Jurnal Ilmiah Teknik Elektro Komputer dan Informatika (JITEKI)
Vol. 8, No. 4, December 2022, pp. 576-586
ISSN: 2338-3070, DOI: 10.26555/jiteki.v8i4.24272

Shibboleth IdP for Single Sign-On with Kubernetes and Persistent Volume Longhorn

Ikhwan Alfath Nurul Fathony, Mukhammad Andri Setiawan
Magister Informatika, Fakultas Teknik Industri, Universitas Islam Indonesia, Yogyakarta

ARTICLE INFO

Article history:

Received June 27, 2022
Revised November 01, 2022
Published December 24, 2022

Keywords:

Single Sign-On;
Shibboleth;
IdP;
Kubernetes;
Persistent Volume Claim;
Container;
Block Storage Longhorn

ABSTRACT

Many organizations do not use centralized user authorization with Single Sign-On (SSO) Management to seamlessly move from one system to another. The same thing also occurred at Universitas Islam Indonesia (UII). Students were having trouble login in from one web service to another. The Board of Information Systems of UII, or Badan Sistem Informasi (BSI), implements SSO to avoid this problem. However, after BSI implemented SSO on the virtual machine, it turned out that the server load became too high. A spiking number of user logins happened in a short period. The centralized system could not handle this. The research's solution is to use a clustered service using Shibboleth IdP. The Shibboleth IdP customization can be carried out to be deployed into the Kubernetes cluster infrastructure ecosystem to meet the needs of authentication login on the business processes at UII. The Shibboleth IdP itself will be equipped with a persistent storage longhorn to support and maintain the service and avoid a single point of failure. The Kubernetes and Persistent Volume Longhorn provide a redundancy function in an application and a more flexible replication process. Inside Kubernetes, there is containerization technology. It was used to optimize the server's resources instead of replicating the application using virtual machines. With the use of centralized login by Shibboleth IdP and persistent storage longhorn, the error because of server load could be minimized. The downtime of the downed services can also be reduced. The research also proves that using Kubernetes and Persistent Volume Longhorn could help the system by preventing a Single Point of Failure using its redundancy function.

Corresponding Author:

Corresponding Author Name, Affiliation, Address, City and Postcode, Country
Email: ikhwan.alfath@uii.ac.id, andri@uii.ac.id

1. INTRODUCTION

In today's Information Technology (IT) era, higher education institutions are increasingly turning to virtual services such as enterprise information systems, Zoom, Panopto, and similar services. Universitas Islam Indonesia (UII) is one institution that practices this approach. They use many different online services to support their business process. The students, the staff, and the lecturer are obliged to follow the University's policy regarding their online services, especially during COVID-19 Pandemic [1][2]. With different kinds of services offered by the University, students with the most numbers often log into multiple online services with different credentials on different services [3]. This leads to multiple problems, such as security issues with password leaks or password reuse [4]. Sometimes, students also have difficulty logging into their account, which is separated into many different web services due to server load problems. This issue is commonly solved with centralized authentication and user account management using the Single Sign-On (SSO) system [5]. The board of information systems of UII or Badan Sistem Informasi (BSI) implements this method. However, the problem persists because the server load is too high to handle. When multiple students, staff, and lecturer login into their accounts from home, they create a spike of user login that happens in a short time. Therefore, it needs a new authentication and authorization management system. One of them is called

Shibboleth IdP. Shibboleth IdP is one of the many centralized SSO management systems that could be run on the Kubernetes clustered system.

The Kubernetes technology is used in many other cases. Konev et al., in their research, propose an algorithm for Persistent Volumes auto-scaling in a container orchestration system Kubernetes [6]. Auto-scaling stands for decreasing/increasing target resources depending on some load. In horizontal Volume scaling, they vary the number of Volumes, while in the vertical case, they vary the size of each Volume. They also introduce mixed scaling that includes both to reach the desired state. However, there is no explanation regarding persistent storage using clustering or redundancy methods implemented on Kubernetes. In this research, however, the container did not contain a persistent data file by default. With Kubernetes clustering service, both RAM and CPU become one and redundant. Previously, UII used TrueNAS Democratic Container Storage Interface (CSI) as the storage centre for all services inside Kubernetes. However, because of the storage's architecture, that can not be clustered into two or more physical hardware server machines separately-moreover, UII chose Longhorn instead. The Longhorn service was implemented in this research to make the storage system persistent and has a replication function.

The research conducted by Lobus Mercl et al. explained how containers function within the Kubernetes system. The storage is connected to containers via volumes, and many types of Volumes can be mounted to containers. There can be used volumes for block storage through dynamic provisioning, but only a few technologies for public cloud solutions can be used [7]. Lobus Mercl conducted testing on several storage classes that were deployed on the Microsoft Azure Kubernetes Service Server. AWS cloud volume, which acts as its cloud storage, was mapped into the instance (via Azure hostPath with attached Azure managed disk), OpenEBS, Portworx, Gluster with Heketi, and Ceph with Rook. According to the information from that paper, this research then conducts several tests to determine which stacked technology will be the most compatible to be implemented into the Kubernetes Cluster on-premise server deployed on BSI. The result is that the Longhorn technology is the most compatible. That is because its default function is similar to OpenEBS.

In the research conducted by John Watt et al., they explained the ability of Shibboleth to transmit extra information about a user, including licenses, roles, and other attributes [8]. However, this is not exploited for many reasons, mainly because institutional Identity Providers (IdPs) are not maintainable sources of fine-grained authorization information. The JISC-funded Shintau project has extended the Shibboleth profile, allowing users to link information from more than one IdP together utilizing a custom Linking Service (LS). With this research, the authors, in collaboration with the UII Board of Information Systems or Badan Sistem Informasi (BSI), conducted an activity to invite other universities in Indonesia to participate in a federation to use this type of service on the https://federasi.id website [9]. Providing seamless access to academic e-resources, e-libraries, cloud service providers, and many other partners [10]. Indonesia Access Federation (Federasi) is a service for research and education network institutions in Indonesia (including but not limited to universities, schools, and research institutes). One of its benefits will be for Academic Staff, Researchers, and Students: Easy access to any organization's subscribed e-resources or to any research education network institution worldwide that provides e-resources via access federation [11][12].

This research offers a solution by implementing a clustered service using Shibboleth IdP to act as the centralized SSO management system for the Kubernetes server cluster to solve the server load problem [13]. The implementation of Shibboleth IdP must also be paired with powerful storage technology called Persistent Volume Longhorn. This storage technology offers a replication function that helps the server minimize metadata loss when an online server is used. Persistent Volume Longhorn also creates redundancy data into two or more separate physical servers to avoid the risk of damaged data during its process. This research implemented Shibboleth IdP as its Single Sign On Management System. This management system runs on the Kubernetes cluster with the help of Longhorn technology as its reliable storage system. This combination to create one reliable and working ecosystem has not been conducted by any current research. Since its initial release, Shibboleth has been based on SAML, which can be deployed on virtual machines. Shibboleth also evaluates the SAML protocol, an open standard widely used for Single Sign On (SSO). SAML communicates between service providers or SPs that act as identity consumers of IdPs or identity providers using an extensible markup language (XML). The authentication protocol eliminates the need for passwords by relying on the security token of an encrypted and digitally assigned XML certificate [14]. The development of the Shibboleth IdP on the Kubernetes system also lets all services, data sources, and the Kubernetes cluster be combined as one.

2. METHODS

The process and steps to understand and conclude this research include understanding the concept of Single Sign On, Shibboleth IdP, Kubernetes, and choosing a reliable persistent storage volume technology.

This research also uses a sequence of methods to conclude its process to produce its results. The steps are as follows:

a) Observation

The first sequence of the method to complete this research is observation. This phase aims to collect the data needed to further process and analyze it. The observation phase is done with direct observation and trying out various techniques to be implemented to solve the problem presented in the environment of the UII.

b) Literature Review

The literature review aims to collect all the information and various research material regarding the research topic to ease up the analysis process further and give more comprehensive insight to the researcher on the problem and other variables. The literature review can be done by finding various references and research material from a book, journal, scientific paper, consultation with experts, internet, and other ways.

c) Data Analysis

Data and information acquired from the observation will be analyzed and processed further to gain a meaningful end result needed to build architecture services that are suitable for the enterprise system. On this topic, the service that will be implemented to solve the problem presented in the environment of the UII is the proper authentication & authorization service.

d) Architecture System Design

Following the results from the data analysis that was conducted, the architecture infrastructure service can be decided. This design is critical to test the solution resulting from all other sequential thinking methods from the start-up until this phase.

e) Implementation

Finally, we can implement the solution according to the proposed architecture service design and all the data analysis conducted. The implementation phase will be focused on the product service part to answer all the research problems.

2.1. Single Sign On (SSO)

Single Sign On (SSO) is a method that allows the user to log in on various sites, services, applications, and other things that are integrated into the SSO service [15]. With this SSO technology that acts as the authentication service, the users will only require to use one account to access different services or applications [16][17]. This will ease the user and eliminate or minimize the risks of forgetting their usernames and/or passwords [18][19]. The enterprise organization will also get various benefits, such as eliminating the need to develop an authentication service separately for each service, cutting off significant expenses regarding the development process, and more. The login process using SSO technology can be observed in Fig. 1.

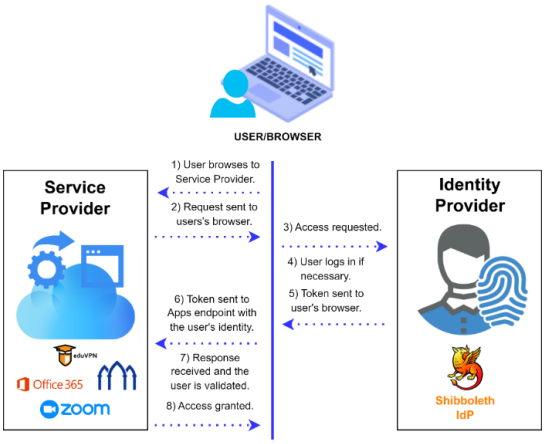

Fig 1. Flow user login with Single Sign On

2.2. Shibboleth IdP

Shibboleth implements the SAML authentication protocol, based on tokens generated by XML encryption and optionally has experience running server-side Java web applications [20]. SP will be able to customize the attributes required by user applications sourced from various databases by using Shibboleth IdP. Furthermore, with the SSO authentication method, many organizations can securely share identities within a federation in the future [21]. Following the authentication process, a strategy is required to increase IdP resilience, in this case, by minimizing downtime on IdP-connected services [22]. One is to use Kubernetes to orchestrate a container-based architecture with the load-balancing method [23].

The Shibboleth Consortium is currently managing the project by developing a product that allows for more secure access, is free to use, and integrates with the primary data source via a protected system. Shibboleth's most popular software components are Shibboleth Identity Provider (IdP) and Shibboleth Service Provider (SP), both of which implement the Security Assertion Markup Language (SAML) protocol [21]. Shibboleth software has been based on SAML since its inception. Later, this software was rewritten to compensate for the missing features in SAML 1.1. As a result, Shibboleth also contributes to evaluating the SAML protocol, an open standard widely used for SSO. SAML uses an extensible markup language (XML) for communication between identity consumers and service providers or SPs acting as identity consumers and IdPs or identity providers. The authentication protocol does not require passwords because it is based on a secure token of an encrypted and digitally assigned XML certificate [24]. As the identity owner organization, higher education can improve services from security and management of user data access by implementing this protocol, reducing abuse of access rights to parties who are not authorized to use the data [25].

2.3. Kubernetes

In terms of information technology infrastructure, Kubernetes is becoming an intriguing thing. Because from its inception to the present, Kubernetes has created an ecosystem that makes web applications serverless [26]. So that the server's physical location does not impede companies or organizations that want to deploy an application system. As a result, users of Kubernetes will find it more manageable [27].

Kubernetes has several components that complement each other, as shown in Fig. 2. Starting with the Primary Node, an etcd serves as a consistent key-value store for the Kubernetes cluster data store. Furthermore, there is a control plane component in charge of observing new Pods that have not been placed in any node and then selecting the node as a location for the new pod to run [28]. In Kubernetes, a pod represents the deployment unit of an application instance, which can be a single container or a group of containers that can share resources [6] [29]. Due to the need to store metadata on each pod connected to the replication of block storage on the Longhorn, a Shibboleth IdP container is used in this study, forming a stateful set [30]. Shibboleth IdP is thus run on three containers in the form of pods, as shown in Fig. 9, to maintain the application's level of resilience.

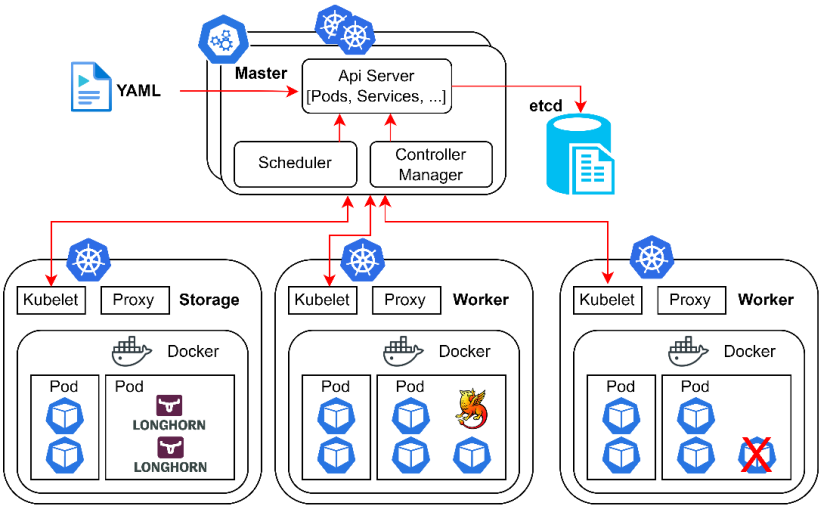

Fig. 2. Kubernetes (k8s) Cluster Ecosystems

An appropriate load-balancing method is required with the Shibboleth IdP replication process into three pod container replications. This study uses Nginx in conjunction with a round-robin algorithm [31]. In the case of implementing SSO Management with Shibboleth IdP, this replication process may be a solution to the SPOF challenge [32]. However, to determine the level of resilience of IdP services, trials with a container architecture, as described in this study, must be conducted [33].

2.4. Longhorn for Storage

Longhorn is the official open-source application development project of the Cloud Native Computing Foundation (CNCF). Longhorn is a cloud-native distributed storage platform for Kubernetes that is reliable and deployable. Longhorn makes it simple, fast, and reliable to deploy persistent block storage on Kubernetes clusters [34]. This storage block communicates via the iSCSI protocol with the storage server hardware [35]. It is critical to pay attention to the use of storage server hardware and the network server configuration, as both

significantly impact the user experience when using Longhorn [36]. The Longhorn's architecture is represented in Fig. 3.

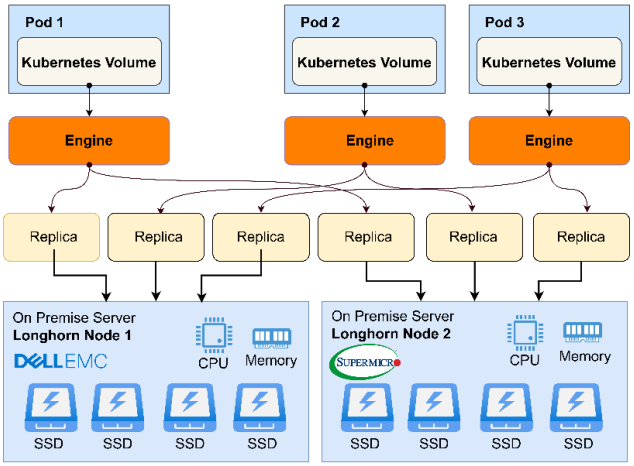

Fig. 3. Architecture block storage Longhorn

2.5. TrueNAS Democratic CSI

Network-Attached Storage (NAS) is a type of computer data storage linked to a network and provides data access to various clients. NAS systems are networked appliances containing one or more hard drives, frequently organized into logical, RAID redundant storage containers [37]. However, it cannot replicate on separate server hardware [38]. Network-attached storage relieves other servers on the network of the responsibility of file serving. Compared to file servers, network-connected storage provides faster data access, easier administration, and simple configuration via TrueNAS and must use Democratic CSI to communicate with Kubernetes.

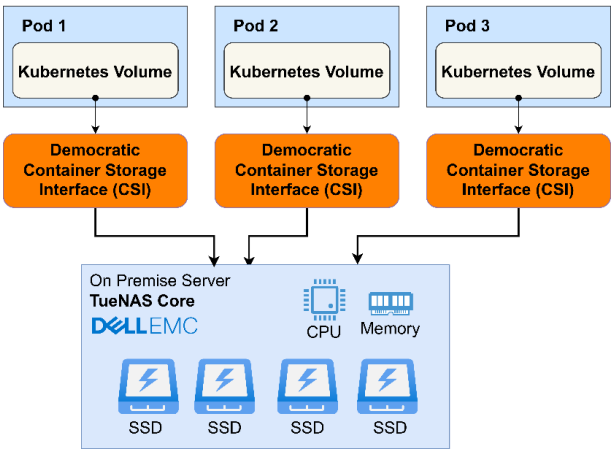

Fig. 4. Architecture TrueNAS Democratic CSI

3. RESULTS AND DISCUSSION

Shibboleth IdP is primarily used as single sign-on management for services connected to SSO, as illustrated in Fig. 5. UII centralized all authentication and authorization processes on Shibboleth IdP in the previous implementation of SSO architecture. However, the architecture was down a few times due to a failure to match metadata on the Shibboleth IdP service. This is due to an unreplicated block storage failure. As a result, replication solutions must be tested and implemented to keep Shibboleth IdP services operational while minimizing the impact on SP services connected to Shibboleth IdP.

Shibboleth IdP connects to multiple user requirements and attributes data sources, including Oracle, Percona database, and Active Directory. Shibboleth IdP can be accessed via a public internet domain via an Nginx reverse proxy [39]. Some Service Providers, such as UIIGateway (https://gateway.uii.ac.id) and eduVPN, are already connected to Shibboleth IdP. Service Providers from external services such as Google, Office365, Panopto, and Zoom application logins are also available.

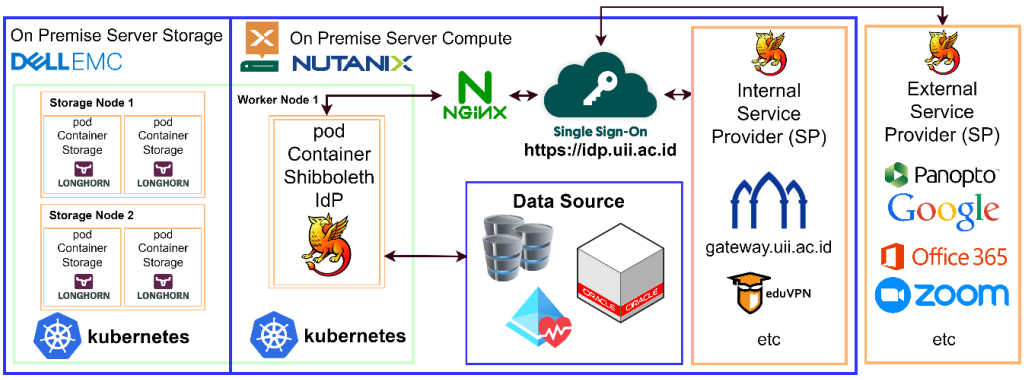

Fig. 5. Flow of communication between Shibboleth IdP services

Shibboleth IdP logins are tested using JMeter as a load testing tool or application load testing. The following is the result of resource usage on Shibboleth IdP after concurrent testing with up to 2000 users. Because the server hardware device used can affect the value generated on the Shibboleth IdP service, the results of this test may differ from those of other case studies. According to the metrics, the disk I/O bandwidth usage of the Shibboleth IdP service is less than 1Mbps. The use of non-replicated persistent storage for its implementation, as in TrueNAS persistent storage, increases the risk of a single point of failure. As a result, another solution for implementing persistent storage in Kubernetes that allows for replication or redundancy is required, namely Longhorn persistent storage based on block storage. Table 1, Table 2, and Table 3 display the performance test results of TrueNAS and Longhorn.

Table 1, Table 2, and Table 3 show performed trials using a tool called *fio*, that the performance of persistent storage between TrueNAS and Longhorn with the same hardware; the mean of the random read and random write from TrueNAS is faster compared to Longhorn, even though on the implementation (according to Fig. 6), Shibboleth IdP only needs a bandwidth for less than 1Mbps. There are *iodepth* variables in integer units on the testing, which means several I/O units to keep in flight against the file. That is the amount of outstanding I/O for each thread. A *bs* variable defines the block size the test will generate the I/O. The default value is 4k; if not specified, the test will use the default value. It is always recommended to specify the block size because the applications do not commonly use the default 4k. Also, the size variable can be defined as the file size on which the Fio will run the benchmarking test. Variable runtime can be defined as the amount of time the test will run in seconds.

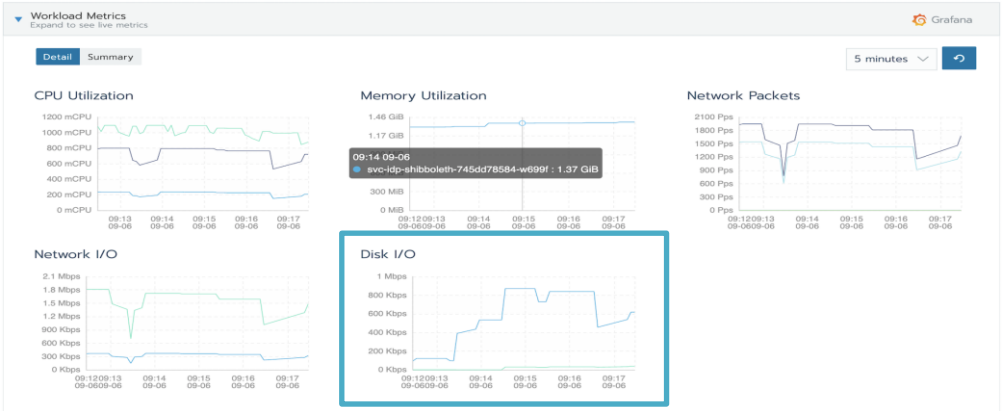

Fig. 6. The usage of resource server on Shibboleth IdP with Kubernetes cluster

We can see that both TrueNAS and Longhorn are still enough to fulfil the bandwidth storage in Shibboleth IdP. However, while using TrueNAS, it will be limited only to single storage and cannot be replicated. In order to compensate for that, the choice of storage technology applied to the Shibboleth IdP service is highly crucial because this service is becoming the centre of the business process, and downtime is not allowed. It requires a unique method to improve the current server to optimize storage. Through Kubernetes and distributed storage, a high-availability server system has been developed in this research [40]. Aside from implementing the Kubernetes that orchestrate containers, we also shall consider the persistent storage replication used in Kubernetes like Longhorn, as shown in Fig. 7.

Table 1. Storage Performance Evaluation random read and write

Environment	-iodepth=128 --bs=4k -size=1G -runtime=60	
	randread	**randwrite**
TrueNAS Core Storage with CSI Democratic	IOPS = 30,1k BW = 118MiB/s (123MB/s)(7051MiB/60003msec)	IOPS=881 BW=3525KiB/s (3609kB/s)(207MiB/60209msec)
Longhorn Storage	IOPS = 9145 BW = 35,7MiB/s (37,5MB/s)(2144MiB/60013msec)	IOPS = 731 BW = 2927KiB/s (2997kB/s)(172MiB/60034msec)

Table 2. Storage Performance Evaluation random read and write

Environment	-iodepth=128 --bs=1M -size=1G -runtime=60	
	randread	**randwrite**
TrueNAS Core Storage with CSI Democratic	IOPS = 428 BW = 429MiB/s (450 MB/s)(25.2 GiB/60215msec)	IOPS = 103 BW = 140MiB/s (109 MB/s)(6339 MiB/61064msec)
Longhorn Storage	IOPS = 259 BW = 260MiB/s (273MB/s)(15,3GiB/60394msec)	IOPS = 76 BW = 76,0MiB/s (79,7MB/s)(4578MiB/60211msec)

Table 3. Storage Performance Evaluation random read and write

Environment	-iodepth=1 -bs=4K -size=1G -runtime=60	
	randread	**randwrite**
TrueNAS Core Storage with CSI Democratic	IOPS=3340 BW=13,0MiB/s (13,7MB/s)(783MiB/60001msec)	IOPS=416 BW=1666KiB/s (1706kB/s)(97,6MiB/60002msec)
Longhorn Storage	IOPS=545 BW=2182KiB/s (2234kB/s)(128MiB/60002msec)	IOPS=84 BW=336KiB/s (344kB/s)(19,7MiB/60008msec)

According to Table 1, the test result showed that IOPS score on random read by using TrueNAS Democratic CSI can reach 30,1k, and random write score up to 881, while on the other hand, when using Longhorn, the IOPS score on random read can reach 9145 and random write score of 731. On Table 2 the bandwidth random write score can reach 140MiB/s (109 MB/s)(6339 MiB/61064msec), while on Longhorn random read can reach bandwidth 260MiB/s (273MB/s)(15,3GiB/60394msec) and random write bandwidth 76,0MiB/s (79,7MB/s)(4578MiB/60211msec). While on Table 3, each IOPS score and the bandwidth can be seen that the score obtained by TrueNAS is bigger than Longhorn. Both storage technologies got their strengths and weaknesses, such as TrueNAS can handle more IOPS than Longhorn and faster than Longhorn. At the same time, Longhorn is focused on its replication level and data redundancy to guarantee its service reliability maintained during hardware failure. In this study, Longhorn could answer the challenge of the existence of hardware failure that used to happen in UII. After several trials with Kubernetes Persistent Storage, Longhorn showed a more stable result. This performance testing result could be different if we compared it to other studies. The difference can result from the different use of hardware or the network link that has already been implemented in the data centre.

Fig. 8 depicts the results of a hardware failure test with a storage status of two failures. However, because it uses Longhorn persistent storage with storage replication, the Shibboleth IdP service continues to function normally, as shown in Fig. 9.

After the risk of the single point of failure on the storage is handled by using replication features on block storage Longhorn, the next step is the monitoring of the request and response on the load-balancing method as described in Table 4.

According to Table 4, we can see that there is no error 5xx significant on the Shibboleth IdP services, yet the error is often experienced while the load request is increasing. Some errors occurred because of the failure to obtain the metadata. This can be caused by problems with the storage service that used TrueNAS Democratic CSI with single storage architecture, so this can cause a single point of failure on the persistent storage. In order to prevent this constraint, this study offers the trial solution; making a persistent storage replication on Kubernetes by using Longhorn, as seen in Fig. 7.

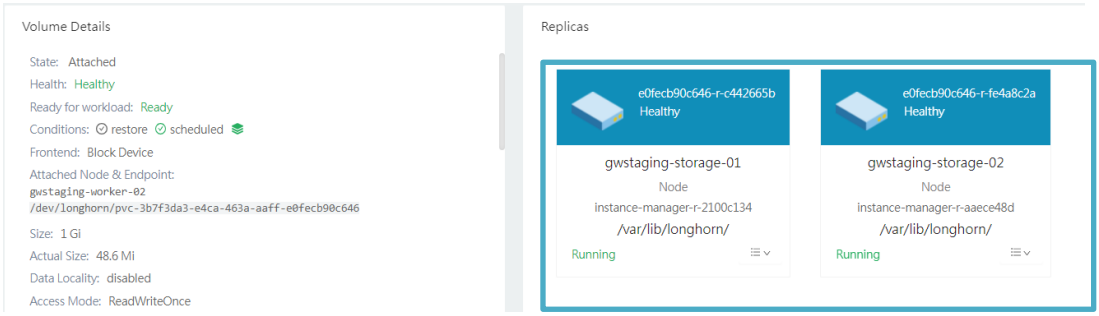

Fig. 7. The usage of replication in block storage Longhorn

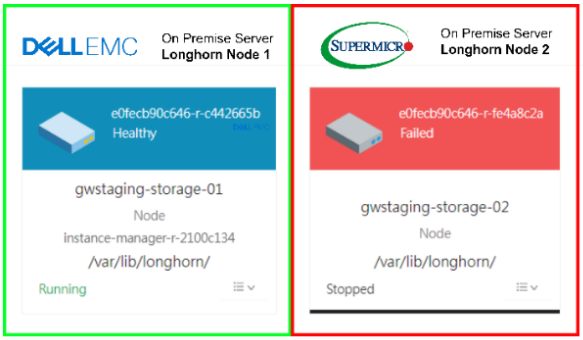

Fig. 8. One of the storage machines was subjected to a hardware failure test using block storage Longhorn

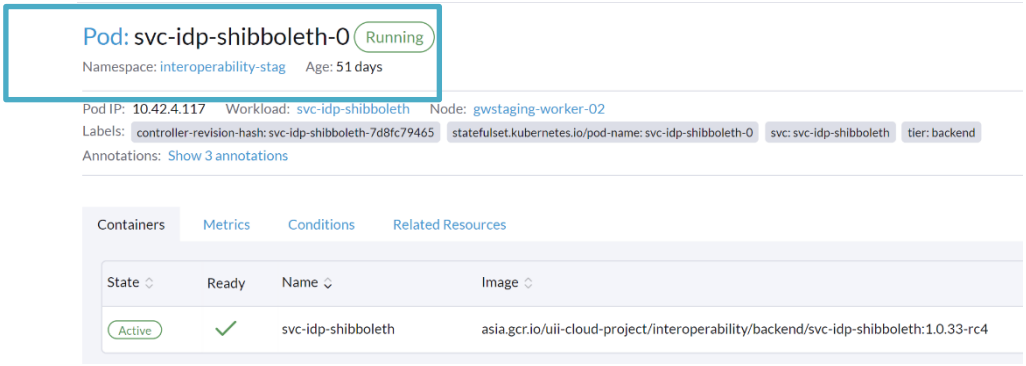

Fig. 9. Shibboleth IdP service is still operational

Table 4. Nginx IdP Shibboleth Load balancing

Server	State	Response Time	Response	
			2xx	5xx
Worker-01	UP	0ms	282144	0
Worker-02	UP	0ms	278691	0
Worker-03	UP	24ms	330677	1
Worker-04	UP	0ms	258052	0
Worker-05	UP	0ms	290366	0
Worker-06	UP	1ms	213927	1
Worker-07	UP	23ms	360018	0
Worker-08	UP	0ms	365679	0

4. CONCLUSION

Based on the research, we can conclude that the container conducted by Kubernetes only is not adequate to prevent a Single Point of Failure. Therefore, we also use block storage Longhorn which has a replication feature in this research. Shibboleth IdP that saves metadata can still be accessed even when the malfunction is found in one of the physical server's machines. Furthermore, the computation machine usage is already handled by Kubernetes, and the storage used is already handled by block storage Longhorn. Using load-balancing using Nginx based on a round-robin algorithm, resulting in the small resource server and low request inside the node

server, can be made relatively with little error responses with the value 5xx error responses [41]. In order to obtain a valid result of the test storage performance, at least we should use minimal SSD, or we better use the NVMe [42]. Another recommendation is that if we already successfully implement the Longhorn on Kubernetes by collating several hardware servers in one data centre location, another challenge is implementing the Datacenter Recovery Centre (DRC). In this case, we make separate and different redundant data centres between the primary data centre and backup data centre that can be used to minimalize downtime in case of a disaster in one data centre location that may cause the loss of electricity and internet access [43].

Acknowledgments

We want to thank our friends and colleagues at Badan Sistem Informasi, who helped the authors conduct the research. We hope that this paper could be beneficial for developing robust and reliable information technology infrastructures.

REFERENCES

[1] Y. B. Hermanto and V. A. Srimulyani, "The Challenges of Online Learning During the Covid-19 Pandemic," *J. Pendidik. dan Pengajaran*, vol. 54, no. 1, p. 46, Mar. 2021, https://doi.org/10.23887/jpp.v54i1.29703.

[2] D. Surani and H. Hamidah, "Students Perceptions in Online Class Learning During the Covid-19 Pandemic," *Int. J. Adv. Sci. Educ. Relig.*, vol. 3, no. 3, pp. 83–95, Nov. 2020, https://doi.org/10.33648/ijoaser.v3i3.78.

[3] I. Nongbri, P. Hadem, and S. Chettri, "A SURVEY ON SINGLE SIGN ON," *Int. J. Creative Res. Thoughts*, vol. 6, no. 2, pp. 595-602, May 2018, https://www.researchgate.net/publication/325119173_A_Survey_on_Single_Sign-On.

[4] B. Cusack and E. Ghazizadeh, "Evaluating single sign-on security failure in cloud services," *Bus. Horiz.*, vol. 59, no. 6, pp. 605–614, 2016, https://doi.org/10.1016/j.bushor.2016.08.002.

[5] I. P. A. Pratama, Linawati, and N. P. Sastra, "Token-based single sign-on with JWT as information system dashboard for government," *Telkomnika (Telecommunication Comput. Electron. Control.*, vol. 16, no. 4, pp. 1745–1751, Aug. 2018, https://doi.org/10.12928/telkomnika.v16i4.8388.

[6] I. Konev, I. Nikiforov, and S. Ustinov, "Algorithm for Containers' Persistent Volumes Auto-scaling in Kubernetes," *2022 31st Conf. Open Innov. Assoc.*, pp. 1–7, Apr. 2022, doi: 10.23919/FRUCT54823.2022.9770916.

[7] L. Mercl and J. Pavlik, "Public Cloud Kubernetes Storage Performance Analysis," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11684, pp. 649–660, 2019, doi: 10.1007/978-3-030-28374-2_56.

[8] J. Watt, R. O. Sinnott, G. Inman, and D. Chadwick, "Federated authentication and authorization in the Social Science domain," *Proc. 2011 6th Int. Conf. Availability, Reliab. Secur. ARES 2011*, pp. 541–548, 2011, https://doi.org/10.1109/ARES.2011.83.

[9] L. Ramamoorthi and D. Sarkar, "Single Sign-on Implementation: Leveraging Browser Storage for Handling Tabbed Browsing Sign-outs," *undefined*, vol. 152, pp. 15–28, 2019, https://doi.org/10.1007/978-981-13-9155-2_2.

[10] F. Jayakanth, A. T. Byrappa, and R. Visvanathan, "Off-campus Access to Licensed Online Resources through Shibboleth," *undefined*, vol. 40, no. 2, Jun. 2021, https://doi.org/10.6017/ital.v40i2.12589.

[11] N. Naik and P. Jenkins, "Securing digital identities in the cloud by selecting an apposite Federated Identity Management from SAML, OAuth and OpenID Connect," *undefined*, pp. 163–174, Jun. 2017, https://doi.org/10.1109/RCIS.2017.7956534.

[12] H. Yildiz, C. Ritter, L. T. Nguyen, B. Frech, M. M. Martinez, and A. Kupper, "Connecting Self-Sovereign Identity with Federated and User-centric Identities via SAML Integration," *undefined*, vol. 2021-September, 2021, https://doi.org/10.1109/ISCC53001.2021.9631453.

[13] N. D. Nguyen and T. Kim, "Balanced Leader Distribution Algorithm in Kubernetes Clusters," *undefined*, vol. 21, no. 3, pp. 1–15, Feb. 2021, https://doi.org/10.3390/s21030869.

[14] G. Mamidisetti, R. Makala, and C. Anilkumar, "A novel access control mechanism for secure cloud communication using SAML based token creation," *undefined*, 2020, https://doi.org/10.1007/s12652-020-02427-8.

[15] N. M. Karie, V. R. Kebande, R. A. Ikuesan, M. Sookhak, and H. S. Venter, "Hardening SAML by Integrating SSO and Multi-Factor Authentication (MFA) in the Cloud," *undefined*, Mar. 2020, https://doi.org/10.1145/3386723.3387875.

[16] O. Mir, M. Roland, and R. Mayrhofer, "Decentralized, Privacy-Preserving, Single Sign-On," *Secur. Commun. Networks*, vol. 2022, 2022, https://doi.org/10.1155/2022/9983995.

[17] B. Alemu, R. Kumar, D. Sinwar, and G. Raghuwanshi, "Fingerprint based authentication architecture for accessing multiple cloud computing services using single user credential in IOT environments," *J. Phys. Conf. Ser.*, vol. 1714, no. 1, Jan. 2021, https://doi.org/10.1088/1742-6596/1714/1/012016.

[18] Y. Zhang, C. Xu, H. Li, K. Yang, N. Cheng, and X. Shen, "PROTECT: Efficient Password-Based Threshold Single-Sign-On Authentication for Mobile Users against Perpetual Leakage," *undefined*, vol. 20, no. 6, pp. 2297–2312, Jun. 2021, https://doi.org/10.1109/TMC.2020.2975792.

[19] A. Syarif, A. A. Nazmi, M. N. Soleh, A. A. Sukmandhani, and J. Ohliati, "Implementation of Single Sign-on System for User Management at PT.XYZ," *In 2022 International Conference on Information Management and Technology (ICIMTech)*, pp. 67-72, 2022, https://doi.org/10.1109/ICIMTech55957.2022.9915069.

[20] S. Michael and Z. J. Anna, "An Identity Provider as a Service platform for the eduGAIN research and education

community," *In 2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pp. 739-740, 2019. https://ieeexplore.ieee.org/abstract/document/8717796.
[21] A. S. Shehu, A. Pinto, and M. E. Correia, "Privacy Preservation and Mandate Representation in Identity Management Systems," *In 2019 14th Iberian Conference on Information Systems and Technologies (CISTI)*, pp. 1-6, 2019, https://ieeexplore.ieee.org/abstract/document/8760690.
[22] N. Hamamoto *et al.*, "Toward the Cross-Institutional Data Integration From Shibboleth Federated LMS," *Procedia Comput. Sci.*, vol. 159, pp. 1720–1729, Jan. 2019, https://doi.org/10.1016/j.procs.2019.09.343.
[23] K. Takahashi, K. Aida, T. Tanjo, and J. Sun, "A Portable Load Balancer for Kubernetes Cluster," *In Proceedings of the International Conference on High Performance Computing in Asia-Pacific Region*, pp. 222–231, Jan. 2018, https://doi.org/10.1145/3149457.3149473.
[24] S. Chiasson, E. Stobert, A. Forget, R. Biddle, and Van Oorschot, P. C. , "Persuasive cued click-points: Design, implementation, and evaluation of a knowledge-based authentication mechanism," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 2, pp. 222-235, 2011, https://ieeexplore.ieee.org/abstract/document/6065736.
[25] S. Binu, M. Misbahuddin, and P. Raj, "A Strong Single Sign-on User Authentication Scheme Using Mobile Token Without Verifier Table for Cloud Based Services," *undefined*, pp. 237–261, Aug. 2018, https://doi.org/10.1007/978-3-319-58424-9_14.
[26] H. Govind and H. González–Vélez, "Benchmarking Serverless Workloads on Kubernetes," *undefined*, pp. 704–712, May 2021, https://doi.org/10.1109/CCGrid51090.2021.00085.
[27] Z. Zhong and R. Buyya, "A Cost-Efficient Container Orchestration Strategy in Kubernetes-Based Cloud Computing Infrastructures with Heterogeneous Resources," *ACM Trans. Internet Technol.*, vol. 20, no. 2, Apr. 2020, https://doi.org/10.1145/3378447.
[28] V. Medel, O. Rana, J. Á. Bañares, and U. Arronategui, "Modelling performance & resource management in Kubernetes," *Proc. - 9th IEEE/ACM Int. Conf. Util. Cloud Comput. UCC 2016*, pp. 257–262, 2016, https://doi.org/10.1145/2996890.3007869.
[29] N. Zhou *et al.*, "Container orchestration on HPC systems through Kubernetes," *J. Cloud Comput.*, vol. 10, no. 1, Dec. 2021, https://doi.org/10.1186/s13677-021-00231-z.
[30] L. A. Vayghan, M. A. Saied, M. Toeroe, and F. Khendek, "A Kubernetes controller for managing the availability of elastic microservice based stateful applications," *J. Syst. Softw.*, vol. 175, 2021, https://doi.org/10.1016/j.jss.2021.110924.
[31] P. Mohan*, T. Jambhale, L. Sharma, S. Koul, and S. Koul, "Load Balancing using Docker and Kubernetes: A Comparative Study," *Int. J. Recent Technol. Eng.*, vol. 9, no. 2, pp. 782–792, 2020, https://doi.org/10.35940/ijrte.B3938.079220.
[32] I. P. Windasari, L. S. Nugroho, A. F. Rochim, and R. Septiana, "An-SPf: as an Alternative Architecture of no Single Point Failure of Scalable Transcoding System Based on Kubernetes," *undefined*, pp. 18–23, 2021, https://doi.org/10.35940/ijrte.B3938.079220.
[33] W. Ren, W. Chen, and Y. Cui, "Dynamic Balance Strategy of High Concurrent Web Cluster Based on Docker Container," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 466, no. 1, 2018, https://doi.org/10.1088/1757-899X/466/1/012011.
[34] Y. Wang, S. Cheng, X. Zhang, J. Leng, J. Liu, and F. Pozo, "Block Storage Optimization and Parallel Data Processing and Analysis of Product Big Data Based on the Hadoop Platform," *Math. Probl. Eng.*, vol. 2021, 2021, https://doi.org/10.1155/2021/3839800.
[35] J. Liao *et al.*, "Toward Efficient Block Replication Management in Distributed Storage," *ACM Trans. Model. Perform. Eval. Comput. Syst.*, vol. 5, no. 3, Nov. 2020, https://doi.org/10.1145/3412450.
[36] T. Fukatani, H. H. Le, and H. Yokota, "Lightweight Dynamic Redundancy Control with Adaptive Encoding for Server-based Storage," *undefined*, vol. 17, no. 4, pp. 1–38, Nov. 2021, https://doi.org/10.1145/3456292.
[37] Y.-W. Wei, T. Zhu, and D. Zhang, "The Pennsylvania State University Schreyer Honors College Department of Computer Science and Engineering Tail Latency Admission Control For Raid Storage Systems," *Doctoral dissertation, PENN STATE*, 2020, https://honors.libraries.psu.edu/files/final_submissions/6772.
[38] M. Ramanathan and K. Narayanan, "Disk storage failure prediction in datacenter using machine learning models," *Applied Nanoscience*, pp. 1-22, 2021, https://doi.org/10.1007/s13204-021-02039-4.
[39] R. Yusuf, R. S. Pasha, and U. Nuha, "Comparative Analysis of HAProxy& Nginx in Round Robin Algorithm to Deal with Multiple Web Request," 2018, http://www.ijctjournal.org/Volume5/Issue6/IJCT-V5I6P13.pdf.
[40] A. A. Khatami, Y. Purwanto, and M. F. Ruriawan, "High Availability Storage Server with Kubernetes," *undefined*, pp. 74–78, Oct. 2020, https://doi.org/10.1109/ICITSI50517.2020.9264928.
[41] L. H. Pramono, R. C. Buwono, and Y. G. Waskito, "Round-robin Algorithm in HAProxy and Nginx Load Balancing Performance Evaluation: a Review," *In 2018 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, pp. 367–372, Nov. 2018, https://doi.org/10.1109/ISRITI.2018.8864455.
[42] D. Niu, H. Zhang, T. Cai, Z. Chen, Y. Zhan, and J. Liang, "DNVMCFS: The Direct Hybrid NVM File System for the Application," *Proc. - 20th Int. Conf. High Perform. Comput. Commun. 16th Int. Conf. Smart City 4th Int. Conf. Data Sci. Syst. HPCC/SmartCity/DSS 2018*, pp. 954–959, Jan. 2019, https://doi.org/10.1109/HPCC/SmartCity/DSS.2018.00158.
[43] N. Dhanujati and A. S. Girsang, "Data Center-Disaster Recovery Center (DC-DRC) for High Availability IT Service," *In: 2018 International Conference on Information Management and Technology (ICIMTech)*, pp. 55–60, Nov. 2018, https://doi.org/10.1109/ICIMTech.2018.8528170.

BIOGRAPHY OF AUTHORS

Ikhwan Alfath Nurul Fathony,S.Kom., ikhwan.alfath@uii.ac.id

Mukhammad Andri Setiawan, S.T.,M.S.c.,Ph.D., andri@uii.ac.id