BENCHMARK ANALYSIS FOR ADVANCED DISTRIBUTED DATA STORAGE FOR HETEROGENEOUS CLUSTERS

Catalin Negru, Florin Pop*, Ciprian Dobre, Valentin Cristea

University POLITEHNICA of Bucharest, Faculty of Automatic Control and Computers, Department of Computer Science
Spl. Independentei, 313, Bucharest 060042, Romania
E-mails: {catalin.negru, florin.pop, ciprian.dobre, valentin.cristea}@cs.pub.ro

KEYWORDS
Data Storage, Distributed Systems, Clusters, Benchmark.

ABSTRACT

The necessity of a Large Scale Distributed Data Storage System offering scalability, reliability, performance, availability, affordability and manageability became a strong requirement for high-level application with multiple user interactions. This paper presents the benchmarking for performance of LUSTRE file system and highlights the results obtained from different test scenarios with IOzone and Intel IMB benchmarks, considering parallel I/O characteristics of Lustre file system. The paper also presents a set of best practices for data integrity security and accessibility and a few techniques for troubleshooting with LUSTRE. The results of the benchmark analysis were used to offer a perspective about the performance of NCIT-Cluster at University Politehnica of Bucharest in I/O and MPI jobs.

INTRODUCTION

The emergence of clustered computers has created a multiplication of scientific, analytic and research data. Many applications such as seismic data processing, financial analysis, computational fluid dynamics, calculations to understand the fundamental nature of matter, including quantum chromo dynamics and condensed matter theory, created growing storage infrastructure challenge as traditional storage systems struggle to keep pace with speed and requirements of this kind of applications.

In this context, in many scientific applications, especially those that use large amount of data exists a gap between processor performance and I/O performance which led to I/O bottlenecks. Parallel file systems represent the solution which in most of the cases solves the bottleneck with I/O problems [1]. Numerous studies have shown that many scientific applications need to access a large number of small pieces of data from file. The I/O performance suffers considerably if applications access data by making many small I/O requests. To improve the parallel I/O performance, the small I/O requests are collected into fewer number of larger size requests. So, storage has become a very important part of clusters and distributed systems and is likely to become even more important as problem sizes grow [16, 17]. File systems such as IBM's GPFS [2], SUN's open source Lustre File System [3] have proven to support concurrent file and file system access across thousands of files and data that are growing up reaching zeta scale.

Lustre represents a leading technology in class of parallel I/O technologies and open source standard for HPC and clusters. Lustre file system is currently used on nearly 1/3 of the world's Top100 fastest computers [4]. MPI-IO represents a parallel I/O interface that allows programs with many processes (like scientific applications) on many nodes to coordinate their I/O read and write and to obtain more efficiency [5].

IOzone is a file system benchmark tool. The benchmark generates and measures a variety of file operations. IOzone has been ported to many machines and runs under many operating systems. IOzone is useful for determining a broad file system analysis of a vendor's computer platform. The benchmark tests file I/O performance for several atomic, parallel and concurrent operations that highlight the performance of a parallel file system [7].

NCIT High Performance Computing Center from University Politehnica of Bucharest includes several research and teaching laboratories in the fields of High Performance Computing, Distributed Systems and Applications, E-Business and e-Government, Artificial Intelligence, Computer Networks. The Center's activity relies on a collaborative virtual environment using high-performance resources and computer-supported cooperative work tools. The solution is flexible, easily adaptable to different activities carried out by the Center, including project development, training, consultancy, technology transfer, etc. The mission of the Center is to promote advanced and inter-disciplinary research, to develop a new paradigm for collaboration among computer scientists, computational scientists and researchers from a diversity of domains, to develop human resources by educational programs [8].

The paper is structured as follow: Section 2 presents the related work in the field of benchmark analysis for distributed data storage. Section 3 presents the proposed model for NCIT cluster and in Section 4 the experimental results. We present the conclusions and future work in Section 5.

* Corresponding Author

RELATED WORK

Large clustered computers provide low-cost compute cycles, and therefore have promoted the development of sophisticated parallel-programming algorithms based on the Message Passing Interface [6]. Chen et al. in [6] evaluated the I/O performance using the IOZONE benchmark on the iSCSI-based TerraGRID parallel filesystem. Their evaluations show that 10GbE, with or without protocol-offload, offered better throughput and latency than IB to socket-based applications. Although protocol-offload in both 10GbE and IB demonstrated significant improvement in I/O performance, large amount of CPU are still being consumed to handle the associated data-copies and interrupts. The emerging RDMA technologies hold promises to remove the remaining CPU overhead. We plan to continue our study to research the applications of RDMA in parallel I/O.

The benchmark could also be used for problem diagnosis in parallel file systems, problem referring to scalability and accessibility. Kasick at al. in [9] focus on automatically diagnosing different performance problems in parallel file systems by identifying, gathering and analyzing OS-level, black-box performance metrics on every node in the cluster. They developed a root-cause analysis procedure that further analyzes the affected metrics to pinpoint the faulty resource (storage or network), and demonstrate that this approach works commonly across stripe-based parallel file systems. Based on that, we tried to identify in this paper and in our approach the lateral effect caused by CPU cache and buffer cache.

Song et al. in [10] demonstrates that the stripe size is a vital performance parameter, but the optimal value for it is often application dependent. How to determine the optimal stripe size is a difficult research problem. Based on the observation that many applications have different data-access clusters in one file, with each cluster having a distinguished data access pattern, in [10] is proposed a segmented data layout scheme for parallel file systems. The basic idea behind the segmented approach is to divide a file logically into segments such that an optimal stripe size can be identified for each segment. We conduct out tests for benchmarks, considering parallel I/O characteristics of Lustre file system and different stripe size for data.

Another important sector that requires distributed data storage refers to server virtualization. Here, the challenge is on profiling physical resource utilization information of VMs when consolidated on a single server. In [11] Lu et al. formulate profiling as a source separation problem as studied in digital signal processing, and design a directed factor graph (DFG) to model the multivariate dependence relationships among different resources (CPU, memory, disk, network) across virtual and physical layers. The methodology outputs estimates of physical resource utilization on individual VMs and physical server aggregate resource utilization. The Xen-virtualization platform was used in order to evaluate the methodology for different consolidation scenarios with diverse applications including RUBiS, IOzone, SysBench, and Netperf.

DATA STORAGE SOLUTION FOR CLUSTERS

In clusters, in general, data is stored on multiple virtual servers, generally hosted by third parties, rather than being hosted on dedicated servers. The center operators, in the background, virtualizes the resources according to the requirements of the customer and expose them as storage pools, which the customers can themselves use to store files or data objects. Physically, the resource may span across multiple servers. Based on cluster storage the Cloud storage began to offer a ‘hot’ new storage technology for different users with necessities. The fundamental challenge facing cloud storage is scalability. Here new multi–terabyte disk drives are the norm, but traditional RAID data protection technologies are lagging due to longer disk rebuild times. Per Bit addresses the challenge of scalable data protection with a new purpose–built, cloud storage solution. In a field as complex as enterprise storage in heterogeneous clusters, building testing mechanisms that accurately reflect real life and provide any real value to end-users is fantastically difficult. With such an incredibly wide range of enterprise storage workloads and products, it’s very hard to build a benchmark that has any hope of resembling all of them

The important criteria for evaluating a data storage solution for clusters are:

- *Manageability*: the ability to manage a system with minimal resources;
- *Access method*: protocol through which cloud storage is exposed;
- *Performance*: performance as measured by bandwidth and latency;
- *Multi-tenancy*: support for multiple users (or tenants);
- *Scalability*: ability to scale to meet higher demands or load in a graceful manner;
- *Data availability*: measure of a system's uptime;
- *Control*: ability to control a system—in particular, to configure for cost, performance, or other characteristics;
- *Storage efficiency*: measure of how efficiently the raw storage is used;
- *Cost*: measure of the cost of the storage (commonly in dollars per gigabyte).

All of these aspects must be considered in concordance with all important levels in cluster storage architecture: network and storage infrastructure, storage management, metadata management, storage overlay and interface service.

Lustre is a storage architecture for clusters (see Figure 1). The central component of the Lustre architecture is the Lustre file system, which is supported on the Linux operating system. Lustre is a parallel file system and is designed to enable I/O performance. Mainly used in High Performance Computing environments, Lustre is also applicable to any enterprise storage environment where very high I/O bandwidth is required. Lustre is an object-based file system. It is composed of three components: Metadata servers (MDSs) object storage servers (OSSs), and clients. Figure 29 presents the Lustre architecture. Lustre uses block devices for file data and metadata storages and each block

device can be managed by only one Lustre service. The total data capacity of the Lustre file system is the sum of all individual OST capacities. Lustre client's access and concurrently use data through the standard POSIX I/O system calls [12].

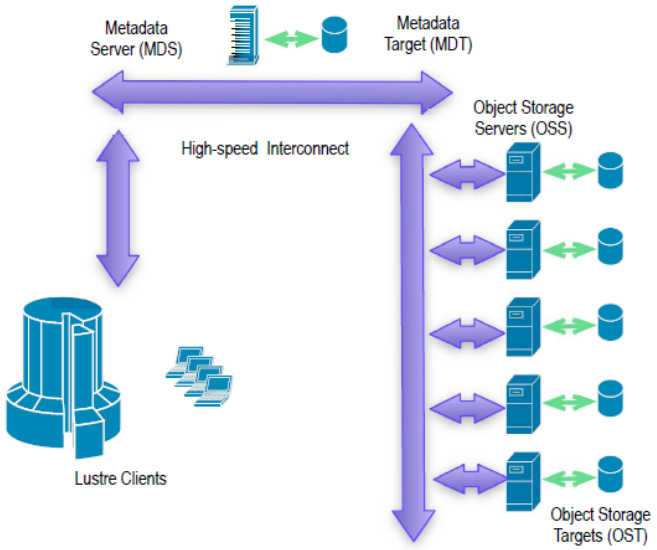

Figure 1. Architecture of a Lustre file systems [12]

The main features of Lustre are: scalability, high-availability high-performance heterogeneous networking, security, access control list (ACL) with extended attributes, interoperability, object-based architecture, byte-granular file and fine-grained metadata locking, controlled striping, disaster recovery tool, internal monitoring and instrumentation interfaces.

We analyze the Lustre file system in the context of NCIT cluster as a support for complex applications. The NCIT cluster has the following characteristics regarding data storage system: the storage system is composed of the following DELL solutions: 2 PowerEgde 2900 and 2 PowerEdge 2950 servers, and 4 PowerVault MD1000 Storage Arrays. There are four types of disk systems you can use local disks, NFS, LustreFS and FibreChannel disks. All home directories are NFS mounted. There are several reasons behind this approach: many profiling tools cannot run over LustreFS because of its locking mechanism and second, if the cluster is shut down, the time to start the Lustre file system is much greater than starting NFS (see Figure 2).

For benchmarking we defined the following scenarios that are important for different case-studies:

- Write: measures the performance of writing a new file.
- Re-write: measures the performance of writing a file that already exists. When a file is written that already exist the work required is less as the metadata already exists.
- Read: measures the performance of reading an existing file.
- Re-Read: measures the performance of reading a file that was recently read.
- Random Read: measures the performance of reading a file with accesses being made to random locations within the file.
- Random Write: measures the performance of writing a file with accesses being made to random locations within the file.
- Random Mix: measures the performance of reading and writing a file with accesses being made to random locations within the file.
- Backwards Read: measures the performance of reading a file backwards.
- Record Rewrite: measures the performance of writing and re-writing a particular spot within a file.
- Fwrite: measures the performance of writing a file using the library function fwrite().
- Fread: measures the performance of reading a file using the library function fread().

The benchmark tests file I/O performance for the scenarios mention before using IOzone tool. We want to estimate the capacity of NCIT cluster in order to support different type of data storage operations, with different stripe size. The benchmark generates and measures a variety of file operations. IOzone has been ported to many machines and runs under many operating systems (we test on Linux OS).

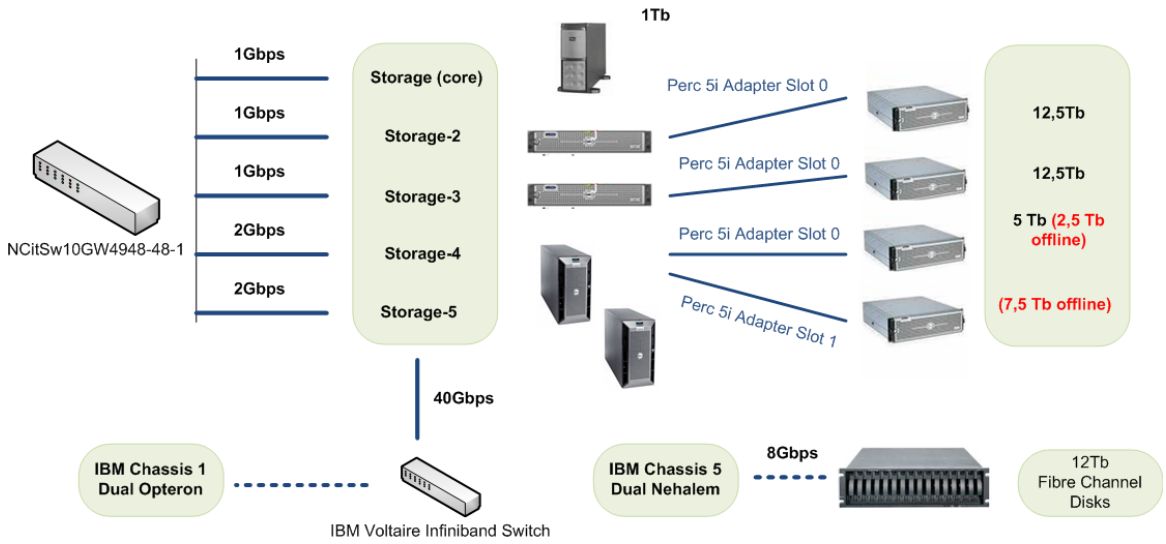

Figure 2. NCIT Data Storage Architecture [13]

EXPERIMENTAL RESULTS

We conduct out test considering all mentioned scenarios in previous section using parallel I/O over Lustre FS. MPI-IO represents a parallel I/O interface that allows programs with many processes on many nodes to coordinate their I/O read and write and to obtain more efficiency [13]. One of the known issues of Lustre in MPI applications is represented by the not aligned I/O on stripe boundaries. One file might be distributed across two stripes which is representing a drawback in the performance of the application. Another problem is represented by large, contiguous writes, can cause significant contention at the network layer.

ROMIO implements the collective I/O operations using a technique termed two-phase I/O. Consider a collective write operation. In the first phase, the processes exchange their individual I/O requests to determine the global request. The processes then use inter-process communication to re-distribute the data to a set of aggregator processes. The data is redistributed such that each aggregator process has a large, contiguous chunk of data that can be written to the file system in a single operation. The parallelism comes from the aggregator processes performing their writes concurrently. This is successful because it is significantly more expensive to write to the file system than it is to perform inter-process communication [14].

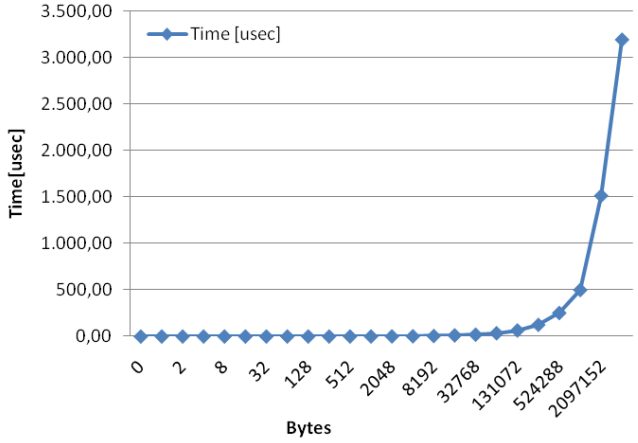

Figure 3. Ping-Pong Benchmark between two processes (Parallel-I/O: MPI ROMIO over Lustre)

Collective IO will apply read-modify-write to deal with non-contiguous data by default. However, it will introduce some overhead (IO operation and locking).

In Figure 3 is presented a Ping-Pong benchmark which passes messages of different size between two processes which run on two machines on Quad queue on NCIT-Cluster at UPB. Can be observed that over 512KB transfer time rise exponential. So, the conclusion with this test shows that the Parallel-I/O paradigm offers performance for HPC collaborative application only for small messages.

In Figure 4 is presented a IOzone test for a 256MB file in throughput mode with 5 active threads for Ping-Pong Benchmark between two processes. The conclusion with this test shows that the Read and Re-read operations are performant for this type of communication, so application that read in a high loop the same set of variables are the good candidate for parallel-I/O over Lustre.

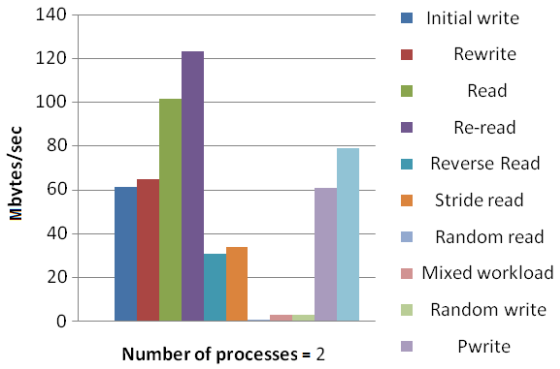

Figure 4. Lustre throughput for Ping-Pong Benchmark

Continuing the experiments, certain offsets have very high latencies. Considering this point, Lustre FS allocate its first indirect block. One can see from the data, the impact of this allocation is translated in latency for different operations (see Figure 5).

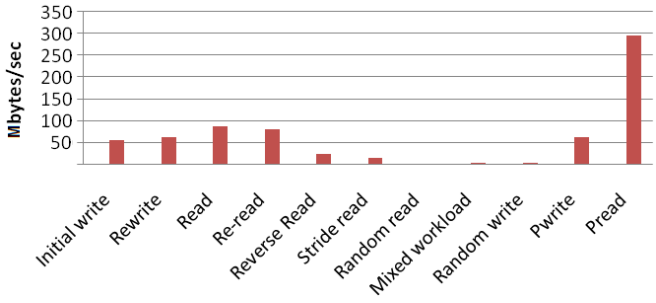

Figure 5. Lustre latency for Ping-Pong Benchmark

Considering the results presented for Read and Re-Read operations, Figure 6 presents a report regarding Read operation considering different file sizes and different stripe sizes. There are some lateral effects that influence the performance of this operation: first is the effect of buffer cache (maintaining the data stored in memory for a while) and the CPU cache effect, storing data for processing.

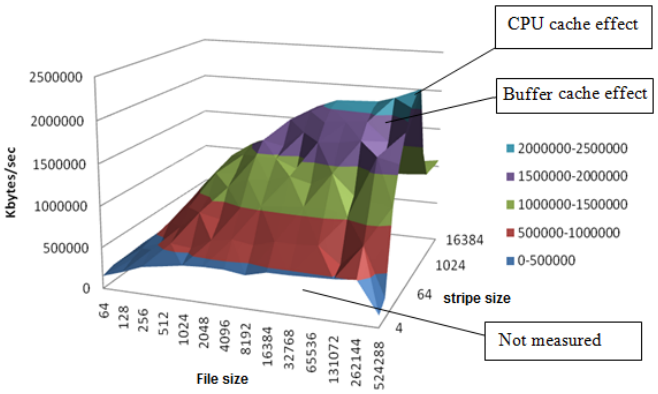

Figure 6. Reader Test report for Ping-Pong Benchmark

A sample of log file for our tests is:

```
Time Resolution = 0.000001 seconds.
Processor cache size set to 1024 Kbytes.
Processor cache line size set to 32 bytes.
File stride size set to 17 * record size.
Throughput test with 5 processes

Each process writes a 262144 Kbyte file in 4
Kbyte records

Children see throughput for 5 initial writers
       =   61570.22 KB/sec
Parent sees throughput for 5 initial writers
       =   37467.55 KB/sec
Min throughput per process
       =    9911.92 KB/sec
Max throughput per process
       =   15581.68 KB/sec
Average throughput per process
       =   12314.04 KB/sec
Min xfer = 167936.00 KB

Children see throughput for 5 rewriters
       =   65089.98 KB/sec
Parent sees throughput for 5 rewriters
       =   61649.19 KB/sec
Min throughput per process
       =   10390.88 KB/sec
Max throughput per process
       =   16787.00 KB/sec
Average throughput per process
       =   13018.00 KB/sec
Min xfer = 151552.00 KB
```

An adequate application that uses at the maximum level the performance of parallel storage in NCIT cluster is Air flow Simulator (Air (2011)), which is a solution that can be used for simulation and visualization of air flow and heat transfer in buildings using existing meshing tools such as SALOME and computational fluid dynamic engines such as Code-Saturne. The developed user-interface and post-processing procedures are also discussed. The paper provides an overview of existing technologies and protocols and shows how these technologies are used in the implementation of the proposed system [15].

CONCLUSIONS

Cluster storage is an important piece of this puzzle called Cloud, and together with cloud computing represents cloud as technology destined for solving for example many distributed applications. Worth saying that without a well optimized storage system much application that runs in cloud or a cluster can have a breakdown in performance.

This paper gives a perspective on performance on a storage system using IOzone tool, and special to characterize NCIT-

CLUSTER and established the correct configuration parameters for different type of applications. I/O performance can suffer considerably if access data pattern is not the right one, especially in a parallel file system like Lustre. One important facility that a storage system based on Lustre file systems can give to the user is ability to set the stripe size and the number of stripes can be placed everywhere, because in this way the user can do better optimization of his application.

Regarding future work can be made a system that automatically makes profiling of the data model in applications that run on cluster or cloud and to suggest measures to improve performance.

ACKNOWLEDGMENTS

The research presented in this paper is supported by national project: "SORMSYS - Resource Management Optimization in Self-Organizing Large Scale Distributes Systems", Contract No. 5/28.07.2010, Project CNCSIS-PN-II-RU-PD ID: 201.
The work has been co-funded by the Sectorial Operational Program Human Resources Development 2007-2013 of the Romanian Ministry of Labor, Family and Social Protection through the Financial Agreement POSDRU/89/1.5/S/62557.

REFERENCES

[1] Huaiming Song, Yanlong Yin, Xian-He Sun, Rajeev Thakur, and Samuel Lang. 2011. Server-side I/O coordination for parallel file systems. In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis* (SC '11). ACM, New York, NY, USA, Article 17 , 11 pages.

[2] Frank Schmuck and Roger Haskin. 2002. GPFS: a shared-disk file system for large computing clusters. In *Proceedings of the 1st USENIX conference on File and storage technologies*(FAST'02). USENIX Association, Berkeley, CA, USA, 16-16.

[3] Lustre file system, High Performance and Scalability, Web site: www.lustre.org, Accessed on February 2012.

[4] Niklas Edmundsson, Erik Elmroth, Bo Kagstrom, Markus Martensson, Mats Nylen, Ake Sandgren, and Mattias Wadenstein. 2004. Design and evaluation of a TOP100 Linux Super Cluster system: Research Articles. *Concurr. Comput. : Pract. Exper.* 16, 8 (July 2004), 735-750.

[5] Jean-Pierre Prost, Richard Treumann, Richard Hedges, Bin Jia, and Alice Koniges. 2001. MPI-IO/GPFS, an optimized implementation of MPI-IO on top of GPFS. In *Proceedings of the 2001 ACM/IEEE conference on Supercomputing (CDROM)* (Supercomputing '01). ACM, New York, NY, USA, 17-17.

[6] H. Chen, J. Decker, and N. Bierbaum. 2006. Future networking for scalable I/O. In *Proceedings of the 24th IASTED international conference on Parallel and distributed computing and networks*(PDCN'06), T. Fahringer (Ed.). ACTA Press, Anaheim, CA, USA, 128-135.

[7] IOzone Filesystem Benchmark, http://www.iozone.org/ - web page of the project, Accessed on February 2012.

[8] NCIT High Performance Computing Center Homepage, Web site: http://cluster.grid.pub.ro/, Accessed on February 2012.

[9] Michael P. Kasick, Jiaqi Tan, Rajeev Gandhi, and Priya Narasimhan. 2010. Black-box problem diagnosis in parallel file systems. In *Proceedings of the 8th USENIX conference on File and storage technologies* (FAST'10). USENIX Association, Berkeley, CA, USA, 4-4.

[10] Huaiming Song, Yanlong Yin, Xian-He Sun, Rajeev Thakur, and Samuel Lang. 2011. A Segment-Level Adaptive Data Layout Scheme for Improved Load Balance in Parallel File Systems. In *Proceedings of the 2011 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing* (CCGRID '11). IEEE Computer Society, Washington, DC, USA, 414-423.

[11] Lei Lu, Hui Zhang, Guofei Jiang, Haifeng Chen, Kenji Yoshihira, and Evgenia Smirni. 2011. Untangling mixed information to calibrate resource utilization in virtual machines. In*Proceedings of the 8th ACM international conference on Autonomic computing* (ICAC '11). ACM, New York, NY, USA, 151-160.

[12] Understanding Lustre, Lustre 2.0 Operations Manual (821-2076-10), Chapter 1, ©2011, Oracle and/or its affiliates, http://wiki.lustre.org/manual/LustreManual20_HTML/UnderstandingLustre.html, Accessed on February 2012.

[13] Jean-Pierre Prost, Richard Treumann, Richard Hedges, Bin Jia, and Alice Koniges. 2001. MPI-IO/GPFS, an optimized implementation of MPI-IO on top of GPFS. In *Proceedings of the 2001 ACM/IEEE conference on Supercomputing (CDROM)* (Supercomputing '01). ACM, New York, NY, USA, 17-17.

[14] Rajeev Thakur, William Gropp, and Ewing Lusk. 1999. Data Sieving and Collective I/O in ROMIO. In *Proceedings of the The 7th Symposium on the Frontiers of Massively Parallel Computation*(FRONTIERS '99). IEEE Computer Society, Washington, DC, USA, 182-.

[15] Alexandru Stroe, Emil Slusanschi, Ana Stroe, Simona Posea, Alexandru Herisanu, *Airflow Simulator Heat Transfer Computer Simulations of the NCIT-Cluster Datacenter*, The 18th International Conference on Control System and Computer Science, 2011, pp: 569-575

[16] Devarshi Ghoshal, Richard Shane Canon, and Lavanya Ramakrishnan. 2011. I/O performance of virtualized cloud environments. In *Proceedings of the second international workshop on Data intensive computing in the clouds* (DataCloud-SC '11). ACM, New York, NY, USA, 71-80.

[17] Julian Borrill, Leonid Oliker, John Shalf, and Hongzhang Shan. 2007. Investigation of leading HPC I/O performance using a scientific-application derived benchmark. In *Proceedings of the 2007 ACM/IEEE conference on Supercomputing* (SC '07). ACM, New York, NY, USA, , Article 10 , 12 pages.