ATCSJ
Applied Technology and Computing Science Journal

Open Access

p-ISSN 2621-4458 | e-ISSN 2621-4474

Available online to journal2.unusa.ac.id

UNUSA

S4-Accredited – SK No.200/M/KPT/2021

Journal Page is available to https://journal2.unusa.ac.id/index.php/ATCSJ

Load Balancing Analysis Using Round-Robin and Least-Connection Algorithms for Server Service Response Time

Trinugi Wira Harjanti[1], Hari Setiyani[2], Joko Trianto[3]

[1-3]*Teknik Informatika, Sekolah Tinggi Teknologi Informasi NIIT, Indonesia*

Jl. Asem Dua No.22, Cipete Selatan, Kec. Cilandak, Kota Jakarta Selatan, Daerah Khusus Ibukota Jakarta

[1*]trinugi@i-tech.ac.id, [2]hari.setiyani@i-tech.ac.id, [3]joko.trianto@i-tech.ac.id

Article history:

Received 27 December 2022
Revised 1 January 2023
Accepted 1 January 2023
Available online 3 January 2023

Keywords:

Load Balancing,
Least-Connection,
Response Time,
Round-Robin,
Server Service.

Abstract

Today's internet network is expanding quickly, which encourages consumers to connect to more server services. A reliable server system is required to handle these circumstances. One approach that can be taken is to implement multiple servers. However, using many servers affects the response time of the server to be able to serve requests. Improving server system services through the load balancing method needs to be balanced with analysis, testing and evaluation of the network architecture and algorithms used in order to produce an optimal server system. In order to determine the algorithm to be tested optimal response time value, this study compares and analyses the load balancing approach utilizing the least-connection algorithm and the round-robin algorithm. Based on the test results using a request value of 500 connections/second for 1000 requests and 600 connections/second for 1200 requests, the round-robin algorithm looks slightly better than the least-connection algorithm. However, in the test scenario for the value of 700 connections/second for 1400 requests, 800 connections/second for 1600 requests, and 900 connections/second for 1800 requests, the least-connection algorithm looks superior. So that the average response time of the least-connection algorithm is smaller when there is an increase in connections. This condition has an impact on the faster the server responds to requests from users so that server performance can be continuously improved.

I. INTRODUCTION

The rapid growth of the internet network today, encourages an increasing number of users for the various services provided through the website. This situation can lead to an increase in the burden on service server providers in line with the increase in the number of clients [1]. This of course will require maximum and powerful server equipment. In addition, various Web application services are increasing which trigger access to internet services. These services include e-business, e-marketplace, e-commerce, e-marketing and other internet services [2], [3]. This of course will require maximum and powerful server equipment. In

[1] Corresponding author

ISSN 2621-4474 (online) 2621-4458 (print) © 2022 The Authors. Published by UNUSA

DOI: https://doi.org/10.33086/atcsj.v5i2.3743.

addition, various Web application services are increasing which trigger access to internet services. These services include e-business, e-marketplace, e-commerce, e-marketing and other internet services [4], [5]. The presence of cloud computing technology will make it easier to design infrastructure [6]. Quite a lot of research has used cloud computing as an option to provide infrastructure resources when setting up various web-based services. Therefore, to improve the performance of web server system services, the best server architecture is needed [7]. So, with optimal server development requests for access to a high number of servers can be handled properly. A good server system can support the availability of services according to needs. Server architecture that is supported by many servers is the best solution to be implemented. Load balancing is one of several network technology approaches that have the ability to provide increased performance on server services. The load balancing procedure is through distributing the workload received jointly on each server or by dismantling one server [8], [9].

Improving server system services through the load balancing method needs to be balanced with analysis, testing and evaluation of the network architecture and algorithms used in order to produce an optimal server system. Therefore, parameters such as response time can be a problem that must be resolved in an effort to build a good server system. Another option is to implement a server cluster model to deal with the overload associated with increased network traffic. A server cluster is a collection of many computing devices that are connected and work together in different ways. Therefore, server clusters can be interpreted as a unified system. Server clusters are usually intended to ensure availability and improve system performance [10], [11]. By implementing the server cluster model, it will be able to increase system availability and system reliability [12], [13]. Load balancing provides distribution to request incoming data to all servers and computers to make the most of available resources, reduce response time, increase throughput, and reduce demand congestion.

Server load balancing provides support for many functions and can also play a regulatory role and server traffic. Load balancing can also be used to assess the state of applications and content on servers, increasing available services and simplifying management [11], [14]. So, we need the right scheduling algorithm to support the implementation of load balancing. Of course, this becomes a serious problem if the number of clients accessing the web server increases and the server is unable to handle it. Often the cause is overload which causes existing services to have a long response time. There are a number of sites that are even said to experience thousands of connections from clients simultaneously [15], [16]. Termination of service is the result of server failure providing service. There are several algorithms that can be used in load balancing, including the round-robin algorithm and the least-connection algorithm. These two algorithms have different ways of working, the round-robin algorithm provides a queue process alternately and the least-connection algorithm works by dividing the load based on the number of connections served. Analysis and testing are needed to get the right algorithm for implementing load balancing so that the right architecture can be determined to overcome the response time on server services. For this reason, the main objective of this research is to improve server services through the use of a load balancing approach by conducting a comparative analysis of round-robin and least-connection algorithms to get the best algorithm in overcoming response time problems. Through the implementation of the scheduling method on the server, large incoming loads are distributed to each service provider server so that the response time will be faster.

II. METHODS

Research methods are needed to conduct research so that it can run well through the preparation of research stages or workflows in conducting research. The phases of conducting research that are organized in a structured and planned way to accomplish the study objectives are contained in the stages of the

research [17]. This study compares the round-robin algorithm with the least-connection on the load balancing method in an effort to improve server services in overcoming response times. Figure 1 depicts the process or phases of the research that was done.

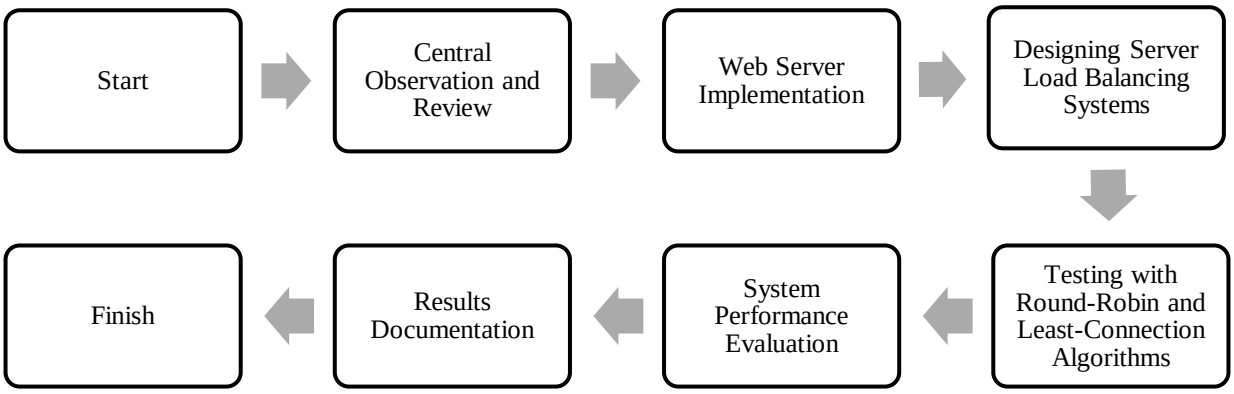

Figure 1. Research Stages

In Figure 1, shows the various phases that must be followed while carrying out this research. This test is carried out using a load balancing approach that is used in distributing a large number of connections on each web server and testing the conditions before and after applying the load balancing method as a measuring tool in improving server performance. In the early stages of research, a literature study was first carried out. The next step is to create a web server. Once the web server is created, proceed to create a load balancing server. While performing performance measurements, the web application server is gradually granted more and more connections. Then, the connection is routed to a collection of web servers that are integrated into the backend server. The next step is to test the load balancing approach using round-robin and least-connection algorithms. Testing is carried out by applying various test scenarios. The results obtained from the testing are evaluated to determine the level and role of load balancing in system performance, especially in response time.

A. Load Balancing

Load balancing is an approach model in the field of computer networking that aims to provide distribution of received connection loads to a number of computers or computer clusters in order to get the maximum benefit from resources through optimizing throughput values, response time and avoiding overload. A computer cluster is composed of a number of interconnected PC devices to do certain jobs. For this reason, computer clusters are considered as a network that has various aspects. In addition, this computer cluster is generally used to increase performance and the availability of data collected from a number of computers [18]. Apart from being a load balancer, load balancing can perform a number of functions such as traffic engineering and intelligent traffic switching. Load balancing can monitor the health of servers, systems and content to improve service availability and management [12].

In building load balancing there are various software that can be used, one of which is HAProxy. In addition, Haproxy can also redirect process failures to other nodes provided on a computer cluster.

HAProxy runs on the Linux operating system and is licensed in an open source manner so that the software can be used freely and developers can develop it [13].

When performing load balancing, a load balancing requires an approach at the distribution stage, which is known as the scheduling method. There are 8 scheduling algorithms in HAProxy that are used to configure server systems, including: RDP Cookie, URL_parameter, Header Name, Static Round-robin, Round-robin, Least-connection and Source, URI (Uniform Resource Identifier). To configure HAProxy, several components are required, such as: Backend Settings, Fronted Settings, Default Settings and Global settings. The scheduling algorithm configuration that will be applied to HAProxy is located in the Backend Setting component. In this study the algorithm to be used is a round-robin algorithm with the least-connection.

B. Round-robin Algorithm

Round-robin is one of the algorithms for scheduling or scheduling that works by distributing the load evenly to each backend server in the group. This algorithm works by evenly distributing incoming connections to each group of actual backend servers. The round-robin runs its processes through all the required server nodes in accordance with the load assignment by each server. Figure 2 below shows the round-robin algorithm process.

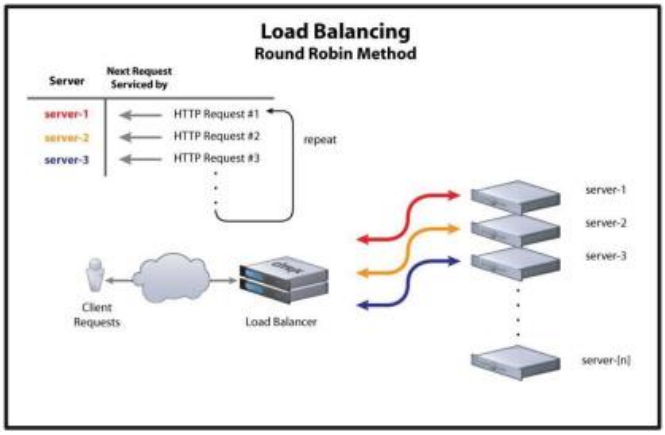

Figure 2. Round-Robin Algorithm Workflow

The main concept of this algorithm is through the use of time sharing, the focus is to provide a queue process by alternating [19]. The Round-robin algorithm does not give permission to dynamically switch loads because everything is statically determined at the start [20]. There is no limit to the number of active servers that are positioned as backend.

C. Least-connection Algorithm

Least-connection is a scheduling algorithm that works by distributing more requests to real servers by connecting active servers with fewer portions. Not only that, the Least-connection algorithm considers that all backend servers have equal computational performance [12]. In essence, the least-connection algorithm works by dividing the load based on the number of connections served by a server. The least-connection algorithm process is shown in Figure 3.

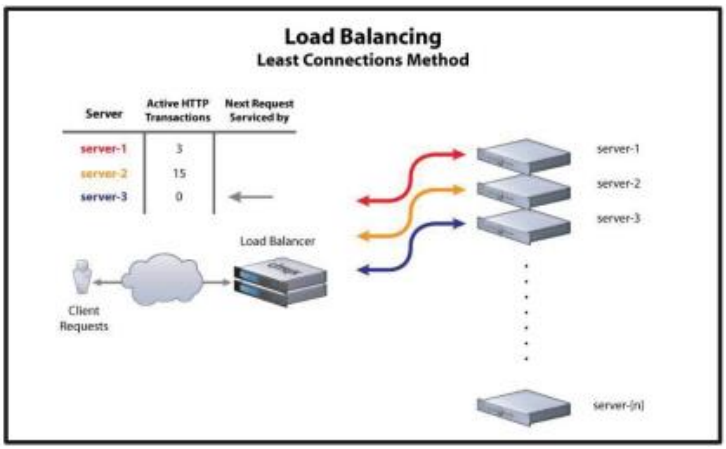

Figure 3. Least-Connection Algorithm Workflow

The Least-connection algorithm is very suitable when applied to clusters that have dynamic conditions with sessions that often experience changes [19]. The Least-connection algorithm can be used via the balance leastconn command in the backend server settings.

D. Response Time

Response time is known as response time, which is the total time it takes the server from receiving a request to receiving the answer [12]. In research conducted response time is used to measure the speed of web applications through load balancing using round-robin and least-connection algorithms in response to requests coming from users.

E. Load balancing Test Scenario

There are several scenarios in this research in testing the algorithm on load balancing. This test scenario is run for each algorithm. The aim is to see the performance of the round-robin and least-connection algorithms on a load balancing system. Testing is done by generating a number of connections. Connection generation will be carried out in stages, starting from 1000/500, 1200/600, 1400/700, 1600/800 and 1800/900 connections to obtain parameter values from response time.

III. RESULTS AND DISCUSSIONS

To test a set of load balancing parameters on the server, it is intended that the performance of the server system can be measured when it receives a large number of incoming requests from clients based on a specific timeframe. To test the server system, the variable to be measured is response time. The response time measurement is carried out before and after floating the load balancer. When building a load balancing system, virtual machines are used on the main computer to develop server infrastructure. The architecture used consists of two designs which will be evaluated later. The first model is a blueprint for a single server architecture, but that architecture does not implement load balancing techniques. While the second model uses a load balancing architecture. In this architecture, only one server needs to handle requests from users. Therefore, when many requests come, the request is sent to the server. The second architecture then implements a set of servers dedicated to processing incoming requests, which can be seen in Figure 4. The server system architecture shown in Figure 4 uses the concept of a load balancing mechanism that was created. This architectural design consists of a load balancing server and two web servers. Incoming requests from clients are not processed by one server, but by two or more servers together.

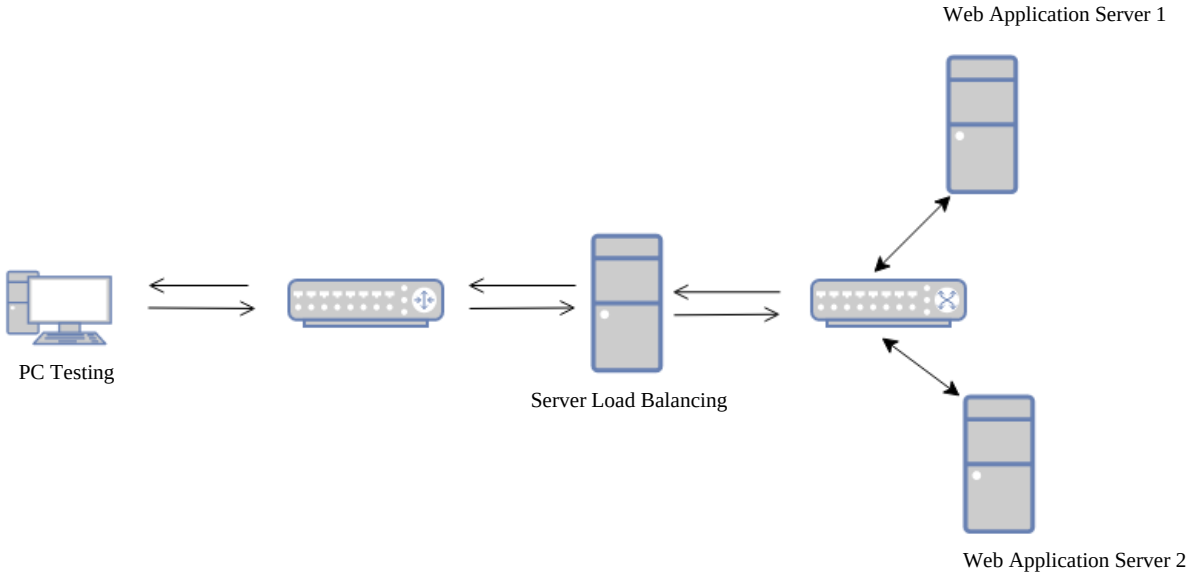

Figure 4. Server System Models

Figure 4 shows the design of a system device consisting of load balancing, Web-Application server1, Web-Application server2, and test computers that are connected to each other. Table 1 contains information about configuration of device names and IP addresses.

Table 1. Device Name and IP Address

Device Name	IP address
Testing Computer	192.168.109.182
Web Application 1	192.168.109.179
Web Application 2	192.168.109.178
Load balancing servers	192.168.109.184

Based on Table 1, it can be seen that the IP address of the load balancing server is 192.168.109.184, so that every request will always be directed to that IP. Server load balancing has the ability to manage incoming services from clients to web servers. When a client connects to a web server to obtain service, the client automatically redirects to the load balancing server. the request is then forwarded to each web server located on the backend. Therefore, the client will not know which web server is receiving and is responsible for processing incoming requests. The tests carried out include observing the response time value that can be measured. This aims to measure the performance of services provided by web servers when server load balancing is used. At the testing stage, the client uses httperf software to make connections simultaneously to measure the performance of the system through the obtained response time values. The results of the tests performed are presented in Table 2 for the round-robin algorithm and Table 3 for the least-connection algorithm.

Table 2. The Results of The Response Time (ms) Test for The Round-Robin Algorithm

No	Response Time (ms) with Round-Robin Algorithm						
	Connection	Test-1	Test-2	Test-3	Test-4	Test-5	Test Average Results

1	1000/500	3.7	3	2.9	3.8	2.8	3.24
2	1200/600	3.5	3.4	3.4	3.4	3.5	3.44
3	1400/700	3.9	4.9	4.2	5.7	4.1	4.56
4	1600/800	5.1	4.3	5.4	5.6	8.5	5.78
5	1800/900	6.9	12.8	8.7	10.3	11.3	10

Table 3. Response Time (ms) Test Results for The Least-Connection Algorithm

No	**Response Time (ms) with Least-Connection Algorithm**						
	Connection	**Test-1**	**Test-2**	**Test-3**	**Test-4**	**Test-5**	**Test Average Results**
1	1000/500	3.7	3.5	2.7	3.6	3.4	3.38
2	1200/600	3.5	3.5	3.4	3.7	3.6	3.54
3	1400/700	4	3.9	4	4.1	3.9	3.98
4	1600/800	4.4	5.4	6.3	5.4	5.2	5.34
5	1800/900	9.1	8.4	7.8	10.3	10.7	9.26

Table 2 and Table 3 show information related to the server load balancing test results through a series of connections from the client to the web server using the httperf software. Connection requests from the client to the web server are executed in stages. The first scenario starts with 1000 requests at 500 connections/sec. Then continue with 1200 requests at 600 connections/sec. Continuing 1400 requests at 700 connections/sec. Followed by 1600 requests at 800 connections/second, and finally 1800 requests at 900 connections/second, which are gradually directed to the web server application. Test results will be different for each test scenario. This condition occurs because the duration of each test scenario is long. This creates unequal queues between scenarios for each request to be processed. This situation affects the response time value, because the response time value increases as the demand for web server services increases. Based on the response time value obtained from the test results, load balancing performance can be identified by comparing the two algorithms. For more details, the results of the comparison between the round-robin and least-connection algorithms for testing responses time are presented in Table 4 and are visualized in graphical form in Figure 5.

Table 4. The Average Response Time (ms) Test Results for The Round-Robin and Least-Connection Algorithms

Connection	**Response Time (ms)**	
	Round-Robin Algorithm	**Least-Connection Algorithm**
1000/500	3.24	3.38
1200/600	3.44	3.54
1400/700	4.56	3.98
1600/800	5.78	5.34
1800/900	10	9.26

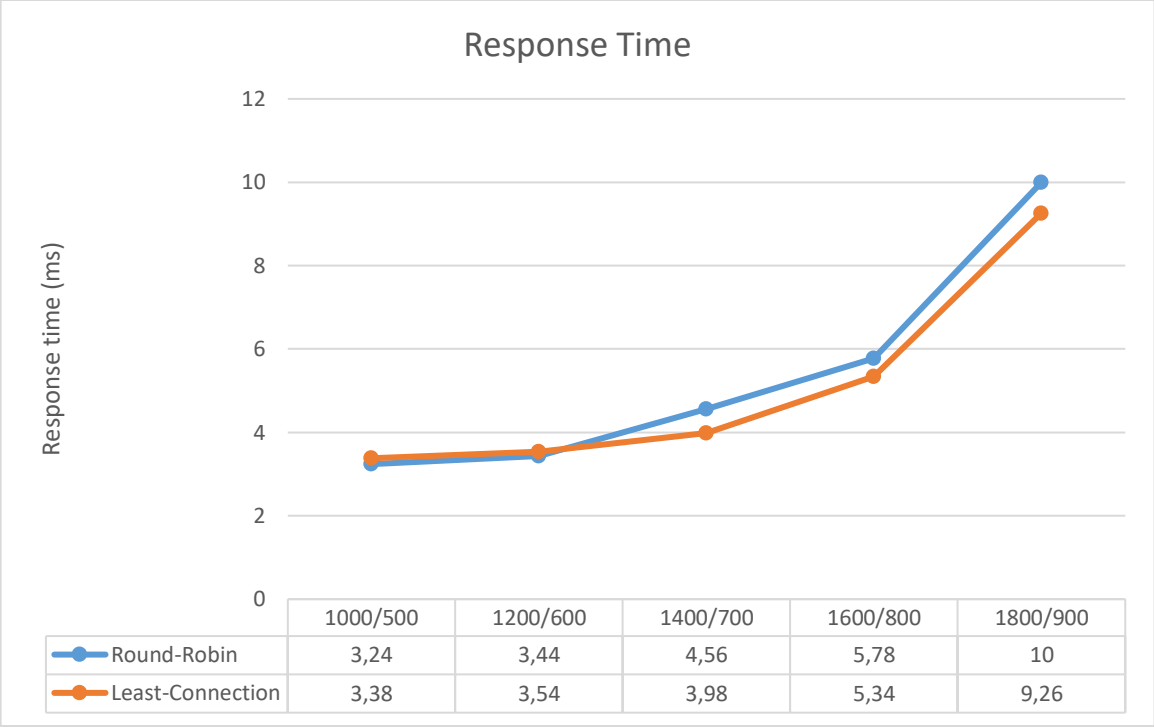

Figure 5. The Result of The Response Time Value Between the Round-Robin Algorithm and the Least-Connection

Figure 5 shows a comparison graph between the round-robin and least-connection algorithms. This condition can be seen from the test results based on the response time obtained. Table 4 shows the average values when testing the load balancing architecture model with round-robin and least-connection algorithms. The test results show that applying load balancing with the least-connection algorithm results in a much smaller response time for an increasing number of connections compared to the round-robin algorithm. Tests are run at different connection ranges and times. The load balancing with this round-robin technique appears a little better in connection testing with a request value of 500 connections/second for 1000 requests, and 600 connections/second for 1200 requests. The inequitable distribution, which each server may still manage, is what leads to this situation. However, there was a shift in the capacity to respond to incoming requests during the test time of 700 connections/second for 1400 requests, 800 connections/second for 1600 requests, and 900 connections/second for 1800 requests. When compared to load balancing with the round-robin algorithm, the least-connection algorithm has a faster reaction time. This happens because the server will receive the next task when the previous task was successfully completed faster, so that efficiency occurs.

IV. CONCLUSIONS AND RECOMMENDATIONS

The application of the round-robin and least-connection algorithms on the load balancing server has been successfully implemented and from the results of the load balancing test it can distribute requests to each web server. According to the results of testing the connection with 1000 requests at 500 connections per second and 1200 requests at 600 connections per second, load balancing using the round-robin algorithm appears to be marginally superior to the least-connection method. The inequitable distribution, which each server may still manage, is what leads to this situation. The ability to respond to incoming requests changed

during the test time, making load balancing with the least-connection technique appear superior for values of 700 connections/second for 1400 requests, 800 connections/second for 1600 requests, and 900 connections/second for 1800 requests. These findings unquestionably have a substantial impact on how quickly the server responds to requests. For future development, load balancing servers should be reserved through increasing the number of servers running. This is useful for providing support to the fault tolerance model needed to increase service providers in server systems. For further research, you can analyze other load balancing methods and measure the performance of other attributes, for example throughput.

V. REFERENCES

[1] H. Gupta and K. Sahu, "Honey Bee Behavior Based Load Balancing of Tasks in Cloud Computing," *Int. J. Sci. Res.*, vol. 3, no. 6, pp. 842–846, 2014, doi: http://dx.doi.org/10.1016/j.asoc.2013.01.025.

[2] S. D. Riskiono and D. Pasha, "Analisis Metode Load Balancing Dalam Meningkatkan Kinerja Website E-Learning," *J. Teknoinfo*, vol. 14, no. 1, p. 22, 2020, doi: 10.33365/jti.v14i1.466.

[3] N. Angsar, "Pengujian Distribusi Beban Web dengan Algoritma Least Connection dan Weighted Least Connection," *Jnteti*, vol. 3, no. 1, pp. 24–28, 2014.

[4] S. Suresh and S. Sakthivel, "A novel performance constrained power management framework for cloud computing using an adaptive node scaling approach," *Comput. Electr. Eng.*, vol. 60, pp. 30–44, 2017, doi: 10.1016/j.compeleceng.2017.04.018.

[5] Y. Afrianto and A. H. Hendrawan, "Implementasi Data Center Untuk Penempatan Host Server Berbasis Private Cloud Computing," *Krea-Tif*, vol. 7, no. 1, p. 50, 2019, doi: 10.32832/kreatif.v7i1.2031.

[6] D. E. Kurniawan, M. Iqbal, J. Friadi, R. I. Borman, and R. Rinaldi, "Smart Monitoring Temperature and Humidity of the Room Server Using Smart Monitoring Temperature and Humidity of the Room Server Using Raspberry Pi and Whatsapp Notifications," in *Universitas Riau International Conference on Science and Environment 2020 (URICSE-2020)*, 2019, pp. 1–8. doi: 10.1088/1742-6596/1351/1/012006.

[7] I. Ahmad, E. Suwarni, R. I. Borman, A. Asmawati, F. Rossi, and Y. Jusman, "Implementation of RESTful API Web Services Architecture in Takeaway Application Development," in *International Conference on Electronic and Electrical Engineering and Intelligent System (ICE3IS)*, 2022, pp. 132–137. doi: 10.1109/ICE3IS54102.2021.9649679.

[8] H. Ren, Y. Lan, and C. Yin, "The load balancing algorithm in cloud computing environment," *Proc. 2nd Int. Conf. Comput. Sci. Netw. Technol. ICCSNT 2012*, pp. 925–928, 2012, doi: 10.1109/ICCSNT.2012.6526078.

[9] D. Lukitasari, F. Oklilas, F. I. Komputer, and U. Sriwijaya, "Analisis Perbandingan Load Balancing Web Server Tunggal Dengan Web server Cluster Menggunakan Linux Virtual Server," vol. 5, no. 2, pp. 31–34, 2010.

[10] U. Haluoleo, K. Bumi, and T. Anduonohu, "Peningkatan Kinerja Siakad Menggunakan Metode Load Balancing dan Fault Tolerance Di Jaringan Kampus Universitas Halu Oleo," vol. 10, no. 1, pp. 11–22, 2016.

[11] N. Kumar and N. Mishra, "Load Balancing Techniques: Need, Objectives and Major Challenges in Cloud Computing- A Systematic Review," *Int. J. Comput. Appl.*, vol. 131, no. 18, pp. 11–19, 2015, doi: 10.5120/ijca2015907523.

[12] S. D. Riskiono, S. Sulistyo, and T. B. Adji, "Kinerja Metode Load Balancing dan Fault Tolerance Pada Server Aplikasi Chat," *Pros. Semin. Nas. ReTII*, 2017.

[13] S. D. Riskiono, S. Sulistyo, and T. B. Adji, "EVALUASI METODE LOAD BALANCING MENGGUNAKAN HAPROXY SERVER CHAT SOCIAL NETWORK," pp. 635–639, 2016.

[14] T. Chakraborty and I. S. Misra, "Designing a real-time spectrum handoff algorithm for VoIP based Cognitive Radio Networks," *2015 IEEE Radio Antenna Days Indian Ocean. RADIO 2015*, pp. 3–4, 2015, doi: 10.1109/RADIO.2015.7323370.

[15] A. Amarudin and S. D. Riskiono, "Analisis Dan Desain Jalur Transmisi Jaringan Alternatif Menggunakan Virtual Private Network (Vpn)," *J. Teknoinfo*, vol. 13, no. 2, p. 100, 2019, doi: 10.33365/jti.v13i2.309.

[16] D. Aribowo, "Cluster Server IPTV dengan Penjadwalan Algoritma Round Robin," vol. 1, no. 2, pp. 1–5, 2012.

[17] R. Napianto, Y. Rahmanto, R. I. Borman, O. Lestari, and N. Nugroho, "Dhempster-Shafer Implementation in Overcoming Uncertainty in the Inference Engine for Diagnosing Oral Cavity Cancer," *CSRID (Computer Sci. Res. Its Dev. Journal)*, vol. 13, no. 1, pp. 45–53, 2021, doi: 10.22303/csrid.13.1.2021.46-54.

[18] S. D. Riskiono, "Implementasi Metode Load Balancing Dalam Mendukung Sistem Kluster Server," pp. 455–460, 2018, doi: 10.31227/osf.io/9vuzx.

[19] H. Triangga, I. Faisal, and I. Lubis, "Analisis Perbandingan Algoritma Static Round-Robin dengan Least-Connection Terhadap Efisiensi Load Balancing pada Load Balancer Haproxy," *InfoTekJar (Jurnal Nas. Inform. dan Teknol. Jaringan)*, vol. 4, no. 1, pp. 70–75, 2019, doi: 10.30743/infotekjar.v4i1.1688.

[20] G. Triono, "Implementasi Load Balancing Dengan Menggunakan Algoritma Round Robin Pada Kasus Pendaftaran Siswa Baru Sekolah Menengah Pertama Labschool Unesa Surabaya," pp. 169–176, 2015.