ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 3, Issue 6, December 2013

Enhance Load Rebalance Algorithm for Distributed File Systems in Clouds

Kokilavani .K,
Department Of Pervasive Computing Technology, Kings College Of Engineering,
Punalkulam, Tamil nadu

Abstract— This paper examines the load rebalancing problem in cloud computing. The main objective of the paper is to Enhance distributed load rebalancing algorithm to cope with the load imbalance factor, movement cost, and algorithmic overhead. The load rebalance algorithm is compared against a centralized approach in a production system and the performance of the proposal implemented in the Hadoop distributed file system for cloud computing applications. We investigate to implement security provided for cloud computing and Evaluate the Quality of Service-QOS (Ex. Response Time) of whole system. In cloud computing one server controls number of sub servers, files, it can add, delete, and append dynamically. Any file sharing (uploading or downloading) are stored in sub servers and retrieved from sub servers. Before uploading and downloading that files are stored in encrypted format and retrieve in a decrypted format. In our implementation the key is sent to the user's email id, user uses that key to view and download the files from sub server. For encryption and decryption schemes we are using RSA algorithm.

***Index Terms*—Hadoop distributed file systems, QOS, RSA, cloud computing, load rebalancing.**

I. INTRODUCTION

Cloud computing refers to delivery of computer resources from a remote place based on user needs. Network connections are necessary to access information and utilize resources. According to Gartner [1], cloud computing can be defined as, a style of computing, where massively scalable IT-enabled capabilities are delivered 'as a service' to external customers using Internet technologies. According to the Seccombe [2] and National Institute of Standards & Technology [3], guidelines for cloud computing, it has four different deployment models namely private, community, public and hybrid. Performance, security, data locality to both cloud architects and end users are the key features of public model. Increase in the challenges on how to transfer and where to store and compute data are the issues caused by large distributed file systems in cloud computing. Cloud Computing Technologies such as Map Reduce paradigm [4], virtualization [5] and Distributed File Systems ([6], [7]) are used to achieve scalability and reliability in clouds. Hadoop File Systems (HDFS) and Google File Systems (GFS) are used to overcome the issues which arise in achieving those factors .HDFS cluster consist of single name node and a server manages the file system namespace and regulates access to files. Load balancing is the main issue in large scale distributed computing. It is the process of distributing tasks to all nodes involved in cloud computing. Efficient allocation of resources to every computing task helps to achieve resource utilization ratio and high user satisfaction. Minimizing resource consumption, avoiding bottlenecks, implementing fail-over, enabling scalability, reducing network inconsistencies and solving network traffic are the main goals of load computing. Whole cloud gets fail while analyzing existing system clouds performance bottleneck due to failure of central node. Functional difficulties and technical difficulties are caused because of those failures. Cloud computing allocate resources dynamically, which connects and add thousands of nodes together. The main goal is to allocate files to these nodes, for avoiding heavy nodes that files are uniformly distributed to these nodes. Load balancing provides maximization of network bandwidth, reduction of network traffic and network inconsistencies. We can add, delete and update nodes dynamically for heterogeneity of the nodes. Heterogeneity of the nodes will increase the scalability and system performance. In Distributed File System the main functionalities of nodes is to serve computing and storage functions. In this paper, we introduced new load rebalancing algorithm in distributed file systems, aim to reduce network traffic (or *movement cost*) caused by rebalancing the loads of nodes as much as possible to increase the network bandwidth to cloud applications. For security in cloud computing, we maintained these files in encrypted format using cryptographic algorithms. Finally, for performance improvement, high speed, reducing time consistency, we calculate the response time of whole system.

II. LITERATURE SURVEY

When you submit your Load balancing in cloud computing system [8] **Ram Prasad Padhy, P Goutam Prasad Rao** discussed on basic concepts of Cloud Computing and Load balancing and studied some existing load balancing algorithms, which can be applied to clouds. In addition to that, the closed-form solutions for minimum measurement and reporting time for single level tree networks with different load balancing strategies were also studied. The performance of these strategies with respect to the timing and the effect of link and measurement speed were studied. A Comparative Study into Distributed Load Balancing Algorithms for Cloud Computing [9] **M. Randles, D.Lamb, and A.Taleb-Bendiab** discussed on Distributed Load Balancing

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 3, Issue 6, December 2013

Algorithms are Honeybee Foraging Behavior, Biased Random Sampling, and Active Clustering. Honeybee Foraging Behavior investigated a decentralized honeybee-based load balancing technique that is a nature-inspired algorithm for self-organization. It achieves global load balancing through local server actions. Performance of the system is enhanced with increased system diversity but throughput is not increased with an increase in system size. It is best suited for the conditions where the diverse population of service types is required. Biased Random Sampling investigated a distributed and scalable load balancing approach that uses random sampling of the system domain to achieve self-organization thus balancing the load across all nodes of the system. The performance of the system is improved with high and similar population of resources thus resulting in an increased throughput by effectively utilizing the increased system resources. It is degraded with an increase in population diversity. Active Clustering investigated a self-aggregation Load Balancing Techniques that is self aggregation algorithm to optimize job assignments by connecting similar services using local re-wiring. The performance of the system is enhanced with high resources thereby increasing the throughput by using these resources effectively. It is degraded with an increase in system diversity. Game-theoretic static load balancing for distributed systems [10] **Penmatsa and Chronopoulos** discussed on static load balancing strategy based on game theory for distributed systems. And this work provides us with a new review of the load balance problem in the cloud environment. As an implementation of distributed system, the load balancing in the cloud computing environment can be viewed as a game. Load Balancing in Structured P2P Systems [11] **A.Rao, K.Lakshminarayanan, S.Surana, R.Karp and I.Stoica** based on the concept of virtual servers the many-to-many framework is to cope with the load imbalance in a DHT. In the many-to-many framework light and heavy nodes register their loads with some dedicated nodes namely, the directories. The directories compute matches between heavy and light nodes and then respectively, request the heavy and light nodes to transfer and to receive designated virtual servers. Load Balancing in Dynamic Structured P2P Systems [12] **S.Surana, B.Godfrey, K. Lakshmi narayanan, R. Karp and I.Stoica** discussed on the many-to-many framework essentially reduces the load balancing problem to a centralized algorithmic problem. As the entire system heavily depends on the directory nodes, the directory nodes may thus become the performance bottleneck and single point of failure. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications [13] **Ion Stoic a, Robert Morris, David Karger, M. Frans Kaashoek and Hari Balakrishnan** propose the notion of virtual servers as a means of improving load balance. By allocating log *N* virtual servers per physical node, Chord ensures that with high probability the number of objects on any node is within a constant factor of the average. However, these schemes assume that nodes are homogeneous; objects have the same size, and object IDS are uniformly distributed. Simple Load Balancing for Distributed Hash Tables [14] **John Byers, Jeffrey Considine, and Michael Mitzenmache** has proposed the use of the power of two choices paradigm to achieve better load balance. However, this scheme was not analyzed or simulated for the case of heterogeneous object sizes and node capacities, and in any case **is** not prepared to handle a dynamic system of the kind which we have described. This is largely complementary to the work presented in this paper. Competitive Hill-Climbing Strategies **for** Replica Placement in a Distributed File System [15] **J. R. Douceur and R. P. Watlenhofer** have proposed algorithms for replica placement in distributed file systems which are similar in spirit with our algorithms. However, their primary goal is to place object replicas to maximize the availability in an un-trusted **P2P** system, while we consider the load balancing problem in a cooperative system.

III. EXISTING SYSTEM

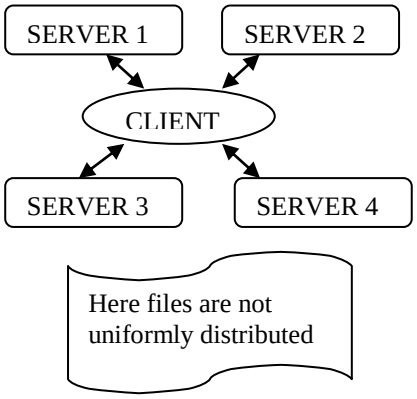

Fig1. Existing system Diagram

When examine the existing system cloud computing depends on centralized nodes. Performance bottleneck occurs because the failure of central nodes causes the failure of the whole system. Load imbalance exists, since the loads are not uniformly distributed to these nodes. This dependence is inefficient i.e. not achieving maximum productivity of whole system. It conducts technical and functional problems.

Limitations of Existing System

- High Movement Cost
- High Network Traffic
- Algorithm Overhead
- Load Imbalance
- Relying on Central Node
- Security difficulties

IV. PROPOSED SYSTEM

Let the set of files be F. Files in F may be dynamically created, deleted, and appended. Our proposal maintains main server and sub servers. Sub servers are denoted by S. Nodes are divided into two types based on their load, they are

- Light node
- Heavy node

The proposed enhance load rebalancing algorithm first evaluates whether the loads are light (under loaded) or heavy

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 3, Issue 6, December 2013

(overloaded) in each sub servers without global knowledge. All heavy loads are changed in to light nodes. F are downloading or uploading with the aid of the centralized system. Load equalization technique used to distribute the F uniformly into sub servers.

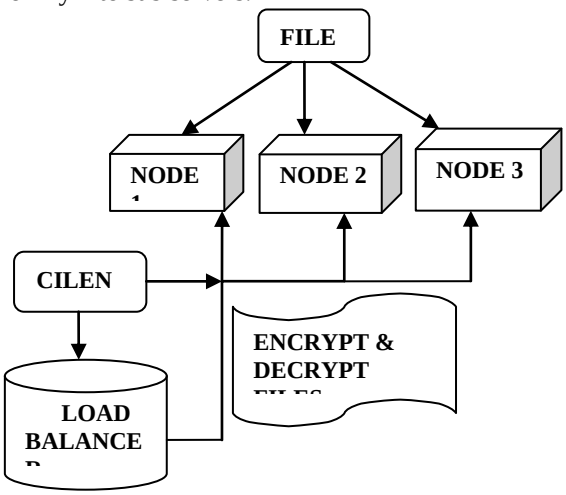

Fig2. Load Rebalance in DFS - System Architecture

The advantage of the technique is to reduce latency, isolated overload, and great utilization of resource provident outcome. Finally examine the QOS (i.e., bandwidth, response time) owing to enhance system performance. For security reason, these files F are encrypted before sharing (uploading & downloading) the files F to sub servers using cryptographic techniques and decrypted while downloading. Our proposal, Eliminates the dependence on central nodes. The storage nodes are structured as a network based on distributed hash tables. DHTs enable nodes to self-organize and repair while constantly offering lookup functionality in node dynamism, simplifying the system provision and management. Our algorithm is compared against a centralized approach in a production system which uniformly distributes across sub servers. Due to centralized approach, outperforms the previous distributed algorithmic rule in terms of load imbalance issue, movement value, and recursive overhead. Our proposal is implemented in the Hadoop Distributed File System.

Advantages of Proposed System

- Deploy wide-range and failure error domain
- Reduce Network Traffic or Movement Cost
- Maximize the Network Bandwidth
- Improve overall System Performance
- Utilizes Physical Network Locality
- Better throughput and response time
- System consistency (i.e., avoid data loss)
- Excellent security
- Picking lead of node heterogeneity
- Extends resource utilization
- Speedy & Diminishes time consistency

V. IMPLEMENTATION

In the proposed system five different modules are implemented. They are discussed below,

- File Allotment
- DHT Devising
- Load Rebalancing algorithm
- Security
- Assist QoS

File Allotment

Our intention is to allocate files in to sub servers. Number of files can be add, delete, or appended dynamically in to sub servers. The files are distributed or allocated as uniformly as feasible among the sub servers. This will help to retain system consistent (avoid data loss).

DHT Devising

Load rebalancing charge to storage nodes by having the storage nodes balance their loads un-promptly. This abolishes the addiction on central nodes. The storage nodes are structured as a network based on distributed hash tables (DHT). It authorizes nodes to self-organize and repair while constantly offering lookup functionality in node dynamism, simplifying the system provision and management.

Load Rebalancing Algorithm

First evaluate whether the loads are light (under loaded) or heavy (overloaded) in each sub servers without global knowledge. All heavy loads are changed in to light nodes. F are downloading or uploading with the aid of the centralized system. Load equalization technique used to distribute the F uniformly in to sub servers. The advantage of the technique is reducing latency, isolated overload, and great utilization of resources provident outcome. Files F are needed to upload; these files are stored in all nodes. Files F are needed to download; these files are retrieved from all nodes. For this to achieve Utilize Physical Network Locality, Picking lead of node heterogeneity, handle replicas and improve overall system performance.

Security

Cloud computing is an emerging technology that is still unclear to many security problems. Ensuring the security of stored data in cloud servers is one of the most challenging issues in such environments. The main aim of this project is to use the cryptography concepts in cloud computing communications and to increase the security of encrypted data in cloud servers with the least consumption of time and cost at the both of encryption and decryption Processes. To make sure the security of data, our proposed a method of providing security by implementing RSA algorithm, the encrypted data's that will be stored in the sub servers. The key send to user's mail id, user can access original data through this key. Otherwise user's can get only cipher text without key.

RSA algorithm

The most commonly used asymmetric algorithm is Rivest-Shamir-Adleman (RSA). It was introduced by its three inventors, Ronald Rivest, Adi Shamir and Leonard Adleman

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 3, Issue 6, December 2013

in 1977. It is mostly used in key distribution and digital signature processes. RSA is based on a one-way function in number theory, called ―integer factorization. RSA is a block cipher in which every message is mapped to an integer. RSA consists of public-key and private-key [16]. Encryption is done by the Cloud service provider and decryption is done by the Cloud user or consumer. Once the data is encrypted with the Public-Key, it can be decrypted with the corresponding Private-Key only.

We use the RSA algorithm (William, 2005) as a basis to provide data-centric security for shared files.

RSA algorithm involves three steps [17].

(i) First, in Key generation before the data is encrypted, Key generation should be done. This process is done between the Cloud service provider and the user.

(ii) Second, in Encryption is the process of converting original plain text (data) into cipher text (data).

(iii)Third, Decryption is the process of converting the cipher text (data) to the original plain text (data).

A. Key Generation Algorithm

1) Randomly and secretly choose two large primes: p, q and compute n = p. q

2) Compute ϕ (n) = (p – 1) (q – 1).

3) Select Random Integer: e such as 1< e< n and gcd (e, ϕ) = 1.

4) Compute d such as e. d ≡ 1 mod ϕ (n) and 1 < d < ϕ (n).

5) Public Key: (e, n)

6) Private Key: (d, n).

7) This key sends to user's email id.

B. Encryption Process

1) Suppose user needs to view uploaded files from server.

2) Server should send his public key to user's email id.

3) Cipher text c = message e mod n

C. Decryption Process

1) Plain text p = cipher text d mod n.

2) User can view a file from server after entering the correct key. That key will be generate and send to the user's email id.

Only authorized user retrieve the encrypted data and decrypt it. Even if anyone happens to read the data accidentally, the original meaning of the data will not be understood. The most important advantage of RSA is ensuring about the privacy of the private key because this key will not be transmitted or revealed to another user.

Assist Qos

Quality of Service (QoS) is a broad term used to describe the overall experience a user or application will receive over a network. In cloud computing is a way of computing model in which frequently real time scalable resources acting as a file, data, programs, hardware, and third party services can be accessible from a Web browser via the Internet to users. These customers pay only for the used computer resources and services by means of customized service level agreement. The SLA is a contract negotiated and agreed between a customer and a service provider. That is, the service provider is required to execute service requests from a customer within negotiated quality of service (QoS) requirements for a given price. According to QoS requirements a customer is in general concerned about response time rather than throughput in QoS requirements. Cloud computing specifically, in an effort to deliver QoS guaranteed services in such a computing environment, we find the relationship among the maximal number of customers, the minimal service resources and the highest level of services, calculate response time. Our proposed calculate the response time of whole system. The main advantage of QoS in cloud computing, to improve the performance of whole system, ensures high speed, reduce time consistency. Uniformly distributed these files so, avoid delay, data loss of whole system.

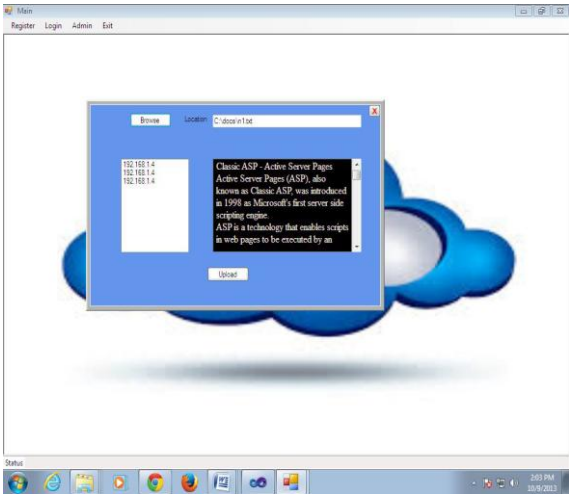

Fig 3.User Browse Files from Sub Servers

Fig 4.User enter key for view files from sub servers

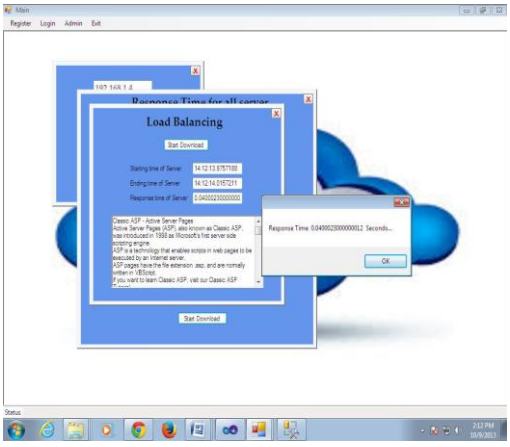

Fig5.Evaluate response time of whole system

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 3, Issue 6, December 2013

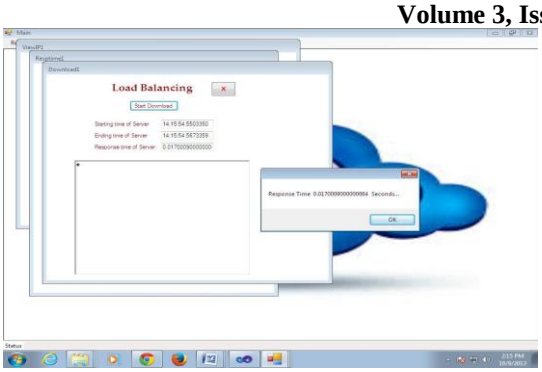

Fig 6. User download files without authentication (so, user get only cipher text not original files)

VII. CONCLUSION

Balancing the loads of nodes and reducing the demanded movement cost as much as possible, while taking advantage of physical network locality and node heterogeneity is done by using proposed idea. Competing algorithms through synthesized probabilistic distributions of file chunks are compared with proposed idea. Emerging distributed file systems in production systems strongly depend on a central node for chunk reallocation. This dependence is clearly inadequate in a large-scale, failure-prone environment because the central load balancer is put under considerable workload that is linearly scaled with the system size, and may thus become the performance bottleneck and the single point of failure. Centralized approach in a production system and a competing distributed method are compared with proposed algorithm. Load imbalance factor, movement cost, and algorithmic overhead are handled by developed algorithm efficiently. To securing the data, implemented the RSA algorithm. Examine the Performance measures of whole system.

VII. FUTURE WORK

Use any other balancing algorithm strategy in cloud computing environment. In this paper only file sharing (file download & upload). In future sharing multimedia files (audio & video).This algorithm used between two various cloud computing environments. Use various cryptographic algorithms for security purposes.

REFERENCES

[1] Heiser J. What you need to know about cloud computing security and compliance, Gartner, Research, ID Number: G00168345, 2009.

[2] Seccombe A.., Hutton A, Meisel A, Windel A, Mohammed A, Licciardi A, (2009). Security guidance for critical areas of focus in cloud computing, v2.1. Cloud Security Alliance, 25 p.

[3] Mell P, Grance T (2011) The NIST definition of Cloud Computing. NIST, Special Publication 800–145, Gaithersburg, MD.

[4] J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," in Proc. 6th Symp,Operating System Design and Implementation (OSDI'04), Dec. 2004, pp. 137–150

[5] J. Byers, J. Considine, and M. Mitzenmacher, Simple load balancing for distributed hash tables, in Proceedings of IPTPS, Berkeley, CA, Feb. 2003.

[6] Hadoop Distributed File System, http://hadoop.apache.org/hdfs/.

[7] Hadoop Distributed File System "Rebalancing Blocks," ,"http://developer.yahoo.com/hadoop/tutorial/module2.html#rebalancing, 2012.

[8] Ram Prasad Padhy (107CS046), PGoutam Prasad Rao (107CS039)."Load balancing in cloud computing system" Department of Computer Science and Engineering National Institute of Technology, Rourkela Rourkela-769 008, Orissa, India May, 2011.

[9] M. Randles, D. Lamb, and A. Taleb-Bendiab, ―A Comparative Study into Distributed Load Balancing Algorithms for Cloud Computing‖, Proceedings of 24th IEEE International Conference on Advanced Information Networking and Applications Workshops, Perth, Australia, April 2010, pages 551-556.

[10] S.Penmatsa and T.Chronopoulos, Game-theoretic static load balancing for distributed systems, Journal of Parallel and Distributed Computing, vol.71, no.4, pp.537-555, Apr. 2011.

[11] A.Rao, K.Lakshmi narayanan, S.Surana, R.Karp and I.Stoica, Load Balancing in Structured P2P Systems‖, Proc. Second Int'l Workshop Peer-to-Peer Systems (IPTPS '02), pp. 68-79, Feb. 2003.

[12] S.Surana, B.Godfrey, K.Lakshminarayanan, R.Karp and I.Stoica,―Load Balancing in Dynamic Structured P2P Systems,‖ Performance Evaluation,vol.63, no. 6, pp. 217-240, Mar. 2006.

[13] 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, 2001, pp. 149-160.

[14] John Byers, Jeffrey Considine, and Michael Mituenmacher, ―Simple Load Balancing for Distributed Hash Tables._'in Pmc.IPTfS, Feb. 2003.

[15] J. R. Douceur and R. P. Watlenhofer, "Competitive Hill-Climbing Strategies for Replica Placement in a Distributed File System,‖ in Pmc. DISC,2001.

[16] R.L. Rivest, A. Shamir, and L. Adleman. "A Method for Obtaining Digital Signatures and Public key Cryptosystems", Communications of the ACM, 21(2), 120-126, February 1978.

[17] RSA Laboratories, High Speed RSA Implementation, RSA Data Security, November 1994.