International Journal of Grid and Distributed Computing
Vol.7, No.1 (2014), pp.89-98
http://dx.doi.org/10.14257/ijgdc.2014.7.1.09

A Low Cost Two-Tier Architecture Model for High Availability Clusters Application Load Balancing

A. B. M. Moniruzzaman[1] and Syed Akther Hossain[2]

[1]Member, IEEE, [2]Member, IEEE & ACM
[1,2]Department of Computer Science and Engineering
Daffodil International University
Dhaka, Bangladesh
abm.moniruzzaman.bd@ieee.org, aktarhossain@daffodilvarsity.edu.bd

Abstract

This article proposes a design and implementation of a low cost two-tier architecture model for high availability cluster combined with load-balancing and shared storage technology to achieve desired scale of three-tier architecture for application load balancing e.g. web servers. The research work proposes a design that physically omits Network File System (NFS) server nodes and implements NFS server functionalities within the cluster nodes, through Red Hat Cluster Suite (RHCS) with High Availability (HA) proxy load balancing technologies. In order to achieve a low-cost implementation in terms of investment in hardware and computing solutions, the proposed architecture will be beneficial. This system intends to provide steady service despite any system components fails due to uncertainly such as network system, storage and applications.

Keywords: *Load balancing, high availability cluster, web server clusters*

1. Introduction

High-availability clusters provide continuous availability of services by eliminating single points of failure. Node failures in a high-availability cluster are not visible from clients outside the cluster. (High-availability clusters are sometimes referred to as failover clusters.) Red Hat Cluster Suite [1] provides high-availability clustering through its High-availability Service Management component.

Load-balancing clusters dispatch network service requests to multiple cluster nodes to balance the request load among the cluster nodes. Load balancing provides cost-effective scalability because you can match the number of nodes according to load requirements. If a node in a load-balancing cluster becomes inoperative, the load-balancing software detects the failure and redirects requests to other cluster nodes. Node failures in a load-balancing cluster are not visible from clients outside the cluster. Red Hat Cluster Suite provides load-balancing through LVS (Linux Virtual Server) [2].

This article focuses on how to build a two-tier architecture model combined with load-balancing technology and shared storage technology to achieve full facilities of the three-tier architecture for application load balancing *e.g.*, web servers Cluster. This system can overcome in the cases of node failover, network failover, storage limitation and distributions load as like as the all the facilities of three-tier architecture model for high availability cluster for application load balancing. Web Sever clusters have gained much attention and have become increasingly popular for handling requests because of their unique design [3]. When

ISSN: 2005-4262 IJGDC

handling large amounts of complex data, load-balancing is a crucial necessity [3].We use for application load balancing for web server cluster as a prototype for implementing this system.

In this research work, we propose a design and implementation of a low cost two-tier architecture model for high availability cluster combined with load-balancing and shared storage technology to achieve desired scale of three-tier architecture for application load balancing e.g. web servers. The paper is organized as follows: Section II discusses Tradition three-tier architecture model for application load balancing; Section III explores literature review of related works and motivation of this article; Section IV describes proposed design of model and implementation details of a low cost two-tier architecture model for high availability cluster combined with load-balancing and shared storage technology; Section V demonstrates performance test of model implementation; and finally Section VI summarizes the paper work.

2. Three-tier Architecture Model for Application Load Balancing

The three-tier architecture consists of Load Balancer, which is the front-end machine of the whole cluster systems, and balances requests from clients among a set of servers, so that the clients consider that all the services is from a single IP address. Server Cluster, which is a set of servers running actual network services, such as Web, Mail, FTP, DNS and Media service. Shared Storage, which provides a shared storage space for the servers, so that it is easy for the servers to have the same contents and provide the same services [4].

For scalability and availability of the system, usually three-tier architecture is adopted in LVS (Linux Virtual Server) clusters illustrated in the following Figure 2.1.

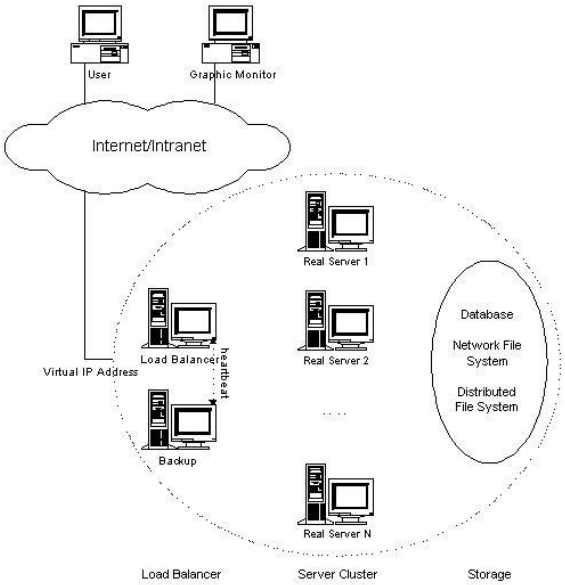

Figure 2.1. Three Tier Architecture of LVS Clusters source: www.linuxvirtualserver.org/architecture.html

Load balancer is the single entry-point of server cluster systems, it can run IPVS (IP Virtual Server) [5] that implements IP load balancing techniques inside the Linux kernel, or KTCPVS stands for Kernel TCP Virtual Server [6]. It implements

application-level load balancing inside the Linux kernel, so called Layer-7 switching that implements application-level load balancing inside the Linux kernel [4]. The node number of server cluster can be changed according to the load that system receives. When all the servers are overloaded, more new servers can be added to handle increasing workload. For most Internet services such as web, the requests are usually not highly related, and can be run parallel on different servers. Therefore, as the node number of server cluster increases, the performance of the whole can almost be scaled up linearly [4].

Shared storage can be database systems, network file systems, or distributed file systems. The data that server nodes need to update dynamically should be stored in data based systems, when server nodes read or write data in database systems parallel, database systems can guarantee the consistency of concurrent data access. The static data is usually kept in network file systems such as NFS and CIFS, so that data can be shared by all the server nodes.

3. Related Works and Motivation

Many researchers [12-18] worked on their Research on Load Balancing of Web-server cluster System in the three tier architecture model. Z. Han and Q. Pan [7] focuses on how to build an LVS load-balancing cluster technology combined with virtualization and shared storage technology to achieve the three-tier architecture of Web server clusters. Y Jiao, W Wang [8] designs and implements load- balancing system of distributed system based web-servers, their design cut costs by reducing time; but does not cut cost by reducing hardware resources, as this system also in the three tier architecture model. C Zheng, J Xia, Q Wang, X Chu [9] design and implement a model to adopt a web-server cluster systems with load balancing algorithms with multiple parameters. Their paper proposes the implementation of real time monitoring status of tasks and dynamic dispatch strategy in web-sever cluster systems [9]. A Krioukov, P Mohan, S Alspaugh and L Keys [10] design a system architecture for web service applications in a standard three-tier architecture model and implement as a power-proportional web cluster. Jiang, Hongbo, *et al.*, [11] Present Design, Implementation, and Performance of A Load Balancer for distributing Session Initiation Protocol Server Clusters, and several load balancing algorithms for distributing Session Initiation Protocol (SIP) requests to a cluster of SIP servers in the three tier architecture. Many researchers worked on their Research on Load-Balancing Algorithm for Web Server Clusters [19-23], these algorithms are being well implemented in different projects in the three tier architecture model.

4. Proposed Model Design and Implementation Details

For this design model implementation, some open-source software are installed and configured as prototype and tested on the Center for Innovation and Technologies (CIT) Lab, Research center for Science and Technology at DIU. In this experiment, two set of models are conducted to implement high availability Clusters for application load balancing - (1) Web Server Cluster with load balance infrastructure for web servers in three-tier architecture model; (2) Web Server Cluster with load balance infrastructure for web servers in two-tier architecture model.

4.1. Three-tier Architecture Model

Requirements for hardware resource in the three-tier architectural model of high availability cluster combined with load-balancing technology and shared storage technology for any application load balancing; these four types of nodes are needed – (1) Cluster nodes, (2) Load balancer nodes, (3) SAN Box, and (4) Network File System (NFS) Server Nodes. The typical design of these systems a "High Availability Cluster with Load Balance Infrastructure in three-tier architecture model" describe in the Figure 4.1.

In the Figure 4.1 real servers and the load balancers are interconnected with high-speed LAN (Private Network). The load balancers dispatch requests to the different servers and

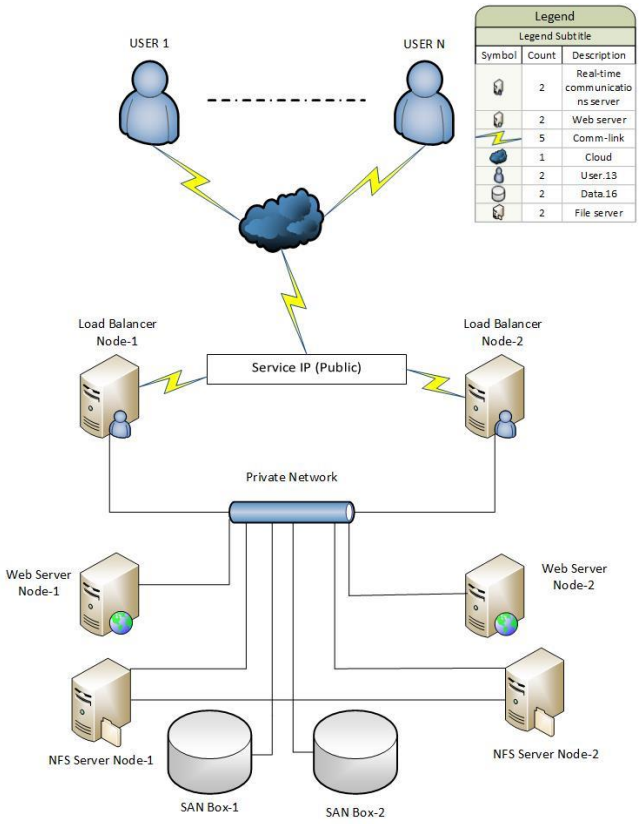

Figure 4.1. Web Server Cluster with Load Balance Infrastructure for Web Servers in Three-tier Architecture Model

make parallel services of the cluster to appear as a virtual service on a single IP address, and request dispatching can use IP load balancing technologies or application-level load balancing technologies. Scalability of the system is achieved by transparently adding or removing nodes in the cluster. High availability is provided by detecting node or daemon failures and reconfiguring the system appropriately. All requests will be processed by Round Robin Algorithm.

4.2. Proposed two-tier Architecture Model

In the two-tier architecture model these three types of nodes are needed – (1) Cluster nodes, (2) Load balancer nodes, and (3) SAN Box. We physically omit Network File System (NFS) Server Nodes and implements within the Load balancer node. The Figure 4.2 describes the cost effective proposed two-tier architecture model for High Availability Cluster with Load Balance Infrastructure for web servers.

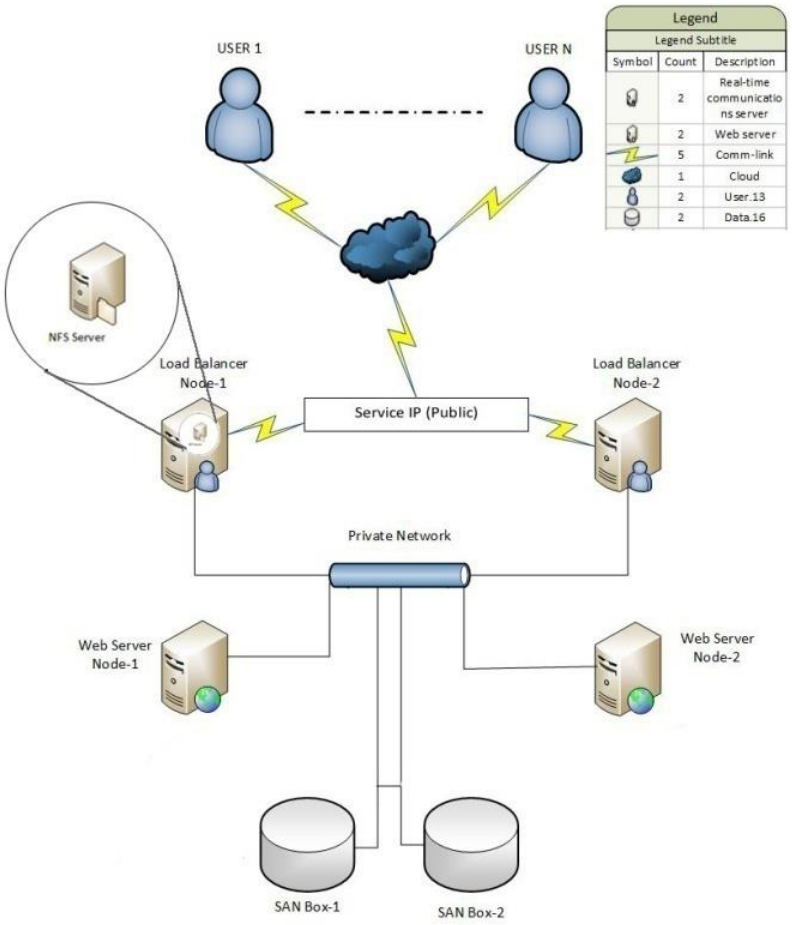

Figure 4.2. Web Server Cluster with Load Balance Infrastructure for Web Servers in Two-tier Architecture Model

In this design we used 2 nodes for web server Clusters, 2 nodes for Load balancer and two SAN. For any customized system the modified design can be used up-to 16 nodes for Web Server Clusters, any numbers (n) of nodes for Load balancer. For this system some open-source software are used for configuration and implementation; these are – Red Hat Enterprise Linux (RHEL) 6.4 for load balancer nodes and Cluster nodes; OpenFiler for SAN; Red Hat Cluster Suite (RHCS) for 2 cluster nodes; HAProxy for 2 load balancer nodes; Apache, Network File System (NFS) and PHP.

5. Performance Testing

To test High Availability Cluster and Load Balancing need some GUI tools and command line tools. For this testing PuTTy, Command Prompt (DOS), cURL, Web Browser are required. The Figure 5.1 shows Shared Storage for Load Balancing Node which service (NFS) is configured in Clusters.

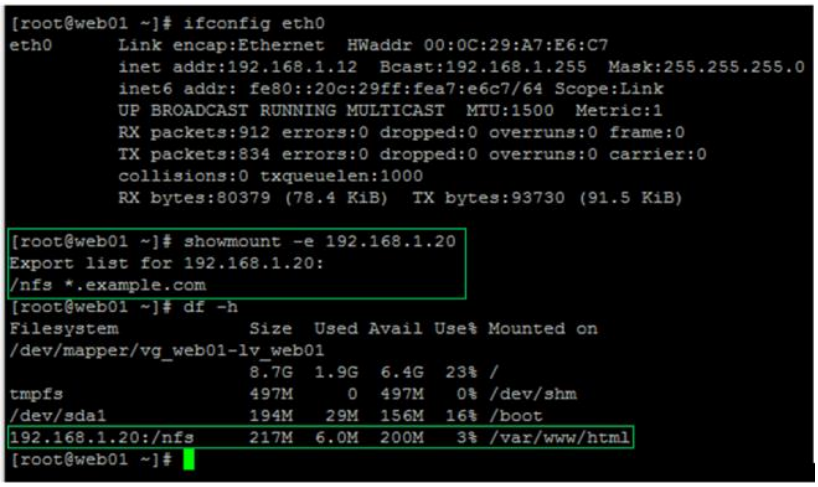

Figure 5.1. Status of Shared Storage for Load Balancing Node which service (NFS) is configured in Cluster

Now we test of High Availability Cluster with these status (in the Figure 5.2 and Figure 5.3) we can make sure High Availability Cluster is running status is ok. Here service name HAPC and owner hap01.

Figure 5.2 and Figure 5.3 both show the status of Cluster service. Currently all Application are running on HAP01.EXAMPLE.COM node and all Application are standby on HAP02.EXAMPLE.COM node. If incase any kind of failure like Network, Hard Disk, Application then all Application will move to the Standby node within very short time. Without any kind of notification to the client the system will overcome failover.

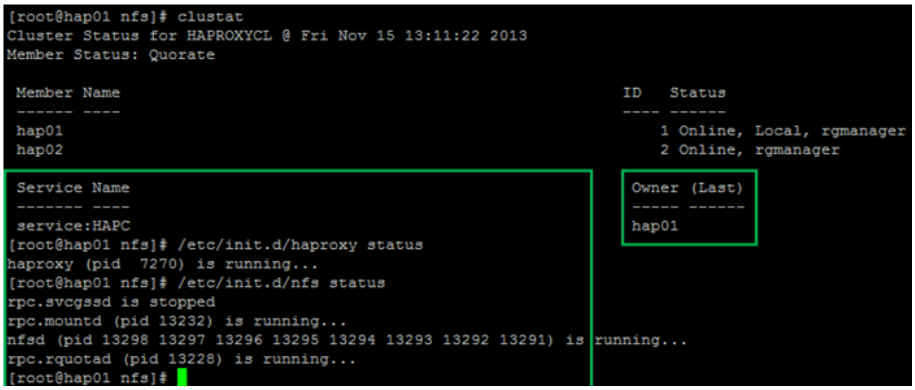

Figure 5.2. HAP01.EXAMPLE.COM Node Status

5.1. Testing of Load Balancing using GUI

Open any browser and type cluster node Virtual IP in this example here IP is: 192.168.1.20. In production environment this IP should be Public IP.

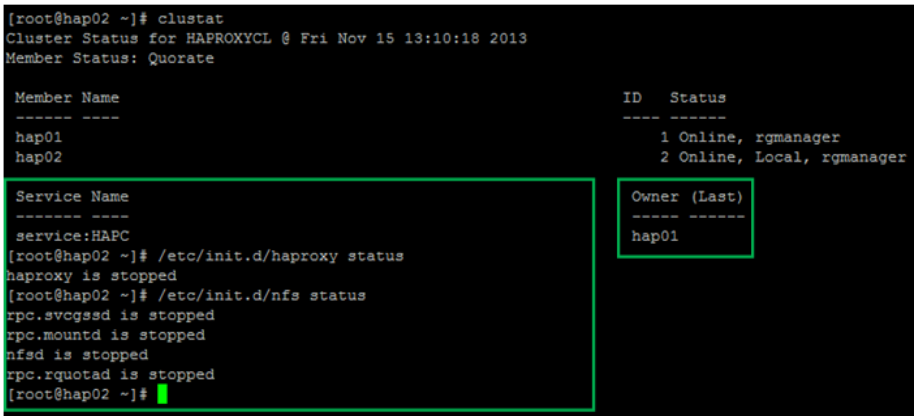

Figure 5.3. HAP02.EXAMPLE.COM Node Status

In the Figure 5.4 shows, node is changed for each request for server with same Virtual IP 192.168.1.20. The changing for node is hidden from the users.

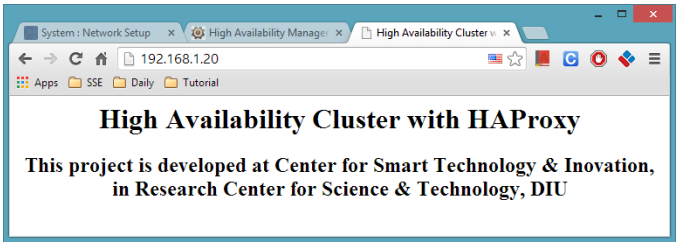

Figure 5.4. Requests serve by Cluster Application Server

5.2. Testing of Load Balancing using cURL

First request for server is served by node02 then subsequent by node01 then after node02 and go on serving request in this way (in the Figure 5.5). If system has 16 numbers of nodes, then requests are served subsequently 01 number to 16 number nodes then again 01 node. Every request handled by new server (in the Figure 5.5). This system managed by an algorithm which is called Round-Robin algorithm or Round-Robin scheduling which is handled request in parallel.

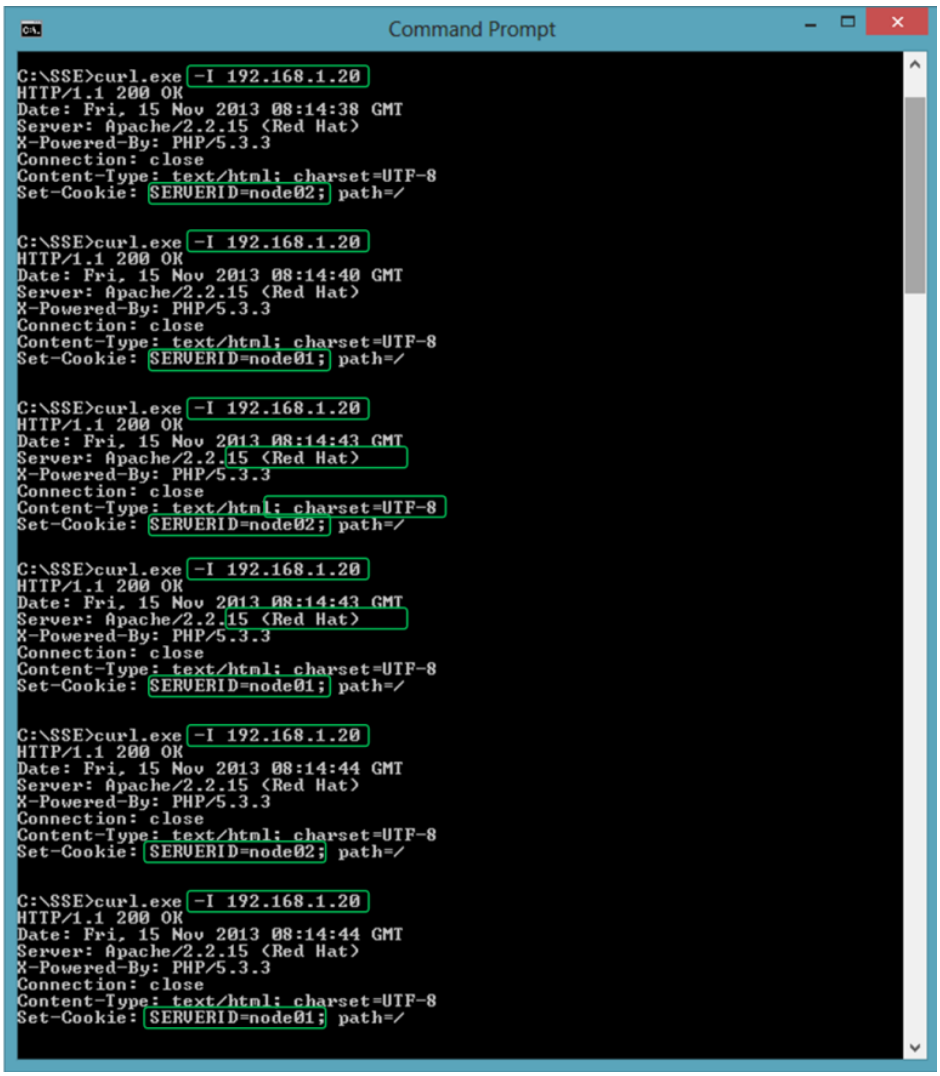

Figure 5.5. Load Balancing Testing using cURL

The summarizes of performance test as follows: First Shared Storage for Load Balancing Node for servicing (NFS) is configured in Cluster; after that High Availability Cluster is tested with showing the statuses of Clusters. Load balancing performance is tested using GUI and finally tested by using cURL. Result shows that every request is handled by new server by round-robin scheduling and the system will overcome failover by moving all application to the standby node within very short time.

6. Conclusion

This article designs and implements a low cost two-tier architecture model for high availability cluster combined with load-balancing technology and shared storage technology to achieve full facilities of the three-tier architecture for application load balancing *e.g.*, web servers. The paper described design physically omits Network File System (NFS) Server Nodes and implements NFS server functionalities within the Cluster Nodes, through Red Hat Cluster Suite (RHCS) with High Availability (HA) Proxy load balancing technologies - in

order to achieve low-cost investment in expensive hardware and computing solutions. For this design implementation, we use 2 nodes for web server Clusters, 2 nodes for Load balancer and two SAN. For any customized system the modified design can be used up-to 16 nodes for Web Server Clusters, any numbers (n) of nodes for Load balancer. For this system some open-source software are used for configuration and implementation; these are – Red Hat Enterprise Linux (RHEL) 6.4 for load balancer nodes and Cluster nodes; OpenFiler for SAN; Red Hat Cluster Suite (RHCS) for 2 cluster nodes; HAProxy for 2 load balancer nodes; Apache, Network File System (NFS) and PHP.

This system can provide continuous service though any system components fail uncertainly such as network system, storage and application. This system can overcome in the cases of node failover, network failover, storage limitation and distributions load as like as the all the facilities of three-tier architecture model for high availability cluster for application load balancing with limited physical resources for low cost implementation.

References

[1] Red Hat cluster suite, from web: en.wikipedia.org/wiki/Red_Hat_cluster_suite.
[2] Linux Virtual Server, from web: www.linuxvirtualserver.org/.
[3] C. Zheng, "Design and Implementation of a Load-Balancing Model for Web-Sever Cluster Systems", Engineering and Technology (S-CET), 2012 Spring Congress on. IEEE, **(2012)**.
[4] www.linuxvirtualserver.org/architecture.html.
[5] IPVS (IP Virtual Server) from web www.linuxvirtualserver.org/software/ipvs.html.
[6] KTCPVS (Kernel TCP Virtual Server) from web www.linuxvirtualserver.org/software/ktcpvs/ktcpvs.html.
[7] Z. Han and Q. Pan, "LVS Cluster Technology in the Research and Application of State-owned Asset Management Platform", Proceedings of the 2nd International Conference on Computer Application and System Modeling, **(2012)**.
[8] Y. Jiao and W. Wang, "Design and Implementation of Load Balancing of Distributed-system-based Web server", Electronic Commerce and Security (ISECS), Third International Symposium, IEEE, **(2010)** July, pp. 337-342.
[9] C. Zheng, J. Xia, Q. Wang and X. Chu, "Design and Implementation of a Load-Balancing Model for Web-Sever Cluster Systems", Engineering and Technology (S-CET), Spring Congress, IEEE, **(2012)** May, pp. 1-4.
[10] A. Krioukov, P. Mohan, S. Alspaugh, L. Keys, D. Culler and R. Katz, "Napsac: Design and implementation of a power-proportional web cluster", ACM SIGCOMM computer communication review, vol. 41, no. 1, **(2011)**, pp. 102-108.
[11] H. Jiang, "Design, implementation, and performance of a load balancer for SIP server clusters", IEEE/ACM Transactions on Networking (TON), vol. 20, no. 4, **(2012)**, pp. 1190-1202.
[12] V. Cardellini, M. Colajanni and P. S. Yu, "Dynamic load balancing on web-server systems", Internet Computing, IEEE, vol. 3, no. 3, **(1999)**, pp. 28-39.
[13] T. Schroeder, S. Goddard and B. Ramamurthy, "Scalable web server clustering technologies", Network, IEEE, vol. 14, no. 3, **(2000)**, pp. 38-45.
[14] L. Aversa and A. Bestavros, "Load balancing a cluster of web servers: using distributed packet rewriting", Performance, Computing, and Communications Conference, 2000. IPCCC'00. Conference Proceeding of the IEEE International. IEEE, **(2000)**.
[15] L. A. Barroso, J. Dean and U. Holzle, "Web search for a planet: The Google cluster architecture", Micro, IEEE, vol. 23, no. 2, **(2003)**, pp. 22-28.
[16] V. Cardellini, M. Colajanni and P. S. Yu, Redirection algorithms for load sharing in distributed Web-server systems", Distributed Computing Systems, 1999. Proceedings. 19th IEEE International Conference on. IEEE, **(1999)**.
[17] V. Cardellini, M. Colajanni and P. S. Yu, "Geographic load balancing for scalable distributed web systems", Modeling, Analysis and Simulation of Computer and Telecommunication Systems, 2000. Proceedings. 8th International Symposium, IEEE, **(2000)**.
[18] Z. Shan, "Modeling and performance analysis of QoS-aware load balancing of web-server clusters", Computer Networks, vol. 40, no. 2, **(2002)**, pp. 235-256.
[19] G. U. O. C. C. Y. A. N. Liu, "A Dynamic Load-Balancing Algorithm for Heterogeneous Web Server Cluster", Chinese Journal of Computers, vol. 2, **(2005)**, pp. 004.

[20] L.-C. Chen and H.-A. Choi, "Approximation Algorithms for Data Distribution with Load Balancing of Web Servers", cluster, vol. 1, **(2001)**.
[21] E. Casalicchio and M. Colajanni, "A client-aware dispatching algorithm for web clusters providing multiple services", Proceedings of the 10th international conference on World Wide Web. ACM, **(2001)**.
[22] S. Sharifian, S. A. Motamedi and M. K. Akbari, "A content-based load balancing algorithm with admission control for cluster web servers", Future Generation Computer Systems, vol. 24, no. 8, **(2008)**, pp. 775-787.
[23] V. Cardellini, M. Colajanni and P. S. Yu, "DNS dispatching algorithms with state estimators for scalable Web-server clusters", World Wide Web, vol. 2, no. 3, **(1999)**, pp. 101-113.

Authors

A B M Moniruzzaman Received his B.Sc (Hon's) degree in Computing and Information System (CIS) from London Metropolitan University, London, UK and M.Sc degree in Computer Science and Engineering (CSE) from Daffodil International University, Dhaka, Bangladesh in 2005 and 2013, respectively. Currently he is working as a Lecturer of the department of Computer Science and Engineering ad Daffodil International University. He is also working on research on Cloud Computing and Big Data Analytics as a research associate at RCST (Research Center for Science and Technology) at Daffodil International University (DIU), Dhaka, Bangladesh. Besides, his voluntarily works as reviewer of many international journals including IEEE, Elsevier, IGI-Global. He has got some international publications including journals in IGI-Global, InderScience, SERSC, Global, IJCA, IJACSA and proceedings. He is a member of IEEE. His research interests include Cloud Computing, Cloud Applications, Open-source Clouds, Cloud Management Platforms, Building Private and Hybrid Cloud with FOSS software, Big Data Management, Agile Software Development, Hadoop, MapReduce, Parallel and Distributed Computing, Clustering, High performance computing, Distributed Databases, NoSQL Databases.

Prof. Dr. Syed Akhter Hossain is Post Doctoral Fellow, Informatics and Systems Engineering, LIESP Laboratory, Universite Lyon2, Lyon, France.He received PhD in Computer Science and Engineering from University of Dhaka, Bangladesh and MSc degree in Applied Physics and Electronics, First Class (First), and BSc (Hons) in Applied Physics and Electronics, First Class (First), Gold Medalist from Rajshahi University, Rajshahi, Bangladesh. Currently he is working as Professor and Head, Department of Computer Science and Engineering, Daffodil International University, Dhaka, Bangladesh. Besides, he received best professor award from Singapore and has got more than 60 international publications including journals and proceedings and 3 book chapters with IGI Global and John Wiley. He is a member of ACM, and member of IEEE. His research areas includes simulation and modeling distributed system design and implementation, signal and image processing, internet and web engineering, network planning and management, database and data warehouse modeling.

