Design and Implementation of Load Balancing Technology on Linux-Based Servers

Sukarno¹, Nur Laila Meilani²

¹National Population and Family Planning Board, Indonesia ²Universitas Riau, Indonesia
lararashida@gmail.com, meilanimeilani09@gmail.com

*Corresponding Author
Received: 9 October 2017, Accepted: 6 November 2017
Published online: 25 February 2018

Abstract: The rapid growth of smartphone applications and other applications requires cloud computing with continuous expansion, increasing cloud computing is an important issue to improve performance and increase system resources appropriately. One of the computing system is the server computer, but when using simultaneously and in many points there will be overload or overcapacity. This paper explains and proposes a method, design and development of Linux Virtual Server to distribute the workload to multiple hosts so that the load becomes lighter. The method offered will provide a better solution that can sort requests based on the first server processing or lock cache and implement the Round Robin algorithm to distribute the workload. The addition of HAProxy to a front-end / controller for several separate server applications can monitor web-based statistics can see the real conditions of each server. With the implementation of the system will provide solutions to overloading problems.

Keywords: Server; Load Balancing; Linux; HAProxy; Smartphone.

1. Introduction

The skyrocketing growth of smartphones in Indonesia that is expected to reach 100 million by 2018 is a good opportunity to take advantage of existing resources [1]. Smartphones are in great demand for most people because of their role and usefulness especially for health field surveys which hope to be fast information media about maternal, infant and toddler death information and public health status [2]. Field survey data collected in the smartphone stored first then can be sent to the server that has been provided. This smartphone-based application will increase the number of clients that must be served by the server so that users will increase drastically according to growth, some sites have even received hundreds of thousands of connections from clients simultaneously. To increase flexibility, server availability and security can be developed from single server to multi server by load balancing method that can distribute its task to multiple hosts so that the average of the task will be shorter and can increase the processor utility.

Therefore, the requirements for hardware and software solution to support highly scalable and highly available services can be summarized as follows [3]:

a. Scalability, when the load offered to the service increases, system can be scaled to meet the requirement.
b. 24x7 availability, the service as a whole must be available 24x7, despite of transient partial hardware and software failures.
c. Manageability, although the whole system may be physically large, it should be easy to manage.
d. Cost-effectiveness, the whole system must be economical to afford and expand.

A single server is usually not sufficient to handle this aggressively increasing load. The server upgrading process is complex, and the server is a single point of failure. The higher end the server is
Design and Implementation of Load Balancing Technology on Linux-Based Servers

upgraded to, the much higher cost we have to pay. One of the computing system is the server computer, but when using simultaneously and in many points there will be overload or overcapacity. This paper explains and proposes a method, design and development of Linux Virtual Server to distribute the workload to multiple hosts so that the load becomes lighter.

2. Related Works

This research will learn about the design of server load balancing system which has several advantages, including flexibility, availability and security. Load balancing is a method for distributing loads to multiple hosts so that the workload becomes lighter. This is so that the average time to work on tasks becomes short and can raise the processor utility. Load balancing can be implemented with special hardware, software or both. The existing standard configuration illustrates that one machine is placed between client and server, this machine is called a load balancer because the task is to provide balancing of requests from the client to the server.

A load balancer is a network device installed between client and server, working as a switch for client requests. Load balancer implements several scheduling methods that will determine which server the client requests will be forwarded.

Cloud Computing has extensively been used by the IT industry even though there are many existing issues like Load Balancing, Virtual Machine Migration, Server Consolidation, Energy Management, etc. Currently, load balancing can be classified from several different angles, including (1) static load balancing and dynamic load balancing, in which dynamic load balancing [4][5] can determine the state of cluster nodes in real time and balance scheduling timely, thus improving the overall performance without adding too much computational overhead; (2) centralized and distributed load balancing.

3. Load Balancing Server Design and Implementation

3.1. Design and system used

Prior to designing, an analysis of the architecture and specificity of the load balancing system is likely to be established. Load balancing is a technique to distribute workload evenly across two or more computers, network links, CPUs, hard drives, or other resources, in order to get optimal resource utilization, maximize throughput, minimize response time, and avoid overload. It is commonly used to mediate internal communications in computer clusters, especially high-availability clusters [6].

The Docker platform is the only container platform to build, secure and manage the widest array of applications from development to production both on premises and in the cloud. Docker Community Edition [CE] empowers developers with tools to build applications and and Docker Enterprise Edition [EE] provides IT with multi-architecture operations at scale. Docker delivers both an engine for innovation that modernizes without disruption while reducing total costs by more than 50%. Unlike other solutions that specialize in a narrow slice of the enterprise IT portfolio, Docker delivers operations at scale by addressing a diverse set of applications and infrastructure for both developers and IT[7].

HAProxy installed on the front-end server. Front-end servers are generally servers that have static IP and are registered with DNS. For load balancer high availability then normally there are at least 2 pieces of front-end server. Server applications can be installed together in front-end or installed separately. Front-end works to connect users with available server applications.

The scheme of a server load balancer system using HAProxy with a single front-end and several separate server applications can be seen in Figure 1. Nodes in the system, both load balancers and real servers all use the Ubuntu 16.04 operating system and other installed software open source. The server load balancing system model designed in this study consists of 5 nodes where 1 node as a load balancer and 4 other nodes as a real server that provides webserver application and database server. The configuration performed on the 4 nodes is as in Figure 1.

3.2. Design and Methods of Multi Server Implementation with Load Balancer

This paper is provide for survey tools that use Open Data Kit (ODK) platform, its is a free and open-source set of tools which help organizations author, field, and manage mobile data collection solutions[8]. The data collected by the user in the smartphone is stored first then can be sent to the
Design and Implementation of Load Balancing Technology on Linux-Based Servers

server that has been provided. Figure 1 shows the overall division of servers in their implementation, consisting of one controller server as (LB) Load Balancer, three webservers and one database server. Load Balancer node is also responsible for managing and automating the provisioning and de-provisioning of the underlying cloud resources. The cloud resources are typically compute (or worker) VMs (Virtual Machines) that may require access to storage or database servers.

Figure 1. Design Server Load Balancing

Also in load balancer server we install HAPROXY as Frontend. The advantages of using HAPROXY as an application for load balancing one of them is the existence of static reports in the form of web pages that can be enabled in the configuration file. The condition of each real server in the load balancing cluster can be seen in the statistical report. The following is an analysis of the HAPROXY statistics report from when the two real servers are still active until web server 1 fails and is taken over by web server 2 or web server 3.

Figure 2. Setup deployed with HAPROXY

Figure 2 we ran Ubuntu Linux Server 16.04 as the underlying operating system for all of our instances. We used the popular Apache Tomcat 7 to generate synthetic HTTP traffic which was first directed to an HAPROXY load balancer [9] that was readily available at the cloud. In addition to the basic features that come with Apache Tomcat 7 by default, we added the standard set of plugins to create aggregate summary reports. The HAPROXY was configured to distribute the HTTP requests evenly in a round robin fashion among the available medium sized VM instances to process the HTTP requests[10]. All of these Virtual Machine including the Apache Tomcat 7 and HAPROXY were hosted within the same VPC (Virtual Private Cloud).
We also install Device Hive Docker to manage management and configure across multiple servers. Figure 3 shown that the middle stack depicts Docker container software on a Linux host. Docker uses the host Linux system and packages the application and dependencies into modular containers. No VM is necessary and the OS resources for the two application stacks are shared between different containers. The right stack depicts Docker on a non-Linux system. Because Docker requires Linux, a lightweight VM with a mini-Linux Guest OS is necessary to run Docker and encapsulate the software containers. This still has the advantage that only a single VM and Guest Linux system is required regardless of the number of containers[11].

Docker is a relatively new method of virtualization available natively for 64-bit Linux. Compared to more traditional virtualization techniques, Docker is lighter on system resources, offers a git-like system of commits and tags, and can be scaled from your laptop to the cloud[12]. Running state of service nodes is reflected through a variety of load information, and thus the load evaluation determines the merits of the request-allocation algorithm. There are many the common load balancing algorithms in cloud computing environments: Round-Robin Scheduling (RR) [13], Weighted Round-Robin Scheduling (WRR) [14], Least-Connection Scheduling (LC) [15]. We introduce three of them brief as follows:

1. Round-Robin Scheduling (RR)
   RR Scheduling [13] is to distribute assignment requests sequentially to multiple cluster nodes, and that cycle repeats. RR Scheduling is simple and has high efficiency when the configurations and performance of hardware and software nodes are consistent, but when the cluster nodes have different performance and processing capacities, as RR Scheduling does not consider the load of each node, it is likely to cause load imbalance, thereby making the entire system perform poorly. RR Scheduling is adopted as the load balancing strategy for the Eucalyptus-based cloud computing platform [16].

2. Equally Spread Current Execution Load (ESCEL)
   ESCEL is a dynamic load balancing algorithm that requires a load balancer to monitor tasks to be addressed [17]. The function of the load balancer is to put the requested tasks into queue and assign them to different service nodes for processing. The load balancer frequently checks new tasks in the queue and then assigns them to a series of idle service nodes for treatment, while at the same time also maintaining the assignment list that has been assigned to be processed in the service node, which can help identify which services node is idle and new tasks can be assigned to it.

3. Throttled load balancing algorithm ( Throttled)
   The Throttled load balancing algorithm is entirely based on the virtual machine and it is a dynamic load balancing algorithm [18]. In this algorithm, the user requests the load balancer to a suitable virtual machine to perform the task requested. In cloud computing of multiple virtual machine cases, according to the capabilities of the virtual machine to process assignment requests, first pre-assign a maximum number of user requests. When the requested tasks have reached the maximum number of the virtual machine, it will no longer continue to receive tasks.

Here, the cloud computing data center configuration into two kinds when testing the balancing algorithm performance: one is cloud computing data center CDC1 which is constituted by the data center DC1, and the other is the cloud computing data center CDC2. The algorithm is slightly better than the other three load balancing algorithms, which is effective and feasible [19]. This design used Round-Robin algorithm by default, requests are distributed in order across the list of servers.
Load balancing is the process of distributing a workload evenly across multiple servers. In the case of a web application, HTTP requests are load balanced across a pool of application servers. There are two main benefits to load balancing. One is to scale out and handle more users than you can with a single server. The second is redundancy if one server fails, others are available to ensure the application stays online [20].

Figure 4. Scale out your applications with NGINX and NGINX Plus load balancing

Both the open source NGINX software and NGINX Plus can load balance HTTP, TCP, and UDP traffic. NGINX Plus extends open source NGINX with enterprise-grade load balancing that includes session persistence, active health checks, on-the-fly reconfiguration of load-balanced server groups without a server restart, and additional metrics.

Table 1 describes detailed system specifications that have been created and implemented and show the server name, application name, port installation path and the service required to perform its functionality on each server.

<table>
<thead>
<tr>
<th>SERVER NAME</th>
<th>APPLICATIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>App Name</strong></td>
</tr>
<tr>
<td>Load Balancer (Frontend)</td>
<td>1. DeviceHive Docker</td>
</tr>
<tr>
<td></td>
<td>2. Nginx</td>
</tr>
<tr>
<td></td>
<td>3. DeviceHive Admin</td>
</tr>
<tr>
<td></td>
<td>4. HAProxy</td>
</tr>
<tr>
<td>Web Server 1 (moco1)</td>
<td>1. Tomcat 7</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Web Server 2 (moco2)</td>
<td>1. Tomcat 7</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Web Server 3 (moco3)</td>
<td>1. Tomcat 7</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Database Server (Backend)</td>
<td>1. PostgreSQL</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1. Specification of Load Balancer Server

4. Results

4.1. **HAProxy Install Setup on Linux Ubuntu 16.04:**

a. Install HAProxy:
   
   # apt-get install haproxy

b. Make new config:

   # nano /etc/haproxy/haproxy.cfg
# Define frontend

# Backend Server Setup

4.2. DeviceHive – Docker Install on Linux Ubuntu 16.04:

4.2.1. Install Docker

a. Install docker:
   
   ```
   $ sudo apt-get install docker-engine
   ```

b. Start docker daemon:
   
   ```
   $ sudo service docker start
   ```

c. Check that the docker is already running:
   
   ```
   $ sudo docker run hello-world
   ```

4.2.2. Install Docker-Compose

1. Change the docker-compose into an executable by this way:
   
   ```
   $ sudo chmod +x /usr/local/bin/docker-compose
   ```

2. To check if the docker-compose is installed correctly, run this command:
   
   ```
   $ docker-compose --version
   ```

   Jika sukses maka akan keluar output :

4.2.3. Install DeviceHive (Docker)

   ```
   # docker-compose up -d
   ```

   (for running background)

   | 192.168.4.165:8030/api/rest/info |

   If successful, the display will appear as follows:

4.2.4. DeviceHive admin Console :

a. Install nginx:
   
   ```
   $ sudo apt-get install nginx
   ```

b. Install dhadmin and create soft link to folder sites-enabled :

```
Design and Implementation of Load Balancing Technology on Linux-Based Servers

$ sudo ln -s /etc/nginx/sites-available/dhadmin
/etc/nginx/sites-enabled/dhadmin
Akses : 192.168.xx.xx:8031/admin

4.3. PostgreSQL 9.3 Ubuntu 16.04 Xenial
   a. Install PostgreSQL 9.3:
      # apt-get install postgresql-9.3 postgresql-9.3-contrib
      # apt-get
   b. move data directory Postgresql
   c. PostGIS Install:
      # apt-get install -y postgis postgresql-9.3-postgis-2.2

4.4. Tomcat 7 Install dan Config on the Server Linux Ubuntu 16.04

There are 3 Web server will install as follows:
Web Server 1 : IP 192.168.x.xxx
Web Server 2 : IP 192.168.x.xxx
Web Server 3 : IP 192.168.x.xxx
# sudo apt-get install tomcat7 tomcat7-admin

Start Aggregate Web Apps form Open Data Kit:

Accessing aggregate with Domain:

Web based display to control and manage Load Balancer Server.
5. Conclusion

Server for Open Data Kit Platform is often have to deal with big data that are stored in the cloud. The modular Docker container repository paradigm is particularly well suited for building cloud based applications. The primary purpose of this manuscript is to provide a proof of concept for using the Load Balancing Server method to increase performance of the server. We anticipate that Load Balancing Server will be an important step in solving the problem of receiving survey data that come from various place in Indonesia and complicated multi-component software.

In this server load balancing system, load balancer acts as a regulator in the work process performed by the system. Incoming requests on the system will be distributed to a single real server as a front end using a round robin scheduling algorithm, so the burden borne by each real server becomes relatively lighter.

Cloud Computing has extensively been used by the IT industry even though there are many existing issues like Load Balancing, Virtual Machine Migration, Server Consolidation, Energy Management, etc. These issues are yet have not been fully addressed. Load balancing is a central issue in which its required to distribute the excess dynamic workload evenly to all the nodes in the Cloud so that a high user satisfaction and resource utilization ratio can be achieved. Existing Load Balancing techniques mainly concentrate on reducing overhead, service response time and improving performance etc., but very few techniques has considered the execution time of any task at the run time. Therefore, there is a need to develop such load balancing technique that can improve the performance of cloud computing along with efficient resource utilization.

References


Design and Implementation of Load Balancing Technology on Linux-Based Servers

