2021 International Conference on Computer Communication and Artificial Intelligence (CCAI) | 978-1-7281-9401-1/20/$31.00 ©2021 IEEE | DOI: 10.1109/CCAI50917.2021.9447535

Load Balancing Design of Web Cluster Based on Nginx under Novel Virtualization Platform

Yuanqing Cao
College of Mathematics and Computer Science
Hetao University
Bayannur, China
e-mail: htxycyq@163.com

***Abstract*—With the increase of net citizen and the degradation of web server performance, application of load balancing technology in web cluster is a common approach to resolve the above conflict. Several popular load balancing technologies and advantages of Nginx is introduced first. Then a new implementation of Nginx load balancing is proposed, which is based on novel Cloud-View Virtual Machine (CVM) virtualization platform. After the optimization of Nginx, the response time of the proposed CVM solution and the traditional KVM solution are compared and tested. Experimental results indicate that new method can choose the rear server more reasonably, improve the performance and stability of the cluster more effectively, in addition, it can reduce system latency, and satisfy the design requirements.**

Keywords—load balancing, web cluster, nginx, CVM

I. INTRODUCTION

As the most used network services on the modern Internet, web services have penetrated all aspects of our daily lives, including online shopping, social networking, finance, and education. The growing number of web users directly leads to the insufficient of web server performance. In addition, the personal experience of users with the characteristics of high concurrency has become worse and worse, and the comprehensive performance of web sites has become lower and lower. As a result, the development of services is severely restricted [1].

The most effective way to solve the above problems is web server cluster strategy. Specifically, the web system is no longer composed of one server, but is composed of multiple distributed servers [2], however, it is just like visiting one server for users. All the high concurrent requests of users are reasonably allocated to the rear servers according to a certain algorithm, and the resources of each server (CPU, memory, hard disk, SCSI bus, etc.) are fully mobilized. Moreover, the operating mode of web service shift from "one (server) to one (client)" to "multiple (server) to one (client)" [3]. Consequently, user access speed and web system performance are greatly improved, client response time is shortened, and the problem of high concurrent access to application server is solved. In addition, the web cluster scheme not only defends the attack from the external network, but also defends the attack from the internal network.

II. LOAD BALANCING TECHNOLOGY

Load balancing is one of the most common strategies in server clusters, its most significant feature is that the relationship between each server is equal, and they work together to provide services to the user group according to certain algorithm rules [4]. In addition, load balancing can also improve the comprehensive performance of the web system, especially in a high load environment. There are many methods to realize load balancing, such as hardware load balancing strategy, software load balancing strategy, IP load balancing strategy, LVS load balancing strategy, the most widely used of which are hardware load balancing strategy and software load balancing strategy.

Hardware load balancing is to install a physical equipment between the Internet gateway and the web server, we usually call it load balancer, because of a single device completing a single task, the system performance is particularly high, and the load balancing effect of which is excellent, but the only drawback is the expensive price [5].

Different from hardware load balancing, software load balancing strategy is to install given software on the operating system of each web server to achieve load distribution. The mainstream software includes Connect Control, DNS Load Balance, Nginx, Check Point Firewall-1 [6], HA Proxy, etc. They have the advantages of easy configuration, simple operation, cheap price and so on, but the performance is worse than hardware Load balancing, among which Nginx is the most common and classic strategy.

III. NGINX

A. Origin and Application of Nginx

Nginx was created by Igor Sysoev [7], a well-known Russian system administrator, and was originally used to develop the Russian national web portal named Rambler.ru. The source code for Nginx was released in 2004, and it has the advantages of strong stability, easy deployment, strong performance, and high code quality.

Relevant statistics show that 11.6% of web application servers in the world are using Nginx software from 2004 to 2019. In addition, 28% of the top two thousand popular websites are based on the Nginx solution [8], and this ratio is increasing day by day. For example, many well-known portals

such as Netease, Phoenix, Sohu, Xinhua, Ranger, etc. are all in use of Nginx.

B. *Advantages of Nginx*

Nginx is so popular because of the following advantages:

- Response time to client TCP requests is extremely short in the case of tens of thousands of concurrent.
- Expansibility of cluster is very strong. Nginx is made up of several different functional modules that are different in functionality and hierarchy, and are particularly weak-coupled. This means that when one of the modules fails, the others can operate without a single module failure, which is very beneficial to the troubleshooting process.
- Nginx has simple architecture design, excellent module deployment and strong reliability. Most Web sites with high concurrency requirements will choose Nginx.
- The number of concurrencies on a single server can be up to 100,000, which on the cluster is greater.
- The Nginx open-source code is a boon for programmers who can customize the user experience by modifying the code [9].

IV. DESIGN OF CVM VIRTUALIZATION PLATFORM

Cloud-View Virtual Machine (CVM) virtualization platform has an open architecture, which is based on OSGi modular mechanism, and its functional components can be flexibly combined. What's more, its standard service interface can be compatible with third-party servers, storage, network solutions, and compatible with Xen, VMware, and other heterogeneous virtualization platforms. In addition, it also has high scalability, which can realize physical grouping and logical grouping management of physical resources and virtual resources in the platform, therefore, the scalability requirements of server, storage and network resources are met. Besides mainstream operating systems such as Windows, RedHat, SUSE are supported, supporting virtual machine creation, start and stop, dynamic migration, configuration modification, performance monitoring, remote access and other functions are provided by Cloud-View virtualization platform.:

A. *Shared Storage Model of CVM System*

Proposed design of Shared Storage model is based on the fiber channel SAN (Storage Area Network) of disk subsystem, altogether consists of five RAID 5.0 SCSI disk array of virtualization resources pool, two of them are provided by the disk array, three of them are provided by the local SCSI Storage of physical servers. Five disk arrays connected by optical storage switches constitute a fiber channel storage LAN, this architecture not only combines the channel transmission characteristics and long-distance characteristics of I/O bus, but also provides the flexible connectivity and high scalability that traditional network has.

B. *Architecture of CVM System*

CVM system is deployed on three physical Server (hvn0, hvn1 and hvn2), in this way, a virtualization platform resource pool was built, then, together with the SAN disk subsystem, four virtual servers were created, one is used as NGINX load balancing virtual machine (NGINX-server), and the other three are used as web application service virtual machines (WEB-server1, WEB-server2, and WEB-server3). The platform architecture is shown in Fig. 1

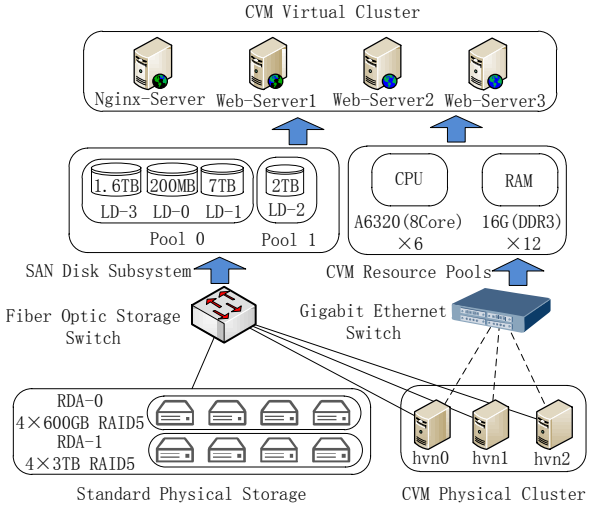

Fig. 1. The cluster architecture of CVM virtualization platform.

C. *Nginx Configuration*

There are five major Nginx algorithms, which are polling, IP_HASH, WEIGHT, FAIR and URL_HASH, and each algorithm has its own application conditions. Due to the emphasis of the new design is the contrast of Nginx load balancing based on different virtualization platform, rather than design of a new Nginx algorithm, so, in order to ensure the consistency and reliability of the test result, the configuration of all virtual machines is set to be the same, and the default Nginx rotation algorithm is polling. Nginx version is 1.9.9, and the configuration is shown below:

```
upstream foreverdant{
                server 192.168.0.168;
                server 192.168.0.169;
                server 192.168.0.170;
 }
  server {
     listen       80;
     server_name  localhost;
     location / {
        root   html;
        index  index.html index.htm;
        proxy_pass http://foreverdant;
```

D. *Nginx Optimization*

As a high-performance load balancing solution with flexible configuration, Nginx can improve the performance of the load balancing system by changing some configuration items in the configuration file. The following is to optimize the performance of Nginx by changing default configurations, with the purpose of faster visit speed of Nginx clients.

```
worker_processes  4;
events {
   worker_connections  4096;
}
worker_cpu_affinity 0001 0010 0100 1000;
```

First optimization item is "worker processes", which refers to the number of worker processes. Nginx master module is the object of setting up the worker processes, which can directly control the number of worker, specifically, if the value is too high, it can lead to system constantly switching back and forth between the different worker processes, and the burden of the system greatly increase; If the value is too low, it will makes it harder for the system to handle concurrent requests, thus many system is unable to make full use of idle resources, so, it is recommended to set this parameter to the number of CPU cores.

"worker_connections" is the second optimization item, it refers to the maximum number of connections that each worker process can process concurrently, which cannot exceed the maximum number of files already opened. The recommended value for this parameter is the product of the number of CPU cores and 1024.

The third optimizer, "worker_cpu_affinity", binds each kernel of the CPU to the worker process. The purpose of this optimization is to reduce the time required to select the kernel in the process of processing, thus greatly reducing the system overhead.

V. TESTING AND COMPARATIVE ANALYSIS

In order to verify the performance of Nginx load balancing based on CVM virtualization platform, The KVM-based load balancing strategy and CVM-based load balancing strategy with the same Nginx configuration were compared.

Firstly, CVM and KVM are deployed on three physical machines with the same configuration respectively. Then, four virtual machines with the same configuration are created on the CVM platform and KVM platform respectively. The configuration of physical machines and virtual machines is shown in Table I.

TABLE I. CONFIGURATION OF PHYSICAL MACHINE AND VIRTUAL MACHINE

Machine	CPU	Memory	Storage	Subhead
physical machine	A6320(8 Core 3.3GHz)×2	64G DDR3	600G_10K _6G×2	Redhat6.5 x64
virtual machine	4 Core 2.8 GHz	4G DDR3	50G	CentOS6.5 X64 Minimal

The architecture of the server cluster in the test environment is shown in Fig. 2. The Nginx load balancing server is used to receive and process concurrent requests from clients, and the requests are reasonably distributed among the cluster web servers according to the load balancing algorithm.

Finally, the stress tests of CVM and KVM are carried out by using Httperf, a stress testing tool. The reliability of a solution is always determined by the choice of testing tool, httperf is a highly efficient and reliable HTTP pressure testing tool, it can simulate multiple clients and the practical application of the high number of concurrent connections scenarios, or large load for a long time to test the performance of web application server can support 10000 concurrent access, can fully test the performance of the web server. Its test results focus on response rate and response time. It can simulate the actual application scenarios of multiple clients and high concurrent connections, and test the performance of the Web application server for a long time or a large load. It can also support up to 10,000 concurrent visits, which can fully meet the performance test requirements of Web server. In addition, its test results focus on response rate and response time.

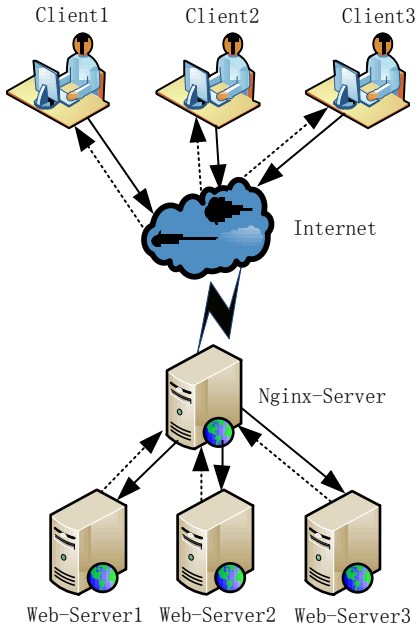

Fig. 2. Architecture of virtual server cluster.

After installing HTTPERF-0.9.0 under CENOS 6.4, enter the following command to perform the stress test:

```
httperf --server 192.168.0.167 --port 80 --uri /index.html --rate 500 --num-conn 1000--num-call 1 –timeout 10
```

In the above command, rate represents requests sent per second, num-conn refers to the total number of connections, and num-call represents the number of requests sent per connection.

The newly designed test process is as follows: firstly, the response time of single physical machine and Nginx cluster based on CVM virtualization is compared and tested, the number of concurrent connections is increased from 500 to 5000, and 10 data points (step size is 500) are tested respectively; Then the response time of the Nginx virtual machine cluster based on CVM virtualization and KVM virtualization were compared and tested, the number of concurrent connections is increased from 1000 to 7000, and 7 data points (step size was 1000) are tested respectively. All the data points were tested for three times, and finally the arithmetic average was taken. After completion of the test, the test results were visualized by Excel and the line chart was output. The test data of Nginx cluster based on CVM virtualization and KVM virtualization are shown in Table II and Table III respectively, and the visual charts are shown in Fig. 3 and Fig. 4 respectively:

TABLE II. CLUSTER RESPONSE TIMES OF SINGLE PHYSICAL MACHINE AND CVM + NGINX

Number of concurrent connections	Cluster Response Time	
	Single physical machine	***CVM + Nginx***
500	6.08	5.68
1000	8.98	4.92
1500	30.86	8.18
2000	69.3	6.12
2500	127.9	8.78
3000	170.58	9.9
3500	188.06	11.64
4000	202.98	18.22
4500	222.2	16.7
5000	239.06	24.52

TABLE III. CLUSTER RESPONSE TIMES OF CVM + NGINX AND KVM + NGINX

Number of concurrent connections	Cluster Response Time	
	KVM + Nginx	***CVM + Nginx***
1000	4.76	4.7
2000	4.66	4.66
3000	4.96	5.86
4000	10.06	9.04
5000	16.96	14.8
6000	41.96	33.22
7000	63.36	47.46

It can be seen from Fig. 3 that the response time of a single physical machine and a CVM + Nginx cluster is basically the same when the concurrency is lower than 1000. When it is higher than 1000, the response time of a single physical machine increases sharply as the concurrency increases. The response time of the CVM + Nginx cluster is still very fast, which shows that the performance of the CVM + Nginx cluster is significantly better than that of a single physical machine, and the level of multitasking will be even better.

As can be seen from Fig. 4, when the number of concurrencies is lower than 5000, the response time of KVM + NGINX cluster and CVM + NGINX cluster are both ideal, within 20ms. However, when the number of concurrencies is higher than 5000, the response time of KVM + NGINX cluster is slower than that of CVM + Nginx cluster, which indicates that the performance of CVM + Nginx cluster is better than that of KVM + Nginx cluster.

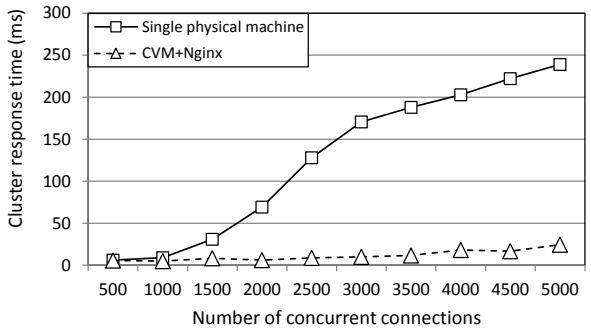

Fig. 3. Response time compare of single physical machine and CVM + Nginx.

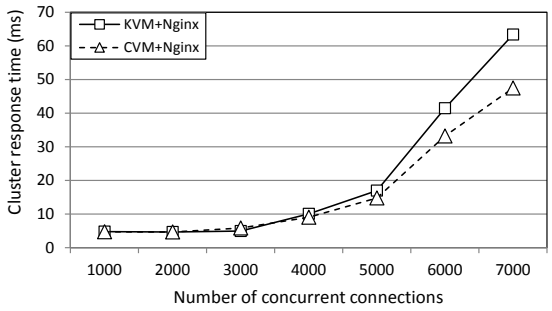

Fig. 4. Response time compare of CVM + Nginx and KVM + Nginx.

VI. CONCLUSION

A new Nginx implementation method is proposed, that is, the Nginx load balancing cluster solution based on CVM. System response time and other parameters of the traditional KVM-based Nginx cluster and the novel CVM-based Nginx cluster are compared under the same hardware conditions. The results show that the new solution has faster corresponding speed and obvious performance advantages, which verifies that the new solution is feasible and can allocate load servers more reasonably for client requests. Accordingly, the goal of web load balancing is achieved, and the basic requirements of the design are met.

Future work is to compare the performance of CVM with other virtualization platforms in more detail, such as Xen, etc. In addition, further research on Nginx optimization algorithm that is more suitable for the new virtualization platform is needed. The application of virtualization should also be extended from simple Web application to cloud computing and big data field.

REFERENCES

[1] J. K. Ramana and M. Ponnavaikko, A Multi-Class Load Balancing Algorithm（MCLB）for Heterogeneous Web Cluster, vol. 27. Studies in Informatics& Control, 2018.

[2] J. L. Jiayue, F. Zhiyi and W. Baiqi, The Improvement and Implementation of the High Concurrency Web Server Based on Nginx, vol. 1. Computing, Performance and Communication Systems, 2016.

[3] J. L. Ruoyu, L. Yunchun and L. Wen, An Integrated Load-balancing Scheduling Algorithm for Nginx-Based Web Application Clusters, vol. 1060. Journal of Physics: Conference Series, 2018.

[4] J. H. Zhijie, W. Yalu and Z. Hui. Web Load Balancing Based on DNS Coordination and Reducing Energy Consumption Strategy, vol. 91. Springer US, 2019.

[5] J. T. Bezboruah and A. Bora, Some Aspects of Implementation of Web Services in Load Balancing Cluster-Based Web Server, vol. 10. International Journal of Information Retrieval Research (IJIRR), 2020.

[6] J. Shukla, Kumar and Singh, Fault tolerance based load balancing approach for web resources, vol. 42. Journal of the Chinese Institute of Engineers, 2019.

[7] J. P. Krill, Nginx web server upgrade focuses on web security, JavaScript configuration, InfoWorld.com, 2016.

[8] J. B. She, S. Bo, W. Qiang, Z. Xiaoge, Z. Zhe, Q. Zunying and L. Guodong, The Design and Implementation of Campus Network Streaming Media Live Video On-Demand System Based on Nginx and FFmpeg, vol. 1631. Journal of physics. Conference series, 2020.

[9] J. S. Qifeng, Y. Tianchi and H. Wei, The Design of High Available Single Sign-On Server of Nginx-Based, vol. 2111. Applied Mechanics and Materials, 2013.