Volume 6, Issue 4, April 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Hadoop MapReduce Auditing Mechanisms

C. Gazala Akhtar*, S. Nuzhath Pasha
Asst. Professor, Department of CSE, Ravindra College of Engineering for Women, Kurnool, Andhra Pradesh, India

Abstract*— *Apache Hadoop framework is an emerging Big Data computing platform for organizing and processing large amounts of rapidly growing data over a cluster of nodes. Apache Hadoop uses Hadoop Distributed File System (HDFS) technology as the underlying distributed file system for Big Data analytics. MapReduce is the underlying programming model which is supported by Hadoop frame work for data analytics. MapReduce model uses inbuilt scheduling algorithms using map and reduce functions. As, the map and reduce functions are executed over several nodes, there could be a change of intruding the map and reduce functions altering the execution procedure. This would generate the unwanted results, leading to security breaches. Hence, it is necessary to audit the log information generated by map reduce functions while execution. To address the problem, here, we explore the several auditing procedures for the logs which were generated by MapReduce functions.

Keywords*— *Hadoop, HDFS, MapReduce, Auditing mechanism

I. INTRODUCTION

Apache Hadoop framework - It is a Big Data computing platform for organizing and processing large amounts of rapidly growing data over a cluster of nodes. Hadoop is used as a multi-tenant service and stores sensitive data such as personal identification data, financial organization data, using Hadoop stores sensitive data on Hadoop clusters.

A. Hadoop base components

Hadoop has 2 base components:

a. Hadoop distributed file system (HDFS)
b. MapReduce.

a. HDFS is an open source file system that can store very large data sets by scaling out across hosts of clusters. It also promises users to have efficient HDFS with Quality of Service.

b. MapReduce is the heart of Hadoop, a underlying programming model which is supported by Hadoop frame work for data analytics.MapReduce works by breaking the process into two phases:

1. **Map phase**: The first is the map job that takes the set of data and converts it into further set of data, where particular elements are broken down into tuples (key/valve pairs).
2. **Reduce phase**: Second the reduce job takes the output from a map as input and combines those data tuples into a smaller set of tuples. As the sequence of MapReduce, the reduce job is always performed after the map job.

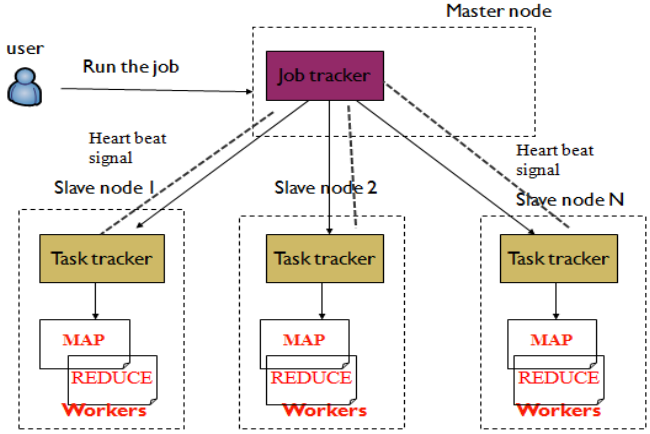


Figure 1: MapReduce Architecture overview

Types of MapReduce daemons

MapReduce mainly contains two daemons:

- ✓ Job tracker(master node)
- ✓ Task tracker(slave node)

Job tracker (MapReduce job)

- ✓ Accept MapReduce jobs
- ✓ Assign tasks to workers
- ✓ Monitor tasks
- ✓ Handle failures
- ✓ Respond to heartbeat message from task trackers

Task tracker (MapReduce task)

- ✓ Run map and reduce tasks
- ✓ Periodically report the progress of tasks via
- ✓ Heartbeat message

Functions in the Model

- **Map**
  - ✓ Process a key/value pair to generate intermediate key/value pairs
  - ✓ Map (key-input, valve-input)→ list (key-map-output, valve-map-output)
  - ✓ Map (input-valve)→list(key, valve)
- **Reduce**
  - ✓ Merge all intermediate values associated with the same key
  - ✓ Reduce(key-map-output, list(valve-map-output))→list(key-output ,valve-output**)**
  - ✓ Reduce(key, list(valve))→list(valve)

II. RELATED WORK

In its current state, Hadoop MapReduce is an extensible framework that allows users to write their own tests/rules for analyzing MapReduce applications. Job Configuration and Job History logs are input for this study, but going ahead we plan to merge the tool with data from Green plum Command Center, a management and tracking principle for both GPDB and GPHD. This will enable Hadoop map reduce to incorporate more sources of information from the cluster such as daemon/user logs, audit logs, job queue data and system standard into its survey. It will also authorize real-time job study when running MapReduce jobs. [4]

Parallel to this work, other researchers did a large scale characterization of MapReduce workloads, including some insights on data access patterns. [4] Their work concentrates on interactive query workloads and did not study the batch type of workload that PROD has. Furthermore, the logs they processed were those of the Hadoop scheduler, and for this reason the authors did not have access to information like age of the files in the system, or when a file is deleted. Perhaps the work most similar to ours (in approach) is that of Cherkasova and Gupta [5], who characterized enterprise media server workloads. An analysis of the influence of new files and file life span was made, but they did not possess file creation and deletion time stamps, so a file is considered to be "new" the first time it is accessed, and its lifetime "ends "the last time it is accessed. No analysis on the burstiness of requests was made. Their results have been cited in this paper where appropriate, to enable us to contrast MapReduce workloads with a more traditional workload. [6][7]

III. PROPOSED SYSTEM

We have proposed an audit logs where a logs shows the entire details of the Hadoop and MapReduce mechanisms carry out, it automatically create logs per each Hadoop daemons. If client perform some operations on files such as editing, modifying, deleting these details are stored in a log4 of history viewer. This entire system of Hadoop frame work specifies with a logs which are an essential part of any operating system, which holds capabilities from audits to bug management. This exercise session explores processing logs with Apache Hadoop from a typical Linux system. The existence of these audit logs provides with an efficient processing of data, best performing of map reduce job.

These map reduce daemons generates audit logs for each of the daemons. Hadoop contains Hadoop logs, HDFS logs, MapReduce job tracker logs, MapReduce task tracker logs. As, the map and reduce functions are executed over several nodes, there could be a possibility of intruding the map and reduce functions, thus altering the execution procedure. This would generate the unwanted results, leading to security breaches. Hence, it is necessary to audit the log information generated by map reduce functions while execution. To address the problem, here, we explore the several auditing procedures for the logs which were generated by MapReduce functions.

A. Job Tracker Audit Logs

The Job Tracker is the service within Hadoop that farms out Map reduce tasks to specific nodes in the cluster, preferably the nodes have the information of data, or at least present in the same rack. Client approaches jobs which submit to the Job tracker, the Job Tracker talks to the name node to determine the location of the data. After locating the data, the Job Tracker locates task tracker nodes with available slots at or near the data. The Job Tracker submits the work to the chosen task tracker nodes. The task tracker nodes are observed, if they do not submit heartbeat signals frequently, they are consider to have failed and the task is scheduled on a different task tracker. A task tracker will notice the result of Job Tracker when a task fails, it may resubmit the job somewhere else, it may mark out that specific record of information as something to neglect, and it may even block the task tracker as untrustworthy. When the work task is finished, the Job Tracker updates its status. The Job Tracker is a point of failure for the Hadoop map reduce service, If it goes down, all running jobs are halted.[8]

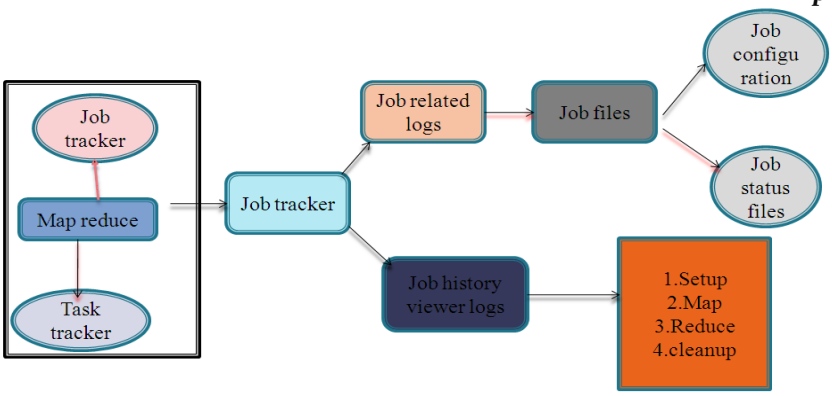


Figure 2 job tracker audit logs

B. Task Tracker Audit logs

The Job Tracker is the master overseeing the overall execution of a MapReduce job and the Task Trackers manage the execution of individual tasks on each slave node. Each Task Tracker is responsible for executing the individual tasks that the Job Tracker assigns. Although there is a single Task Tracker per slave node, each Task Tracker can spawn multiple JVMs to handle many map or reduce tasks in parallel. One responsibility of the Task Tracker is to constantly communicate with the Job Tracker. If the Job Tracker fails to receive a heartbeat from a Task Tracker within a specified amount of time, it will assume the Task Tracker has crashed and will resubmit the corresponding tasks to other nodes in the cluster. [9]

These include the Standard Out and Standard Error logs as well as custom user logs created by code using the logging framework within MapReduce jobs. The location is hardcoded to be ${hadoop.log.dir}/userlogs. It is written in the userlogs subdirectory of the directory defined by the HADOOP_LOG_DIR environment variable.these job tracker and task tracker generates the counters,where all the details of the logs of word count is carry out.

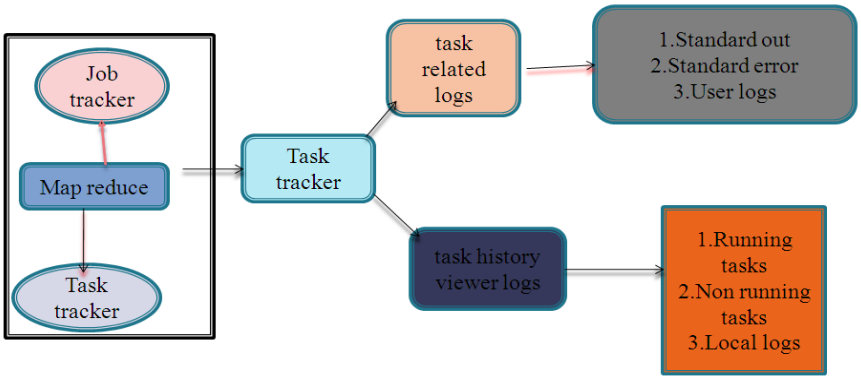
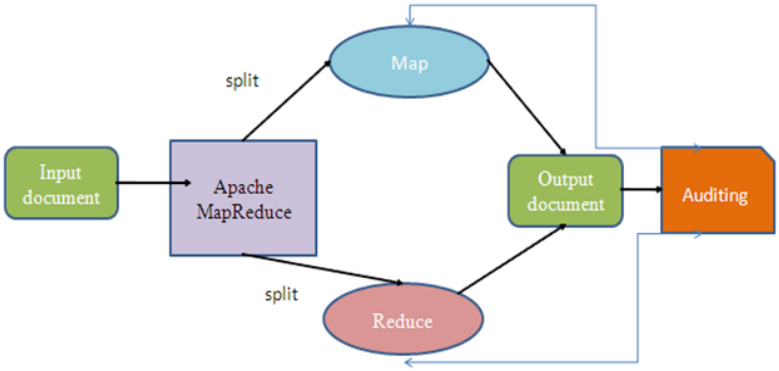


Figure 3 Task tracker audit logs

IV. SYSTEM ARCHITECTURE AND WORKFLOW



This paper gives the work flow of MapReduce and auditing mechanisms which states as

Auditing

Auditing is a process of tracing the significant log events which are occurred during the application/system run time of MapReduce programming module..

MapReduce

Map reduce auditing is a mechanism of auditing the log which are generated while the map and reduce functions are executed.

V. SIMULATION AND RESULTS

We carry out simulation by audit logs which gives us a performance module of map reduce.
The following map reduce audit logs provides with a detail list of auditing mechanisms. . It contains a key component for security mechanisms.

A. Analyse the job logs

We can analyse the working of map reduce job in detail. It mainly shows the time taken by the best performing map task

1. Average time taken by Map tasks: 4sec
2. Worse performing map tasks
3. The last Map task i.e. task_201409031606 _m _000000 finished at (relative to the Job launch time): 3/09 16:08:37 (11sec)

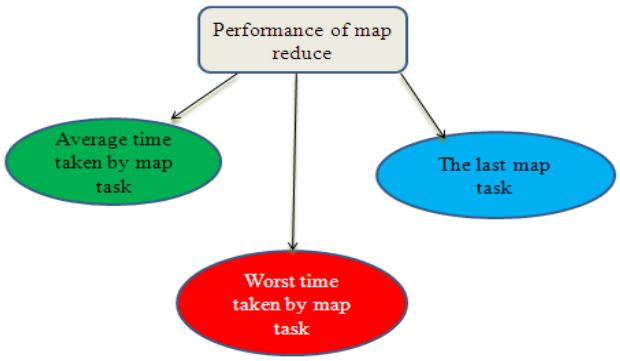


Figure 4 Performance module of map reduce

B. Detailed log list of map reduce word count program

During the execution of map reduce word count program, the following logs are being executed line by line. These logs simulates the mechanisms of the Hadoop.

Table Execution logs of MapReduce

Logs	**Example**
1. Job	Job JOB_ID =”201409021524_0001”
2.Job name	JOBNAME=”word count”
3.user	USER=”hduser1”
4. Submit time	SUBMIT TIME=”140965183261”
5. Job-conf	JOBCONF=”hdfs://localhost:9000/tmp/hadoop-hduser1/mapred/staging/job_201409021524_0001\.xml”
6. View job	VIEW_JOB=”*”
7. Modify job	MODIFY_JOB=”*”
8. Job queue	JOB_QUEUE=”default”
9. Job job_id	job JOB_ID=”job_201409021524_0001”
10. Job priority	JOB_PRIORITY=”NORMAL”
11. launch time	LAUNCH_TIME=”1409651832866”
12. Total maps	TOTAL MAPS=”1”
13. Total reduces	TOTAL_REDUCES=”1”
14. Job status	JOB_STATUS=”PREP”
15. Task id	task TASK_ID=”Task_201409021524_0001_M_000002”
16. Task type	TASK_TYPE=”SETUP”
17. Start time	START_TIME=”1409651833040”
18. Splits	SPLITS=””
19. Map attempt task time task id	Mapattempt TASK_TYPE=”SETUP”
20. Task attempt id	TASK_ATTEMPT_ID=”attempt_201409021524_0001_m_000002_0”
21. Start time	START_TIME=”1409651833582”
22. Tracker name	TRACKER_NAME=”tracker_akhtar:localhost/127\0.\.0\1:55078”
23. Http port	HTTP_PORT=”50060”
24. Task status	TASK_STATUS=”SUCCESS”
25.Finish time	FINISH_TIME=” 1409651837932”
26.Host name	HOSTNAME=”/default-rack/Akhtar”
27.state	STATE-STRING=”SETUP”
28.Counters	COUNTERS=” {FileSystemCounters}{mapreduceframework}{file formats}”

In this paper, we aim to interact the job tracker and task tracker with audit logs in activity and enforce the execution along expected output. Whenever a program get executes the logs are being generated for the particular hadoop daemon.

VI. CONCLUSIONS

Auditing is one of the major requirements to ensure the integrity of the map reduce functions within Hadoop framework. Here, we developed work flow mechanism for Map Reduce programming in Apache Hadoop. These auditing procedure explores the information of the jobs those were executed by the respective map reduce functions for the consistency. The work carries the setup of the Apache Hadoop, Hadoop Map Reduce programming and mechanisms for audit the logs generated by the MapReduce

REFERENCES

[1] "What is map reduce?" is taken from
[2] http://www-01.ibm.com/software/data/infosphere/hadoop/mapreduce/
[3] " Map reduce programming model" is taken from http://en.wikipedia.org/wiki/MapReduce
[4] "Map reduce key features Cloudera" is taken from http://www.cloudera.com/content/cloudera/en/products-and-services/cdh/hdfs-and-mapreduce.html
[5] "Performance advisor for hadoop and map reduce job" is taken from http://blog.pivotal.io/pivotal/products/hadoop-vaidya-performance-advisor-for-hadoop-mapreduce-jobs
[6] "Analysis of enterprise media server workloads: Access patterns, locality, content evolution, and rates ofchange," by L. Cherkasova and M. Gupta IEEE/ACM Trans. Netw., vol. 12, no. 5, 2004
[7] Fan, W. Tantisiriroj, L. Xiao, and G. Gibson, "DiskReduce: RAIDfor data-intensive scalable computing," inProc. PDSW, 2009, pp. 6–10
[8] "A Storage Centric Analysis of Map Reduce Workloads" from https://wiki.engr.illinois.edu/download/attachments/194990492/cabad_IISWC_2012.pdf
[9] "Working of job tracker" http://wiki.apache.org/hadoop/JobTracker
[10] "Working of task tracker" http://www.guruzon.com/6/hadoop-cluster/architecture/what-is-tasktracker-hadoop-cluster-functions-limitations
[11] "hadoop counters –hadoop-the definitive guide" www.inkling.com/read/hadoop-definitive-guide-tom-white-3rd/chapter-8/counters
[12] **"**Audit logs for hadoop daemons**"** http://blog.cloudera.com/blog/2009/09/apache-hadoop-log-files-where-to-find-them-in-cdh-and-what-info-they-contain/

