**Apache Hadoop**

Anahit Sarao - 008435583 | Brendan Hart - 008329061

Jay Bajaj - 008241740 | Maxwell Cheshier - 009193717

Nadim Sarras - 008382608

Course: CMPE 142 Spring 2016

Professor: Juan Gomez

Date: May 5, 2016

**Index**

History of File Systems……...…………………………………………………………………… 2

Evolution………………………………………………………………………………………..... 3

Functionality……………………………………………………………………………………... 5

API..……………………………………………………………………………………………… 7

Sample Code…………………………………………………………………………..…. 8

Other DFS………………………………………………………………………………….…….. 9

State of the Art……………...…………………………………………………………..………. 10

Applications……...……………………………………………………..………………………. 11

Advantage and Disadvantages………………………………………………………………….. 12

The Future of Hadoop Distributed File System………...………………………………………. 15

Conclusion…………………………………………………………………………....………… 16

Bibliography………………………………………….………………………………………… 17

**History of File Systems**

A file system is the methods and data structures a computer operating system uses to retrieve, catalog, and store files. Without a file system present in an operating system, data stored on a machine will gather into one large body of data with no efficient way of tell where one file starts and the other begins. By giving each piece of data a name, the information can easily be identified and stored into different directories for easy access. There are a wide variety of different file systems, and each one has been developed to serve a specific purpose. Some of these different file systems have been designed to share files between computers, transfer files using network protocols, and quickly store large amounts of data known as big data.

The first file system was created in 1973 by Gary Kildall and was called CP/M. Gary was a programmer who wrote a program called “Control Program for Microcomputers” (Reimer, 2008) that stored files and ran programs from a floppy drive. The CP/M filesystem, stored files in a completely flat hierarchy with no directories, which was limited to eight characters plus an extra dot three character extension that helped determine the file type. What made Gary’s program unique is that it “separated all the computer specific bits from the rest of the operating system” (Reimer, 2008). The program became what is now known as the basic input out system (BIOS). Eventually, Microsoft bought out the program and renamed the package MS-DOS, which became the main operating system for many industrial and personal computers we see today.

Back when the first file system was created, memory was scarce and expensive. Not many people had a personal computer, which meant sharing and storing a large amount of data wasn’t an issue. Fast forward to today and now everyone has multiple personal computers that constantly share information and data through a computer network. In order to cope with this “Big Data,” a new file system had to be developed that would quickly sort, store, and retrieve the enormous amount of data floating around the network. The Hadoop program created in 2005 by Doug Cutting and Mike Cafarella has proven to be very useful in sorting and managing large amounts of data efficiently. Now managed by Apache Hadoop, the “Hadoop Distributed Filesystem (HDFS) allows storage of enormous data sets through distributed clusters of servers and then analysis applications within each data cluster” (Proffitt, 2013). The program is designed to be robust and fail proof, it allows for “Big Data” applications to continue running on an individual server, even when a hard drive cluster fails. The program is made up of two core parts: a data processing framework, and a distributed filesystem for data storage. The data processing framework deals with the data itself, and is responsible for getting the data processed. The distributed file system is responsible for holding the actual data. Overall, the creation of the Hadoop filesystem has helped with managing large amounts of data quickly, cheaply, and efficiently, making it an ideal filesystem to use for processing big data.

**Evolution**

Overall the use of the HDFS (Hadoop Distributed File System) has increased over time because of the amount of large data that is required to be processed. Over time because of the need for improvement in efficiency, there have been numerous changes to the file system in order to meet expectations. Through uncovering the history of the hadoop file system, which is inspired by Google’s MapReduce, it is required for processing large amounts of data with “affordable, flexible data structures” (Kim, 2015). In order to deal with increasing data sizes and improve analytics in relation to the big data infrastructure of many companies, HDFS has evolved through the years. By studying the evolution of HDFS, we need to understand how it started, the need for improvement, and the present state of Hadoop Distributed File System in order to understand its importance.

First, it is important to analyze the starting point of the Hadoop Distributed File System, which began from Google’s MapReduce. HDFS main goal is to provide an infrastructure to manage large amounts of data. It began through the need of an infrastructure to manage, large clusters of data effectively as companies had to spend a lot of money to create that infrastructure. In order to save company resources, Hadoop provided this capability in an affordable and flexible data structure. A huge infrastructure needed by companies to manage their data , evolved to the HDFS. HDFS provides all the functionality needed by companies through scaling data clusters which is, “break down big data into manageable chunks and run smaller jobs in parallel, using low cost hardware, where fault tolerance and self-healing would be managed in the software” (Kim, 2015). HDFS started as an open source software, which has resulted in one of the biggest inventions in data analytics.

Due to the evolution of technology through the years, HDFS needed to meet expectations and evolve along with what was needed by the changing industries. Even, though it started as a revolution to change the world of Big Data, there needed to be changes and improvements to its features. According to an article in the Converge Blog powered by MapR, Dale Kim identifies the aspects of Hadoop, “batch-oriented job processing and consolidated resource management ‘as’ limitations” (Kim, 2015). In order to deal with the issues of the infrastructure, the invention of YARN (Yet Another Resource Negotiator) was the next evolution that was implemented in order to improve on the Hadoop architecture. YARN allowed “ multiple data processing engines to handle data stored in one platform” (Kim, 2015). This improvement resulted in an increase in usage and popularity because it allowed developers to add code to containers instead of running them separately as jobs advancing the capabilities of the system (Diaz, 2014). These overview of the capabilities hadoop can now implement due to YARN, are shown below in the figure, this need for improvement due to limitations of data processing resulted in an evolution in the functionality of HDFS allowing more job processing and data handling.

Presently, the Hadoop File distribution system has resulted in a lot of changes to conform to market expectations and the growing industry. In order to understand the big data analysis landscape, it is important to see the present status of the file system in order to analyze, how it is going to change in the future as explained later in the report. Hadoop has grown alot since the last 10 years when it was first introduced, these improvements have resulted in a system that can process large amounts of data flexibly and affordably, adding improvements through the years to meet with industry requirements making it more versatile, and by leading the Big Data industry. Evolving technologies such as Apache Spark, are making more and more an impact on the benefits of the Hadoop architecture, by adding “a virtual variable to represent the large dataset … pick the right functions for the right data manipulation” (Diaz, 2014). This evolving industry of Big Data has a lot of potential and has been changing at a rapid pace to meet demand.

**Functionality**

Hadoop distributed file system (HDFS) is a powerful, scalable and portable file system written for the Hadoop framework. It is an open source piece of software that analyzes gigabytes and petabytes of both structured and unstructured data. It then takes this data and makes it more manageable for other applications (Sivaraman, 2014). What makes Hadoop so powerful is that the platform is operated on commodity hardware. The idea of commodity hardware is to use several low cost low power machines computing in parallel instead of fewer, costlier high powered machines working in parallel. These computers working and communicating in parallel are often referred to as a cluster. Unfortunately, as the node grows, there becomes more machines in the system than the mean time to failure which means that the system has to have some sort of fault tolerance. The functional aspects of Hadoop include: high availability, fault tolerance, scalable, flexible, and tunable replication (Sivaraman, 2014).

Hadoop distributed file system’s architecture largely enables it to be sleek as well as maintaining fault tolerance. It utilizes a write-once, read-many model that splits the data into smaller blocks that are distributed across many nodes to ensure high performance and fault tolerance (Sivaraman, 2014). Each HDFS cluster contains a primary NameNode that is a server in charge of managing the file system namespace as well as allowing interaction between the client and the stored data. There is also a backup node, a Secondary NameNode which is a replication of the primary NameNode to be used in the occurrence of system failure adding to its fault tolerance. It is not simply an instantiated backup, because it constantly snapshots the content of the primary NameNode, providing an exact copy of the primary NameNode at the time of failure.

Much of Hadoop’s reliability and power comes from its leveraging the concepts of map and reduce functions. The software that Hadoop employs to carry out these functions is referred to as MapReduce. MapReduce takes large sums of data and then reduces it into small, more manageable chunks that is distributed across all of the machines within the node. As the name implies, there are two key steps to MapReduce: the map step processes input data and the reduce step takes the intermediate results and finalizes them to create the final result. Inputs and outputs from the user act as a key-value pair which means the output of one job can act as the input as a another, vastly increasing efficiency (Sivaraman, 2014). Another aspect that increases the efficiency of using Hadoop and MapReduce is the fact that the computation is executed at the location of the data as opposed to moving the data somewhere to be computed because the physical data and the computation coexist on the same node within a cluster. Using this data proximity allows Hadoop and MapReduce to bypass the traditional bottlenecks of bandwidth along with dividing the tasks up to executed in parallel.

Using the two components of Hadoop: HDFS and MapReduce, we are able to massively scale down the amount of physical space data takes up by replicating it across various machine that work in parallel. The HDFS enables a large degree of fault tolerance whereas MapReduce allows for the scalability of data as well as improving overall performance. These two functions make Hadoop a very enticing platform for businesses or apps to store large amounts of data safely and efficiently.

**API**

The current version of Apache Hadoop’s API is 2.7.2. It is split into three main packages named Common, MapReduce, and Yarn. The commons package allows the user with many functionality, configuration and utility packages. One of the packages is Knuth’s Dancing algorithm were “generic model for problems, such as tile placement, where all of the valid choices can be represented as a large sparse boolean array where the goal is to pick a subset of the rows to end up with exactly 1 true value in each column” (Apache Hadoop). Many other packages such as client, ssl, server, http parser’s, and hash fulfil as vast list of usable API. A commonly used API package is the Rumen pre built analysis tool. An example taken from (Apache Hadoop) shows how this data extraction tool can be implemented. The Rumen is written in JSON and uses a custom Jackson Library to implement its features.

JobConfigurationParser:  
A parser to parse and filter out interesting properties from job configuration.   
  
**Sample code:**  
 // An example to parse and filter out job name  
   
 String conf\_filename = .. // assume the job configuration filename here  
   
 // construct a list of interesting properties  
 List<String> interestedProperties = new ArrayList<String>();  
 interestedProperties.add("mapreduce.job.name");  
   
 JobConfigurationParser jcp =   
 new JobConfigurationParser(interestedProperties);  
  
 InputStream in = new FileInputStream(conf\_filename);  
 Properties parsedProperties = jcp.parse(in);

The other two sections known as MapReduce and Yarn contain less packages but contains the core Hadoop’s main functions. The MapReduce API has packages gridmix, jobcontrol, lib.db, mapreduce, jobtracker, tasktracker and much more. This set of packages allows the manipulation of data within the grid for better sorting and less file cluster. The mapreduce package implements a queue and stack to fully implement the cluster within the data set. The last section known as Yarm is part of the Apache Software Foundation (Apache Hadoop). Yarm is a special operating system built as a data negotiator for Hadoop. It is the main reason Hadoop has many advantages over other big data filesystems. MapReduce and Yarn are at a stable build 2.0. Yarn is implemented through many packages using the NodeManger and has many functions to be control resources within the library (Apache Hadoop). The whole MRV2 API contains a major feature that allows backward compatibility to hadoop 1.xx releases. Which is a big advantage as many new filesystems do not offer this.

**Other DFS**

The HDFS file system has been at the top of industry for many years but the major problem now is how to move forward, and if HDFS can not move forward other DFS’s are looking to take its place. There are many alternatives to the HDFS file system the top being Cassandra (DataStax) which offers fast data analysis and access (Harris, 2012). Even though HDFS has been improving in performance and reliability it seems to have plateaued. Other open source software has risen to show many new ways of handling large file systems. Another alternative being Ceph which “features a high-performance parallel file system that some think makes it a candidate for replacing HDFS (and then some) in Hadoop environments” (Harris, 2012). The major key changes in many alternatives is the parallel processing. This would mainly rely on the performance of the hardware. HDFS has been widely used due to its pairing and performance under many different hardware configurations. However as new hardware is released the software must be kept on par.

IBM has released their own file system computing software in the recent years and claims to be one of the best in the industry. GPFS is a Parallel File System to high performance computing consumers one main advantage it has over HDFS is “much faster than Hadoop in part because it runs at the kernel level as opposed to atop the OS like HDFS” (Harris, 2012). This shows that HDFS even at top of its game there are alternatives than can be faster and more reliable. This does not mean HDFS is a bad product but it can make improvements. A very funded file system is the MapR File System is claims to be 20 times faster than HDFS and provides more features such as mirroring and extensive availability (Harris, 2012). There are many alternatives to HDFS which are being used in the industry but many companies still prefer to use HDFS due to its highly functioning API and massive pairing support with almost any architecture.

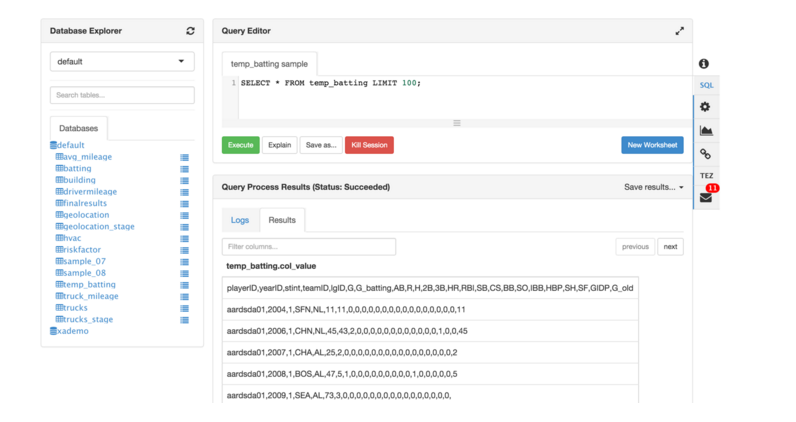
**State of the Art**

The contemporary era of technology today, centers itself on the management of big data. Therefore, there are many big data management and file systems out there. Hadoop’s state of the art approach to tackling file management problems is unparalleled. According to the company website of SAS, Hadoop is able to provide storage for massive amounts of data and excels in handling concurrent tasks. (Hadoop What is it and why does it matter?) Hadoop’s state of the art is derived from four main performance categories, open-source software, framework, massive storage, and processing power. (Hadoop What is it and why does it matter?) These terms describe Hadoop’s main capabilities when it comes to managing data and performing jobs. However, many other file systems share the same characteristics. What separates Hadoop Distributed File Systems from its respective market segment are; its unparalleled computing power, flexibility, fault tolerance, low cost, and scalability. (Hadoop What is it and why does it matter?) Furthermore, data security is of critical importance within modern times and HDFS provides just that. In the article, “Hadoop, Spark, Cassandra, Oh My!,” it states, “Hadoop is maturing fast from the security perspective.” (Busch, 2015) Hadoop is able to provide security on a commercial level do its large frameworks and scalability. (Busch, 2015) As proof of Hadoop’s state of the art functionality, many vendors support it, as well as open source Apache products. (Busch, 2015) HDFS is the clear and logical choice when picking a state of the art file system.

**Applications**

Hadoop Distributed File Systems have a wide array of contemporary applications. Due to HDFS being able to lower costs to analyze big data, the door opens to many potential uses of HDFS. (Hadoop Applications) In the article, “7 real-life use cases of Hadoop,” Stangarone explains that Hadoop has many real life applications, one of which include being able to identify security issues. (Stangarone 2015) Using Hadoop’s ability to analyze and manage file systems, Hadoop is able to provide early detection of security breaches. HDFS can also be utilized for corporations to increase their bottom line. Companies rely on the collection and analyzation of big data to market their products. However Hadoop has a unique ability to tell companies when is the best time to sell their products, another vital application of HDFS. (Stangarone 2015) This can be achieved by analyzing product sales data with an array of factors that portray their effect on sales. (Stangarone 2015) Pertaining to business applications as well, Hadoop is able to effectively to target consumers based on big data collections. (Stangarone 2015) These collections can include geographic location, median income, etc. Collection and Analyzation of this data is made possible by HDFS. (Stangarone 2015)

All applications listed for HDFS, have been general uses. However there are many specific, real life applications to which HDFS has provided the basis for. Hive and Pig are examples of applications based on the Apache Hadoop system. (HORTONWORKS) According to HORTONWORKS, “Hive provides a database query interface to Apache Hadoop.” (HORTONWORKS) Hive is mainly used to store data, while Pig is used to search and filter through the data. If Apache Hadoop did not provide the base for these applications then they would not be able to serve their functions. The figure below represents a Hive Table being utilized to access Big Data.



***Figure 1:*** Hive Table Interface (HORTONWORKS)

**Advantage and Disadvantages**

Through learning about the applications of Hadoop Distributed File Systems, we can see how versatile the platform is and how it meets the demands of many industries. However, due to competing platforms in the industry there have been many companies pointing out the disadvantages of Hadoop in order to raise questions. In order to understand this situation, it is important to analyze the advantages, disadvantages of HDFS, and compare it to its competing platforms.

By researching into file systems and focussing on the Hadoop architecture, there are many advantages this new system has brought to many industries. According to an article in the Hortonworks blog, hadoop provides “Extreme low cost per byte, Very high bandwidth to support MapReduce workloads, and Rock solid data reliability” (Baldeschwieler, 2012). These are just some of the main advantages that are considered when discussing HDFS. As discussed, HDFS provides affordability which protects the company's resources to spend on other problems at stake. Also by being an open-source software it provides developers with a free environment and less costs, this attracts companies as well, as they can implement their own infrastructure using the HDFS open-source platform. This also provides increased compatibility with the large amounts of data that is processed. Another advantage is improved bandwidth which means the data is processed at a high speed which provides companies with fast results. Hadoop can read “2+ gigabits per second per computer … Hadoop clusters can easily read/write more than a terabyte of data per second continuously to the MapReduce layer (Baldeschwieler, 2012). This improvement is key to why Hadoop is the leading platform for data analysis. Another advantage, that plays an important role for Hadoop is the reliability of its infrastructure when managing data. Hadoop is designed in a way that proves its reliability, “designed from the ground up to correctly store and deliver data while under constant assault from the gremlins that huge scale out unleashes in your data center” (Baldeschwieler, 2012). Hadoop has proven its reliability over the years through the usage of many consumers as it keeps growing and evolving. These advantages show how HDFS is the key leader in Big Data analysis.

However, due to competing infrastructures some disadvantages have been pointed out that need to be understood in relation to HDFS. According to the article in the computer business review, the author discovers some disadvantages presented by HDFS, which include Hadoop complexity in choosing the right infrastructure as there are many variations, difficult to implement the infrastructure because of not being able to find the right use case, there is a difficult of learning the system in order to implement it “skills gap”, and finally the Hadoop architecture is not as interactive (Nunns, 2016). These disadvantages have been exaggerated in the field of Big Data as people don’t know how to deal with the new system and implement the architecture in terms of their needs.

These competing infrastructures that have pointed out the disadvantages of Hadoop don’t provide a good fight against it. It is important to understand the competing companies and how they are implementing their software in order to manage data and if they are valid to speak out against HDFS, the leader in Big Data Analysis. According to the article by Eric Baldeschwieler, he points out that the competing companies do not scale up to the infrastructure HDFS provides, they are not open source, and they are unproven technologies (Baldeschwieler, 2012). This validates the argument that the companies competing with Hadoop are not strong competitors. Analyzing the issue of scalability by companies to the hadoop standard , these companies cannot deal with large data as hadoop can and also they have problems providing reliable results from their implementation of data management. Another major reason, how hadoop compares to competing companies is that it is open-source making it more cost effective and easier to have the necessary resources to implement, which the competing companies do not. Finally, the competing companies do not have the experience to prove that their system is reliable and better than HDFS. These competing companies do not have the power to defeat the leading industry big data analytics file system HDFS.

**The Future of Hadoop Distributed File Systems**

The next big wave for Hadoop data storage is providing an interface for programming entire clusters with implicit data parallelism and fault-tolerance. The first wave was about establishing technologies and making sure a good foundation was built so that individuals could build open source applications on top of the core program. Apache Spark is said to be the future of Hadoop, and is built off the whole HDFS software. Spark is an open source clustering computing framework; it can “provide programmers with a programming interface focused on data structures called resilient distributed dataset (RDD)” (Apache Spark, 2016). The RDD is maintained in a fault tolerant way through a read-only multiset of data items scattered over an array of machines. It was developed because of limitations in the “MapReduce” cluster computing model, which forces dataflow structure on distributed programs. The Spark program is currently being improved by a company called Cloudera who believe Spark is “great for specific use cases to being the default engine for MapReduce workloads as well as partner products” (Woodie, 2015). Cloudera will seek to improve Spark in four areas: security, scale, management, and streaming. By improving these main areas of the program, Spark can evolve into a powerful file system that can manage and organize a constant stream of live data. Ultimately Spark will be the core engine managing streams of data, phasing MapReduce out since it will become obsolete and unable to keep up with the speed of Spark. In conclusion, Spark is in the works of becoming the next core filesystem engine to handle the large amount of data found on computer networks, and will be able to quickly manage and store the information faster than any current file system used in today’s industry.

**Conclusion**

Apache Hadoop has been created out of the necessity for better file storage and management for businesses and other various applications. Hadoop has been shown to have many advantages compared to other distributed file systems such as its use of MapReduce as well as its fault tolerance. Despite some disadvantages such as the complexity of Hadoop and the various specific use cases, these nuances really highlight the many uses and the specificity of these cases allow Hadoop to be even more effective. With Hadoop being as established as it is, the future of Hadoop only opens up more functionality for the future.

Additionally, Apache Hadoop is considered one of the premiere distributed file systems on the market today due to its power and efficiency. Its use with commodity machines also allows it to be extremely flexible and work in a wide variety of workplaces. Using these commodity computers also enables Hadoop to be fault tolerant which makes it even more enticing to use. As the use of computers increase within not just businesses but everywhere, so increases the amount of data to be dealt with. We are in the era of Big Data and in order to handle these ever increasing sums of data, we must be more efficient and more powerful, and this can be accomplished through Apache Hadoop.

**Bibliography**

Apache Spark. (2016, April 18). Retrieved May 04, 2016, from https://en.wikipedia.org/wiki/Apache\_Spark

The Wikipedia page for Apache Spark gave a great overview of what the program. An individual reading from the web page can quickly get a great understanding of the Spark program and for what purpose it was developed. By reading through a few paragraphs, it was clear what Apache Spark does and how it is able to execute its tasks. In short, this Wikipedia web page will allow an interested reader to swiftly grasp the concept of Apache Spark and how the program manages data based off the Hadoop Distributed File System software.

Apache Hadoop Main 2.7.2 API. (n.d.). Retrieved May 05, 2016, from https://hadoop.apache.org/docs/stable/api/

A website that contains the full documentation of Apache Hadoop’s API. Here the packages are listed by class and broken into different tree indexes. Also the deprecated API is listed in case an older version of Hadoop is being used.

Baldeschwieler, E. (2012, June 25). HDFS vs. Other Storage Tech: Benefits and Advantages of HDFS. Retrieved May 5, 2016, from <http://hortonworks.com/blog/thinking-about-the-hdfs-vs-other-storage-technologies/>

The purpose of this article is to provide both the benefits and advantages of HDFS and provide a comparison of it to other file systems. The author does a great job in defining the advantages of using HDFS and why it has gained so much popularity in the field of Big Data. The article also points out the competition against HDFS and provides reasons why HDFS is the better choice.

Busch, B. (n.d.). Hadoop, Spark, Cassandra, Oh My! Retrieved May 5, 2016, from

https://blogs.perficient.com/dataanalytics/2015/06/17/hadoop-spark-cassandra-oh-my/

The purpose of this article, is to explain the basic utilizations and functions of Apache Hadoop. Furthermore, this article explains the commercial and enterprise applications of Apache Hadoop.

Diaz, A. (2014, September 26). The Evolution of MapReduce and Hadoop - DZone Big Data. Retrieved May 05, 2016, from <https://dzone.com/articles/evolution-mapreduce-and-hadoop>

The purpose of this article by Adam Diaz, is to explain the changes brought to the field of data analysis by analyzing its two major "tools" Hadoop and MapReduce. Diaz articulates the changing landscape and the evolving Hadoop Architecture, an implementation of MapReduce. By analyzing its improvements through YARN (Yet Another Resource Negotiator) and Apache Spark, which are the major improvements to the functionality of Hadoop. After reading through the article, this source has very valuable information in relation to how Hadoop has grown as a distributed file system, through the years.

Hadoop Applications. (n.d.). Retrieved May 05, 2016, from

https://www.talend.com/resource/hadoop-applications.html

This article provides an array of applications of the Hadoop system.

Harris, D. (2012, June 11). Because Hadoop isn’t perfect: 8 ways to replace HDFS. Retrieved May 05, 2016, from <https://gigaom.com/2012/07/11/because-hadoop-isnt-perfect-8-ways-to-replace-hdfs/>

This article discusses the viable alternatives to HDFS and provides a reasonable description as to why other filesystems can be used in place of HDFS.

Hive Tutorial: How to Process Data with Hive. (n.d.). Retrieved May 05, 2016, from

http://hortonworks.com/hadoop-tutorial/how-to-process-data-with-apache-hive/

Provides an array of applications of the utilization of both Pig and Hive. Provides tutorials and tables on the usage of Hive and Pig.

Nunns, J. (2016, February 26). 5 Hadoop problems and how to fix them. Retrieved May 05, 2016, from <http://www.cbronline.com/news/big-data/software/5-hadoop-problems-and-how-to-fix-them-4823399>

Stangarone, J. (2015). 7 real-life use cases of Hadoop. Retrieved May 05, 2016, from

http://www.mrc-productivity.com/blog/2015/06/7-real-life-use-cases-of-hadoop/

Stangarone talked about real life cases of Hadoop. He provided an in depth analysis of how Hadoop allows real life applications.

What is Hadoop? (n.d.). Retrieved May 05, 2016, from

http://www.sas.com/en\_my/insights/big-data/hadoop.html

Provides background information on Hadoop and how it separates itself from its respective market segment. Describes its state of the art capabilities and its advantages over other file systems.

The purpose of this article is to provide the disadvantages brought up against the Hadoop Distributed File System. The author does a great job in common arguments against HDFS and provides concise detail explaining each problem and providing a solution to it. The input this article provided was explaining the disadvantages of HDFS the issues companies are having with the implementation of it.

Proffitt, B. (2013, May 23). Hadoop: What It Is And How It Works - ReadWrite. Retrieved May 03, 2016, from http://readwrite.com/2013/05/23/hadoop-what-it-is-and-how-it-works/

The purpose of this article was to give the reader a broad overview of the Hadoop Distributed File System. The article gave a good explanation of why the Hadoop program was created and how it processes the large amounts of data found throughout the computer network. The article also discussed the main components of the program and briefly described what each component was responsible for. In short, the article gives the reader a general explanation to the purpose behind Hadoop Distributed File System, which is to quickly and cheaply store large amounts of data.

Reimer, J. (2008). From BFS to ZFS: Past, present, and future of file systems. Retrieved May 03, 2016, from http://arstechnica.com/gadgets/2008/03/past-present-future-file-systems/2/

The main idea of this article from ars technica was to give the reader insight on how the first filesystem was created as well as discussing the purpose of a filesystem in a computer. The article points out the past, present, and future of file systems. The purpose of the article is to give individuals an in depth history lesson on filesystems and how they have revolutionized the way we use computers today. Overall, the article does a great job going into detail about the very first filesystem that was created in 1973 as well as discussing the future of file systems and how they will aid individuals with storing through the large amount of data currently found on computer networks today.

Sivaraman, E., & Manickachezian, R. (2014). High Performance and Fault Tolerant Distributed File System for Big Data Storage and Processing Using Hadoop. *2014 International Conference on Intelligent Computing Applications*. doi:10.1109/icica.2014.16

The purpose of this article is to present an overview of the power and effectiveness of Apache Hadoop. The article gives a brief overview of Big Data and Hadoop. It also presents the two aspects of Apache Hadoop that make it so effective: Hadoop Distributed File System as well as MapReduce. Lastly, the authors cover some scenarios where employing Apache Hadoop would be an effective way for a particular company to store their data.

Woodie, A. (2015, September 09). Spark Is the Future of Hadoop, Cloudera Says. Retrieved May 04, 2016, from http://www.datanami.com/2015/09/09/spark-is-the-future-of-hadoop-cloudera-says/

The main idea of this article written by Alex Woodie from Datanami was to inform the reader of the next steps to improve Apache Spark. The author gave a good overview of where Spark came from and mentions what is instore for the program. The main idea of the article explains how Cloudera has taken over the open source Spark software and is currently making improvements, which will allow the software to become the next core filesystem engine for processing large amounts of data. Overall, the article is a good source for understanding what will become of Apache Spark as well as the steps that are required to making it a very powerful filesystem to be able to handle storage and management of streamed data.