Comparative Analysis of the Performance of Single Sign-On Authentication Systems with OpenID and OAuth Protocols

Application of Single Sign-On (SSO) in Information Systems at the University of Technology Yogyakarta

Tri Waluyo
Magister Program of Information Technology
University of Technology Yogyakarta
Yogyakarta, Indonesia
Email: triwaluyo [AT] student.uty.ac.id

Sutarman
Magister Program of Information Technology
University of Technology Yogyakarta
Yogyakarta, Indonesia
Email: sutarman [AT] uty.ac.id

***Abstract*— a vast number of people use the internet on a regular basis. The growing number of users will inadvertently bring new issues for both users and administrators as user managers. Users forget their user accounts and passwords when they have too many accounts to surf the internet. Web-based application services at University of Technology Yogyakarta include the Academic Information System (SIA) and E-Learning without exception. Both have the same issue: figuring out how to establish an authentication mechanism that will prevent users from forgetting their accounts on the system. The goal of this research is to create a prototype using Single Sign On (SSO) and compare the performance of the two SSO protocols utilized, OpenID and OAuth. The Explicate Problem, Define Requirements, Design and Develop Artifact, Demonstrate Artifact, Evaluate Artifact, and Communication processes are all part of this study. The results of prototype testing are obtained by attempting to log in using an academic service system account, and users are not required to login/authenticate again while accessing the e-learning page. Performance studies on both protocols revealed that the highest number of users who could login to the system at the same time was 1230 (OpenID) and 1219 (OAuth). In comparison to the OpenID protocol, the OAuth protocol is more consistent in terms of average response time for handling user requests. A greater specification is also required to suit the demands of additional users.**

Keywords-Single Sign On; OpenID; OAuth; Academic Information System

I. INTRODUCTION

University of Technology Yogyakarta (UTY) is a university that uses information technology, particularly the internet, to facilitate learning and academic administration. Because of the extensive usage of the internet, individuals frequently visit many services throughout the day, necessitating the use of multiple identities and passwords [1]. Some basic characteristics that the internet possesses and that many users desire are that it is simple and enjoyable to use, therefore making the internet more versatile. With the extensive use of the internet, the number of people who utilize it is growing. However, when the number of users grows, it will inadvertently create new issues for both the user and the administrator as a user manager.

Users suffer a challenge in that many forget their user accounts and passwords because they must remember many usernames and passwords to access into various services [2]. In this situation, UTY is one of the universities that offers a variety of web-based services to aid communication and information exchange for academic reasons. The Academic Information System (SIA) and E-Learning are two web-based application services offered by UTY.

With the growing number of services, it is obviously wasteful if a user must undertake a login or authentication process using many combinations of their username and password every time they access the administrative system of a service. LDAP is one of the authentication techniques available (Lightweight Directory Access Protocol). LDAP is a protocol for storing and retrieving data that works similarly to a relational database [1].

The fundamental distinction between LDAP and databases is that LDAP organizes information using a tree model, which allows it to provide a faster query service than relational databases. Because of this paradigm, LDAP is remarkably comparable to an organization's actual structure. It is intended that by employing this strategy, users will be able to solve the problem of multiple usernames and passwords while also gaining more convenience. When authenticating each application service, users must still enter their login and password.

This LDAP technique has a flaw in that it forces the user to authenticate for each application each time they want to access

the web-based application administration system. The user must still supply their username and password to each of the application services while using repeated authentication. Users are indirectly saturated by the number of login or authentication procedures available, or in other words, the number of application services provided.

An authentication system known as Single Sign On (SSO) was created to solve the shortcomings of the LDAP method. Single Sign On (SSO) is a technology that allows users on a network or system to access services with only one user account [3]. Users don't have to remember as many usernames and passwords with the Single Sign On (SSO) system, which makes data processing easier [4]. SSO is beneficial for logging user activity and monitoring user accounts, according to [5.] The SSO solution also lowers human error and eliminates the tedious procedure of logging in with a username and password.

There are numerous protocols for Single Sign On (SSO). OAuth2, SAML, and OpenID are common protocols used in Single Sign On (SSO) implementation. OAuth2 is an open protocol that enables safe authorization from online, mobile, and desktop apps using standard and simple techniques. There is no authentication layer in OAuth2. Authorization determines the breadth of use. With the authorization of the resource owner, OAuth2 allows the server to use limited resources [6]. Protocol vs. OAuth OAuth2 (Open Authorization) is an authentication protocol developed by an Application Programming Interface (API) service provider that allows users to get access to resources by converting usernames and passwords into tokens. Third-party programs can access data from built-in applications utilizing this interface [7].

The Organization for the Advancement of Information Standards (OASIS) developed the Security Assertion Markup Language (SAML), which is essentially a platform independent, non-proprietary protocol used to communicate between identity users and those who ordinarily do business. SSO Federation relies heavily on SAML. This allows domains with different authentication techniques to communicate with one another. Users, Identity Providers (IdPs), and Service Providers (SPs) are the entities engaged in SAML [8].

OpenID is a technology that allows users to authenticate themselves without having to go through a central server. The three major components of OpenID are: 1) OpenId Identifier, which is a text string or email address that identifies a user uniquely. 2) OpenID Relying Party (RP), which is a web application or service provider that requires confirmation that the identificaation belongs to the end user. 3) OpenID Provider (OP), which is a central server that issues, stores, and manages users' OpenID identifiers. Discovery, Authentication, Association, and Verification are the four basic methods utilized in the OpenID protocol.

Human errors caused by mistakenly entered user names or keywords will be reduced by the SSO system. SSO will also remove the tedious task of establishing identification using passwords or other authentication mechanisms on a regular basis [10].

The Single Sign On (SSO) modeling in the information system at the University of Technology Yogyakarta will be discussed in this research, which will use a protocol that is in compliance with the institution's demands and conditions. The first step for both IT developers and professionals in maintaining data security is to determine which standards must be adopted to keep the federation's identity secure. However, distinguishing between OAuth, OpenID, and SAML is not always straightforward; many IT professionals and developers struggle to do so. The federation procedure is organized by each standard.

II. METHODS

A. Research Design

The objective of this research is to compare how the OpenID and OAuth protocols were used to implement Single Sign On (SSO) on a prototype academic service system at the Yogyakarta University of Technology. This study aims to improve the organization and implementation of existing SSO at institutions. Because no other implementation framework exists, a new one must be created from scratch using design principles and scientific approaches. As a result, the design research approach was chosen as the most appropriate strategy for performing this study.

As indicated in Figure 1, this research comprises of various processes that are common in design research. These steps are: Explicate the Problem, Define the Requirements, and Implement the Solution. Artifact Design and Development, Artifact Demonstration, Artifact Evaluation, and Artifact Communication [11].

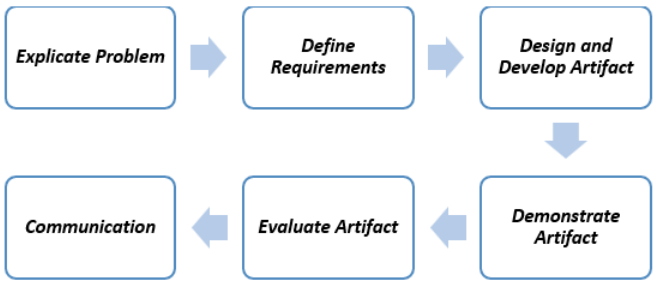


Figure 1. Stages of the Rsesearch

1) Explicate Problem

It is an iterative process that examines the existing knowledge base in order to gain a comprehensive grasp of the problems and possible solutions. Direct observation was used to conduct the problem analysis in this study. The existing teaching and learning service support system was observed and is continuing in operation.

SIA UTY and E-learning UTY are the systems utilized to assist teaching and learning activities at the University of Technology Yogyakarta. Both systems have not made use of SSO technology to take advantage of its benefits. Users of these services have separate accounts for each system, such as a SIA UTY account and an E-learning UTY account. The usage of two separate accounts to access academic system services makes it difficult for users to remember which

accounts they hold. In this instance, it is required to connect accounts that can provide services for signing in with accounts from other systems.

SSO is a type of authentication that allows users to safely log in to various apps and websites with only one account. Certificates are exchanged between identity providers and service providers to enable SSO. The certificate is used to sign the identity information that is delivered from the identity provider to the service provider, ensuring that the data is authentic. Identity data is stored in the form of a token that contains information like a user's e-mail address or username in SSO.

2) Define Requirements

This stage involves a deeper dive into the problem, resulting in a clear set of requirements that must be met in order to solve it. The features that an artifact should contain, as well as the restrictions that it will employ, are determined by the requirements analysis. This chapter's tools and research materials section will go over the system requirements.

3) Design and Develop Artifact

This step entails the creation of artifacts (prototypes) based on the needs analysis and knowledge base gathered during the previous two stages. The prototype only includes a login page for the UTY academic information system and a login page for UTY's e-learning system.

4) Demonstrate Artifact

At this point, the prototype is tested on one or more sample issues to demonstrate its utility. The prototype is used to login to SIA and UTY E-Learning using one of the Google, SIA UTY, or UTY E-Learning accounts.

5) Evaluate Artifact

This step involves critically assessing the prototype that has been created in order to determine the degree to which the requirements set in the needs analysis phase have been met. This study's evaluation was carried out through testing. Table I shows the results of many tests performed on the prototype that was produced.

TABLE I. PROTOTYPE TESTING

Type of Testing	Testing Goals
Load Testing	1. Software testing to test the scalability and speed of the system. 2. Simulate multiple users accessing Web services simultaneously.
Stress Testing	1. Software testing to test system stability and reliability. 2. Test the system beyond its normal operating point and evaluate how the system performs under extreme conditions. 3. Ensure that the system will not fail in critical situations.

6) Communication

The final phase in the research process is to disseminate the findings to others. A research report was chosen as the communication model in this study.

B. Research Tools and Materials

1) Hardware and Software

On the server side, the hardware and software utilized for prototype development are listed in Table II.

TABLE II. DEVICE NEED FOR SERVER

Requirement	Specification
Random Access Memory	6 GB
Processor	2 Core
Solid State Drive (SSD)	256 GB
Operating System	Windows 10 64 bit
Virtual System	Oracle VM Virtual Box
Virtual Server	Linux Debian

Hardware and software requirements for clients are intended for the testing process of developing a server that has been prototyped. Table 3.2 shows the device requirements, both hardware and software for the client.

TABLE III. TABLE III. DEVICE NEED FOR CLIENT

Requirement	Specification
Processor	2 *Core* (minimum) 4 *Core* (recommendation)
Hard Disk	100 MB (minimum) 1 GB (recommendation)
Memory	2 GB (minimum) 4 GB (recommendation)
Operating System	One of the operation system is as follows (minimum specification). *Desktop/Personal Computer*: 1. Microsoft Windows 8.1 (32-bit or 64-bit) 2. MacOS Catalina 3. SUSE Linux Enterprise Desktop 12 Service Pack4 4. Debian 9, 10 5. Ubuntu 16, or Ubuntu 18 *Smartphone*: 1. Android 8.1 2. Apple iOS 12

2) Research Object

The research was conducted with studies and samples taken from University of Technology Yogyakarta, which is located at Jl. Siliwangi (North Ringroad), Jombor, Sleman, Special Region of Yogyakarta. The object of the research is the academic community in Universityof Technology Yogyakarta who use SIA UTY and UTY e-learning, and have a Google account.

C. Prototype Architecture

Figure 3.2 depicts the OpenID and OAuth flow scenarios generated in this investigation on the prototype. The scenario is separated into two stages: logging in (entering SIA/E-Learning UTY) and logging out (using OpenID and OAuth).

The stages in the scenario for logging into SIA or UTY E-Learning, are explained as follows:

1. Users will use the services in the UTY Academic Information System (SIA UTY), so a login is required to enter the system.
2. SIA UTY sends an authorization request to OpenID Connect/OAuth.
3. OpenID Connect/OAuth authenticates the user and runs an SSO session.
4. OpenID Connect/OAuth sends an SSO session to use by SIA UTY.
5. SIA UTY responds by validating and using the token provided by OpenID Connect/OAuth.
6. SIA UTY runs its own local session.
7. Users can access the services provided by SIA UTY.

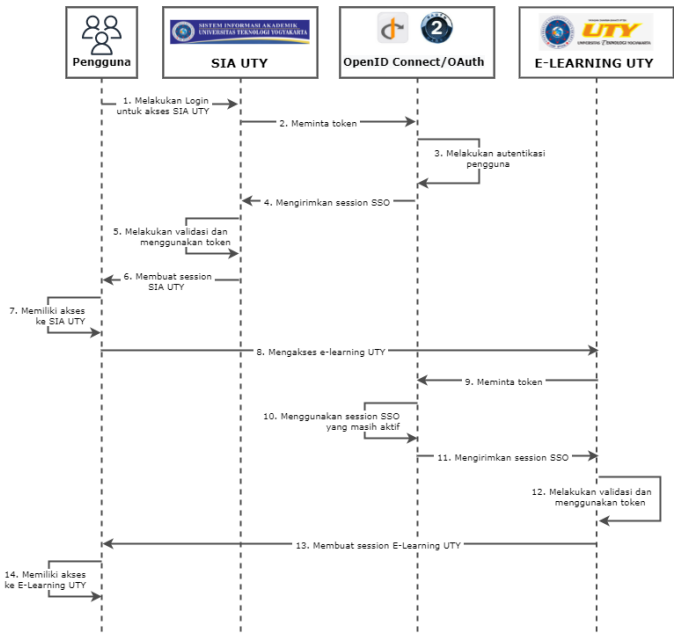


Figure 2. SSO Login Flow Scenario

8. When a user wants to access the UTY E-Learning system, the user is required to enter account information.
9. UTY e-learning system sends an OpenID Connect authorization request by requesting a token.
10. OpenID Connect/OAuth checks the session and finds that there is an active SSO session, so OpenID Connect/OAuth will use the active SSO session.
11. OpenID Connect/OAuth sends a new token for the UTY e-learning system.
12. UTY e-learning system validates and uses the new token to create a local session.
13. UTY e-learning system creates a local session so that users can access the services on the system.
14. Users can access UTY e-learning system services without logging in because the system already has account information from the SSO session.

The next step is the logging out procedure, which is seen in Figure 3 below.

1. Users who have used SIA UTY service, plan to logout from the system.
2. SIA UTY will terminate the local session assigned to the customer, and request OpenID Connect/OAuth to terminate the current SSO session.

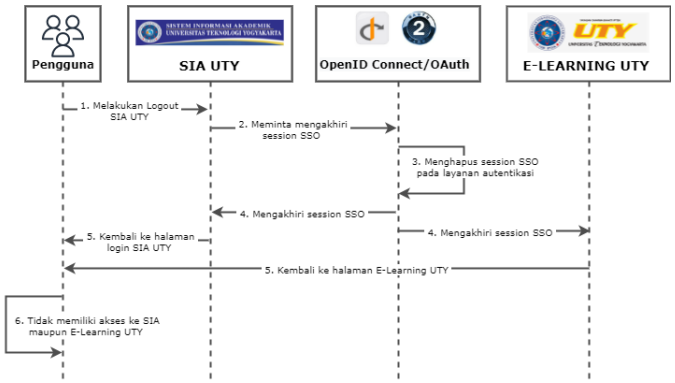


Figure 3. SSO Logout Flow Scenario

3. OpenID Connect/OAuth removes SSO on the authentication service.
4. OpenID Connect/OAuth terminates the running SSO session for all systems connected to the current authentication service (SIA UTY and e-learning UTY).
5. The system that is currently connected to the authentication service will display the login page again
6. Users no longer have access to system services, and are asked to re-enter account information, if they want to use the services in SIA UTY or e-learning UTY.

Based on this scenario, it is clear that both OpenID and OAuth can be used to check authentication (identification process), and provide authorization (granting permission to access data from several websites, without providing authentication information).

III. RESULTS

A. Server Prototype

The prototype server is created virtually using the oracle virtual box tools. The server consists of 4 service providers, namely Debian server, server 02, server 03, and IDP server as shown in Figure 4.

The debian server in the picture is used as a server for Domain Name Server (DNS), server 02 is used as a service provider server for academic information systems (sia.thesis.com), server 03 is a server for providing e-learning systems (elearning.thesis.com), and IDP server as a server for the user database.

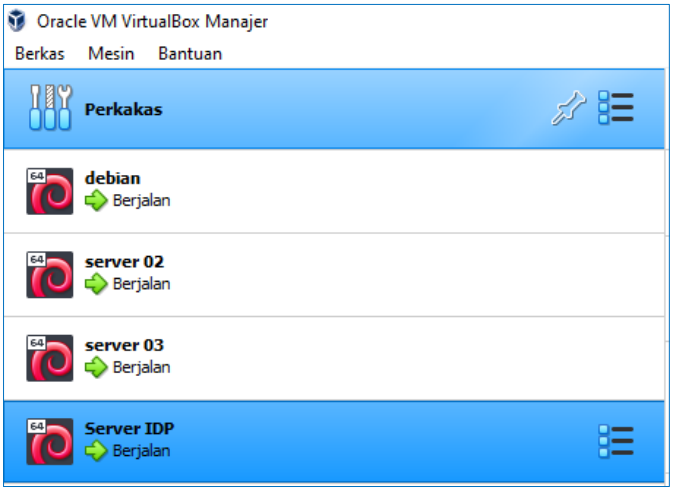


Figure 4. Prototype Server

B. System Prototype

The prototype system was built using the Hypertext Preprocessor (PHP) programming language. The prototype system is in the form of a login page on each academic service system, both SIA UTY and E-learning UTY. Figure 5 shows the login page on SIA UTY prototype.

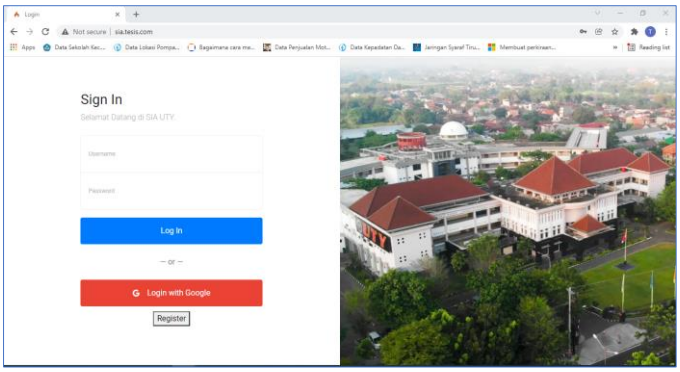


Figure 5. SIA UTY Prototype

On the SIA UTY login page, in addition to the button to login using an SIA account, there is also a menu (button) for logging in using a Google account.

The prototype developed is made as close as possible to the system used in the original. As shown in figure 6, UTY e-learning prototype page has been created as the original e-learning page.

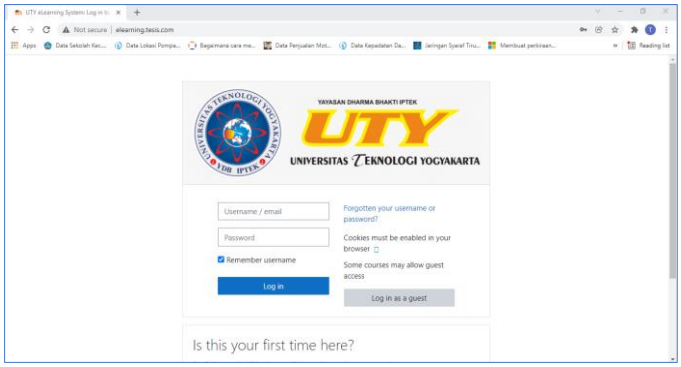


Figure 6. ELearning UTY Prototype

C. Performance Testing

The performance of each prototype, as well as the SSO approach, is then determined by testing the prototypes. Performance testing examines the software's and infrastructure's stability, scalability, dependability, speed, and resource utilization. JMeter tools are used to perform load testing and stress testing.

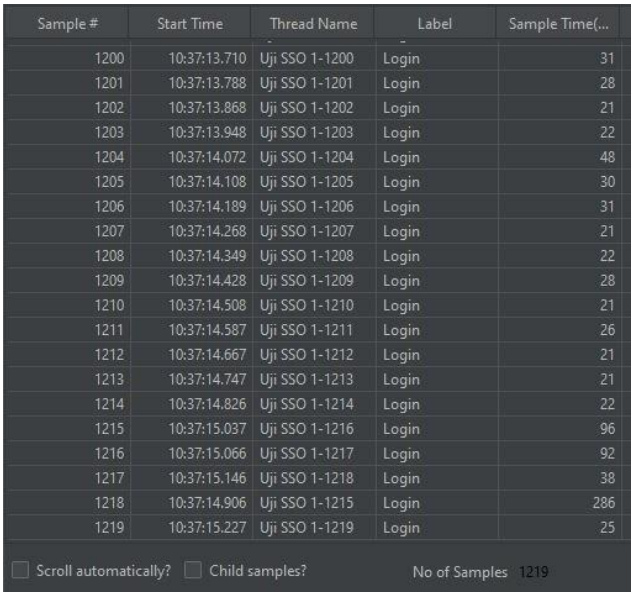
1) Load Testing

Load testing is carried out by arranging the test to imitate the maximum user limit that each protocol can manage. This test is run to guarantee that the system is constantly on target and focuses on the user login portion. JMeter is used for testing, and the Thread Group must be configured first.

Each SSO technique is tested by adding 200 users who make requests at the same time. The results obtained after progressive experimentation with each strategy are given in Figure 7.

Sample #	Start Time	Thread Name	Label	Sample Time(...
1211	10:18:48.523	Uji SSO 1-1211	Login	20
1212	10:18:48.603	Uji SSO 1-1212	Login	21
1213	10:18:48.683	Uji SSO 1-1213	Login	21
1214	10:18:48.762	Uji SSO 1-1214	Login	22
1215	10:18:48.841	Uji SSO 1-1215	Login	22
1216	10:18:48.921	Uji SSO 1-1216	Login	22
1217	10:18:48.999	Uji SSO 1-1217	Login	22
1218	10:18:49.079	Uji SSO 1-1218	Login	21
1219	10:18:49.159	Uji SSO 1-1219	Login	24
1220	10:18:49.341	Uji SSO 1-1220	Login	122
1221	10:18:49.339	Uji SSO 1-1221	Login	130
1222	10:18:49.397	Uji SSO 1-1222	Login	74
1223	10:18:49.476	Uji SSO 1-1223	Login	25
1224	10:18:49.555	Uji SSO 1-1224	Login	21
1225	10:18:49.655	Uji SSO 1-1225	Login	23
1226	10:18:49.716	Uji SSO 1-1226	Login	152
1227	10:18:49.874	Uji SSO 1-1228	Login	77
1228	10:18:49.799	Uji SSO 1-1227	Login	195
1229	10:18:49.955	Uji SSO 1-1229	Login	72
1230	10:18:50.035	Uji SSO 1-1230	Login	41

Scroll automatically? Child samples? No of Samples 1230



Sample #	Start Time	Thread Name	Label	Sample Time(...
1200	10:37:13.710	Uji SSO 1-1200	Login	31
1201	10:37:13.788	Uji SSO 1-1201	Login	28
1202	10:37:13.868	Uji SSO 1-1202	Login	21
1203	10:37:13.948	Uji SSO 1-1203	Login	22
1204	10:37:14.072	Uji SSO 1-1204	Login	48
1205	10:37:14.108	Uji SSO 1-1205	Login	30
1206	10:37:14.189	Uji SSO 1-1206	Login	31
1207	10:37:14.268	Uji SSO 1-1207	Login	21
1208	10:37:14.349	Uji SSO 1-1208	Login	22
1209	10:37:14.428	Uji SSO 1-1209	Login	28
1210	10:37:14.508	Uji SSO 1-1210	Login	21
1211	10:37:14.587	Uji SSO 1-1211	Login	26
1212	10:37:14.667	Uji SSO 1-1212	Login	21
1213	10:37:14.747	Uji SSO 1-1213	Login	21
1214	10:37:14.826	Uji SSO 1-1214	Login	22
1215	10:37:15.037	Uji SSO 1-1216	Login	96
1216	10:37:15.066	Uji SSO 1-1217	Login	92
1217	10:37:15.146	Uji SSO 1-1218	Login	38
1218	10:37:14.906	Uji SSO 1-1215	Login	286
1219	10:37:15.227	Uji SSO 1-1219	Login	25

Scroll automatically? Child samples? No of Samples 1219

Figure 7. Load Testing Results Using the Protocol OpenID (top) and OAuth (bottom)

According to the results of the load testing that was conducted in the experiment for 1400 users, not all requests can be completed. The OpenID protocol can support a maximum of 1230 concurrent user requests, but the OAuth protocol can only handle 1219 users. As a result, stress testing will be done using requests from 1200 people.

2) Stress Testing

Stress testing is a technique for determining a system's upper limit of performance by subjecting it to excessive stresses. The average reaction time, standard deviation, and throughput are all checked to determine if they are the same as they were before the load rise. JMeter is also used for stress testing by building test cases that increase the number of virtual users who use the system at the same time. Figure 8 shows a JMeter configuration for creating load spikes with the Thread Group component.

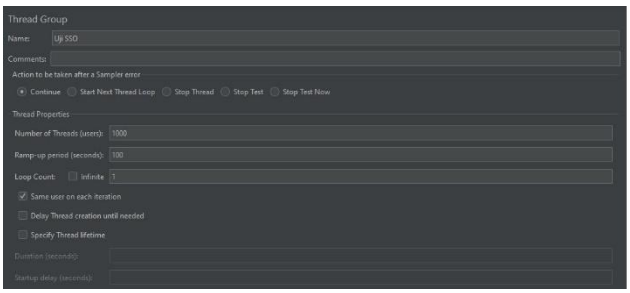


Figure 8. Configure Thread Group on JMeter

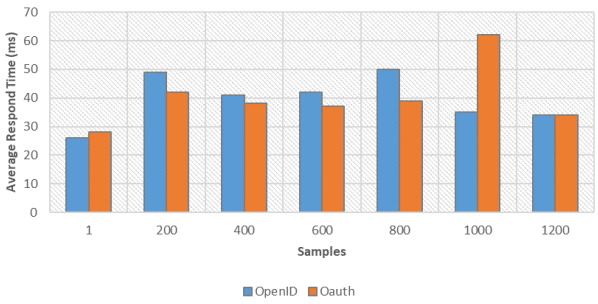
Each SSO protocol, including OpenID and OAuth, is tested by increasing the number of users. The findings of the stress tests are listed in TABLE IV.

TABLE IV. RESULT OF STRESS TESTING

Samples	Average Respond Time (ms)		Standard Deviation (ms)		Throughput (ms)	
	OpenID	OAuth	OpenID	OAuth	OpenID	OAuth
1	26	28	0	0	38.4615	35.7142
200	49	42	49.62	31.59	2.0070	2.0095
400	41	38	50.21	21.97	4.0085	4.0082
600	42	37	49.43	14.5	5.9738	5.9964
800	50	39	91.47	33.93	8.0084	7.9916
1000	35	62	28.69	126.98	10.0066	10.0115
1200	34	34	23.17	32.13	12.0530	12.0444

TABLE IV shows that while just dealing with one user, OpenID is faster than OAuth in terms of average response time. The OAuth protocol, on the other hand, has a faster response time for testing with 200-800 users, which is roughly 38-42 milliseconds. When compared to OpenID, the response time delivered reached 50 milliseconds in tests involving 800 users.

Various test results are displayed. When dealing with 1000 users, the OAuth protocol has a response time that is nearly twice as long as the OpenID protocol. See Graphic 1 for a more detailed comparison of the reaction times provided by each methodology.

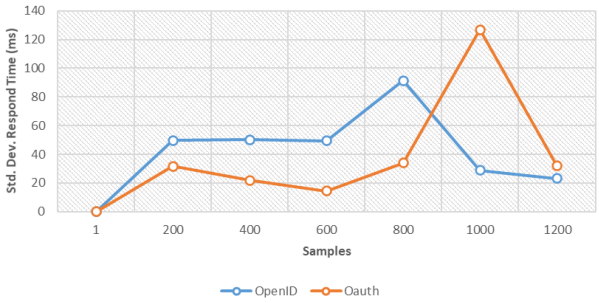


Graphic 1. Average Respond Time Between OpenID and OAuth

TABLE IV demonstrates that the standard deviation of the OpenID protocol is larger than that of the OAuth protocol for 200-800 users. The standard deviation value given increases as the number of users involved increases. When the OpenID protocol contains 800 users, the greatest standard deviation figure is 91.47.

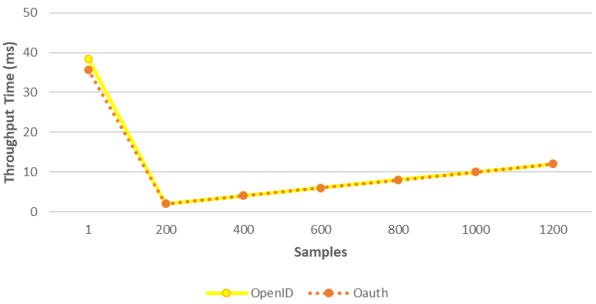
When 1000 users are involved, the findings are different. The OpenID protocol's standard deviation reduced to 28.69. This is inversely proportionate to the OAuth protocol's results, which continue to show a growing standard deviation. Even with 1000 users enrolled into the system, the standard deviation is nearly four times larger than with 800 users.

The standard deviation of the OAuth protocol reduced dramatically when tested with 1200 users. Graphic 2 illustrates the standard deviation statistics for each user involved in the OpenID and OAuth protocols for clarity (1-1200 users).



Graphic 2. Standard Deviation between OpenID and OAuth

Requests per unit of time are used to determine throughput. From the beginning of the first sample to the end of the last sample, time is calculated. TABLE IV shows that the throughput values of the two procedures are not significantly different. Except in single-user tests, the values are nearly identical. When compared to the OAuth protocol, which has a throughput value of only 35.7142, the OpenID protocol has a higher throughput value (38.4615). The accompanying Graphic 3 provides a more detailed discussion of the findings.



Graphic 3. Throughput Between OpenID and OAuth

IV. DISCUSSION

Requests per unit of time are used to calculate throughput. From the start of the first sample to the end of the last sample, time is calculated. The throughput values of the two procedures do not differ much in TABLE IV. Except in tests with only one user, the values are nearly identical between the two. When compared to the OAuth protocol, which only has a throughput value of 35.7142, the OpenID protocol has a higher throughput number (38.4615). The accompanying Graphic 3 provides a more detailed summary of the results.

The average response time of the system prototype under test did not demonstrate a significant difference between the OpenID protocol and OAuth, which is between 26 and 28 milliseconds, based on the test results for one user. When compared to authentication using the Lightweight Directory Access Protocol (LDAP) protocol [1], both approaches have significantly faster average response times.

The load testing findings, based on Figure 7 and the server specifications in TABLE II, reveal that the OpenID protocol can accommodate more users than the OAuth protocol. Even though the difference isn't large, this proves that the OpenID protocol can support more users. Both protocols, on the other hand, can reliably handle all requests from 1-100 users.

The average time response, standard deviation, and throughput are all tested during the stress testing. The OAuth protocol appears to respond to user requests more quickly, as seen in Graphic 1. In comparison to the OpenID protocol, the OAuth protocol delivers a longer response time when 1000 users access the login page at the same time.

SSO using the OpenID protocol has a higher standard deviation than SSO using the OAuth protocol. Despite the fact that the OAuth protocol has a rather high standard deviation value while handling requests from 1000 users, the value lowers to the same level when testing with 1200 users. The OAuth protocol has a more constant response time than the OpenID protocol. This is evidenced by the OAuth protocol's low standard deviation number when compared to the OpenID protocol.

The standard deviation number for a label should, in general, be less than or equal to half of the mean time [12]. According to the data, the OpenID protocol's standard deviation is usually always greater than the average response time. The OAuth protocol, on the other hand, has a value less than the average response time.

The number of requests processed per unit time by the server is referred to as throughput. Starting with tests including 200 to 1200 users, the throughput of the two protocols, OpenID and OAuth, reveals nearly no difference.

According to performance testing, SSO implemented in academic information system services and UTY e-learning using the standards in TABLE II can only manage requests from 1230 users at a time using the OpenID protocol, and only 1219 users using the OAuth protocol. As a result, higher specifications are required to suit the expectations of more users. In comparison to the OpenID protocol, the OAuth protocol has a more consistent average response time while dealing with user requests. Both of them, on the other hand, can respond to user queries in a timely and error-free manner.

ACKNOWLEDGMENT

Special thanks to University of Technology Yogyakarta for supporting this research.

REFERENCES

[1] I. P. A. E. D. Udayana and L. Jasa, "Implementasi dan Analisis Single Sign On Pada Sistem Informasi Universitas Udayana," in *Seminar Nasional Teknologi Informasi dan Multimedia*, Yogyakarta, 2016.

[2] Q. Aini, U. Rahardja and R. S. Naufal, "Penerapan Single Sign On dengan Google pada Website berbasis YII Framework," *SISFOTENIKA*, vol. 8, no. 1, pp. 57-68, 2018.

[3] T. Suryana and A. Amarullah, "Single Sign On (SSO) Menggunakan Standar SAML Pada Sistem Informasi Unikom," *Jurnal Majalah Ilmiah Unikom,* vol. 15, no. 1, pp. 87-94, 2017.

[4] G. Guntoro and M. Fikri, "Perancangan Aplikasi Single Sign-On Menggunakan Autentikasi Gambar," *Digital Zone: Jurnal Teknologi Informasi dan Komunikasi,* vol. 9, no. 1, pp. 12-21, 2018.

[5] N. Dagli, M. Deorukhar, S. Sawant, K. Upadhyaya and N. Shaikh, "Implementation of Single Sign On (SSO) for College websites," *International Research Journal of Engineering and Technology (IRJET),* vol. 7, no. 5, pp. 1285-1289, 2020.

[6] A. G. Andriani, "Implementasi Sso(Single Sign On) Untuk Sentralisasi Login Web Aplikasi Internal Studi Kasus Pt Duta Visual Nusantara Tivi Tujuh," Universitas Komputer Indonesia, Bandung, 2019.

[7] A. Suhardi, E. Fatkhiyah and M. Sholeh, "Perancangan dan Implementasi SSO (Single Sign On) Menggunakan Protokol Oauth 2.0," *Jurnal JARKOM,* vol. 5, no. 1, pp. 65-75, 2017.

[8] T. Bazaz and A. Khalique, "A Review on Single Sign on Enabling Technologies and Protocols," *International Journal of Computer Applications,* vol. 151, no. 11, pp. 18-25, 2016.

[9] V. Radha and D. H. Reddy, "A Survey on Single Sign-On Techniques," *Procedia Technology,* vol. 4, pp. 134-139, 2012.

[10] P. B. Sahare, "Design And Implementation Of Enhanced Single Sign On System For Education Systems," *International Research Journal of Engineering and Technology(IRJET),* vol. 4, no. 7, pp. 717-721, 2017.

[11] A. R. Hevner, S. T. March, J. Park and S. Ram, "Design Science in Information Systems Research," *MIS Q.,* vol. 28, no. 1, pp. 75-105, 2004.

[12] R. Bhatt, "Understand and Analyze Summary Report in Jmeter," Testing Journals, 31 1 2017. [Online]. Available: http://www.testingjournals.com/understand-summary-report-jmeter/. [Accessed 19 3 2022].