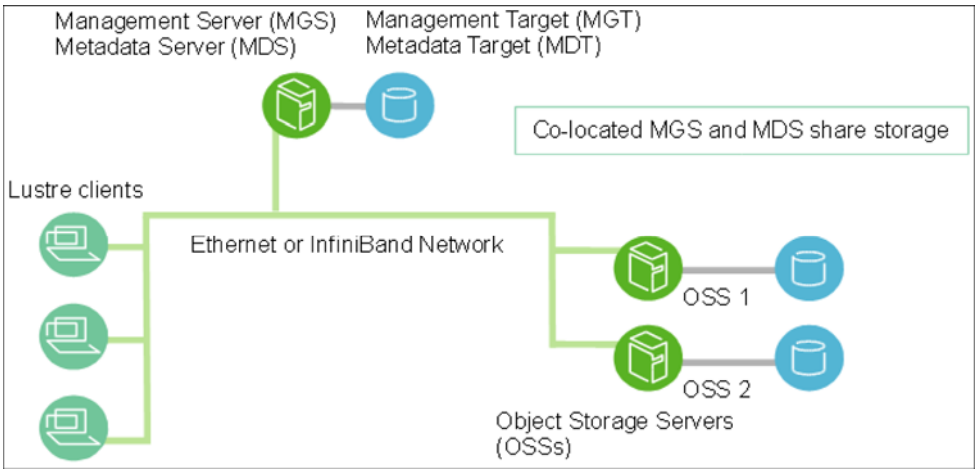

Figure 2-1: The common arrangement of Lustre components. The location of files are stored in the Metadata Target, which, in conjunction with the Metadata Server, help locate a file. Once the file is located, the Object Storage Server retrieves the file. Using instructions from

the Management Server, the user request for a Lustre client is fulfilled.

The Lustre physical storage and the server-side applications for the ExTENCI instance used in the current experiment reside at the UF HPC Center.

2.1 UF ExTENCI Lustre Storage System

The UF ExTENCI Lustre storage system is organized according to the design in **Figure 2-1**. The data chain begins with a call from a user at a Lustre client. The data request is sent to the Management Server (MGS). The location where these data will be written to or read from is determined by querying the metadata server (MDS) which retrieves metadata (such as filenames, directories, permissions and file layout) stored on the metadata target (MDT). The data are stored in a series of storage objects. The objects are mapped onto physical devices, such as RAID arrays of optical hard drives. Lustre refers to these storage objects as Object Storage Targets (OST). Actual data I/O handling is performed by the Object Storage Server (OSS) which may handle up to 8 OSTs of at most 16 TB each [7].

The OSS for the ExTENCI Lustre Storage System is housed on a node running on two AMD Opteron 2350 quad-core CPU with a 2.0 GHz clock and 16 GB of RAM. Connectivity to the OSTs is accomplished through three QLogic QLE2462 dual-channel (4Gbps each) FiberChannel interface cards. The server is connected to the WAN by a Chelsio T310 optical 10GbE NIC. Since experiments between UF and FIU have begun in 2008, the UF Lustre OSTs, have undergone a significant upgrade. At the time of this writing, the ExTENCI Lustre OSTs are arranged into three Falcon III F16SF4G FiberChannel RAID Chassis with 4GB system RAM and backup batteries. The storage

elements are arrayed in a 12 x (4+2) RAID6 configuration using 72 x Seagate 7200RPM 1TB SATA hard drives (Enterprise grade). [11]

2.2 FIU Tier-3 Linux Cluster

The FIU Tier3 facility is a CMS Tier3 site with the usual assortment of hardware and software services needed to operate a CMS grid enabled site. In the current study we only employed the Tier3's 216 core (27 worker nodes) CondorHTC Linux cluster and a server named "DGT" used to run the xrootd services needed to bridge Lustre mounts when the worker nodes were behind the private network. All of the servers, head nodes, worker nodes and xrootd server were configured with dual 2.4 GHz Xeon CPUs with 16 GB of RAM. The DGT machine was outfitted with two 500 GB hard drives, 16 GB of RAM and a dual port 10 Gbps NIC which provided 10 Gbps connectivity to both the WAN and private network. All servers except the xrootd server were connected via dual 1 Gbps NIC to a 10 Gbps capable Dell 6248 switch. The FIU Tier3 network switchers are connected to the FIU campus research backbone at 10 Gbps. [11]

Depending on the configuration used, in all experiments, either the DGT machine, or the worker nodes at the Tier-3 employed the Lustre client. A Lustre client provides the necessary kernel modules to allow mounting of a Lustre file system on the node. The client initiates an I/O transfer by querying the MDS to locate data for a read call, or allocate storage resources in the case of a write call. Once the data location is found or the resources are granted, the MDS sends instructions to the OSS, which handles the actual I/O transfer to the OSTs[7].

2.3 Network

In order to establish a 10 Gbps connection between the sites, the Tier-3 had to upgrade its 1 Gbps network. A dual 10 Gbps network interface card was installed on the DGT machine and a new switch with 28 10 Gbps ports was introduced into the cluster allowing 10 Gbps connectivity to all elements in the cluster. The switch also provided additional ports to connect the 27 worker nodes to the public uplink for the direct Lustre and direct LAN xrootd experiments.

2.3.1. Network tuning

The 10 Gbps upgrade mentioned above required certain software side modifications to the Tier-3 computerst. The network tuning below ensured that the network and the DGT node would be operating at optimal efficiency levels requisite for performance checks. The following tuning parameters were applied to DGT.

Increased the number of Ethernet frames held in buffer before they are transmitted.

```
ifconfig eth2 txqueuelen 10000
ifconfig eth3 txqueuelen 10000
```

Increased the maximum transmission unit (MTU). This is the maximum size in bytes of an in incoming or outgoing data packet.

```
ifconfig eth2 mtu 9000 up
ifconfig eth3 mtu 9000 up
```

When using large MTUs, MTU probing checks the MTU allowed at every path node between connections and adjusts the NIC MTU dynamically. Otherwise, network packets that are too large would be dropped.

```
sysctl -w net.ipv4.tcp_mtu_probing=1
```

Sets the congestion control algorithm to htcp, which enables dynamic control between high- and low-latency connections. Even though connection between the sites was fixed, the times that test were preformed were not. To avoid noise due to minor link congestion differences, this control was set.

```
sysctl -w net.ipv4.tcp_congestion_control=htcp
```

The 10 Gbps link was able to transmit much larger data payloads than the original 1 Gbps connection did. If the NIC is not capable of routing this data as it arrives, it requires a large enough buffer to dock the incoming data, until it can be routed. To increase the docing bay, rmem_max and wmem_max set the maximum receive and send buffers sizes for all types of connections.

```
sysctl -w net.core.rmem_max=67108864
sysctl -w net.core.wmem_max=67108864
```

tcp_rmem and tcp_wmem define, in this order, the minimum, default, and maximum receive (rmem) and send (wmem) buffer the OS allocates to each TCP socket.

```
sysctl -w net.ipv4.tcp_rmem='4096 87380 33554432'
sysctl -w net.ipv4.tcp_wmem='4096 65536 33554432'
```

In high throughput connections the NIC may receive packets faster than the kernel can process them. If the buffer is full, packets get dropped. To remedy this we increase the input queue length.

```
sysctl -w net.core.netdev_max_backlog=250000
```

The ExTENCI Lustre network has been thoroughly optimized, although the extent of the optimization goes beyond the scope of this paper. The xrootd file server at the Tier-2 was optimized to network settings very similar to DGT.

2.3.2. Network benchmarks

Before any of the CMS tests were performed, it was critical to test the maximum transfer rate between the sites. Had the CMS tests returned a very low transfer rate, it would not have been possible to determine whether the low rates were due to the critical components in the test or if the link between the two sites was slow. These measurements also provided a relative benchmark against which to compare the CMS tests.

The network I/O transfer rates between the sites were tested using the iperf Unix utility. The iperf utility specifically measures TCP bandwidth between two nodes. Two separate tests were conducted. Once between DGT at the Tier-3 and the OSS at the Tier-2, and then again between the worker nodes at the Tier-3 and the OSS, once the worker nodes at the Tier-3 were given public connections.

The maximum sustained transfer rates measured between DGT and the OSS is in the range of 8.2 Gbps. To achieve this bandwidth at least nine iperf clients are connected simultaneously, after which the total bandwidth levels off.

Nodes in Group	*(Mbps) Node Avg.*	*(Mbps) Total*
1-11	854.0	8540
2-12	868.5	8685
3-13	851.6	8516
4-14	857.9	8579
5-15	864.5	8645
6-16	851.4	8514
7-17	873.6	8736
8-18	859.1	8591

Nodes in Group	*(Mbps) Node Avg.*	*(Mbps) Total*
9-19	857.9	8579
10-20	862.9	8629
11-21	867.1	8671
12-22	868.4	8684
13-23	865.8	8658
14-24	858.8	8588
15-25	862.3	8623
16-26	863.2	8632

Table 2-1: I//O transfer rates for worker nodes. Nodes are collected into groups of ten. The average bandwidth is 8617 ± 63 Gbps.

The throughput measurements from the worker nodes were run on various groups of ten simultaneous nodes, to mitigate bandwidth artifacts from any one group. While in practice, test with all 27 worker nodes were run, we assume that average throughput will not be considerably different from the highest sustained I/O transfer rates. **Table 2-1** shows results of test performed on 16 groups of worker nodes. The average sustained bandwidth observed was 8617 Mbps with an average variation between tests of 63 Mbps. The group with nodes 7-17 exhibited a peak bandwidth of 8736 Mbps.

Chapter 3 Test bed Configuration

Four file access configurations were tested. Our primary goal was to measure the performance of an xrootd configuration that was located in the wide area network of the Lustre file system and local to the worker nodes. As explained in the introduction, in this configuration, labeled the LAN xrootd configuration, the worker nodes were able to connect to the Lustre file system—via xrootd—without the fully qualified domain names required by the Kerberos authentication layer.

The LAN xrootd configuration was benchmarked against a standard Lustre connection. In the direct Lustre configuration the Lustre file system was directly mounted to each worker node using the Lustre client. In order to meet the Kerberos authentication standard, each worker node was placed on the public network and registered with a fully qualified domain name.

To provide a context for the performance measurements we compared the results from the LAN xrootd configuration to a configuration where the xrootd server was located in the local area network of the Lustre file system (and in the wide area network with respect to the Tier-3 worker nodes). Configurations were the xrootd server are in the WAN of the worker nodes are labeled the WAN xrootd configuration. Tests with the worker nodes in the public network as well as with the nodes on the private network were performed.

3.1 LAN xrootd configuration

In the local area network xrootd configuration all worker nodes at FIU were connected only to a private network. The Kerberos realm extended to a mount point at the xrootd file server, the DGT node. The DGT node is equipped with dual 10 Gbps network interface cards (NIC). One is a public-facing NIC, the other a private-facing NIC that connects to the worker nodes via the private Tier-3 network. On each of the worker nodes we accessed the Lustre storage via built-in xrootd support in the root and CMSSW applications, essentially using the string:

```
root://<xrootdserver.local:1094//extenci/cms/…
```

to specify the path to the data files in the application. Here the string "/extenci" is the top level directory or mount point for the remote Lustre file system.

Naturally, the Kerberos authentication reaches as far as the mount point on the xrootd file server. Once mounted, the files are in the scope of the xrootd file server and are free to be exported elsewhere. Since all worker nodes are behind a private network, each with 1 Gbps capability, the total bandwidth for a batch of jobs is theoretically limited by the maximum network bandwidth capabilities of the xrootd server which is configured with 10 Gbps connectivity to both the private and public networks. We found however, that network bandwidth was likely not an important factor in degraded I/O performance of the test bed since throughput never exceeded more than 20 percent of the 10 Gbps link.

3.2 WAN xrootd configuration

In most production environments, the xrootd file server would be installed local to the file system being served. We replicated this setup for the WAN xrootd configuration. At

the Tier-2, the xrootd server is installed on a machine called VMUF that connects to the Lustre file system through the private network. The Kerberos realm extends to the mount point at VMUF. Like DGT, VMUF is also equipped with dual 10 Gbps NICs.

The WAN xrootd configuration is tested two ways. The first test was done with all of the Tier-3 worker nodes on the private network, in close analogy to the LAN xrootd configuration. Here the Tier-3 worker nodes accessed the VMUF xrootd file server via the DGT machine acting as a NAT[5]. This configuration was meant to show whether the results observed on the LAN xrootd configuration were related to the use of an xrootd server.

Still using the WAN xrootd server, the tests were run with a direct connection from each of the Tier-3 worker nodes. Each worker node was given an IP and connected to the wide area network in order to connect to the xrootd server on VMUF. This configuration was chosen to pinpoint the effect of making data requests to the WAN xrootd server with multiple computers simultaneously, as was the DGT xrootd server instance.

3.3 Direct Lustre configuration

In the final configuration, which we label “direct Lustre” we eliminated the xrootd layer. Instead we assigned each worker node its own public IPv4 address and a fully qualified domain name as well as Kerberos principles for each one. Each worker node was also configured with the Lustre client. The worker node was able to mount the Lustre file

[5] The job of a NAT is to handle traffic to the public network for a machine in the private network. In the process, the NAT assigns the public IP it operates under to all private nodes that route their traffic through it. To the outside world, all the private nodes have an identical IP address—that of the NAT.

system to a local directory. Therefore file addressing in job scripts was done in the usual way one addresses files mounted locally, with a string similar to the following:

```
/extenci/cms/…
```

The direct Lustre configuration served as a control experiment. Whatever influence the DGT server played on the results was eliminated altogether. Additionally, this is the configuration that would be used in a production environment, with every accessing machine providing its own authentication.

Chapter 4 Procedure

Test for two different I/O profiles were prioritized in the experiment. A root I/O application designed to test I/O output performance with minimum CPU processing and a Higgs analysis which executes significantly more CPU cycles between data calls and is more reflective of a standard CMSSW routine.

The trials were conducted in batches of 32 to 192 jobs. This scheme was selected, specifically because the Higgs analysis consisted of 32 unique root data files. To reproduce the effect of a batch running 192 simultaneous data files, the original 32 files were copied and renamed with unique file names. The root routine included 42 unique files, which were similarly replicated to form 192 unique files. Since the computation was done at the Tier-3, the retrieval was, by design, done from the Tier-2. Therefore, all the files were kept in directories within the Tier-2 Lustre file system.

4.1 Measurement technique

The purpose of the experiment was to benchmark the use of xrootd at the processing site against the direct access of the files. While I/O transfer rates are certainly an illuminating measure, these data may hide the actual efficiency of the underlying technology. The data caching that occurs at various points in the data transfer path have a significant effect on network data transfer rates. Caching is standard operating protocol for most file systems. To optimize data retrieval, an operating system makes an intelligent guess about what data that have recently been served will be requested again. It stores this data in the local

system in the random access memory for very fast retrieval. Data caching is aggressively performed both by the Linux kernel on the worker nodes and by the Lustre file system.

As explained above, all files used in the trials had unique file names to mitigate caching. Additionally, the cache at the worker nodes was reset before performing any tests[6]. These measures notwithstanding, caching was still observed. Since bandwidth measures cannot provide any insight to the level and efficiency of file transfer in systems that implement data caching, comparing I/O transfer rates alone may prove inconclusive[7]. To compensate for the shortcomings in bandwidth measures, another key gauge was used, the total time to completion of a batch, and hereafter referred to as the wall time. Ultimately, productivity is measured by the amount of time that is saved. The wall time offers a strict comparison of the relative productivity of each system and while many other considerations play a part in choosing a data transfer strategy, productivity is a key parameter.

The I/O transfer rate for each trial was measured to investigate the relative throughput of the data transfer strategy. The rate was measured from the OSS itself, as it acts as the centralized point of departure for all data requests to the Lustre file system used in the experiment. The Linux kernel maintains a record of the total bytes of data that have entered and exited the network interface of the underlying device. These records are refreshed at intervals on the order of a second. By querying the record at given intervals

[6] The standard method for flushing the cache and memory buffers is to send `sync; echo 3 > /proc/sys/vm/drop_caches`, were `/proc/sys/vm/drop_caches` is a virtual kernel directory designed for the purpose of clearing the cache.

[7] In fact, the higher the caching efficiencies gained, the lower the data requests will be and therefore the lower the measured bandwidth. Caching is not a crime, quite the opposite. The intelligent allocation of data is usually a hallmark of the effectiveness of the system.

and dividing the difference in the bytes of two successive queries by the query interval, the bandwidth is the result. This is given by the formula,

$$\frac{Q_1 - Q_2}{I} = B$$

Where Q_1 and Q_2 are the bytes recorded in the first and second query, respectively and I is the interval in seconds. The total change in bytes per second is the bandwidth, B.

The wall time per job batch represents the time to complete *all* the jobs in the batch. As opposed to assigning an arbitrary processing interval and measuring the progress achieved in that duration, it was decided that the best approximation of use case efficiency would be given by mimicking real case scenarios and allowing the batch to proceed to the end.

Chapter 5 Results

Transfer rate and wall time tests were conducted on the three xrootd configurations against the direct Lustre connection. For both experiments, the LAN xrootd configuration performance was significantly below the controls. For the more CPU bound Higgs Analysis experiment, the LAN xrootd configuration performance results were somewhat closer to those from the other configurations. However, the root Routine experiment, which is significantly more I/O intensive than the Higgs Analysis easily saturated the transfer rate capacity of the LAN xrootd configuration.

As the results will show, the transfer rates in this experiment exhibited a significant amount of volatility. From one measurement to the next the difference in transfer rate was observed to change drastically. Because of the large number of factors that may affect traffic on a wide area network it is not always possible to find the precise source of transfer rate fluctuations. The error bars in the figures below capture the volatility of a job batch. Volatility was calculated as the standard deviation of the instantaneous transfer rates measured at 5 second intervals.

5.1 Higgs Analysis results

Figure 5-1 is a comparison of the average transfer rate measured while running the Higgs Analysis prepared for the experiment. The LAN xrootd configuration clocked the lowest average transfer rate, and is significantly lower than the next highest measured configuration. Across all trials the average transfer rate measured was 544.5 Mbps with a

standard deviation of 185.8 Mbps. The peak transfer rate for the LAN xrootd configuration test of the Higgs Analysis was observed with 192 jobs at 820.7 Mbps.

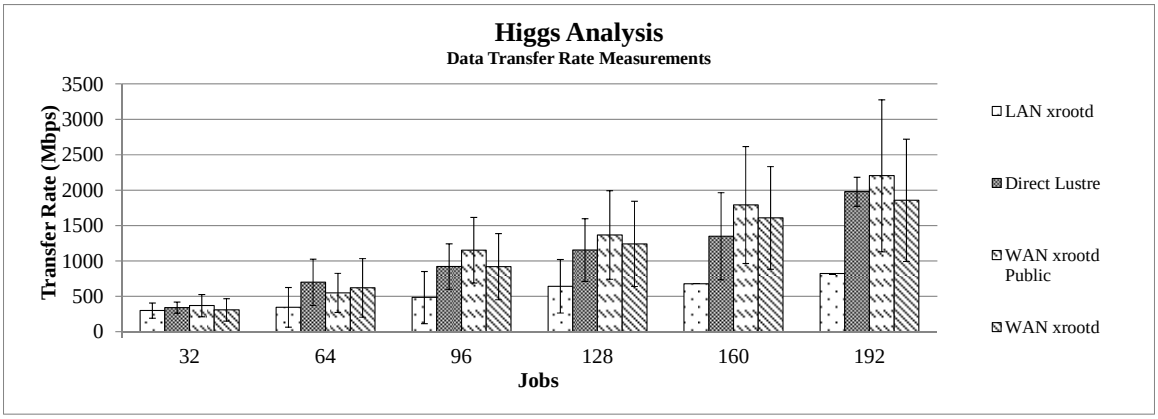

Figure 5-1: The average transfer rate measured when the four tested configurations executed a batch of Higgs Analysis jobs. To demonstrate a measure of volatility, the error bars on this chart are the computed standard deviation of the sample.

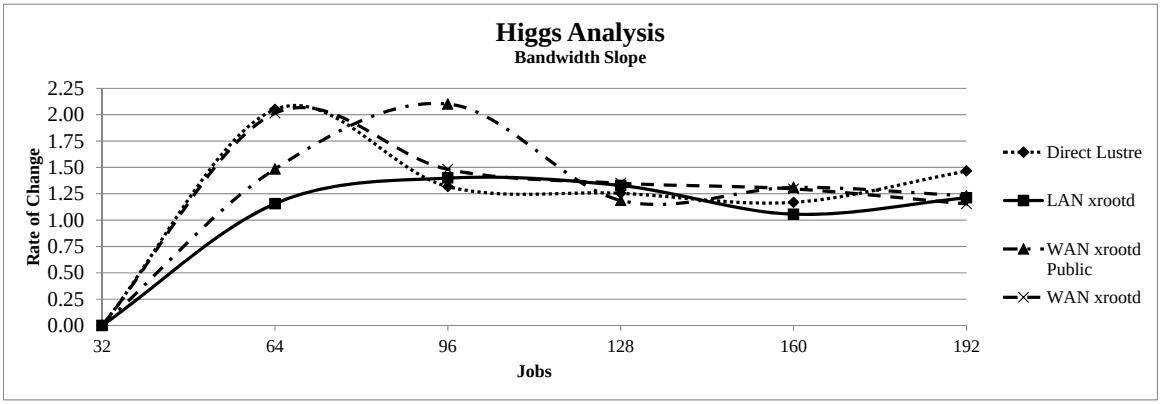

Figure 5-2: The rate of increase in transfer rate from one job batch to the next. A rate of 1.0 indicates no change, while a rate of 2.0 indicates a linear increase in transfer

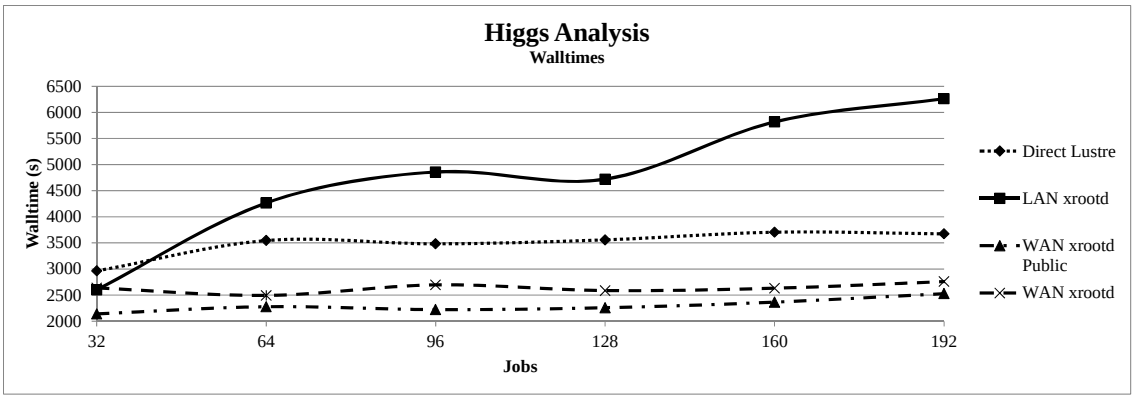

Figure 5-3: The LAN xrootd configuration exhibits signs of I/O saturation from the outset. Working at maximum capacity throughout each trial, the wall time for the LAN xrootd configuration scales linearly with increasing batch quantities.

The performance measurements from the direct Lustre configuration tell a different story. No considerable saturation is evidenced until batches of more than 64 jobs were tested. Moreover, **Figure 5-1** shows that the transfer rate scaled by approximately 50% between batches of 64 and 96 jobs and only then approached a saturation behavior analogous to that of the LAN xrootd configuration.

The incongruence between the LAN xrootd configuration transfer rate results and the other configurations, as well as the mostly flat slope (i.e. a slope of the order of unity) of the LAN xrootd configuration is evidenced in **Figure 5-2**. While all the direct Lustre and WAN xrootd configurations increased their throughput twofold with as the number of jobs increased from 32 to 64, the LAN xrootd configuration transfer rate increased only by 16% in that same range. It is clear from **Figure 5-2** that all configurations reached some amount of saturation beyond 96 jobs for the WAN xrootd configuration connecting to public worker nodes and 64 job batches for all other configurations. Nonetheless, the LAN xrootd configuration had saturated its maximum transfer rate limit. This limitation becomes pronounced in the results of the I/O bound root routine.

The WAN xrootd configuration accessed through public facing worker nodes matched up better to the direct Lustre benchmark. In fact, in almost all trials it recorded a higher transfer rate. On average the transfer rate was measured at 1,238.5 Mbps with an average volatility of 644.5 Mbps and a peak at 2,203.2 Mbps with an average volatility of 1,072.8 Mbps for 192 jobs compared to an average transfer rate of 1,073.5 Mbps with 516.4 Mbps of volatility and a peak rate of 1,977.2 Mbps at 204.6 Mbps of volatility for the direct Lustre configuration. The significant volatility for the WAN xrootd measurement will be dealt with in more depth in the discussion section.

The wall time measurements of **Figure 5-3** neatly arrange themselves in the inverse order of the average transfer rate. The high correlation is a result of the efforts spent minimizing data caching between trials to ensure each test would be independent.

5.2 root Routine results

The root Routine tests clearly overwhelmed the LAN xrootd configuration. The flat performance of the LAN xrootd configuration in **Figure 5-4** is supported by the flat slope curve of **Figure 5-5**. A peak saturation transfer rate of 1,000 Mbps with a volatility of 414.2 Mbps is achieved by the LAN xrootd configuration for the 32 job batch. As the job numbers increase, the average transfer rate actually drops, in a clear sign that the LAN xrootd configuration became saturated.

While the WAN xrootd configurations achieved a higher performance than a direct Lustre configuration in the Higgs tests, with the more I/O intensive root routine the direct Lustre configuration recorded average transfer rates at least 27% higher than either WAN xrootd configuration. The average transfer rate for the direct Lustre configuration was

recorded at 4,537.95 Mbps with a volatility of 1,526 Mbps, with the next fastest configuration clocked at 3,565 Mbps with a volatility of 941 Mbps for the WAN xrootd configuration and 3,227 Mbps with a volatility of 294 Mbps for the WAN xrootd configuration accessed by public facing worker nodes.

Examination of the slope graph on **Figure 5-5** the direct Lustre configuration was able to scale with better performance than the other two configurations for job batches with less than 128 processes. This batch size represents the transfer rate saturation point, after which performance for all configurations was impacted, as evidenced by the graphs in **Figure 5-5**.

Again the wall time results in **Figure 5-6** confirm the independence of the trials.

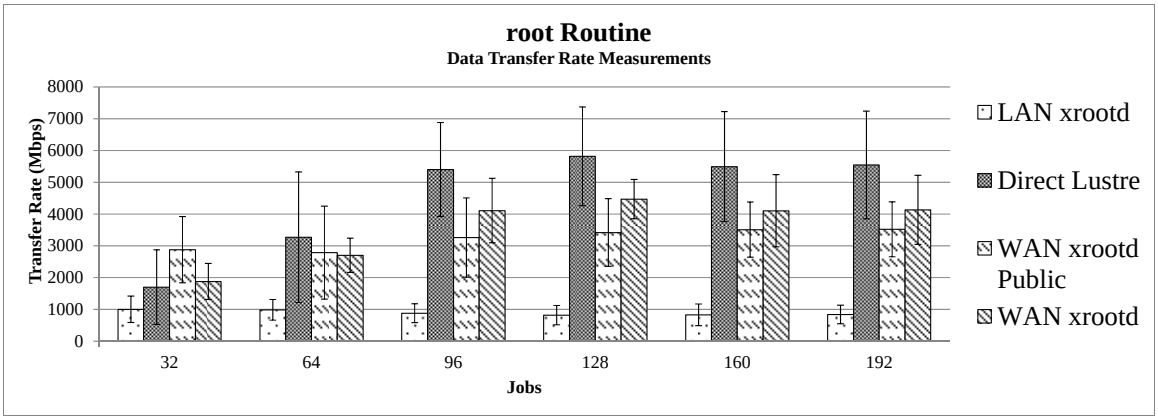

Figure 5-4: The rate of increase in transfer rate from one job batch to the next. A rate of 1.0 indicates no change, while a rate of 2.0 indicates a linear increase in the transfer rate. Again, the error bars are the standard deviation of the sample.

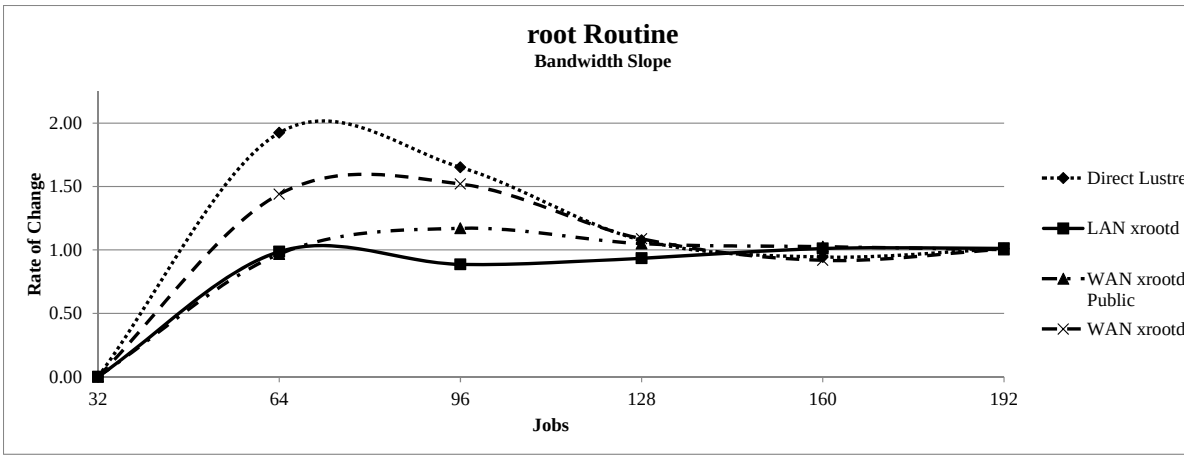

Figure 5-5: Unlike the Higgs test, the LAN xrootd slopes on the root routine is consistently at or below unity as the configuration was quickly overwhelmed.

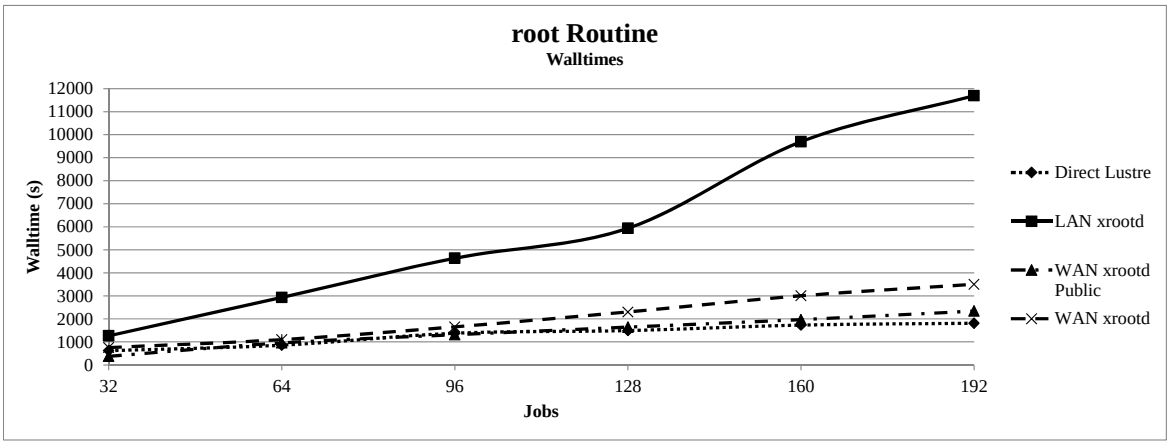

Figure 5-6: The wall time records the time elapsed from the instance the batch of jobs is submitted to the batch processor until the final job has completed.

Chapter 6 Discussion

The discrepancy between the LAN xrootd configuration results and the other tests is significant enough to warrant deeper investigation. Because the gap is so wide, and because the saturation point is so consistent in the root tests, it was originally supposed that a software throttle had been imposed at some point in the connection chain. The following systems were interrogated:

- The Lustre server
- The Lustre client
- The xrootd server
- The Tier-3 switch configuration
- The DGT network interface cards
- The DGT kernel

None of these showed any evidence of throttling.

6.1 LAN xrootd configuration failure points

The implication then was that at least one of the main components of the LAN xrootd configuration failed. The likely points of failure could be found in the connection between the Tier-2 and the Tier-3, or in the network layer at the DGT machine, or at the processor level for DGT.

6.1.1. Test site connection pipe

The connection quality between the test sites was established in two ways. As mentioned in Chapter 2.3.2, iperf tests demonstrated that transfer rates above 8 Gbps between the sites are common. Moreover, the consistency of the results also eliminates the possibility that any group of worker nodes operated with a network disadvantage. The most

convincing evidence, of course, are the transfer rates achieved by the WAN xrootd and direct Lustre configurations, with rates above 5.5 Gbps reported.

6.1.2. The DGT network layer

The DGT network layer was also shown to be functioning correctly as results of the WAN xrootd configuration trials attest. There, all the worker nodes routed their inbound public connection through DGT. Had the network congestion control on the DGT machine failed, the same saturation patterns observed in the LAN xrootd configuration test would have been evident. The opposite is true. The results from both the Higgs Analysis and the root Routine experiments in the WAN xrootd configuration are comparable to the same experiments run with direct connections to Lustre file system.

6.1.3. The DGT processor

Having isolated both the connection pipe between the test sites, and the network layer at DGT and finding them to be operational, we then turned investigate whether the xrootd service had overwhelmed the DGT processor. The concern here was that due to the latency of the connection between the Lustre file system at the Tier-2 and the xrootd server at the Tier-3, the DGT machine has trouble properly buffering and serving the incoming data. This a processing issue.

Testing this effect was not a straightforward exercise. To isolate problems at the processor level on the DGT machine and design a test comparable with the other configurations it would be necessary to eliminate the latencies of a wide area network storage target. In other words, a test bed analogous to the ExTENCI test bed would have to be replicated at the Tier-3 site, thereby eliminating the effect of placing the xrootd

server in the WAN of the file system. We would then conjecture, that if the xrootd service on DGT would be employed to export a local file system (i.e. a file system in the local network), but the sustained transfer rates of that configuration are still low, then, having eliminated all other factors, the problem must reside at the computational level.

A test bed like the Tier-2 ExTENCI Lustre was not possible to reproduce. Instead, we turned to the existing storage resources available at the Tier-3 site. On the local area network one of two storage sources could, in principle, be employed. The first is the hard drive resident in the DGT machine, the second being an NFS storage element in the Tier-3 local area network.

Neither proved to be an adequate solution. The Lustre ExTENCI test bed operates a dedicated file server (the OSS) connected via optic fiber to a bed of 72 drives arranged in a RAID cluster. By comparison the local NFS storage system consist of 16 drives housed in the same machine the NFS file server. More significantly, the network interface card at the local NFS target has a maximum transfer rate capacity of 1 Gbps, compared to the 10 Gpbs interface of the ExTENCI OSS.

Using the hard drive resident on the DGT machine posed a different challenge. While the published interface speed between the hard drive and the processor is around 3 GB/s, this transfer rate assumes that only one thread is requesting data at one time. Our test rely on dozens of parallel data requests. Due to the physical limitation of using one hard drive, simulating the concurrent data seeking capacity of the Lustre file system was not possible.

To partially overcome both these limitations, two concessions were made. First, the storage targets for the jobs would be split between the local NFS storage and the resident hard drive. And second, the job batches ran for this experiment would be small. Batches of 8, 16, and 32 jobs were tested.

To eliminate the DGT processor as a failure point, it would be sufficient to see average transfer rates at least 50% above the LAN xrootd configuration average of 1 Gbps for the root Routine experiment (see Chapter 5.2).

The results of this test were unexpected. The steady-state bandwidth for all tests was approximately constant at 500 Mbps with occasional excursions above 700 Mbps for all three job batches. This behavior indicates that the file serving capability of both the resident hard drive and the local NFS storage had been saturated. We therefore concluded that it was not possible to test the impact of the xrootd server on the DGT processor for the LAN xrootd configuration with the resources available at the Tier-3 site.

6.1.4. Final observation

It should be noted that it is unlikely that the xrootd file server did overwhelm the DGT processor. The DGT machine has ample resources; in fact, it is tooled with equipment similar to that found in the VMUF machine at the Tier-2, which was able to outperform the direct Lustre connection. Therefore, it is most prudent to conclude, that the major point of failure was the inability for xrootd file server to properly handle the wide area network connection to the underlying storage target. While we have demonstrated that an xrootd server is capable of serving files found across the wide area network, it was not indeed designed to operate that way [citation needed]. Additionally, as we will in the

following section, it was observed that the connection between the Tier-2 Lustre storage and the Tier-3, while very stable, fluctuated widely. It is possible that the LAN xrootd file server had difficulty properly serving files while the connection to the underlying storage experienced a highly uneven transfer rate.

6.2 Volatile transfer rates between the test sites

The results described in Chapter 5.2 were remarkable for the high standard deviation recorded for many of the trials, as **Table 6.1** shows. We only focus on the root Routine experiments, since it was expected that the data transfer rate would be approximately even, as very little computations, and thus CPU cycles, were involved.

The standard deviation for a set of results would often be in the range of 40 – 50% of the average transfer rate, implying a significant fluctuation between measurements. A typical result set is

shown in **Figure 6-1** where a batch of 96 root jobs was measured in the WAN xrootd

	Jobs	*(Mbps) Avg. Rate*	*(Mbps) Std. Dev.*	*Std. Dev. % Of Avg*		*Jobs*	*(Mbps) Avg. Rate*	*(Mbps) Std. Dev.*	*Std. Dev % Of Avg*
Direct Lustre	32	1700	1176	69%	*WAN xrootd Public Worker Nodes*	32	2876	1042	36%
	64	3270	2053	63%		64	2784	1462	53%
	96	5401	1477	27%		96	3258	1248	38%
	128	5817	1555	27%		128	3420	1064	31%
	160	5494	1731	32%		160	3509	869	25%
	192	5546	1696	31%		192	3518	863	25%
LAN xrootd	32	1000	414	41%	*WAN xrootd*	32	1877	568	30%
	64	986	325	33%		64	2702	541	20%
	96	874	297	34%		96	4108	1012	25%
	128	818	307	38%		128	4470	618	14%
	160	827	343	41%		160	4104	1135	28%
	192	838	292	35%		192	4129	1087	26%

Table 6.1: The LAN xrootd configuration has the highest standard deviation of the configurations tested.

configuration that was accessed directly from public-facing worker nodes.

6.2.1. Observing steady-state standard deviation

As **Figure 6-1** clearly indicates, job ramp up and ramp downs can be significant. To eliminate the influence of these events, a new data set was selected using only the intervening measurements. The standard deviation calculations from these sets are shown in **Table 6.2**.

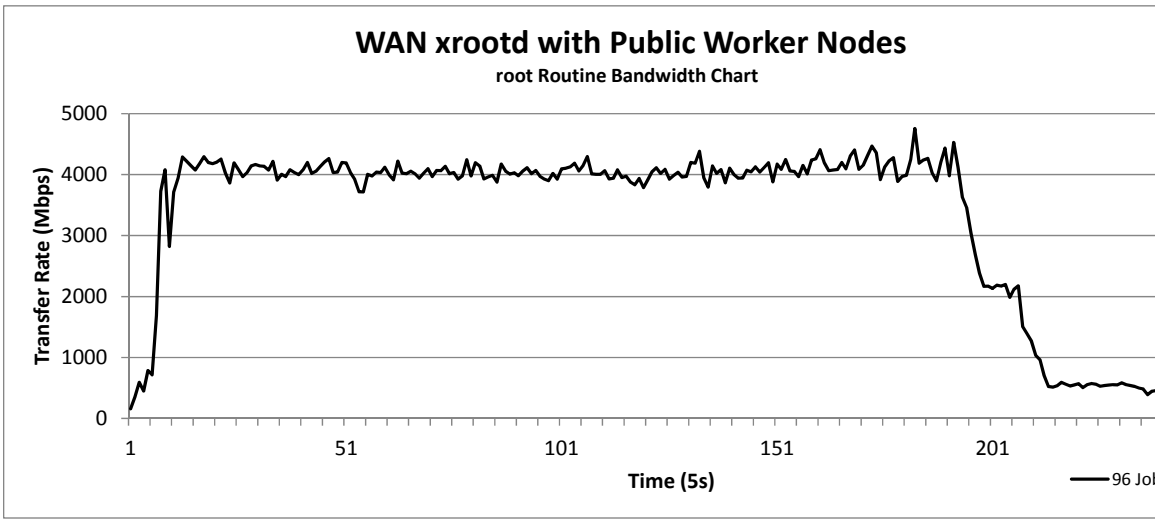

Figure 6-1: A typical job batch collection measurement. Volatility at the ramp-up and ramp-down stage is common

	Jobs	*(Mbps) Avg. Rate*	*(Mbps) Std. Dev.*	*Std. Dev. % Of Avg.*
Direct Lustre	32	2105	1253	60%
	64	2930	2277	78%
	96	5998	788	13%
	128	6376	829	13%
	160	6337	903	14%
	192	6123	1053	17%

	Jobs	*(Mbps) Avg. Rate*	*(Mbps) Std. Dev.*	*Std. Dev. % Of Avg.*
WAN xrootd Public Worker Nodes	32	3195	837	26%
	64	3215	1395	43%
	96	3806	782	21%
	128	3898	275	7%
	160	3922	271	7%
	192	3884	227	6%

LAN xrootd	32	1120	321	29%
	64	947	263	28%
	96	858	270	31%
	128	795	274	35%
	160	766	258	34%
	192	772	261	34%

WAN xrootd	32	2131	221	10%
	64	2875	230	8%
	96	4476	314	7%
	128	4645	308	7%
	160	4702	290	6%
	192	4582	228	5%

Table 6.2: The sliced results significantly reduced the standard deviation for all but the LAN xrootd configuration for large job batches.

Comparing the two data sets we can glean a few interesting observations. First, the WAN xrootd configurations both evidence much lower standard deviation numbers for the original and sliced set. Even more interesting, is that once the subsets were selected, the steady-state operation showed expected standard deviations, mostly under 10% of the average transfer rate.

The direct Lustre connection, if we only focus on job batches of 96 jobs and above, also underwent a significant reduction of in the transfer rate fluctuation. However, comparing that to the WAN xrootd configurations the fluctuation is still approximately double on a trial by trial basis. The decreased volatility implies that having an xrootd server local to the underlying storage has an evening effect on the I/O jitter.

The increased volatility seen in the direct Lustre configuration may also explain why the LAN xrootd configuration suffered from the highest standard deviation and possibly explain its poor performance. This concept was touched upon in Chapter 6.1.4, and we propose that future investigations pay close attention to the effect of bandwidth jitter to the performance of the xrootd file server.

REFERNCES

[1] CMS Collaboration, *LHC computing Grid TDR*, CERN-LHCC-2005-024, no. June. 2005.

[2] R. Kaselis, S. Piperov, N. Magini, J. Flix, O. Gutsche, P. Kreuzer, M. Yang, S. Liu, N. Ratnikova, a Sartirana, D. Bonacorsi, and J. Letts, “CMS Data Transfer operations after the first years of LHC collisions,” *J. Phys. Conf. Ser.*, vol. 396, no. 4, p. 042033, Dec. 2012.

[3] CMS Collaboration, *The Computing Project*, CERN-LHCC-2005-023, vol. CERN-LHCC-. 2005.

[4] D. Bonacorsi, “The CMS Computing Model,” *Nucl. Phys. B - Proc. Suppl.*, vol. 172, pp. 53–56, Oct. 2007.

[5] K. Bloom, “CMS Use of a Data Federation,” in *Conference on Computing in High Energy and Nuclear Physics*, 2013.

[6] J. L. Rodriguez, P. Avery, T. Brody, D. Bourilkov, Y. Fu, B. Kim, C. Prescott, and Y. Wu, “Wide area network access to CMS data using the Lustre TM filesystem,” *J. Phys. Conf. Ser.*, vol. 219, no. 7, p. 072049, Apr. 2010.

[7] “Lustre 2.0 Operations Manual,” 2011.

[8] M. Boon, “ExTENCI – A recipe for science success | iSGTW,” 2011. [Online]. Available: http://www.isgtw.org/feature/extenci-–-recipe-science-success. [Accessed: 25-Dec-2013].

[9] “Kerberos V5 UNIX User ’ s Guide,” 2010.

[10] C. Boeheim, A. Hanushevsky, D. Leith, R. Melen, R. Mount, T. Pulliam, and B. Weeks, “Scalla: Scalable Cluster Architecture for Low Latency Access Using xrootd and olbd Servers,” 2006.

[11] G. Kaganas, J. L. Rodriguez, M. Chen, P. Avery, D. Bourilkov, Y. Fu, and K. Palencia, “Distributing CMS Data between the Florida T2 and T3 Centers using Lustre and Xrootd-fs,” in *J. Phys.: Conf.*, 2014.

[12] S. Bradner and A. Mankin, “The recommendation for the IP next generation protocol,” 1995.

[13] “IPv6 Adoption,” 2014. [Online]. Available: http://www.google.com/intl/en/ipv6/statistics.html. [Accessed: 29-May-1BC].