IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 2, Ver. III (Mar-Apr. 2014), PP 44-50
www.iosrjournals.org

Loadbalancing and Maintaining the QoS On Cloud Computing

S.Saranya ,K.Dinesh
PG Student, Assistant Professor, Dept of CSE
SVS College of Engineering, Coimbatore, Tamilnadu, India.

Abstract: *Loadbalancing is the key of cloudcomputing environment.Loadbalancing main function is solving the unbalance or overload problem.Now a days many algorithm using for solve this problem. In this project we introduce load balancer concept.The cloud sever transfer all load to the load balancer.Load balancer concerned with the number of strong nodes and weak nodes.Strong nodes represent the normal load and weak nodes represent the heavy load or overload.Strong nodes and weak nodes denoted green and yellow colour respectively.Loadbalancer split the load among these node and also intimate the normal and overload node.The yellow colour or overload node loads are charged by neighbour node in cyclic manner using this algorithm and also make quick recovery of overload node.Loadbalncer maintain the index for server loading process and create database backup on server when any malicious attack the server.Main objective of this paper is solve the overload problem and make efficient searching process on cloud environment.*
Keywords: *Loadbalance, clouds*

I. Introduction

Cloud computing is a emerging technology. The software and databases are move to the centralized large data centres on cloud environment. Cloud computing environment concerned with the cloud clients and cloud server The cloud server mainly consist three types of layers.The first one is Software as a Service(SaaS) like Email,games,applications.This SaaS layer is the software commercial software access.The second one is Platform as a Service(PaaS).PaaS layer denoted the platform information,connectivity and integration service.It also represent the database,runtime and development tools.Third one is Infrastructure as a Service(IaaS).The IaaS layer provide information about the infrastructure of servers,loadbalancers,network devices and storage disks.The IaaS layer load balancer is the main concept of this paper.

Loadbalancer is the key for building nodes on cloud server.This nodes get the load from the load balancer.These nodes are created,deleted and appended dynamically in [7].Loadbalance perform the critical function among these nodes.Load-balanced cloud maximizing the performance of cloud sever and making the efficient searching process.Each nodes intimate the loads are underload or overload. Large number of clients want to access the data leads the central node become a performance bottleneck.The central nodes tackle the load imbalance problem exhibit their heavy loads.

The loadbalancing problem in cloud server specialized for large-scale and dynamic.Our objective is allocate the loads to the corresponding nodes and recover the overload node.Additionally,we aim to the efficient searching and maintaining the index for cloud server using loadbalancer.This capability improve the system performance.The nodes are structured as a network based on distributed hash tables.Nodes are enable to self organize and repair while constantly offering lookup functionality using distributed hash tables.

Load balancing algorithm [1] for distributing load to the nodes as uniformly as possible and minimizing the movement cost as much as possible.Our proposed algorithm operates in a distributed manner in which nodes perform their loadbalancing tasks independently without global knowledge regarding the loads.

The load balancer prevent the database from malicious attack.It will create immediate backup on server while some malicious function attack the loadbalancer.This database consists of current loading process of server and indexing function.Indexing function maintain which nodes carrying which data sets and also this function very useful to the searching process.

II. Loadbalancing Problem

We consider a large-scale load balancer consisting of a set of nodes V in a cloud,where the cardinality of V is $|V| = n$.The nodes are grouped by n number of groups.In the system,a number of data sets are allocated in the n chunkservers.F denote the set of data sets.Each data set $f \in F$ is partitioned into a number of disjointed.The load of a node is proportional to the number of data sets by the server.The nodes are added,replaces and deleted dynamically. Fig. 1 illustrates an example of the loadbalancing problem with the assumption that the nodes are homogeneous and have the same capacity.

Our objective in the current study is to design a loadbalancing algorithm to reallocate datasets.The data sets can be distributed to the nodes uniformly as possible while reducing the movement cost as much as

possible.The loadbalancing algorithm aims to recover the overload node.Note that nodes and loadbalancer are interchangeable in this paper.First assume a homogeneous environment,where migrating a data set between any two nodes takes a unit movement cost.Each node has the identical storage capacity.Node capacity and movement cost based on the data set migration in physical network locality [2].The node location useful to the indexing function and indexing useful to the efficient monitoring. The loadbalancing for distributed file systems in clouds

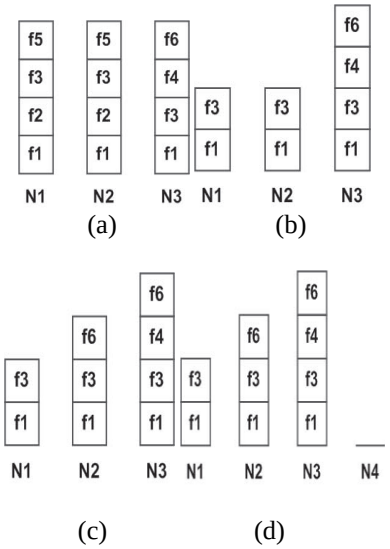

FIGURE 1:Loadbalancing problem,where (a) an initial distribution of data sets of f1,f2,f3,f4,f5 and f6 in three nodes N1,N2, and N3, (b)data sets f2 and f5 are deleted, (c) f6 is appended, and (d) node N4 joins.The nodes (b), (c), and (d) are in a load-imbalanced state.

III. Architecture

The loadbalancer placed between cloud server and number of nodes.Each nodes implements a distributed hash table such as chord or pastry [3]. Fig. 2 illustrates the experimental environment.Cloud server is the initial phase to the architecture and sever receive the request from clients. The name nodes represent the indexing function.

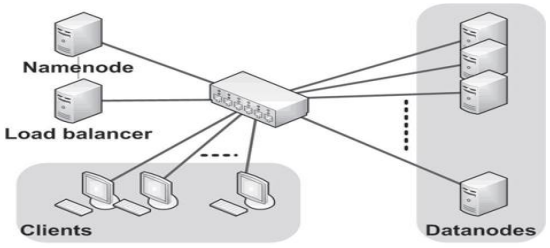

FIGURE3: Loadbalancing Experimental Enviornment

Second phase is loadbalancer.It receives the load from cloud server and also distribute the loads among the number of nodes.The third base is set of nodes.It consists two types of nodes. They are strong nodes and weak nodes.These nodes are differentiated by using colours.Fig. 3 illustrates the load balancing diagram.we are using two kind of colours in this architecture.The green colour denotes the strong nodes and yellow colour denotes the weak nodes or overload node.

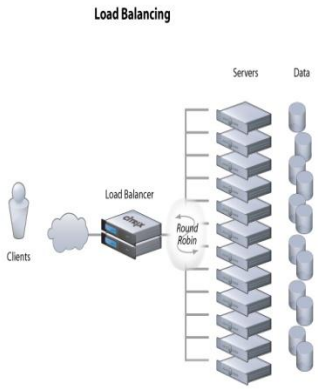

FIGURE 3: Loadbalancing Diagram

The weak nodes overload data transfer to the neighbour node in cyclic manner using load balancing algorithm.This algorithm also performs the searching process.Processes are proceed by round robin manner.

IV. Loadbalancing Algorithm

A large-scale load balancer hosts more number of nodes.In our proposed algorithm,each node estimates whether it is underloaded (normal) or overloaded (heavy) without global knowledge [4].A node either normal or heavy decided by this node characteristics as well as node colours.A loadbalancing algorithm by mapping the strong and weak nodes in the system has properties as follows:

1. Low movement cost
2. Fast convergence rate

The low movement cost denotes the overload nodes are processed by cyclic manner so the movement cost is very low and performance is high.Fast convergence rate represent the seeking of normal node is very easy and quick automatic recovery of overload node [8].These properties increase the lifetime of cloud server.Loadbalancing algorithm perform the continuous monitoring of the system and prevent the system from malicious attack.It will create immediate data backup on the server.

The mapping between the strong and weak nodes at each time sequence can be further improved to reach global system state [11].strong nodes concurrently request load from weak nodes and this significantly reduces the latency of sequential algorithm.We have introduced our algorithm node has global knowledge of the loads.It supports the large-scale environment [6].Our proposal by taking advantage of save the system from any kind of attacks as well as make the backup on server.

V. Basic Algorithms

Algorithm 1 specifies the operation that a strong node seeks an overloaded node and Algorithm 2 shows request some data sets from weak node.Each node represented in dataview function.Dataview function consists of enteries, and each entry contains the ID. Fig. 4 depicts example of our proposed algorithm.Nodes N1,N2,N3,N4and N5 are strong,and nodes N6,N7,N8,N9 and N10 are weak.The load balancing algorithm independently perform to the all nodes. Distribution of nodes based on this algorithm formulation. The nodes **N** are allocated with loads **L** in this formula. This formulation expressed by

$$A_{N1=} \frac{L_{N1}+L_{N2}+L_{N3}+L_{N4}+....+L_{Nk}}{5}$$

The basic algorithm exploits the network loacality,managing the failure nodes and taking the advantage of node heterogeneity.

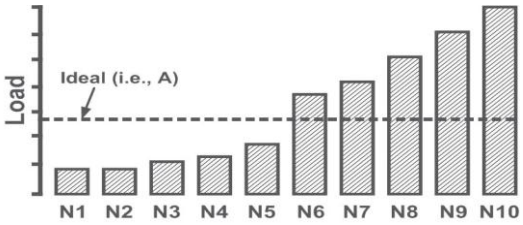

FIGURE4: Loads of chunkservers N1,N2,.....N10

VI. Exploiting physical network locality

We improve our proposal by defining the network locality. Each nodes are having separate IDs [5] and enteries.Fig. 5 illustrates the dataview of nodes.It denotes the total number of datasets.This is the initial phase of this paper is to collect set of dataset processing. The dataset or process will be initialized by the server.The data set might be in any order in [2], changing the data set will not affect the result. Nodes participating in the file system using their IDs and entries are possibly heterogeneous in terms of the numbers of data sets that the nodes can accommodate. Bottleneck resource for optimization although a node's capacity in practice should be a function of computational power, network bandwidth, and storage space. The load of a node is typically proportional to the number of data sets of the system. The locality improves the searching function.

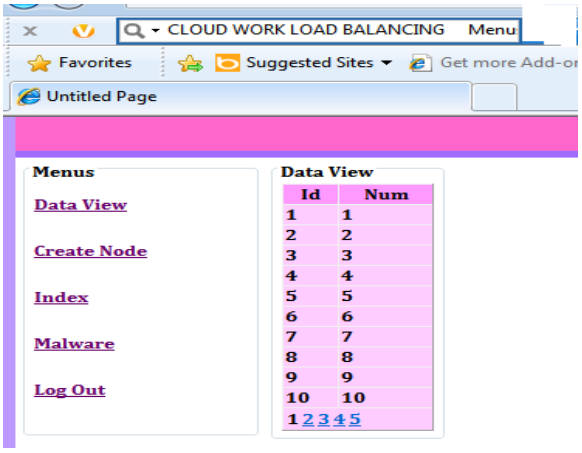

FIGURE 5: Dataview diagram

In this phase concerned with create nodes and clear nodes functions.Create node function using for node creation. It supports maximum eight number of nodes.Clear node function used to clear the node for previous dataset loading process. Fig. 6 illustrates the node creation.It supports maximum eight number of nodes.

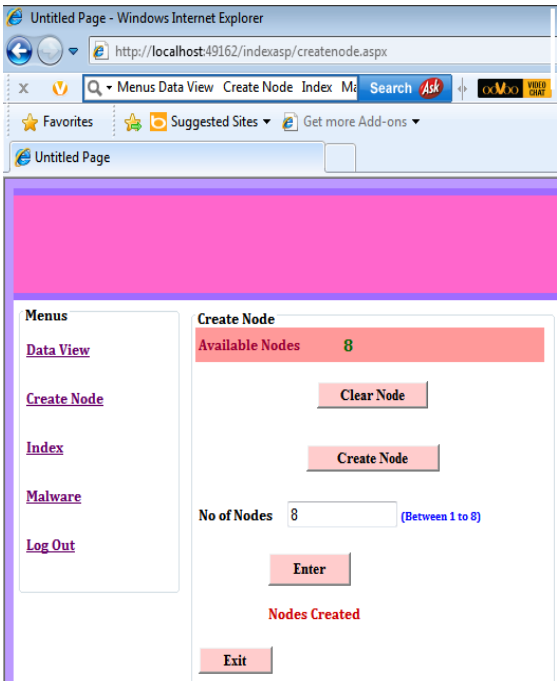

FIGURE 6: Node creation diagram

VII. Simulations

The performance of our algorithm is evaluated through computer simulations.These distributions indicate that a small number of nodes initially possess a large number of datasets. Fig . 7 depict indexing of loads.The loads are indexed by this simulation evaluation.

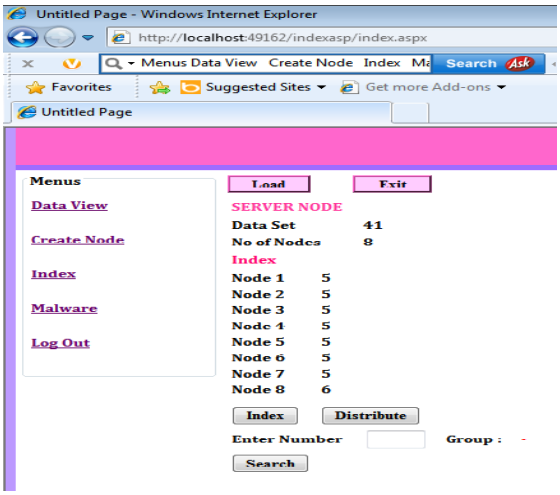

FIGURE 7: Loading index nodes

According to this concept there are two types nodes are available, they are strong node and weak node. This denotes the data carrying capacity of this node. Weak nodes will transfer their data to their neighbour stronger nodes in case of bottle neck problems.Fig. 8 illustrates the overlodingnode.The below figure demonstrate Node2,Node5 and Node8 are overload node or weak node.Load balancing efficiency in the MANET architecture.

There are exactly LE large jobs which are not assigned to any processor. An arbitrary subset of the small jobs are not assigned to any processor.There are exactly LT – LE processors which have exactly one large job; in these processors, the small jobs must have total size not exceeding ½ OPT .The remaining processors do not have any large jobs. Of these, at least LE processors have total load at most ½ OPT , and the remaining m − LT processors have total load at most OPT.

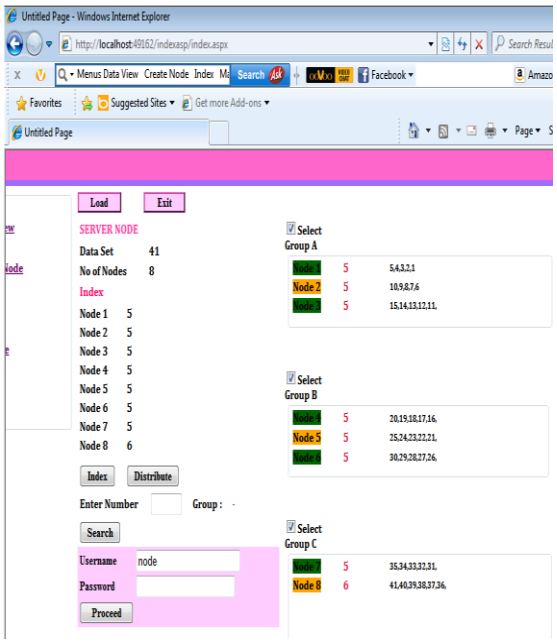

FIGURE 8:Overload nodes are 2,5,8

This simulation give the result of searching function.Type the data set ID then click search button.Fig. 9 illustrates the searching process

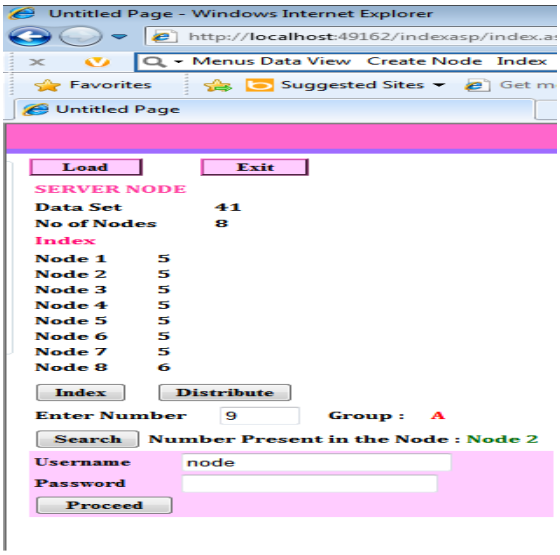

FIGURE 9:Searching process simulation

Load balancing algorithms prevent the system from different kind of malware attacks. Fig. 10 illustrate malware attacks. This simulation phase main process is creating immediate database backup while malicious code attack the system.

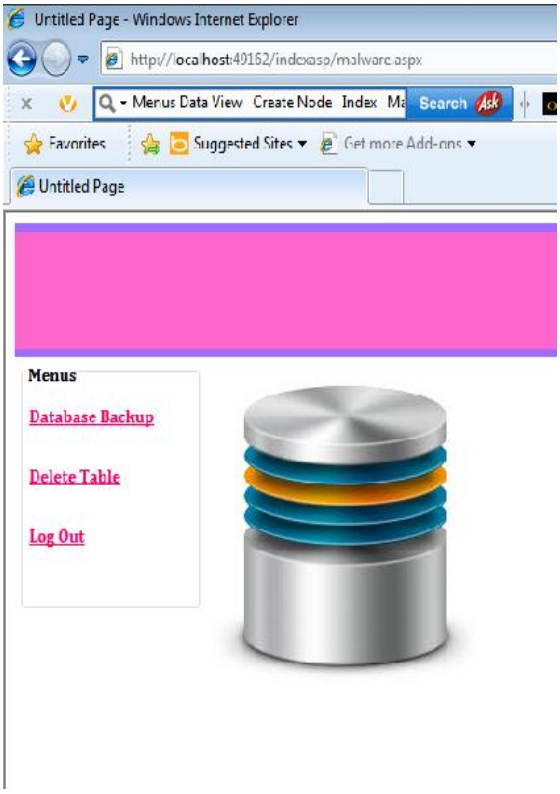

FIGURE 10: Malware attacks

VII. Simulation Results

According the simulation results the overload nodes are recovered and solve the imbalance problem of all nodes using loadbalncing algorithm

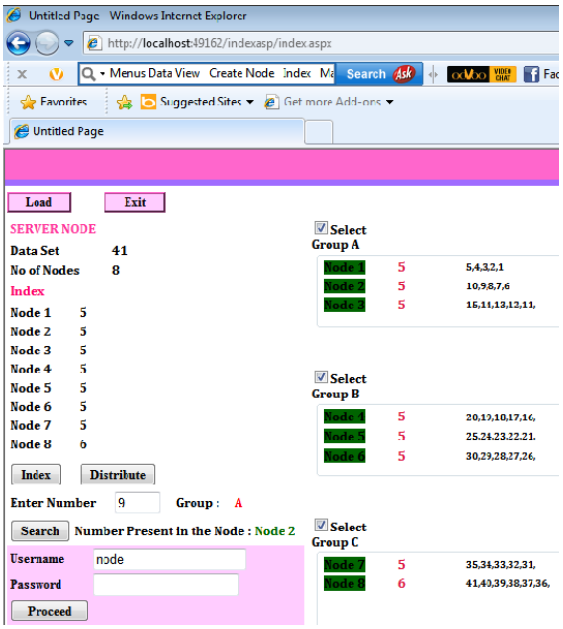

FIGURE: Efficient loadbalncing on cloud

VIII. Conclusion

This paper is presented to deal the load balancing problem in large-scale manner. The load balancer keeps the records of current undergoing tasks and process. So that no needs to check the nodes for the current process and also it makes simple searching process.The neighbour node will take cares the process, in case of overloading in cyclic approach.This process reduce the movement cost.Continuous node life monitoring will be done and prevent the system from malware attacks.The overload nodes are detected and recovered very quickly.Our loadbalancing algorithm exhibits the higher performance and improve the quality of cloud.

References

[1] Y. Zhu and Y. Hu, "Efficient, Proximity-Aware Load Balancing for DHT-Based P2P Systems," *IEEE Trans. Parallel and Distributed Systems, vol. 16, no. 4, pp. 349-361, Apr. 2005.*

[2] H. Shen and C.-Z.Xu, "Locality-Aware and Churn-Resilient Load Balancing Algorithms in Structured P2P Networks," *IEEE Trans. Parallel and Distributed Systems, vol. 18, no. 6, pp. 849-862, June 2007.*

[3] 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., vol. 21, no. 4, pp. 595-608, Apr. 2009.*

[4] H.-C. Hsiao, H. Liao, S.-S.Chen, and K.-C. Huang, "Load Balance with Imperfect Information in Structured Peer-to-Peer Systems," *IEEE Trans. Parallel Distributed Systems, vol. 22, no. 4, pp. 634-649, Apr. 2011.*

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

[6] M. Jelasity, A. Montresor, and O. Babaoglu, "Gossip-Based Aggregation in Large Dynamic Networks," *ACM Trans. Computer Systems, vol. 23, no. 3, pp. 219-252, Aug. 2005.*

[7] M. Jelasity, S. Voulgaris, R. Guerraoui, A.-M.Kermarrec, and M.V. Steen, "Gossip-Based Peer Sampling," *ACM Trans. ComputerSystems, vol. 25, no. 3, Aug. 2007.*

[8] 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.*

[9] 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.*

[10] 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.*

[11] 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.*