MANTECH Publications

Journal of Cloud Computing and Internet of Things
Volume 1, Issue 2, July-December, 2023

Navigating Cloud Efficiency: A Comprehensive Guide to MapReduce Monitoring and Implementation

Sameer Joshi[1], Akash Gupta[2]

Assistant Professor[1], Professor[2]

Dept. of Computer Science and Engineering

Nitte Meenakshi Institute of Technology Bangalore, Career Launcher, Bangalore

Corresponding Author's Email: *akgupta31@gmail.com[2]*

Abstract

We live in an Internet-based world. On-demand information or data is required wherever and whenever it is wanted. Big Data is defined as a large volume of data in many formats that cannot be handled using typical methods such as a database management system. Hadoop's MapReduce is one of the computational techniques used in Big Data Analytics. MapReduce as a Service is provided by the cloud. We examine and analyse the problems and requirements of MapReduce integrity in the cloud in this work. The capabilities of some of the most major integrity assurance frameworks are also examined, as well as future research directions. Algorithms for detecting collusive and non-collusive personnel were discussed.

Keywords: *Cloud Computing, Hadoop, MapReduce, Security, Integrity*

INTRODUCTION

Cloud Computing is defined as "a pay-per-use model for enabling convenient, on-demand network access to a shared pool of configurable computing resources for example networks, servers, storage, applications, and services, that can be rapidly provisioned and released with minimal management effort or service provider interaction," according to the National Institute for Standards and Technology. [1] Cloud computing is a method of offering software, storage, and processing as a service to consumers at a fair cost for their applications. Users can connect to apps through the internet using cloud computing. Cloud

services include a pool of resources, flexibility, and extensive network connectivity, as well as accessibility, efficiency, and measurable services. It allows for the storing and processing of data that may vary in terms of volume and pace. All of these characteristics of cloud computing have made it a popular tool for big-data analytics. Cloud-based big data analytics meet the needs of interdisciplinary domains such as health-care, banking, commercial sectors, government initiatives, academia, and many others for difficult statistical analysis, linear regression, and predictive analysis on huge data.

When cloud-based big data analytics is employed, costs such as hardware, software upgrades, maintenance, and network configuration can be reduced. Enterprises may focus solely on data analysis, not on hardware or any other difficulties associated to its upkeep.

HADOOP- HDFS AND MAPREDUCE

Cloud computing and big data are two separate technologies that may work together or independently. In big data analytics, the cloud must play a significant role. Cloud computing is a cost-effective way to store vast amounts of data. In a cloud setting, big data analytics may be delivered as a platform as a service. Due to the rate at which data is proliferating these days, Hadoop may be regarded of as a platform that works on cloud computing to give us with distributed data mining.

This chapter delves into the Hadoop fundamental storage component, the Hadoop Distributed File System, as well as the MapReduce computing paradigm.

Hadoop File system-HDFS

The default file system is Hadoop Distributed File System. Hadoop Framework supports local file systems, HFTP, F2, S3, and other mountable distributed file systems. HDFS is built on top of the Google File System. It's built to function on tens of thousands of low-cost, dependable, and fault-tolerant devices.

In HDFS master/Slave architecture single Name node becomes a master node and N, N>=1 number of data nodes becomes Slave nodes. Metadata and actual data are stored in master and slave node respectively.

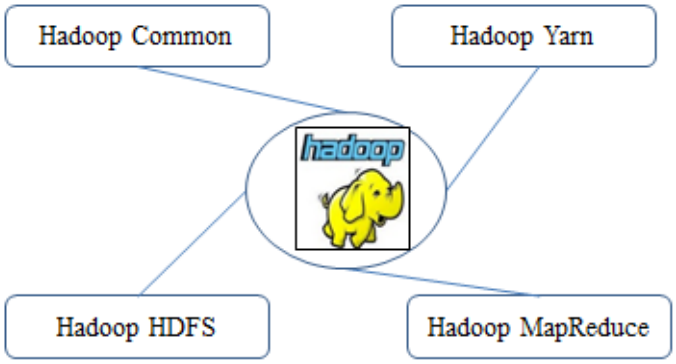

Figure: - 1 Main Modules of Hadoop-Hadoop common, Hadoop Yarn, HDFS and MapReduce constitute Hadoop Framework.

- Determining mapping of fragmented input files to the slave nodes in terms of fixed block size; 64KB by default.

- Instructing data nodes for block creation/deletion/replication. Replication factor is three by default.

- Making data nodes to perform read/write operations. HDFS provides commands to interact with file system which are similar to those of UNIX shell commands.

How Does Hadoop Work?

The first stage is Submitting Job, which allows a user/application to send a job/task to Hadoop's Job-client to be processed. The user specifies the location of the input and output files to do this. Also included in the java files are java classes that contain computational techniques such as Map and Reduce.

Different parameters are also given to the job which does Job configuration. From the JobClient, these jar executable files and input configuration goes to the JobTracker.

Tasks to be handled by JobTracker are as follows:

- Assigning jar files and configuration to slaves/TaskTracker.
- Scheduling tasks
- Monitoring tasks.
- Report about diagnostic/status information to the JobClient.

Last third stage is execution phase where tasks are executed by TaskTracker on different nodes as per the MapReduce code written by the user.

Hadoop MapReduce

MapReduce comprises two components namely Map task and Reduce task (Fig.4). Following the entry of data, the Map Task turns individual items into key/value pairs. This output is fed into the following reduction process as input. It condenses these tuples of data into a smaller set. Both input and output are kept in the HDFS file system by default.

MapReduce is a distributed computing system that may scale to thousands of nodes. In the beginning, a data processing programme must be split into Mapper and Reducer, and the same will be done to data stored on various computers. This is a benefit of the MapReduce technique. The key concept is to deploy a machine to the location where the data is stored. The three key steps of an algorithm are Map, Shuffle, and Reduce. By default, Map or the Mapper reads data from HDFS line by line and processes it by producing tiny blocks of data. The next step is a mix of Shuffle and Reduce. The HDFS receives a new batch of output. During processing, MapReduce is delivered to the cluster's selected servers. Cluster gathers individual results, reduces them, and delivers them to the HDFS server in final result form.

Executing the task

Input to the Mapper has to be converted as pair of <Key,Value>. Output also is in this form only. Map function is applied to all pairs. Zero or more intermediate (1key, 1value) pairs are output of map function. Next stage is grouping of pairs based on 1k value of each pair. Reducer is called for each such group. Output is final result. MasterNode/NameNode is the Node where JobTracker runs and which accepts job requests from clients. JobTracker works with the name node which is single master node. It manages all Hadoop resources. It keeps track of resource consumption and also schedules the master and slave jobs to appropriate machines.

SlaveNode/DataNode is where Map and Reduce programs runs. TaskTracker runs in SlaveNodes. Job of TaskTracker is monitoring tasks assigned to SlaveNodes and re-executing the failed tasks are also additional responsibilities of it.

Word Count Example

One of the well-known illustrations to understand MapReduce is word count program. It is depicted in Fig.5. In the newly setup infrastructure of Hadoop-based big-data ecosystem, many questions need to be answered. These questions are about security of Hadoop ecosystem, security of data residing in Hadoop, secure manner of accessing Hadoop ecosystem, the way to enforce security models and many more.

MAPREDUCE SECURITY THREATS AND PRIVACY CHALLENGES

When the cloud offers MapReduce as a service, it faces additional security and privacy concerns. Some of these issues are discussed here. MapReduce is used to deal with Big Data, which is huge and arrives at a rapid rate from a variety of sources. In MapReduce, replicated pieces of data must be sent and kept securely, whereas a single system holds a single copy of data in one location. For some tasks, the cloud alone may not be able to provide distributed computing. However, MapReduce refers to the process of calculating a distributed replicated piece of data. It must protect dispersed nodes as well as duplicated data. The attack may cause incorrect output from the afflicted mapper or reducer, data modification, data transmission to a third party, and so on. Data flow can take place between clouds, storage nodes, or computation nodes. The addition of security and privacy features should not slow down MapReduce's performance.

When MapReduce is used in the public cloud, it is more exposed to security concerns than when it is used in the private cloud. For MapReduce computational nodes, "authentication, authorisation, and access control" are critical needs. The process of identifying a hostile mapper, reducer, or user is known as authorization. Mappers and reducers' access privileges are tested after successful authentication so that they may proceed to access and framework. The "availability" of data, mappers, and reducers is always and without much latency for authenticated and authorised users. When all duplicated jobs are handled by a collusive group, the strategy provides an excellent method for identifying rogue employees, but it is difficult to recognise. Table I summarises the security risks associated with MapReduce.

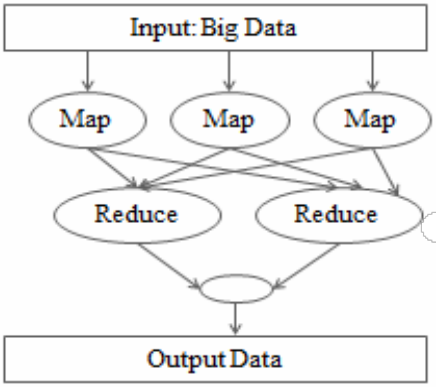

Figure:-2 Hadoop- Map and Reduce algorithm Overview

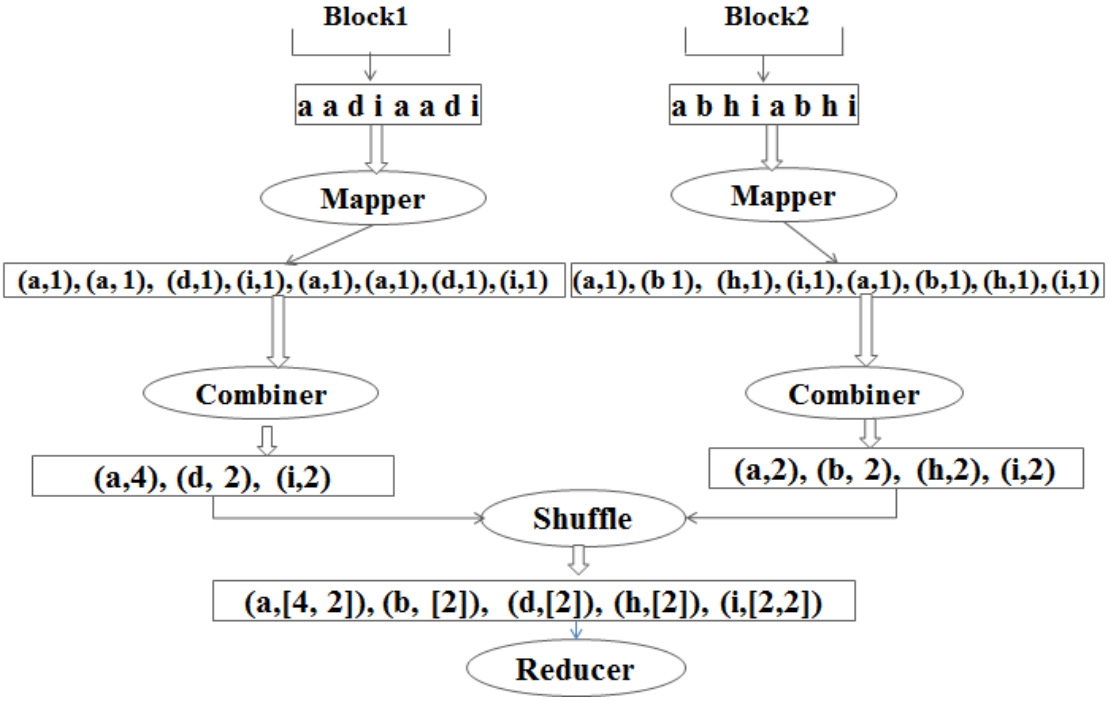

Figure:-3 Work Flow of MapReduce Algorithm in word count task for given file fragmented into block1 and block2.

RELATED WORK ON INTEGRITY ASSURANCE IN MAPREDUCE

”Data Integrity is the assurance that data received are exactly as sent by an authorized entity that means without containing modification, insertion, deletion, or replay”.

In SecureMR framework [5], master is assumed to be safe and workers are not trusted. Distributed File System DFS incorporated with integrity assurance so that workers are provided with integrity protected data. “Each worker is having public/private key pair and any worker can generate and verify signatures and no worker can forge other’s signatures”.

Along with master node, intermediate results obtained from two different map reducers of same replicated task is checked for consistency by other worker also. It provides scalability and efficiency. Commitment protocol and verification protocol are implemented to provide security for MapReduce tasks. Future research direction is applying sampling technique to find inconsistency and provide integrity when all duplicated tasks are processes by collusive attackers. [5] Strategy provides effective solution for detecting malicious workers, but difficult to identify when all replicated tasks are handled by collusive group.

Table:-1 Various Attacks on MapReduce

	Attack	**Passive**	**Active**	**Definition**	**Effect on MapReduce**	**Attack on**
1	Impersonate attack		√	Attacker pretends to be a legal user by gaining passwords or weak encryption schemes with brute force attack	Effected legal user sometimes may be charged for using cloud. Attacker may do data leakage or wrong computations	Authentication
2	Denial-of-Service (DoS)		√	Attacker causes system non-functional. In MapReduce context system means nodes, or mappers or reducers.	Attacked node may cause other working node non-functional by sending repeated execution request. DoS cause heavy network traffic.	Availability of data, mapper, reducer
3	Replay attack		√	Adversary resends the valid message to mappers or reducers	Make node busy. It may replay authentication details and cause impersonate and DoS.	Authorization
4	Eavesdropping	√		Observing inputs, outputs and intermediate results of nodes without knowledge of computing owner	Adversary gain knowledge of intensive computations.	Confidentiality computations and data

5	Man-in-the-middle attack		√	Attacker modifies the data of communication between two nodes	It may lead to DoS, impersonation or replay.	Confidentiality computations and data
6	Repudiation	√		Node falsely denies execution request.	Mapper or reducer falsely denies the execution request of already accomplished task.	Authorization, Authentication

Paper presents "integrity framework for both collusive and non-collusive mappers". This framework assumes both storage and masters are trusted. Mappers and special limited verifier workers are executing on trusted node but mappers are not trusted. It is based on replicating each mapper which identifies non-collusive mappers. It computes result without contacting any other non-collusive nodes. Next is identifying collusive nodes, which communicate with other malicious node in order to send same type of output.

It is very tedious job to identify collusive nodes. Dedicated verification nodes send quiz-type verification to each computing node which verifies portion of result randomly. Whichever node fails to answer quiz is considered as malicious node. Each mapper worker prepares intermediate result as well as MD5 hash code of this result. These are cached for obtaining result of replicated task. Once both workers clear k quizzes. Future research direction is making this framework worth in case of untrusted reducer workers. Second possible enhancement to this work is reusing verification node computation of reused tasks which reduces its workload. Table2 gives detailed comparison of some of the algorithms on assuring integrity in MapReduce.

Table 2 MapReduce Integrity Assurance Frameworks

	Framework	**Verification Schemes**	**Attack Model**	**Concept**	**Future research directions**
1	"VC3: Trustworthy Data Analytics in the Cloud	Hardware / checkpoint based	Physical processors ensure integrity of	User uploads encrypted MapReduce code to work on encrypted file. Key exchange protocol	Tampering Processor packages, DoS attack, traffic analysis, fault

	using SGX" [7]		memory region of systems.	is executed to decrypt MapReduce inside workers. Result is again encrypted.	injections need to be addressed.
2	"TMR: Towards a Trusted MapReduce Infrastructure" [8]	Hardware / checkpoint based	Computing infrastructure is trusted. Master can be verified periodically by third party.	Trusted Platform hardware Module does remote attestation of workers and programs loaded to workers are checked for reliability.	Framework can be verified for large scale infrastructure and real-life practical workloads.
3	"Towards Trusted Services: Result verification schemes for mapreduce" [9]	Watermarking / sampling based	Workers of MapReduce cannot decide whether input is having injected data or not.	First data is preprocessed by verifier with injected watermark. It verifies the correctness of MapReduce result. Random sampling also applied.	Instead only "text-intensive task", "numerical data-intensive" tasks also need to be considered.
4	"Viaf: Verification-based integrity assurance framework for mapreduce" [5]	Watermarking / sampling based	Both storage and masters are trusted	Dedicated verification nodes send quiz-type verification to each computing node which verifies portion of result randomly. Whichever node fails to answer quiz is considered as malicious node.	Making this framework worth in case of untrusted reducer workers and reusing verification node computation of reused tasks which reduces its workload.
5	"A Result Veri1fication Scheme for	Watermarking / sampling	Master is trusted and computation	In prepressing, secondary cluster is formed with equal	Some workers may not being verified at all. Randomized

	MapReduce" [10]	based	providers are malicious without disturbing accuracy level of output.	range of data to every mapper. Small fraction of total number of workers is selected for verification.	worker selection can be improved further for complex computations.
6	"Securemr: A service integrity assurance framework for mapreduce" [6]	Replication/ double check based	Only master is trusted. All workers are in untrusted domain.	Commitment protocol and verification protocol are implemented to provide security for MapReduce tasks. Along with master node, intermediate results obtained from two different map reducers of same replicated task is checked for consistency by other worker also.	Sampling technique to be applied to find inconsistency when all duplicated tasks are processes by collusive attackers.
7	Distributed Results Checking for MapReduce in Volunteer Computing [11]	Replication/ double check based	Master is trusted and workers are not. All workers are non-collusive independent.	Design of the "result certification" in MapReduce computation in Desktop Grid has been addressed using "Majority Voting method". Verification is decentralized involving not only master but workers also. In Naming scheme workers attach a key to the result computed	Collusive workers also may be present in the system and produce erroneous results. Framework can be redesigned t address this issue.

				which helps reducers to check the origin of result.	
8	"Achieving accountable MapReduce in cloud computing"[12]	Replication/ double check based	Cloud data is correct. Workers are unaware of existence of auditors who replay the task. Auditor Group is not malicious actions. Workers cannot reclaim till completion.	It forces each machine to be held responsible for its behavior. Accountability test is done by auditors which check all machines and detect malicious nodes. Auditor replays the task and compares result with original result. Probability-accountability reduces the number of records to be checked.	If more than one auditor is involve in testing A's task, computational task increases. Master need to verify the correctness of idle worker to select as an auditor.

The Authors of [13] HuseyinUlusoy and others developed a novel framework for computation Integrity problem of MapReduce based on replication scheme. In [14] Yan Ding and others proposed a framework for protecting MapReduce against collusive attackers and assuring integrity without extra re-computations. Analysis of "undirected Integrity Attestation Graph" helps to identify both collusive and non-collusive attackers. Assumption is both master and reducers are in trusted domain whereas mappers are untrusty. To verify workers' trust in terms of consistency of mappers' results, edges of graphs have marked either zero or one. Zero indicates both mappers have given different intermediate results but otherwise no. Earlier case is termed as inconsistency pair of workers and later one as consistency pair.

In [14] Y Ding and others proposed a scheme to detect attacks in both map phase and reduce phase and is based on not replication based. First assumption of attack model-nodes are communicating in cryptographically secured way and only master node is assumed to be trusted. It is probe injection based verification method to achieve integrity of MapReduce computations and also to identify malicious workers. “A probe consists of data injected into the original input dataset. The result of the probe set can be pre-computed before the entire input dataset is processed in the MapReduce system. A probe has two attributes: data value and location. The data value is the specific value in the computation; the location is its position in the entire dataset after injection.”

CONCLUSION

MapReduce became a fault-tolerant, efficient, and scalable data processing tool for large datasets. But when MapReduce introduced over public and hybrid cloud computing, addressing security and privacy became important concerns. Since then many algorithms and frameworks are coming into picture to address these sensitive issues. Some of the important algorithms and frameworks had been surveyed here in details. Comparison table is also presented. Based on survey we listed these frameworks under three categories namely hardware/check based, sapling bases and finally replication based. During this survey we observed that lot of research need to be done in the area of security issues in file system, trust on hardware etc. No much work is shown where master is malicious or in case of untrusted cloud provider.

Two or more of these algorithms can be integrated to provide more sophisticated secured BigData-MapReduce Analytics model in Cloud.

REFERENCES

1. Ajith Bailakare, and Meenakshi, “An Introduction to Cloud Computing and its Security Issues and Challenges - A Literature Review”, IJEECS, Vol. 6, Issue 5, 2017.
2. Han hu, Yonggangwen,Ttat-sengchua, and Xuelong li, “Toward scalable systems for big data analytics: A Technology Tutorial”, IEEE transactions, vol. 2, Pp. 652-687, 2014.

3. RaghavendraKune, Pramod Kumar Konugurthi, Arun Agarwal Raghavendra Rao Chillarige and Rajkumar Buyya, “The anatomy of big data computing”, Wiley Online Library, 2015.

4. Shankar Ganesh Manikandan and Siddarth Ravi, “Big Data Analysis Using Apache Hadoop”, IEEEE Explore, International Conference on IT Convergence and Security, Pp. 1-4, Oct. 2014.

5. W. Wei, J. Du, T. Yu, and X. Gu, “SecureMR: A service integrity assurance framework for mapreduce,” in ACSAC, 2009, pp. 73–82.

6. Y. Wang and J. Wei, “VIAF: verification-based integrity assurance framework for MapReduce”, International Conference on Cloud Computing, Washington, DC, USA, Pp. 300–307, 2011.

7. F. Schuster, M. Costa, C. Fournet, C. Gkantsidis, M. Peinado, G. Mainar-Ruiz, and M. Russinovich, “Vc 3: Trustworthy data analytics in the cloud”, IEEE Symposium on Security and Privacy, Vol. 15, 2014.

8. A. Ruan and A. Martin, “TMR: Towards a trusted mapreduce infrastructure,” World Congress, Pp. 141–148, 2012.

9. Huang Chu, Zhu Sencun and Wu.Dinghao, “Towards Trusted Services: Result Verification Schemes for MapReduce”, International Symposium on Cluster, Cloud and Grid Computing 2012.

10. Pareek G., Goyal C. and Nayal M., “A Result Verification Scheme for MapReduce Having Untrusted Participants”, Intelligent Distributed Computing, Advances in Intelligent Systems and Computing, Vol 321, 2015.

11. M. Moca, G. C. Silaghi and G. Fedak, “Distributed Results Checking for MapReduce in Volunteer Computing”, International Symposium on Parallel and Distributed Processing Workshops and Phd Forum, Shanghai, Pp. 1847-1854, 2011.

12. Zhifeng Xiao, Yang Xiao, “Achieving Accountable MapReduce in cloud computing”, Future Generation Computer Systems, Pp. 1-13, 2014.

13. HuseyinUlusoy, Murat Kantarcioglu, ErmanPattuk and LalanaKagal, “AccountableMR: Toward Accountable MapReduce systems”, International Conference on Big Data, Big Data 2015, Santa Clara, CA, USA, Pp. 451–460, 2015.

14. Yan Ding, Huaimin Wang, Songzheng Chen, Xiaodong Tang, Hongyi Fu and Peichang Shi, “PIIM: Method of Identifying Malicious Workers in the MapReduce System with an Open Environment”, International Symposium on Service Oriented System Engineering, 2014.

15. Meenakshi, A. C. Ramachandra, M. N. Thippeswamy, and AjithBailakare, "Role of Hadoop in Big Data Handling", ICICI 2018, LNDECT 26, Pp. 482–491, 2019.

16. R. Chandana, D. Harshitha, Meenakshi, and A. C. Ramachandra, "Big Data Migration and Sentiment Analysis of Real Time Events Using Hadoop Ecosystem", ICICI 2018, LNDECT 26, Pp. 1–7, 2019.