HADOOP MAPREDUCE FRAMEWORK FOR GENERIC LOG ANALYSER

Savita D. Naykar[1], Himali J. Vadhan[2], Reshma B. Chaudhari[3], Pratibha A. Shelke[4]

[1,2,3,4] *Computer Department, BVCOE&RI, SPPU,(India)*

ABSTRACT

Recent technological advancements have led to a deluge of data from distinctive domains(e.g., health care and scientific sensors, user-generated data, Internet and financial companies, and supply chain systems) over the past two decades. The term big data was coined to capture the meaning of this emerging trend. In addition to its sheer volume, big data also exhibits other unique characteristics as compared with traditional data. For instance, big data is commonly unstructured and require more real-time analysis. This development calls for new system architectures for data acquisition, transmission, storage, and large-scale data processing mechanisms. In this paper, we present a literature survey and system tutorial for big data analytics platforms, aiming to provide an overall picture for non expert readers and instill a do-it-yourself spirit for advanced audiences to customize their own big-data solutions. First, we present the definition of big data and discuss big data challenges. Next, we present a systematic framework to decompose big data systems into four sequential modules, namely data generation, data acquisition, data storage, and data analytics. These four modules form a big data value chain. Following that, we present a detailed survey of numerous approaches and mechanisms from research and industry communities. In addition, we present the prevalent Hadoop framework for addressing big data challenges. Finally, we outline several evaluation benchmarks and potential research directions for big data systems.

Keywords: Big data analytics, cloud computing, data acquisition, data storage, data analytics, Hadoop.

I. INTRODUCTION

Recent software applications often produce (or can be configured to produce) some auxiliary text files known as log files. Such files are used during various stages of software development, mainly for debugging and profiling purposes. Use of log files helps testing by making debugging simple. It allows you to follow the logic of the program, at high level, lack having to run it in debug mode. Nowadays, log files are commonly used at customer's installations for the cause of permanent software monitoring and/or fine-tuning. Log files became a standard part of large application and are essential in operating systems, networking in computers and distributed systems. Log files are often the only way how to recognize and locate an error in software, because log files analysis is not affected by any time-based issues known as probe effect. This is not similar to an analysis of a running program, when the analytical process can interfere with time-analytical or resourceanalytical conditions within the analyzed program. Log files are often very large and can have difficult structure. Although the process of creating log files is quite simple and straight forward, log files analysis could be a tremendous task that requires huge computational resources long time and experienced procedures. This often leads to a common situation, when log files are continuously generated and employ valuable space on storage devices, but nobody uses them and deploy enclosed information. The inclusive goal of this project is to design a generic log analyzer by using a hadoopmapreduce framework. This generic log analyzer can analyze dissimilar kinds of log files such as- Email logs, Web logs; Firewall logs Server logs, scream data logs.Motivation:The available log analyzers are able to analyze single type of log files only. To make the system capable of analyzing different kinds of log files and to make smart use of hadoop map-reduce framework to increase the efficiency is main motivation to build the system. Built system is able to analyze different kinds of log files with its increased efficiency due to use of hadoop mapreduce framework.Background:The different types of log analyzers like awstats, firegen are able to analyse particular types of log files only. These log analyzers may not be efficient over distributed systems unlike the proposed system which makes use of hadoop mapreduce framework.

Need:There are various applications (known as log file analyzers for log files to produce easily human readable summary reports. Such tools are doubtless useful, but their usage is limited only to log files of certain structure. While such products visualization tools) that can digest a log file of specific vendor or structure and have configuration options, they can answer only infused questions and create infused reports to design an open, very flexible modular tool, that would be capable to analyze almost. There is also a opinion that it is useful to research in the field of log files analysis and any log file and answers any questions, including very difficult ones. Such analyzer should be programmable, extendable, efficient (because of the volume of log files) and easy to use for end users. It should not be limited to analyze just log files of specific structure or type and also the type of query should not be restricted.

II. LITERATURE SURVEY

Accomplishing the data stored in search logs of Web search engines, Intra nets, and Websites can provide important insights into generous the data searching tactics of on-line searchers. This understanding can inform information system layout, interface improvement , and information architecture construction for content collections. This presents a review of and foundation for organizing Web search transaction log analysis[1].

The following item summarizes current possible usage of log files:

- Generic program debugging and describe
- Tests whether program conforms to a given state machine different usage statistics, top tens, etc.
- Security monitoring[2]

Log files have become a definitive part of large applications and are crucial in operating systems, computer networks and distributed systems. According to the available scientific papers it materialize that Antonyms industry is the most evolving and developed area of log files analysis[3]. Log files of HTTP servers are currently used not only for system load statistic but they offer a very expensive and cheap source of feedback. supplier of web content were the first one who lack more detailed and accomplished reports based on server logs. They require to detect behavioural patterns, paths, trends etc. Simple statistical methods do not realize these needs so an advanced approach must be used.[4]

Log files are often very large and can have complicated structure. Although the process of generating log files is quite simple and genuine, log file analysis could be a incredible task. There are over 30 commercially available applications for web log analysis and many so many are free available on the internet. Regardless of their price, they are disliked by their user and considered too low, inflexible and difficult to maintain[5,6]. Some log files, especially small and simple, can be also analyzed using recognized spreadsheet or database programs. In such case, the logs are imported into a worksheet or database and then analyzed by using accessible functions and tools. The stay but probably the most promising way of log file processing represents data driven tools like AWK. In connection with regular appearance are such tools very efficient and flexible[7,8]. On the other hand, they are too low-level, i.e. their usability is restricted to text files, one-way, single-pass operation. Or higher-level tasks they lack mainly advanced data structures. Some Currently present log analysers are as follows:

- Server log analyser eg.AWStats
- Firewall log Analyzer eg. Firegen
- Call Data Analyzer eg. aMiner

Current Log analyzers are able to analyze particular log files only. Eg. Capsa a network examine is able to analyze single type of log file such as http log file.[9]

II. TECHNOLOGY DESCRIPTION AND SYSTEM OVERVIEW

2.1 Technology Description

To build a system for generic log analysis using Hadoop Map Reduce framework by providing user to analyze different types of large-scale log files.

2.2 System Overview

- **Hadoop:**

Hadoop is a software framework that supports data-intensive distributed applications under a free license. It allows applications to work with thousands of computational independent computers and petabytes of data. Hadoop was extracted from Google's Mapreduce and Google File System (GFS) papers. Hadoop is buildup of mainly two parts:

1. HDFS
2. Map-Reduce Framework

HDFS

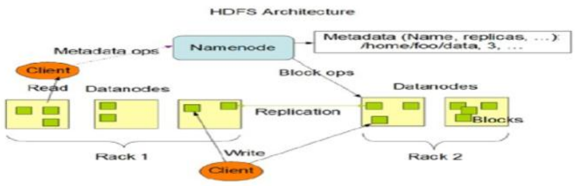


Fig.1 HDFS Architecture

Fig1 shows HDFS architecture. HDFS is the primary distributed storage used by Hadoop applications. A HDFS cluster primarily made up of a NameNode that manages the file system metadata and Datanodes that store the actual data. The architecture of HDFS is described in outfit here. This user guide primarily deals with interaction of users and administrators with HDFS group. The diagram from HDFS architecture depicts basic interactions among NameNode, Datanodes, and the clients. Essentially, clients contact NameNode for file metadata or file modifications and perform actual file I/O directly with the Datanodes. The following are some of the salient features that could be of interest to many users. The terms in italics are explained in later segments. Hadoop, including HDFS, is well suited for distributed storage and distributed processing using product hardware. It is fault tolerant, scalable, and extremely simple to expand. Map-Reduce, well known for its simplicity and materiality for large set of distributed applications, is an integral part of Hadoop. HDFS is largely configurable with a default configuration well suited for many installations. Most of the time, configuration needs to be tuned only for very large clusters. It is written in Java and is supported on all large platforms. Supports shell like commands to interact with HDFS directly. NameNode and Datanodes have fixed web servers that make it easy to check present status of the cluster.

Map Reduce

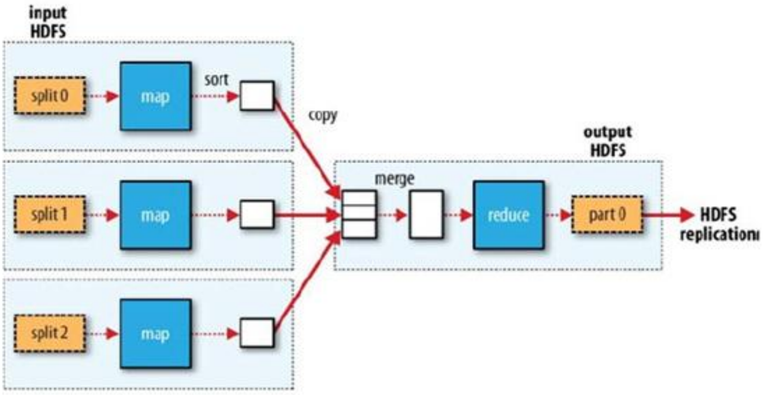


Fig 3.2 Map-reduce data flow

Fig 3.2. Shows MapReduce data flow. Hadoop Map-Reduce is a software framework for easily writing applications which process large amounts of data (multi-terabyte data-sets) in-parallel on large clusters (thousands of nodes) of commodity hardware in a decisive, fault-tolerant manner. A Map-Reduce job usually splits the input data-set into independent chunks which are processed by the map projects in a completely parallel manner. The framework sorts the outputs of the maps, which are then input to the reduce project. Typically both the input and the output of the job are stored in a file-system. The Map-Reduce framework consists of a sole master Job Tracker and one slave Task Tracker per cluster-node.The master is answerable for scheduling the jobs' component work on the slaves, monitoring them and re-executing the failed tasks. The slave finishes the tasks as directed by the master. The log file may have extensions like .txt or .log. Minimally, applications specify the input/output locations and supply map and reduce functions through implementations of suitable interfaces and/or abstract-classes. These, and other job parameters, comprise the job configuration. The Hadoop job client then agree the job (jar/executable etc.) and configuration to the JobTracker which then considers the responsibility of distributing the software/configuration to the slaves, scheduling tasks and monitoring them, contributing status and diagnostic information to the job-client.

III. ALGORITHMS

1) Job Scheduling:

Job Scheduling Early versions of Hadoop had a very simple approach to scheduling users' jobs: they ran in order of submission, adopting a FIFO scheduler. Typically each job would use the whole cluster, so jobs had to wait their turn. Although a shared cluster attempts a great potential for offering large resources to many users, the problem of sharing resources fairly between users desires a better scheduler. Production jobs need to complete in a timely manner, while allowing users who are creating smaller ad hoc queries to get results back in a reasonable time. Later on, the capability to set a job's priority was combined, via the mapred.job.priority property or the setJobPriority() process on JobClient (both of which take one of the characters VERY_HIGH, HIGH, NORMAL, LOW, VERY_LOW). When the job scheduler is selecting the next job to run, it selects one with the chief priority. However, with the FIFO scheduler, priorities do not support preemption, so a large-priority job can still be blocked by a long-running low priority job that started before the large-priority job was anticipated. MapReduce in Hadoop now comes with a choice of schedulers. The default is the real FIFO queue-based scheduler, and there also a multi-user multicipator called the Fair Scheduler.

IV. IMPLEMENTATION TECHNIQUE DESCRIPTIONSINGLE NODE CLUSTER FORMATION

4.1 Single Node Cluster Formation

Prerequisites

Hadoop requires a working Java 1.5+ (aka Java 5) installation. However, using Java 1.6 (aka Java 6) is recommended for running Hadoop. Commands used for java installation:

a) sudoapt getinstall python software properties sudo add-apt-repository ppa :ferramrober to/java

b) sudoapt getu-pdatesudo apt-get install sun-java6-jdk

c) sudoupdate java alternatives java-6-sun

4.2 ADDING A DEDICATED HADOOP SYSTEM USER

We have used a dedicated Hadoop user account for running Hadoop. While that's not required it is recommended because it helps to separate the Hadoop installation from other software applications and user accounts running on the same machine (reasons: security, permissions, backups, etc). sudo add group hadoop sudo add user in group hadoop hduser. This commands add the user hduser and the group hadoop to local machine.

a) Configuring SSH

Hadoop requires SSH access to manage its nodes, i.e. remote machines plus local machine if user wants to use Hadoop on it .For our single-node setup of Hadoop, we therefore need to configure SSH access to localhost for the hduseruser we created in the previous section. First, we have to generate an SSH key for the hduser user.

b) Disabling IPv6

One problem with IPv6 on Ubuntu is that using 0.0.0.0 for the various networking related Hadoop configuration options will result in Hadoop binding to the IPv6 addresses of my Ubuntu box. Theres no practical point in enabling IPv6 on a box when user is not connected to any IPv6 network. Hence, It is recommended to disable IPv6 on Ubuntu machine.

1. Multi Node Cluster Formation

Prerequisites

Configure single-node clusters frstWe need two single-node clusters up and running, we modify the Hadoop configuration to make one machine the master (which will also act as a slave) and the other machine a slave.

2. Stopping The Multi-Node Cluster

Like starting the cluster, stopping it is done in two steps. The workflow however is the opposite of starting. We begin with stopping the MapReduce daemons: the JobTracker is stopped on master, and TaskTracker daemons are stopped on all slaves (here: master and slave). Then we stop the HDFS daemons: the NameNode daemon is stopped on master, and DataNode daemons are stopped on all slaves (here: master and slave). Run the command bin/stop-mapred.sh on the JobTracker machine. This will shut down the MapReduce cluster by stopping the JobTracker daemon running on the machine you ran the previous command on, and TaskTrackers on the machines listed in the conf/slaves file.

EXPECTED INPUT

a)Input to the system is a log file that can be of different kinds.

b)This input log file needs to be given to map-reduce program.

c)The log file may have extensions like .txt or .log.

EXPECTED OUT COMES

First step was to form the hadoop single node cluster. The steps for single node cluster formation are as follows.

1. Installation of Java
2. Add dedicated hadoop system user
3. Configure ssh
4. Disable IPv6
5. Install hadoop tar file and extract it.
6. Update HOME=:bashrc file
7. Update various configuration fles

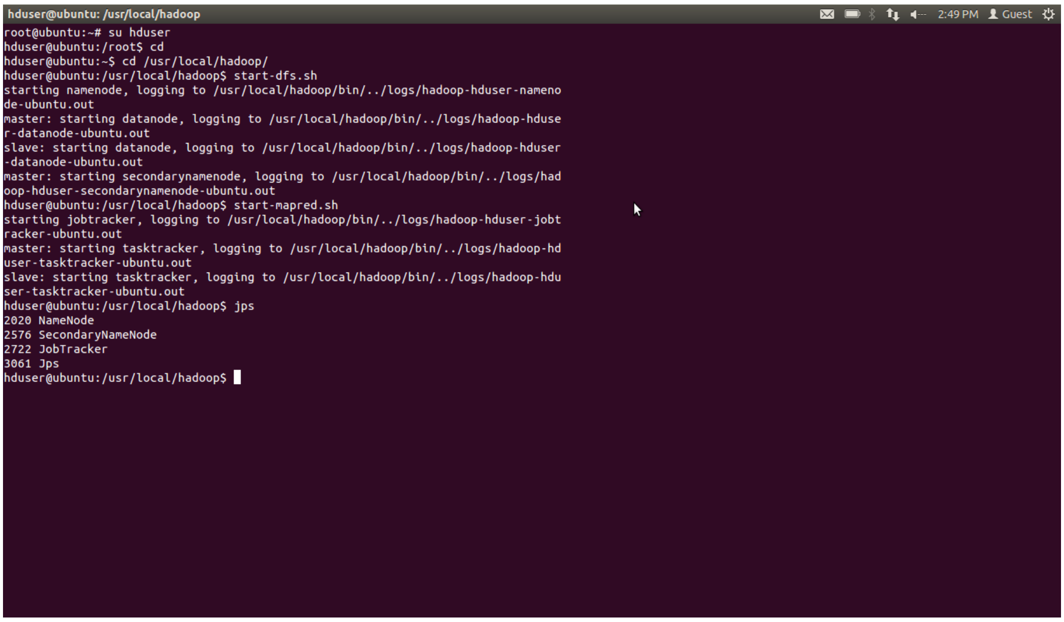
Steps for multinode cluster formation as follow:

1. configure single node
2. Make changes in /etc/hosts(for master and slave)

V.RESULT AND SNAPSHOTS MODULE

Snapshots of modules

Running Single Node Cluster



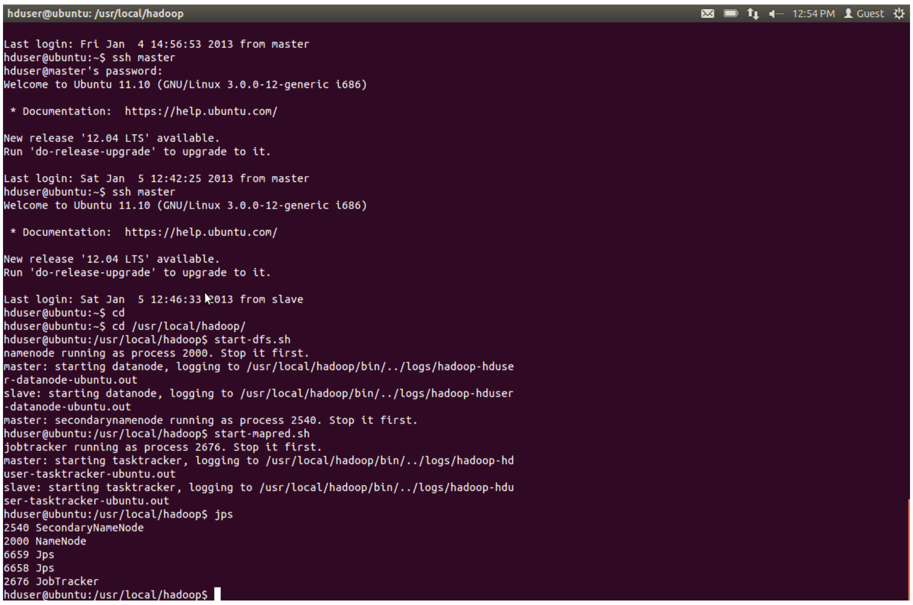
```
root@ubuntu:~# su hduser
hduser@ubuntu:/root$ cd
hduser@ubuntu:~$ cd /usr/local/hadoop/
hduser@ubuntu:/usr/local/hadoop$ start-dfs.sh
starting namenode, logging to /usr/local/hadoop/bin/../logs/hadoop-hduser-nameno
de-ubuntu.out
master: starting datanode, logging to /usr/local/hadoop/bin/../logs/hadoop-hduse
r-datanode-ubuntu.out
slave: starting datanode, logging to /usr/local/hadoop/bin/../logs/hadoop-hduser
-datanode-ubuntu.out
master: starting secondarynamenode, logging to /usr/local/hadoop/bin/../logs/had
oop-hduser-secondarynamenode-ubuntu.out
hduser@ubuntu:/usr/local/hadoop$ start-mapred.sh
starting jobtracker, logging to /usr/local/hadoop/bin/../logs/hadoop-hduser-jobt
racker-ubuntu.out
master: starting tasktracker, logging to /usr/local/hadoop/bin/../logs/hadoop-hd
user-tasktracker-ubuntu.out
slave: starting tasktracker, logging to /usr/local/hadoop/bin/../logs/hadoop-hdu
ser-tasktracker-ubuntu.out
hduser@ubuntu:/usr/local/hadoop$ jps
2020 NameNode
2576 SecondaryNameNode
2722 JobTracker
3061 Jps
hduser@ubuntu:/usr/local/hadoop$ 
```

Hadoop requires a working Java 1.5+ (aka Java 5) installation. However, using Java 1.6 (aka Java 6) is recommended for running Hadoop.

Running Multinode Cluster

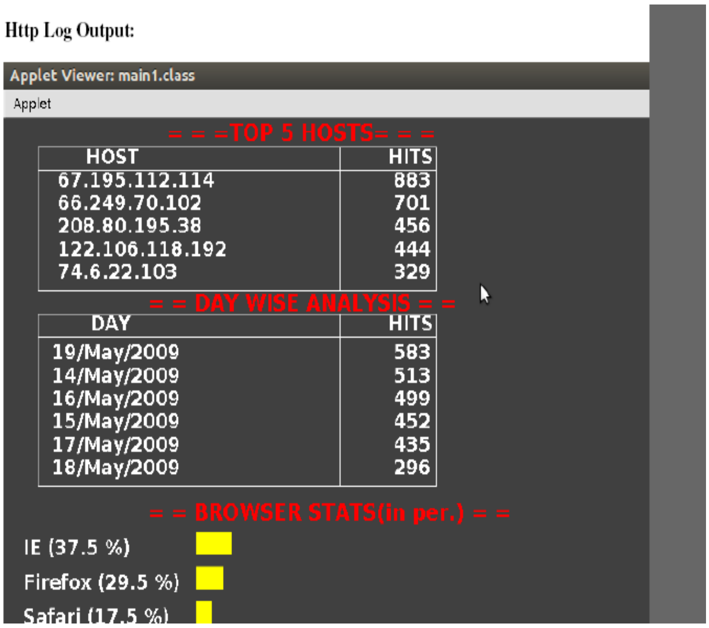
Configure single-node clusters frstWe need two single-node clusters up and running, we modify the Hadoop configuration to make one machine the master (which will also act as a slave) and the other machine a slave.

Running Multinode Cluster



```
Last login: Fri Jan  4 14:56:53 2013 from master
hduser@ubuntu:~$ ssh master
hduser@master's password:
Welcome to Ubuntu 11.10 (GNU/Linux 3.0.0-12-generic i686)

 * Documentation:  https://help.ubuntu.com/

New release '12.04 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Sat Jan  5 12:42:25 2013 from master
hduser@ubuntu:~$ ssh master
Welcome to Ubuntu 11.10 (GNU/Linux 3.0.0-12-generic i686)

 * Documentation:  https://help.ubuntu.com/

New release '12.04 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Sat Jan  5 12:46:33 2013 from slave
hduser@ubuntu:~$ cd
hduser@ubuntu:~$ cd /usr/local/hadoop/
hduser@ubuntu:/usr/local/hadoop$ start-dfs.sh
namenode running as process 2000. Stop it first.
master: starting datanode, logging to /usr/local/hadoop/bin/../logs/hadoop-hduse
r-datanode-ubuntu.out
slave: starting datanode, logging to /usr/local/hadoop/bin/../logs/hadoop-hduser
-datanode-ubuntu.out
master: secondarynamenode running as process 2540. Stop it first.
hduser@ubuntu:/usr/local/hadoop$ start-mapred.sh
jobtracker running as process 2676. Stop it first.
master: starting tasktracker, logging to /usr/local/hadoop/bin/../logs/hadoop-hd
user-tasktracker-ubuntu.out
slave: starting tasktracker, logging to /usr/local/hadoop/bin/../logs/hadoop-hdu
ser-tasktracker-ubuntu.out
hduser@ubuntu:/usr/local/hadoop$ jps
2540 SecondaryNameNode
2000 NameNode
6659 Jps
6658 Jps
2676 JobTracker
hduser@ubuntu:/usr/local/hadoop$ 
```

Output Of HTTP Log



Http Log Output:

Applet Viewer: main1.class

Applet

= = =TOP 5 HOSTS= = =

HOST	HITS
67.195.112.114	883
66.249.70.102	701
208.80.195.38	456
122.106.118.192	444
74.6.22.103	329

= = DAY WISE ANALYSIS = =

DAY	HITS
19/May/2009	583
14/May/2009	513
16/May/2009	499
15/May/2009	452
17/May/2009	435
18/May/2009	296

= = BROWSER STATS(in per.) = =

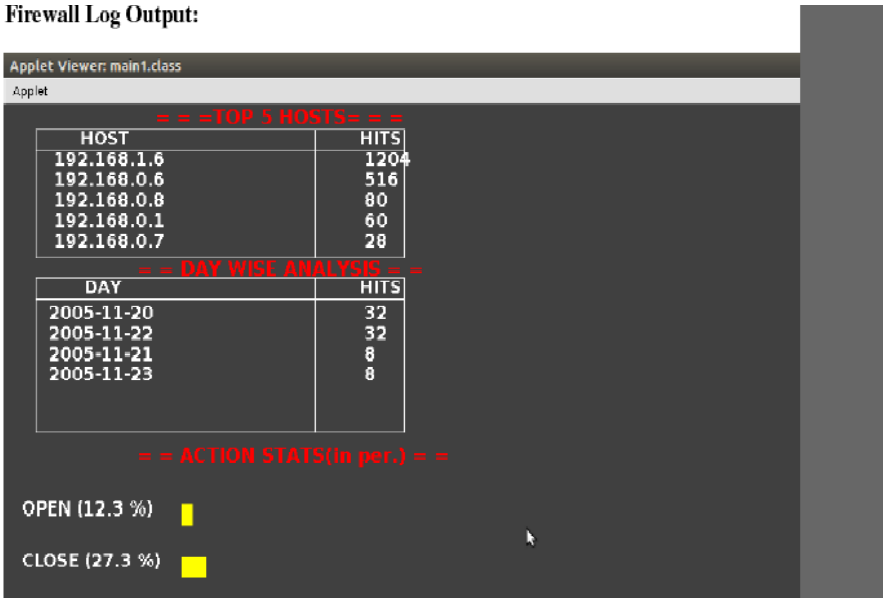
IE (37.5 %)

Firefox (29.5 %)

Safari (17.5 %)

The above figure shows the output of a HTTP log file

Output of a Firewall log



Firewall Log Output:

Applet Viewer: main1.class

Applet

= = =TOP 5 HOSTS= = =

HOST	HITS
192.168.1.6	1204
192.168.0.6	516
192.168.0.8	80
192.168.0.1	60
192.168.0.7	28

= = DAY WISE ANALYSIS = =

DAY	HITS
2005-11-20	32
2005-11-22	32
2005-11-21	8
2005-11-23	8

= = ACTION STATS(in per.) = =

OPEN (12.3 %)

CLOSE (27.3 %)

The above figure shows the output of a Firewall log file

VI. CONCLUSION

At present, for all real domain problems such as web log analysis, fraud detection and text analysis the big data technology is in corporate successfully. Efficiency of log analysis has improved due to the use of Hadoop framework. It will be easy to extend our project to analyze that log file, if any new standard format log file is created. Data generation, data storage, data acquisition and data analysis are the four stages of big data value chain. In the big data generation phase, we have listed various potentially rich big data sources and discussed the data attributes. In the big data storage phase, various cloud based NoSQL stores were introduced, and several key features were compared to assist in big data design judgments. Also, we introduced the backbone of the big data movement, Hadoop, and mapreduce framework [1]. We have examined the design and architecture of Hadoop MapReduce framework in great detail. Particularly, our analysis has focused on data processing. Future work can be extended to number of nodes for prediction analysis to find the next page that will be accessed by the user in a particular website. Also the resulting text files stored in HDFS can be binded with NoSQL databases, and other data mining tools for analysis.

REFERENCES

[1] A Novel Approach For Generic Log Analyzer

[2] Bernard J. Jansen," The Methodology of Search Log Analysis", Pennsylvania State University, USA.

[3] J.H. Andrews," Theory and Practice of Log File Analysis Technical Report", Pennsylvania Western Ontario.

[4] Jan Waldamn," Log File Analysis Technical Report".

[5] W. House. (2012, Mar.). Fact Sheet: Big Data Across the Federal Government [Online]. Available:http://www.whitehouse.gov/sites/default/files/microsites/ostp/big_data%_fact_sheet_3_29_2012.pdf

[6] J. Kelly. (2013). Taming Big Data [Online]. Available:http://wikibon.org/blog/taming-big-data/

[7] Wiki. (2013). Applications and Organizations Using Hadoop [Online]. Available: http://wiki.apache.org/hadoop/PoweredBy

[8] J. H. Howard et al., ''Scale and performance in a distributed file system,'' ACM Trans. Comput. Syst., vol. 6, no. 1, pp. 51–81, 1988.

[9] R. Cattell, ''Scalable SQL and NoSQL data stores,'' SIGMOD Rec., vol. 39,no. 4, pp. 12–27, 2011.