Performance Optimization of Hadoop Cluster Using Linux Services

Hameeza Ahmed[*], Muhammad Ali Ismail[†], Muhammad Faraz Hyder[‡]

Department of Computer & Information Systems Engineering, NED University of Engineering & Technology
University Road, Karachi-75270, Pakistan[*†‡]

Email: [*]hameeza@neduet.edu.pk, [†]maismail@neduet.edu.pk, [‡]farazh@neduet.edu.pk

CS

Abstract— **Hadoop is an open source tool. It enables the processing and distributed storage of big data sets using commodity cluster computing. With Hadoop occupying a core status in the current processing era, its performance optimization is also being heavily studied. This paper introduces one such method to improve Hadoop cluster performance by using a Remote Procedure Call (RPC), rpcbind service of the Linux system. The comparison is done by executing multiple Hadoop benchmarks on a configured multi-node Hadoop cluster. The final outcome turns in rpcbind favor depicting how the service improves the cluster performance by reducing the elapsed time of the benchmark executed.**

I. Introduction

With the current advances in technology, now data is being continuously accumulated from daily life including phones, credit cards, social networks, sensor equipped areas, trains, buses, RFID, medical records, biological, astronomy, atmospheric science, genomics, military surveillance, photography archives, video archives, large-scale eCommerce, and the internet of things [1]. So it is getting difficult to process such huge quantity of data by existing tools as this involves high volume, velocity, and variety (3 V's). Big data is not an individual technology, but it is a combination of many existing technologies such as parallel processing, virtualization, distributed file systems, in-memory databases and many more. Apache Hadoop is amongst the best open source technologies, which offers high performance processing of big data applications [2, 3]. It enables the data processing in an efficient, cost effective and timely manner. The Apache Hadoop uses programming models to process the large data sets across clusters in a distributed manner. Hadoop makes both the storage and processing feasible on inexpensive, industry-standard servers rather than relying on expensive, proprietary hardware. It also allows the scale up provision from single servers to thousands of machines to form a cluster environment to process the tasks in parallel. Hadoop framework contains two main components MapReduce and HDFS. MapReduce is a programming paradigm that is used for processing massive data sets across large clusters in a parallel distributed manner. Hadoop Distributed File System (HDFS), enables storage of high volume of data in a reliable fashion. It allows the data transmission at high bandwidth to user applications [4].

A number of Hadoop machines can be grouped together to design a Hadoop cluster which results in scale up of both the computation and storage capabilities. The Hadoop cluster stores and analyzes huge amount of unstructured data in a distributed atmosphere. The cluster performance can be improved by tuning and adjusting various operating system parameters, HDFS block size, and Hadoop configuration settings etc [5].

A Linux service is a daemon program that runs in the background carrying out vital tasks. The clients provide information to services, and the services perform the actions accordingly. Clients also make the services to listen for network connections. There are number of services in Linux, but among those the rpcbind service is a server service that is responsible to convert remote procedure calls (RPC) into universal addresses. A host is able to make RPC calls on a server by enabling this service. Additionally, a client is able to make an RPC call to a given program number by establishing a contact with rpcbind on the server machine in order to determine the address for sending the RPC requests. It is a popular interprocess communication technique in distributed systems. It is powerful, simple, and flexible paradigm for constructing client-server based, and distributed applications.

This paper mainly focuses on performance optimization of a Hadoop cluster using rpcbind service of Linux operating system. The rpcbind service is required to be configured first After the Hadoop cluster and rpcbind configuration, various MapReduce benchmarks are executed on the cluster. Comparison is made on the impacts caused by the service optimization using elapsed time as the metric.

Rest of the paper is as follows: Section 2 discusses related work. A brief introduction of Hadoop framework is provided in section 3. Configuration of a multi-node Hadoop cluster is shown in section 4. Section 5 provides description about Hadoop benchmarks, performance metrics and Linux rpcbind service. Performance results are presented in section 6, followed by section 7 highlighting conclusion.

II. Related Work

Several numbers of experiments have been conducted to improve the performance of Hadoop cluster. The performance of virtualized Hadoop on VMware is presented through benchmarking case study in [6]. It reports the three Hadoop benchmarks performances on VMware. A comparison among native configurations is shown. The outcome depicts that the

ISBN: 978-1-4799-5754-5/14/$26.00 ©2014 IEEE

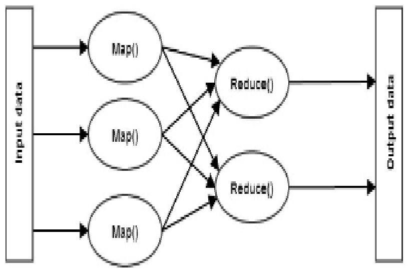

Fig. 1. High-level Map-Reduce Diagram

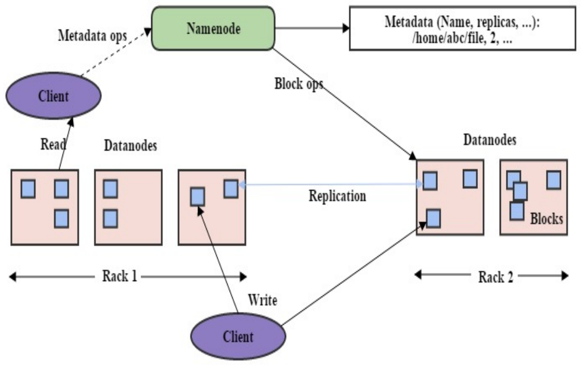

Fig. 2. HDFS Architecture

average performance difference between virtualized and native configurations is 4%. Better performance on virtualized configurations can be achieved by flexibility that is enabled by virtualization. Several BIOS, file system, OS, Hadoop, and Java tunings are explained in [5]. These tunings can be used to increase performance of Hadoop cluster. Similarly in [7] a profiling and performance analysis based self-tuning system PPABS is presented. Optimizing the configuration of the Hadoop MapReduce cluster was the main objective of the system. Predator, an experience guided configuration optimizer is introduced in [8]. In this optimization algorithm of the optimizer prevent local optimum problem by using subspace division. The searching time is reduced as the cost of visiting unpromising points is cut down within search space. Furthermore, the efficiency of the optimizer is demonstrated by the experiments carried out on Hadoop clusters.

Up till now, more focus has been made on optimizing Hadoop cluster performance by tuning Hadoop, OS, and Java parameters. This paper focuses on how the Linux service rpcbind plays an integral part in optimizing Hadoop cluster performance.

III. Hadoop Framework

Apache Hadoop is an open source tool originated from Big Table and MapReduce. It is intended to parallelize data processing across computing nodes to hide latency and speed computations. It enables the processing of large data sets across clusters of commodity servers in a distributed manner. The resiliency of these clusters comes from the software's ability to handle and detect failures at the application layer, instead of relying on hardware infrastructure. Hadoop provides a computing solution that is scalable, cost effective, flexible, and fault tolerant. Apache Hadoop primarily consists of two main sub projects namely MapReduce and HDFS [9].

A. MapReduce

Hadoop MapReduce is a software programming model. It writes applications that perform parallel processing of huge amounts of unstructured and structured data in a reliable and fault-tolerant manner across a cluster of thousands of machines. The workload amongst the nodes in a cluster is distributed by MapReduce which was originally developed at Google, and has become the real standard for large scale data analysis [10]. Many applications use MapReduce, such as machine learning, web access log stats, and statistical machine translation. MapReduce's key benefits include simplicity, speed, scalability, minimal data motion, built-in recovery, and freedom to focus on the business logic [11]. The task distribution across a large number of systems is done by "map" component. Additionally, it handles the placement of the tasks in a load balancing and reliable manner. As soon as the distributed computation is completed, "reduce" function generates final result by aggregating all the elements back together as shown in Fig.1.

B. HDFS

Hadoop Distributed File System (HDFS) is a Java-based file system. HDFS spans all the nodes in a Hadoop cluster for data storage. HDFS creates one big logical file system by linking individual file systems present on local nodes. HDFS replicates data across multiple nodes to achieve reliability in case of failure of nodes. It provides reliable, fault tolerant and scalable data storage that is intended to span large clusters of commodity servers [12]. Figure 2 shows HDFS architecture.

IV. Hadoop Multi node Cluster

The Hadoop cluster offers a great potential of resources to multiple users in a shared manner. A great number of commodity hardware is employed for cluster creation.

A. Hadoop Cluster Architecture

Hadoop cluster comprises of a single master and multiple slave nodes. The master node consists of DataNode, NameNode, ResourceManager and NodeManager. A slave node behaves as both a NodeManager and DataNode. Additionally, the secondary NameNode can be set up on master node while working on a test cluster. Previously, Job Tracker and Task Tracker were part of MapReduce framework but then MapReduce had experienced a complete revamp in Hadoop-0.23. As a result, presently we have YARN or MapReduce 2.0 (MRv2). The two major functionalities of the JobTracker, job scheduling / monitoring, and resource management, are split up into separate daemons by MRv2. The objective is to have a per-application ApplicationMaster

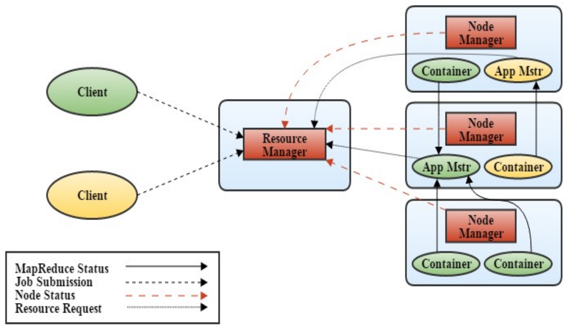

Fig. 3. YARN Architecture

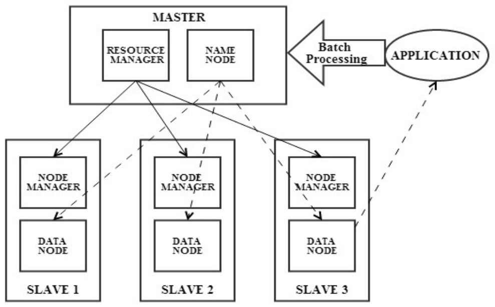

Fig. 4. Master/Slave Architecture of Hadoop Cluster

(AM), a global ResourceManager (RM), a per-node slave NodeManager (NM) and a per-application Container running on a NodeManager. The NodeManager (NM) and ResourceManager (RM) form the data-computation framework [13]. Further description is as follows. Figure 3 depicts ResourceManager and NodeManager in yarn architecture

1) NameNode: The master NameNode acts as the main server or coordinator of HDFS. NameNode manages the file system namespace by keeping data location index. Additionally, the access to files by clients is regulated by it. A file system namespace is provided by HDFS. HDFS permits the user data to be stored in files. A file is split up into one or more blocks that are stored in a set of DataNodes. The NameNode is responsible for file system namespace operations namely closing, opening, and renaming of directories and files. NameNode acts as a repository for all HDFS metadata as it decides the mapping of file blocks to DataNode. Neither does any data stored at NameNode nor any data flows through it. NameNode is contacted first by the application for data acquisition. It then provides locations of data blocks containing the file. [13]

2) DataNode: DataNodes manage the stored data.They are responsible for storing the chunks of file as determined by the NameNode. Data file to be stored is first split into one or more blocks internally. Additionally, they are responsible for serving write and read requests from the file system's clients, block creation, deletion and replication upon instructions from the NameNode. [13]

3) ResourceManager: The ResourceManager (RM) is responsible to arbitrate the resources among all the applications in the system. RM facilitates in the management of distributed applications running on the YARN system. It works together with the ApplicationMasters (AMs) and NodeManagers (NMs). The two components of resource manager are Scheduler and ApplicationsManager. [13]

4) NodeManager: The NodeManager (NM) is a per-machine framework agent of YARN. NM takes instructions from the ResourceManager and manages a single node's available resources. It is responsible for launching the applications containers, tracking node-health, log's management, monitoring resource usage, and other auxiliary services. It also reports the same information to the ResourceManager/Scheduler. [13]

B. Multi- Node Hadoop Cluster Configuration

A 4-node Hadoop cluster is configured in order to perform the required experiment. The cluster consists of one master node and three slave nodes. A Hadoop cluster requires both the HDFS and YARN cluster to be started. Other important Hadoop daemons include NameNode, DataNode, Secondary NameNode, ResourceManager, NodeManager, WebAppProxy, and Map Reduce Job History Server respectively. On master node three daemons including NameNode, ResourceManager, and Secondary NameNode are started. On slave node two daemons namely DataNode and NodeManager are started. Similarly, Map Reduce Job History Server can be started on the master node to track the history. The cluster is configured on virtual machines using virtual box. The basic architecture of the cluster is shown in Fig. 4.

The cluster is configured with virtual machines which have more advantages as compared to using physical ones. By using virtual machines for Hadoop deployment, virtualization benefits can be taken. It would be easier to manage the cluster and the reliability can also be improved as the virtual machines have a greater recoverability from crash as compared to the physical ones. Additionally, the system resources can be fully utilized by virtualization. But, the disadvantage of using above technique is the potential for poor performance and heavy load. The master and slave nodes operating system is Linux centos 6.5. Hadoop-2.4.0 along with Java SE development kit 8u5 (Java version 1.8.0_05) was used for Hadoop cluster configuration. As, there are so many configuration parameters available in Hadoop, we only tuned replication factor in the cluster configuration. Replication factor is the number of times the block of data would get replicated. By default Hadoop has replication factor of 3. Some of the configuration parameters for Hadoop cluster are shown in Table 1.

V. Hadoop Cluster Benchmarks and Metrics

A. Hadoop Benchmark Suite

A significant number of benchmarks and testing tools are available in the Apache Hadoop Distribution, including

TestDFSIO, MRBench, NNBench, Pi, TeraSort and many more [25]. These tools are consistently being used to report various results. The performance of the Hadoop cluster is evaluated by means of three major benchmarks namely TestDFSIO, Pi, TeraSort consisting of generate, sort and validate functionalities.

1) TestDFSIO: This benchmark performs a read and write test for HDFS. It is used to carry out various tasks including stress testing of HDFS, discovering network performance bottleneck, to shake out the hardware, OS and Hadoop setup of the cluster machines. Additionally, it informs how fast the cluster is in terms of I/O. The benchmark is split into two parts TestDFSIO-write and TestDFSIO-read. [14]

2) Pi: It is a purely computational application involving little storage I/O or network traffic. It approximates the value of pi using a Monte Carlo method. It is very nearly embarrassingly parallel as the map tasks are all independent and the single reduce task gathers very little data from the map tasks. [14]

3) TeraSort: This benchmark is probably the most well-known Hadoop benchmark. A large number of records are sorted by TeraSort. Additionally, it tests the HDFS and MapReduce layers of a Hadoop cluster. It does significant storage I/O, networking, and computation. It is often considered to be representative of real Hadoop workloads. It involves three phases: generation, sorting, and validation. TeraGen creates the data. TeraSort does the actual sorting and writes sorted data to HDFS in a number of partitioned files. TeraValidate reads all the sorted data to verify whether it is in correct order. [14]

The run time parameters of all the above mentioned benchmarks are shown in Table 2.

B. Performance Metrics

Although a number of metrics are obtained from the experiment but elapsed time is the most significant metric. Elapsed time or real time is the time to perform an event. It is the difference between beginning time and an ending time. Linux time command is used to measure the elapsed time. The same information can be obtained from resource manager web interface but Linux time command is preferred [14]. Elapsed time can easily differentiate the performance, as the results having less elapsed time are considered to have good performance.

C. Linux Services

A Linux service is an application that runs in the background carrying out some essential tasks or waiting to be used [15]. A number of network services are available namely rpcbind, nfs, xinetd, telnet, rsh and many more. This experiment deals with rpcbind. The RPC in rpcbind stands for remote procedure call. RPC is an inter-process communication that allows a computer program to cause the execution of a procedure or subroutine in another address space. The rpcbind utility is a server that transforms RPC program numbers into universal addresses. It allows the host to make RPC calls on a server [16]. In order to evaluate rpcbind service on the Hadoop cluster, the package needs to be installed as the service rpcbind does not exist in default settings of the cluster.

VI. Performance Results

In order to assess the performance of the configured Hadoop cluster with respect to rpcbind service, 3 different Hadoop benchmarks namely TestDFSIO, TeraSort and Pi have been executed on the cluster. Figure 5 shows the Hadoop cluster performance in terms of elapsed time for TestDFSIO benchmark. The TestDFSIO has two parts write and read; first data is written then read. The benchmarking is done with 10 files each of 1000MBs constituting a total 10 GB data. The results are shown with both rpcbind on and off. The benchmarking results give high elapsed time when rpcbind service is off for both the TestDFSIO write and read operations. As, the rpcbind service gets started the elapsed time is reduced for both the reading and writing respectively. Similarly, another benchmark called TeraSort is executed on Hadoop cluster. TeraSort consists of 3 steps; data generation by TeraGen, then sorting of data by TeraSort, and finally validation with TeraValidate. The data used in this benchmark is 3GB in size. Figure 6 shows elapsed time with each 3 of the phases of TeraSort. The results show elapsed time is greater with rpcbind off, in all three phases of TeraSort. Although, TeraGen and TeraSort show significant difference when rpcbind status is changed but TeraValidate shows a minor difference. This happens due to less I/O involvement in the validation process. The greater the I/O, greater will be the impact of rpcbind. The third benchmark executed on the cluster is Pi. The computation is performed with 500 maps and 10 GB samples per map. As, Pi involves more computation than I/O, the performance is not highly affected by changing the service status. It can be observed from Fig.7, the rpcbind service status does not highly affect the elapsed time. The elapsed time with rpcbind on is still smaller as compared to the other scenario. It can be seen in Fig. 7 the bar corresponding to rpcbind off is larger as compared to the bar with rpcbind on. Similarly, Table 3 shows the same results in tabular form.

TABLE I. Hadoop Cluster Configuration Parameters

Properties	Values
No of Nodes	4 Nodes, 1 Master Node, 3 Slave Nodes
Virtual Box Host Machine	Intel Xeon X5670 Server System, 24 Cores, 20GB RAM, Windows Server 2008 (64 bit Operating System), 500 GB Hard Disk
Virtual Box Guest Machines	Linux CentOS Release 6.5 64 bit Kernel, 2 GB RAM and 20 GB Disk
Hadoop Version	Hadoop-2.4.0
Java Version	Java SE Development Kit 8u5 (Java Version 1.8.0_05)
Hadoop Replication Factor	2

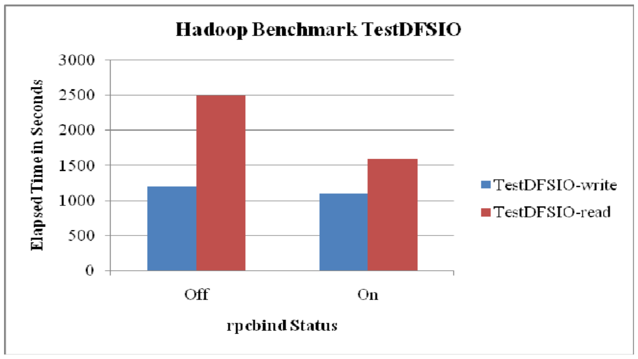

Fig. 5. Hadoop Cluster Performance with TestDFSIO Benchmark

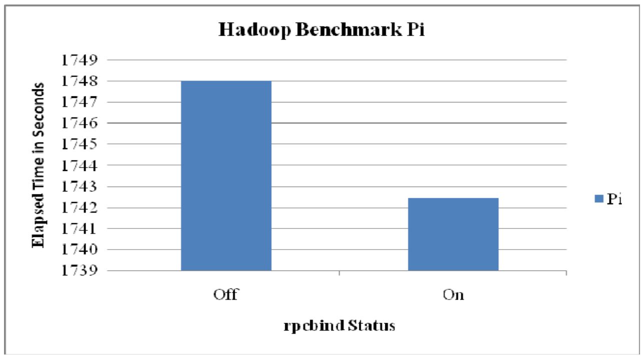

Fig. 7. Hadoop Cluster Performance with Pi Benchmark

CS

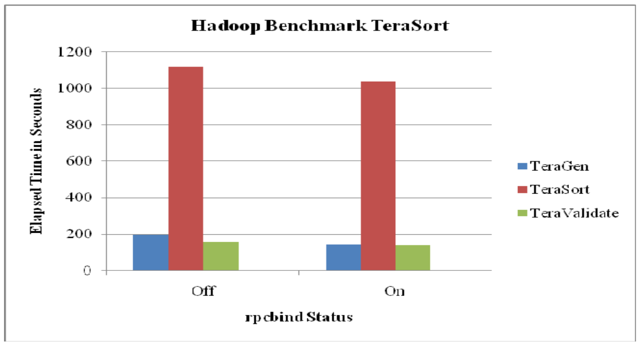

Fig. 6. Hadoop Cluster Performance with TeraSort Benchmark

VII. Conclusions

Linux services play an important part in cluster configurations. Certain services might contribute in increasing the elapsed time of a job. But, rpcbind is contributing towards improvement of the overall cluster performance by reducing the elapsed time needed for job execution. The cluster is tested with 3 major Hadoop benchmarks and the results showed favorable outcomes reducing the overall elapsed time when the rpcbind service is on. rpcbind service is needed in order to improve the Hadoop cluster performance. Thus, while configuring a Hadoop cluster it would be a better initiative to get the rpcbind service started as it does not exist by default in most Linux distributions.

TABLE II. Hadoop Benchmarks Parameters

Benchmarks	Parameters
TestDFSIO	-write/-read, -nrFiles (Number of files)=10, -filesize (Size of each file)=1000
Pi	No of Maps=500, Samples per Map=100000000
TeraSort	TeraGen/TeraSort/TeraValidate, DataSize=30000000 (3GB)

TABLE III. Hadoop Cluster Performance with 3 Benchmarks

Benchmarks / **Performance Metric**	**TestDFSIO**		**TeraSort**			**Pi**
	TestDFSIO- write	***TestDFSIO- read***	***TeraGen***	***TeraSort***	***TeraValidate***	***Pi***
Elapsed Time in Seconds (rpcbind Off)	1207.163	2489.184	197.464	1117.285	155.077	1748.016
Elapsed Time in Seconds (rpcbind On)	1108.138	1588.137	142.671	1038.17	136.448	1742.444
Speedup	1.08	1.56	1.38	1.07	1.13	1.01

References

[1] J. Shaw, "Why Big Data", Reprinted from Harvard Magazine, Apr. 2014, Available [online]: https://www.cfa.harvard.edu/~agoodman/newweb/0314-30.pdf.

[2] Nandimath, J.; Banerjee, E.; Patil, A.; Kakade, P.; Vaidya, S.; Chaturvedi, D., "Big data analysis using Apache Hadoop," Information Reuse and Integration (IRI), 2013 IEEE 14th International Conference on , vol., no., pp.700,703, 14-16 Aug. 2013, doi: 10.1109/IRI.2013.6642536.

[3] Rasooli, A.; Down, D.G., "A Hybrid Scheduling Approach for Scalable Heterogeneous Hadoop Systems," High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion: , vol., no., pp.1284,1291, 10-16 Nov. 2012, doi: 10.1109/SC.Companion.2012.155.

[4] Singh, K.; Kaur, R., "Hadoop: Addressing challenges of Big Data," Advance Computing Conference (IACC), 2014 IEEE International, vol., no., pp.686,689, 21-22 Feb. 2014, doi: 10.1109/IAdCC.2014.6779407.

[5] D. Russell, “Tuning Hadoop on Dell PowerEdge Servers”, Available [online]: http://en.community.dell.com/cfs-file/__key/telligent-evolution-components-attachments/13-4491-00-00-20-43-80-92/tuning_2D00_hadoop_2D00_on_2D00_dell_2D00_poweredge_2D00_servers.pdf?forcedownload=true.

[6] "A Benchmarking Case Study of Virtualized Hadoop Performance on VMware vSphere 5", http://www.vmware.com/files/pdf/VMW-Hadoop-Performance-vSphere5.pdf. (2014, July. 17).

[7] rpcbind Service, Available [online]: http://linux.die.net/man/8/rpcbind.

[8] Dili Wu; Gokhale, A., "A self-tuning system based on application Profiling and Performance Analysis for optimizing Hadoop MapReduce cluster configuration," High Performance Computing (HiPC), 2013 20th International Conference on , vol., no., pp.89,98, 18-21 Dec. 2013, doi: 10.1109/HiPC.2013.6799133.

[9] Kewen Wang; Xuelian Lin; Wenzhong Tang, "Predator — An experience guided configuration optimizer for Hadoop MapReduce," Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on , vol., no., pp.419,426, 3-6 Dec. 2012, doi: 10.1109/CloudCom.2012.6427486.

[10] Apache Hadoop, Available [online]: http://hortonworks.com/Hadoop/.

[11] H. Karloff, S. Suri, and S. Vassilvitskii, “A Model of Computation for MapReduce.”, Available [online]: http://theory.stanford.edu/~sergei/papers/soda10-mrc.pdf.

[12] MapReduce, Available [online]: http://hortonworks.com/Hadoop/MapReduce/.

[13] HDFS, Available [online]: http://hortonworks.com/Hadoop/hdfs/.

[14] Hadoop YARN, Available [online]: http://hortonworks.com/Hadoop/yarn/.

[15] Michael G. Noll, "Benchmarking and Stress Testing an Hadoop Cluster with TeraSort,TestDFSIO &Co", Available [online]: http://www.michael-noll.com/blog/2011/04/09/benchmarking-and-stress-testing-an-hadoop-cluster-with-terasort-testdfsio-nnbench-mrbench/.

[16] Linux Services, Available [online]: http://www.linux.com/news/enterprise/systems-management/8116-an-introduction-to-services-runlevels-and-rcd-scripts.