Implementation of Load Balancing Technology Using Raspberry Pi as a Server for Computer Based Examination

1st Budi Tjahjono[1], 2nd Ade Sulaeman[1], 3rd Fransiskus Adikara[1], 4th Kundang Karsono Juman[1]
{budi.tjahjono@esaunggul.ac.id[1], ade.sulaeman@esaunggul.ac.id[1],
fransiskus.adikara@esaunggul.ac.id[1]}

Faculty of Computer Science, Esa Unggul University Jakarta, Indonesia[1]

Abstract. Since 2014, the Indonesian government has implemented a computer-based National Examination as a substitute for paper-based National Examination. Schools that conduct computer-based National Examinations should provide the Server as a provider of questions for students to work on. Not all schools have sufficient funds for server procurement. This research demonstrates load balancing technology in the server of computer-based examination at an educational institution using Raspberry Pi as a first step to handling the computer-based National Examination. Raspberry PI is a relatively low-cost single-board computer technology that will serve as a solution to reduce costs for conducting the computer-based examination. This research using PPDIOO method as a research methodology to design and implement the Raspberry Pi as a load balancing server for the computer-based examination server. The result shows that Rasberry Pi can be used as a server and the load balancing technology can also be implemented to improve the access quality of user in the server.

Keywords: load balancing technology, raspberry pi, computer networking.

1 Introduction

Computer-based exams are derivatives that refer to the Computer-Based National Examination which since 2014 has been implemented by the government as a Computer Based Test (CBT) Test and replaces the National Examination (UN) system based on paper. The implementation of UNBK currently uses a semi-online system where the exam is sent from the central server in real time through the network to be synchronized to local servers in schools. The student exam will be served by a local server offline. When finished, the test results are sent back from the local server to the central server online.

Based on data from the Ministry of Education and National Culture (Kemendikbud) of the Republic of Indonesia as of February 5, 2018, stated that in the academic year of 2016/2017 55,802 schools could not implement UNBK. This figure is higher than the number of schools that can implement UNBK, namely 23,342 schools throughout Indonesia. While 3,682 schools join in implementing UNBK by referring to schools that have been able to implement UNBK. The data illustrates that there are still many schools in Indonesia that cannot implement UNBK.

ICONQUHAS 2018, October 02-04, Bandung, Indonesia

DOI 10.4108/eai.2-10-2018.2295570

This UNBK turned out to motivate several schools to create a system similar to UNBK as a school examination system. By making a similar system as a first step, it is hoped that further UNBK implementation can run better. Also, the computer-based examination system will facilitate teachers in correcting values (Susanti, 2016).

In building a server on a local network, a server that is flexible and easy to carry around is needed. This needs to be considered because the computer-based exam process is usually done in class. With this in mind, a server with flexible specifications is needed to adjust to the place.

The development of Raspberry PI as a web server and the load balancer is exciting to be used as research because Raspberry PI as a small computer that has a Linux-based operating system, does not require large power and data storage power to be operated into server clusters (Putra and Sugeng, 2016). With this Raspberry Pi, the server on computer-based exams will be built.

The purpose of this study is to make Raspberry Pi as a load balancing server so that it can become a high availability server and reduce the cost of procuring infrastructure to create a computer-based exam system. The next section will discuss the previous research, followed by research methodology and the steps taken to design and use Raspberry Pi as a server. The results of the study will be discussed in the fourth and final section, concluded in the conclusion part and subsequent work.

2 Research Methods

This server design will be made with Cisco Lifecycle Service or PPDIOO method (Chandrashekhar et al., 2011; Fernando et al., 2016; Hernandez and Jimenez, 2018). Figure 1 shows a framework of thought which describes the steps of the method for developing the system:

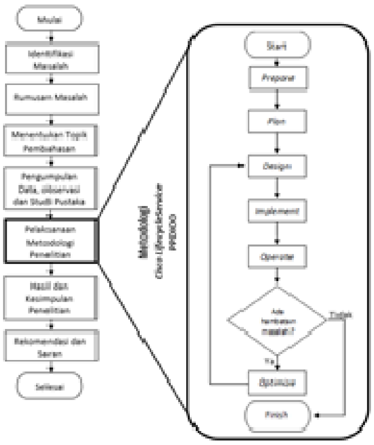

Figure 1. Research Framework

3 Implementation of Raspberry pi as server

In this section, detailed steps are carried out according to the research methodology chosen to implement the Raspberry Pi as a server that can be used as a computer-based Exam server.

3.1 Preparation

Topology design for Computer Based Exams is almost the same as UNBK but does not use the Internet as shown as in Figure 2.

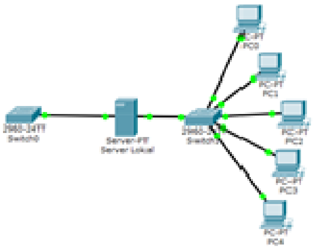

Figure 2. UBK topology

3.2 Plan

Computer networks in this study were described as follows:

- The type of computer network used is client-server.
- The Network Topology used is the Bus Topology on the client network and the Star topology on the server network.
- The computer cluster technique will be used which will form the server design.
- The type of cluster computer for the server to be used is load balancing.

3.3 Design

The topology design that will be implemented in a computer laboratory as shown in Figure 3.

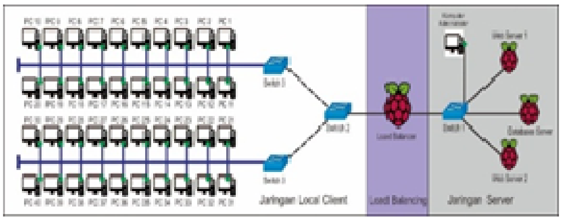

Figure 3. UBK topology

The diagram block that will be used in load balancing using Raspberry Pi is shown in Figure 4.

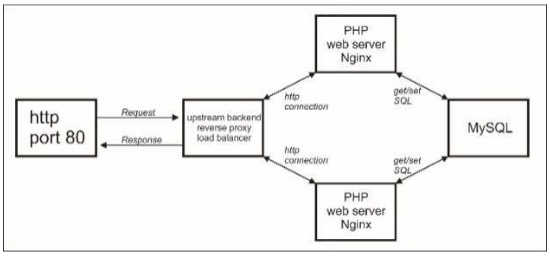

Figure 4. Raspberry Pi load balancing diagram block

In designing this application, it will be made using programming methods with native PHP language. This is done as an effort to maximize performance on load balancing. The structure of the web application page can be seen in Figure 5.

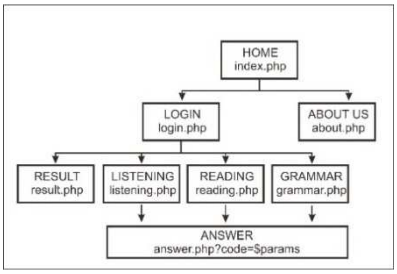

Figure 5. Web application page structure

3.4 Implementation

The first implementation step is setting up an Internet Protocol (IP) address. The IP address of each device that will be implemented in the topology can be seen in Table 3.

Table 1. IP Address Configuration

Device Name	IP Address	Subnet Mask	Gateway
Load Balancer	Eth0: 192.168.10.1 Eth1: 192.168.0.1	255.255.255.240 255.255.255.0	- -
Web Server 1	Eth0: 192.168.10.2	255.255.255.240	192.168.10.1
Web Server 2	Eth0: 192.168.10.3	255.255.255.240	192.168.10.1
Server Database	Eth0: 192.168.10.4	255.255.255.240	192.168.10.1
Computer Admin	Eth0: 192.168.10.5	255.255.255.240	192.168.10.1

User's Computer	Eth0: 192.168.0.2 – 192.168.0.40	255.255.255.0	192.168.10.1

3.5 Operate

At this stage, several scripts will be created on the connected connection to see which web server is actively serving the user. The script is using PHP as follows:

```
<?php
header('Content-Type: text/plain');
session_start();
echo “Web Server 1 <br>”;
echo "IP Server: ".$_SERVER['SERVER_ADDR']; ?>
```

3.6 Optimize

To find out the performance optimization of the load balancer and web server, it will be tested with Web Server Stress Tool software. This can generate reports in the form of data from each user connection.

The following are the scenarios that will be carried out to optimize performance optimization:

- Scenario 1

Method: *Load Balancing*
Device: Raspberry PI 3 Model B
Optimization Duration: 60 minutes
Number of Simulation Users: 50 *users*
Each User's Click Time: 5 seconds

- Scenario 2

Method: Load Balancing
Device: Raspberry PI Model B
Optimization Duration: 60 minutes
Number of Simulation Users: 100 users
Each User's Click Time: 5 seconds

- Scenario 3

Method: Load Balancing
Device: Raspberry PI 3 Model B
Optimization Duration: 60 minutes
Number of Simulation Users: 150 users
Each User's Click Time: 5 seconds

- Scenario 4

Method: Load Balancing
Device: Raspberry PI Model B
Optimization Duration: 60 minutes
Number of Simulation Users: 200 users
Each User's Click Time: 5 seconds

- Scenario 5

Method: Single Server
Device: MSI i5-6400 CPU @ 2.70GHz 16GB
Optimization Duration: 60 minutes
Number of Simulation Users: 200 *users*

Each User's Click Time: 5 seconds

4 Results and Discussions

The following are the test results and the results of the implementation and discussion that has been carried out. This discussion will be explained based on each research factors.

1) Scenario 1

Table 2. Scenario 1 website page testing

URL No.	***Click***	***Error***	***Error [%]***	***Time Spent [ms]***	***Avg. Click Time [ms]***
1	3.187	0	0	1.711.964	537
2	3.192	0	0	1.684.130	528
3	2.239	0	0	2.923.020	1.306
4	2.553	0	0	1.354.100	530
5	2.435	0	0	1.947.686	800
6	2.429	0	0	1.969.809	811
7	2.433	0	0	1.951.161	802
8	2.387	0	0	2.181.128	914
9	2.556	0	0	1.336.245	523
10	2.429	0	0	1.970.465	811
11	2.430	0	0	1.969.450	810
12	2.421	0	0	2.007.114	829
AVG	**2.558**	**0**	**0**	**1.917.189**	**767**

2) Scenario 2

Table 3. Scenario 2 website page testing

URL No.	**Click**	**Error**	**Error [%]**	**Time Spent [ms]**	**Avg. Click Time [ms]**
1	4.923	4.714	95,75	3.372.342	16.136
2	4.948	4.730	95,59	3.117.262	14.299
3	4.899	4.758	97,12	210.988	1.496
4	4.955	4.741	95,68	3.037.322	14.193
5	4.358	4.199	96,35	2.324.231	14.618
6	4.340	4.182	96,36	2.088.325	13.217
7	4.340	4.181	96,34	2.102.674	13.224
8	4.374	4.219	96,46	1.949.025	12.574
9	4.411	4.218	95,62	2.766.360	14.333
10	4.370	4.212	96,38	2.027.006	12.829
11	4.335	4.179	96,4	1.968.268	12.617
12	4.346	4.190	96,41	1.901.862	12.191

AVG	**4.550**	**4.377**	**96**	**2.238.805**	**12.644**

3) Scenario 3

Table 4. Scenario 3 website page testing

URL No.	***Click***	***Error***	***Errors [%]***	***Time Spent [ms]***	***Avg. Click Time [ms]***
1	6.941	32	0,46	10.442.384	1.511
2	6.935	34	0,49	10.499.925	1.522
3	5.262	43	0,82	19.119.125	3.663
4	6.938	29	0,42	10.436.747	1.511
5	6.096	27	0,44	14.952.281	2.464
6	6.098	31	0,51	14.941.108	2.463
7	5.637	32	0,57	13.725.211	2.449
8	5.546	33	0,6	14.135.478	2.564
9	6.403	30	0,47	9.652.627	1.515
10	5.630	32	0,57	13.747.329	2.456
11	5.614	27	0,48	13.858.413	2.480
12	5.622	28	0,5	13.827.995	2.472
AVG	**6.060**	**32**	**1**	**13.278.219**	**2.256**

4) Scenario 4

Table 5. Scenario 4 website page testing

URL No.	***Click***	***Error***	***Errors [%]***	***Time Spent [ms]***	***Avg. Click Time [ms]***
1	8.717	549	6,3	14.439.270	1.768
2	8.700	526	6,05	14.573.025	1.783
3	6.450	460	7,13	26.284.810	4.388
4	8.713	566	6,5	14.393.945	1.767
5	7.464	408	5,47	21.190.571	3.003
6	7.422	375	5,05	21.479.664	3.048
7	7.481	427	5,71	21.133.943	2.996
8	7.348	363	4,94	21.827.892	3.125
9	8.173	488	5,97	13.799.477	1.796
10	7.029	401	5,7	19.926.871	3.006
11	7.025	376	5,35	20.043.314	3.014
12	6.989	360	5,15	20.165.396	3.042
AVG	**7.626**	**442**	**6**	**19.104.848**	**2.728**

5) Scenario 5

Table 6. Scenario 5 website page testing

URL No.	***Clicks***	***Errors***	***Errors [%]***	***Time Spent [ms]***	***Avg. Click Time [ms]***
1	9.530	0	0	12.079.092	1.267
2	10.276	0	0	8.284.757	806
3	8.323	0	0	18.310.717	2.200
4	10.284	0	0	8.229.207	800
5	9.408	0	0	12.715.358	1.352
6	9.410	0	0	12.696.090	1.349
7	9.414	0	0	12.705.133	1.350
8	9.362	0	0	13.037.187	1.393
9	9.670	0	0	7.792.409	806
10	8.851	0	0	12.004.091	1.356
11	8.847	0	0	12.045.974	1.362
12	8.850	0	0	12.052.230	1.362
AVG	**9.352**	**0**	**0**	**11.829.354**	**1.284**

In Table 10, the number of clicks is different. Average errors, time spent on requests, and the average time for each click.

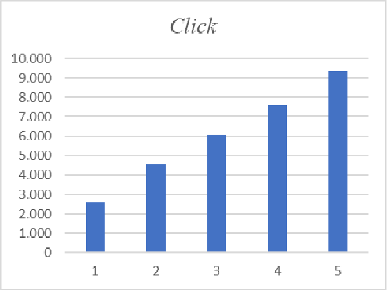

Figure 6. Testing the number of clicks

According to Figure 6, the most significant amount of clicks occurs in scenario 4 as many as 7,626 and scenario 1 gets the lowest number of clicks which is only 2,558. This can happen because the higher number of users, the higher number of clicks will be

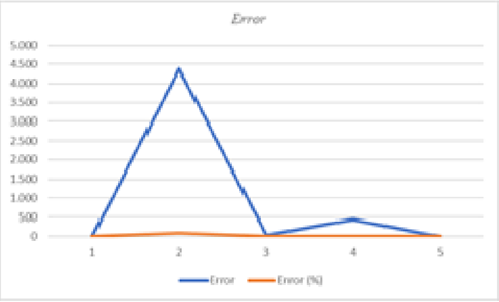

Figure 7. Testing the number of errors

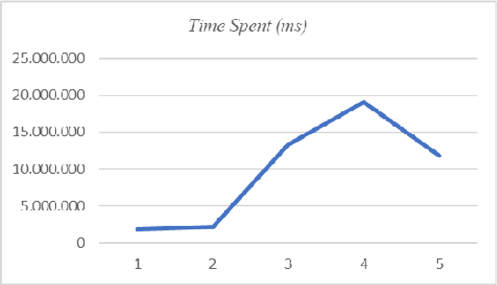

Figure 8. Testing on the amount of Time Spent (ms)

The time spent on the website for each user request follows the number of users themselves. The more the users, the busier the website's response will be. This is illustrated by the table where the total time spent in scenario 4 is greater because in this scenario the number of users reaches 200 within 60 minutes. Greater than other scenarios.

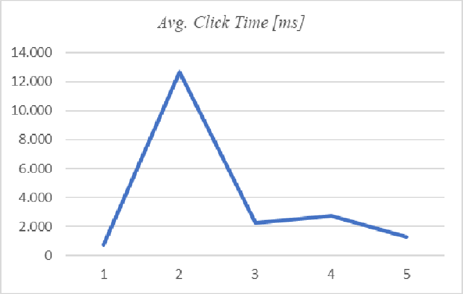

Figure 9. Testing on the amount of Average Click Time (ms)

According to Figure 9, the average time for each click in scenario 2 is the biggest, which is 12,644 ms, this is because in scenario 2 there is an error during testing.

Server and Bandwidth Test Result

1) Scenario 1

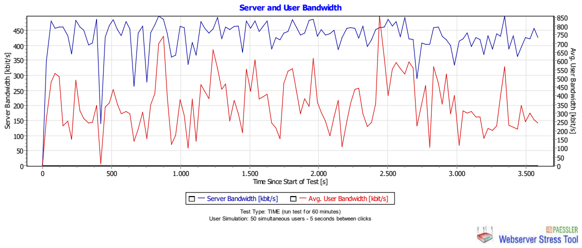

Figure 10. Scenario 1 server and user bandwidth test

As shown as in Figure 10, the average bandwidth is so large, and in scenario 1 it can be seen that the bandwidth of the server is much larger.

2) Scenario 2

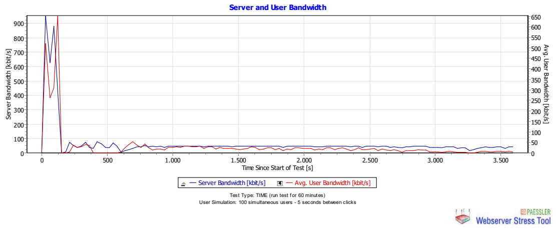

Figure 11. Scenario 2 server and user bandwidth test

According to Figure 11, in scenario two the amount of bandwidth was not stable, this is affected because in scenario 2 there are many up to 96%.

3) Scenario 3

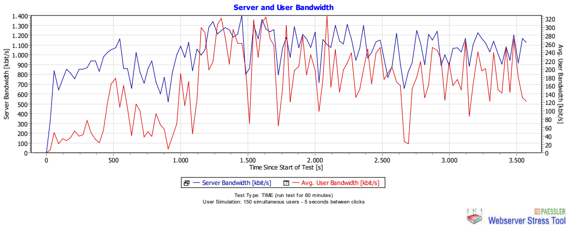

Figure 12. Scenario 3 server and user bandwidth test

As shown as in Figure 12, in the third scenario, there was an increase in the user's average bandwidth.

4) Scenario 4

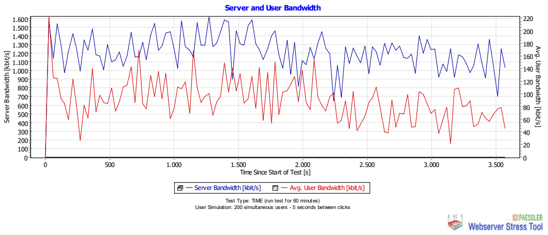

Figure 13. Scenario 4 server and user bandwidth test

5) Scenario 5

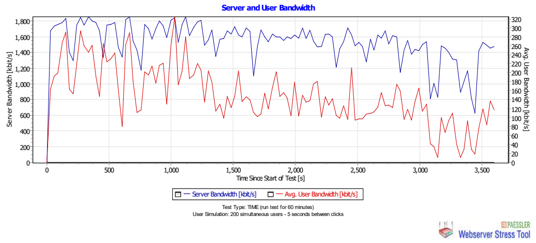

Figure 14. Scenario 5 server and user bandwidth test

From some of these results, it can be concluded that the bandwidth of the clustering load balancing system can meet bandwidth with up to 200 users in 60 minutes.

Results of Data Transfer Test, Memory System, and CPU Load

In the graphs, as shown in Figure 15 until 19, show the traffic on the network that is related to memory and the load that will be received by the server CPU. The following is the graph:

1) Scenario 1

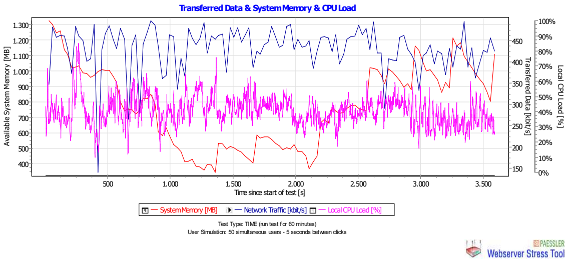

Figure 15. Scenario 1 data transfer, memory system, and a CPU load test

2) Scenario 2

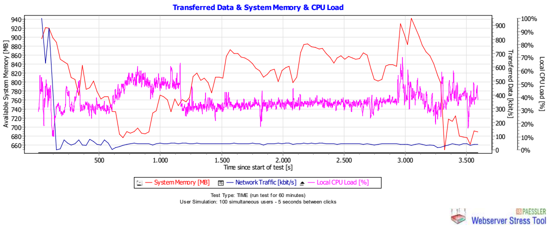

Figure 16. Scenario 2 data transfer, memory system, and a CPU load test

3) Scenario 3

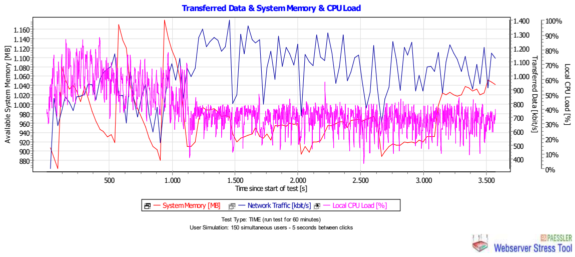

Figure 17. Scenario 3 data transfer, memory system, and a CPU load test

4) Scenario 4

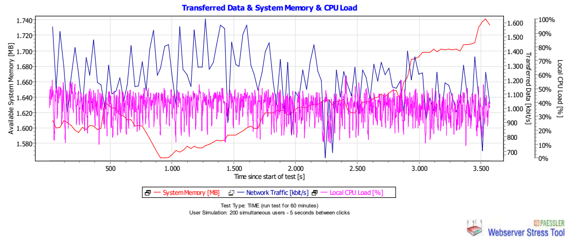

Figure 18. Scenario 4 data transfer, memory system, and a CPU load test

5) Scenario 5

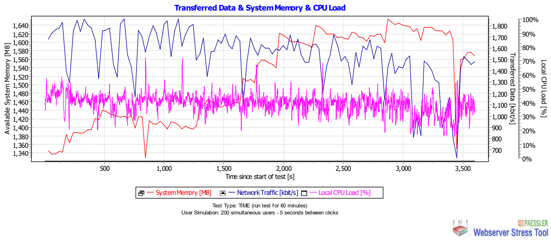

Figure 19. Scenario 5 data transfer, memory system, and a CPU load test

5 Conclusions

The following are the conclusions obtained from the results of this study. The conclusions are as follows:

a. The use of Raspberry Pi can be implemented in the computer lab of SMPN 89 Jakarta with 30 students per class.
b. Raspberry Pi with Raspbian/Linux system operation can be used as a load balancer, web server, database server, FTP server and as a cluster system
c. The higher the number of users will affect the average response time for each click and the response time for each click will be longer
d. The number of clicks will be directly proportional to the number of users. Where if there are many users, then the click received by the server is even greater.

e. The higher the number of users, the website's response will be busier greater the number of average click time that the user receives.

f. Stable bandwidth in each test indicates that the bandwidth in the clustering system with one Raspberry Pi as a load balancer, two Raspberry Pi as web servers and one Raspberry as database server is enough to meet the number of users as many as 200 users for 60 minutes.

g. Traffic on load balancing is still stable with a period of 60 minutes, and the number of users is between 1-200.

References

[1] Cardellini, V., Colajanni, M., and Yu, P. S. (1999). Dynamic load balancing on web-server systems. *IEEE Internet Computing*, 3(3): 28-39.

[2] Chandrashekhar, P. K., Rangaswamy, S., and Srigiri, A. (2011). Plan Design Implement Framework Tool For NCE. *International Journal of Innovative Technology & Creative Engineering*, 1(3): 34-39.

[3] Fernando, M. R. R., Magaly, L. M. N., and Jose, C. S. M. (2016). Analysis of Methodologies of Data Networks LAN. *International Journal of Advanced Engineering Research and Science*, 3(9): 52-61.

[4] Hernandez, L., and Jimenez, G. (2018, April). Design and Validation of a Scheme of Infrastructure of Servers, Under the PPDIOO Methodology, in the University Institution-ITSA. In *Computer Science Online Conference* (pp. 367-379). Springer, Cham.

[5] Moniruzzaman, A. B. M., Waliullah, M., and Rahman, M. (2014). A High Availability Clusters Model Combined with Load Balancing and Shared Storage Technologies for Web Servers. *arXiv preprint arXiv:1411.765.*

[6] Panwar, R., and Mallick, B. (2015, October). Load balancing in cloud computing using dynamic load management algorithm. In *2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*, pp. 773-778. IEEE.

[7] Puthal, D., Obaidat, M. S., Nanda, P., Prasad, M., Mohanty, S. P., and Zomaya, A. Y. (2018). Secure and Sustainable Load Balancing of Edge Data Centers in Fog Computing. *IEEE Communications Magazine*, 56(5): 60-65.

[8] Putra, R. H., and Sugeng, W. Implementasi Cluster Server pada Raspberry Pi dengan Menggunakan Metode Load Balancing. *Jurnal Edukasi dan Penelitian Informatika (JEPIN)*. 2(1).