Journal of Emerging Computer Technologies
Research Article
Received: *2020-10-04* | ***Reviewing:*** *2020-10-28* & *2020-11-18*| ***Accepted:*** *2020-12-02* | ***Online:*** *2020-12-10* | ***Issue Date:*** *2021-06-30*

Log Analysis with Hadoop MapReduce

Gligor Risteski
Faculty of Communication Network and Security
University of Information Science and Technology, Ohrid, Macedonia
gligorr@gmail.com

Mihiri Chathurika
Faculty of Communication Network and Security
University of Information Science and Technology, Ohrid, Macedonia
mihiritxla@gmail.com

Beyza Ali
Faculty of Communication Network and Security
University of Information Science and Technology, Ohrid, Macedonia
beyzaali34@gmail.com

Atanas Hristov
(Corresponding Author)
Faculty of Information and Communication Sciences
University of Information Science and Technology, Ohrid, Macedonia
atanas.hristov@uist.edu.mk,
ORCID: 0000-0003-2741-8370

***Abstract*— Pretty much every part of life now results in the generation of data. Logs are documentation of events or records of system activities and are created automatically through IT systems. Log data analysis is a process of making sense of these records. Log data often grows quickly and the conventional database solutions run short for dealing with a large volume of log files. Hadoop, having a wide area of applications for Big Data analysis, provides a solution for this problem. In this study, Hadoop was installed on two virtual machines. Log files generated by a Python script were analyzed in order to evaluate the system activities. The aim was to validate the importance of Hadoop in meeting the challenge of dealing with Big Data. The performed experiments show that analyzing logs with Hadoop MapReduce makes the data processing and detection of malfunctions and defects faster and simpler.**

***Keywords*— Hadoop, MapReduce, Big Data, log analysis, distributed file systems.**

I. Introduction

The term "Big Data" is gaining more popularity every day. The first thing we should know about it is that it does not have a commonly held definition. Basically, as one can understand from its name, Big Data means a big amount of data. Sethy, R. [1] in his article defines "Big Data describes any massive volume of structured, semi-structured and unstructured data that are difficult to process using a traditional database system."

Researches show that data volumes are doubling every year. Although there is not a specific reason behind this rapid growth rate, the new data sources, contribute to that growth highly. Smartphones, tablet computers, sensors, and all other devices that can be connected to the internet generate a vast amount of data. Enterprises improve their technological infrastructures and adopt more powerful platforms, which play an important role in the growth rate of the data that is generated [2].

II. Hadoop

Hadoop is a collection of open-source utilities which allows the use of network to deal with the problems which include big amounts of data. Hadoop provides framework for distributed storage and framework for processing big data with MapReduce programing model. The core of Apache Hadoop framework contains the following parts:

- Hadoop Common
- Hadoop Distributed File System (HDFS)
- Hadoop YARN
- Hadoop MapReduce

Hadoop common represent libraries and utilities which are needed by other Hadoop parts in order to operate. Hadoop distributed File System (HDSF) is a distributed file-system that stores data on commodity hardware, allowing very high bandwidth across the cluster. Hadoop YARN is a platform responsible for managing computing resources in clusters and uses them for scheduling users' applications. Lastly, Hadoop MapReduce is a programing model for big data processing in the cluster.

Java is a programming language that is mostly used for writing MapReduce programs but Hadoop allows the use of any programming language to write MapReduce programs.

To exploit the parallel processing that Hadoop gives, we have to express our query as a MapReduce job. After some local, little scope testing, we can have the option to run it on a cluster of machines.

The Hadoop structure is demonstrated in Figure 1.

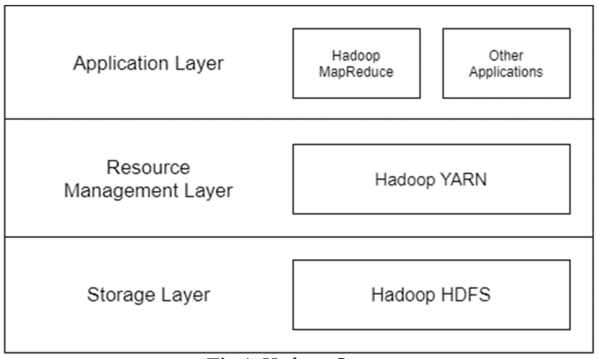

Fig 1. Hadoop Structure

A. *Hadoop Distributed File System (HDFS)*

Hadoop Distributed File system is one of the most reliable storage systems designed to store a smaller number of large files rather than a greater number of small files. Among many of the features HDFS provides, the fault-tolerant storage layer can be mentioned as one of the most important features.

Replication of data in the Hadoop file system helps the user to attain this feature. Even in situations where hardware failure happens, the data reliability is still high [3].

Apache Yarn, introduced in Hadoop 2.x, is the resource management layer of Hadoop, which is also used for job scheduling and data operating system. It allows different data processing engines to run and process data stored in HDFS. Some of these processing engines are graph processing, interactive processing, stream processing, and batch processing. Data processing platform Yarn has the functionality named MapReduce, which empowers Hadoop by allowing processing numerous different frameworks on the same hardware where Hadoop is deployed [4].

HDFS was developed using distributed file system design and is designed using low-cost hardware. It is more fault-tolerant than other distributed systems. HDFS is capable of holding larger amounts of data providing easy access and parallel processing. In HDFS files are stored across multiple machines in order to prevent possible data losses in case of system failure [3]. Features of HDFS are: distributed storage and processing, command interface, checking the status of clusters, streaming access to the file system, file permission, and authentication.

- Distributed storage and processing
- Command interface to interact with HDFS
- Checking the status of cluster easily
- Streaming access to file system data
- File permissions and authentication

1) HDFS Architecture

HDFS follows the master-slave architecture. It has elements such as namenode, datanode, and blocks, where the built-in servers of the first two elements enable users to easily check the status of the cluster. Similarly, they are commodity hardware [point]. In Figure 2 the architecture of the Hadoop file system is demonstrated.

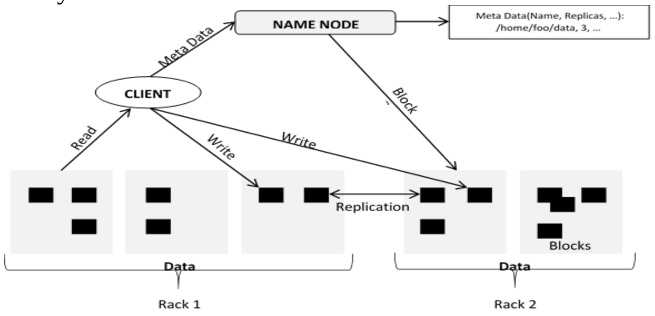

Fig 2. Architecture of a Hadoop File System

- Namenode

The namenode, containing GNU/Linux operating system and the namenode software, can act as the master server running on commodity hardware. Renaming and opening/closing files/directories are executed in this element. It also manages the file system and regulates clients' access to files.

- Datanode

Similar to namenode, the datanode too, contains the GNU/Linux operating system and datanode software. For every node in a cluster, there exists a datanode, which is able to manage the data storage of the system. Depending on the instructions of the namenode, datanode performs block creation, block deletion, and block replication. It also performs read/write operations on the file system when requested by a client.

- Block

. Usually, the user data are stored in files such that a file in a file system is divided into segments and then are stored in individual data nodes. Those file segments are called blocks. A block is the minimum amount of data that HDFS can read/write. The block size can be increased if needed but the normal size is 64MB.

B. *Apache Hadoop Yarn*

To create a split between Resource Manager (RM) and Application Master (AM), YARN separates the functions of resource management and job scheduling into separate daemons. An application can be an individual job or a DAG of jobs.

- ResourceManager (RM)

Resource Manager (RM) together with the Node Manager (NM) comprise the data-computation framework. The RM adjudicates the resources in the system and NM is responsible for containers and monitoring resource usage. In other words, RM is the ultimate authority and NM is the framework agent. RM has two main components known as Scheduler and Application Manager.

- Scheduler

Scheduler controls the allocation of resources to the several running applications. It performs the scheduling function depending on the resource requirements of applications, but it does not take any responsibility for performing monitoring, tracking the status of applications or application or hardware failures.

- ApplicationsManager

Applications Manager: responsible for restarting the AM container in case of a failure. It also accepts job submissions.

- ApplicationMaster (AM)

The AM is responsible for requesting resources from the RM and then executing and monitoring the tasks. It works together with NM when executing the tasks.

The main idea behind using YARN on Hadoop is the notion of resource reservation via the Reservation System. The Reservation System tracks the resources, performs admission control for reservations, and reserves resources to ensure the execution of important jobs [4].

YARN supports the notion of Federation via the YARN Federation feature. The idea behind this is to scale Yarn up to very large amounts of nodes by wiring YARN clusters and sub-clusters. The Federation feature makes this transparent wiring of clusters appear as a single big cluster [5].

III. The Role of MapReduce In Handling Big Data

Big Data means "big power" when handled efficiently. It can give new aspects to the enterprises, like which strategy will increase the profitability, which customers buy which products, the current situation of the company versus the situation of the competitors, and so on.

As the data comes from different sources and different structures it is important to categorize it with respect to some characteristics of the data. The most important and the most known characteristics of Big Data are known as the "3Vs of

Big Data" where the Vs stands for volume, variety, and velocity. Volume refers to the amount of data, variety refers to the type of data, i.e. text, image, video, etc. and velocity refers to the speed at which the data comes from different sources [6].

Big Data comes with its own set of problems that need to be resolved. Processing power, storage, data issues, and cost are the most important problems. The old techniques for working with or analyzing information are not enough to deal with Big Data. Therefore, new technologies are needed and this is where MapReduce comes into the picture [2].

A. MapReduce

MapReduce is a programming model that is used for accessing and processing big data stored in HDFS. Programs written in MapReduce are executed on a distributed system where big data is split into smaller chunks of data and are executed in parallel [7]. MapReduce has two functionalities, Map () and Reduce (). This model has been used in Google's search index, machine learning, and statistical analysis [8]. Implementation of MapReduce is highly scalable and easy to use. The run-time system allows programmers with no prior knowledge or experience with parallel processing to utilize the resources of distributed systems easily, by handling details like partitioning the inputs, scheduling the program's execution, handling failures, and managing inter-machine communication [7, 9].

Although it is impossible to prevent failures, the objective is to minimize the probability of failure to a level that will not harm the overall process. Two methods that would help to increase the "fault tolerance" in Big Data are the following [6]:

- First divide the whole computation into smaller tasks and then assign each task to a different node.
- Assign a node to observe if the other nodes are working properly. In case a node fails to complete its task, the task is restarted. But this may cause a complication in the process if some tasks are recursive.

MapReduce is one of the core building blocks of processing in the Hadoop framework. Hadoop uses the MapReduce algorithm to run the applications in parallel. It provides the necessary solution to keep the process going since it can survive failures without losing data. In 2004 Google published about MapReduce technology [7]. MapReduce comprises of two distinct tasks: Map and Reduce. Mapping is the first phase and Reducing happens after the Mapping phase is completed. In Map phase data is processed and key-value pairs are produced. This is known as the map job. Then the produced key-value pairs are fed into the Reducer. After collecting all the key-value pairs from all of the map jobs the Reducer groups the pairs into a smaller set of key-value pairs, producing the final output [7, 8].

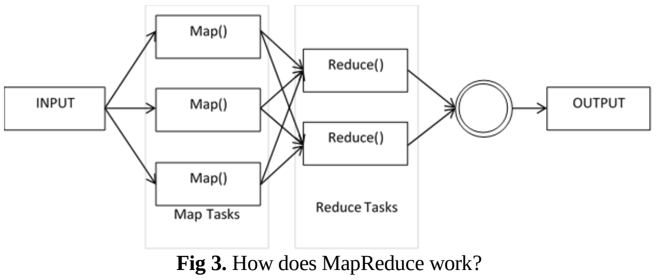

Fig 3. How does MapReduce work?

B. Algorithm of MapReduce

MapReduce program has three stages of executing: map, shuffle, and reduce.

- Map stage: at this stage, the map job processes the input data which is in the form of a file or directory. As an output of the map job several small chunks of data are produced.
- Shuffle stage: at this stage, the output of the map stage is accepted and the relevant records are consolidated.
- Reduce stage: reduce stage acts together with the shuffle stage. The reducer processes the data that comes from the map stage and produces a new output which is then stored in the Hadoop file system.

During the process, a task from Map and Reduce phases are sent to the appropriate servers in the cluster. All the data passing details like issuing tasks, verifying task completion, copying data between nodes, are managed by the MapReduce framework. Because the computing is performed on nodes using the data from local disks the network traffic is reduced significantly. After the tasks are completed the results are sent back to the Hadoop server [10].

C. MapReduce with Python

'mrjob' is a Python library for MapReduce for writing and running Hadoop streaming jobs. It is created by Yelp. When a MapReduce job is written using 'mrjob', it can be tested locally and run on a Hadoop cluster or in the cloud. Using 'mrjob' for writing MapReduce applications has many advantages. Some of them are:

- It is a dynamically developing framework.
- It has extensive documentation.
- Installing Hadoop is not enforced. Applications written using 'mrjob' can be executed and tested without installing Hadoop.
- It allows the MapReduce applications to be written in a single class rather than writing separate programs for Map and Reduce phases.

Although it provides a great solution, 'mrjob' has its disadvantages. The most important disadvantage is that it does not provide the level of access to Hadoop that other APIs provide. This is because it is a simplified library [11].

IV. Log Analyzer with Hadoop

Log analysis is both art and science which aims to make sense out of computer-generated records. These records are called log or audit trail records. The process of creating these records is known as data logging [12]. Some of the most important reasons for performing log analysis are:

Understanding user behavior

- System troubleshooting
- Proper resource allocation
- Improved business operations
- Improved security
- Achieve compliance

Data centers generate thousands of terabytes or petabytes of log files every single day. It is very challenging to store and analyze these data not only because of its large volume but also because of the different structure of log files. Due to not being able to deal with a large volume of log files efficiently, conventional database solutions run short for the needs in log analysis. As a result of the comparison of SQL DBMS and Hadoop MapReduce in [13], Hadoop MapReduce overperforms DBMS in the means of tuning up with the task and loading data. As it can be seen from this result, with the unprecedented increase in the data generated traditional methods fall short with providing a solution for data analysis. This is, exactly, the point where the new technologies stepped in [8]. Hadoop MapReduce has a wide area of applications for Big Data analysis [3],[9],[11]. The true power of Hadoop lies in its ability to scale up to a great number of computers, where each computer has several processor cores, by connecting commodity computers to work in parallel. This plays an important role in log analysis as it can benefit thousands of nodes which will store multiple blocks of log files.

In this paper, we propose an idea on how Hadoop can be used to analyze web server logs, in our case Nginx access log. Web server access logs are generated by the web servers all the time, recording all accesses on the hosted web pages. This means that the access logs can be very big. The web access log contains information about time, IP addresses, browser type, etc. All of this information is important for the system administrators as it provides information about system usage, security, and system troubleshooting.

This idea is proposed for analyzing one kind of logs only, but Hadoop can be used in every situation where big log files are generated, such as system logs, logs from some business application, etc.

The real-world usage (practical usage) of this system can be implementing it as a base of a larger system used by many users for log analyzing.

A. Environment Setup

We have installed Hadoop for demonstrating purpose on two virtual machines hosted on Digitalocean. Each virtual machine has 2 cores CPU and 8GB of RAM memory (installing is explained in details in Installing Hadoop section) with installed Ubuntu 16.04. Also, for testing purpose we have installed Hadoop on one local virtual machine with 1 core CPU and 7GB of RAM memory on hosted hypervisor VMware Workstation Pro 15 using Bitnami Hadoop Stack image. In this project the log generated from Python Fake Logs script is used [14].

V. Results

Our aim was to validate the role and importance of Hadoop in meeting the challenge of dealing with Big Data, by an analysis of log files in order to evaluate the system activities. We benefitted from Hadoop's ability to scale up to as many computers as needed. Firstly, log files are broken up into blocks with MapReduce class created with mrjob python library (dividing into blocks is explained in section 3.1). Blocks are then distributed over the nodes in a Hadoop cluster. With parallel computation, the job is divided into a number of tasks, which in return improves the performance. In our log analysis, we used Python's 'mrjob' library.

We use a dummy log file only for testing. Our log file is the Nginx access log file. The log file contains information about:

- visitor's IP address,
- date and time of the access,
- visited page,
- type of request,
- type of the user's web browser,
- type of the user's operating system.

In our case, two reports are extracted from the Nginx access file as an example. These two reports are only an example of what can be achieved. The following charts demonstrate the results of our experiments. In Figure 4 the number of visits is shown based on the hours the visitor visited the websites. We can see that the number of visits is the highest at 20:00 – 21:00. The number of visits is the lowest at 10:00 – 11:00.

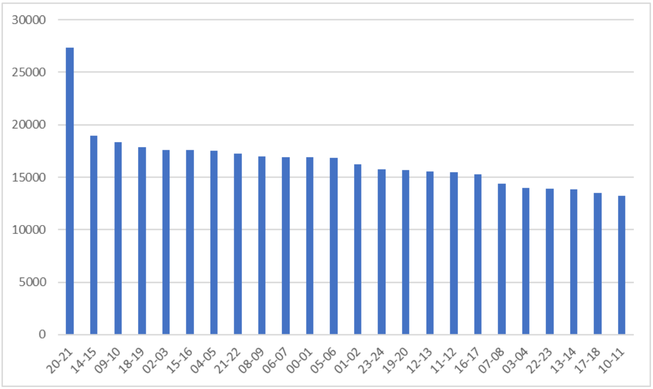

Fig 4. Number of visits by hour

The second report analyzes which are the most visited pages. It is possible to make many other reports depending on the user's needs. For example, the most used browser, most used operating system, detecting suspicious behavior, etc.

In Figure 5 the distribution of the visits is displayed with regard to the most visited 10 sites.

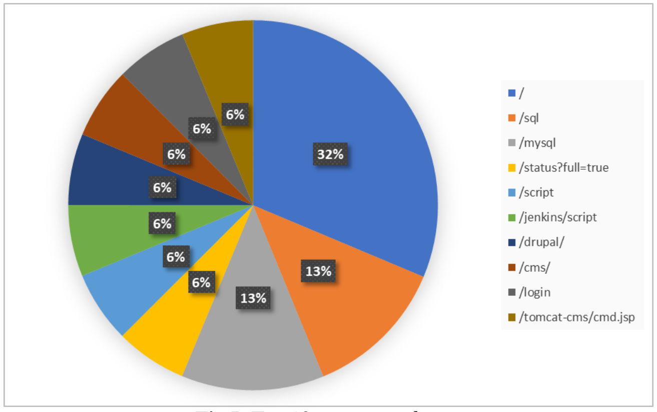

Fig 5. Top 10 most visited sites

VI. Discussion

Log data grows very quickly as they are generated at a record rate. Processing power, storage, and cost are some of the biggest challenges that come with this Big Data. Using conventional methods to deal with such data is too costly in terms of both time and money. Hadoop provides frameworks for distributed storage and processing for Big Data, which makes it possible to use a network for processing and analyzing big amounts of data. In this way, companies save time and money when trying to diagnose problems, solve issues, or obtain the knowledge they wouldn't be able to obtain otherwise. The ability to store as well as distribute large datasets across numerous servers in a cost-effective way makes Hadoop very advantageous with regard to the traditional relational database management systems. Moreover, Hadoop MapReduce processes terabytes of data in minutes and by executing tasks in parallel shortens the processing of data by a considerable amount.

VII. Conclusion

Log analysis is important for businesses and system administrators in many aspects, as they give information about malfunctions, defects, security-related issues, and so on. On the other hand, being generated constantly provides a good example of Big Data, therefore a good example of our experiments. Our analysis shows that analyzing logs with Hadoop MapReduce makes the detection of malfunctions, defects, and so on faster and simpler. The results also show that there is not a big difference if we run the code in a real cluster or in the test environment.

References

[1] Sethy, R. et al. Big Data Analysis using Hadoop: A Survey. International Journal of Advanced Research in Computer Science and Software Engineering 5(7), 2015, pp. 1153-1157.

[2] Schneider, R.D. Hadoop For Dummies, Special Edition. John Wiley & Sons Canada, Ltd. 2012.

[3] Borthakur, D. HDFS architecture. Document on Hadoop Wiki. http://hadoop. apache. org/common/docs/r0 20. 2010.

[4] Vavilapalli, V. K.; et al. Apache hadoop yarn: Yet another resource negotiator. Proceedings of the 4th annual Symposium on Cloud Computing. ACM, 2013.

[5] Hadoop, Apache. Hadoop Archives Guide. The Apache Software Foundation, http:// hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html (2019). Retrieved Oct. 15, 2019.

[6] Kaur, I. et al. Research Paper on Big Data and Hadoop. IJCST, 7(4), 2016, pp. 50-53.

[7] Dean, J. and Ghemawat, S. MapReduce: Simplified data processing on large clusters. Proceedings of Operating Systems Design and Implementation, 2004.

[8] Yang, H. et al. Map-reduce-merge: simplified relational data processing on large clusters. Proceedings of the ACM SIGMOD international conference on Management of data. ACM, 2007.

[9] Rohloff, K. and Schantz, R.E. High-performance, massively scalable distributed systems using the MapReduce software framework: the SHARD triple-store. Programming support innovations for emerging distributed applications. ACM, 2010.

[10] Point, Tutorials. Retrieved Oct. 15, 2019 from Internet Site https://www.tutorialspoint.com.html. Tutorials Point.

[11] Miner, D. and Radtka, Z. Hadoop with Python. O'Rilley Media. 2016.

[12] Log analysis https://en.wikipedia.org/wiki/Log_analysis

[13] Sayalee Narkhede and Tripti Baraskar - Hmr Log Analyzer: Analyze Web Application Logs Over Hadoop MapReduce International Journal of UbiComp (IJU), Vol.4, No.3, July 2013.

[14] "Python Fake Logs." Internet: https://github.com/s4tori/fake"Python Fake Logs." Internet: https://github.com/s4tori/fake-logs.-logs.