IRACST - International Journal of Computer Science and Information Technology & Security (IJCSITS), ISSN: 2249-9555
Vol. 3, No.4, August 2013

AUTOMATIZED LOG ANALYSER THROUGH WEB BASED HDFS

Prof.G.Shyama Chandra Prasad[1] B.Sudheer Kumar[2] Sree lekha.G[3] Vaishali Chavan[4] Aditya jetta[5] Apurva.k[6]
Information Technology Information Technology I.T Information Technology I.T I.T
ACE Engineering College ACE Engineering College ACEEC ACE Engineering College ACEEC ACEEC
Hyderabad,India Hyderabad,India Hyderabad,India Hyderabad,India Hyderabad,India Hyd,India

Abstract— **Web component based platforms are becoming widespread, both for the constrain devices and cloud computing, for which there is a need for automatic log framework. This paper presents our investigation on an automated log based architecture called as automatized log analyzer through web HDFS. The early adopters like facebook, twitter and yahoo are successfully using Hadoop to tackle their big data analytics challenges. There are few organizations who need scalability power for their big data requirements, this paper provides statistical solution to overcome these current limitations.**

KEYWORDS- ***Hadoop,Mapreduce,HDFS(hadoop distributed file system)***

I. INTRODUCTION

The establishment of internet as brought a tremendous change in the lifestyle and also the way we look at things .Today there are many users using internet and the need for data is also tremendously increasing and the fast speed needed to retrieve the data[1]. so as to make this possible google has designed a google file system which is a system-to system interaction but not user to system interaction and also mapreduce program .Almost everywhere we are going online for example social networking where you directly interact with your friends and families and also update our status ,want to share different things with our friend and that time we need large amount of storage and cannot increase the size of the storage if it fails, then the whole data gets crashed then what is the solution to such storage therefore google came up with an idea that instead of increasing the size of the single harddisk let the data get distributed into different node so that if one node fails it can be replaced with another. The fast processing of the data can also be done by **MapReduce** which is a programming model for processing large data sets with a parallel, distributed algorithm on a cluster. Yahoo, which was equally impressed by the Google File System and MapReduce papers and wanted to build open source technologies based on them and finally came up with Hadoop an open source project.Hadoop was inspired by Google file system and also emerged as an open source software framework that supports the data intensive distributed applications. Hadoop to analyze the unique types of data they're collecting and generating . There are two important functions which helps while failure occurs in parts in Hadoop i.e, map reduce and HDFS. This paper consists of two types of configuration 1) pseudo distributed 2) fully distributed system.

A. Pseudo Distributed System

In Hadoop, Pseudo distributed system is a distributed node on a single host in which the hadoop daemons are distributed in the cores or processes of the single machine.

B. Fully Distributed System

In this distributed system, the hadoop daemons are distributed on different machines. In this system , cluster setup is done where it consists of single Namenode and more number of Datanodes.The basic hadoop daemons are*a) Namenode* comprises the index of the indexes of data node containing data in it. *b) Secondary Namenode* keeps the track and functions of the Namenode.(checkpoints of the namenode) *c) Datanode* contains data of the file. The client communicates with datanode through Namenode and it also manipulates the metadata contained in the name node *.d) Jobtracker* keeps a track on job of the tasktracker. The JobTracker pushes work out to available *TaskTrackere) Tasktracker* is a node which does the map and reduce job that job tracker has provided.

II. BACKGROUND AND MOTIVATION

A. Mapreduce

Map reduce is a programming model that is used in Hadoop for processing large data sets parallel .It mainly comprises of map and reduce[2] . Map performs a filtering and sorting where as reduce performs shuffling and reducing.Map step involves the master node which takes the input and divides them into smaller sub problems and distribute them to the worker nodes. The worker node processes the smaller problems and passes the answer to the master node. Next comes the reduce step in which the master node then collects the answer to all the sub problems and combines them to form the output. The map reduce framework consists list of (key, value) pairs .The map reduce frameworks collects all pairs with the same key from all the lists coming from sub nodes and groups them together creating one group for each key then the reduce function is applied in parallel to each group which produces a collection of values for the same key and then finally the output is provided.Mapreduce also contains two

parts – job tracker and tasktracker. The main task of the mapreduce is to reduce the work that clients have. The unimportant things that a client doesn't need are removed through mapreduce program . In this the jobtracker splits various task and thus processing the data parallel on different nodes so that there is fast computation and thus retrieving the data from a big datasets. The job tracker is responsible in keeping track of mapreduce jobs that are processing in different nodes . If one of the node fails then it can immediately assign the job to the another node so that the data that is provided to the client is in a reliable manner. Task tracker is mainly for mapreduce jobs to be done that are assigned by the job tracker and also gives an heartbeat to the jobtracker so that the jobtracker could know whether to assign another task or not.

B. HDFS (Hadoop distributed file system)

HDFS is derived from google file system and is a distributed, scalable, and portable file system written in Java for the Hadoop framework. Each node in a Hadoop instance typically has a single namenode; a cluster of datanodes form the HDFS cluster[3]. The situation is typical because each node does not require a datanode to be present. Each datanode serves up blocks of data over the network using a block protocol specific to HDFS. The file system uses the TCP/IP layer for communication. Clients use Remote procedure call (RPC) to communicate between each other. HDFS stores large files (an ideal file size is a multiple of 64 MB), across multiple machines.An advantage of using HDFS is data awareness between the job tracker and task tracker.When a big file is provided to HDFS the namenode splits the files into smaller sizes and divides those parts into different datanode and it also keeps the track of the blocks that contains the data .There will be multiple datanodes that will store the data blocks .Hadoop will make sure that the failure of any nodes may not result in the loss of the data.In addition to that there is replication of the data so that if one nodes fails the other node contains the data[4].Earlier the namenode used to be a single node failure to overcome such loss there is a secondary namenode which keeps track of the namnode so that at the time of failure it would come in handy while processing the data until the namenode gets recovered.The datanode stores the datablocks that are been provided to them by the namenode and also report their status to the namenode regarding what blocks they have, so that the namenode can easily provide information to the client about the datanodes it has to consult.Finally, HDFS has different advantages over a non-distributed system those are Highly fault-tolerant, High throughput,Suitable for applications with large data set,Streaming access to file system data,Can be built out of commodity hardware[4]. The best features of Hadoop 1.0.4 version supports the framework which will copy the necessary files on to the slave node before any tasks for the job are executed on that node. Its efficiency stems from the fact that the files are only copied once per job and the ability to cache archives which are un-archived on the slaves.DistributedCache can be used to distribute simple, read-only data/text files and/or more complex types such as archives, jars etc. Archives (zip, tar and tgz/tar.gz files) are un-archived at the slave nodes. Jars may be optionally added to the classpath of the tasks, a rudimentary software distribution mechanism. Files have execution permissions. Optionally users can also direct it to symlink the distributed cache file(s) into the working directory of the task.

III. AUTOMATIC LOG ANALYZER

Unstructured data comes from many sources and takes many forms – web logs, text files, sensor readings, user-generate content like product reviews or text messages, audio, video and still imagery and more. To process such massive amount of data hadoop environment is chosen. Hadoop which is an incredibly fast moving technology and new features are being added everyday to make sure that the organizations have access to all the latest Hadoop tools . Our underlying system is dealing with the logs of a website in which we automatically retrieve the website port and the locations of the users where there is more access to the particular website and also updating is done for every 24 hours.

IV. IMPLEMENTATION

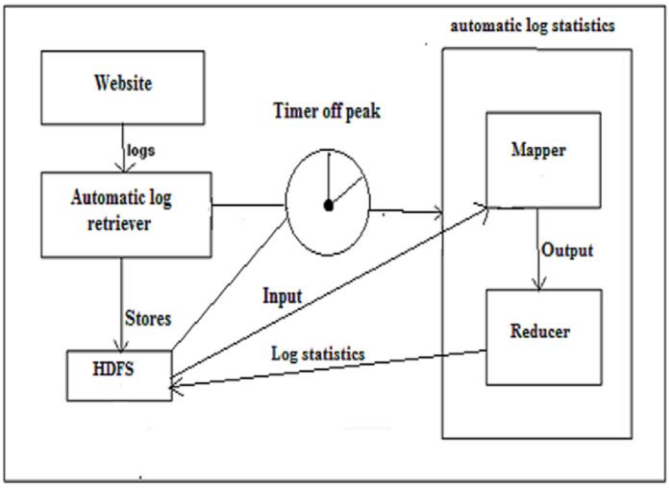
An application adds data to HDFS by creating a new file and writing the data to it. After the file is closed, the bytes written cannot be altered or removed except that new data can be added to the file by reopening the file for append. HDFS implements a single-writer, multiple-reader model[5].The HDFS client that opens a file for writing is granted a lease for the file; no other client can write to the file. The writing client periodically renews the lease by sending a heartbeat to the NameNode. When the file is closed, the lease is revoked. The lease duration is bound by a soft limit and a hard limit. Until the soft limit expires, the writer is certain of exclusive access to the file. If the soft limit expires and the client fails to close the file or renew the lease, another client can preempt the lease. If after the hard limit expires (one hour) and the client has failed to renew the lease, HDFS assumes that the client has quit and will automatically close the file on behalf of the writer, and recover the lease. The writer's lease does not prevent other clients from reading the file; a file may have many concurrent readers.An HDFS file consists of blocks. When there is a need for a new block, the NameNode allocates a block with a unique block ID and determines a list of DataNodes to host replicas of the block. The DataNodes form a pipeline, the order of which minimizes the total network distance from the client to the last DataNode. Bytes are pushed to the pipeline as a sequence of packets. The bytes that an application writes first buffer at the client side. After a packet buffer is filled (typically 64 KB), the data are pushed to the pipeline. The next packet can be pushed to the pipeline before receiving the acknowledgment for the previous packets. The number of outstanding packets is limited by the outstanding packets window size of the client.After data are written to an HDFS file, HDFS does not provide any guarantee that data are visible to a new reader until the file is closed. If a user application needs the visibility guarantee, it can explicitly call

Identify applicable sponsor/s here. *(sponsors)*

the hflush operation[6]. Then the current packet is immediately pushed to the pipeline, and the hflush operation will wait until all DataNodes in the pipeline acknowledge the successful transmission of the packet. All data written before the hflush operation are then certain to be visible to readers. With out affecting the single writer and multiple reader model we have developed the automatic log analyser. Theoretical explanation of our system is automatic retrieval of website name and the location of users . This data which is retrieved from the website is stored and mapreduce processes this data at idle time. In this paper we have implemented the application in both pseudo distributed mode and fully distributed mode.

We focus on our program named under servlets of javascript which has to be inserted in a website, followed by retrieving data which is directly being added to the HDFS in a log file format . Now we go for appending the data in the created file .As we come across the off peak the whole data which is in the input file is processed by map reduce and finally we get the desired output.

V. IMPLEMENTATION



The paper mainly focuses on following functions:

In mapper we provide an input log file in which the data is divided in the form of (key, value) pair and the main job of mapper is to retrieve the data which is desired by the user, the mapper outputs are sorted and then partitioned as per the reducer so we can say that number of partitions is equal to the number of reduce tasks for the job , next comes the combiner in which in which the similar keys are collected and the value is been added.it basically performs aggregation of intermediate outputs,which helps to cut down amount of data transferred from mapper to reducer. The intermediate,sorted outputs are always sorted in a simple format. Now the second main phase of the map reduce framework starts, that is the reducer in which the output from the combiner and is taken as input which collect the website name and location key and stores it in an output file and this output will change its value for every 24 hours based on the required output.The data retrieved is stored in the log file with date and time which gets updated for every 24 hours and get the correct desired display.

VI. CLUSTER SETUP

A. *Single Node:*

This features a program which is to be loaded into a web browser, coming across to a particular user whenever he opens any website his data is getting retrieved and is loaded into the input file automatically and there follows the completion of our basic need of retrieving the data file and we finally get the users statistics of the accessing website. we mainly use Hadoop 1.0.4 as it contains the concept of Append in HDFS that is necessary for our application and we also use Java 1.6 version and Apache TOMCAT 6 version for retrieval of the data from the website.

1.RAM	4GB
2.HARD DISK	500GB
3.PROCESSOR	I3
4.TRANSFER RATE	10MBPS

TABLE 1: CONFIGURATION DETAILS FOR SINGLE NODE

B. *Multiple Node*

One computer is said to be the master and the remaining nodes as slaves. Now the program is loaded into the master. Whenever a slave node accesses the websites the master node gets the complete details of the slave node along with its updates and then when it is off peak the master node will start the map reduce program and retrieves the end products.Here also we mainly use Hadoop 1.0.4 as it contains the concept of Append in HDFS that is necessary for our application and we also use Java 1.6 version and Apache TOMCAT 0.6 version for retrieval of the data from the website.

1.RAM	4GB
2.HARD DISK	500GB
3.PROCESSOR	I3
4.TRANSFER RATE	10MBPS

TABLE2: CONFIGURATION DETAILS FOR NAMENODE AND SECONDARY NODE OF MULTIPLE NODE

1.RAM	4GB
2.HARD DISK	350GB
3.PROCESSOR	dualcore
4.TRANSFER RATE	10MBPS

TABLE3: CONFIGURATION DETAILS FOR DATANODES OF MULTIPLE NODE

VII. EXPERIMENTAL RESULTS

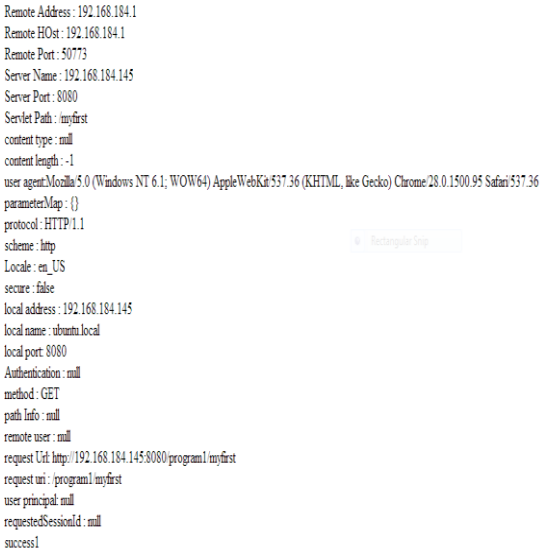


FIG1:LOG DATA RETRIEVED FROM THE WEBSITE

Figure1 shows the output that application retrieves from the website when a client opens the website. Here the data that is been retrieved is remote address, remote port, server port, server name,remote host ,server name, server path, content type,user agent,request url,local port, loacale,secure,path info, requestedSession id,local name,content length,scheme etc..

FIG2:LOG FILE IN HDFS

The fig2 shows the data that is retrieved from the website is stored in hdfs by creating a new file . The file is automatically created without creating it from the console of the system and the data that is stored in the file is of the specific format. Most of the time the data which is retrieved would be stored in the database of the system ,instead of that the data is been considered in the form of text and directly stored in the HDFS so that the following step task i.e.,storing it in database, creating a new file, adding the file to HDFS etc are to shortened and then processing is reduced . Then appending the data is done to the specified file.

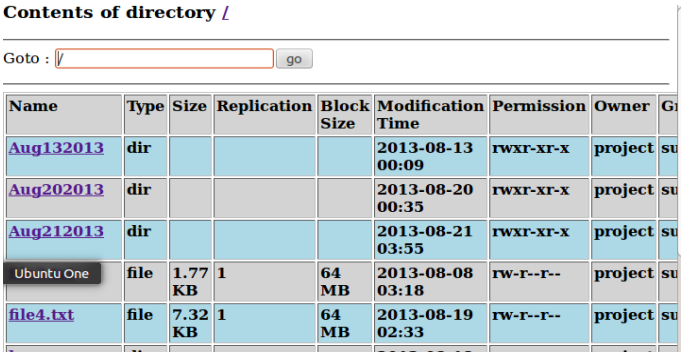


FIG3:OUTPUT FILE CREATED

The above figure show the way the output files are been created . As it is already mentioned that the output file is created for every twenty four hours so that there would be difference of what the status and the data that is been obtained everyday which the website gets its own improvements based on the data the output is providing.

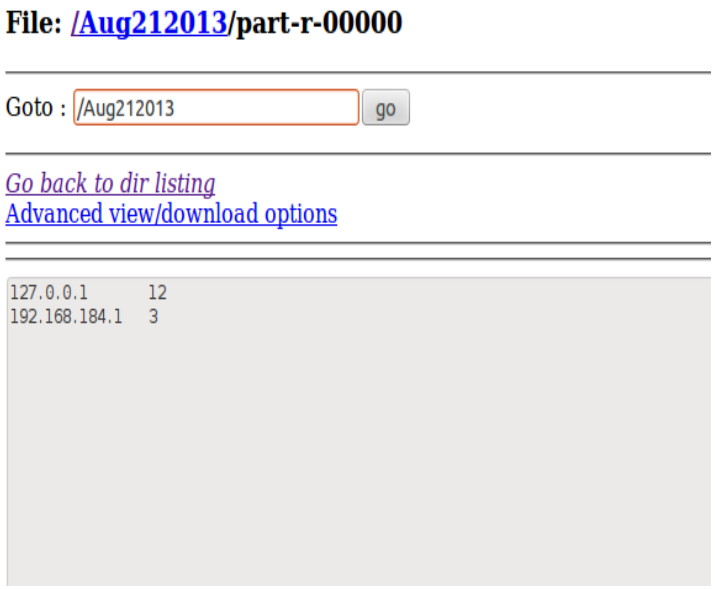


FIG4: THE OUTPUT FILE DATA THAT IS NEEDED

The figure 4 shows the output file which contains the data that we want to obtain.Thus we configure the related statistics obtained for the improvement of the websites.

VIII. CONCLUSION

This paper discusses MULTI-data center databases are fast becoming the new norm for the database operations and how apache Hadoop and our application can comprise a smart and agile data store that is truly location independent.

We believe that the future progress in automatic summarization will be driven both by the development of more sophisticated , linguistically informed model which opens the door in achieving the goals we have develop the

techniques for automatically producing logins. This paper describes experiments that we have carried out to analyze the ability of Hadoop and based on these analysis we have efficiently attend ourproposed system . Hope this would be one of the possible explanation for the wide spread use of these models which are the good techniques so far developed to extract the appropriate statistics of the logins.

REFERENCES

[1] M. Y. Chen, E. Kiciman, E. Fratkin, A. Fox, and E. Brewer. Pinpoint: Problem determination in large,dynamic internet services. In IEEE Conference on Dependable Systems and Networks, June 2002.

[2] YANG, C., YEN, C., TAN, C., AND MADDEN, S. Osprey: Implementing MapReduce-style fault tolerance in a shared-nothing distributed database. In ICDE (2010).

[3] OLSTON, C., REED, B., SILBERSTEIN, A., AND SRIVASTAVA, U. Automatic optimization of parallel dataflow programs. In USENIX Technical Conference (2008).

[4] Yuji Wada, Yuta Watanabe, Keisuke Syoubu, JunSawamoto, Takashi Katoh. Virtual Database Technology for Distributed Database, 2010 IEEE 24th, International Conference on Advanced Information Networking and Applications Workshop.

[5] Wenhao Xu, Jing Li, Yongwei Wu, Xiaomeng Huang, Guangwen Yang, VDM: Virtual Database Management for Distributed and File System, Grid And Cooperative Computing (2008).

[6] D.Borthakur. The Hadoop Distributed File System: Architecture and Design. The Apache Software Foundation, 2007.

AUTHORS PROFILE

G.Shyama Chandra Prasad did his B.Tech in Computer Science and Engineering from Nagarjuna University ,Guntur in 1999, M.Tech from Jawaharlal Nehru Technological University (JNTU), Hyderabad in 2003 and he is pursuing his Ph.D from Jawaharlal Nehru Technological University, Hyderabad (JNTUH) from 2007.He is having 14 years of teaching experience in various engineering colleges and presently working as Professor in ACE engineering college ,Hyderabad. His areas of interest include Computer Networks, Image Processing and Neural networks, Compiler Design, Theory of computation.

Sudheer Kumar Battula Received the Bachelor‟s degree from Aurora‟s Scientific and Technological Institute affiliated to JNTU Hyderabad University in 2009. I‟m currently pursuing my Masters from the Anurag Group of Institutions affiliated to JNTU Hyderabad University in Computer Science & Engineering department. presently working as Asst.Professor in ACE engineering college ,Hyderabad. His areas of interests are Distributed Systems and Cloud Computing.

Sree Lekha.G persuing her graduation in Information Technology from ACE Engineering college Ghatkesar and my research interests on Distributed Computing and Cloud Computing.

Vaishali Chavan persuing graduation her in Information Technology from ACE Engineering college Ghatkesar and my research interests on Distributed Computing and Cloud Computing.

Apurva.K persuing her graduation in Information Technology from ACE Engineering college Ghatkesar and my research interests on Distributed Computing and Cloud Computing.

Aditya Jetta persuing his graduation in Information Technology from ACE Engineering college Ghatkesar and my research interests on Distributed Computing and Cloud Computing.