CyberSecurity dan Forensik Digital e-ISSN: 2615-8442
Vol. 8, No. 1, Mei 2025, hlm. 17-24

Perbandingan Load Balancing Router Mysql Dan HAProxy Menggunakan SysBench dan Cluster Innodb Pada Sistem Operasi Centos

Arini [1], Andrew [2], Ridwan Baharsyah [3]

1,2,3 UIN Syarif Hidayatullah Jakarta
Email: 1. arini@uinjkt.ac.id, 2. andrew_fiade@uinjkt.ac.id, 3. rbaharsyah33@gmail.com

Abstrak

Load balancing merupakan penyeimbang server dalam mendistribusikan beban kerja ke beberapa server dengan mempertimbangkan kapasitas masing-masing server. Ketika beberapa server digunakan, layanan yang ada dapat tetap berfungsi meskipun salah satu server mengalami kegagalan. Dua model load balancing yang akan digunakan adalah MySQL Router dan HAProxy. Penelitian ini bertujuan untuk membandingkan kinerja MySQL Router dan HAProxy dari segi waktu respons, throughput, dan distribusi beban server. Selain itu, penelitian ini juga menguji sinkronisasi data antar server database dengan menggunakan Sysbench sebagai alat pengujian. Sysbench merupakan utilitas benchmark yang dapat mengevaluasi kinerja sistem melalui berbagai parameter pengujian. Hasil penelitian menunjukkan bahwa MySQL Router memiliki kemampuan load balancing yang signifikan dalam mendistribusikan beban dan memastikan ketersediaan server dibandingkan dengan HAProxy. Pengujian dengan thread (beban) terkecil hingga terbesar pada load balancer MySQL Router menghasilkan rentang TPS (Transaction Per Second) 2900 hingga 2600; seiring bertambahnya thread (beban), TPS yang diperoleh semakin menurun, dengan rentang waktu respons 2 hingga 50 ms. Namun, HAProxy menunjukkan nilai TPS yang lebih kecil, berkisar antara sekitar 900 hingga 800 TPS, tetapi menghasilkan waktu respons yang relatif lama, berkisar antara 8 hingga 160 ms. Pengujian sinkronisasi database menunjukkan efisiensi kedua model dalam menangani perubahan data pada server yang berbeda. Penelitian ini memberikan kontribusi yang signifikan terhadap pengembangan infrastruktur TI yang lebih andal dan efisien dalam organisasi, khususnya dalam konteks penggunaan MySQL InnoDB Cluster dan HAProxy pada OS CentOS.

***Kata kunci**: Load Balancing, MySQL Router, HaProxy, InnoDB Cluster, Centos Os, Networking*

Comparison Of Load Balancing Mysql Router And HAProxy Using SysBench In Innodb Cluster On Centos OS

Abstract

Load balancing is a server balancer that distributes the workload among several servers, taking into account the capacity of each server. When multiple servers are used, existing services can continue to function even if one server fails. The two load balancing models to be used are MySQL Router and HAProxy. This study aims to compare the performance of MySQL Router and HAProxy in terms of response time, throughput, and server load distribution. Additionally, this study also tests data synchronization between database servers using Sysbench as a testing tool. Sysbench is a benchmark utility that can evaluate system performance through various test parameters. The results of the study show that MySQL Router has significant load balancing capabilities in distributing loads and ensuring server availability compared to HAProxy. Testing with the smallest to the largest threads on the MySQL Router load balancer resulted in a TPS range from 2900 to 2600; as the thread (load) increases, the TPS obtained decreases, with a response time range of 2 to 50 ms. However, HAProxy showed a smaller TPS value, ranging from around 900 to 800 TPS, but resulted in a relatively long response time, ranging from 8 to 160 ms. Database synchronization tests also reveal the efficiency of both models in handling data changes on different servers. This research makes a significant contribution to the development of more reliable and efficient IT infrastructure within organizations, particularly in the context of using MySQL InnoDB Cluster and HAProxy on CentOS OS.

***Keywords**: Load Balancing, MySQL Router, HaProxy, InnoDB Cluster, Centos Os, Networking.*

1. INTRODUCTION

The application of information technology in a company or organization is something that is really needed at this time. A good server is a necessity for a company's external relations. With the current high number of internet users, of course servers are needed for companies and organizations. Database storage is needed to process and optimize data on the server (Cynthia et al., 2020) (Umar Ali Ahmad, 2021), (Jader, 2019)

MySQL's default storage engine, InnoDB, strikes a compromise between fast performance and great dependability. The secret to handling and optimizing data on the server is InnodB. A server system that can manage a lot of accesses is required to improve data transactions and storage services. Load balancing offers a multiple server service paradigm, which allows for system scalability (Georgiou, 2020), (Riskiono, & Pasha, 2020), (Wijayanti, 2020), (Arnqvist, 2023), (Bell, 2019).

Load balancing, application connection failure, and client routing are one of the solutions to the problems above, all three of which are features of the MySQL Router. Apart from MySQL Router, HAProxy can also be used for TCP/HTTP load balancing and proxy solutions that can be run on Linux operating systems (Umar Ali Ahmad, 2021), (Rawls, C., 2022).

As the OS used for Load Balance, using Centos OS is a strong reason to maintain security, CentOS is equipped with advanced architecture such as Security Enhanced Linux (SELinux). The SELinux system is an access control policy that can be activated to manage the security configuration of all processes and files (Cloudmatika, 2023), ((Šušter & Ranisavljević, 2023).

In this research, we tested using Sysbench on MySQL Router and HaProxy to get the response time and throughput of the server node. Both benchmarking processes are carried out on CentosOS. Testing was carried out using a public database from US National Flight Data which contains 200 records that refers to publicly available records of airline flights collected by the U.S. Department of Transportation, specifically by the Bureau of Transportation Statistics (BTS). This data includes detailed records of commercial airline flights, both domestic and international. This included FL_DATE (Flight Date), OP_UNIQUE_CARRIER (Airline), OP_CARRIER_FL_NUM (Flight Number), ORIGIN and DEST (Airports), DEP_TIME, ARR_TIME (Times), DEP_DELAY, ARR_DELAY (Delays) (Kraska et al., 2021), (Ahmad, et al., 2021),

As a test tool for both load balances, Sysbench is one of the benchmark utilities that functions to evaluate parameter feature testing for system performance. According to (Jesper Wisborg Krogh, 2020), Sysbench is the most commonly used benchmark tool in the MySQL environment. It has built-in tests for OLtp workloads, non-database tests (such as pure i/O, CpU, and memory tests), and more. In addition, the latest version supports custom workloads. It is open source, most sysbench is used on Linux. We will conduct how the response time, throughput and load sharing testing of the servers in both load balancing, as well as testing the synchronization of each database based on (Alankar et al., 2020). Database synchronization testing is carried out to find out data changes on one server in another database server where the measurement calculates the data change process and the time required using PPDIO (Prepare, Plan, Design, Implement, Operate, Optimize). The data analysis method, we used for server testing is a benchmark with transaction per second and average response time parameters based on (Harefa, et al., 2021). Our research are how to overcome the failover system on both MySQL Router and HAproxy Load Balancers then How is the performance of each load balancer (Response and Throughput) produced. The research outcome are knowing the results of the system failover test on both MySQL Router and HAproxy Load Balancers so that server data remains consistent, also knowing the performance of each load balancer (Response and Throughput) produced.

2. RELATED WORK

Several studies have done load balance detection task using several engines. Research by Taufiq et al., 2015) conducted a testing response time and throughput using various thread parameters with TPS. The number of MySQL cluster load balancing TPS which is worth 50.65 is better than the default MySQL cluster which is worth 25.73 and the response times parameter on the MySQL cluster load balancing is 3576.01, faster than the default MySQL cluster which is 11900.58.

Other research was also carried out by (Dani et al.,2017) which is doing load sharing testing with 10 - 50 requests. Fail over testing with 100 requests and server performance using 10000-30000 requests with different response and throughput results. The result shows that MySQL is unable to handle heavy queries from 3 backend servers even though it has been optimized. With HAProxy as Load Balancer, Carrying out High Availability and Failure tests by simultaneously testing high availability and failure synchronization, the data entered on server1 and vice versa must be able to be synchronized. The result shows that in terms of high availability, the amount of data on server 1 and server 2 is the same even though a transaction has occurred when the test failure occurs when one of the nodes is turned off and the amount of data on the server remains the same.

By (Alankar et al., 2020) that used the round robin algorithm performs better than the least connections algorithm. Based on (Trinugi et al., 2022) show that on the faster the server responds to requests from users so that server performance can be continuously improved. The result from (Waluyo et al., 2023) that calculate throughput and response time between web server and HAProxy server. Through HAProxy the throughput load is shared between one server and another. Testing proves that HAProxy is able to share the load between web servers.

Based on the paper (Assegaff et al., 2023) resulted that the eReview System found increased performance in responding to requests and processing data. Using APDEX values, which increased by 6% and 9%. For the data processing each operation experienced an increase also. At the automated test of saving and displaying files show both scenarios were successful in storing and displaying files in Amazon S3.

The result show that Web server load balancing using HAProxy can improve website server performance based on server availability (uptime) of 99.49% and an average click time of 7,291ms per use (Riska et al., 2021). By (Cynthia et al., 2020), the research found that the HAProxy server is capable of handling problems and to be a solution to bridge changes in existing data. The research by (Joshua et al., 2021) compare the performance of NGINX and HAProxy based on response time and error rate which run on docker using virtual machine type in both AWS and GCP. The result shows that the NGINX could handle medium and heavy load better than HAProxy. The AWS could handle medium and heavy load better than GCP.

(Dede et al., 2021) give result that With the replication technique in this database, it will make it easier for us to move data or create data on two or more devices at once, and will shorten our time at work. From (Arifin et al., 2022) informed that implementation of cloud server clustering load balancing with HAProxy is better in terms of response time, concurrent requests, and failover system compared to the results of a single local server.

3. METHODOLOGY

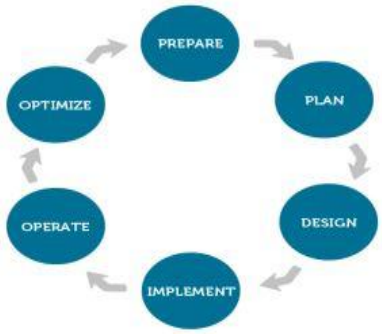

Figure 1. PPDIOO Method

This section explains the details of the proposed approach shown in figure 1. The proposed approach consists of six basic phases based from PPDIOO method in order to get the output from each load balancer and use it for comparation.

3.1. Preparation

The preparation step, data are required in order to doing further step of testing load balancing function. In this research, flying records database are used as primary database. Laptop with 16 GB RAM are used in order to operate Centos OS. And last are software, and the software are shown as Table I.

Table 1. Software Preparation

No	Tools	Description
1	Oracle VM Virtual Box Manager 6.1	Used as a virtual machine for Centos OS
2	GNS 3	Used for MySQL Router and HaProxy Network Load Balancer architecture design
3	Centos OS versi 8 or newest	As an operating system on virtual box
4	InnoDB Cluster	Functions as a database storage machine
5	Load Balancer Haproxy	Server load balancer
6	Load Balancer MySQL Router	Server load balancer
7	SysBench	As a tool for measuring load balancer performance

In order to test failover, we need prepare for the network design to. Five address are created, and each spesification are listed in Table 2.

Table 2. OS Specification

No	IP Address	Spesification
1	192.168.86.18 subnet mask 255.255.255.0 (/24)	Ip app 2 Centos OS Ram : 1,5 GB
2	192.168.86.104 subnet mask 255.255.255.0 (/24)	Ip app 1 Centos OS Ram : 1,5 GB
3	192.168.86.206 subnet mask 255.255.255.0 (/24)	Ip db 1 Centos OS Ram : 1,5 GB
4	192.168.86.147 subnet mask 255.255.255.0 (/24)	Ip db 2 Centos OS Ram : 1,5 GB
5	192.168.86.95 subnet mask 255.255.255.0 (/24)	Ip db 3 Centos OS Ram : 1,5 GB

3.2. Planning

The plan step is to create configuration of MySQL Router, HAProxy, InnoDB Cluster and Sysbench. For MySQL Router command is used in order to set static IPv4 in the hosts settings in accordance with IPv4 on each node, app1, db1, db2, and db3 as shown in Figure 2 and make sure the 'hosts' script

allows you to change the IP on each node. The output of MySQL Router is shown as Figure 3.

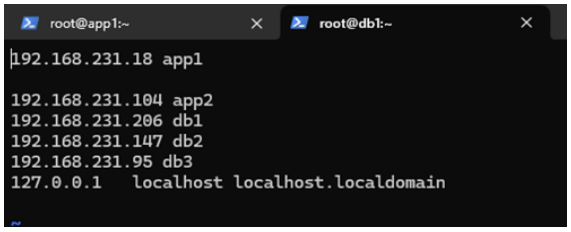

Figure 2. Configure the IP 'hosts' script on the MySQL Router

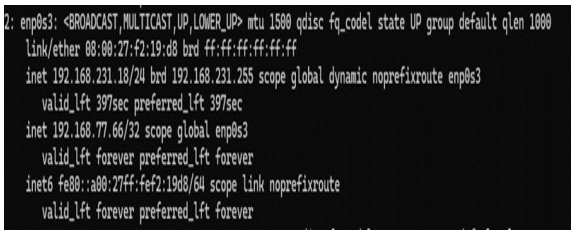

Figure 3. Display ip on each mode (bridge enp0s3)

For HaProxy configuration we setting Set static IPv4 into host configuration according to IPv4 on each node with commands. We ensure that the IP in each node can be adjusted in “hosts” script. The configuration HAProxy result shown at the Figure 4.

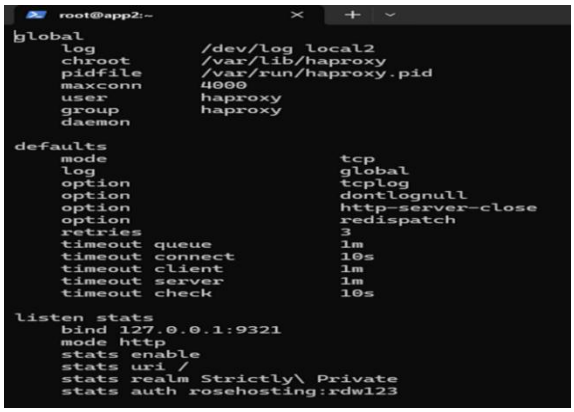

Figure 4. Configuration HaProxy

The 'global' and 'defaults' configurations are automatic settings from the HAProxy configuration. Meanwhile, 'listen statistics' is a configuration for connecting the HAproxy display to 'localhost' which can be seen in the browser display. The 'listen-innodb-cluster' shows ip 127.0.0.1:3307 (admin cluster) which binds to app2 which automatically uses the load balance algorithm (round robin) and ip 192.168.231.206.

For innodb cluster configuration is in the admin db, namely db1 as the 'primary' main or master node. Meanwhile, nodes 2 and 3 will become secondary nodes. All 3 nodes must be online. InnodB cluster configuration settings shown as Figure 5. And lastly the configuration of Sysbench are conducted by taking data from database that we used in this research.

Figure 5. ImnodB configuration

3.3. Design

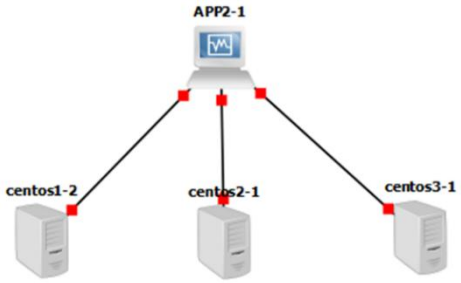

Figure 6. Testing Topology

In this phase, we creating a Topology based on prepare and plan step. We used Centos OS that are listed in Table II to test load balance function. In the MySQL Router Load Balance, it has 1 node as the app1 server and 3 server nodes, namely db1, db2 and db3. Likewise with HAProxy Load Balance, it has 1 node as the app2 server and 3 server nodes, namely db1, db2 and db3, then finally the design shown as Figure 6.

3.4. Implement and Operate

In this phase, we ensure configuration of load balancing HAProxy, MySQL Router were conducted before we continue to test it using Sysbench. Server load balancing divides the burden among several servers based on each server's capability in order to minimize server failures. As a

result, each server must be tested with thread configuration input = 10 in a minute.

The goal of using the Sysbench program is to test the load balancer's functionality. The difference in the average respon time and the quantity of received/read transactions will prove the funcionality. The script code for testing the MySql Router shown at figure 7 bellow.

```
sysbench --threads=10  --time=60 --events=0
--report-interval=1  --mysql-db=sbtest --
mysql-user=sbtest --mysql-
password=Password_123 --mysql-
host=192.168.231.18 sysbench_test.lua run
```

Figure 7. The script code for testing the MySql Router

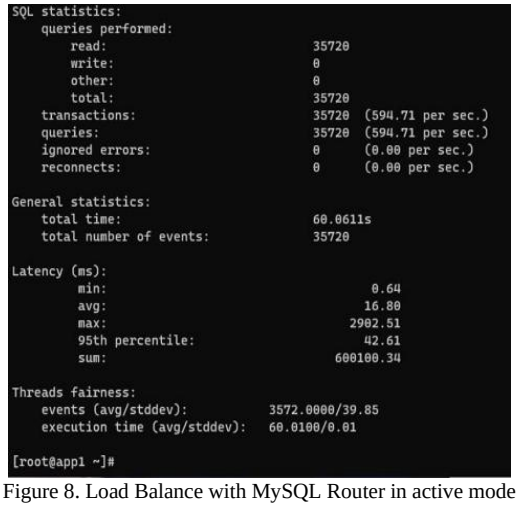

Figure 8. Load Balance with MySQL Router in active mode

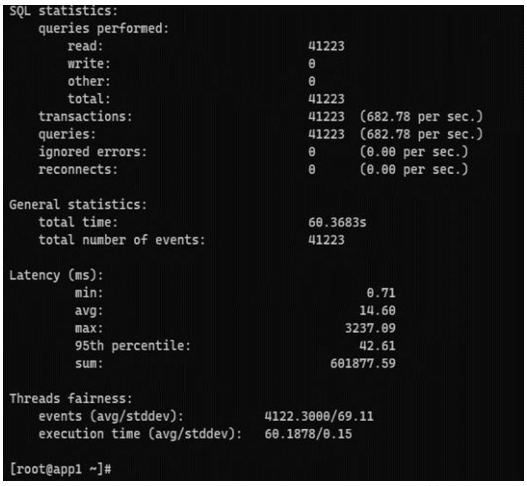

Figure 9. Load Balance with MySQL Router in offline mode

Output result of the load balance with MySQL Router shown as Figure 8, and without MySQL Router or inactive condition shown as Figure 9.

Load balancer analysis on HAProxy calls the sysbench_test.lua configuration with configuration test time = 1 minute, thread = 10, database = sbtest, ip = 192.168.231.104 (app2), the output result of the load balance with HAProxy shown as Figure 10, and without Haproxy or inactive condiction shown as Figure 11.

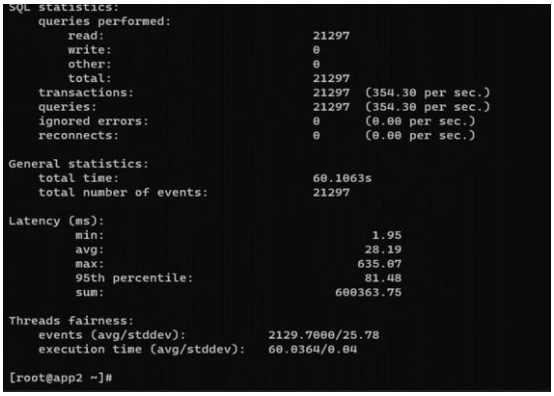

Figure 10. Load Balance with HAProxy Router in active mode

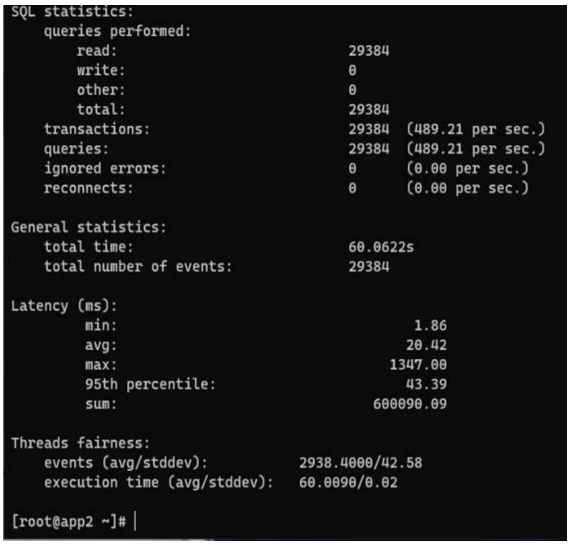

Figure 11. Load Balance with HAProxy Router in offline mode

3.5. Optimize

The result of each test is gathered in this phase. In order to determine if a particular approach is possible for load balancing or not, there are always certain constraints, limits, and dimensional limitations.

These constraints are as follows:

1. **Throughput**: This parameter emulates the server's capabilities. Server capability means how much weight it can take. This is one of the important parameters that supports web application performance calculations. Maximum throughput is always expected. Throughput is calculated as the number of requests in a certain time or transactions per second.
2. **Average Response Time**: This is the aggregate time taken to start fulfilling a user's request after processing the request.
3. **Fault tolerance**: The capability of a load balancing algorithm that allows a structure to perform in multiple drops below the system state.

4. EXPRIMENTS AND RESULT ANALYSIS

In this section, we present experimental results from two load balancing using MySQL Router and HAproxy. Each load balance was tested using the Sysbench tool as a benchmark tool for measuring response time and threads per second (TPS) to compare the results.

This test is carried out after the load balancing system and fault tolerance system have been successfully completed. By providing a test load (thread) on each load balancer with values of 8, 16, 32, 64, 80, 96, 112 and 128 to get parameter values for response time and throughput. The result are shown in Table 3.

Table 3. Performance Result

Thread	MySQL Router		HaProxy	
	TPS	Response Time (ms)	TPS	Response Time (ms)
8	2921.10	2.74	904.55	8.84
16	2978.46	5.37	907.09	17.63
32	2981.71	10.73	883.24	36.20
64	2909.37	21.98	825.16	77.46
80	2881.97	27.74	816.04	97.88
96	2752.12	34.86	800.54	119.68
112	2664.57	42.00	799.44	139.82
128	2612.20	48.94	801.22	159.38

It can be observed from MySQL Router output in Table 3 that the more threads in the test, the longer the load balancer response time. The amount of time used in the graph is milliseconds. Even though there was a spike in tps in threads 16 and 32, specifically from 2978.46 to 2981.71, the results indicate that the average response time value for each thread has increased significantly. The tps value decreases with the number of threads (load) supplied into the test.

We can observe from HAProxy output in Table 3 that the HAProxy load balancer experienced a decrease in the number of threads per second (tps) from the 8 thread parameters tested. The more threads (load) that are input into the test, the smaller the tps value will be. However, the resulting tps value is better than MySQL Router with tps on thread 8 below 2000 tps. Even though there was an increase in tps on threads 16 and 123, namely 907.09 and 801.22. However, the results show that the average response time value for each thread has increased significantly.

Finally, fault tolerance testing is carried out to see system availability when a failure or error occurs from the server, both hardware and software. Testing system services to ensure the system is running well. The fault tolerance testing plan is as follows. Server load balancing test scenario:

1. **The condition of the master and slave nodes is on**. The results shown as Figure 12, that data taken from the 'Flights' table in db1 with the last input being the cities 'jkt' and 'bgr' in the active db3 (secondary) and db1 (primary) conditions.

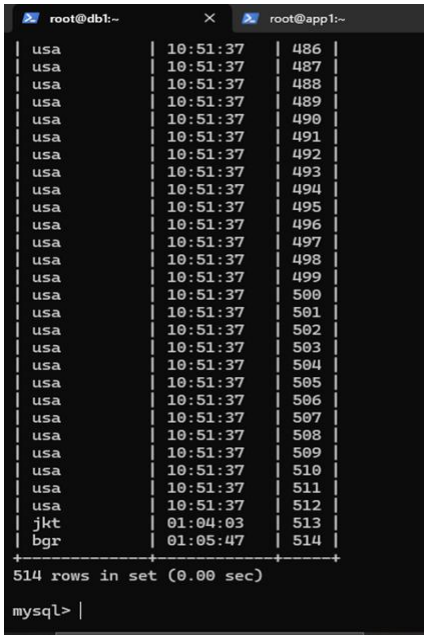

Figure 12. db1 (primary) dan db3 (secondary live)

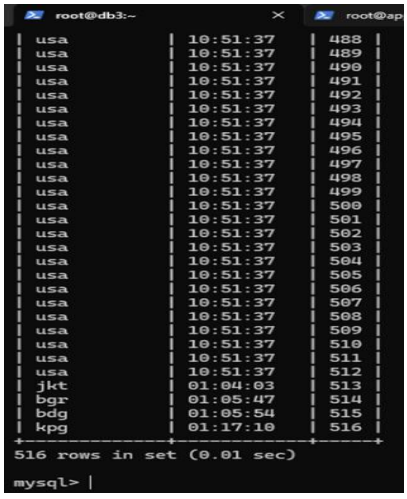

Figure 13. Input data on dbl (primary)

2. **The condition of the master node is on and the slave node is turned off**. The result shown as Figure 13, shows the condition of DB3 when it is turned back on and it is proven that data entered while DB3 is off can still be synchronized.
3. **The condition of the master node being turned off and on again**. When the db1 is turned off and on again, The master node (primary) changes to belong to DB3 and it is proven that if the master node is turned off, another slave node will replace it as the master node. The result shown as Figure 14.

```
"status": "OK",
"statusText": "Cluster is ONLINE and can tole
"topology": {
    "db1:3306": {
        "address": "db1:3306",
        "memberRole": "SECONDARY",
        "mode": "R/O",
        "readReplicas": {},
        "replicationLag": null,
        "role": "HA",
        "status": "ONLINE",
        "version": "8.0.26"
    },
    "db2:3306": {
        "address": "db2:3306",
        "memberRole": "SECONDARY",
        "mode": "R/O",
        "readReplicas": {},
        "replicationLag": null,
        "role": "HA",
        "status": "ONLINE",
        "version": "8.0.26"
    },
    "db3:3306": {
        "address": "db3:3306",
        "memberRole": "PRIMARY",
        "mode": "R/W",
        "readReplicas": {},
        "replicationLag": null,
        "role": "HA",
        "status": "ONLINE",
        "version": "8.0.26"
    }
```


Figure 14. Primary condition moves to db3

CONCLUSION

From this research regarding load balancing and fault tolerance methods on network servers, the following conclusions are obtained:

1. In the condition of using 10 threads, applying the load balancing method can reduce the value of response time and can increase the throughput value, where the results of MySQL Router with tps results of 35720 594.71 per sec are better than Haproxy Response 21297 354.30 with conditions of 10 threads. Meanwhile, the MySQL Router thrughput results with thread events results of 3572,000/39.85 are better than HaProxy's 2129.7000/25.78. This means that the system can avoid overloads that come from users using MySQL Router better than Haproxy.
2. Meanwhile, in testing thread variations 8, 16, 32, 64, 80, 96, 112 and 128, it shows that the tps value obtained from the smallest to the largest thread on the MySQL Router load balancer ranges from tps 2900 to 2600, the larger the thread (load) then the tps obtained will be smaller, with a response time range of 2 - 50 ms. However, HaProxy shows a smaller tps value, namely around 900 to 800 tps but produces a fairly long response time, namely in the range of 8 – 160 ms. These results show that the more TPS you get, the better it is and the response time is smaller. So it can be concluded that MySQL Router is superior to HaProxy in this research.
3. A system with fault tolerance has been successfully implemented so that when the master load balancer fails, its role can be replaced by the slave load balancer within 2 seconds and when the master load balancer recovers, its role can be taken back from the slave load balancer within the same time, namely 2 seconds so that system availability continues to be maintained

Future development still needs to be done. several aspects can be considered, such as the using Load Balancer with scheduling algorithm configurations such as Round Robin and Least Connection. In addition, next study Load Balancer testing can be done with other tools such as Apache Jmeter or Apache Benchmark. The other comparasion can be done by using Nginx, Traevik or Envoy. Other aspects include various methodologies and techniques, which are opportunities that can be studied further.

REFERENCES

Eka Pandu Cynthia, Iwan Iskandar, Anwar Alfaruqi Sipayung.(2020). Rancang Bangun Server HAproxy Load Balancing Master to Master MySQL (Replication) Berbasis Cloud Computing. ALGORITMA: Jurnal Ilmu Komputer dan Informatika Volume: 04, Number: 01, April 2020 ISSN 2598-6341 (online)

Umar Ali Ahmad, R. E. (2021). Implementasi High Availability Server Menggunakan Platform Haproxy (Studi Kasus: Aplikasi Zammad Untuk Online Help Desk). *e-Proceeding of Engineering : Vol.8, No.5*, 6238.

Jader, O. H., Zeebaree, S. R., & Zebari, R. R. (2019). A state of art survey for web server performance measurement and load balancing mechanisms. International Journal of Scientific & Technology Research, 8(12), 535-543.

Georgiou, M. A. (2020). ENABLING WORKLOAD SCALABILITY, STRONG CONSISTENCY AND ELASTICITY WITH TRANSACTIONAL DATABASE REPLICATION. Cyprus University of Technology, 14.

Riskiono,, S. D., & Pasha, D. (2020). Analisis Perbandingan Server Load Balancing dengan Haproxy & Nginx dalam Mendukung Kinerja Server E-Learning. Bandar Lampung: InComTech: Jurnal Telekomunikasi dan Komputer.

Wijayanti, W. (2020). HIGH PERFORMANCE DATABASE SERVER (HIGH AVAILABILITY DATABASE SERVER) MENGGUNAKAN MARIADB GALERA CLUSTER . Universitas Muhammadiyah Surakarta Jurnal, 2.

Arnqvist, A. (2023). EVALUATING FAILOVER AND RECOVERY OF REPLICATED SQL DATABASES. Umea Universitet.

Bell, C. (2019). Introducing InnoDB Cluste. Warsaw, Virginia, USA: Apress.

Ahmad, U. A., Saputra, R. E., & Harahap, R. M. (2021). Implementasi High Availability Server Menggunakan Platform Haproxy (studi Kasus: Aplikasi Zammad Untuk Online Help Desk). EProceedings of Engineering, 8(5).

Rawls, Connor & Salehi, Mohsen. (2022). Load Balancer Tuning: Comparative Analysis of HAProxy Load Balancing Methods. 10.48550/arXiv.2212.14198.

Cloudmatika : https://cloudmatika.co.id/blog-detail/centos-adalah, (2023, January 23).

Šušter, Ivan & Ranisavljević, Tamara. (2023). Optimization of MySQL database. Journal of Process Management and New Technologies. 11. 141-151. 10.5937/jouproman2301141Q.

Jesper Wisborg Krogh. (2020). MySQL 8 Query Performance Tuning A Systematic Method for Improving Execution Speeds. Hornsby, NSW, Australia: Apress Media LLC.

Alankar, Bhavya & Sharma, Gaurav & Kaur, Harleen & Valverde, Raul & Chang, Victor. (2020). Experimental Setup for Investigating the Efficient Load Balancing Algorithms on Virtual Cloud. Sensors. 20. 7342. 10.3390/s20247342.

Harefa, H. S., Triyono, J., & Raharjo, S. (2021). Implementasi Load Balancing Web Server untuk Optimalisasi Kinerja Web Server dan Database Server. Jurnal Jarkom, 9(1), 10–20.

Abdul Gani, Taufiq & Arafat, Aulia & Melinda, Melinda. (2015). Analisis Kinerja MySQL Cluster Menggunakan Metode Load Balancing. Jurnal Rekayasa Elektrika. 11. 10.17529/jre.v11i4.2358.

Dani, Rahmad & Suryawan, Fajar. (2017). Perancangan dan Pengujian Load Balancing dan Failover Menggunakan NginX. Khazanah Informatika: Jurnal Ilmu Komputer dan Informatika. 3. 43. 10.23917/khif.v3i1.2939.

Alankar, Bhavya & Sharma, Gaurav & Kaur, Harleen & Valverde, Raul & Chang, Victor. (2020). Experimental Setup for Investigating the Efficient Load Balancing Algorithms on Virtual Cloud. Sensors. 20. 7342. 10.3390/s20247342.

Harjanti, Trinugi & Setiyani, Hari & Trianto, Joko. (2022). Load Balancing Analysis Using Round-Robin and Least-Connection Algorithms for Server Service Response Time. Applied Technology and Computing Science Journal. 5. 40-49. 10.33086/atcsj.v5i2.3743.

Waluyo, Muhammad & Antony, Fery & Setiawan, Candra. (2023). IMPLEMENTASI LOAD BALANCING WEB SERVER DENGAN HAPROXY MENGGUNAKAN ALGORITMA ROUND ROBIN. Journal of Intelligent Networks and IoT Global. 1. 46-52. 10.36982/jinig.v1i1.3074.

Assegaff, Husin Muhammad (2023) Implementasi Load Balancing Web Server dan Basis Data Terdistribusi MySQL Cluster pada Perangkat Lunak eReview: Marketplace untuk Sistem Telaah Artikel Ilmiah, http://repository.its.ac.id/id/eprint/101860

Riska, Riska & Alamsyah, Hendri. (2021). Analisa Dan Perancangan Load Balancing Web Server Mengunakan HAProxy. Techno.Com. 20. 552-565. 10.33633/tc.v20i4.5225.

Kurniawan, R., Sari, L. H., Aspriyono, H. (2023). Web Server Load Balance Design In Internet Network. Jurnal Media Computer Science, 2(2).

Mulyana, Joshua & Raharjo, Willy & Indriyanta, Gani. (2021). Studi Komparasi Performa NGINX dan HAPROXY Sebagai Load Balancer di Cloud Menggunakan Teknologi Kontainer. Jurnal Terapan Teknologi Informasi. 5. 11-17. 10.21460/jutei.2021.51.229.

Yusup, Dede & Anwar, Nisrina & Aminatu Haya, Alliqza & Fauzan, Fikry & Gilang Suherman, Intan Permatasari. (2021). Load Balancing pada database menggunakan HaProxy dengan Algoritma Roundrobin, serta replikasi pada MySQL dan MongoDB dengan pengujian database menggunakan Python.

Ariansyah Arifin , Gito Putro Wardana, Syamsul Arifin, M Fadlan Ridho, Hamonangan Kinantan Prabu. (2022). Penerapan Cloud Load Balancing Dengan menggunakan HAProxy Dalam Meningkatkan Server Availability Pada Studi Kasus Learning Management System (LMS) Universitas XYZ. Seminar Nasional Mahasiswa Ilmu Komputer dan Aplikasinya (SENAMIKA) Jakarta-Indonesia, 20 Agustus 2022, e-ISSN 2962-6129