Open Access

Open Journal of Big Data (OJBD)
Volume 1, Issue 2, 2015

www.ronpub.com/ojbd
ISSN 2365-029X

Data Transfers in Hadoop: A Comparative Study

Ujjal Marjit [A], Kumar Sharma [B], Puspendu Mandal [B]

[A] Center for Internet Resource Management (CIRM), University of Kalyani, Kalyani-741235, Nadia, West Bengal, India, marjitujjal@gmail.com

[B] Department of Computer Science and Engineering, University of Kalyani, Kalyani-741235, Nadia, West Bengal, India, kumar.asom@gmail.com, mandal.puspendu@gmail.com

ABSTRACT

Hadoop is an open source framework for processing large amounts of data in distributed computing environment. It plays an important role in processing and analyzing the Big Data. This framework is used for storing data on large clusters of commodity hardware. Data input and output to and from Hadoop is an indispensable action for any data processing job. At present, many tools have been evolved for importing and exporting Data in Hadoop. In this article, some commonly used tools for importing and exporting data have been emphasized. Moreover, a state-of-the-art comparative study among the various tools has been made. With this study, it has been decided that where to use one tool over the other with emphasis on the data transfer to and from Hadoop system. This article also discusses about how Hadoop handles backup and disaster recovery along with some open research questions in terms of Big Data transfer when dealing with cloud-based services.

TYPE OF PAPER AND KEYWORDS

Research review: Hadoop Distributed File Systems, *HDFS, HDFS Import, Big Data, Hadoop, Flume, Sqoop, Scribe, Kafka, Slurper, DistCp.*

1 INTRODUCTION

Big Data is an extremely large dataset consisting of variety of data including text, images, multi-media and logs. The volume and velocity of such data is so huge that the traditional data processing applications are incapable to store and process. Data storage and analysis are two main aspects that Big Data is mostly associated with [31]. Nowadays, most organization's crucial data are stored in relational databases. A large amount of data is being stored into various types of file systems for better manipulation, especially for analyzing and reporting. Data are extracted, filtered and loaded into some sort of data warehousing tools and then data mining techniques are applied to retrieve the useful information. However, unless these file systems are distributed in nature the challenges remain as they are in terms of storage and computation.

Usually, data in general file systems and relational databases are stored into single hard disk. Such types of storage system fail when data grows in volume and velocity. These systems should be able to compute and analyze huge amount of data in parallel. Using single processor, it is difficult for doing this or it takes longer than days unless the storage and processing capacities are increased by many factors. Now, the real problem

is in analyzing and extracting useful information from variety of voluminous data. Instead of increasing storage and processing capacities in general data processing systems, a distributed technology has been emerged. This technology is known as Hadoop [10], a distributed storage and processing of very large amount of datasets across clusters of commodity hardware. Considering the storage and processing power of Hadoop, many organizations have started data migration from traditional data storage systems into Hadoop Distributed File Systems (HDFS).

In real world, data is being generated from various sources such as web server, social media applications, and sensor devices. These sources generate several petabytes of data in a small fraction of time. Processing and analyzing such data is not possible for general processing systems. Rather, Hadoop commands have been used to move data into HDFS for processing and analyzing by MapReduce paradigm [27] [35]. Moreover, some commonly used tools are used to move data into HDFS. Though, we can write MapReduce programs for moving data. However, writing MapReduce programs need considerable amount of time, efforts and many other factors, such as no data loss or corruption should take place and no duplicity should occur. Instead of writing MapReduce programs manually, we can use existing tools to load data into HDFS. As these tools save time and effort by focusing only on the data manipulation and analyzing instead of developing tools by writing complex programs.

The size of the Big Data always ranges from terabytes to exabytes exceeding the capacity of general processing systems. In view of this, Big Data suffers mainly from three types of challenges – processing, storage, and data transfer [38]. Processing a huge amount of data needs parallel processing systems and complex analytical algorithms. Hadoop already has its own and overcomes storage as well as processing challenges with the help of HDFS and MapReduce components. Hadoop can store terabytes/petabytes of data with very low storage cost per byte and process large amount of data using parallel computation. MapReduce can process data in the stored data-block itself reducing the time in transmitting data for processing. However, the actual problem occurs during transferring Big Data into the cloud [23]. Since volume is the primary aspect of Big Data which concerns the size of the dataset. The larger the dataset size is, the longer will take to extract, transfer and load data into analyzing tools.

Additionally, Hadoop ecosystems require new technologies and architectures to be deployed with latest hardware configuration [23]. Such type of configuration is cost effective except for large organizations. Hence, most of the users go for cloud-based services such as Cloudera, Hortonworks, Amazon Web Service, and Google Cloud Platform etc. Data transfer is a critical task when dealing with cloud-based services. It entirely depends on the network latency, file size and transport mechanisms. However, there are dynamic benefits of cloud-based services. Such that, cloud-based service provides improved efficiency, matured technologies, fast and reliable backup services, suitable for handling large files, and provides emergency recovery options [40]. In this article at the outset, concentration has been given how each tool performs data imports and then we leave problem of data transfers as open research question.

Another common challenge is backup and disaster recovery. In case of Hadoop, this is critical aspect as it deals with very large amount of data [32]. For any organization, the data is integral part of the business, however, very less researchers concern about backup and recovery. Data must be protected against any sort of disastrous situation. Due to the exponential data growth in organizations, the issue of backup and recovery is becoming challenging. There is constant growth of the data, 20-30% annually, which is affecting the manual backup and recovery processes. This is not only because of volume; velocity and variety are also playing their role in bringing complexities in the data. Thus, Big Data needs dynamic backup and recovery solutions. Such that, the storage system should be scalable and centralized (cloud-based) and multi-purpose based solution [41]. Farther, the recovery process must be smooth so that organizations can continue their activities without affecting ongoing tasks.

In this article, first we discuss the various components of Hadoop and then go into how these components assist in backup and recovery processes. We then discuss various tools for importing/exporting data in Hadoop environment along with their characteristics such as the nature of data and goal of data transformation. Side-by-side comparisons and recommendations are shown to guide the user for taking the decision where to use one tool over the other for importing data into HDFS.

The rest of this article is structured as follows. It begins with motivation of our study in section 2. In Section 3 an elaboration of Hadoop architecture and its application domain is presented. Further, Section 4 presents the backup & disaster recovery mechanism in Hadoop, section 5 deals with the taxonomy of data transformation; in Section 6 we discuss different tools, characteristics and their use cases. Section 7 presents the discussion and finally, Section 8 concludes the article.

2 MOTIVATION

Relational Database Management System (RDBMS) mainly helps in improving productivity applications [24]. Because of its ability to handle complex queries it remains the primary storage system for many organizations. However, for analyzing purposes, many times the database management system fails due to high input and heterogeneous nature of data. Hence, it becomes need of smart applications whose processing architecture is distributed in nature. Heterogeneous nature of data is found mainly in log data generated by frequently used social media and web applications. Different kinds of log data exist that serve the basis for extracting useful information [30] and interesting patterns [25] using mining approaches. Again, the rate of log generation has been increasing rapidly, because of which general mining applications fail to manage such log data.

As we can see, with increase in technologies and electronic devices the Internet users are growing constantly. Internet users all over the world do lot of searching, browsing, sharing and posting several kind of data from one end to another. With growing number of Internet users the rate of data transfer has also been increased rapidly. As data grows, there comes affect in performance, data management and analyzing. The more data becomes voluminous; the data management becomes fairly difficult. It implies that without using distributed applications such type of data remains out of care. Therefore, Hadoop is the best way for handling voluminous data. However, to achieve the outcome using Hadoop, first of all, data needs to be migrated into HDFS. For this, we need to explore how the data migration job is executed and what different tools are exists for doing this. These tools have been highlighted in the following sections.

3 HADOOP ARCHITECTURE AND ITS APPLICATION DOMAINS

Hadoop is an open-source software package which supports distributed processing of large datasets on clusters of machine. It consists of mainly two core components: HDFS and MapReduce. HDFS is the storage component, whereas MapReduce is a distributed data processing framework, the computational component of Hadoop. Both HDFS and MapReduce are master-slave architecture consisting of master and slave node having different roles. HDFS master node is called the Name-Node responsible for managing names and data blocks. Data blocks are present in the Data-Nodes, the slave component of HDFS. Data-Nodes are distributed across each machine, responsible for actual data storage. MapReduce master node is called the Job-Tracker responsible for scheduling jobs on Task-Trackers. Task-Tracker again is distributed across each machine along with the Data-Nodes, responsible for processing map and reducing tasks as instructed by the Job-Tracker.

Apart from HDFS and MapReduce, HBase, Hive and Pig are important components of Hadoop. HBase [11] is a distributed and scalable data store that can host billions of large tables having millions of rows and columns. Hive [14] [15] is a data warehousing software package built on top of Hadoop. It provides SQL like user interface for writing HQL (Hive Query Language) queries for querying and managing large datasets stored in HDFS. Pig [17] [22] is a platform for writing MapReduce programs. Pig programs are useful for analyzing large datasets that run in parallel. It uses Pig Latin [21] programming language, a textual based language, which supports parallel execution of data flows.

The key characteristic of Hadoop is that we do not have to pre-define the data schema before loading data into HDFS. Regardless of the format of data (structured, semi-structured and unstructured) we can load data into HDFS as it is. Also, data pre-processing such as cleansing, normalization & aggregation can be done on HDFS itself after loading the data. Using Hadoop, business domains can reduce operational cost, infrastructure cost, avail data storage and new analytical models [10].

The rule of thumb of Hadoop is that “throw more nodes at the problem”. Since, the slave nodes are scalable, which can be scaled to any number of nodes as per the requirement. Apache Hadoop provides cost-effective and massively scalable platform for ingesting Big Data and preparing it for analysis. It reduces time to analyze data by hours or even days. Taking the advantage of distributed storage and computations, Hadoop is being used by many companies for data analytics, behavioral analysis & targeting, clickstream analysis, log data aggregation, information extraction, web data mining, machine learning, RDF graph processing, search engine optimization, data filtering, session analysis, events processing and many more. The Industries who use Hadoop are China Mobile, Yahoo, Facebook, LinkedIn, NetFlix, IBM, Twitter, Zynga, Amazon, Accela Communications, Adobe and many more.

4 BACKUP AND RECOVERY MECHANISM IN HADOOP

For any information processing system any kind of disastrous situations may arise. Some of the common situations are hard disk failure, node crash, rack failure,

data corruption, permanent loss of system (e.g., natural disaster), network and power loss. The disastrous situation may not be same at all locations. However, the system must be fault tolerant against any kind of failures. Hadoop provides an effective way for storing large files distributed across cluster of nodes. Each such node consists of processing and storage power of more than terabytes. Files are divided into blocks and each block is distributed across nodes along with their replica. By default, data blocks are replicated to three nodes, ensuring data reliability, high availability and fault tolerant [12].

Hadoop also maintains active and passive (primary and secondary) Name-Nodes. Additionally, a zookeeper and metadata services are used to coordinate between primary and secondary Name-Nodes. In case the primary Name-Node fails, the secondary Name-Node takes over the role of primary Name-Node and zookeeper service ensures that both nodes are in sync always. Metadata is a special factor in data backup, which routes datasets to specific Data-Nodes keeping various information such as block size, replication factor, list of blocks, list of Data-Nodes, acknowledgement (ACK) package, checksums, and the number of under replicated nodes [12]. All these information take respective role during data backup and recovery. For example, checksum is used to detect the data corruption. When data corruption occurs checksums are verified and subsequently data is resent. This way Hadoop provides fault tolerant. There is no way of losing data even in case of hardware failures or power loss.

Further, the snapshot-based solutions also exist to handle data backups. S. Agarwal et al. [36] present the selective-copy-on-appends solution for taking snapshots of HDFS. Sometimes the Job-Tracker may fail to execute its job and stops permanently. In such situations, the Job-Tracker has to be recovered to its last state and resume the job without loosing results. Snapshot based solution as presented in [34] prevents the failure of Job-Tracker, which is based on checkpoint & recovery method. Checkpoint & recovery method periodically takes the snapshot of current state of the Job-Tracker and stores into distributed storage (file systems). When the failure situation occurs, a new Job-Tracker is created using the last state restored from the snapshot and continues execution of the job.

Furthermore, the problem of loosing data remains as it is in case of natural disaster situation such as flood, earthquake, or fire. In such situations, the single location-based for data backup fails and data may not be recovered. At this condition the efficient and cloud based data backup & recovery solution is required. Since, cloud-based solution provides fast, reliable and immediate backup & recovery facilities. A multi-purpose based data recovery has been presented in [41]. The multi-purpose approach offers the data to be restored from multiple locations using multiple techniques. This solution provides a strong recovery process because of having options for restoring data from multiple locations and using multiple techniques. Hence, Hadoop is fault-tolerant data storage and processing system, which can protect data from hardware or system failure automatically [42]. That means the data is not lost even if the entire node fails or goes down.

5 TAXONOMY OF DATA TRANSFORMATION

Different tools exist for transferring variety of data and serving specific task. It is essential to understand about what, how and why these tools transform data from one source to another. In this section, we present the taxonomy for understanding different tools, which is based on the nature of data being transformed, mode of data transfer, hardware and operating system platforms, user interface, reliability and fault tolerant. The summary of each of them are given below:

a) *Nature of the data*:
Today, most of the web data are stored in RDBMS and text based file systems. Almost 80% of the web applications are dominated by relational data. Organizations and companies are constantly using their database systems, mostly, the data generation systems. The speed of such data generation systems is being increased. Because of which, RDBMS may fail at one point when the volume of data becomes bigger than the capacity. Again, there exist different applications, which are constantly generating log data. Such applications are weather forecasting applications, social networking sites, web servers etc. Analyzing of these log data is essential in different use cases. We mostly study the tools for transferring RDBMS data, log data, local files systems, and copying data from different HDFS clusters.

b) *Goal of Data Transfer*:
There are many tools, which perform similar kind of job. However, they are different in many use cases though they may take same data input. Some of their job is to collect log data and store into different formats, others are processing and analyzing etc. It is important to understand what exactly their goal is, so that it becomes clear understanding before using them.

c) *Mode of Data transfer*:
Once data has been imported into HDFS, sometimes, it is essential to export data from HDFS in the same format as it was in the source. Since, HDFS stores data in different format, making it compatible for

MapReduce to process. However, there exist some tools that assist the job of data importing and exporting. These tools are intermediate between users and Hadoop ecosystems.

d) *Hardware and Software Requirement*:
The Hadoop ecosystems especially run in Linux Platforms. Initially, the Hadoop team has focused development and testing of Hadoop on Linux platforms. This is basically a barrier for windows users. Indeed Hadoop runs on Mac and Windows platforms, however, the operations are not popular on these platforms and users may face challenges in integrating Hadoop applications. Everyone is not familiar with Linux platform, for which it becomes difficult in availing the Hadoop benefits. The Hardware requirement describes the amount of RAM, CPU speed and the disk size required for Hadoop tools. Here, we review whether the discussed tools can be run in cross-domain platforms meeting minimum hardware requirements.

e) *User Interface*:
User interface is the main factor by which users are able to learn the goal of application quickly. In case of writing commands in some sort of tools, such as, command line, often users find confusions on remembering commands and writing them in proper order. Command line tools are more suitable for highly technical users. Hence, user interface provides easy to follow and learn quickly, as it guides step by step about how to proceed to the goal.

f) *Fault Tolerant and Reliability*:
The fault tolerant is defined as how the system tolerates during failure. There can be several failures such as software crash, disk failure, and power loss and bus error [9] [18]. How the system behaves, what happens to the state of the data or whether the system remains operational after a failure. This is called reliability. For a foolproof software tool, it should be able to tolerate the failure urges and should remain operational. This is what top clients want focuses on this feature to get the right software tool. The data should remain persistent either in the changed state or initial state.

6 IMPORTING AND EXPORTING DATA IN HADOOP

In our study, we have encountered a couple of tools for importing and exporting data in and out of Hadoop. However, providing a complete list of tools is out of scope of this article. Here, we present well-known tools as well as relevant to us. These tools include Sqoop, Flume, Chukwa, Scribe, Kafka, HDFS File Slurper and DistCp. These tools are mainly used for moving structured data; log data and files to HDFS. In the following sub-section we have described them completely.

6.1 Moving Structured Data with Sqoop

Sqoop [33] [1] is a connection oriented, non-event based and open source software program for moving data between structured data stores and distributed file systems. Sqoop mainly used for moving structured data (relational tables) stored in MySQL, Oracle or Microsoft SQL Server databases on a periodic basis as shown in Figure 1. Import is performed by reading tables row-by-row and writing records into multiple delimited-text or Sequenced files in HDFS. Here, each row of a relational table is mapped to a record in HDFS. Map-only jobs are applied to select or insert data from RDBMS. Multiple tables can be used to import data. Data from each table is stored into separate directory in HDFS. It uses Map-Reduce programming paradigm for parallel processing and fault-tolerance. Sqoop consists of a set of command-line tools, which can be used independently based on the requirement.

For example, for importing data, “sqoop-import” tool is used and for exporting data from HDFS to RDBMS “sqoop-export” is used. Apart from import and export, it also supports saved jobs, merging datasets and generation of Java classes that encapsulates the imported record in it. Mainly distributed agents have been participated in importing and exporting data. Microsoft Inc. uses Sqoop for transferring data from MS-SQL to Hadoop. Additionally, Sqoop is used for importing data into Hive and Hbase from the database systems that has JDBC feature enabled. Sqoop supports Linux operating systems and it can only run where Hadoop libraries and configuration files are installed on the machine [33] and requiring JDBC drivers installed separately. To start importing, database has to be configured using “sqoop-import” command following the arguments.

6.2 Moving Streaming Data using Flume

Apache Flume [37] is used for importing event-based data into HDFS. Unlike Sqoop, it is one-way data collection, i.e., only importing. Apache Flume is a standard, straightforward, robust and flexible tool for streaming data ingestion into Hadoop. This Apache project has received incubator status in a year later in 2012, but originally developed by Cloudera. It is mainly consists of a set of agents, each having an instance of the Java Virtual Machine (JVM), requiring at least three components such as Flume Source, Flume

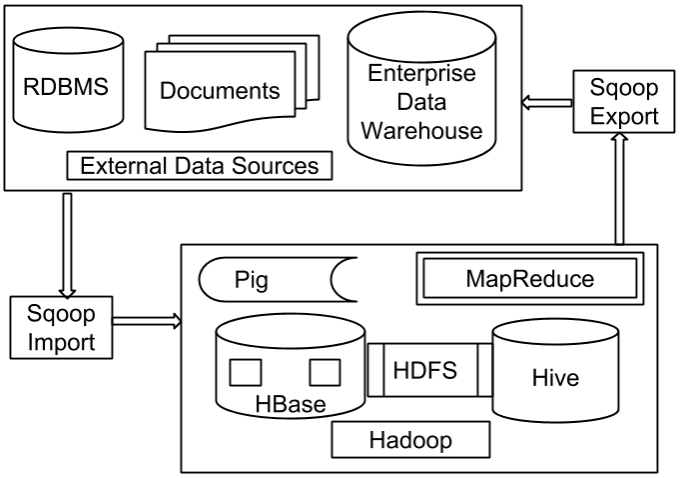

Figure 1: Apache Sqoop Architecture

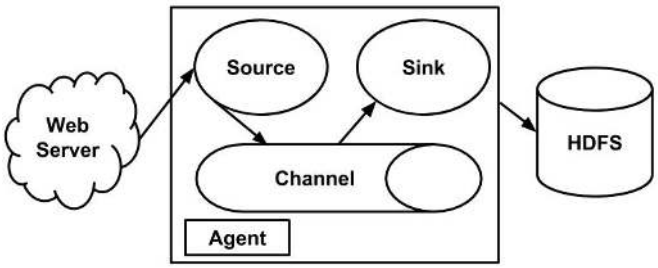

Figure 2: Apache Flume Architecture (Source: http://www.flume.apache.org)

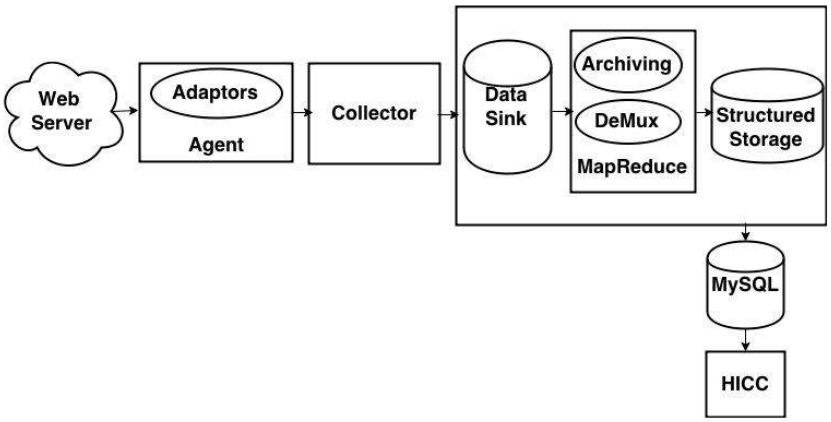

Figure 3: Apache Chukwa Architecture (Source: https://chukwa.apache.org)

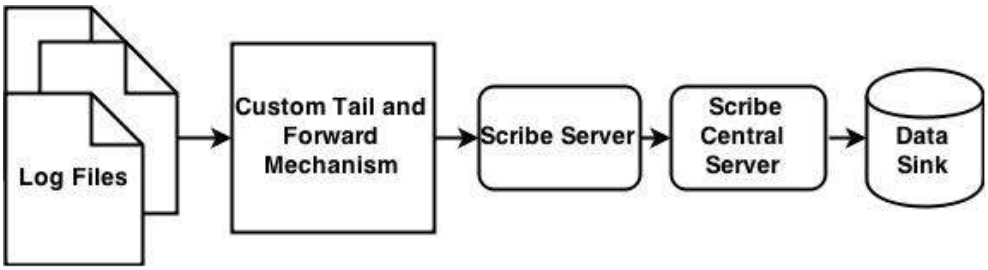

Figure 4: Scribe Architecture

Channel and Flume Sink as shown in Figure 2. Flume Source is responsible for collecting incoming streams as events that are passed to the Flume Sink via Flume Channel. Flume Channel uses in-memory and disk queues for storing events. Flume Sink later removes events from Flume Channel after writing data into HDFS files. Flume is robust, self-contained, reliable, fault tolerant and event-based tool, which also supports filtering events by enabling multi-hope events transmission.

Flume has been developed to handle the streaming logs and data. But the data is being changed time to time from the system. It is difficult to batch the data due to the dynamic nature. Configuring flume agents is an easy task that can be written in Java programming language. We can build custom sources by using Facebook [4] or Twitter [8] APIs for receiving data from each end and having provisions for sending received data to the Flume Channel and Flume Sink [39].

6.3 Moving Data with Chukwa

The main objective of Chukwa, as shown in Figure 3, is similar to Flume having slight differences in monitoring and analyzing large distributed systems as well as reliably delivering data [26]. The main goal of Chukwa is to support fault-tolerant [13]. Chukwa is built on top of Hadoop so as to enable all the features of Hadoop are available to it. Chukwa is comprised of Agents, Collector and Map-Reduce. It is very much similar to Flume Source, Agents are responsible for collecting logs from different sources and the Collector receives logs from Agents and stores into local disk. Finally, the Collector sends received logs to the Map-Reduce component for archiving and storing logs into HDFS. Chukwa is an Apache sub-project of Hadoop that offers a large-scale mechanism for collecting, storing and monitoring data in HDFS. It is also included in incubator status.

Chukwa's reliability model supports two levels: end-to-end reliability, and fast-path delivery, which minimizes latencies. After writing data into HDFS Chukwa runs a Map-Reduce job to de-multiplex the data into separate streams. Chukwa also offers a tool called Hadoop Infrastructure Care Center (HICC), which is a web interface for visualizing system performance.

6.4 Collecting System Logs using Scribe

Scribe [16], developed by Facebook Inc., is a powerful tool for collecting and distributing system logs from several servers. The collected logs are stored in a centralized scribe server, which is later analyzed by Map-Reduce or Hive. Multiple scribe servers are connected to central scribe server(s) by forming a directed graph, where, each server acts as a node and the edges represent the communication between nodes. The logs from each node are passed over to the next node. Each node maintains local disk storage for storing logs in case the central node is not available because of power or network failure. The locally saved logs are synced to the superior upon availability. Eventually, central nodes collect logs from all nodes and passed to the destination file such as HDFS. When the central node fails, the logs are stored into the local disk and resends upon recovering. However, with such configuration there is always a high chance of losing messages, duplicity and delay in delivering messages.

Apart from Facebook, Scribe is also used by Twitter [29] and Zynga. Not only HDFS, scribe also supports regular file systems and NFS (Network File Systems). Reliability is the key goal where it comes from a file-based mechanism. Unlike Flume or Chukwa, Scribe does not include any convenience mechanisms to collect log data. Rather the load is on the user side, to stream the source data to the Scribe server running on the local systems. For example, to move Apache log files, a daemon has to be written in the tail and forward the log data to the Scribe server, as shown in Figure 4.

6.5 Processing Log Files using Kafka

Kafka [7] is a distributed messaging system. It has been used for collecting and delivering large volumes of log data with low latency [28]. Kafka provides the functionality of messaging system and it also caters better throughput and fault tolerance. Kafka also performs some major tasks which includes monitoring operational data, log aggregation, website activity tracking, and event sourcing. Log aggregation collects log files from web server and places them into central storage area (HDFS). To prevent data loss, messages are persisted on disk and replicated within the cluster.

6.6 Moving Files using HDFS File Slurper

HDFS File Slurper [5] is an open source project, which automatically copies any formats of files from local file systems to HDFS and vice versa. It is helpful in automating the copying of files from local file system to HDFS [6]. File Slurper consists of HDFS file location script for determining the location of file in destination directory. After writing files into destination it verifies the movement of the file.

6.7 Using DistCp to Move Data between Different HDFS Clusters

DistCp [2] [3] is used to copy files and directories between different HDFS clusters. It uses MapReduce jobs for performing copy operation and error handling. It allows copying files from multiple sources in a single command. DistCp also allows updating and overwriting files and their contents into destination directories using "-update" and "-overwrite" commands. DistCp consists of mainly three components: DistCp Driver, Copy-Listing Generator and Input Formats & MapReduce Components. DistCp Driver parses each argument, assembles the arguments into DistCpOptions object, initializes the DistCp and coordinates the copy operation. Copy-Listing Generator creates the list of files to be copied by checking the contents of each file path. Finally, the Input Formats & MapReduce Components perform the actual copy operation, which copy entire files and directories to the destination path.

7 DISCUSSION

The summary of characteristics of different tools discussed above is shown in Table 1. Each of the tool is made for specific purpose, however, most of them share similar characteristics in terms of hardware & software requirement, user-interaction, reliability and fault-tolerance. The similar characteristics are mainly because of they are all playing with Hadoop ecosystem. Data is increasingly being generated and many organizations are striving to acquire real benefits of Hadoop. There is urgent need of sophisticated and user-friendly tools for moving data in and out of Hadoop. However, at this present condition, most of the tools are executed using command line tools. It has been observed that there is serious need of Big Data tools which provide user-friendly interface and are easy to operate. So that, end-users can focus only on data processing and analyzing.

As we have already been pointed out, that, data transfer is a critical task when dealing with cloud-based data processing systems. Customers get enormous success while using Hadoop and its ecosystems. However, challenges in moving large amounts of data to Hadoop will remain as open research questions. Like, can we move terabytes of data or even more using current network setup? Can existing network protocols like FTP and HTTP move large files in a small fraction of time? Since, most of the analysing applications are not available at the location where raw data are generated. Also, for some organisations the data is generated at multiple locations, so, data needs to be moved to a centralised location before processing it. Hence, it is required to move large volumes of data instantly. If needed, smart tools should be developed for handling data transfer tasks. Without which, no one can avail the benefits of Hadoop. These sorts of concerns arise before using Big Data tools, which we must have to learn in detail.

Table 2 shows some recommendations and use cases on the tools as discussed above. These tools help for migrating data from traditional data storage systems to HDFS. Sqoop is helpful for transferring structured data into HDFS. It can transfer bulk data and supports bidirectional data transfer. Flume, Chukwa and Scribe on the other hand, made for collecting different kind of log data and storing into HDFS for further processing. Apache Kafka is used for messaging services as well as log aggregation and events sourcing. HDFS File Slurper is useful for transferring file transfer automatically from local file systems to HDFS and vice versa. DistCp is also similar to Slurper, but it is mainly used when both source and destination are running in distributed environment.

All these tools work mainly in Linux and Mac platforms requiring CPU to be Dual Core or higher and 2.0 GHz clock speed minimum. At least 2GB of RAM size is required but 4 GB is recommended for better performance.

Table 1: Characteristics of different tools for importing and exporting data into HDFS

Characteristics							Tools							
							Sqoop	**Flume**	**Chukwa**	**Scribe**	**Kafka**	**Slurper**	**DistCp**
Fault-tolerant	**Reliability**	**User Interface**	**Software Requirement**	**Hardware Requirement**	**Mode of Data Transfer**	**Goal of Data Transfer**	**Nature of Data**						
Yes	Yes	Command line	GNU/Linux, Mac OS, JVM 1.6.x or later.	Dual core or upgraded, 2.0 GHz clock speed minimum, Min 2 GB RAM (4GB Recommended)	Vice Versa	Data Import and Export	Relational Data						
Yes	Yes	Command line	GNU/Linux, Mac OS, JVM 1.6 or later	Dual core or upgraded, 2.0 GHz clock speed minimum, Min 2 GB RAM (4GB Recommended)	One way (to Hadoop)	Log collection, aggregation and writing to HDFS	Log Data						
Yes	Yes	Command line	GNU/Linux, Mac OS, JVM 1.6 or later	Dual core or upgraded, 2.0 GHz clock speed minimum, 16 GB RAM	One way (to Hadoop)	Monitoring and analysis of log data	Log Data						
No	Yes	Command line	GNU/Linux		One way (to Hadoop)	Log aggregation	Streaming Log Data						
Yes	Yes				One way (to Hadoop)	Messaging, Monitoring, Log aggregation, activity tracking	Log Data						
Yes	Yes	Command line	GNU/Linux, Java, Maven, Git		Vice Versa	File Copy	Files of any format						
Yes	Yes	Command line	CGNU/Linux		Vice Versa	Distributed File Copy	Distributed Files						

Table 2: Recommendation and Use Cases on different tools

Tools	Recommendation/Use Cases
Sqoop	Especially useful to import bulk data transfer between RDBMS and HDFS. Data transfer is only required to analyze and gain some intuitions from the data.
Flume	Analyzing huge amounts of log data assists to identify threats or unique patterns. Flume helps in collecting, aggregating and moving such log data into HDFS. It is useful for sources, which generate log and streaming data, like web servers, sensor devices and social media applications. Flume is highly recommended as Cloudera actively supports it.
Chukwa	Chukwa aids in collecting & aggregating log data; however, Chukwa is batch/mini-batch system in contrast to Flume, which is continuous stream processing system. Chukwa also has Hadoop clusters and MySQL dependencies. Chukwa helps in viewing results of log analysis into its powerful toolkits.
Scribe	Scribe is also applied for collecting and distributing system logs from several servers. However, according to Github, Scribe is no longer supported or updated by Facebook.
Kafka	Useful in case of sending large volume of messages in a client server setup. Kafka is reliable and scalable messaging system. However, data ingestion to HDFS is more challenging than Flume or Scribe.
HDFS File Slurper	HDFS File Slurper is just a utility application for moving files between local file system and HDFS. The source and destinations can be local file system or HDFS or both.
DistCp	It is used to migrate data between two Hadoop clusters or any distributed systems. DistCp is mainly used during data backup in Hadoop.

8 CONCLUSION AND FUTURE WORKS

Hadoop allows processing of huge amount of variety of data in a parallel computing environment. Since, everyday a variety of voluminous data is being generated using applications such as social media sites, web servers, and sensor devices. The nature of such data is mostly unstructured like texts, images and multimedia data. In order to analyze such data, the traditional data processing softwares fail when data size goes beyond the capacity. At this moment, Hadoop is the best platform for handling variety of voluminous data. However, data needs to be imported into HDFS before processing. In this article, we have reviewed different tools for importing and exporting data in HDFS environment. These tools assist in migrating data into HDFS without worrying about data formats and internal operations. Having these tools we can have data loaded into HDFS and get analyzing results quickly.

With the help of these tools or concepts we plan to perform Extract, Transform and Load (ETL) based works in the Semantic Web environment. As the goal of the Semantic Web is to bring traditional data into RDF (Resource Description Framework) format, which allows data to be shared and reused across multiple domains. The research on this field is already in action, however, there are certain data sources that produce huge amount of data. Processing, storing and at the same time converting such data into RDF is becoming challenging for the applications which are not distributed in nature. We plan to use some commonly used tools and approaches [19] [20] to load traditional data into Hadoop for converting, processing and finally storing into RDF format.

REFERENCES

[1] "Apache Sqoop", http://sqoop.apache.org/, accessed 18th December 2015.

[2] "DistCp Guide", https://hadoop.apache.org/docs/r1.2.1/distcp.html, accessed 18th December 2015.

[3] "DistCp Version 2 Guide", hadoop.apache.org/docs/r1.2.1/distcp2.htm, accessed 18th December 2015.

[4] "Facebook API", www.programmableweb.com/api/facebook, accessed 18th December 2015.

[5] "HDFS File Slurper", https://github.com/alexholmes/hdfs-file-slurper, accessed 18th December 2015.

[6] "HDFS Slurper V2", http://grepalex.com/2012/08/20/slurper-v2/, accessed 18th December 2015.

[7] K. M. M. Thein, "Apache Kafka: Next Generation Distributed Messaging System," International Journal of Scientific Engineering and Technology Research, Vol.03, Issue.47, pp. 9478-9483, 2014.

[8] "Twitter API", https://twitter.com/twitterapi, accessed 18th December 2015.

[9] A. Bala and I. Chana, "Fault tolerance-challenges, techniques and implementation in cloud computing," IJCSI International Journal of Computer Science Issues, vol. 9, issue 1, no. 1, pp. 288-293, 2012.

[10] A. Holmes, "Hadoop in practice," *Manning Publications Co.,* 2012.

[11] A. Khetrapal and V. Ganesh, "HBase and Hypertable for large scale distributed storage systems," *Dept. of Computer Science, Purdue University, 2008*.

[12] A. Patel, C. Mehta, and G. Barot, "Hadoop Backup and Recovery Solutions," Packt Publishing, ISBN: 9781783289042, 2015.

[13] A. Rabkin and R. Katz, "Chukwa: A System for Reliable Large-Scale Log Collection," *LISA*, vol. 10, pp. 1-15, 2010.

[14] A. Thusoo, J. S. Sarma, N. Jain, Z. Shao, P. Chakka, S. Anthony, H. Liu, P. Wyckoff, and R. Murthy, "Hive: a warehousing solution over a map-reduce framework," *Proceedings of the VLDB Endowment*, Vol. 2, issue 2, pp. 1626-1629, 2009.

[15] A. Thusoo, J. S. Sarma, N. Jain, Z. Shao, P. Chakka, S. Anthony, H. Liu, and R. Murthy, "Hive-a petabyte scale data warehouse using hadoop," *Data Engineering (ICDE)*, *IEEE 26th International Conference*, pp. 996-1005, 2010.

[16] A. Thusoo, Z. Shao, S. Anthony, D. Borthakur, N. Jain, J.S. Sarma, R. Murthy, and H. Liu, "Data warehousing and analytics infrastructure at facebook," *In Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, ACM, pp. 1013-1020, 2010.

[17] A. F. Gates, O. Natkovich, S. Chopra, P. Kamath, S. M. Narayanamurthy, C. Olston, B. Reed, S. Srinivasan, and U. Srivastava, "Building a high-level dataflow system on top of Map-Reduce: the Pig experience," *Proceedings of the VLDB Endowment*, vol. 2, issue 2, pp. 1414-1425, 2009.

[18] A. K. Somani and N. H. Vaidya, "Understanding fault tolerance and reliability," *Journal Computer*, vol. 30 issue 4, pp. 45-50, 1997.

[19] A. M. F. Husain, P. Doshi, L. Khan, and B. Thuraisingham, "Storage and Retrieval of Large RDF Graph Using Hadoop and MapReduce," In Cloud computing, Springer Berlin Heidelberg, pp. 680-686, 2009.

[20] M. Ali, K. S. Bharat, and C. Ranichandra, "Processing RDF Using Hadoop," Advances in Computing and Information Technology, Springer Berlin Heidelberg, pp. 385-394, 2013.

[21] C. Olston, B. Reed, U. Srivastava, R. Kumar, and A. Tomkins, "Pig Latin: A Not-So-Foreign Language for Data Processing," *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, ACM, pp. 1099-1110, 2008.

[22] C. Olston, G. Chiou, L. Chitnis, F. Liu, Y. Han, M. Larsson, A. Neumann, V. B. N. Rao, V. Sankarasubramanian, S. Seth, C. Tian, and T. ZiCornell, "Nova: Continuous Pig/Hadoop Workflows," *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, ACM, pp. 1081-1090, 2011.

[23] C. L. P. Chen and C. Y. Zhang, "Data-intensive applications, challenges, techniques and technologies: A survey on Big Data," Information Sciences, vol. 275, pp. 314-347, 2014.

[24] E. F. Codd, "Relational Database: A Practical Foundation for Productivity," *Communications of the ACM,* vol. 25, issue 2, pp. 109-117, 1982.

[25] F. M. Facca and P. L. Lanzi, "Mining interesting knowledge from weblogs: a survey," *Data & Knowledge Engineering*, vol. 53, issue 3, pp. 225-241, 2005.

[26] J. Boulon, A. Konwinski, R. Qi, A. Rabkin, E. Yang, and M. Yang, "Chukwa: A large-scale monitoring system," *Proceedings of CCA*, vol. 8, pp. 1-5, 2008.

[27] J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," *Communications of the ACM,* vol. 51, issue 1, pp. 107-113, 2008.

[28] J. Kreps, N. Narkhede, and J. Rao, "Kafka: a Distributed Messaging System for Log Processing," *In Proceedings of 6th International Workshop on Networking Meets Databases (NetDB)*, Athens, Greece, pp. 1-7, 2011.

[29] J. Lin and D. Ryaboy, "Scaling Big Data Mining Infrastructure: The Twitter Experience," *ACM SIGKDD Explorations Newsletter,* vol. 14, issue 2, pp. 6-19, 2013.

[30] J. Pei, J. Han, B. Mortazavi-Asl, and H. Zhu, "Mining Access Patterns Efficiently from Web Logs," *In Knowledge Discovery and Data Mining*, *Current Issues and New Applications*, Springer Berlin Heidelberg, pp. 396-407, 2000.

[31] J. S. Ward and A. Barker, "Undefined By Data: A Survey of Big Data Definitions," *arXiv preprint arXiv*:1309.5821, 2013.

[32] K. Kambatla, G. Kollias, V. Kumar, and A. Grama, "Trends in big data analytics," Journal of Parallel and Distributed Computing, vol. 74, issue 7, pp. 2561-2573, 2014.

[33] K. Ting and J. J. Cecho, "Apache Sqoop Cookbook," *O'Reilly Media, Inc*, 2013.

[34] N. Kuromatsu, M. Okita, and K. Hagihara, "Evolving fault-tolerance in Hadoop with robust auto-recovering JobTracker," Bulletin of Networking, Computing, Systems, and Software, vol. 2, no. 1, pp. 4, 2013.

[35] P. Zhou, J. Lei, and W. Ye, "Large-Scale Data Sets Clustering Based on MapReduce and Hadoop," *Journal of Computational Information Systems*, vol. 7, no. 16, pp. 5956-5963, 2011.

[36] S. Agarwal, D. Borthakur, and I. Stoica, "Snapshots in Hadoop Distributed File System," UC Berkeley Technical Report UCB/EECS, 2011.

[37] S. Hoffman, "Apache Flume: Distributed Log Collection for Hadoop," *Packt Publishing Ltd*, 2015.

[38] S. Kaisler, F. Armour, J. A. Espinosa, and W. Money, "Big Data: Issues and Challenges Moving Forward," System Sciences (HICSS), 2013 46th Hawaii International Conference on, IEEE, pp. 995-1004, 2013.

[39] S. Rathee, "Big Data and Hadoop with components like Flume, Pig, Hive and Jaql," *International Conference on Cloud, Big Data and Trust, pp. 78-82,* 2013.

[40] V. Chang and G. Wills, "A Model to Compare Cloud and non-Cloud Storage of Big Data," Future Generation Computer Systems, 2015.

[41] V. Chang, "Towards a Big Data system disaster recovery in a Private Cloud," Ad Hoc Networks, vol. 35, pp. 65-82, 2015.

[42] V. K. Vavilapalli, A. C. Murthy, C. Douglas, S. Agarwal, M. Konar, et al., "Apache Hadoop YARN: yet another resource negotiator," Proceedings of the 4th annual Symposium on Cloud Computing, ACM, no. 5, 2013.

AUTHOR BIOGRAPHIES

Dr. Ujjal Marjit is the System-in-Charge at the C.I.R.M.(Centre for Information Resource Management), University of Kalyani. He obtained his M.C.A. degree from Jadavpur University, India in 2000. His vast areas of research interest reside in Web Service, Semantic Web, Semantic Web Service, Ontology, Knowledge Management, e-Governance as well as Software Agents etc. More than 40 papers have been published in the several reputed national and international conferences and journals.

Kumar Sharma holds bachelor & master degree in Computer Application. Presently, he is pursuing Ph.D. degree in the Department of Computer Science & Engineering, University of Kalyani, West Bengal, India. His research interests include Semantic Web, Ontology, Web Technologies and Big Data. He also has vast experience in mobile (iOS) application development in the field of Education, Point of Sale, and utility applications.

Puspendu Mandal obtained his bachelor degree (BCA) from Vidyasagar University, West Bengal, India in 2011, and master degree (MCA) from University of Kalyani, West Bengal, India in 2015. His research interests include Web Technologies and Big Data.