Log Analysis Engine with Integration of Hadoop and Spark

Abhiruchi Shinde[1], Neha Vautre[2], Prajakta Yadav[3] , Sapna Kumari[4]

[1]Abhiruchi Shinde, , Dept of Computer Engineering, SITS, Maharashtra, India
[2]Neha Vautre, , Dept of Computer Engineering, SITS, Maharashtra, India
[3]Prajakta Yadav, Dept of Computer Engineering, SITS, Maharashtra, India
[4]Sapna Kumari , Dept of Computer Engineering, SITS, Maharashtra, India
Prof. Geeta Navale Dept of Computer Engineering, SITS, Maharashtra, India

---***---

Abstract - *Log file or logs in computing are the files for keeping record of the events that occur in the operating system or communication between the users or operating systems. Log files contains large amount of valuable information about the system operation status, usage, user behavior analysis etc. Due to extensive use of digital appliances in today's modern era log file analysis has become a necessary task to track system operation or user behavior and acquire important knowledge based on it. These kinds of files are generated at stupendous rate and to analyze them is tedious task and a burden to corporations and various organizations. In order to analyze large dataset, and to store it efficiently, economically and effectively we need to have an effective solution which needs not only the massive and stable data processing ability but also the adaptation to a variety of scenarios under the requirement of efficiency. Such capabilities can't be achieved from standalone analysis tools or even single cloud computing framework. The main objective of the proposed system is to design an application for log analysis and applying the data mining algorithm to get the results which will be useful for system administrator to take proper decisions. The combination of Hadoop, Spark and the data warehouse and analysis tools of Hive and Shark makes it possible to provide a unified platform with batch analysis and in-memory computing capacity in order to process log in a high available, stable and efficient way. Statistics based on customer feedback data from the system will help in greater expansion of business and a company that will have such data to its disposal, and ready to use.*

Key Words: **Log, Weblog, Hadoop, Spark, Log analysis.**

1. INTRODUCTION

There are various types of logs as database logs, binary logs, etc. Based on the type and the requirement of the client, formation of various log analysis tools or platforms can be targeted and brought in to use as per the compatibility of the system. There have been some free powerful log analysis tools like Webalizer , Awstats , and Google Analytics. But they are either standalone or have the limitation of data scale. With the rapid development of Internet technology, the scale of log data is sharply increasing. How to deal with large-scale data becomes a new challenge. However, the emergence of cloud computing with batch process capacity provides a solution to solve this kind of problem. Hadoop is a popular open source distributed computing framework, providing a distributed file system named HDFS and distributed computing framework which is called Map/Reduce. Hive is an analytical tool of data warehouse based on Hadoop, which converts SQL statement to Map/Reduce job to execute. Hadoop and Hive mainly deal with the processing of large data and data storage. The systems designed just based on Hadoop Map/Reduce or even combination of Hive have solved the large scale data processing and storage problems but are not suitable for a class of applications like interactive query and iterative algorithm which is common in analysis system. Spark is designed to speed up data analysis operation. Spark is suitable for the treatment of iterative algorithm (such as machine learning, graph mining algorithm) and interactive data mining. It also provides a data analysis tool Shark, which is compatible with Hive. It provides another choice for large-scale data processing to us. In this paper, we propose a system for log analysis which will overcome the standalone system issues and data scale problems of the previous versions of log analysis tools. In this paper, we will have a generalized view on web server logs, the role of Hadoop and Hive including Spark and Shark in the proposed system, the flow of log analysis strategy for single node as well as multi cluster over a distributed environment

2. WEB SERVER LOGS

Server logs are the logs maintained by the server to keep a detailed and unadulterated record of the various activities performed by the server in a speculated timestamp. It contains various logs of the infinite operations and actions held under the server and the operating system environment. This information needs to be maintained as well as updated from time to time; otherwise it may remain useless for further operations or tasks based on servers.

Whereas, web server logs also known as web log or Weblog, is sort of a website that accommodates a series of various entries often arranged in reverse chronological order, updated on regular basis with variety of new information on a particular topic. Maintaining Weblogs is tedious task unlike what it seems to be. To maintain and track this type of entire log files and to update them on a regular basis is mandatory but difficult. So to make this task a bit less easily we can identify and study particular patterns in the logs which is done only after properly analyzing the log sets we have. Here we introduce log analysis as a mere solution to this problem. By performing log analysis on various data sets we can answer the above mentioned issue. The data generated by a web server or a server or computer is huge, so to filter the required and useful data that makes sense is known as log analysis.

There are various types of logs:-

Event Logs:
Event logs are the logs that keep the record of the various events taking place in the computing environment, so that it provides an audit trail for understanding the activity of the system and to diagnose the various problems it has to face.

Transaction Logs:
Transaction logs maintained by database systems are not necessarily in a human readable format, they are not maintained in intend of keeping an audit trail for further analysis. These records allow or help the different databases to recover from the crashes or any other errors and help to maintain the already stored data in a consistent state.

Message Logs:
Message logs are almost universally plain text files, but IM and VoIP clients (which support textual chat, e.g. Skype) might save them in HTML files or in a custom format to ease reading and encryption.

Database Logs:
Database logs are the logs that are created, maintained and stored for further analysis or audit trail by databases itself. It is a history of the various actions performed the database management system used to guarantee ACID properties in case the system crashes or any of the hardware or software failure.

A Web log is a file to which the Web server writes Information each time a user requests a resource from that particular site. Most logs use the format of the common log format. The following is a fragment from the server logs for loganalyzer.net.

The figures 1 and 2 shows the snapshot of NASA's preprocessed web server logs and it reflects following information about each session.

1. Remote IP address or domain name: An IP address is a 32-bit host address defined by the Internet Protocol; a domain name is used to determine an unique internet address for any host on the internet. One IP address is usually defined for one domain name.

2. Auth user: Username and password if the server requires user authentication

3. Entering and exiting date and time.

4. Modes of request: GET, POST or HEAD method Of CGI (Common Gateway Interface).

5. Status: The HTTP status code returned to the client, e.g., 200 is "ok" and 404 are "not found".

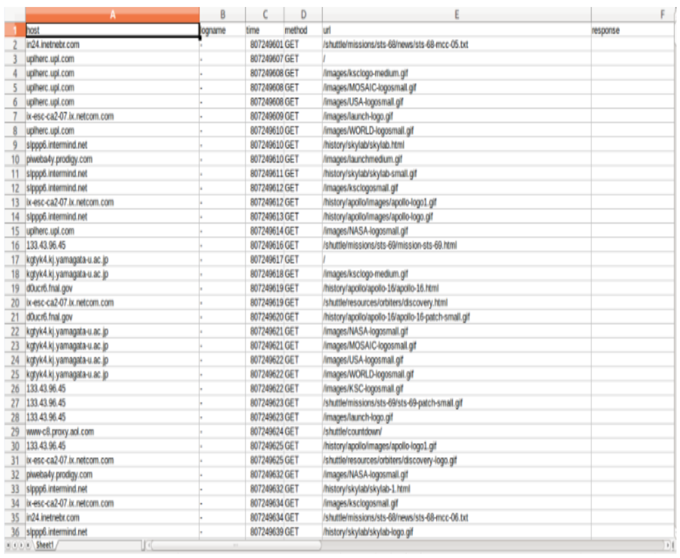
6. Bytes: The content-length of the document transferred.

7. Remote log and agent log.

8. Remote URL.

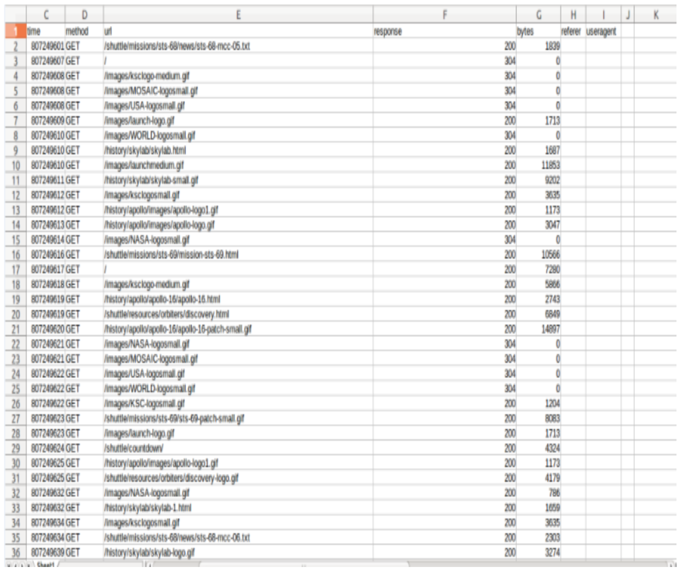
9. "request:" The request line exactly as it came from the client.

10. Requested URL



	host	logname	time	method	url	response
2	in24.inetnebr.com	-	807249601	GET	/shuttle/missions/sts-68/news/sts-68-mcc-05.txt	
3	uplherc.upl.com	-	807249607	GET	/	
4	uplherc.upl.com	-	807249608	GET	/images/ksclogo-medium.gif	
5	uplherc.upl.com	-	807249608	GET	/images/MOSAIC-logosmall.gif	
6	uplherc.upl.com	-	807249608	GET	/images/USA-logosmall.gif	
7	ix-esc-ca2-07.ix.netcom.com	-	807249609	GET	/images/launch-logo.gif	
8	uplherc.upl.com	-	807249610	GET	/images/WORLD-logosmall.gif	
9	slppp6.intermind.net	-	807249610	GET	/history/skylab/skylab.html	
10	piweba4y.prodigy.com	-	807249610	GET	/images/launchmedium.gif	
11	slppp6.intermind.net	-	807249611	GET	/history/skylab/skylab-small.gif	
12	slppp6.intermind.net	-	807249612	GET	/images/ksclogosmall.gif	
13	ix-esc-ca2-07.ix.netcom.com	-	807249612	GET	/history/apollo/images/apollo-logo1.gif	
14	slppp6.intermind.net	-	807249613	GET	/history/apollo/images/apollo-logo.gif	
15	uplherc.upl.com	-	807249614	GET	/images/NASA-logosmall.gif	
16	133.43.96.45	-	807249616	GET	/shuttle/missions/sts-69/mission-sts-69.html	
17	kgtyk4.kj.yamagata-u.ac.jp	-	807249617	GET	/	
18	kgtyk4.kj.yamagata-u.ac.jp	-	807249618	GET	/images/ksclogo-medium.gif	
19	d0ucr6.fnal.gov	-	807249619	GET	/history/apollo/apollo-16/apollo-16.html	
20	ix-esc-ca2-07.ix.netcom.com	-	807249619	GET	/shuttle/resources/orbiters/discovery.html	
21	d0ucr6.fnal.gov	-	807249620	GET	/history/apollo/apollo-16/apollo-16-patch-small.gif	
22	kgtyk4.kj.yamagata-u.ac.jp	-	807249621	GET	/images/NASA-logosmall.gif	
23	kgtyk4.kj.yamagata-u.ac.jp	-	807249621	GET	/images/MOSAIC-logosmall.gif	
24	kgtyk4.kj.yamagata-u.ac.jp	-	807249622	GET	/images/USA-logosmall.gif	
25	kgtyk4.kj.yamagata-u.ac.jp	-	807249622	GET	/images/WORLD-logosmall.gif	
26	133.43.96.45	-	807249622	GET	/images/KSC-logosmall.gif	
27	133.43.96.45	-	807249623	GET	/shuttle/missions/sts-69/sts-69-patch-small.gif	
28	133.43.96.45	-	807249623	GET	/images/launch-logo.gif	
29	www-c8.proxy.aol.com	-	807249624	GET	/shuttle/countdown/	
30	133.43.96.45	-	807249625	GET	/history/apollo/images/apollo-logo1.gif	
31	ix-esc-ca2-07.ix.netcom.com	-	807249625	GET	/shuttle/resources/orbiters/discovery-logo.gif	
32	piweba4y.prodigy.com	-	807249632	GET	/images/NASA-logosmall.gif	
33	slppp6.intermind.net	-	807249632	GET	/history/skylab/skylab-1.html	
34	ix-esc-ca2-07.ix.netcom.com	-	807249634	GET	/images/ksclogosmall.gif	
35	in24.inetnebr.com	-	807249634	GET	/shuttle/missions/sts-68/news/sts-68-mcc-06.txt	
36	slppp6.intermind.net	-	807249639	GET	/history/skylab/skylab-logo.gif	

Fig1: Screenshot of web server logs



	time	method	url	response	bytes	referer	useragent
2	807249601	GET	/shuttle/missions/sts-68/news/sts-68-mcc-05.txt	200	1839		
3	807249607	GET	/	304	0		
4	807249608	GET	/images/ksclogo-medium.gif	304	0		
5	807249608	GET	/images/MOSAIC-logosmall.gif	304	0		
6	807249608	GET	/images/USA-logosmall.gif	304	0		
7	807249609	GET	/images/launch-logo.gif	200	1713		
8	807249610	GET	/images/WORLD-logosmall.gif	304	0		
9	807249610	GET	/history/skylab/skylab.html	200	1687		
10	807249610	GET	/images/launchmedium.gif	200	11853		
11	807249611	GET	/history/skylab/skylab-small.gif	200	9202		
12	807249612	GET	/images/ksclogosmall.gif	200	3635		
13	807249612	GET	/history/apollo/images/apollo-logo1.gif	200	1173		
14	807249613	GET	/history/apollo/images/apollo-logo.gif	200	3047		
15	807249614	GET	/images/NASA-logosmall.gif	304	0		
16	807249616	GET	/shuttle/missions/sts-69/mission-sts-69.html	200	10566		
17	807249617	GET	/	200	7280		
18	807249618	GET	/images/ksclogo-medium.gif	200	5866		
19	807249619	GET	/history/apollo/apollo-16/apollo-16.html	200	2743		
20	807249619	GET	/shuttle/resources/orbiters/discovery.html	200	6849		
21	807249620	GET	/history/apollo/apollo-16/apollo-16-patch-small.gif	200	14897		
22	807249621	GET	/images/NASA-logosmall.gif	304	0		
23	807249621	GET	/images/MOSAIC-logosmall.gif	304	0		
24	807249622	GET	/images/USA-logosmall.gif	304	0		
25	807249622	GET	/images/WORLD-logosmall.gif	304	0		
26	807249622	GET	/images/KSC-logosmall.gif	200	1204		
27	807249623	GET	/shuttle/missions/sts-69/sts-69-patch-small.gif	200	8083		
28	807249623	GET	/images/launch-logo.gif	200	1713		
29	807249624	GET	/shuttle/countdown/	200	4324		
30	807249625	GET	/history/apollo/images/apollo-logo1.gif	200	1173		
31	807249625	GET	/shuttle/resources/orbiters/discovery-logo.gif	200	4179		
32	807249632	GET	/images/NASA-logosmall.gif	200	786		
33	807249632	GET	/history/skylab/skylab-1.html	200	1659		
34	807249634	GET	/images/ksclogosmall.gif	200	3635		
35	807249634	GET	/shuttle/missions/sts-68/news/sts-68-mcc-06.txt	200	2303		
36	807249639	GET	/history/skylab/skylab-logo.gif	200	3274		

Fig.2: Screenshot of web server logs

3. ANALYSIS USING HADOOP AND HIVE

Hive is an analytical tool of data warehouse based on Hadoop. Hive provides a complete SQL query function and converting the statement to the Map/Reduce tasks to execute. The data in Hive is stored in HDFS. Hive offers a way to store, query large scale data stored in various different databases and file systems integrated with hadoop and provides mechanism for analysis. Thus, it can be used for the ETL operations. The extract, transform and load processes include extraction of the required data or data sets from the largely available sets or logs, transforming them into the desired format and then loading them as per the application in which we need to include the log analysis phase. Hadoop and Hive combined together for performance can minimize the limitations put forth by the standalone tools for large scale data processing and storage. We choose Hadoop and Hive as the main cloud framework and warehouse for batch applications by considering the scalability, high availability and fault tolerance nature and facilities provided. Map/Reduce is the batch processing engine for operation like data cleaning and distributed processing of large data sets for computing data clusters on commodity hardware's. Then data is loaded into Hive for further processing with simple SQL queries. The combined results from this step will further be passed on for processing based on particular application and respective logs from various log data sets.

Map reduce is a programming technique for parallel processing of big data. This data can be in any format, but it is particularly designed to process various lists of data. The main concept of map reduce operation is to transform the lists of input data to lists of output data. Because many a times the input data is not in a readable format, there is difficulty in understanding the input datasets. In such cases we need a model or a technique which can transform the input data lists into a readable format of output data lists. Map reduce operation does the fulfillment of this task twice for the two major tasks that are map and reduce by dividing the work into tasks and then further distributing them over different machines in the hadoop cluster. Hence, map reduce is divided into two phases i.e. Map phase and the reduce phase described below in details.

Map Phase

The basic input to this phase is the log file, each record itself is considered as an input to this task. Map function takes a key-value pair as an input thus producing intermediate result in terms of key-value pair. It takes each attribute in the record as a key and Maps each value in a record to its key generating intermediate output as key-value pair. Map reads each log from simple text file, breaks each log into the sequence of keys and emits value for each key which is always 1. If key appears n times among all records then there will be n key-value pairs among its output.

The diagram below shows the process of mapping in the map reduce phase in more detail.

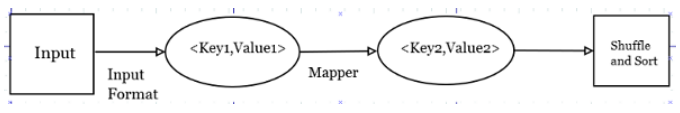


Fig.3: Map Phase

Grouping

After the successful completion of the mapping of the key value pairs, the master controller combines the intermediate results of each map phase for a particular each reduce phase task and then gives the combined file to reduce phase as a sequence of key value pairs.

Reduce Phase

Reduce task takes key and its list of associated values as an input. It combines values for input key by reducing list of values as single value which is the count of occurrences of each key in the log file, thus generating output in the form of key-value pair.

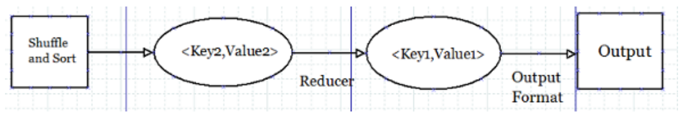


Fig.4: Reduce Phase

The above figure gives the description of the architecture of the reduce phase in the map reduce operation in more detail which will thus take sorted values as their input and after applying reduce operation will give the output in the desired format.

4. ANALYSIS USING SPARK AND SHARK

Spark is an in memory engine for data processing which is comparatively faster than other existing engines. It accelerates the processing of data queries. Spark provides expressive and elegant application processing units to the data workers, which can be used for efficiently executing streaming, machine learning or SQL workloads that need faster and iterative access to various datasets.

In contrast to Hadoop, Spark may or may not have the various advantages of scalability, stability, batch processing ability and the higher rate of availability. Spark has the framework based on memory. Spark also supports a DAG (Directed Acyclic Graph) type schedule somewhat similar to other tools like Dryad, instead of the only map and reduce stage. Spark is a generalised and more flexible framework. It is also suitable for iterative algorithms and interactive queries. Spark uses an event driven architecture which launches the tasks depending on various schedules in merely 5ms, whereas others may take as long as 5-10 seconds. Hence it has task scheduling with low latency.

Shark is a component of Spark, an open source, distributed and fault-tolerant, in-memory analytics system, that can be installed on the same cluster as Hadoop. In particular, Shark is fully compatible with Hive and supports *HiveQL*, Hive data formats, and user-defined functions. In addition Shark can be used to query data in HDFS, HBase, and Amazon S3. Shark does not have its own file system. Hence the data required for this particular purpose is actually stored in HDFS. This keeps the similar management with Hive. Thus Hive and Shark share the same data set for processing of various data sets for respective logs. So basically Shark is an engine primarily based on Spark engine for accelerating the execution of SQL queries on data sets.

The creators of Shark just released a paper where they systematically compare its performance against Hive, Hadoop, and MPP databases. They found Shark to be much faster than Hive on a wide variety of queries: roughly speaking Shark on disk is 5-10X faster, while Shark in-memory is 100X faster. Significantly, Shark's performance gains are comparable to those observed in MPP databases. RDD is the key concept of Spark.

Resilient Distributed Datasets (RDD) is a fundamental data structure of Spark. It is an distributed collection of objects. Each dataset in RDD is divided into logical partitions, which may be computed on different nodes of the cluster. RDD's contain two types of operations also known as transformations i.e. map and filter and actions, which return a value to the application. Here user can decide which data should cache to memory and layout across the nodes. The lost partitions can be rebuilt parallel to rest of the nodes, whenever the node is not gained. Spark computes RDD's, and also can pipeline transform operations. It avoids materializing the intermediate records through pipeline operations to decrease input and output.

5. LOG ANALYSIS FLOW

The basic Log analysis flow in our system starts with the log data sets collection, and then ETL processing is done on log data to make it suitable for processing. Log analysis is done on hadoop as well as spark using the analytical tools hive and shark respectively. Spark does not have it's own file system so it will store it's data in HDFS. Parallel processing will be done on hadoop and for iterative query processing on shark. The result of log analysis can be given as an input to the tool named zeppeline which will show the statistical analysis of result in the form of chart.

Hadoop framework access a large semi structure data in a parallel computation model. Log files usually generated from the web server comprise of large volume of data that cannot be handled by a traditional database or other programming languages for computation. The proposed work aims on pre-processing the log file and keeps track on sessions accessed by the user, using Hadoop and spark. Our work is divided in two phases first is storage and another is processing .storage will be done in HDFS and processing that is analysis with map reduce and shark spark's analytical tool.

Data cleaning is the first phase carried out in the proposed work as a pre-processing step in NASA web server log files. The NASA log file contains a number of records that corresponds to automatic requests originated by web robots, that includes a large amount of erroneous, misleading, and incomplete information. The entries that have status of “error” or “failure” have been removed. The foremost important task carried out in data cleaning is the identification of status code. All the log lines even though satisfy the above constraints, only the lines holding the status code value of “200” is identified as correct log. The corresponding fields containing the correct status code are forwarded to the output text file. After applying data cleaning step, applicable resources are stored in the HDFS as text file, and feed back to the session Identification algorithm as input file. The cleaned web log data is used for further analysis of session identification utilized by the NASA users and also to identify unique user, unique URLs accessed. The session identification plays a major role in web log mining. The processed session file also resides in the HDFS that can be downloaded for further analysis. The pre-processed log file is used to find the user identification, as Map reduce in general identifies the unique values based on key value pair. The log file consists of different fields and the user is identified based on IP address, which is considered as key and their corresponding count as value. Once all the keys are found, the combiner is used to combine all the values of a specific key, the aggregated result is passed to the reduce task and total counts of IP accessed is listed as text file in HDFS. Other than the identification of unique user, unique fields of date, URL referred, and status code is also identified. These unique values is retrieved and used for further analysis in order to find the total URL referred on a particular date or the maximum status code got successes on specific date.

The following diagram describes the log analysis architecture in our proposed system in more detail and makes a more clear view of the log analysis flow.

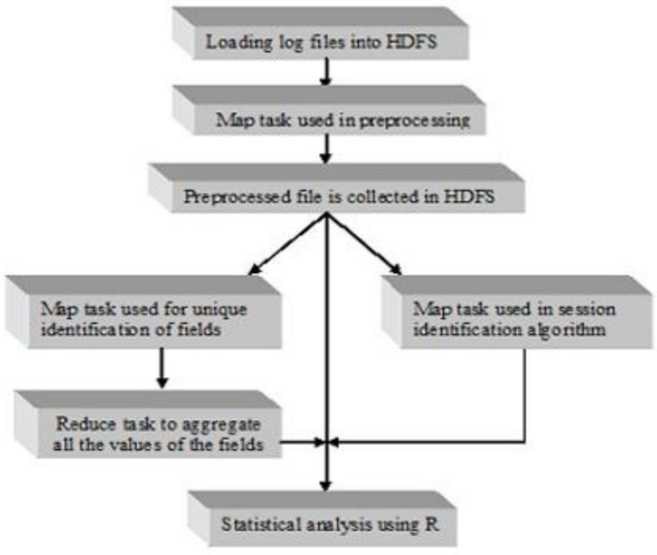

Loading log files into HDFS
Map task used in preprocessing
Preprocessed file is collected in HDFS
Map task used for unique identification of fields
Map task used in session identification algorithm
Reduce task to aggregate all the values of the fields
Statistical analysis using R

Fig.5: Log Analysis Flow

6. CONCLUSION

Log analysis is the process to gather information of the number of access users, user behavior, and system operation status, etc. We design and implement a lightweight distributed framework, consisting of a minimized set of components. The framework is different from the general ones and is specially designed for log analysis of web server logs. This paper analyzes and compares the respective characteristics of Hadoop and Spark framework and Hive/ Shark. Combining the characteristics that are useful to us of Hadoop as well as Spark we propose a cloud platform analysis model with high stability, availability and efficiency for batch data analysis in comparison with standalone log analysis tools and system simply based on Hadoop or the combination of Hive.

REFERENCES

[1] Y. Q. Wei, G. G. Zhou, D. Xu, Y. Chen, "Design of the Web Log Analysis System Based on Hadoop", Advanced Materials Research, Vols. 926-930, pp. 2474-2477, 2014

[2] Xiuqin LIN, Peng WANG, Bin WU “LOG ANALYSIS IN CLOUD COMPUTING ENVIRONMENT WITH HADOOP AND SPARK”, Beijing University of Posts and Telecommunications, Beijing 100876.

[3] Xiaokui Shu, John Smiy 2013.Massive Distributed and Parallel Log Analysis For Organizational Security Industry/University Cooperative Research Center (I/UCRC), and NSF grant CAREER CNS0953638

[4] Yanish Pradhananga, Shridevi Karande, Chandraprakash Karande, "CBA: Cloud-based Bigdata Analytics",978-1-4799-6892-3/15 $31.00 © 2015 IEEE DOI 10.1109/ICCUBEA.2015.18

[5] Markku Hinkka, Teemu Leht, Keijo Heljanko, "Assessing Big Data SQL Frameworks for Analyzing Event Logs",978-1-4673-8776-7/16 $31.00 © 2016 IEEE DOI 10.1109/PDP.2016.26

[6] Jaladi Udaya Santhi, Sivaiah Bellamkonda, Dr.N.Gnaneswara Rao, "Analysis of web server log files using Hadoop MapReduce to preprocess the log files and to explore the session identification and network anomalies", 3rd International Conference on Electrical, Electronics, Engineering Trends, Communication, Optimization and Sciences (EEECOS)-2016, 2016.