ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

A Load Balancing Model Using a New Algorithm in Cloud Computing For Distributed File with Hybrid Security

Dhafar Hamed Abd
Department of Computer Science,
Al Maaref University College.

Abstract:

A fully distributed load balancing algorithmic rule is given to deal with the load imbalance drawback. Our algorithmic rule is compared in contradiction of a centralized approach throughout a creation system and a competitive distributed resolution given within the literature. The simulation results indicate that our proposal is comparable the present centralized approach and significantly outperforms the previous distributed algorithmic rule in terms of load imbalance issue, movement value, and recursive overhead. The performance of our proposal enforced within the Hadoop distributed classification system is more investigated during a cluster setting. Moreover, period applications have Quality-of-Service (QoS) needs (e.g. bandwidth). The target of the routing protocol is to make a tree that's each possible (i.e. satisfies the requested QoS) and least expensive.

The value of a tree depends on the prices of its links. The value of a link ought to replicate the impact of allocating resources to the new affiliation on existing and future connections. The nodes area unit shared and several studies have projected inexpensive QoS-constrained trees. This research has employed a security side, which can be trustworthy to save data with some restricted rule from suspicious people. RSA and AES Algorithm is consider a multi-factor authentication solution that verity confirmation requests and centrally administers confirmation policies for enterprise networks. The major purpose of using Authentication Manager are to manage security tokens, users, multiple applications, agents, and resources across physical sites, and to help secure access to network and web-accessible applications.

However, these studies assume that some information processing address is shared in laptop, and that they don't examine the impact of the link value perform. We tend to conjointly investigate the impact of inaccurate network state data. Performance analysis is additionally maintained here through authentication, security maintained through permission key generation.

INTRODUCTION

Cloud Computing (or cloud for short) is a compelling technology. In clouds, clients can dynamically allocate their resources on-demand without sophisticated deployment and management of resources. Key enabling technologies for clouds include the Map Reduce programming, distributed file systems virtualization, and so forth. These techniques emphasize scalability, so clouds can be large in scale, and comprising entities can arbitrarily fail and join while maintaining system reliability.

Distributed file systems are key building blocks for cloud computing applications based on the Map Reduce programming paradigm. In such file systems, nodes simultaneously serve computing and storage

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

functions; a file is partitioned into a number of chunks allocated in distinct nodes so that Map Reduce tasks can be performed in parallel over the nodes.

There are different reasons that can be considered as causes of load-balancing problem in distributed file systems specialized for large-scale, dynamic and data-intensive clouds. The terms “rebalance” and “balance” is interchangeable in cloud. A novel load-balancing algorithm to deal with the load rebalancing problem in large-scale, dynamic, and distributed file systems in clouds. Distributed file systems are key building blocks for cloud computing applications 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 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 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.

it is necessary to take steps in a model development in order to deal with the temporal variability in the proposed system can easily implemented, as this is based on some vital programming languages such as C#.Net .The database created is with SqlServer which is more secure and easy to handle. The resources that are required to implement/install these are available. Therefore, the project is operationally feasible.

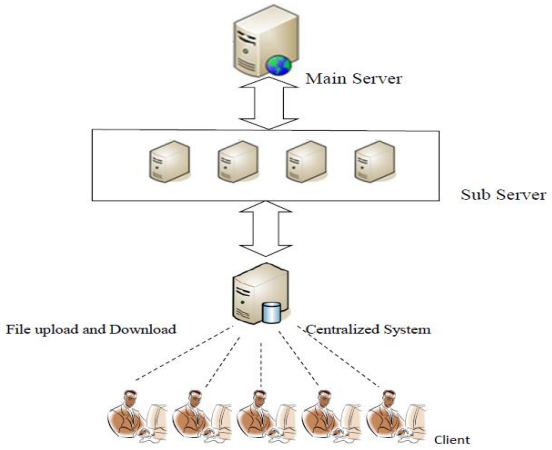

Figure 1.1 Distributed file using load balancing system

Economic analysis is the most frequently used method for evaluating the effectiveness of a new system. More commonly known cost/benefit analysis, the procedure is to determine the benefits and savings that are expected from a candidate system and compare them with costs. If benefits outweigh costs, then the decision is made to design and implement the system. An entrepreneur must accurately weigh the cost versus benefits before taking an action. This system is more economically feasible which assess the brain capacity with quick & online test. So it is economically a good project

LITERATURE SURVEY

This implementation load balance peer to peer protocol base internet application distributed IP address application interface system. Researches and studies in Load Balance have been extensive. Most often, the technology with which such control is implemented uses the underlying concepts and theories in many applications. Great improvements have been made with the This means what are the existing in this application find to the load balancing function used the problem of the object and what is function using system in load balance concept analysis. The researcher also observed that other researchers have used various systems’ parameters to model the load balance problems in order to derive the optimal solutions. Many of these researches involve problems that are associated with load balance system of large

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

and small areas with varying degree of success in finding the optimum results.

Virtualization

Virtualization is a framework or methodology of dividing the resources of a computer into multiple execution environments, by applying one or more concepts or technologies such as hardware and software partitioning, timesharing, partial or complete machine simulation, emulation, quality of service, and many others. We're used to a simple equation; one physical machine runs one OS at any given time. By virtualization the machine we are able to several operating systems

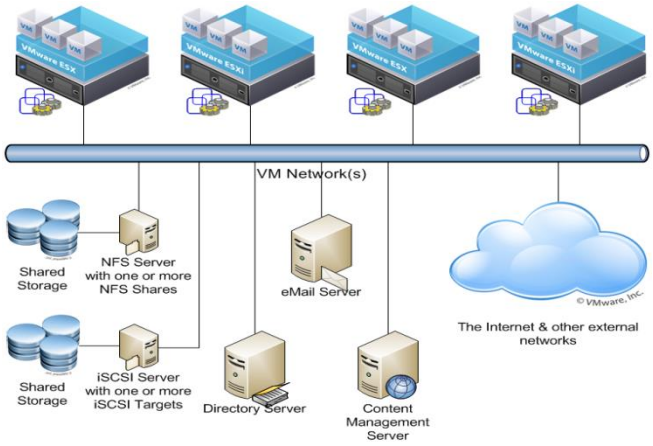

Figure 2.1 virtual machine

Load Balancing

Load balancing is essential for efficient operations in distributed environments. It means distributing the amount of work to do between different servers in order to get more work done in the same amount of time and serve clients faster. In this case, consider a large-scale distributed file system. The system contains N chunkservers in a cloud (N can be 1000, 10000, or more), where a certain number of files are stored. Each file is split into several parts or chunks of fixed size (for example 64 megabytes). The load of each chunkserver is proportional to the number of chunks hosted by the server. In a load-balanced cloud, the resources can be well used while maximizing the performance of MapReduce-based applications.

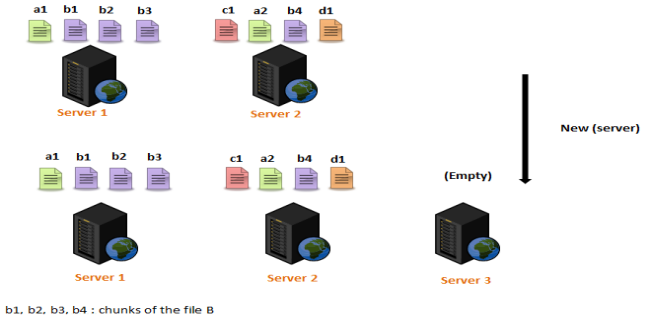

Figure 2.2 Load balancing design

Heterogeneity and Load Balance in Distributed Hash Tables

Existing solutions to balance load in DHTs incur a high overhead either in terms of routing state or in terms of load movement generated by nodes arriving or departing the system. In this paper, we propose a set of general techniques and use them to develop a protocol based on Chord, called Y0 that achieves load balancing with minimal overhead under the typical assumption that the load is uniformly distributed in the identifier space. In particular, we prove that Y0 can achieve near-optimal load balancing, while moving little load to maintain the balance and increasing the size of the routing tables by at most a constant factor.

Load Rebalancing

In a cloud computing environment, failure is the norm and chunkservers may be upgraded, replaced, and added in the system. Files can also be dynamically created, deleted, and appended. That leads to load imbalance in a distributed file system, meaning that the file chunks are not distributed equitably between the nodes.

Distributed file systems in clouds such as GFS and HDFS rely on central servers (master for GFS and Name Node for HDFS) to manage the metadata and the load balancing. The master rebalances replicas periodically: data must be moved from a Data Node/chunk server to another one if its free space is below a certain threshold However, this centralized approach can provoke a bottleneck for those servers as they become unable to manage a large number of file

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

accesses. Consequently, dealing with the load imbalance problem with the central nodes complicates more the situation as it increases their heavy loads. The load rebalance problem is NP-hard.

In order to manage large number of chunk servers to work in collaboration, and solve the problem of load balancing in distributed file systems, several approaches have been proposed such as reallocating file chunks such that the chunks can be distributed to the system as uniformly as possible while reducing the movement cost as much as possible.

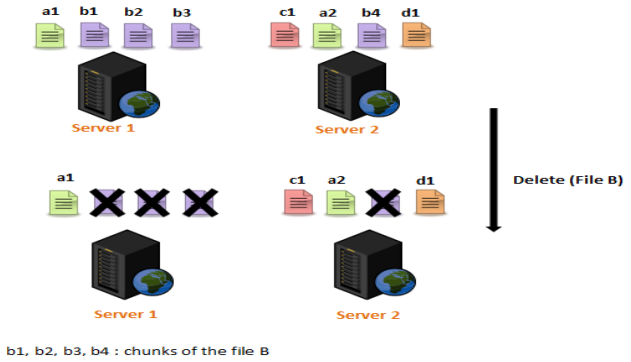

Figure 2.3 Load Rebalancing design

METHODS AND DESIGN

Overview of Distributed File Systems by using Load balancing

The overview provides a brief account of the overall project method including the workflow of activities, program development. The initial construction of the design is done on a project board for easier testing and modification. In developing the expected prototype for research purpose, it is important to choose the best components with consideration to the factors that to determine the suitability of the components. The factors are the environment in which this prototype works the reliability and cost effectiveness. The choice of incorrect components may cause problems to the development of the project. The problem may cause damage to the other parts and may escalate cost. Generally, this system is built according to requirements to make sure all the components function properly. However, if any problems arise, they can be troubleshoot and solved effectively.

In this project, the system of the block diagram is designed to identify the connection with all the components. It is most important to know the input and output components, in order to know whether the proposed system is working correctly or otherwise. Also, the flow chart of the system is designed to emphasize specifically on the flow of the system. Finally, to make the system complete ip address for each device has to be constructed before they can be connected together.

Research Method

The main purpose of using research method is to give a brief explanation and description of the components involved the study including the design flow for software, hardware, design architecture and detailed explanation of software development such as graphical user interface (GUI) as well as creating a web page. Figure 3.1shows the project flow to complete this project in three steps:

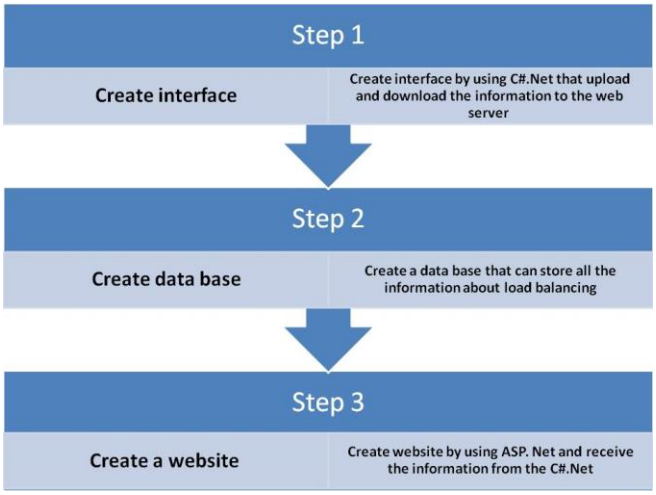

Figure 3.1 Research Method

Design Flow

Based on the necessary requirements to complete Load Balancing without any causing problem to the system, the design is divided into two important developments; Distributed file and security. In order to make sure that all the system components are working and meeting the objectives of the project, a few methods are employed to fulfill the requirements. The first method identifies the requirements and specifications for

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

distributed file. The second method involves the design and development of the security of prototype together with all the software for the interface between the client and servers.

Figure 3.2 shows the project design flow that included all the development of distributed file and security together with the methods to fulfill the requirement of the distributed file system using load balancing. The diagram also shows the steps in designing and programming the interface between Clint and server.

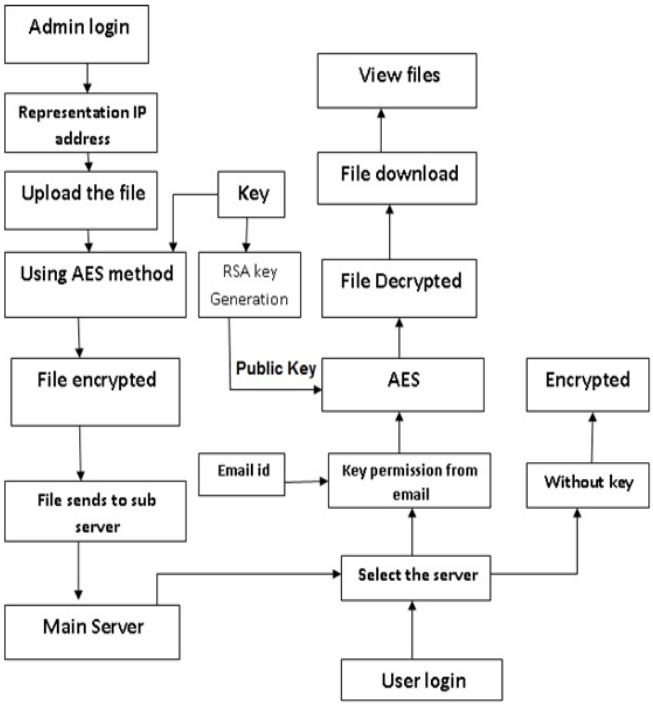

Figure 3.2 Design Flow

Basically, the first step after admin login to system for distributed file. Admin will upload a file after that encrypted by AES and RSA. The RSA use for encrypted key by privet key and send public key with encrypted key and that file when encrypted by AES . If constructed and the program for the software development is written correctly, file send to sub server after that send to main server for collection all files. After admin accept that registration send key permission to mail when but that key can see all files. If not have that key permission will not see anything because it was encrypted.

Load Balancing Architecture

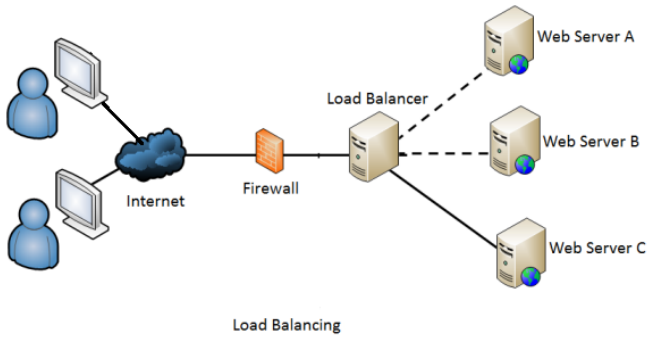

Figure 3.3 Load Balancing Architecture

Figure 3.3 shows the project architecture. It shows the overall process flow for the hardware and software development. In this architecture, there are two users connected together in the same period to the internet. Furthermore, the firewall has been located to keep the whole system between load balancing and user to be restricted about any suspicious data might destroyed the information that have stored in the main server. In addition to that, load balancing are considering one most significant has divided into three virtual servers, all the data will be distributed to three servers with specific amount of data for each server. Moreover, each server has special function. The experimental setup can be controlled and monitored from anywhere covered by internet service through exchanging data with the server. This is done using C#.Net language which manages the flow and direction of messages from one source to others destination.

Security concept

In this research, there are two algorithms have been taken place, which are symmetric and asymmetric. Firstly, we considerate about symmetric AES that consider very strong but has problem with exchange key. Secondly, we discover to find out the suitable method to solve this problem by using asymmetric. The main reason behind the view we did not involve RSA for encrypted files because great deal of information and spent plenty of time for encrypted file. Eventually, in this project we have chosen huge amount of files instead of small size.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

AES Concept

The Advanced Encryption Standard (AES) is a symmetric-key block cipher algorithm. It works with permutations and substitutions. Permutations are rearrangements of data, and substitutions replace one unit of data with another. AES performs permutations and substitutions using several different techniques.

Key size (words/bytes/bits)	4/16/128	6/24/192	8/32/256
Number of rounds	10	12	14
Expanded key size (words/byte)	44/176	52/208	60/240

Figure 3.4 AES Process

Phases of AES

The AES algorithm consists of following phases:

1. **Key Expansion.** Round keys are derived from the cipher key using the Rijndael's
2. **Initial Round.AddRoundKey**—each byte of the state is combined with the round key using a bit-wise operation.
3. **Middle Rounds.**Nr = 1 till Nr-1 Repeatedly perform the following transformations:
    1. **SubBytes**—a non-linear substitution step where each byte is replaced with another according to a lookup table.
    2. **ShiftRows**—a transposition step where each row of the state is shifted cyclically a certain number of steps.
    3. **MixColumns**—a mixing operation which operates on the columns of the state, combining the four bytes in each column.
    4. **AddRoundKey**—same as described above.
4. **Final Round (no MixColumns)**
    1. **SubBytes**—same as described above.
    2. **ShiftRows**—same as described above.
    3. **AddRoundKey**—same as described above.

Concept of RSA

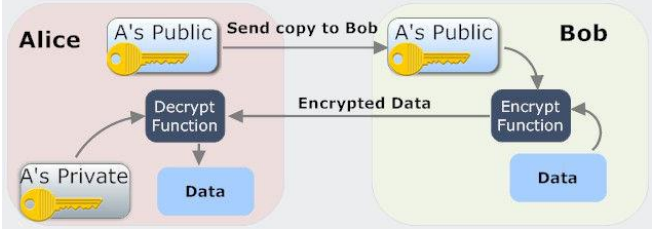

Figure 3.5 RSA Process

The public-key algorithms use two different keys to encrypt and decrypt the message. The keys are generated by a particular algorithm that doesn't allow obtaining one key from another. One of the keys, called public key, is shared and will be used for the encryption process. The other key, called private, must be kept secret and is used to decrypt the messages.

If person A wants to send a confidential message to the person B, for example, Alice starts by sending her public key (A's public) to Bob so Bob can use the public key to encrypt a message. Bob uses Alice's public key and encrypts a message. The encrypted message is sent to Alice. Since only Alice has the private key, only Alice can decrypt the encrypted message. An eavesdropper will not be able to decode the encrypted message.

Key generation of RSA

RSA involves a public key and a private key. The public key can be known by everyone and is used for encrypting messages. Messages encrypted with the public key can only be decrypted in a reasonable amount of time using the private key. The keys for the RSA algorithm are generated the following way:

1. Choose two distinct prime numbers p and q.

    For security purposes, the integer's p and q should be chosen at random, and should be of similar bit-length. Prime integers can be efficiently found using a primality test.
2. Compute $n = p*q$.

    n is used as the modulus for both the public and private keys. Its length, usually expressed in bits, is the key length.
3. Compute $\varphi(n) = \varphi(p)\varphi(q) = (p - 1)(q - 1)$
4. Choose an integer e such that $1 < e < \varphi(n)$ and $\gcd(e, \varphi(n)) = 1$

    e is released as the public key exponent.
5. Determine d as $d \equiv e^{-1} \pmod{\varphi(n)}$
    - This is more clearly stated as: solve for d given $d\square e \equiv 1 \pmod{\varphi(n)}$
    - This is often computed using the extended Euclidean algorithm. Using the pseudo code in the Modular integers section.
    - Kept as the private key exponent.
6. Public key = me mod n
7. Privet key = md mod n

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

The public key consists of the modulus n and the public (or encryption) exponent e. The private key consists of the modulus n and the private (or decryption) exponent d, which must be kept secret. p, q, and φ(n) must also be kept secret because they can be used to calculate d.

Why Use RSA

- Secrecy and privacy: the content of the information and communication must be ONLY accessible to the sender and the recipient of the information
- Integrity: the content must not be altered during the exchange phase, there for it must stay in its original form
- authentication: this aspect is very important because RSA guarantees the origin of the sent information, only the sender with his own private key is able to encrypt the message there for transform the message into an unreadable form consequently the receiver will have confirmation of the origin because he will be able to decrypt the message only through the corresponding public key
- non repudiation: the sender cannot state that the message has not been encrypted with his private key because the private key used for the encryption is unique and it's the owner's responsibility to make sure that it is not used by non authorized third parties
- Well established

Comparison between AES and RSA

The table below compares the important features of the AES and RSA algorithms, used within global cryptographic systems.

Feature	AES	RSA
algorithm	symmetric	asymmetric
Speed	high	low
deposit of keys	needed	needed
independence	no	no
Trojan Horse	not proved	no
data block length	128 bits	minimum 512 bits
key length	128 bits	minimum 512 bits
use of data space	full, 128 bits (2^{128}), 8 bytes	variable, limited, not defined,
ciphering & deciphering key	same	different
ciphering & deciphering algorithm	different	same
algorithm contains only XOR and branching	no	no
average number of key for one pair E&C=1	probably not	probably yes
cryptanalysis method	differential method	product factorization
global system including algorithm	not suitable	not suitable
Time for encrypt 153 KB	1.6 Sec	7.3 Sec
Time for decrypt 153 KB	1.1Sec	4.9 Sec

Hybrid Algorithms AES and RSA

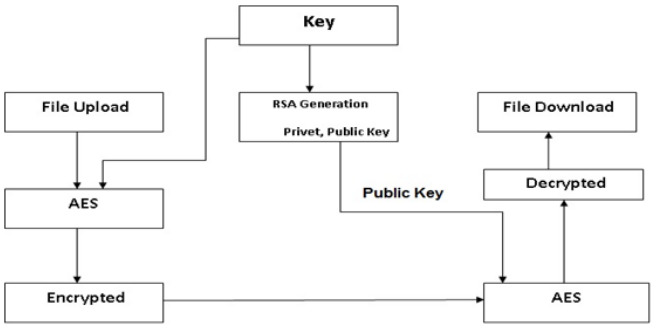

Figure 3.6 Hybrid AES and RSA

Usually asymmetric key systems ensure a good security level but are slower and computationally more demanding than symmetric key encryption. Hybrid Algorithms system uses a symmetric and an asymmetric public key system by combining the advantages of two systems. The safety of public key and the speed of the symmetric key.

Specifically the hybrid system uses a public key algorithm. We used RSA, in order to safely share the symmetric encryption system's secret key. The real

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

message is then encrypted using that key .We used AES because very fast and then sent to the recipient. Since the key sharing method is secure, the symmetric key used for the encryption changes for each message sent. For this reason it is sometimes called the session key. This means that if the session key was intercepted, the interceptor would only be able to read the message encrypted with that key. In order to decrypt other messages the interceptor would have to intercept other session keys.

The session key, encrypted using the public key algorithm, and the message being sent, encrypted with the symmetric algorithm, are automatically combined into a single package. The recipient uses private key to decrypt the session key and then uses the session key to decrypt the message. In this research, we have used some significant key as mentioned below.
1-AES with 128 bit key encrypted message
2-RSA with 1024 bit key encrypted key from AES

Load Balancing Concept

In our proposed algorithm, each chunk 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 chunks it hosts is smaller than the threshold.
Load statuses of a sample of randomly selected nodes. Specifically, each node contacts a number of randomly selected nodes in the system and builds a vector denoted by V. A vector consists of entries, and each entry contains the ID, network address and load status of a randomly selected node.

The Algorithm

Load balancing algorithms help easily to fine-tune how traffic is distributed across connections. Each deployment has a unique setup, and Peplink's enterprise grade load balancing features can fulfil all of your special requirements. Create your own rule with the following algorithms and you can sit back and enjoy the high performance routing that Peplink brings to you.

Implementation

Modules Details

1. Authentication Module.
2. IP Address Representation Module.
3. Load Servers Module.
4. Load Balancing Module.
5. Load Rebalancing Module.
6. Report Module

Authentication Module:

The authentication module is to register the new users and previously registered users can enter into our project. The admin only can enter and do the uploading files into the servers. After login by every user and the admin the sql server checks the login id and password is valid or not. If the login is not valid it displays that the login is not correct.

IP Address Representation Module:

The IP Address Representation module is to give the IP addresses which we are going to assign those as servers. We can enter and view IP addresses from this module. In load balancing system we can connect the three servers [system]. The connection has to be represented by the IP Address representation only.

Load Servers Module:

The Load Servers module has the authentication for the administrator only can enter into this module. The administrator will do the encryption of the text file and store into the servers which we are assigned in IP representation module. This module will make the both public and private key for the cryptography.

Load Balancing Module:

The Load Balancing module has the authentication for users can enter into the upload page and can view the file name which the administrator stored into the servers. The user can select the file from the list and can download from the server which is in idle state. We will get the response time and from which server we are getting the file. Finally we can get the decrypted file from the key pair.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Load Rebalancing Module:

The Load Rebalancing module has the authentication for users can enter into the upload page and can view the file name which the administrator stored into the servers. The user can select the file from the list and can download from the sub-server which is in idle state. We will get the response time and from which server we are getting the file. Finally we can get the decrypted file from the key pair.

Report Module:

We will get the response time and from which server we are getting the file. From the response time produce the chart report here. It compares the response time between the servers and downloads the given file in the better performance response time server.

Data Model

A context-level DFD for the system the primary external entities produce information for use by the system and consume information generated by the system. The labeled arrow represents Data flow diagram.

The DFD takes an input-process-output view of a system i.e. data objects flow into the software, are transformed by processing elements, and resultant data objects flow out of the software. Data objects represented by labeled arrows and transformation are represented by circles also called as bubbles. DFD is presented in a hierarchical fashion i.e. the first data flow model represents the system as a whole. Subsequent DFD refine the context diagram (Stage 0 DFD), providing increasing details with each subsequent level. The DFD enables the software engineer to develop models of the information domain & functional domain at the same time. As the DFD is refined into greater levels of details, the analyst performs an implicit functional decomposition of the system. At the same time, the DFD refinement results in a corresponding refinement of the data as it moves through the process that embody data objects or object hierarchy.

CONCLUSION

The main objectives of this project have successfully achieved.

We are interested in studying the load rebalancing problem in distributed file systems specialized for large-scale, dynamic and data-intensive clouds. Distributed file systems are key building blocks for cloud computing applications based on the Map Reduce 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 be performed in parallel over the nodes.
Finally, the devices itself bring advantages such as:
1-simplicity
2- Easy to use and to troubleshoot
3-Low cost
4-Immediate and fast detection of an impending disaster
5-Appropriate technology use.

It is hopefully that this project will be success in the future and can give a lot of benefits to people. It is also hope that with this project.

Based on the literature review, testing and results, this project is deemed a success in load balancing and relies of information. The system of this project can be progressed further as the hands-on skills can be improved in applying high technology devices to obtain new knowledge with regards to electronics components and communication technology.

Future Research

In future we have increase efficiency and effectiveness of our design are further validated by analytical models and a real implementation with a small-scale cluster environment highly desirable to improve the network efficiency by reducing each user's download time. In contrast to the commonly-held practice focusing on the notion of average capacity, we have shown that both the spatial heterogeneity and the temporal correlations in the service capacity can

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

significantly increase the average download time of the users in the network, even when the average capacity of the network remains the same.

REFERENCES

1).I. 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), pp. 11-18, June 2011.

2).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

3).C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, and S. Lu, "BCube: A High Performance, Server-Centric Network Architecture for Modular Data Centers," Proc. ACM SIGCOMM '09, pp. 63-74, Aug. 2009.

4).H. Abu-Libdeh, P. Costa, A. Rowstron, G. O'Shea, and A. Donnelly, "Symbiotic Routing in Future Data Centers," Proc. ACM SIGCOMM '10, pp. 51-62, Aug. 2010.

5).M. Jelasity, S. Voulgaris, R. Guerraoui, A.-M. Kermarrec, and M.V.Steen, "Gossip-Based Peer Sampling," ACM Trans. Computer Systems, vol. 25, no. 3, Aug. 2007.

6)J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," Proc. Sixth Symp. Operating System Design and Implementation (OSDI '04), pp. 137-150, Dec. 2004. Hadoop Distributed File System "Rebalancing Blocks," http://developer.yahoo.com/hadoop/tutorial/module2.html#rebalancing,2012.

7)K. McKusick and S. Quinlan, "GFS: Evolution on Fast-Forward,"Comm. ACM, vol. 53, no. 3, pp. 42-49, Jan. 2010.

8)HDFS Federation, http://hadoop.apache.org/common/docs/ r0.23.0/hadoop-yarn/hadoop-yarn-site/Federation.html, 2012.

9) Ubuntu, http://www.ubuntu.com/, 2012.

10) I. Stoica, R. Morris, D. Liben-Nowell, D.R. Karger, M.F. Kaashoek,F. Dabek, and H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications," IEEE/ACM Trans.Networking, vol. 11, no. 1, pp. 17-21, Feb. 2003.

11) A. Rowstron and P. Druschel, "Pastry: Scalable, Distributed Object Location and Routing for Large-Scale Peer-to-Peer Systems," Proc. IFIP/ACM Int'l Conf. Distributed Systems Platforms Heidelberg, pp. 161-172, Nov. 2001.

12) G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A.Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W.ogels, "Dynamo: Amazon's Highly Available Key-Value Store,"Proc. 21st ACM Symp. Operating Systems Principles (SOSP '07), pp. 205-220, Oct. 2007

13)A. Rao, K. Lakshminarayanan, 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.

14) D. Karger and M. Ruhl, "Simple Efficient Load Balancing Algorithms for Peer-to-Peer Systems," Proc. 16th ACM Symp.Parallel Algorithms and Architectures (SPAA '04), pp. 36-43, June 2004.

15) P. Ganesan, M. Bawa, and H. Garcia-Molina, "Online Balancing of Range-Partitioned Data with Applications to Peer-to-Peer Systems," Proc. 13th Int'l Conf. Very Large Data Bases (VLDB '04), pp. 444-455, Sept. 2004.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

16) J.W. Byers, J. Considine, and M. Mitzenmacher, "Simple Load Balancing for Distributed Hash Tables," Proc. First Int'l Workshop Peer-to-Peer Systems (IPTPS '03), pp. 80-87, Feb. 2003.

17) G.S. Manku, "Balanced Binary Trees for ID Management and Load Balance in Distributed Hash Tables," Proc. 23rd ACM Symp. Principles Distributed Computing (PODC '04), pp. 197-205, July 2004

18)M. Armbrust et al., "Above the Clouds: A Berkeley View of Cloud Computing," technical report, Univ. of California, Berkeley, Feb. 2009.

19) L. Siegele, "Let It Rise: A Special Report on Corporate IT," The Economist, vol. 389, pp. 3-16, Oct. 2008.

20) P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the Art of Virtualization," Proc. ACM Symp. Operating Systems Principles (SOSP '03), Oct. 2003.

21) "Amazon elastic compute cloud (Amazon EC2)," http://aws.amazon.com/ec2/, 2012.

22)Y. Agarwal, S. Hodges, R. Chandra, J. Scott, P. Bahl, and R. Gupta, "Somniloquy: Augmenting Network Interfaces to Reduce Pc Energy Usage," Proc. USENIX Symp. Networked Systems Design and Implementation (NSDI '09), 2009.

23) D. Meisner, B.T. Gold, and T.F. Wenisch, "Powernap: Eliminating Server Idle Power," Proc. Int'l Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS '09), 2009

24) T. Das, P. Padala, V.N. Padmanabhan, R. Ramjee, and K.G. Shin, "Litegreen: Saving Energy in Networked Desktops Using Virtualization," Proc. USENIX Ann. Technical Conf., 2010.

25)Y. Agarwal, S. Savage, and R. Gupta, "Sleepserver: A Software- Only Approach for Reducing the Energy Consumption of PCS within Enterprise Environments," Proc. USENIX Ann. Technical Conf., 2010.

26)N. Bila, E.d. Lara, K. Joshi, H.A. Lagar-Cavilla, M. Hiltunen, and M. Satyanarayanan, "Jettison: Efficient Idle Desktop Consolidation with Partial VM Migration," Proc. ACM European Conf.Computer Systems (EuroSys '12), 2012.

27)Ciurana, E. Developing with Google App Engine. Apress, Berkely, CA, USA, 2009.

28) Downey, A. B. The structural cause of file size distributions. SIGMETRICS Perform. Eval. Rev. 29, 1 (2001), 328–329.