International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Secure Load Rebalancing in Cloud Environment

Mannava Praveen Kumar[1], Srinivas LNB[2]

[1]Cloud Computing, Department of Information Technology, SRM University, Kattankulathur, Chennai, India

[2]Guide, Department of Information Technology, SRM University, Kattankulathur, Chennai, India

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***

Keywords: DFS, MapReduce, Load balancing, Distributed Hash Table, cloud

1. Introduction

In Cloud computing, the number of computers that are connected using communication network. The notation of cloud indicates that internet is mandatory to perform the various cloud operations i.e. to create delete, append and replace. It is used in IT-companies to share information and resources with the all users. There are various characteristics of cloud i.e. Scalable, on demand service, User centric, Powerful, Versatile, Platform independent etc. In cloud three technologies are included the MapReduce programming, Virtualization and distributed file systems for the data storage purpose. Distributed file system is classical model of file system that is used in the form of chunks for cloud computing. Cloud computing application is based on the MapReduce programming used in distributed file system. MapReduce is the master-slave architecture in Hadoop. Master act like Namenode and Slave act like Datanode. Master takes large problem, divides it into sub problem and assigns it to worker node i.e. to multiple slaves to solve problem individually. In distributed file system, a large file is divided into number of chunks and allocates each chunk to separate node to perform MapReduce function parallel over each node. For example in word count application it identifies the occurrences of each distinct word in large file. In this application a large file is divided into fixed-size chunks (parts) and assigns each chunk to different cloud storage node. Then each storage node calculates the occurrences of each distinct word by scanning and parsing its own chunk. Then give its result to master to calculate the final result. In distributed file system, the load of each node is directly proportional to number of file chunks that node consists. As the increase in storage and network, load balancing is the main issue in the large scale distributed systems. Load should be balance over multiple nodes to improve system performance, resource utilization, response time and stability. Load balancing is divided into two categories: static and dynamic. In static load balancing algorithm, it does not consider the previous behavior of a node while distribute the load. But in case of dynamic load balancing algorithm, it checks the previous behavior of node while distribute the load. In cloud, if number of storage nodes, number of files and assesses to that file increases then the central node (master in MapReduce) becomes bottleneck. The load rebalancing task is used to eliminate the load on central node. In load balancing algorithm, storage nodes are structured over network based on the distributed hash table (DHT); each file chunk having rapid key lookup in DHTs, in that unique identifier is assign to each file chunk [1]. DHTs enable nodes to self-recognize and repair while it constantly offers lookup functionality in node. Here aim to reduce the movement cost which is caused by load rebalancing of nodes to maximize the network bandwidth. Each Chunk server first find whether it is light node or heavy node without global knowledge of node. The numbers of file chunks are migrated from heavy node to light node to balance their load. This process repeats until all heavy nodes becomes the light nodes. To overcome this load balancing problem each node perform load rebalancing algorithm independently without global knowledge about load of all nodes. 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.

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. This dependence is clearly inadequate in a large-scale, failure-prone environment because the central load balancer is put under considerable workload that is

Licensed Under Creative Commons Attribution CC BY

linearly scaled with the system size, and may thus become the performance bottleneck and the single point of failure. In this paper, a fully distributed load rebalancing algorithm is presented to cope with the load imbalance problem. Our algorithm is compared against a centralized approach in a production system and a competing distributed solution presented in the literature**.**

2. Comparative Study

Literature survey is the most important step in software development process. Before developing the tool it is necessary to determine the time factor, economy n company strength. Once these things r satisfied, ten next steps are to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need lot of external support. This support can be obtained from senior programmers, from book or from websites. Before building the system the above consideration are taken into account for developing the proposed system.

MapReduce: Simplified Data Processing on Large Clusters MapReduce is the programming model used in implementation for processing and generating large scale datasets. It is used at Google for many different purposes. Here map and reduce functions are used. Map function generate set of intermediate key pairs and reduce function merges all intermediate key values associated with same intermediate key. The map and reduce function allows to perform parallelize operation easily and re-execute the mechanism for fault tolerance. At the run-time, system takes care of detail information of partitioning the input data, schedule the program execution across number of available machines,handling features and managing intercommunication between machines.

In distributed file system nodes simultaneously perform computing and storage operations. The large file in partitioned into number of chunks and allocate it to distinct nodes to perform MapReduce task parallel over nodes. Typically, MapReduce task processes on many terabytes of data on thousands of machines. This model is easy to use; it hides the details of parallelization, optimization, fault-tolerance and load balancing. MapReduce is used for Google's production Web search service, machine learning, data mining, etc. Using this programming model, redundant execution used to reduce the impact of slow machines, handle machine failure as well as data loss.

Load Balancing in Dynamic Structured P2P Systems may thus become the performance bottleneck and single point The performance of the system is enhanced with high resources thereby increasing the throughput by using these resources effectively. It is degraded with an increasing system diversity. Game-theoretic static load balancing for distributed systems.

[1] 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 the distributed system, the load balancing in the cloud computing environment can be viewed as a game. Load balancing in structured P2P systems.

[2] A.Rao, K.Lakshminarayanan, S.Surana, R.Karp and I.Stoica based on the concept of virtual 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

[3]. 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.

The chunkservers in our proposal are organized as a DHT network; that is, each chunkserver implements a DHT protocol such as Chord or Pastry . A file in the system is partitioned into a number of fixed-size chunks, and "each" chunk has a unique chunk handle (or chunk identifier) named with a globally known hash function such as SHA1 . The hash function returns a unique identifier for a given file's pathname string and a chunk index.

3. Proposed Architecture

We are interested in studying the load rebalancing problem in distributed file systems specialized for large-scale, dynamic and data-intensive cloud. (The terms "rebalance" and "balance" are interchangeable in this paper.) Such a large-scale cloud has hundreds or thousands of nodes (and may reach tens of thousands in the future). Our objective is to allocate the chunks of files as uniformly as possible among the nodes such that no node manages an excessive number of chunks. Additionally, we aim to reduce network traffic (or movement cost) caused by rebalancing the loads of nodes as much as possible to maximize the network bandwidth available to normal applications. Moreover, as failure is the norm, nodes are newly added to sustain the overall system performance, resulting in the heterogeneity of nodes. Exploiting capable nodes to improve the system performance is, thus, demanded. Our proposal not only takes advantage of physical network locality in the reallocation of file chunks to reduce the movement cost but also exploits capable nodes to improve the overall system performance.

3.1 Advantages of Proposed System

This eliminates the dependence on central nodes. Our proposed algorithm operates in a distributed manner in which nodes perform their load-balancing tasks independently without synchronization or global knowledge regarding the system. Algorithm reduces algorithmic overhead introduced to the DHTs as much as possible.

Licensed Under Creative Commons Attribution CC BY

3.1.1 Algorithm Used

Load Rebalancing Algorithm

3.1.2 Load Rebalancing Algorithm

In computing, loadrebalancing distributes workloads acro ss multiple computing resources, such as computers, a computer cluster, network links, central processing units or disk drives. Load balancing aims to optimize resource use, maximize throughput, minimize response time, and avoid overload of any single resource. Using multiple components with load balancing instead of a single component may increase reliability through redundancy. Load balancing usually involves
dedicated software or hardware, such as a multilayer switch or a Domain Name System server process.

3.2 Modules

Module creation DHT formulation Load balancing algorithm Group Evaluation

3.2.1 Module Creation:

Our objective is to allocate the modules of files as uniformly as possible among the nodes such that no node manages an excessive number of modules. A file is partitioned into a number of modules allocated in different nodes so that Map Reduce Tasks can be performed in parallel over the nodes.

3.2.2 DHT Formulation:

The module servers in our proposal are organized as a DHT network. Typical DHTs guarantee that if a node leaves, then its locally hosted modules are reliably migrated to its successor; if a node joins, then it allocates the modules whose IDs immediately precede the joining node from its successor to manage.

3.2.3 Load Rebalancing Algorithm

In our proposed algorithm, each module server node I first estimate whether it is under loaded (light) or overloaded (heavy) without global knowledge. A node is light if the number of modules it hosts is smaller than the threshold. First of all we will find the lightest node to take the set of modules from heaviest node. So we can do the process without failure.

3.2.4 Graph evaluation

The overall dynamic resource allocation in large cloud environment is evaluated and displayed as pie graph. The graph shows the overall usage of the virtual machines in the cloud. The graph shows the maximum utility of the Virtual machine under memory constraints. Equal usage of all virtual machines in the cloud is shown in the graph.

4. Implementation

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective.

The implementation stage involves careful planning, investigation of the existing system and it's constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

1. Data Owner Registration
2. Data User Registration
3. TTP (Trusted Third Party) LOGIN
4. CSP(Cloud Service Provider) LOGIN
5. Download File

Data Owner Registration

In this module if a owner of data(File) have to store data on a cloud server, he/she should register their details first. These details are maintained in a Database.Then he has to upload the file in a file database. The file which are stored in a database are in an encrypted form. Authorized users can only decode it.

Data User Registration

In this module if a user wants to access the data which is stored in a cloud server, he/she should register their details first. These details are maintained in a Database.

TTP LOGIN:

In this module TTP has monitors the data owners file by verifying the data owner's file and stored the file in a database. Also ttp checks the CSP and find out whether the csp is authorized one or not.

CSP LOGIN

In this module CSP has to login first. Then only he can store the file in his cloud server.Ttp can only check the csp whether the csp is authorized csp or not. If its fake, ttp won't allow the file to store in cloud server.

Based on this Paper

A file is partitioned into a number of chunks allocated in distinct nodes so that MapReduce tasks can be performed in parallel over the nodes. For example, consider a wordcount application that counts the number of distinct words and the frequency of each unique word in a large file. In such an application, a cloud partitions the file into a large number of disjointed and fixed-size pieces (or file chunks) and assigns them to different cloud storage nodes (i.e., chunkservers).Also we have developed word count application and word search application.

Download File

If the user is an authorized user,he/she can download the file by using meta data of the file which have uploaded and divided.

System Design

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective. The implementation stage involves careful planning, investigation of the existing system and it's constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

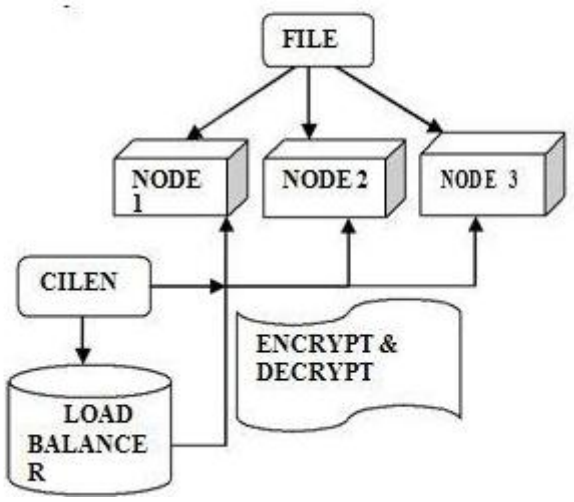

Figure 1: Load Rebalance - System Architecture

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.

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

Mechanism: AES_ENCRYPT & AES_DECRYPT Here we are using this aes_encrypt & aes_decrypt for encryption and decryption. The file we have uploaded which has to be in encrypted form and decrypt it by using key, also we are sending mail to the recipient using the following codes.

5. Conclusion

A novel load-balancing algorithm to deal with the load rebalancing problem in large-scale, dynamic, and distributed file systems in clouds has been presented in this paper. Our proposal strives to balance the loads of nodes and reduce the demanded movement cost as much as possible, while taking advantage of physical network locality and node heterogeneity. Particularly, our load-balancing algorithm exhibits a fast convergence rate. The efficiency and effectiveness of our design are further validated by analytical models and a real implementation with a small-scale cluster environment. 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.

References

[1] 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.
[2] 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.
[3] 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.
[4] 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.
[5] fig(1) from Enhanced Load Rebalancing algorithm for Distributed File Systems In Clouds. (International Journal of Engineering and Innovative Technology)(IJEIT).
[6] Survey on Load Rebalancing for Distributed File System in Cloud(International Journal of Innovative Research in Advanced Engineering (IJIRAE) Volume 1 Issue 2 (April 2014)).
[7] Load rebalancing for Distributed file systems in clouds (IEEE Transaction on parallel and distributed systems.vol.24.no.5,year2013).
[8] A. Bharambe, M. Agrawal, and S. Seshan, "Mercury: Supporting Scalable Multi-Attribute Range Queries," Proc. ACM SIGCOMM '04, pp. 353-366, Aug. 2004