Comparative Analysis of Key Management Service Performance on AWS, Google Cloud, and Oracle Cloud with Performance Testing

1st Aries Susanto
Department of Information Systems
UIN Syarif Hidayatullah Jakarta
South Tangerang, Indonesia
ariessht@uinjkt.ac.id

2nd Ahnaf Hadi Fathulloh
Department of Digital and Analytics
Mitra Solusi Telematika
Jakarta, Indonesia
ahnaf.hadi@gmail.com

3rd Nuryasin
Department of Information Systems
UIN Syarif Hidayatullah Jakarta
South Tangerang, Indonesia
nuryasin@uinjkt.ac.id

4th Aida Fitriyani
Department of Informatics
Bhayangkara University Jakarta
Jakarta, Indonesia
aida.fitriyani@gmail.com

***Abstract*—Although switching to cloud technology for free exclusive can give cost advantage and efficiency, it requires policies, processes, and practices best security level business calculated. Based on survey to industry experts, available problem security in cloud computing, including data breaches identified as a problem security top need _ attention more. Most entry way sense to prevent data breach involves practice security in data storage, for example data encryption. But if happen lost key encryption, then it will also data loss. It supported by abundance data breaches that have occurred in the last 5 years recently, especially in Indonesia. To overcome security and management issues key encryption, some cloud providers provide Key Management Services (KMS) services. This research will compare the performance of Key Management Services from cloud providers AWS, Google Cloud, and Oracle Cloud with load testing methods, stress testing, and benchmark testing. The parameters assessed from this research are response time, error rate, throughput, and latency. The resulting research results are recommendations for the best cloud providers in Key Management Services. The result shows AWS can say better compared Google Cloud and Oracle because constant error rate lower from Google Cloud and Oracle, competitive response time and latency. Throughput (requests per second) obtained almost always more excels at every testing.**

Keywords— Performance Analysis, Cloud Computing, Cloud Security, Data Breaches, JMeter

I. INTRODUCTION

In recent years, the demand for data has increased drastically as has the number of online users. Therefore, an external storage system is required to store data. Traditional computing is unable to handle the increasing number of online users due to the growth of the global internet. A new concept in data storage systems appears, namely cloud computing [1].

Cloud computing is a model of easy resource access and can be delivered on demand [2]. In cloud computing, customers only pay for services used with the PAYG model. Benefits include flexibility in customizing software, storage, development platforms, and computing resources [3].

In Indonesia, there have been several cases of user data violations on cloud service platforms such as Bukalapak, Tokopedia, and BPJS [4]. These cases show that inadequate implementation of encryption can lead to leaking of user data [15]. To address security issues and management of encryption keys, several cloud service providers provide Key Management Services (KMS) services. KMS is a system that leverages cryptography and key lifecycle management functionality to connect applications and services, and automate key management processes [5].

Cloud service providers such as Amazon Web Services (AWS), Google Cloud, and Oracle Cloud are recognized as leaders in cloud platform infrastructure and services [6]. They have invested heavily in building data centres and offering on-premises services, as well as offering a wide range of services, tools and support to their customers [6], [7].

Based on the importance of key management in cryptography and the need to reduce the impact of data breaches, the author will conduct research that focuses on comparative analysis of Key Management Service (KMS) performance on AWS, Google Cloud, and Oracle Cloud using load testing, stress testing, and benchmark testing methods.

II. RELATED WORK

A. Key Management Service

Key Management Service (KMS) is system that utilizes functionality cryptography and management cycle life key to connect applications and services. Functions too expanded to manage secrets and certificates. NIST defines KMS as a system for management key cryptography and its metadata, include various processes such as creation, distribution, storage, backup, archiving, restoration, use, revocation, and destruction key. KMS got used in a manner automation to oversee, automate, and secure management processes key [8].

B. Load Testing

Load testing is a method of performance testing that involves the continuous operation of a system within the pressure it can withstand. The purpose is to test the stability of the system by ensuring the system can run consistently under these stresses. *Load testing* helps understand the performance capacity of the system and can be used as a basis for performance tuning [16]. It is important to distinguish *load testing* from *stress testing* , in which *stress testing* involves evaluating a system's performance beyond its normal capacity. *Load testing* is part of performance testing that ensures testing is carried out within the specified resource capacity [9], [10].

C. Stress Testing

Stress testing is a performance testing method that involves continuously increasing pressure on the system under test until the system fails. The goal is to test the maximum pressure that the system can withstand. *Stress testing* involves a gradual increase in system load to test for changes in system performance and determine conditions under which the system fails to deliver the maximum level of performance service. The difference with *load testing* is that in *stress testing*, the test is carried out at the maximum pressure that the system can withstand [9].

D. Benchmark Testing

Overall, *benchmark testing* is a performance testing method that involves measuring and comparing system performance using standardized tests [11]. The purpose of benchmark testing is to provide a standardized and objective way to evaluate and compare the performance of different computer systems, as well as identify performance bottlenecks that can be optimized to improve system performance. *Benchmark testing* is used in a variety of fields, including computer architecture, hardware and software design, and system optimization [11].

E. Apache JMeter

Apache JMeter is a desktop application used to test and measure the performance and functional behavior of client/server applications, such as web applications or FTP applications. As one of the most popular *open-source testing applications*, JMeter is designed in Java and has expandability via a provided API. By acting as the "client side" of a "client/server" application, JMeter measures response time and server resources such as CPU load, memory usage, and other resource usage. This enables automated functional testing [10].

JMeter can be used to test the performance of static and dynamic resources such as static files, *Servlets*, *FTP servers*, Databases, and *queries*. To test and measure the robustness of *HTTP or FTP servers* or networks, JMeter users can simulate various types of loads on the tested system. With its graphical tools, JMeter facilitates better performance analysis in heavy load situations [10].

F. Similar Literature

After determining the topic of research and the formulation of the problem, the author collects data by reading and studying books, journals, and also thesis which is used as a reference in order to obtain a theoretical basis regarding the problem to be studied. Similar literature can be seen in Table I

III. Research Method

A. Method Data Analysis

At stage analysis performance, this study uses three method testing that is *load testing* , *stress testing* , and *benchmark testing* . *Load testing* is carried out to test the stability of the system by running operations continuously within the pressure limit that can be withheld. Apache JMeter is used as a tool testing with a test plan.

TABLE I Load Test Plan Scenario 1

Group Threads	**name**	Load Test Scenario 1
Thread Properties	**Number of threads**	100
	Ramp-up Period (in seconds)	1
	Loop Count	10

TABLE II Load Test Plan Scenario 2

Group Threads	**name**	Load Test Scenario 2
Thread Properties	**Number of threads**	250
	Ramp-up Period (in seconds)	1
	Loop Count	10

Stress testing is performed to test changes in system performance by gradually increasing system load. The pressure on the system is tested continuously until the system collapses or is unable to withstand the applied load. Apache JMeter is used for measurements in stress testing with a test plan.

TABLE III Stress Test Plan Scenario 1

Group Threads	**name**	Stress Test Scenario 1
Thread Properties	**Number of threads**	5000
	Ramp-up Period (in seconds)	1
	Loop Count	10

TABLE IV Stress Test Plan Scenario 2

Group Threads	**name**	Stress Test Scenario 2
Thread Properties	**Number of threads**	10000
	Ramp-up Period (in seconds)	1
	Loop Count	10

Benchmark testing is a performance testing methodology that involves measuring and comparing system performance using a series of standardized tests known as benchmarks. At this stage, the system is tested by encrypting files of various sizes and the results are compared. Apache JMeter is also used to perform benchmark testing with the test plan listed in the table.

TABLE V Benchmark Test Input Plan

No	**file sizes**	**Number of threads**	**Extension**
1	100 kb	10	pdf
2	1mb	10	pdf
3	10mb	10	pdf

TABLE VI Benchmark Test Plan

Group Threads	**name**	Benchmark Test Plan
Thread Properties	**Number of threads**	10
	Ramp-up Period (in seconds)	1
	Loop Count	1

The output of the test is in the form of response time, error rate, throughput, and latency which will be compared.

IV. Results

A. Load Testing

1) Average Response Time

Table 8 is comparison of the average *response time load testing* function encryption and function Key Management Service decryption from scenario 1 and scenario 2 on AWS, Google Cloud, and Oracle providers. The smallest average *response time* from the encryption function is obtained by AWS in scenario 1 and scenario 2. The largest average *response time* from scenario 1 for the encryption function is obtained by Google Cloud, in scenario 2 is obtained by Oracle. In the decryption function, the least average *response time* from scenario 1 and scenario 2 is obtained by AWS, and the largest average *response time* from scenario 1 and scenario 2 is obtained by Oracle.

TABLE VII Comparison of Average Response Time Load Testing

Label	Provider	Scenario 1	Scenario 2
Encrypt Function	AWS	197	402
	Google Cloud	1124	2694
	Oracles	978	2920
Decrypt function	AWS	230	295
	Google Cloud	1245	2336
	Oracles	1370	2560

2) Minimum Response Time

Table 9 is comparison *minimum response time load testing* function encryption and function Key Management Service decryption from scenario 1 and scenario 2 on AWS, Google Cloud, and Oracle providers. *The minimum response time* for the encryption function is obtained by Oracle in scenario 1 and scenario 2. *The minimum response time* for scenario 1 is obtained by AWS and scenario 2 is obtained by Oracle.

TABLE VIII Comparison of Minimum Response Time Load Testing

Label	Provider	Scenario 1	Scenario 2
Encrypt Function	AWS	65	85
	Google Cloud	74	71
	Oracles	56	63
Decrypt function	AWS	34	90
	Google Cloud	79	73
	Oracles	101	70

3) Maximum Response Time

Table 10 is comparison *maximum response time load testing* function encryption and function Key Management Service decryption from scenario 1 and scenario 2 on AWS, Google Cloud, and Oracle providers. *The minimum maximum response time* of the encryption and decryption functions is obtained by AWS in scenario 1 and scenario 2. *The maximum maximum response time* of the encryption and decryption functions is obtained by Oracle in scenario 1 and scenario 2.

TABLE IX Comparison of Maximum Response Time Load Testing

Label	Provider	Scenario 1	Scenario 2
Encrypt Function	AWS	2790	715
	Google Cloud	7170	21069
	Oracles	7623	16355
Decrypt function	AWS	2417	537
	Google Cloud	7404	21063
	Oracles	7706	21411

4) Error

Table 11 is *error comparisons load testing* the encryption function and decryption function Key Management Service from scenario 1 and scenario 2 on AWS, Google Cloud, and Oracle *providers.* The fewest errors in the encryption and decryption functions in scenario 1 and scenario 2 were obtained by AWS. The biggest error from the encryption function in scenario 2 is obtained by Google Cloud. The biggest error from the decryption function in scenario 2 is obtained by Oracle.

TABLE X Comparison of Error Load Testing

Label	Provider	Scenario 1	Scenario 2
Encrypt Function	AWS	2790	715
	Google Cloud	7170	21069
	Oracles	7623	16355
Decrypt function	AWS	2417	537
	Google Cloud	7404	21063
	Oracles	7706	21411

5) Throughput

Table 12 is *throughput comparisons load testing* the encryption function and decryption function Key Management Service from scenario 1 and scenario 2 on AWS, Google Cloud, and Oracle *providers . The greatest throughput* of scenario 1 and scenario 2 encryption and decryption functions was obtained by AWS. The smallest *throughput of the scenario 1 encryption function is obtained by Google Cloud and the scenario 2 encryption function is obtained by Oracle.* In the decryption function the smallest *throughput from scenarios 1 and 2 is obtained by Google Cloud.*

TABLE XI Comparison of Throughput Load Testing

Label	Provider	Scenario 1	Scenario 2
Encrypt Function	AWS	161.81	524,21
	Google Cloud	65,6	68,51
	Oracles	76,3	63,38
Decrypt function	AWS	169.3	651.04
	Google Cloud	63,2	73,40
	Oracles	64,3	78,21

6) Latency

Table 13 is comparison *latency load testing* function encryption and function decryption Key Management Service from scenario 1 and scenario 2 on AWS, Google Cloud, and Oracle *providers*. Least *latency* on functions encryption and function decryption Scenario 1 and Scenario 2 are obtained by AWS. The biggest *latency on* the function encryption Scenario 1 is obtained by Google Cloud and scenario 2 is obtained by Oracle. In the decryption function, the greatest *latency in scenario 1 and scenario 2 is obtained by Oracle.*

TABLE XII Comparison of Latency Load Testing

Label	Provider	Scenario 1	Scenario 2
Encrypt Function	AWS	197,059	402,628
	Google Cloud	1124.52	2694.68
	Oracles	978,498	2920.76
Decrypt function	AWS	230,262	295.8836
	Google Cloud	1245.82	2336.78
	Oracles	1370,435	2560.05

B. Stress Testing

1) Average Response Time

Table 14 is a comparison of the average *response time stress testing* the encryption function and decryption function Key Management Service from scenario 1 and scenario 2 on AWS, Google Cloud, and Oracle *providers* . In scenario 1 and scenario 2, the smallest average *response time* obtained by Oracle is 6687 ms and 6877 ms. Meanwhile, the largest average *response time* in scenario 1 encryption function was obtained by Google Cloud at 15673 ms and in scenario 2 encryption function obtained by AWS. In the scenario 1 decryption function, the smallest average *response time* is obtained by AWS of 6093 and in scenario 2 obtained by Oracle is 5576 ms. While the highest average *response time* in the scenario 1 and scenario 2 decryption function was obtained by Google Cloud at 13859 ms and 9705 ms.

TABLE XIII COMPARISON OF AVERAGE RESPONSE TIME STRESS TESTING

Label	Provider	Scenario 1	Scenario 2
Encrypt Function	AWS	7047	12858
	Google Cloud	15673	9828
	Oracles	6687	6877
Decrypt function	AWS	6093	8574
	Google Cloud	13859	9705
	Oracles	6635	5576

2) Minimum Response Time

Table 15 is minimum *response time* comparison *stress testing* function encryption and function decryption Key Management Service from scenario 1 and scenario 2 on AWS, Google Cloud, and Oracle *providers*. Oracle gets the smallest minimum *response time* on the encryption and decryption functions in scenario 1. There are several *minimum response times* that produce a value of 0 so that it is not visible on the graph. This is because there are so many *errors that occur at once that the request* is immediately countered.

TABLE XIV COMPARISON OF MINIMUM RESPONSE TIME STRESS TESTING

Label	Provider	Scenario 1	Scenario 2
Encrypt Function	AWS	851	0
	Google Cloud	0	0
	Oracles	63	0
Decrypt function	AWS	909	0
	Google Cloud	0	0
	Oracles	61	0

3) Maximum Response Time

Table 16 is comparison maximum *response time stress testing* function encryption and function decryption Key Management Service from scenario 1 and scenario 2 on AWS, Google Cloud, and Oracle *providers*. AWS gets the smallest result the maximum *response time* of function encryption and decryption in scenario 1, but in scenario 2 it functions encryption, AWS maximum *response time increases* substantially drastic up to 320678 ms.

TABLE XV COMPARISON OF MAXIMUM RESPONSE TIME STRESS TESTING

Label	Provider	Scenario 1	Scenario 2
Encrypt Function	AWS	21250	320678
	Google Cloud	45262	50427
	Oracles	57959	107945
Decrypt function	AWS	10529	75578
	Google Cloud	101837	72516
	Oracles	81213	71820

4) Error

Table 17 is a comparison of the percentage error *stress testing* of the encryption function and the decryption function Key Management Service from scenario 1 and scenario 2 on AWS, Google Cloud, and Oracle *providers* . In the scenario 1 encryption and decryption function, AWS gets the least percentage of errors, namely 0.995% and 0.747%. But in scenario 2, the percentage of AWS errors rises to the range of 46% to 53%. Google Cloud and Oracle got somewhat consistent results in the range of 76% to 93% in scenario 1 and scenario 2.

TABLE XVI COMPARISON OF ERROR STRESS TESTING

Label	Provider	Scenario 1	Scenario 2
Encrypt Function	AWS	0.995	46,743
	Google Cloud	78,638	91,292
	Oracles	89,943	90,994
Decrypt function	AWS	0.747	53,568
	Google Cloud	76.06	92,195
	Oracles	90,374	93.77

5) Throughput

Table 18 is *throughput comparisons stress testing* the encryption function and decryption function Key Management Service from scenario 1 and scenario 2 on AWS, Google Cloud, and Oracle *providers.* AWS got the highest *throughput result on scenario 1 encryption function, and scenario 1 and scenario 2 decryption function.* Google Cloud got results highest *throughput* on function encryption scenario 2.

TABLE XVII COMPARISON OF THROUGHPUT STRESS TESTING

Label	Provider	Scenario 1	Scenario 2
Encrypt Function	AWS	701.1285	554,8832
	Google Cloud	278,369	582.5547
	Oracles	514,774	457,2054
Decrypt function	AWS	808.5473	916.5491
	Google Cloud	298.9898	644,1867
	Oracles	498.4205	817.0162

6) Latency

Table 19 is comparison of the average *latency* of the test *stress testing* function encryption and function decryption Key Management Service from scenario 1 and scenario 2 on AWS, Google Cloud, and Oracle *providers*. AWS got the lowest average *latency on scenario 1 encryption function, scenario 1 and scenario 2 decryption function. Oracle got* the lowest average *latency on scenario 2 encryption function.*

TABLE XVIII COMPARISON OF LATENCY STRESS TESTING

Label	Provider	Scenario 1	Scenario 2
Encrypt Function	AWS	7076,899	21624.73
	Google Cloud	19552,47	23219.05
	Oracles	19814,49	20406
Decrypt function	AWS	6108,458	17165.57
	Google Cloud	16974,61	22899.32
	Oracles	20140,7	18436,87

C. Benchmark Testing

1) Average Response Time

Table 20 is comparison of the average *response time* of testing *benchmark testing* of the function encryption and function decryption Key Management Service on the insert file sizes 100KB, 1MB, and 10MB. In the 100KB file encryption function, AWS gets an average response time of less than Google Cloud. Meanwhile, the 1MB and 10MB file encryption, Google Cloud has an average response time that is lower than AWS. Oracle Cloud produces a constant *average response time* of 0 ms due to unsuccessful *benchmark testing*.

TABLE XIX COMPARISON OF AVERAGE RESPONSE TIME BENCHMARK TESTING

Label	Cloud Provider	Average Response Time (ms)
Encrypt Files 100KB	AWS	97
	Google Cloud	133
	Oracles	0
Encrypt Files 1MB	AWS	192
	Google Cloud	147
	Oracles	0
Encrypt Files 10MB	AWS	2117
	Google Cloud	1023
	Oracles	0

2) Minimum Response Time

Table 21 is comparison of minimum *response time* testing *benchmark testing* function encryption and function decryption Key Management Service on the insert file sizes 100KB, 1MB, and 10MB. At 100 KB file encryption, AWS gets minimum *response time* results more A little compared to Google Cloud. Meanwhile, at 1 MB and 10 MB file encryption, Google Cloud gets minimum *response time* results are more A little compared to AWS. Oracle Cloud generates a constant *minimum response time* of 0 ms due to unsuccessful *benchmark testing.*

TABLE XX COMPARISON OF MINIMUM RESPONSE TIME BENCHMARK TESTING

Label	Cloud Provider	Min Response Time (ms)
Encrypt Files 100KB	AWS	88
	Google Cloud	102
	Oracles	0
Encrypt Files 1MB	AWS	162
	Google Cloud	123
	Oracles	0
Encrypt Files 10MB	AWS	781
	Google Cloud	534
	Oracles	0

3) Error

Table 22 is comparison error presentation testing *benchmark testing* function encryption and function decryption Key Management Service on the insert file sizes 100KB, 1MB, and 10MB. From pictures we can conclude that there are none of the errors generated at the time testing good *benchmark testing* from AWS or Google Cloud *providers.* Especially for Oracle Cloud, it produces a constant error of 0% due to unsuccessful benchmark testing.

TABLE XXI COMPARISON OF ERROR BENCHMARK TESTING

Label	Cloud Provider	Error %
Encrypt Files 100KB	AWS	0
	Google Cloud	0
	Oracles	0
Encrypt Files 1MB	AWS	0
	Google Cloud	0
	Oracles	0
Encrypt Files 10MB	AWS	0
	Google Cloud	0
	Oracles	0

4) Throughput

Table 23 is a comparison *of throughput for benchmark* testing of the encryption function and decryption function Key Management Service in the file enter size 100KB, 1MB, and 10MB. At 100 KB file encryption, AWS gets slightly more *throughput* than Google Cloud. Meanwhile, for 1 MB and 10 MB file encryption, Google Cloud gets greater *throughput than AWS.* Oracle Cloud generates a constant *throughput* of 0 requests per second due to unsuccessful *benchmark testing.*

TABLE XXIII COMPARISON OF THROUGHPUT BENCHMARK TESTING

Label	Cloud Provider	Throughput(s)
Encrypt Files 100KB	AWS	10.04016
	Google Cloud	9.8912
	Oracles	0
Encrypt Files 1MB	AWS	9.43396
	Google Cloud	9.68054
	Oracles	0
Encrypt Files 10MB	AWS	2.6462
	Google Cloud	4.58926
	Oracles	0

5) Latency

Table 24 is comparison of the average latency of benchmark testing function testing encryption and function decrypt Key Management Service in the insert file sizes 100KB, 1MB, and 10MB. At 100 KB file encryption, AWS gets more latency results A little compared to Google Cloud. However, at 1 MB file encryption and 10 MB Google Cloud gain more latency results A little compared to AWS. Oracle Cloud produces an average constant latency of 0 ms due to unsuccessful *benchmark testing*.

TABLE XXIII COMPARISON OF LATENCY BENCHMARK TESTING

Label	Cloud Provider	Latency (ms)
Encrypt Files 100KB	AWS	97.1
	Google Cloud	133,4
	Oracles	0
Encrypt Files 1MB	AWS	171
	Google Cloud	142.9
	Oracles	0
Encrypt Files 10MB	AWS	1774,4
	Google Cloud	975.3
	Oracles	0

V. CONCLUSION

Based on results analysis Key Management Service performance on AWS, Google Cloud, and Oracle using load testing methods, stress testing, and benchmark testing, can concluded things following:

1. In the load testing test, AWS shows the best results compared to Google Cloud and Oracle. AWS has more response time, throughput, and latency well, as well No generates an error in the load testing test. Google Cloud and Oracle experienced an error rate of 0.44% to 2.44%.

2. In stress testing, the percentage of errors is a determining factor for the success of requests. AWS has more error percentage low than Google Cloud and Oracle, while the best latency, throughput, and response time alternate between AWS, Google Cloud, and Oracle.
3. In benchmark testing, AWS has more results good at encrypting file sizes of 100 KB, while Google Cloud is more both in encryption file size of 1 MB and 10 MB in terms of response time, throughput, and latency. No, there is an error that occurs in benchmark testing.
4. Based on testing whole, author recommend AWS provider in implementing Key Management Service because more error rate low, competitive response time and latency, as well as more throughput superior.

ACKNOWLEDGMENT

The author would like to thank various parties who have supported the author in completing this research as well as possible.

REFERENCES

[1] H. Tabrizchi and M. K. Rafsanjani, "A survey on security challenges in cloud computing: issues, threats, and solutions," *J Supercomput*, vol. 76, no. 12, pp. 9493–9532, Dec. 2020.

[2] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," Sep. 2011.

[3] N. Dimitri, "Pricing cloud IaaS computing services," *Journal of Cloud Computing*, vol. 9, no. 1, pp. 1–11, Dec. 2020.

[4] P. G. Bhwana and R. M. Nugraha, "Bukalapak Confirm of an Attempted Customer Data Breach - News En.tempo.co," Mar. 18, 2019. https://en.tempo.co/read/1186473/bukalapak-confirm-of-an-attempted-customer-data-breach (accessed Feb. 02, 2023).

[5] E. Barker, "NIST Special Publication 800-57 Part 1 Revision 5 Recommendation for Key Management: Part 1-General," May 2020.

[6] Gartner, "Magic Quadrant for Cloud Infrastructure as a Service, Worldwide," 2021.

[7] IDC, "IDC MarketScape: Worldwide Public Cloud Infrastructure as a Service 2020 Vendor Assessment," 2021.

[8] D. Egan *et al.*, "Key Management in Cloud Services: Understanding Encryption's Desired Outcomes and Limitations 2," 2020, Accessed: Feb. 05, 2023. [Online]. Available: https://cloudsecurityalliance.org/research/working-groups/cloud-key-management/

[9] J. Wang and J. Wu, "Research on performance automation testing technology based on JMeter," *Proceedings - 2019 International Conference on Robots and Intelligent System, ICRIS 2019*, pp. 55–58, Jun. 2019.

[10] E. H. Halili, *Apache JMeter : a practical beginner's guide to automated testing and performance measurement for your websites*. Packt Pub, 2008.

[11] X. Huang, "Task scheduling in cloud computing using particle swarm optimization with time varying inertia weight strategies," *Cluster Comput*, vol. 23, no. 2, pp. 1137–1147, 2020.

[12] H. Martins, F. Araujo, and P. R. da Cunha, "Benchmarking Serverless Computing Platforms," *J Grid Comput*, vol. 18, no. 4, pp. 691–709, Dec. 2020.

[13] A. C. Rompis and R. F. Aji, "Perbandingan Performa Kinerja Node.js, PHP, dan Python dalam Aplikasi REST Performance Comparison of Node.js, PHP, and Python Performance," *Cogito Smart Journal*, vol. 4, no. 1, 2018.

[14] V. Janani and K. Krishnamoorthy, "Evaluation of cloud based performance testing for online shopping websites," *Indian J Sci Technol*, vol. 8, no. 35, Dec. 2015.

[15]Nurbojatmiko, A. Susanto, E. Shobariah. "Assessment of ISMS based on standard ISO/IEC 27001: 2013 at Diskominfo Depok City." In International Conference on Cyber and IT Service Management (CITSM), pp. 1-6, April 2016.

[16] A.Susanto, L. Latifah, Nuryasin, A. Fitriyani. "Decision support systems design on sharia financing using Yager's fuzzy decision model." In International Conference on Cyber and IT Service Management (CITSM), pp. 1-4, August 2017.

[17] E. F. Noviani, B. Kembara, B. A. Yudha Pratama, D. A. Permata Sari, A. M. Shiddiqi, and B. J. Santoso, "Performance Analysis of AWS and GCP Cloud Providers," *Proceedings - 2022 IEEE International Conference on Cybernetics and Computational Intelligence, CyberneticsCom 2022*, pp. 236–241, 2022.