Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 4, Issue. 12, December 2015, pg.67 – 73

Evaluation of Active Storage System Realized Through Hadoop

Naveenkumar J[1], Prof. Dr. S.D. Joshi[2]

[1][2]Computer Engineering Department, BVDUCOEP, Maharashtra, Pune

[1] pro_naveen@hotmail.com; [2] sdj@live.in

Abstract— With the advent of big data and third platform application technologies, enterprise class applications are going through rapid transformations. These enterprise class applications are expected to deliver actionable business intelligence as against presenting volume of information drawn from raw data. As these enterprise applications move up in the value chain, it become more and more data intensive thereby increasing the volume of data to be moved between the hosts and the data storage sub-systems. However, the storage IO sub systems delayed in advancement in consistency with the enhancement of processors. The memory coerces the system to deliver huge amount of data for execution of the data powered applications due to bandwidth limitation, excessive power consumption, network congestions and redundant copies of data in the end-to-end application stack. This bottleneck in the IO path negatively affects end-to-end application performance.

Keywords— Hadoop, Active Storage, virtualisation, Embedded Storage, Performance, Benchmarking

I. INTRODUCTION

The three IT platforms vary in the scope of usage, the scalability of applications and services and use of these applications and services leveraged by the technologies. These technologies are empowered by considerate to what extent users are accessing the application/services and how many users are using it, which in turn decides the platform. Storage systems' work pattern is same as used to be in first and second platform. [1], [2]. The storage systems are traditionally scaled up systems and monolithic by virtue of design. The current ongoing digital transformation is significantly hampered by the storage system development trend. The reason behind this is due to the struggle by storage systems to keep up with the increasing storage demand characterized by the proliferating data, needed performance capability and the complexity in data-storage management. [3], [4]

The exponential growth of data thrusts on increased need for storage capacity, more network bandwidth and fast increasing numbers of compute capability resource. Due to the characteristics of user's location and business model, data can be accessed or stored from any location across the world. Storage system that is scattered and further requires additional efforts to manage and organize the scattered data. [5], [6]

Since third platform is strongly based on these four pillars – big data, Cloud, mobile and social network, the application types on these four different technologies vary hugely. The workloads presented by these applications, based on various technologies are completely different and accesses the storage systems differently.

II. RESEARCH OVERVIEW

Storage systems' work pattern is same as used to be in first and second platform. The storage systems are traditionally scaled up systems and monolithic by virtue of design. The current ongoing digital transformation is significantly hampered by the storage system development trend. The reason behind this is due to the struggle by storage systems to keep up with the increasing storage demand characterized by the proliferating data, needed performance capability and the complexity in data-storage management.

The exponential growth of data thrusts on increased need for storage capacity, more network bandwidth and fast increasing numbers of compute capability resource. Due to the characteristics of user's location and business model, data can be accessed or stored from any location across the world. Storage system that is scattered and further requires additional efforts to manage and organize the scattered data.

Since third platform is strongly based on these four pillars – big data, Cloud, mobile and social network, the application types on these four different technologies vary hugely. The workloads presented by these applications, based on various technologies are completely different and accesses the storage systems differently. In the context of research problem, there is a need to understand the configurations of the Storage Array components and behaviour of applications running on it.

Existing Storage system is an example of ad-hoc, proprietary based approach to realize a form of active storage, whereas, approach to embed a guest operating system is the other spectrum of Active Storage enabler.[7]–[9]. While, what is desirable is something like Hadoop ecosystem, which fits in as midway between these two, the Hadoop based ecosystem allows offloading of compute/data storage IO intensive IO tasks among the data servers in the Hadoop cluster.

Active storage is the system, which takes advantages of underutilized processing power of computing units in storage nodes instead of simply storing data. Employing the storage nodes to process the data along with storing will result in drastic reduction in redundant data transfers over the network, which spectacularly boosts the performance. [10], [11]

The above observation made forms the prime foundation to the research problem primarily focusing on the assumption that 'cost of hosting the applications would offset, the effective (aggregate) overhead involved in driving the end-to-end data transfer between the array and the applications that consume the data.'

III. RELATED WORK

The Literature reviewed suggests various issues and challenges raised due to increase in the quantity of data and the distributed pattern of data. The scattering of data and its size put forth a climacteric challenge at the level of I/O devices. The various data types from various sources are processed by the compute and these data are stored at storage nodes, which are de-coupled, in traditional distributed system. Majority of the storage systems have spinning disks rather than SSD's and flash drives. [8],[12]. The read write operations performed by the application at I/O device level are slower and the I/O stack constitutes multiple layers, which adds on delay in the application processing. [13], [14]

Data proliferation section of this chapter discusses about how the data is exponentially springing up and what kind of challenges are brought along with it, which may affect the application performance. The overall solution by various authors proposed to build up a kind of architecture, which can sustain for both ends like for storing of growing data and at the same time providing suitable environment for running the applications on it by moving the application closer to data. The architecture proposed was based on various hardware enhancements and file system tweaks. The virtualization benefits in increasing the media capacity. By using virtualization, the resources are virtualized and shared among various applications and data. Virtualizing storage node would indeed help store striped data in a single storage node. [15]–[17]

The literature survey revealed that the authors who have worked on bringing the compute part closer to data termed it as Active storage. The active storage is superset, which actually works well according to the study and research carried out earlier. The active storage was initiated and realized through the concept of active disks, which is discussed. [18]–[23][24]. The active storage concept revolves around how to migrate or offload the compute closer to data and execute the offloaded application on storage node. It leads to create a storage system, which has processing capacity along with storing capacity. The literature survey also reveals that not all applications are suitable to adapt to this working environment of active storage.

IV. PROPOSED EVALUATION TESTBED

A. *Hadoop*

Hadoop is a framework, which supports most of the distributed computing platform in java. Hadoop is designed, developed and maintained by apache software foundation. Hadoop is the java framework, which is used for processing the huge datasets. It provides a platform for designing and deploying the distributed systems over it that supports storing of massive datasets and processing it for results derivation. Hadoop encapsulates both the storage and compute together that are very closely bound together. [25]–[27]

There are two key technologies that allow users of Hadoop to successfully retain and analyse data: HDFS and MapReduce.

Hadoop is equipped with a computational model termed MapReduce, where the application is developed using this programming model deployed on Hadoop cluster. The user defined functions are exposed in terms of map () and reduce (). The application is distributed into small modules of tasks when deployed in the Hadoop cluster. These tasks may be controlled and executed on assigned nodes in the cluster. For storing the data on the nodes in the cluster, a distributed file system is afforded by Hadoop, which helps in fetching the compute closer to the data. This enables in increasing the cumulative bandwidth in the cluster. The Hadoop by virtue of its design is fault tolerant. If any node fails in the framework, it is automatically managed by the Hadoop.

B. *Why Hadoop*

Hadoop technology has emerged in a big way and very much apt for confronting the problems raised from big data applications. Essentially, Hadoop architecture transfers the implementation closer to data for processing, rather than transferring the data to host for processing. Hadoop is developed and implemented using java. It taps the potential of enhancements in commodity hardware to scale as the deployment model requires. IT Industry can scale the Hadoop cluster elastically as it is required for business needs.

V. DEPLOYMENT MODEL

Testbed 2a: The complete workflow of the Hadoop is as it is used in both the deployments, which is explained in the next section. In this testbed 2a as shown below in the Fig. 1, the job tracker and name node are integrated with the application i.e. client node. These together are regarded as the master node and all these components are hosted on application host node.
The storage node hosts the data node and the task tracker. Thus, the communication between the job tracker and task tracker happens through the network. There were two slave nodes and one master node in this deployment.

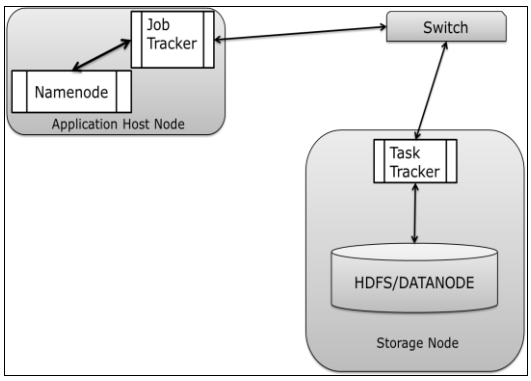

Fig. 1 Testbed 2a: Traditional way of Hadoop Deployment

Testbed 2b: In this testbed 2b as shown in Fig. 2, the master node and slave node are kept separate but the only difference is that both these master and slave are hosted by same storage node. The storage is virtualized and three virtual machines are hosted with two virtual machine holding the master and the other one having the slave nodes. Here master node includes the job tracker and namenode. The slave constitutes datanode and task tracker. In the below Fig. 2 the FA & DA corresponds to processors on disk side and host connecting ports.

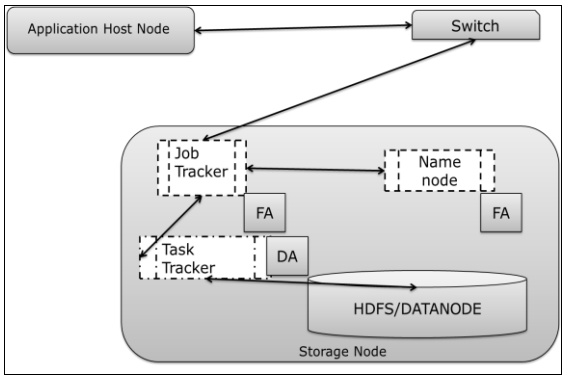

Fig. 2: testbed 2b: Active storage realized using the Hadoop

A. *Hadoop Workflow*

The application programming interface of the application is regarded as the 'Client'. This client always initiates its communication starting with the namenode. The namenode contains the metadata and other details of the datanode. Once the metadata is received from the namenode, the client directly interacts with the datanode for further processing of the data. If the processing pattern is described as a map reduce function, then the client hands over the method to job tracker, which further queues the method for further handling.

Namenode is responsible for storing and maintaining all the information about the underlying distributed file systems like information regarding control, access privilege, block locations, current blocks, mapping between files and blocks. The block locations keep on changing because of the datanode failure or new datanode being added to the cluster because of this, these locations are not stored permanently in the namenode. All the file namespace methods execution is taken care by the namenode.

The file namespace methods including operations on files as well as directories like open, close, rename, delete block, create block and replicate block. The namenode instructs the client regarding the datanode to which it should refer or connect. The instruction depends upon the resource availability and the size of data as input.

Datanode is the storage node that stores the data in its local HDFS. It is the virtue of design of HDFS, which makes to store massive amount of data sets by splitting it into number of blocks that are stored across the datanodes in the cluster. The replication also happens across the datanode in the cluster.

This node also facilitates the execution of operations from the client within the node. The HDFS does not have the information regarding the files in it. When a datanode is started, it scans whole of the local file system and generates a list of data blocks, which are linked to the various files. This list is shared with the namenode.

The above paragraphs discuss about the distributed file system of the Hadoop. The file system also follows the server – client architecture. The namenode is the master node and the data node is the slave. As state earlier by virtue of the Hadoop's design, it fuses the compute and storage together bringing the data processing closer to the data.

The storage component of the Hadoop has been discussed, in the next paragraphs, compute components of the Hadoop are explained.

As name node is master when considering storage on similar terms Job tracker is master when compute is considered in Hadoop. The compute engine is driven by MapReduce programming model. This job tracker is made to run on separate node. If the job tracker fails, then whole compute facet of Hadoop is brought down without affecting the file system. The processing of all jobs and tasks spawned across the cluster is stopped and need to be rebuilt. This job tracker node is the single point of failure of Hadoop processing structure.

The tasktracker is run on separate java virtual machine in each node to process the data. This spawning of task tracker on different java virtual machine, which protects the failure of the tasktracker from failing of some tasks. The jobtracker assigns the tasktracker with the MapReduce job that is based on the tasktracker configuration, which includes the number of slots that indicates the number of tasks that can be accepted by it. [28]–[30]

The tasktracker reports jobtracker through heartbeat message regarding the status of tasks and node also to check the job tracker is alive. Similarly, the job tracker confirms itself regarding available slots and status of that task tracker through heartbeat message from task tracker.

Map reduce is the programming model that is heart of compute facet of Hadoop. The MapReduce has two phases the developer or user is given full freedom to develop and deploy these phases. Out of all phases in the MapReduce programming model, the user can write or custom build methods and operations for map () and reduce (). The map () libraries help in processing the key-value pair dataset and result in an intermediate key-value pair. This is further passed on to the reduce phase.

The reduce () along with libraries will accept the group of keys along with its respective values for each individual keys. As per the operations stated under the reduce () this phase will further reduce the results sets into small size result set by merging the values received from map ().

VI. BENCHMARKING TOOL

The terasort suite comprises three components teragen, terasort and teravalidate. It sorts a huge number of records sizing 100 bytes. This suite simulates the real time Hadoop application workloads. This suite stress on storage I/O, the compute part i.e. stressing the map reduce model and the data transfer between the master and slave nodes.

The teragen is the reason for creating the data for sorting. The random data is written to HDFS local file system directly. This write operation consumes some processor cycle when the data is being generated. Only Map () operation is used to write the data.

The terasort exhibits the sorting operation of the randomly generated data by teragen. Terasort starts reading the data written by the teragen from the local file system, perform the sort operation on the read data and then writes back the sorted data to the local file system. The data are written in partitioned files.

The sorting is done in the correct way and the result of sorting is right that is evaluated and checked by teravalidate. Teravalidate reads all the sorted data from the local file system and the map tasks evaluates all the partitioned files to verify. The reduce phases checks the flow of received partition record.

VII. EVALUATION & RESULT DISCUSSION

Processor in Storage: Observing the processor utilization curve in the Testbed 2b shown in Fig. 3, it shows the high utilization of CPU nearly 96% because of the cycles spent in the virtualization, consuming the bandwidth of the fabric bus and the cycles spent in executing the map and reduce phase. The maximum processor utilization comes at the time when terasort is being executed since it uses the map and shuffle tasks.

Comparing the testbeds 2a & 2b, the processor utilization for Testbed 2a is the measurement of one of the task tracker and datanode system. The Testbed 2b represents the processor utilization of the whole storage node considering all the virtual machine hosted on that storage node. The graphs related to processor utilization is represented in Fig. 3

As concluding from the test setup of Hadoop and evaluating both the testbeds, it is observed from the simulation that, when the job tracker and task tracker are placed within a single storage node sharing the disks among the guest operating system then there is no data transferred through network since all the communication and data exchange happens through the fabric itself. Thus, the results of the network throughput measured are very less in the testbed 2b results. In testbed 2a the network usage is high. All the measurement values are in units of kilobytes per second (KBPS).

Throughput Analysis: The network is high because the client and job tracker were installed on application host node and the tasktracker is installed on the storage node. Thus, the interaction and data exchange between these nodes are high.

The throughput bar chart shown in below Fig. 4 is the network throughput between the client/job tracker and the task tracker in Testbed 2a. Similarly, between the client and Job/task trackers is shown in the testbed 2b. The drastic reduce in the Testbed 2b is because of the Virtual machine hosted on same node sharing the two disks among four virtual machine. The processors of the same node are assigned to the virtual machine and internal disk access is local and only some results and statics are being forwarded to the client machine where sar command was used to find the network statics with this virtualized Hadoop node. Since the job tracker and task trackers were on the same hardware and were communicating using the fabric bus within the same node, there is no transfer of data from the virtualized storage node to application host node. Hence, the time spent by the storage resources is only for executing the three components of the terasort suite.

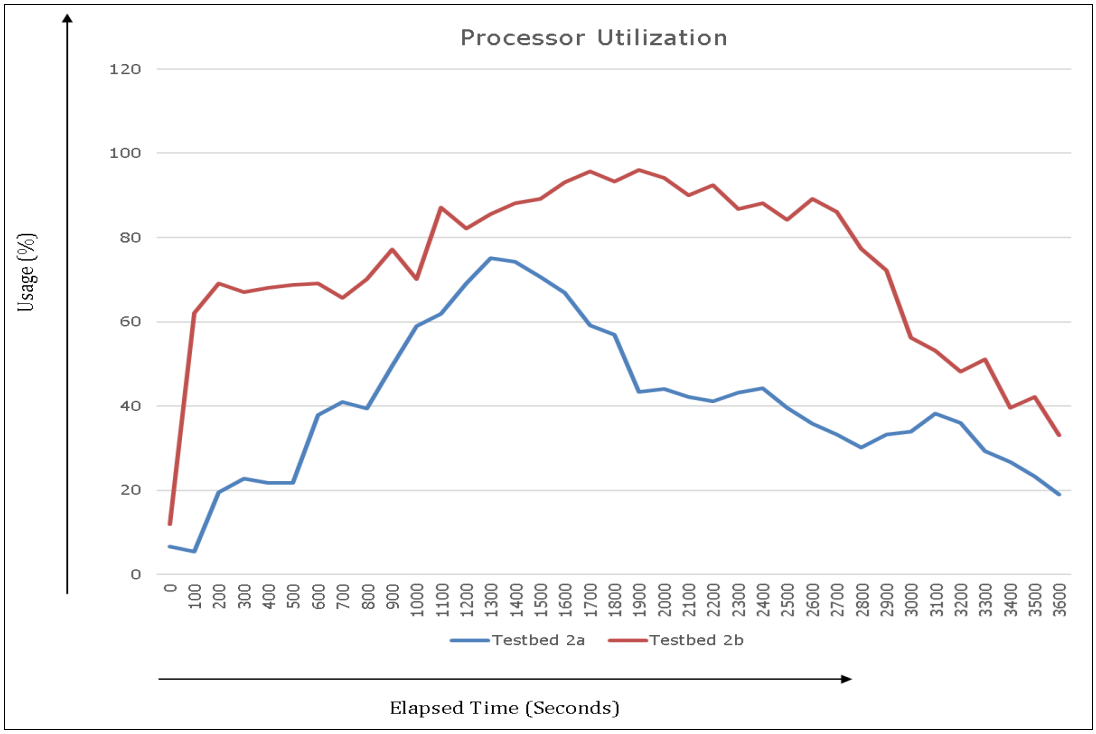

Fig. 3: Storage Processor Utilization Comparison

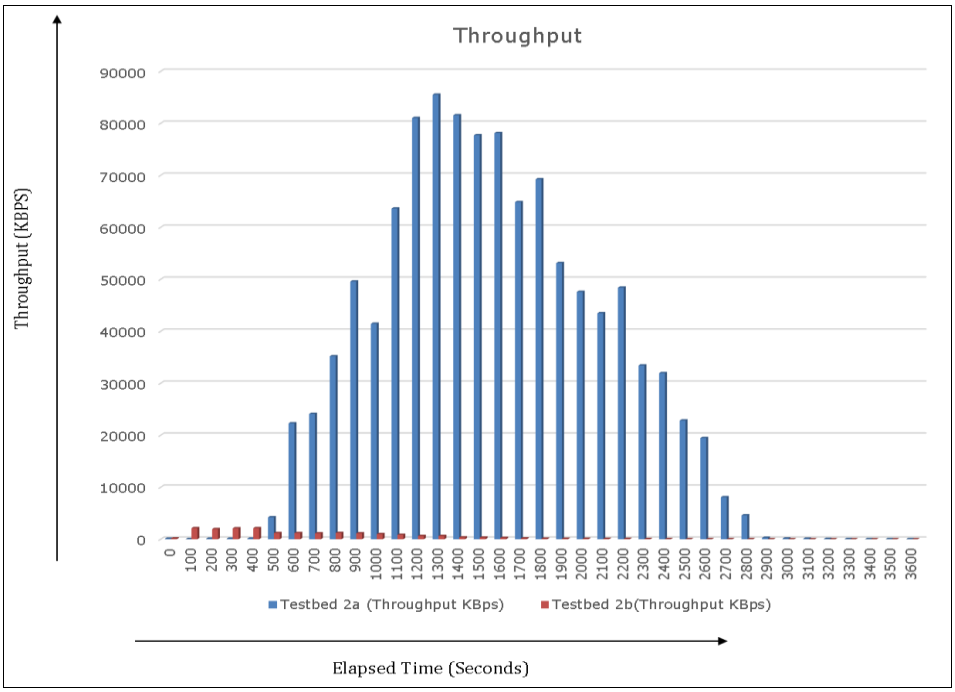

Fig. 4: Throughput Comparison

Elapsed Time Comparison: When considering the elapsed time, the testbed 2a shown elapsed time is measured at the storage host node, which is measured for individual execution of each component of the terasort suit. Similarly, for the Testbed 2b the three components are measured at storage node. In case of CPU utilization, the underutilized power of the compute resource seems to nearly being utilized to its maximum extent.

Considering the elapsed time bar char shown below in fig. 5, it is observed that the virtualization overhead is present in the testbed 2b. However, these virtualizations overhead are not very large. The difference in elapsed time of testbed 2a and Testbed 2b does not seem to be very high but Testbed 2b elapsed time is less than that of the other. Since there was three Virtual Machine in Testbed 2b, the efficiency was improved. Teragen generates the random data but write the data sequentially without any read. From the curve terasort is performed faster in the configuration with respect to other component. The speedup is due to the efficient utilization of the buffer cache and the proper synchronization between map tasks and shuffle operations. The shuffle operations are made to start early as soon as the output of the map tasks is spilled to disks.

The elapsed time seen above in the Testbed 2a and Testbed 2b do not show much a difference where the total elapsed time for completing all the three execution is low in Testbed 2b. If the response time will have been measured at the client node, then the Testbed 2a will have huge response time compared to Testbed 2b. The number 1,2, and 3 in the above graph represents the teragen, terasort and teravalidate respectively.

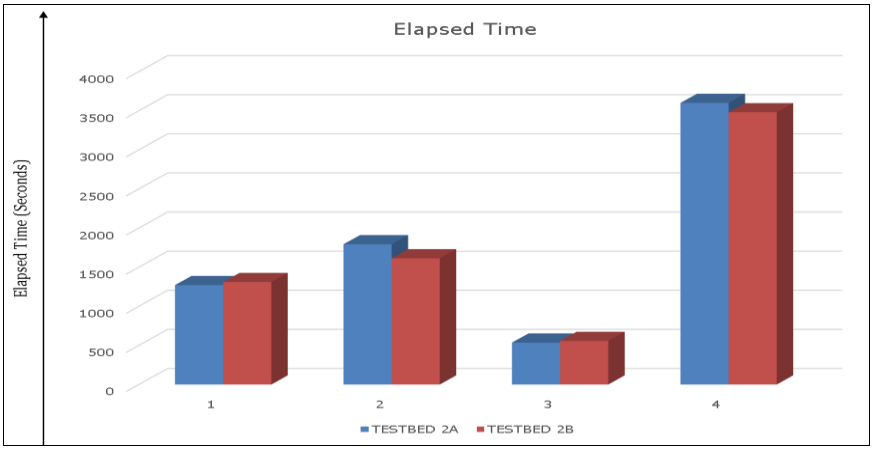

Figure 5: Elapsed Time at storage node

VIII. CONCLUSIONS

The observation from the graphs shown in all the testbed (2b) setup shows that the cost of hosting the application inside the storage varies with the type of application and workload profile but reduces the data transfer drastically and increases the performance of the application by improving the latency. While on the other hand, the performance of the traditional storage testbed (2a) has performed very poor compared to the proposed model testbeds.

REFERENCES

[1] A. Mcafee, "The third platform," 2013.
[2] F. Gens, "The 3rd Platform: Enabling Digital Transformation," *Idc*, no. November, pp. 1–13, 2013.
[3] A. Konary and R. P. Mahowald, "The Coming of the 3rd Platform and What This Means for Software Business Models," no. May, 2013.
[4] A. D. Technical and W. Paper, "Array Tuning Best Practices."
[5] J. Dijcks, "Oracle: Big data for the enterprise," 2012.
[6] B. J. Gantz, D. Reinsel, and B. D. Shadows, "Big Data , Bigger Digital Shadow s , and Biggest Grow th in the Far East Executive Summary: A Universe of Opportunities and Challenges," *Idc*, vol. 2007, no. December 2012, pp. 1–16, 2012.
[7] H. L. H. Lim, V. Kapoor, C. Wighe, and D. H.-C. Du, "Active Disk File System : A Distributed, Scalable File System," *2001 Eighteenth IEEE Symp. Mass Storage Syst. Technol.*, pp. 101–116, 2001.
[8] M. T. Runde, "Active Storage and SSD Caching in an Object Storage Environment Active Storage and SSD Caching in an Object Storage," 2012.
[9] S. V. Anastasiadis, R. G. Wickremesinghe, and J. S. Chase, "Lerna: An active storage framework for flexible data access and management," *Proc. IEEE Int. Symp. High Perform. Distrib. Comput.*, pp. 176–187, 2005.
[10] T. M. John, A. T. Ramani, and J. a. Chandy, "Active storage using object-based devices," *Proc. - IEEE Int. Conf. Clust. Comput. ICCC*, pp. 472–478, 2008.
[11] J. Piernas, J. Nieplocha, and E. J. Felix, "Evaluation of active storage strategies for the lustre parallel file system," *Proc. 2007 ACM/IEEE Conf. Supercomput. (SC '07)*, no. 1, 2007.
[12] S. Cho, C. Park, H. Oh, S. Kim, Y. Yi, and G. Ganger, "Active disk meets flash: a case for intelligent SSDs.," *ICS '13 Proc. 27th Int. ACM Conf. Supercomput.*, pp. 91–102, 2013.
[13] L. Cherkasova and R. Gardner, "Measuring CPU overhead for I/O processing in the Xen virtual machine monitor," *Proc. USENIX Annu. …*, pp. 387–390, 2005.
[14] P. Study, "Storage I / O Performance on VMware vSphere 5 . 1 over 16 Gigabit Fibre Channel," pp. 1–8.
[15] J. G. Elerath and M. Pecht, "Enhanced reliability modeling of RAID storage systems," *Proc. Int. Conf. Dependable Syst. Networks*, pp. 175–184, 2007.
[16] A. N. Uwaechia and O. Akinsanmi, "Reliability Assessment on the Performance Model of Ahmadu Bello University Data Network Repositoriesfor Storage Area Network Design," vol. 2, no. 7, pp. 3311–3315, 2013.
[17] Y. Wang and D. Kaeli, "Profile-Guided I / O Partitioning," 2003.
[18] M. T. Runde, W. G. Stevens, P. a. Wortman, and J. a. Chandy, "An active storage framework for object storage devices," *IEEE Symp. Mass Storage Syst. Technol.*, 2012.
[19] J. a Chandy, "Active Storage Networks," *Development*, 2006.
[20] E. Riedel and D. Nagle, "Active Disks - Remote Execution for Network-Attached Storage Thesis Committee :," *Science (80-.).*, no. December, 1999.
[21] G. Chockler and D. Malkhi, "Active disk paxos with infinitely many processes," *Distrib. Comput.*, vol. 18, pp. 73–84, 2005.
[22] Z. Tan, Y. Yuan, T. Zhan, and Y. Xie, "MO-AOBS: Researches of method object in active object-based storage systems," in *Proceedings of 2011 International Conference on Computer Science and Network Technology, ICCSNT 2011*, 2011, vol. 2, pp. 1175–1180.
[23] J. Naveenkumar, R. Makwana, P. S. D. Joshi, and P. D. M. Thakore, "Performance Impact Analysis of Application Implemented on Active Storage Framework International Journal of Advanced Research in Performance Impact Analysis of Application Implemented on Active Storage Framework," *Int. J. Adv. Res. Comput. Sci. Softw. engiinerring*, vol. 5, no. 2, pp. 1–6, 2015.
[24] A. Acharya, M. Uysal, and J. Saltz, "Active disks: programming model, algorithms and evaluation," *SIGPLAN Not.*, vol. 33, pp. 81–91, 1998.
[25] P. Study, "Virtualized Hadoop Performance with VMware," pp. 1–20.
[26] Sun Microsystems Inc., "Using Lustre with Apache Hadoop Overview and Issues with Hadoop + HDFS," *System*, pp. 1–25, 2010.
[27] M. Hadoop, H. Fs, T. Mapr, and M. Hadoop, "MapR Overview."
[28] H. S. Brief, "What is Hadoop ? Why virtualize Hadoop nodes ? How will Hadoop Nutanix & Hadoop = Enterprise Grade Big Data."
[29] B. Hedlund, "Understanding Hadoop Clusters and the Network," *Stud. Data Cent. Networking, …*, 2010.
[30] J. Buell, "A Benchmarking Case Study of Virtualized Hadoop Performance on VMware vSphere 5," *Tech. white Pap. VMware, Inc*, 2011.

