Jinu Shibu et al. International Journal of Recent Research Aspects ISSN: 2349-7688, Vol. 5, Issue 1, March 2018, pp. 388-390

Weblog analysis and identifying bot traffic using big data

Jinu Shibu, Mr.N Arivazhagan, Vineeth Thomas, Arjun R

Department Of Information Technology, SRM Institute Of Science And Technology

***Abstract*— Spiders are small web programs that harvest information for search engines. These spiders track the websites. In some ways, these are good by quickly showing up the websites. These programs follow certain links on the web and gather information. Like the good spiders, bad spiders also exist and are known as spam spiders. Bad spiders attempt to harvest one's email address. Some spiders may not work efficiently and run in endless loops which are built by dynamically created web pages. So in this project, we try to identify the bad spam spiders present in the webpages and try to eradicate them. And also we minimize the bot traffic. This idea was firstly proposed by Google namely Google Analytics. Proposed methodology used in preprocessing of the huge volume of web log files and finding the statics of website and learning the user behavior.**

I. INTRODUCTION

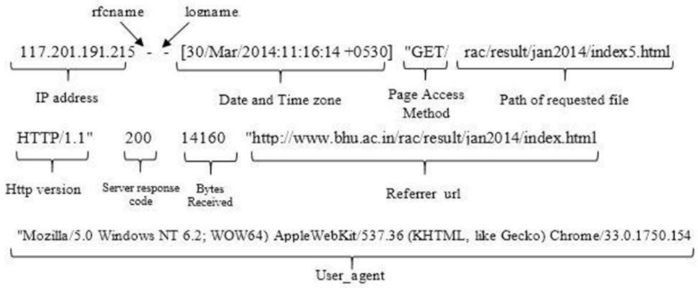
In the world today, everything is set to a fast-paced transformation to a world of digitalization. In a competitive environment that is so fierce and aggressive, there is a rising demand for the best of the services are available. Whether it be regarding the acquirement of a specific product, or whether the customers are satisfied with an application and find it friendly to use or in an online forum how many customers are rapt about a new scheme. These are the typical information that the service providers are keen on acquiring. The service providers need to be aware of the changing trends, to ascertain that a website or a web application is engrossing, details of products which may not be selling as well and in such a case how to refine the existing marketing strategies to attract more customers. For finding an apt solution for all these queries and doubts is where log files come into play. The log file records a list of activities or events that occur when a website or a web application is accessed by a device. They are generated and are stored by the web server. Each event that occurs, be a view of object, document or image, gets registered in the log file. The format of the raw weblog format is a line of text for every hit to a website. Information such as who visited the site, when it was accessed and the browser used for access can be found. Analysis all the details from a weblog helps in creating a pattern which helps in understanding the interaction that occurs between a user and a website.

II. MATERIALS AND METHODS

A. Data

In any particular situation, if the web resources of a particular site are accessed by a user then every information related to the user's activity is stored in the server log file. A server consists of various types of log files and each has a distinct use e.g. error log, referrer log and agent log. A standard format of a weblog would contain the following attributes

(i) IP address
(ii) Date and Time zone
(iii) Page Access Method
(iv) Path of the requested file
(v) Http version
(vi) Server response code
(vii) Bytes Received
(viii) Referrer URL
(ix) User agent


rfcname
logname
117.201.191.215 - - [30/Mar/2014:11:16:14 +0530] "GET/ rac/result/jan2014/index5.html
IP address
Date and Time zone
Page Access Method
Path of requested file
HTTP/1.1" 200 14160 "http://www.bhu.ac.in/rac/result/jan2014/index.html
Http version
Server response code
Bytes Received
Referrer url
"Mozilla/5.0 Windows NT 6.2; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/33.0.1750.154
User_agent

B. Data Preparation

(i) Data extraction: It can be collected from the desired network either using a tool such as Wireshark or can be manually extracted by implementing code for batch extraction using shared folder technique. For accurate readings the data must be collected over a specified interval of time, to ensure that the results obtained are accurate and efficient.

(ii) Individual User Log Identification: To differentiate the data based upon records of individual users, user identification is done. The obtained data need to be filtered and special characters need to be removed before processing the data.
(iii)Data Formatting: The preprocessed data is then changed to a simpler format for the smooth application of the analysis techniques.

C. Bot Discovery

Handling of large sets of server log data requires parallel processing capabilities, hence Hadoop system is used along with hive query language for processing the data. In hive string tokenizer is used for splitting a string into multiple tokens for individually processing each token. The user agent field is mapped for identifying the bot agents. The user agent field would contain the details of the browser used and number of times it accessed a site through a system. Hadoop is used as it has the ability to handle large sets of data, unlike RDBMS.

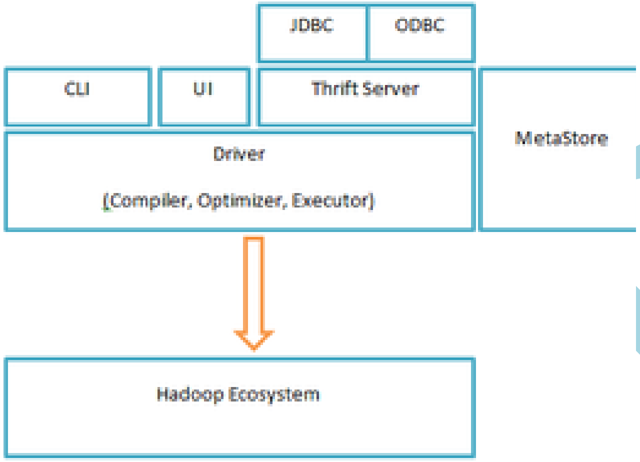
D. Analysis and Graphics

Using the Hive query language, the identified bot are selected and stored in a new table. This table helps in the categorization of the data based on the browser, system used, page accessed etc. The count of the different bots is identified using HQL. Using the JFreeChart the stored data is graphically presented. The trend patterns can be visualized and comparisons of data sets such as a number of bot per system are achieve

III. ARCHITECTURE

A. Hive Architecture

Major components of the Hive architecture are:



Metastore: Stores metadata for each of the tables such as their schema and location. It also includes the partition metadata which helps the driver to track the progress of various data sets distributed over the cluster.[20] The data is stored in a traditional RDBMS format. The metadata helps the driver to keep a track of the data and it is highly crucial. Hence, a backup server regularly replicates the data which can be retrieved in case of data loss.

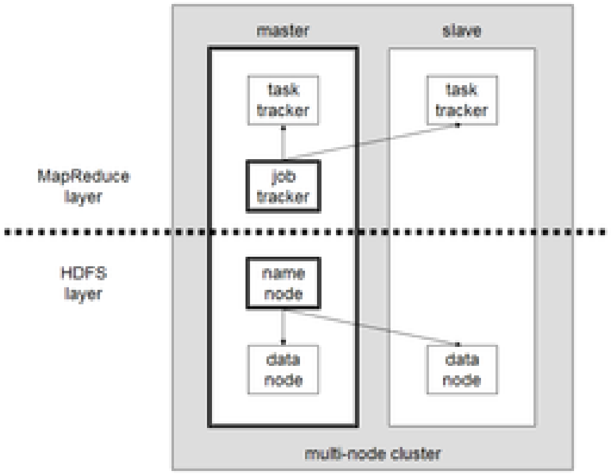
Driver: Acts as a controller which receives the HiveQL statements. It starts the execution statement by creating sessions and monitors the life cycle and progress of the execution. It stores the necessary metadata generated during the execution of an HiveQL statement. The driver also acts as a collection point of data or query result obtained after the Reduce operation.

Compiler: Performs compilation of the HiveQL query, which converts the query to an execution plan. This plan contains the tasks and steps needed to be performed by the Hadoop MapReduce to get the output as translated by the query. The compiler converts the query to an abstract syntax tree (AST). After checking for compatibility and compile-time errors, it converts the AST to a directed acyclic graph (DAG). The DAG divides operators into MapReduce stages and tasks based on the input query and data.

Optimizer: Performs various transformations on the execution plan to get an optimized DAG. Transformations can be aggregated together, such as converting a pipeline of joins to a single join, for better performance. It can also split the tasks, such as applying a transformation on data before a reduce operation, to provide better performance and scalability. However, the logic of transformation used for optimization used can be modified or pipelined using another optimizer.

Executor: After compilation and optimization, the executor executes the tasks. It interacts with the job tracker of Hadoop to schedule tasks to be run. It takes care of pipelining the tasks by making sure that a task with dependency gets executed only if all other prerequisites are run.

CLI, UI, and Thrift Server: A command-line interface (CLI) provides a user interface for an external user to interact with Hive by submitting queries, instructions and monitoring the process status. Thrift server allows external clients to interact with Hive over a network, similar to the JDBC or ODBC protocols.



B. Hadoop Architecture

Hadoop consists of the Hadoop Common package, which provides file system and operating system level abstractions, a MapReduce engine (either MapReduce/MR1 or YARN/MR2) and the Hadoop Distributed File System (HDFS). The Hadoop Common package contains the Java Archive (JAR) files and scripts needed to start Hadoop.

For effective scheduling of work, every Hadoop-compatible file system should provide location awareness – the name of the rack (or, more precisely, of the network switch) where a worker node is. Hadoop applications can use this information to execute code on the node where the data is, and, failing that, on the same rack/switch to reduce backbone traffic. HDFS uses this method when replicating data for data redundancy across multiple racks. This approach reduces the impact of a rack power outage or switches failure; if any of these hardware failures occur, the data will remain available. A small Hadoop cluster includes a single master and multiple worker nodes. The master node consists of a Job Tracker, Task Tracker, NameNode, and DataNode. A slave or worker node acts as both a DataNode and TaskTracker, though it is possible to have data-only and compute-only worker nodes. These are normally used only in nonstandard applications.

Hadoop requires Java Runtime Environment (JRE) 1.6 or higher. The standard start-up and shutdown scripts require that Secure Shell (SSH) be set up between nodes in the cluster.

In a larger cluster, HDFS nodes are managed through a dedicated NameNode server to host the file system index and a secondary NameNode that can generate snapshots of the namenode's memory structures, thereby preventing file-system corruption and loss of data. Similarly, a standalone JobTracker server can manage job scheduling across nodes. When Hadoop MapReduce is used with an alternate file system, the NameNode, secondary NameNode, and DataNode architecture of HDFS are replaced by the file-system-specific equivalents.

IV. PREVIOUS WORKS

1. Andrew Pavlo and Erik Paulson in 2009 compared the SQL DBMS and Hadoop MapReduce and suggested that Hadoop MapReduce loads data faster than RDBMS.
2. Tom White described Hadoop is specially designed to work on a large volume of information by using commodity hardware in parallel.
3. Hadoop-MR log file analysis tool that provides a statistical report on total hits of a web page, user activity, and traffic sources was performed in two machines with three instances of Hadoop by distributing the log files evenly to all nodes.
4. A generic log analyzer framework for different kinds of log files was implemented as a distributed query processing to minimize the response time for the users which can be extendable for some format of logs.

V. PROPOSED SYSTEM.

Hadoop can be used on a single machine, its true power lies in its ability to scale to hundreds or thousands of computers. Hadoop breaks up log files into equal-sized blocks and these blocks are evenly distributed over thousands of nodes in a Hadoop cluster. Also, it does the replication of these blocks over multiple nodes to provide features like reliability and fault tolerance. Parallel computation of MapReduce improves performance for large log files by breaking the job into a number of tasks. The Hadoop implementation shows that MapReduce program structure can be an effective solution for analyzing very large weblog files in Hadoop environment

VI. FUTURE PROSPECTS

The identification of bots in weblog has a very high potential in the field of information technology. Human behavior analysis, navigational pattern and server efficiency are few of the enhanced applications of bot identification using web log analysis. In the above-mentioned applications, bot data needs to be separated to identify the user's true server access history and the non-removal of these may lead to incorrect results. DDoS attack identification and prevention is another elaborate application of the web bot discovery.

VII. RESULT

The following figure shows the analysis of web log file uploaded by the user. This analysis is performed directly on retrieving data from hive database. The following figure represents the number of bots/crawlers identified in different browser and versions and plotted.

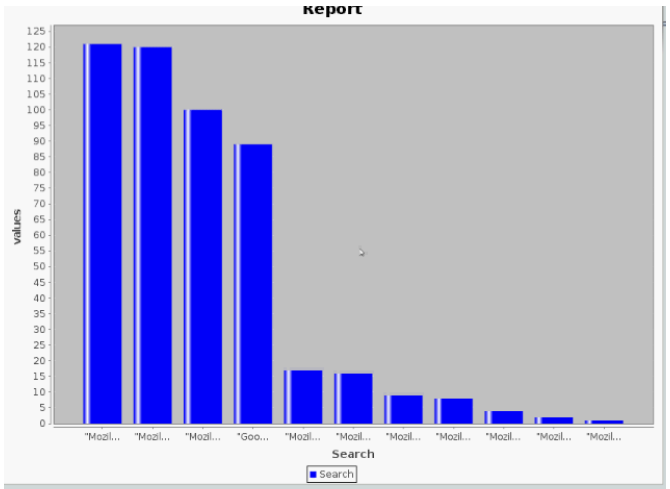


Figure: Result and Analysis of Weblog server for bots/crawlers

VIII. CONCLUSION

Log analysis is the way to gather information of the number of access users, user behavior, and system operation status etc. The increase of log data scale has exceeded standalone tools processing capacity. A stable and efficient data processing platform is indispensable for large-scale log data processing. This paper analyzes and compares the respective characteristics of Hadoop and Spark framework and Hive/Shark. Using Visualization tool for log analysis will provide us graphical reports showing hits for web pages, user's activity, in which part of website users are interested, traffic sources, etc.

IX. REFERENCES

[1] Webalizer. http://www.webalizer.com/.

[2] Awstats. http://sourceforge.net/projects/awstats/

[3] Hadoop. Hadoop Homepage. http://hadoop.apache.org/.

[4] THUSOO A, SARMA J S, JAIN N, et al. Hive-a petabyte-scale data warehouse using hadoop[C]//DataEngineering (ICDE), 2010 IEEE 26th International Conference on. 2010: 996–1005.

[5] THERDPHAPIYANAK J, PIROMSOPA K. Applying Hadoop for log analysis toward distributed IDS[C]//Proceedings of the 7th International Conference on Ubiquitous Information Management and Communication. New York, NY, USA: ACM, 2013: 3:1–3:6.

[6] Spark. Spark Homepage http://spark-project.org/.

[7] Apache Hive. http://hive.apache.org/.

[8] ZAHARIA M, CHOWDHURY M and FRANKLIN M J, et al. Spark: cluster computing with working sets[C]//Proceedings of the 2nd USENIX conference on Hot topics in cloud computing. 2010: 10–10.

[9] ZAHARIA M, CHOWDHURY M, DAS T, et al. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing[C]//Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation. 2012: 2–2.

[10] ENGLE C, LUPHER A, XIN R, et al. Shark: fast data analysis using coarse-grained distributed memory[C]//Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data. 2012: 689–692.