Implementation of Linux Virtual Server Clustering

(For Higher Availability of Server)

Prof. Savita Lohiya
Information Technology
SIES Graduate School of Technology
Navi Mumbai, India
savita.lohiya@siesgst.ac.in

Harisree Mukku
Information Technology
SIES Graduate School of Technology
Navi Mumbai, India
harisreemukku@gmail.com

Pooja Salunkhe
Information Technology
SIES Graduate School of Technology
Navi Mumbai, India
poojasalunkhe20@gmail.com

Aman Salunkhe
Information Technology
SIES Graduate School of Technology
Navi Mumbai, India
amansalunkhe16@gmail.com

***Abstract*— With the explosive growth of the network, the workload on the servers is increasing rapidly, servers will be easily overloaded for a short time. Linux Virtual Server (LVS) project which is an open source project overcomes the overloading problem of the servers. Virtual server is a highly scalable and highly available server built on a cluster of real servers. The architecture of server cluster is fully transparent to end users, and the users interact with the cluster system as if it were only a single high-performance virtual server.**
The implemented ways of the virtual server are introduced. A LVS cluster of web servers based on VS/DR is presented for an online score system. The result of the application shows that the highly available LVS system is available and effective.

I. INTRODUCTION

To overcome the overloading problem of the servers, there are two solutions. One is the single server solution, i.e. to upgrade the server to a higher performance server, but it will soon be overloaded when requests increases so that we have to upgrade it again, the upgrading process is complex and the cost is high. The other is the multiple server solution, i.e. to build a scalable network service system on a cluster of servers. When load increases, we can simply add a new server or more into cluster to meet the increasing requests, and commodity server is of highest performance/cost ratio. Therefore, it is more scalable and more cost-effective to build server cluster system for network services.

More and more mission-critical applications move on the Internet, providing highly available services becomes increasingly important. One of the advantages of a clustered system is that it has hardware and software redundancy, because the cluster system consists of a number of independent nodes, and each node runs a copy of operating Ease of Use system and application software. High availability can be achieved by detecting node or daemon failures and reconfiguring the system appropriately, so that the workload can be taken over by the remaining nodes in the cluster.

High availability is a big field. An advanced highly available system may have a reliable group communication sub-system, membership management, quorum sub-systems, concurrent control sub-system and so on. There must be a lot of work to do. However, we can use some existing software packages to construct highly available LVS cluster systems now.

II. NEED OF PROJECT

In the existing system the server which handles the request of the users, can handle those requests only up to a certain limit. Once the server crosses its limit it crashes. And the users are unable to connect due to unavailability of the server. So to overcome the drawback of the existing system we have come up with our project “Implementation of LVS clustering for the higher availability of a server “

III. LITERATURE SURVEY

We have referred the following papers for our project:

1. Mr. Haesun Shin, introduces a load-balancing cluster system based on Linux Virtual Server designed for Internet-based Electrical Engineering Virtual Lab, and presents a dynamic feedback algorithm based on neural network to solve unbalanced scheduling problem and improve the performance and stability of the cluster. [3]

2.Computer Sciences and Applications (CSA), 2013 International Conference, This paper mainly introduces the architecture and basic scheduling algorithm based on the LVS cluster, and also improve the existing load balancing algorithm and put forward a new load balancing algorithm which combines the static and dynamic scheduling algorithm. The simulation test results show that the algorithm has improves the comprehensive performance of the system to some extent and has a good adaption. [7].

3. Haesun Shin , This paper presents a Linux virtual server (LVS) based cluster system which is designed to simulate the operation of E-mail service in the real world. For a complete study of the system we build a load generator that is used to perform a series of stress-tests with LVS load-balancing technology. We obtain important results by comparing a series of stress-testing reports. After we take some steps to resolve problems of the system we obtain the expected results, that is, a well-designed system which provides scalable, reliable and highly cost-efficient E-mail service. [5].

IV. LOAD BALANCING

These days, LVS (Linux Virtual Server) software, which is free of charge and has good performance, has commonly been used to construct web server cluster. But when requests are increased, LVS can raise a bottleneck and can make the cluster system unable after all, because it has only single front-end. In this paper, we suggest a new architecture for web server cluster based on LVS with multiple front-ends which can act as back-ends simultaneously. We also propose a scheduling algorithm to distribute requests equally to servers with considering their load. The proposed system and algorithm remove the bottleneck, and are useful in constructing small and middle-sized applications. In the proposed system, a server can make response directly to clients request only when its load is not large enough. Otherwise, the server should redirect the request to a selected back-end with the lowest load, so it can avoid load.

V. METHODOLOGY

A. *User Reqirements:*

Should be able to run Skype application with the help of kamailio software and the Linux virtual server must be able to load balance the system when more number of users are accessing services.

B. *System Requirements:*

1) Hardware requirements:
   - High configured laptops.
2) Software requirements:
   - Linux
   - Kamailio SIP server
   - Nagios tool

C. *Functional Requirements:*

1) Linux Virtual Server :

Linux Virtual Server is a software tool that directs network connections to multiple servers that share their workload, which can be used to build highly scalable and highly available services

2) Kamailio:

Kamailio® (successor of former OpenSER and SER) is an Open Source SIP Server released under GPL, able to handle thousands of call setups per second. Kamailio can be used to build large platforms for VoIP and real time communications.

3) Database:

It is used to store IP user's data.

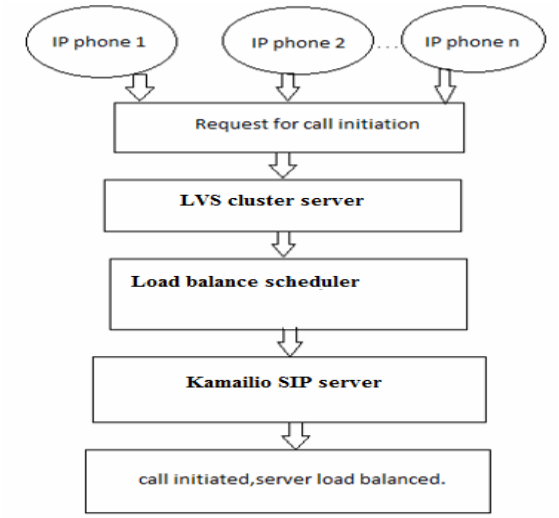

Figure1. Flowchart for LVS clustering

VI. DESIGN

A software design is a description of the structure of the software to be implemented, the data which is part of the system, the interfaces between the system components and sometimes the algorithms used.

A. *Architectural Design*

In this section we present system architecture for building highly scalable and highly available network services on clusters.

1) Load balancer is the front end to the service as seen by the outside world. The load balancer directs network connections from clients who know a single IP address for services, to a set of servers that actually perform the work.
2) Server pool, consist of a cluster of servers that implement the actual services.
3) Users: users make call from one IP to IP, which is scheduled by Linux virtual server to the kamailio server

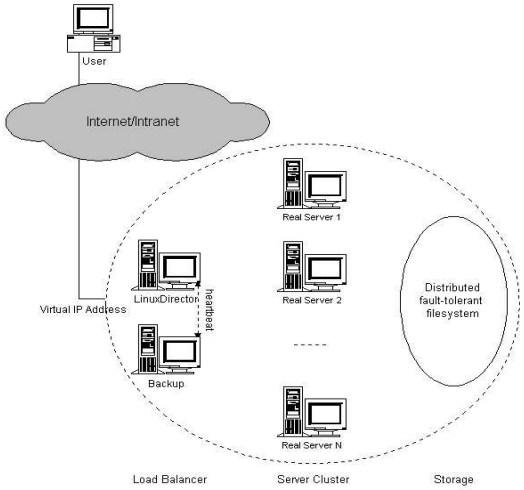

Figure 2. Architecture of Linux Virtual Server

The load balancer handles incoming connections using IP load balancing techniques, it selects servers from the server pool, maintains the state of concurrent connections and forwards packets, and all the work is performed inside the kernel, so that the handling overhead of the load balancer is low. Therefore, the load balancer can handle much larger number of connections than a general server, thus a load balancer can schedule a large number of servers.

The load balancer handles incoming connections using IP load balancing techniques, it selects servers from the server pool, maintains the state of concurrent connections and forwards packets, and all the work is performed inside the kernel, so that the handling overhead of the load balancer is low. Therefore, the load balancer can handle much larger number of connections than a general server, thus a load balancer can schedule a large number of servers.

We have implemented four scheduling algorithms for selecting servers from the cluster for new connections: Round-Robin, Weighted Round-Robin, Least-Connection and Weighted Least-Connection. The first two algorithms are self-explanatory, because they don't have any load information about the servers. The last two algorithms count active connection number for each server and estimate their load based on those connection numbers.

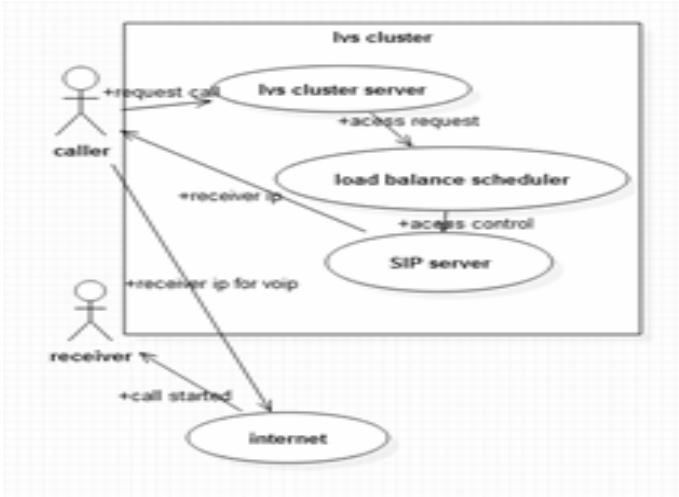

Figure 3. Use case Diagram

VII. IMPLEMENTATION

Linux Virtual Server directs network connections to the different servers according to scheduling algorithms and makes parallel services of the cluster to appear as a virtual service on a single IP address. Client applications interact with the cluster as if it were a single server. The clients are not affected by interaction with the cluster and do not need modification. Scalability is achieved by transparently adding or removing a node in the cluster.

High availability is provided by detecting node or daemon failures and reconfiguring the system appropriately, so that the workload can be taken over by the remaining nodes in the cluster. We usually have cluster monitor daemons running on the load balancer to monitor the health of server nodes, if a server node cannot be reached by ICMP ping or there is no response of the service in the specified period, the monitor will remove or disable the server in the scheduling table of the load balancer, so that the load balancer will not schedule new connections to the failed one and the failure of server nodes can be masked.

A. LEVEL 0

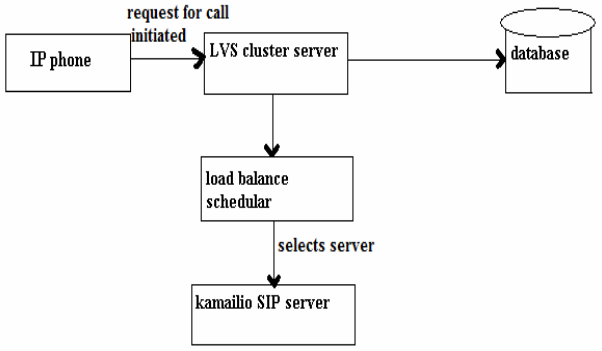

Figure 4. Level 0 Dataflow diagram for LVS clustering

B. LEVEL 1

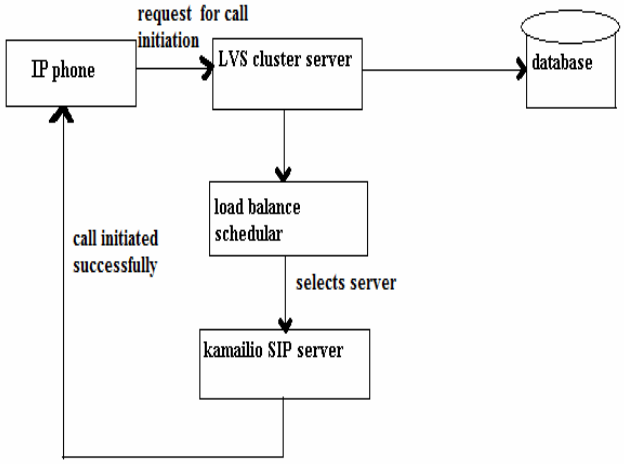

Figure 5. Level 1 Dataflow diagram for LVS clustering.

Now, the load balancer may become a single failure point of the whole system. In order to prevent the failure of the load balancer, we need setup a backup of the load balancer. Two heartbeat daemons run on the primary and the backup, they heartbeat the health message through heartbeat channels such as serial line and UDP periodically. When the heartbeat daemon on the backup cannot hear the health message from the primary in the specified time, it will use ARP spoofing (gratuitous ARP) to take over the virtual IP address to provide the load-balancing service. When the primary recovers from its failure, there are two methods. One is that the primary becomes to the backup of the functioning load balancer; the other is that the daemon receives the health message from the primary and releases the virtual IP address, and the primary will take over the virtual IP address. However, the failover or the takeover of the primary will cause the established

connection in the state table lost in the current implementation, which will require the clients to send their requests again.

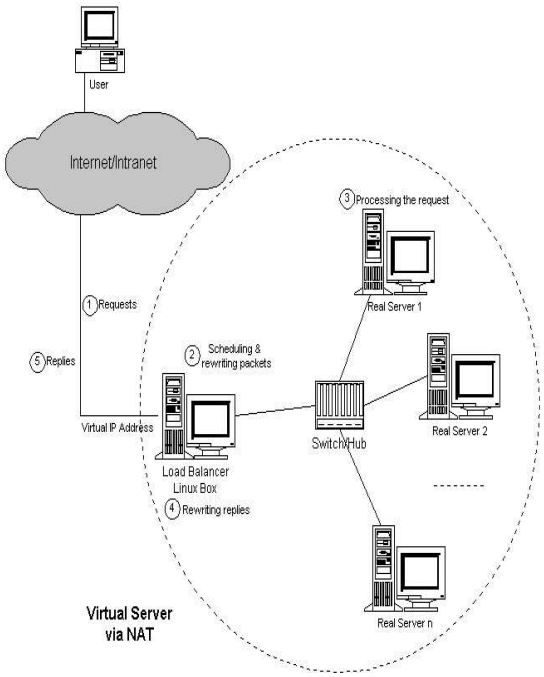

Figure 6. Implementation of Linux Virtual Server

VIII. APPLICATION OF LVS CLUSTERING

It provides IP to IP calling for Inter organizational purpose, where many employees can communicate with each other at low cost over internet.

IX. CONCLUSION

When IP to IP calling traffic increases it increases load on server. To prevent server crash due to overload we use LVS clustering is used. Using LVS clusters we are able to derive more reliable and efficient server. High availability of data is provided by LVS clusters for the users.

X. ACKNOWLEDGMENT

We feel privileged to express our deepest sense of gratitude and sincere thanks to our project guide **Prof. Savita Lohiya** for her excellent guidance throughout our project work. Her prompt and kind help led to the competition of work.

XI. REFERENCES

[1]. Information Processing (ISIP), 2008 International Symposiums on23-25 May 2008 ,YangXu.
[2]. Open-source Software for Scientific Computation (OSSC), 2009 IEEE International Workshop on18-20 Sept. 2009 by Yanping Gao Sch.
[3]. Industrial Electronics and Applications (ICIEA), 2014 IEEE 9th Conference on9-11 June 2014 by Haesun Shin.
[4]. Telecommunications and Signal Processing (TSP), 2012 35th International Conference on3-4 July 2012 by Mill, Olga S.
[5]. Electrical and Computer Engineering (CCECE), 2013 26th Annual IEEE Canadian Conference on5-8 May 2013 by Haesun Shin .
[6]. Computer Sciences and Applications (CSA), 2013 International Conference on 14-15 Dec. 2013 by Ying-Wen.
[7]. Computer Sciences and Applications (CSA), 2013 International Conference on 14-15 Dec. 2013 by Ying-Wen.
[8]. Network and Parallel Computing, 2008. NPC 2008. IFIP International Conference on18-21 Oct. 2008Shouyun Song.