ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 5, Issue 12, December 2016**

# A Secure Extended Approach for Load Balancing for Distributed File System in Cloud

K.Supriya[1], Neelam Poornima[2] , P.Devaki [3]

M. Tech Student, Dept. of CSE, Sri Venkatesa Perumal College of Engineering and Technology, Puttur , India [1, 2] .

Associate Professor, Dept. of CSE, Sri Venkatesa Perumal College of Engineering and Technology, Puttur, India [3].

**ABSTRACT:** A file system (FS) is employed wherever to store and retrieving process is essential along with Naming, sharing and protection of files. Distributed file system (DFS) has transparency mechanisms such as location transparency, failure transparency, replication transparency, etc. Some examples of DFS are NFS, RFS and AFS. DFS is used for cloud computing based on MapReduce programming model, which comprise of Map() for filtering and Reduce() for summary. In the dynamically varying cloud computing environment, nodes may be upgraded, replaced and appended there by arises the chances of load imbalance problem. To counter this, modified load rebalancing algorithm is enforced to reduce burden off the server. The aid of open source Kerberos network authentication protocol is taken to handle multiple nodes in the environment. While retrieving the file from multiple servers, if any one of the servers that have the chunk part of the requested file was failed, the HDFS communicates with MongoDB and gives the requested file, from its own shallow copy, which is nothing but the duplicate copy of the original file. This paper gains effective load rebalancing against previous algorithms and also availing the security by appending Kerberos authentication to the user.

**KEYWORDS:** Cloud computing, Hadoop Distributed file system, Map Reduce, Load Rebalance, Algorithms.

## I. INTRODUCTION

Cloud computing is the delivery of computing services over the web.Cloud services enhance people and business to use software and hardware system elasticity, and area unit managed by third parties at remote locations. The characteristics off cloud computing embrace on-demand self servicebroad network access, resource pooling, fast snap and measured service. As apache Hadoop is used which is a open source frame work that support data intensive distributed application.HDFS is highly fault tolerant and is designed to be deployed on low hardware. While retrieving the file from multiple servers,if any one of the servers that have the chunk part of the requested file was failed, the HDFS communicates with MongoDB and gives the requested file from its own shallow copy, which is nothing but the duplicate copy of the original file. Distributed file systemare the key building blocks for cloud computing applications which partitions files into number of chunks and allocates into the servers and retrieves based on the Map Reduce programming paradigm.In such file systems nodes simultaneously serve computing and storage functions. Files can also be dynamically created ,deleted and appended. This results load imbalance in DFS among the Nodes; Emerging DFS strongly depend on a central node for chunk reallocation.This dependence is clearly inadequate in a large-scale, failure prone environment.

This paper focus on,a fully distributed load rebalancing algorithm proposed to cope with the load imbalance problem as well as providing security while retrieving the file by using Kerberos authentication protocol . This algorithm is compared against centralized approach in a production system and a competing distributed solution. The simulation results outperforms the prior distributed algorithm in terms of load imbalance factor, movement cost and algorithm overhead. Brief study of load balancing problem in DFS specialized for large-scale, cloud has hundreds of or

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 5, Issue 12, December 2016**

thousands of nodes. Objective is to allocate the chunks of files as equally as possible among the nodes such that no node manages excessive number of chunks additionally aim to reduce network traffic caused by load rebalancing and to maximize the network bandwidth available to normal application and to improve the system performance.

## II. RELATED WORK

1. Load balancing in structured P2P systems [1]based on the concept of virtual servers the many-to-many framework is to survive with the load imbalance in a DHT. In the many-to-many framework light and heavy nodes are register their loads with some dedicated nodes name as the directories. This directories compute the matches between heavy and light nodes. Then request the heavy and light nodes to transfer and to receive designated virtual servers
2. Map Reduce is a programming model and an associated implementation[2] for processing and generating large data sets. Users specify a map function that processes a key/value pair to generate a set of intermediate key. Many real world tasks are expressible in this model, shown in the paper. Programs written in this functional style are automatically parallelized and executed on a large cluster of commodity machines. The runtime system takes care of the details of partitioning the input data, scheduling the program's execution across a set of machines, handling machine failures, and managing the required inter-machine communication. This allows programmers without any experience with parallel and distributed systems to easily utilize the resources of a large distributed system.
3. Distributed file systems are key building blocks for cloud computing applications[3] based on the MapReduce programming paradigm. In such file systems, nodes simultaneously serve computing and storage functions; a file is partitioned into a number of chunks allocated in distinct nodes so that Map Reduce tasks can perfom in parallel over the nodes.However, in a cloud computing environment, failure is the norm, and nodes may be upgraded, replaced and added in the system. Files can also be dynamically created, deleted, and appended in the system. This results in load imbalance in a distributed file system; that is, the file chunks are not distributed as uniformly as possible among the nodes. Emerging distributed file systems in production systems strongly depend on a central node for chunk reallocation..

## III. EXISTING SYSTEM

Distributed File Systems support hybrid mode of cloud that may be a combination of public and personal networks. In such file systems, nodes at the same time perform computing and storage functions. During a massive scale, dynamic and data-intensive clouds, load rebalancing is one amongst the crucial issues to be looked into and become inefficient.In DFS , the massive chunk of files area unit allotted equally among the nodes in order that no node manages associate degree excessive range of chunks.

- High movement cost
- High network traffic
- Load imbalancing
- Security difficulties
- Relying on central node
- Not implemented in HDFS.

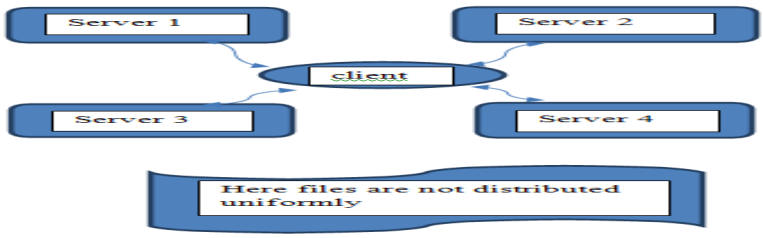



**Figure1: Existing System Diagram**

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 5, Issue 12, December 2016**

## IV. PROPOSEDALGORITHM

An algorithm is devised in such a way that the user can configure the server to upload the data into the cloud. When the user uploads file, it splits into the number of chunks . Different split chunks are stored at different servers through HDFS in data nodes in an encryption format for security .If any unauthorized user can login to the system they can't get the data because of encrypted data. Mongo DB is used for database handling and management of files.
Whenever the user wants to download a data, they should request for Ticket from KDC (Key Distribution Center). The Kerberos third party trusted authentication is used for the sake of security.

**Advantages of proposed system:**

• Implemented in Hadoop distributed file system is further investigated in cluster environment.
• As apache Hadoop is used security problems arises so, Kerberos authentication protocol is implemented for primary authentication.
• Reduced movement cost.

## V. IMPLEMENTATION

This section will discuss Algorithm implementation and Module description

**I.** Load Balancing Algorithm: Here the Load Rebalancing algorithm implements to relieve the load of name node; data nodes perform the load rebalancing task spontaneously without name node and each file is assigned with a unique ID. This algorithm distribute file on various chunk servers (nodes) with equal size.

- File load is proportional to number of chunks.
- Spontaneously interact with Name node to gather light node and heavy data nodes, including locations of chunks.
- Reallocates light nodes with heavy nodes chunks.
- Finally , data node perceives a nearly equal computational load.
-

**II. HDFS Architecture and Algorithmic Flow(per datanode)**

In general, below HDFS architecture shows the node distribution as a Name node and Data node. The Name node which contains the information about the chunk ID only; that is every file chunk is enabled with a unique ID. The Data node contains which contains data of the file which is uploaded. The Map Reduce programming paradigm maps the chunk ID and data in Data node and deploys as a single file.

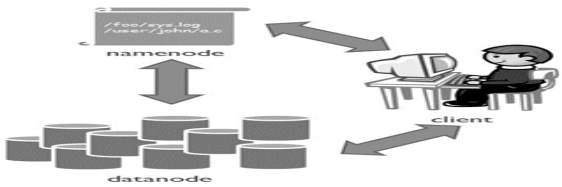



**Figure2: HDFS Architecture**

**FLOW CHART FOR NODE BALANCING**:

The flow chart shows Algorithm implementation , After distribution of file chunk by HDFS whenever the nodes will upgraded by the user ;if any of the will be overloaded automatically the corresponding data nodes will get pair and manages or distributes chunks as equally as possible therefore load rebalancing will be done.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 5, Issue 12, December 2016**

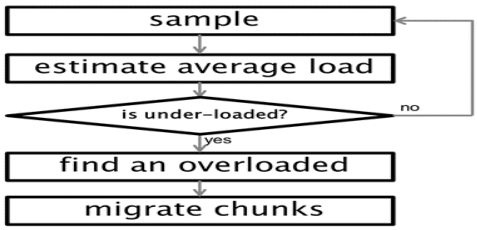



**Figure3: Algorithmic Flow**

## III. MODULES IMPLEMENTATION

### A. REGISTRATION:

In this module , the user has got to register within the page provided, so as to use the cloud services and find into the system. The private detailes of the user has got to be submitted to the system.

### B. KEY AUTHENTICATION:

Both the user and the admin has to enter with their login passwords to get their tokens. Kerberos center will provide tokens to the authorized users and admin. That token will be valid only for a certain of time, in order to prevent fraudulent activities.

### C. SERVER ACCESS WITH MONGODB:

The servers that area unit to be connected with the cloud will gets authentication with the user and prepares for distribution of data which the user uploads.

### D. UPLOAD THE FILE:

Client from a laptop, a Mobile or any authorized user uploads his file to the cloud storage after getting authorization from the system will be stored as chunks distributed by the HDFS, in encryption mode for the sake of security in compressed form..Hence the load balanced between the nodes.Then the user can download his own file whenever needed.

### E. DOWNLOAD THE FILE:

User gets his own file from the server by requesting the cloud when login into the cloud. The HDFS communicates with Mongo DB and gets user requested file from servers , while getting users file distributed as chunks the Map Reduce programming paradigm will gives file as one file by combining the chunks and provides it to user as a single file.

## VI. SYSTEM ARCHITECTURE

The user has to request to the cloud for the desired data it will be downloaded from the cloud. The user will be sent an email with the download token. Upon receiving the token via Gmail . The user has to enter the token while downloading from the Dropbox cloud. The file will be served to the user request from the available servers in the HDFS.

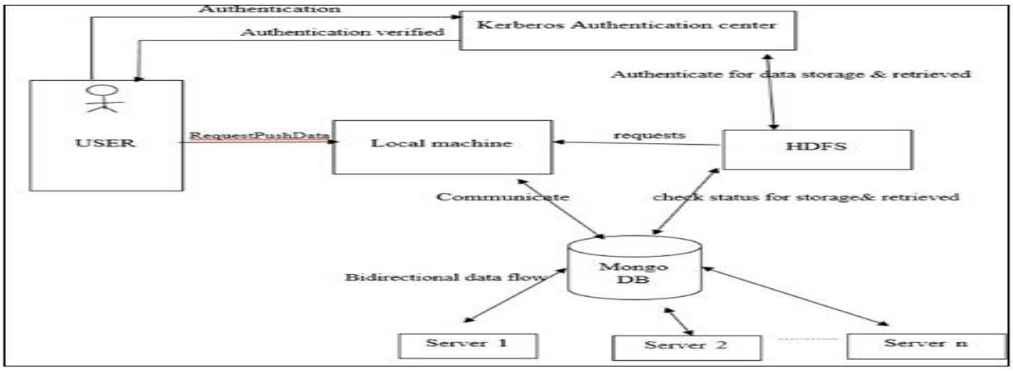



**Figure4: System Architecture**

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 5, Issue 12, December 2016**

## VII. TESTCASES

The user when gets Registration , file upload and download this test cases shows whether passed or failed.
1.The user first Login with his personal details ,which he has assigned with personal Id and password during the Registration into the system.
2.If any unauthorized user wants to login into the system it doesn't provides permission .
3.The user who Logged in can upload his desired data or file into the system, HDFS then divides the file as chunks and distributes to Central node or Name node and Data node.
4.The Mongo DB will stores the uploaded file chunks and converts the data in encryption format .
5. If the user wants to download or upload his file the Kerberos authentication protocol will send token to the user with a message sent to his email immediately for further activities.
6.The user when gets authentication from the system he can get his desired file as he uploaded in a decrypted manner.

| INPUT | EXPECTED BEHAVIOUR | OBSERVED BEHAVIOUR | STATUS P=PASS F=FAIL |
|---|---|---|---|
| 1.Login as user with his personal details | Login page and access granted to his request | -do- | P |
| 2.Login as user with wrong details | Displays not an authorized user | -do- | P |
| 3.Upload user's file | The file will be stored as chunks by HDFS and equally distributed. | -do- | P |
| 4.Mongo DB retrieves file location using Map Reduce paradigm | Decrypted File is downloaded as a single file | -do- | P |
| 5.Kerberos authentication is provided to user | User will get status about his file retrieving | -do- | P |
| 6.Download his own file requested by user | Decrypted single file will be downloaded to user. | -do- | P |

**Table1: Test cases during the process**

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

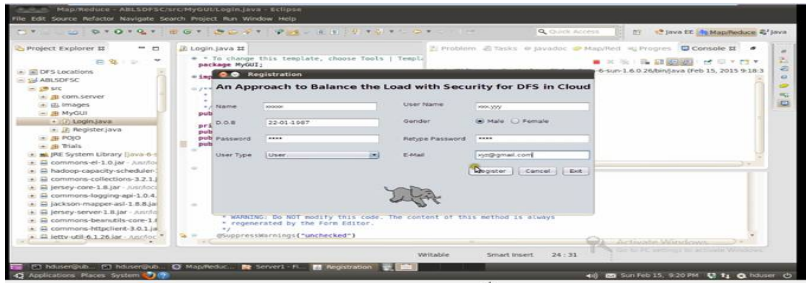
International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 12, December 2016

## VIII. SIMULATION AND RESULTS

### TEST-1: REGISTRATION PAGE FOR USER

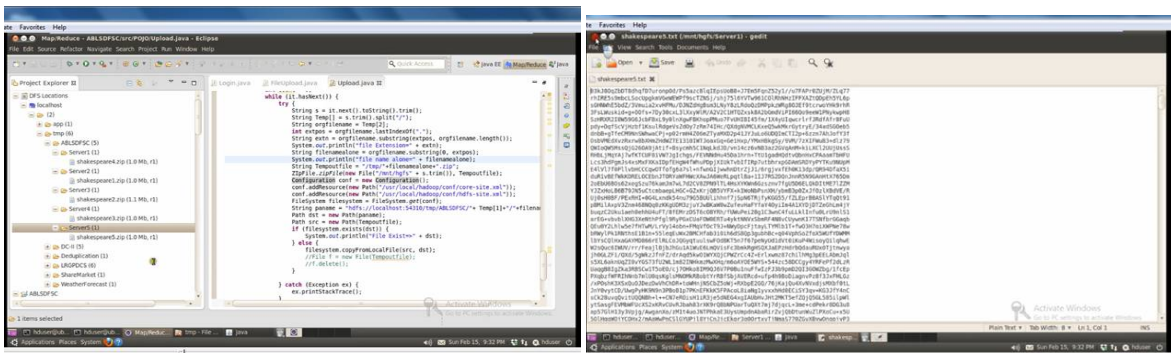



In this registration page the user will enters with his own personal details, After registering the system provides an Id and password for privacy for further Logins and other proceedings and registered successfully. Now the user can Login into the system at anytime and from anywhere.

### TEST-2: UPLOAD THE FILE

This page shows that the users file which is uploaded after the Login into the system and it is divided as a file chunk by the HDFS, which is stored in a data node.

### TEST-3: DISTRIBUTION OF FILE CHUNKS



The file chunks distributed as equally in an compressed format in the nodes by HDFS .Those file chunks will be stored in encryption format because if any of the unauthorized get Login into the system he can't get the data as usual

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

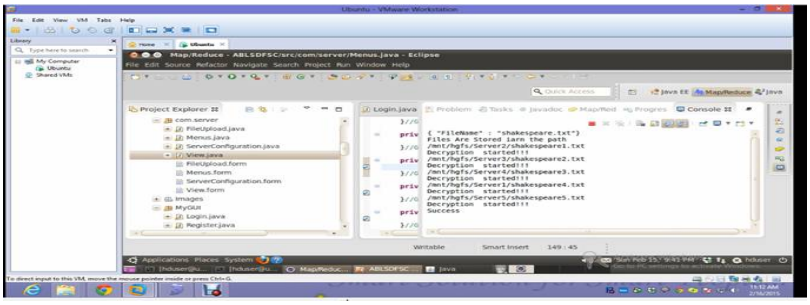
International Journal of Innovative Research in Science, Engineering and Technology

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 5, Issue 12, December 2016**

**TEST-4: DOWNLOAD FILE**

Here the user is getting his uploaded file from the system in an decryption format and is downloaded as a single file successfully, with his personal login and tokens sent to his email.



## IX. CONCLUSION

In this paper balancing the loads of nodes and reducing the demanded movement cost as much as possible. Competing the modified rebalancing algorithms with centralized algorithms through synthesized probabilistic distribution of file chunks are compared in case of movement cost, Network traffic and algorithm overhead are efficiently. The whole implementation is done in strict Hadoop environment and almost reached successful result. Future work, There is scope to this rebalancing algorithm will efficiently works in large-scale cluster environment.

## REFERENCES

[1] .A.Rao, K.Lakshminarayanan, S.Surana, R.Karp and I.Stoica .," Load Balancing in Structured P2P Systems", Second Int |Workshop Peer- to-Peer Systems (IPTS,,02),pp.68-79, Feb.2003.
[2]. Jeffrey Dean and Sanjay Ghemawat.." MapReduce Simplified data processing on large clusters In OSDI", pages 137–150, 2004.
[3]. Hung-Chang Hsiao, Member, IEEE Computer Society, 18Hsueh-Yi Chung, HaiyingShen, Member, IEEE, and Yu-Chang Chao "Load Rebalancing for Distributed File Systems in Clouds", may 2013.
[4]. Raicu, I.T. Foster, and P. Beckman, "Making a Case for Distributed File Systems at Exascale," Proc. Third Int'l Workshop Large-Scale System and Application Performance (LSAP '11), June 2011.
[5]. Q.H. Vu, B.C. Ooi, M. Rinard, and K-L Tan, "Histogram-Based Global Load Balancing in Structured Peer-to-Peer Systems," IEEE Trans. Knowledge Data Eng, Apr 2009.
[6]. D. Karger and M. Ruhl. "Simple Efficient Load Balancing Algorithms for Peer-to-Peer Systems," ACM Symp. Parallel Algorithms and Architectures (SPAA'04), June 2004
[7].Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek and Hari Balakrishnan. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications.‖ in Pmc. ACM SIGCOMM. San Diego, pp. 149-160,2001.
[8]. John Byers, Jeffrey Considine, and Michael Mituenmacher, ―Simple Load Balancing for Distributed Hash Tables._,,in Pmc.IPTfS, Feb. 2003.
[9]. D. N. Rewadkar, and Juhi Shah, "A Survey On Load Rebalancing for Distributed File System," in Proc. Int. Jou. Sci. Research, IJSR, vol. 3, Issue 11, Nov. 2014.
[10]. J. R. Douceur and R. P. Watlenhofer, "Competitive Hill-Climbing Strategies for Replica Placement in a Distributed File System,‖ in Pmc. DISC,2001.
[11]. J. Byers, J. Considine, and M. Mitzenmacher, Simple load balancing for distributed hash tables, in Proceedings of IPTPS, Berkeley, CA, Feb. 2003.
[12]. J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," in Proc. 6th Symp,Operating System Design and Implementation (OSDI"04), pp. 137–150, Dec. 2004.