International Research Journal of Advanced Engineering and Science
ISSN (Online): 2455-9024

Comparison Analysis of Apache and Nginx Webserver Load Balancing on Proxmox VE in Supporting Server Performance

Zuki Pristiantoro Putro[1], Riza Adrianti Supono[2]
[1]Software of Information System, Gunadarma University, Jakarta, Indonesia-10430
[2]Software of Information System, Gunadarma University, Jakarta, Indonesia-10430
Email address: [1]zuki.pp@universitaspertamina.ac.id, [2]adrianti@staff.gunadarma.ac.id

Abstract— *The webserver is one of the important factors so that a website can run well and serve internet needs, the right webserver used for a website system must be ensured that it can always run well, a website with high traffic can cause server performance to overload if it receives many user requests simultaneously. Simultaneously and even overall system failure, this causes server down conditions due to the death of the web server and database server. The solution to this problem is to apply load balancing techniques. Load balancing is a technology for sharing the load between multiple servers, ensuring that no one server is overloaded. webservers used for research are Apache and Nginx. Researchers tested the two web servers to find out which one is the best in completing client requests. The results showed that load balancing can work well when requests from clients have been successfully distributed evenly to each cluster node so that the server does not experience overload. The results of testing the nginx webserver are superior to Apache where the smaller the request, the better the performance of a server.*

Keywords— *Load Balancing, Load Testing, Webserver, Apache and Nginx.*

I. Introduction

Currently virtualization technology is a topic that is starting to get busy. This is inseparable from the technology's ability to reduce the cost of providing infrastructure and operations independently for each service to be served. Through this technology, a service can be configured without affecting the configuration of other services even in the same physical machine. With this technology, one physical machine is used as a shared resource that can be shared and used by several services at once. Each service can have its own operating system. Each of these services is placed in a container or containers. Because each service has its own operating system, the configuration of each service does not affect each other.

In addition, each virtual machine in each container can be turned off or on as needed (maintenance for example) without having to interfere with other services. So it can be said that the availability of a service can be more guaranteed even though there are some services that are experiencing problems or repairs. each container of a virtual machine has its own resources, those resources are allocated and managed by its hypervisor. Hypervisor is a firmware that is in charge of creating, managing, running, and monitoring a virtual machine.

Web Server is a medium for communicating with clients (web browsers) which has its own protocol, namely HTTP (HyperText Transfer Protocol). With this protocol, communication between the web server and the client (browser) can be mutually understood and easier. The standard data format on the World Wide Web is SGML. But it has become commonplace that internet users use the HTML (HyperText Markup Language) format more because its use is simpler and easier to learn.

The research applies the load balancing method to manage the distribution of workloads on the server with the aim of improving system performance. The application of load balancing on a web server is very important and is an appropriate and effective solution to handle busy server loads and is expected to increase scalability in distributed systems. The web server must be free from failures either due to hardware or software. In other cases, fault tolerance can be applied to detect and tolerate faults in real-time on distributed systems.

In this study the virtual machine will be run and will be given a load manually to simulate the load borne by the server. The results of the measurement of server load will be load balancing which in order to divide the server load is concluded in terms of running performance so that the existing system becomes well distributed. failure of access to the website, a distributed system provides a variety of resources, as one of its main advantages is that it can provide better performance and reliability than systems that use a single server to host websites. these conclusions will be used as suggestions or recommendations in the use of Proxmox VE as a managerial operating system that will handle server virtualization.

The use of multiple servers for a distributed system requires a method to manage load sharing fairly or evenly on each server. The load balancing server has the task of distributing workloads to many servers by considering the capacity of each existing server so that it can reduce the occurrence of server failures. With the implementation of many servers, when a failure occurs on one of the servers and the existing services can still run. Another thing is that the load is getting smaller, because the existing load is then distributed to each server.

The connection will be blank for a moment because at that time the browser is requesting data from the web server and that is what makes the obstacles in loading data on the website less effective and tends to hamper business processes, therefore Apache and nginx will be used to solve the problem To do this, it takes a connection between (server) and (client) that is table and quality has speed in processing requests to the server. Currently, the quality or performance of a s table connection has become a requirement that must be applied to the company's business processes.

II. Literature

1.1. *Virtualization architecture*

Server consolidation is a technique for creating multiple virtual machines on a physical server and turning them into a service [1]. Consolidation can also be interpreted as centralizing workloads, namely several virtual servers as an Infrastructure As a Service into a physical server which aims to reduce costs. Server consolidation is a technique that can be used by cloud service providers to implement server machine virtualization and as a reference for researchers in researching virtual machine migration for consolidated servers in particular [2]. Server consolidation can be used to increase the efficiency of the use of computer resources by reducing the number of servers.

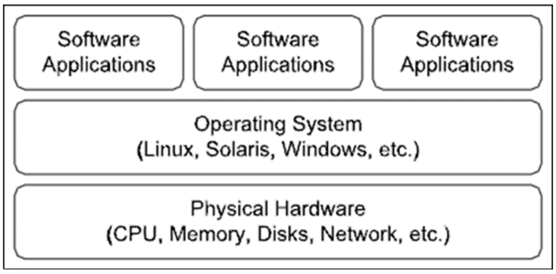

Software Applications
Software Applications
Software Applications
Operating System (Linux, Solaris, Windows, etc.)
Physical Hardware (CPU, Memory, Disks, Network, etc.)

Fig. 1. Traditional architecture

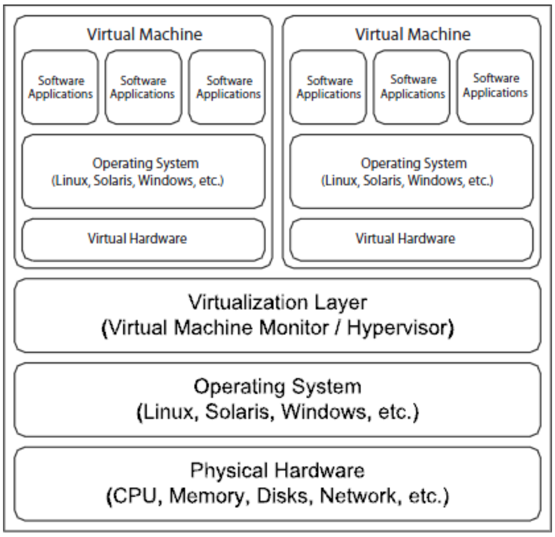

Virtual Machine
Software Applications
Software Applications
Software Applications
Operating System (Linux, Solaris, Windows, etc.)
Virtual Hardware
Virtual Machine
Software Applications
Software Applications
Software Applications
Operating System (Linux, Solaris, Windows, etc.)
Virtual Hardware
Virtualization Layer (Virtual Machine Monitor / Hypervisor)
Operating System (Linux, Solaris, Windows, etc.)
Physical Hardware (CPU, Memory, Disks, Network, etc.)

Fig. 2. Bare-metal hypervisor architecture

2.1. Advantages of Virtualization Techniques

The operational and financial advantages of virtualization are key to the enterprise computing and software development environment that virtualization can provide. The following are the main advantages of virtualization.

1. Make better use of existing hardware.
2. Reducing hardware costs
3. Reducing IT infrastructure
4. Simplify system administration
5. Easier monitoring
6. More efficient
7. Increase Uptime and Speed Up Failure Recovery
8. Simplify Capacity Expansion

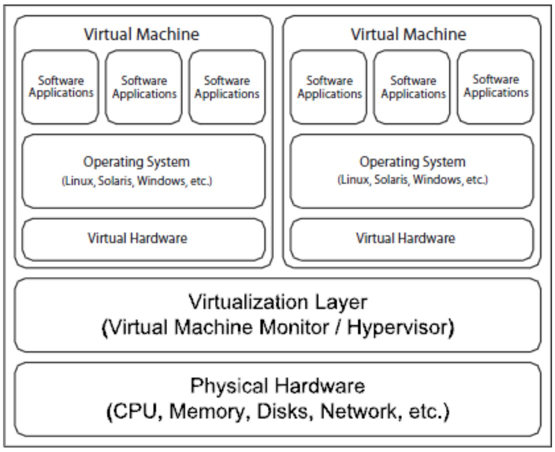

Virtual Machine
Software Applications
Software Applications
Software Applications
Operating System (Linux, Solaris, Windows, etc.)
Virtual Hardware
Virtual Machine
Software Applications
Software Applications
Software Applications
Operating System (Linux, Solaris, Windows, etc.)
Virtual Hardware
Virtualization Layer (Virtual Machine Monitor / Hypervisor)
Physical Hardware (CPU, Memory, Disks, Network, etc.)

Fig. 3. Hosted hypervisor architecture

2.2. Load Balancer

Load balancer is the process of channeling network traffic to multiple servers. This is useful in order to ensure that one of the servers is not receiving too many requests.

2.3. Load Balancer Type

In the world of load-balancing, there are two options to consider when designing a load-balancing solution. The solution option is to use Load Balancing software or Hardware Load Balancing. Each option has its own requirements, advantages and disadvantages. It is up to us to evaluate our business needs, configuration, and growth path so that we can identify the optimal solution to meet the needs. And from the type Load Balancing can be divided into 2 types:

1) *Hardware Load Balancer*

Load Balancing runs on a device that has been prepared from the factory and is ready to use. Hardware Load Balancing type is widely used because of its convenience. Some of the Load Balancing Hardware include: Cisco System Catalyst, Coyote Point, F5 Network BIG-IP, Baraccuda Load Balancer.

2) *Software Load Balancer*

Load Balancing runs on a PC/Server, and the Load Balancing application is installed and needs to be configured before it can work. The advantage is that if there are additional features or additional facilities there is no need to replace the entire Load Balancing device. The performance of the Load Balancing process is influenced by the computer device used; it cannot only rely on sophisticated software capabilities. Hardware that can affect the performance of this method is the network card (Network Interface Card) used, the amount of RAM on the device, large and fast storage media, etc. So, the performance of this method is difficult to predict. There are lots of Load Balancer Software, some of the most widely used are: Linux Virtual Server, Ultra Monkey, and Network Load Balancing.

2.4. Webserver

Webserver is software that functions as a recipient of requests sent through a browser and then provides a response to requests in the form of web site pages or more generally in HTML documents. The web server is used to store all data such as HTML documents, images, CSS stylesheets files, and JavaScript files. While on the software side, the function of the web server is as a control center for processing requests received from web browsers. because the job of the web server is to manage all communication that occurs between the browser and the server to process a website. How the webserver works as follows.

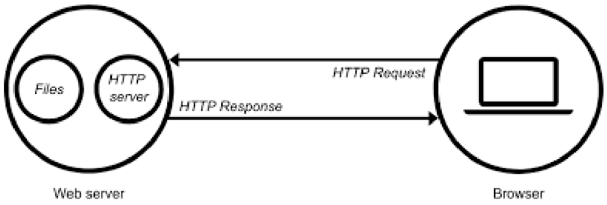


Fig. 4. Webserver

2.5. Apache

Apache is a software from a web server to manage websites that have been hosted on the server. To keep things running smoothly, the web server acts as an intermediary between the server machine and the client. The web server pulls content from the server at each user request and sends it to the web. The biggest challenge of a web server is managing many and different web users at the same time – each user requests a different page. The web server processes files written in various programming languages, such as PHP, Python, Java, and others.

2.6. Nginx

Nginx is a powerful and high-performance web server software, besides that it is also designed for servers with small resources. Nginx is designed to overcome the problem of C10K which is a web server design to handle 10,000 connections, Nginx is also able to do this through an event-based connection-handling mechanism but must use an operating system that supports this mechanism.

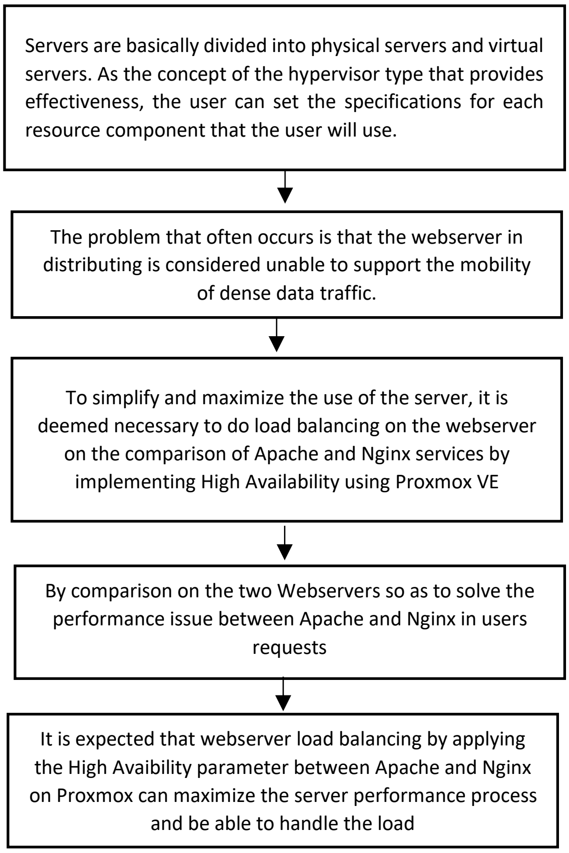
2.7. Performance Testing

Web Performance Test is a series of testing processes to measure the performance of Web Server application software. In Software Engineering, Performance Testing is a test that is carried out to determine how the system responds and is sTABLEunder certain workloads (performance degradation), so that by carrying out testing will produce data that can be used to investigate, measure, validate or verify from quality attributes of the system. With a Web Performance Test, you can easily build a test framework to be repeated that can help in analyzing the performance of a Website application and identifying potential bottlenecks. Parameters to be measured in the test :

a) Throughput
b) Connection
c) Request
d) Reply
e) Error

Where testing is done by giving a load on each attribute.

2.8. Framework



Servers are basically divided into physical servers and virtual servers. As the concept of the hypervisor type that provides effectiveness, the user can set the specifications for each resource component that the user will use.

The problem that often occurs is that the webserver in distributing is considered unable to support the mobility of dense data traffic.

To simplify and maximize the use of the server, it is deemed necessary to do load balancing on the webserver on the comparison of Apache and Nginx services by implementing High Availability using Proxmox VE

By comparison on the two Webservers so as to solve the performance issue between Apache and Nginx in users requests

It is expected that webserver load balancing by applying the High Avaibility parameter between Apache and Nginx on Proxmox can maximize the server performance process and be able to handle the load

Fig. 5. Framework

III. Research Methodology

Research uses load balancing method on the depth of data comparison between Apache and Nginx webservers. The deeper and more detailed the data obtained, the better the quality of this qualitative research. For the results of this study requires a depth of analysis from the researcher. Researchers look for problems at the research site and provide solutions to these problems, namely optimizing server performance in webserver services.

3.1. Scenario Analysis

The design stage is the stage for designing the system that is built and is also the defining stage of the system requirements analysis that describes the scenario to be tested. Scenario 1 still applies the HTTP/S protocol in communicating between the server and client. In scenario 1, it is applied to 2 different web servers, namely using Apache and Nginx. Figure 6 shows the design of scenario 1.

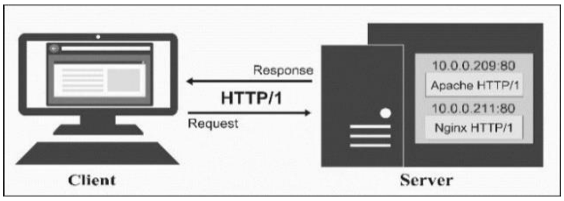


Fig. 6. Scenario 1

This scenario is an extension of the first scenario, which means the HTTP/2 protocol has been enabled. Similar to the first scenario, this scenario also has two different conditions, namely using the apache web server and the nginx web server.

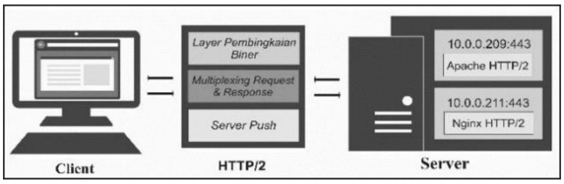


Fig. 7. Scenario 2

3.2. *Requirement Analysis*

The second stage of requirements analysis is planning the system that is made and determining the hardware and software used in this research. The hardware used is a direct device in order to find out server resources that are already running, indicating the hardware needs in building the system.

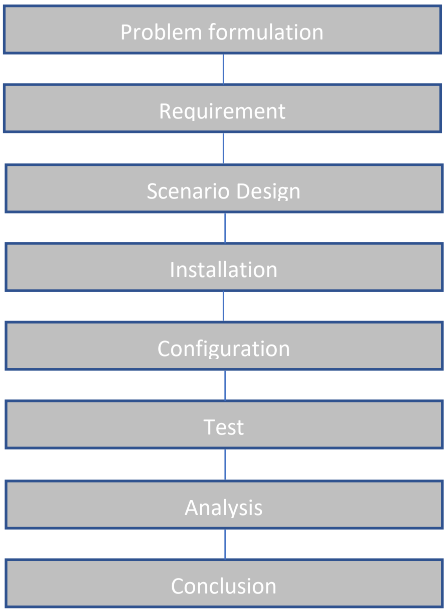


Fig. 8. Research stages

IV. Results and Discussion

Based on the diagram of the course of the research, to perform a performance analysis on the web server, it is carried out with the following steps:

4.1. *Requirement*

TABLE I. Hardware

N	Hardware	Specification
1	PC Server	Intel *Core* i5-4500U CPU @1.80GHz (4 CPUs), ~2.4GHz
		Windows 10 Educatoin 64-bit
		Memory 8192MB RAM
		Hardisk 500GB
2	Laptop	Intel *Core* i5 CPU @1.80GHz (8 CPUs), ~1.2GHz
		Windows 10 Educatoin 64-bit
		Memory 8192MB RAM
		SSD 256GB

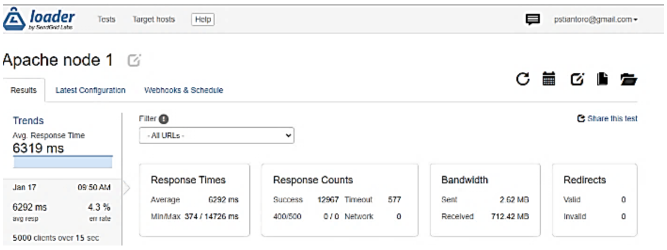
TABLE II. Software

N	Software	Specification
1	Proxmox VE 7.0-11	To create virtualization server cluster
2	*OS Linux Ubuntu 18.04 LTS*	To run the virtualized server operating system
3	Browser	As a Service that will Client Access
4	Loader.io	As a webserver testing application
5	Webserver Apache end Nginx	Application link service on server

4.2. *Webserver Load Testing Application*

1. *Loader.io*

Loader.io application is an application that is used as research for load testing on the webserver that runs the website. Load testing emphasizes the response of a system to requests. the way of testing on the type of load testing is done by doing modeling practices from the application of the system.



Apache node 1

Trends
Avg. Response Time
6319 ms
Jan 17 09:50 AM
6292 ms avg resp
4.3 % err rate
5000 clients over 15 sec

Filter: - All URLs -

Response Times
Average 6292 ms
Min/Max 374 / 14726 ms

Response Counts
Success 12967 Timeout 577
400/500 0 / 0 Network 0

Bandwidth
Sent 2.62 MB
Received 712.42 MB

Redirects
Valid 0
Invalid 0

Fig. 9. Load testing on loader.io

This stage begins with load testing on the webserver displaying test columns on the loader.io application with the registered host being toro.universitaspertamina.ac.id. then the results of the load test will be listed, depending on which webserver will be tested first, you can choose between apache or nginx. Then described in the following explanation.

4.3. *Webserver Load Balancing Testing*

Testing the webserver using Apache and Nginx is carried out load balancing with loader.io tools by providing a load of 5000 client requests in stages starting from node1 , node2 , node3 , to node4 (node = server) requests. From the request load given to the web server, then we can see how long the web server spends to complete the request

4.4. *Apache Testing*

1. *Testing Node 1 Apache*

On node1 Apache shows the results of the load-carrying test from 5000 clients within 15 seconds the result is an average response time of 6298 ms, min 374 ms and max 14726 ms then the response counts successfully reached 12967 and timeout 577, Bandwidth sent 2.65 MB recive error rate there are 43 percent.

The explanation in table III shows the results of the client test per second, Based on the measurement of time at node 1 that in testing the load for 15 seconds in each time increase per 2 seconds, the Response Time as shown in the table was carried out on node 1 on server 1 with configuration The load balancer uses a round robin algorithm on the server to take into account the effectiveness and scalability of load balancing.

TABLE III. Node 1 Apache

No	Node	Response Time (s)	Time (ms)
1	Node 1	0 – 2	1529
2		2 – 3	2262
3		3 – 4	2785
4		4 – 5	3416
5		5 – 6	3994
6		6 – 7	4732
7		7 – 8	5268
8		8 - 9	5867
9		9 – 10	6580
10		10 – 11	7381
11		11 – 12	8077
12		12 – 13	8203
13		13 – 14	7292
14		14 - 15	7907

2. *Testing node 1-2 Apache*

Node 1 and node 2 are located on node1 and node2 on Apache showing the results of load testing that from 5000 clients within 15 seconds the result is an average response time of 6357 ms, min 310 ms and max 14770 ms then response counts reach 12849 error levels are obtained 3.9 percent.

TABLE IV. Node 1-2 Apache

No	Node	Response Time (s)	Time (ms)
1	Node 1,2	0 – 2	1529
2		2 – 3	2262
3		3 – 4	2785
4		4 – 5	3416
5		5 – 6	3994
6		6 – 7	4732
7		7 – 8	5268
8		8 - 9	5867
9		9 – 10	6580
10		10 – 11	7381
11		11 – 12	8077
12		12 – 13	8203
13		13 – 14	7292
14		14 - 15	7907

In table IV it is found that the results of the test client per second, on the measurement of time at node 1 and node 2 that in testing the load for 15 seconds in each time increase per second, the Response Time shown in the table is carried out on node 1 on the server. 1 and node 2 located on server 2 are useful in calculating the level of effectiveness of server optimization.

3. *Testing Node 1-2-3 Apache*

Nodes 1, 2 and 3 are on proxmox node1 , node2 and node3 on Apache. The results of the load test show that from 5000 clients within 15 seconds the result is an average response time of 6476 ms, min 268 ms and max 14793 ms then response counts reach 12800 error obtained 3.6 percent.

Testing data results from the test client per second, when measuring node 1, 2 and 3 demand loads for 15 seconds in every increase in time per second, the Response Time shown in the table is carried out on node 1 in server 1 and node 2 on in server 2 and node 3 in server 3 with a round robin algorithm load balancer on the load testing server is needed to measure or take into account the load capacity of the webserver.

TABLE V. Node 1-2-3 Apache

No	Node	Response Time (s)	Time (ms)
1	Node 1,2,3	0 – 2	1568
2		2 – 3	2240
3		3 – 4	2847
4		4 – 5	3411
5		5 – 6	4072
6		6 – 7	4648
7		7 – 8	5446
8		8 - 9	6084
9		9 – 10	6714
10		10 – 11	7722
11		11 – 12	8175
12		12 – 13	8111
13		13 – 14	8150
14		14 - 15	8278

4. *Testing Node 1-2-3-4 Apache*

Testing Nodes 1, 2, 3 and 4 on the Apache webserver proxmox, the results of the load test show that from 5000 clients within 15 seconds the results on average response time are 6442 ms, min 426 ms and max 14791 ms later response counts reached 12640 and an error value of 3.3 percent.

TABLE VI. Node 1-2-3-4 Apache

No	Node	Response Time (s)	Time (ms)
1	Node 1,2,3,4	0 – 2	1609
2		2 – 3	2275
3		3 – 4	2783
4		4 – 5	3406
5		5 – 6	4084
6		6 – 7	4817
7		7 – 8	5339
8		8 - 9	6051
9		9 – 10	6643
10		10 – 11	7547
11		11 – 12	8051
12		12 – 13	8320
13		13 – 14	8449
14		14 - 15	8447

Testing data results from the test client per second, when measuring the time nodes 1, 2, 3 and 4 receive a request load for 15 seconds in each increase in time per second, the Response Time shown in the table is carried out on nodes 1 to 4 using 4 servers each load balancer configuration node and 1 server is used as a load balancer to measure or calculate effectiveness and scalability.

4.5. Nginx testing

1. *Testing Node 1 Nginx*

Testing node 1 nginx shows the results of carrying load testing from 5000 clients within 15 seconds, the results are an average response time of 6395 ms, min 816 ms and max 14972 ms then the response counts successfully reached 12606 and the error rate was 4.6 percent.

TABLE VII Node 1 Nginx

No	Node	Response Time (s)	Time (ms)
1	Node 1	0 – 2	1731
2		2 – 3	2349
3		3 – 4	3041
4		4 – 5	3648
5		5 – 6	4384
6		6 – 7	5078
7		7 – 8	5638
8		8 - 9	5998
9		9 – 10	6649
10		10 – 11	7474
11		11 – 12	8191
12		12 – 13	7645
13		13 – 14	8085
14		14 - 15	784

The explanation this table above shows the results of the client test per second, Based on the measurement of time at node 1 that in testing the load for 15 seconds in every increase in time per second, the Response Time as shown in the table was carried out on node 1 on server 1 with load configuration balance on the server.

2. *Testing Node 1-2 Nginx*

Node 1 and node 2 showing the results of the load test that from 5000 clients within 15 seconds the results on an average response time of 6395 ms, min 816 ms and max 14972 ms then response counts reached 12606 error obtained 4.6 percent.

TABLE VIII Node 1-2 Nginx

No	Node	Response Time (s)	Time (ms)
1	Node 1,2	0 – 2	1728
2		2 – 3	2329
3		3 – 4	3017
4		4 – 5	3654
5		5 – 6	4387
6		6 – 7	5092
7		7 – 8	5648
8		8 - 9	6017
9		9 – 10	6605
10		10 – 11	7365
11		11 – 12	8136
12		12 – 13	7756
13		13 – 14	7922
14		14 - 15	8142

The explanation in this table above shows the results of the client test per second, Based on the measurement of time at node 1 that in testing the load for 15 seconds in every increase in time per second the Response Time is obtained as in the TABLE carried out on node 1 on server 1 with load configuration balance on the server.

3. *Testing Node 1-2-3 Nginx*

Nodes 1, 2 and 3 are located on proxmox node1 , node2 and node3 on Apache. The results of the load test show that from 5000 clients within 15 seconds, the average response time is 7365 ms, min 821 ms and max 14739 ms, then response counts reach 13185 error obtained 3.9 percent.

Testing the results of the client test per second, on measuring the time of nodes 1, 2 and 3 the request load for 15 seconds in every increase in time per second the Response Time is obtained which is shown in the table, then node 1 receives a request to the next node in receiving a request. client, on the server load testing is needed to measure the effectiveness and scalability of the test node.

TABLE XI Node 1-2-3 Nginx

No	Node	Response Time (s)	Time (ms)
1	Node 1,2,3	0 – 2	1755
2		2 – 3	2343
3		3 – 4	3038
4		4 – 5	3614
5		5 – 6	4332
6		6 – 7	5047
7		7 – 8	5226
8		8 - 9	6116
9		9 – 10	6538
10		10 – 11	7325
11		11 – 12	7969
12		12 – 13	7810
13		13 – 14	7720
14		14 - 15	7535

4. *Testing Node 1-2-3-4 Nginx*

Testing Nodes 1, 2, 3 and 4 on the nginx webserver, the results of the load test show that from 5000 clients within 15 seconds the results are on average 6469 ms, 800 ms min and 14861 ms max then response counts reach 12811 and the value 3.3 percent error.

TABLE X. Node 1-2-3-4 Nginx

No	Node	Response Time (s)	Time (ms)
1	Node 1,2,3,4	0 – 2	1711
2		2 – 3	2331
3		3 – 4	3023
4		4 – 5	3645
5		5 – 6	4375
6		6 – 7	5075
7		7 – 8	6184
8		8 - 9	5584
9		9 – 10	6796
10		10 – 11	7525
11		11 – 12	8046
12		12 – 13	8098
13		13 – 14	6310
14		14 - 15	8073

The test data results from the test client per second, when measuring node 1,2,3 and 4 demand loads for 15 seconds in each time increase per 2 seconds, the Response Time shown in the table is carried out on node 1, node 2 and node 3 and node 4 with load balance configuration on the load testing server needed for the level of effectiveness and scalability in the Apache webserver load balancing test.

4.6. Implementation of Testing Analysis

The implementation stage and analysis of the results, there are several stages, namely the implementation of a load balancer on each node, there are two web servers used in the study, namely Apache and Nginx.

The node 1 scenario is run by one server and one main server to find out whether the website can respond to the Apache configuration. The node 1-2 scenario runs two server nodes 1 and 2 one as the main server, and node 1,2,3 and 4 scenarios run a website that has been configured, the website is marked with a website page according to the node, if the node is reloaded it will be redirected to the node next that looks on the website.

Evaluation of the results of testing the performance speed of the two web servers in terms of time per request Nginx is superior to Apache because the speed of requests from Nginx is faster than Apache, it is proven by the smaller number of table requests that come out of the Nginx webserver. It can be said that the smaller the resulting number, the faster the request process. In addition, the results of the request process from nginx are more than Apache. Apache web server connection times are better with a small average value per millisecond, so that the connection users make when opening websites on the Nginx web server is stable. While the Apache web server connection times the average value generated is greater than that of Nginx. And lastly, in terms of the transfer rate, Nginx has a fairly high transfer value, so that the process of transferring data or files on the Nginx webserver is faster.

4.7. Analysis of Test Results

The results above explain the stages of how the round robin algorithm works in a load balancer with a cluster system on server virtualization, here is the explanation:

The initial stage, the request to be accessed is on the server node 1, and then explains when the node 2 server is turned off, the server will move or the server will automatically move if it receives a request from the client and moves to another server in a stable state.

TABLE XI. Apache virtual machine request migration

N	Node	Migration time (s)			Average (ms)
		1	2	3	
1	N1 - N2	0.4	0.3	0.6	0.5
2	N2 - N3	0.5	0.4	0.3	0.4
3	N3 - N4	0.2	0.2	0.3	0.2

Based on the measurement of migration time when the active path is lost, the reduction of migration time on server 1 to server 2, then from server 2 to server 3 if downtime is directed to server 4 and migrated back (looping) to server 1 until the distribution per request is evenly distributed as the result virtual machine displacement in table XI.

TABLE XII. Nginx virtual machine request migration.

N	Node	Migration time (s)			Average (ms)
		1	2	3	
1	N1 - N2	0.4	0.3	0.6	0.5
2	N2 - N3	0.5	0.4	0.3	0.4
3	N3 - N4	0.2	0.2	0.3	0.2

Based on the measurement of the migration time of virtual machine node 1 to virtual machine 2, then from virtual machine 2 to virtual machine 3 from virtual machine 3 to virtual machine 4 like that (looping) to virtual machine 1 until the distribution per request is evenly distributed because it uses a round robin algorithm, with the virtual machine results in this table.

The steps taken to build a High Availability Server Clustering is to install or implement load balancing on the Webserver in the form of Apache or Nginx services. In the load request, the main server load balancing, the round robin algorithm method is applied to ensure the availability of services for running business process activities so as to reduce or minimize the use of software and hardware resources.

V. Conclusion

5.1. Summary

After testing and analyzing the Apache and Nginx web servers with the parameters of transfer rate, time per request, and connection time as measurement indicators for testing the performance of the two web servers.

1. The comparison of throughput testing between Apache and nginx webserver software four times with the load balancing test method, the results obtained where the results of nginx throughput have a performance that is superior to Apache's throughput performance, where the resulting bandwidth is much better so that it can accommodate a lot of data.
2. Comparison of reply section testing between Apache and nginx webserver software, Apache performance is better than nginx because the reply time is smaller, the smaller the reply time, the faster data can be transferred.
3. In testing the error section on the performance of the Apache and Nginx webserver software, the performance is equally good because the average error rate is the same during the testing of both Apache and Nginx server software.
4. Comparison of requests between Apache and nginx webserver software has almost the same load balancing performance depending on the ability of the client to access the webserver.
5. The Connection Section (Time) test conducted between the Apache and nginx webservers found that nginx is superior in connecting the Client - Server and Server to the Client where the smaller the connection, the better the performance of a server.

5.2. Suggestion

Criticisms and suggestions from the results of this study can be studied in several ways in carrying out the development of the use of a webserver so that it becomes an evaluation material so that the application at the institution becomes better. Criticisms and suggestions as follows:

1. This VPS (Virtual Private Server) system, especially on the web server, focuses on discussions about testing data transfer speeds, request and connection times only and has not paid attention to other aspects. Therefore, the researcher suggests that this system can be developed again in the future.
2. Based on the test results obtained in this study, it is hoped that it can provide input as to which webserver choice to be used later in building a website system that will be used if the website will handle many client requests. Then based on the results obtained, this research can be continued by testing the performance of the two webservers from other aspects such as concurrency level and the number of URLs accessed at the same time.
3. For better and more flexible capabilities, you should use the nginx webserver due to faster performance in transferring and displaying data, but if you want to use it to display a lot of data, you should use the Apache http server software because of its ability to accommodate a lot of data.

4. The author hopes that in the future they can use better methods in testing, using a Web Server software because the development of science, especially in the IT field is very rapid and of course has better capabilities than before and can be tested again later as a comparison for a later date.

REFERENCES

[1]. Deni Marta, Guntoro, Analisis Perbandingan Performa Virtualisasi Server Sebagai Basis Layanan Infrastructure As A Service pada Jaringan Cloud. Jurnal Matrik, Vol.19, Hal 1-8, 2019.

[2]. Sampurna Dadi, Donaya Pasha, Analisis Perbandingan Server Load Balancing dengan Haproxy & Nginx dalam Mendukung Kinerja Server E-Learning. Jurnal Telekomunikasi Dan Komputer, Vol.1, Hal 135-144, 2021

[3]. Saleh Dwiyatno, Edy Rakhmat, Implementasi Virtualisasi Server Berbasis Docker Container. Jurnal PROSISKO, Vol. 7, 2020.

[4]. Dimara Kusuma, Johan Riyanto, (2019). Pengujian Algoritma Load Balancing pada Virtualisasi Server. Jurnal Sianteks, Vol 16, Hal 33-41, 2019.

[5]. Albert Yakobus Chandra, Analisis Performansi Antara Apache & Nginx Web Server Dalam Menangani Client Request. Jurnal Sistem dan Informatika, Vol 14, Hal 48-56, 2019.

[6]. Revan FA, Doan Perdana, Ridha MN, (2019). Analisis Throughput Dan Skalabilitas Virtualized Network Function VyOS Pada Hypervisor VMWare ESXi, XEN, DAN KVM. Jurnal Infotel, Vol 9, 2017.

[7]. Aishiyah., Andryanto, Perancangan Virtual Private Server Menggunakan Proxmox Pada Amik Ibnu Khaldun Palopo. Jurnal Simantik, Hal 19-242, 2017.

[8]. Dessyana K., Anton RP, Pengembangan Virtual Server Dengan Proxmox Ve 6.2 Sebagai Cloud Computing Berbasis Free/Open Source Software. Jurnal Ilmiah Stmik AUB, Vol.26, Hal 85-95, 2020.

[9]. Abdul Azi., Adita Tarkono., (2016). Teknologi Virtualisasi Openvz dan Kernel-Based Virtual Machine sebagai Layanan IaaS pada Jaringan Enterprise. Jurnal Multinetics Vol. 2, 2016.

[10]. Totok Wahyu, Arief (2020). Analisa Dan Perbandingan Performa Hypervisor ESXi, XEN, VMWARE Workstation Pro, Dan VirtualBox. Jurnal INFORMA Politeknik Indonusa Surakarta, Vol. 6 No 2, p-ISSN : 2442-7942, e-ISSN 2716-5051, 2020.

[11]. Abdul Azi., Adita Tarkono, Teknologi Virtualisasi Openvz dan Kernel-Based Virtual Machine sebagai Layanan IaaS pada Jaringan Enterprise. Jurnal Multinetics, Vol. 2 No.1, 2016.

[12]. Dessyana K., Anton RP, Pengembangan Virtual Server Dengan Proxmox Ve 6.2 Sebagai Cloud Computing Berbasis Free/Open Source Software. Jurnal Ilmiah Stmik AUB, Vol.26, Hal 85-95, 2020.

[13]. Abdul Azi., Adita Tarkono., (2016). Teknologi Virtualisasi Openvz dan Kernel-Based Virtual Machine sebagai Layanan IaaS pada Jaringan Enterprise. Jurnal Multinetics Vol. 2, 2016.

[14]. Totok Wahyu, Arief (2020). Analisa Dan Perbandingan Performa Hypervisor ESXi, XEN, VMWARE Workstation Pro, Dan VirtualBox. Jurnal INFORMA Politeknik Indonusa Surakarta, Vol. 6 No 2, p-ISSN : 2442-7942, e-ISSN 2716-5051, 2020.

[15]. Abdul Azi., Adita Tarkono, Teknologi Virtualisasi Openvz dan Kernel-Based Virtual Machine sebagai Layanan IaaS pada Jaringan Enterprise. Jurnal Multinetics, Vol. 2 No.1, 2016.

[16]. Eko Wiyatnanto, Arief Indriarto Haris, Kinerja Arsitektur Interoperabilitas Menggunakan Government Service Bus (GSB) dan Peer to Peer (P2P). Jurnal Ultimatics, Vol 13, Hal 7-11. 2021.

[17]. Ginasaria, Wibawa, Wirdiania, Pengujian Stress Testing API Sistem Pelayanan dengan Apache JMeter. Jurnal Ilmiah Teknologi dan Komputer. Vol. 2, 2021

[18]. Irza1, Zulhendra, Efrizon, Analisis Perbandingan Kinerja Web Server Apache dan Nginx Menggunakan Httperf Pada Portal Berita. Jurnal Vokasional Teknik Elektronika & Informatika. Vol. 5, 2017.

[19]. Lia Umaroh, Machsun Rifauddin,. (2020). Implementasivirtual Private Network (Vpn) Di Perpustakaan Universitas Islam Malang. Jurnal Dokumentasi Dan Informasi. Vol 4, 2020.

[20]. Nasser, Wotono,. (2016). Analisis Algoritma Round Robin, Least Connection, Dan Ratio Pada Load Balancng Menggunakan Opnet Modeler. Jurnal INFORMATIKA, Vol. 12, 2016.

[21]. Dan Ratio Pada Load Balancng Menggunakan Opnet Modeler. Jurnal INFORMATIKA, Vol. 12, 2016.

[22]. Yudha, Wing, Eko, Analisis Teknologi Virtual Mesin Proxmox Dalam Rangka Persiapan Infrastruktur Server. Jurnal Informa. Vol 5 No 3, 2020.