Performance and Security Comparison of Json Web Tokens (JWT) and Platform Agnostic Security Tokens (PASETO) on RESTful APIs

1st Adiva Fiqri Nugraha
Politeknik Siber dan Sandi Negara
Bogor, Indonesia
adiva.fiqri@student.poltekssn.ac.id

2nd Herman Kabetta
Politeknik Siber dan Sandi Negara
Bogor, Indonesia
herman.kabetta@poltekssn.ac.id

3rd I Komang Setia Buana
Politeknik Siber dan Sandi Negara
Bogor, Indonesia
komang.setia@poltekssn.ac.id

4th R Budiarto Hadiprakoso
Politeknik Siber dan Sandi Negara
Bogor, Indonesia
raden.budiarto@poltekssn.ac.id

2023 IEEE International Conference on Cryptography, Informatics, and Cybersecurity (ICoCICs) | 979-8-3503-3943-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICoCICs58778.2023.10277377

Abstract— **This research aims to compare the performance and security between Json Web Token (JWT) and Platform Agnostic Security Tokens (PASETO) on RESTful API. Performance is measured through the parameters of token generation time, token size, and token transfer time, while security is measured through the top three API OWASP 2019 vulnerabilities and testing CSRF attacks on tokens. The results from 50 samples show that the average token generation time of JWT is 0.5068 ms and PASETO is 2.4044 ms. The average transfer time of JWT tokens is 95.4604 ms and PASETO 190.4344 ms. The T-test results confirm the significant difference between token generation time and token transfer time in JWT and PASETO. In addition, the number of PASETO tokens generated is also greater at 320 tokens compared to JWT at 186 tokens. In security testing, JWT tokens are proven safe from OWASP 2019 API attacks, such as Broken Object Level Authorization and Excessive Data Exposure, but are not safe from Broken User Authentication attacks. PASETO tokens were shown to be safe from all three attacks. However, neither JWT nor PASETO tokens are safe from CSRF Cookie attacks. The result can be concluded that PASETO has a longer token generation time and transfer time compared to JWT. However, PASETO shows a higher level of security compared to JWT, as it can protect against the top three attacks of the OWASP 2019 API.**

Keywords—JWT, PASETO, RESTful, API, Security API, Token Authentication

I. Introduction

REST is a web-based service architecture standard for data communication, such as APIs, and uses the HTTP protocol standard [1],[2]. REST is still weak in terms of security when you want to limit access rights to the API you want to use [3]. REST security aims to secure data communication paths, protect data confidentiality and integrity, and use authentication to limit user access to APIs [4]. In 2019, more than 75% of credential stuffing attacks targeted API services [5]. Based on data from the API security trends report 2022 written by Dan Kennedy in his black and white paper, it states that the increase in attacks over the past 12 months has resulted in security vulnerabilities in the API, namely 55% data breaches, 46% accidental exposure of data, 35% Denial of service (DOS), and 31% Account takeover (ATO) [6]. The increasing attacks on APIs also need to be considered to improve security when implementing RESTful API services in an application.

In 2019, OWASP released a project called OWASP API Top Ten 2019, which contains ten vulnerabilities in the API, and the three most common vulnerabilities are Broken Object Level Authorization, Broken User Authentication and Excessive Data Exposure [7]. These vulnerabilities are caused by errors in the authentication mechanism, authorization, and data used to identify user credentials. Therefore, steps are needed to reduce these vulnerabilities by implementing access rights restrictions to improve the authentication mechanism [5].

Common authentication techniques used in applications include session-based authentication and token-based authentication [8]. Token-based authentication shows better performance than session-based authentication, because token-based authentication does not store the session created each time a user logs in but only the time between login and logout so as to balance the server workload [8] [9]. One of the token-based authentication methods is JSON Web Tokens (JWT) [10].

JWT is a JSON string token that is useful for authentication systems and information exchange [11]. JWT allows users to choose which algorithm to use in its implementation. The signing algorithms provided by JWT are RSA, ECDSA, and HMAC. Besides JWT, there is a token-based authentication called Platform-Agnostic Security Tokens (PASETO) that can perform encryption and signature [12]. The algorithms and libraries used are only known by the PASETO server. This aims to prevent attackers from knowing the algorithm used so as to reduce the occurrence of token forgery. PASETO only provides its own version of the protocol [12]. Compared to JWT which allows users to choose any algorithm used, PASETO directs users to use the protocol version which includes PASETO's own algorithms and libraries.

There are several studies on token authentication, one of which is research conducted by Rahmatulloh et al in 2018 and 2019 comparing the performance of JWT tokens with token generation time parameters, token size, and token transfer time on RSA, ECDSA, and HMAC algorithms [13], [14]. Other research conducted by Gunawan and Rahmatulloh implemented JWT on the RESTful API [15]. Other research was also conducted by Dalimunthe et al in 2022 by applying JWT tokens to cookies. However, of the four studies, no security testing has been carried out. Other research conducted on JWT with the HMAC-SHA256 algorithm performs security tests with CSRF techniques on cookie storage, but there is no comparison of token performance and security, this research is only limited to JWT with the HMAC-SHA256 algorithm [16]. On the other hand, in PASETO, Sitorus et al. tested the security of PASETO versions v2.l and v2.p with

Base65 encode/decode parameters, none algorithm exploitation, and symmetric-asymmetric algorithm exploitation [17], but did not conduct performance testing as done by Rahmatulloh et al. and also CSRF security testing by Darmawan et al. Therefore, it can be concluded that JWT and PASETO tokens have not been tested for security which includes the none algorithm exploitation test, and the OWASP API Security Top 10 test and there is no performance testing and security testing in the form of CSRF penetration tests on JWT and PASETO in versions and algorithms that have the same characteristics.

Based on this background, the focus of this research is to conduct comparative testing of the performance and security of JWT and PASETO authentication tokens. The token performance testing parameters are token generation time, token size, and token transfer time. The security testing parameters carried out are Broken Object Level Authorization, Broken User authentication, and Excessive Data Exposure based on the OWASP API Security 2019 standard, and penetration testing with the CSRF method. This research aims to ensure that developers and future researchers can use and know the differences between the two to get Restful API authentication tokens that are more efficient and still safe.

II. Related Work

There have been several studies on token authentication that have been conducted previously. One of them is a study conducted by Rahmatulloh et al in 2018 and 2019. This research aims to compare the performance of JWT tokens with parameters such as token generation time, token size, and token transfer time in RSA, ECDSA, and HMAC algorithms [13], [14].

Another relevant study was conducted by Gunawan and Rahmatulloh, who implemented JWT in a RESTful API [15]. In addition, Dalimunthe et al also conducted research in 2022 by implementing JWT tokens in cookies. However, none of these four studies involved security testing.

There is also another study that focuses on JWT with the HMAC-SHA256 algorithm. This research conducted security testing using CSRF techniques on cookie storage. However, this study did not include a comparison of token performance and security. It was limited to JWT with the HMAC-SHA256 algorithm [16].

On the other hand, Sitorus et al conducted security testing on PASETO versions v2.l and v2.p. This study included parameters such as Base65 encoding/decoding, exploitation of the "none" algorithm, and exploitation of symmetric-asymmetric algorithms [17]. However, this research did not involve performance testing like the study conducted by Rahmatulloh et al, nor did it involve security testing for CSRF as done by Darmawan et al.

From the summary of these studies, it can be concluded that comprehensive security testing has not been conducted on JWT and PASETO tokens, including testing for exploitation of the "none" algorithm and compliance with OWASP API Top Ten 2019. Furthermore, there has been no performance testing or security testing in the form of CSRF penetration testing on JWT and PASETO in versions and algorithms with similar characteristics.

III. Methodology

In this study, the JWT algorithm used is HMAC-SHA384, while for PASETO, the protocol version v3.l with the AES-256-CTR + HMAC-SHA384 algorithm is used. After building the RESTful API with JWT and PASETO token authentication, performance and security testing will be conducted. Performance testing includes token generation time, token size, and token response speed. The security testing parameters conducted are the top three of OWASP API Security Top Ten 2019, which are Broken Object Level Authorization, Broken User Authentication, and Excessive Data Exposure, as well as conducting penetration testing using the CSRF method.

The design of this study uses the Design Research Methodology (DRM) research method with the Software Development Life Cycle (SDLC) Waterfall software development method. DRM is used because it supports an appropriate approach to obtain a more effective and efficient research design. DRM consists of four stages, including Research Clarification (RC), Descriptive Study I (DS I), Perspective Study (PS), and Descriptive Study II (DS II) [18]. The research design can be seen in Figure 1.

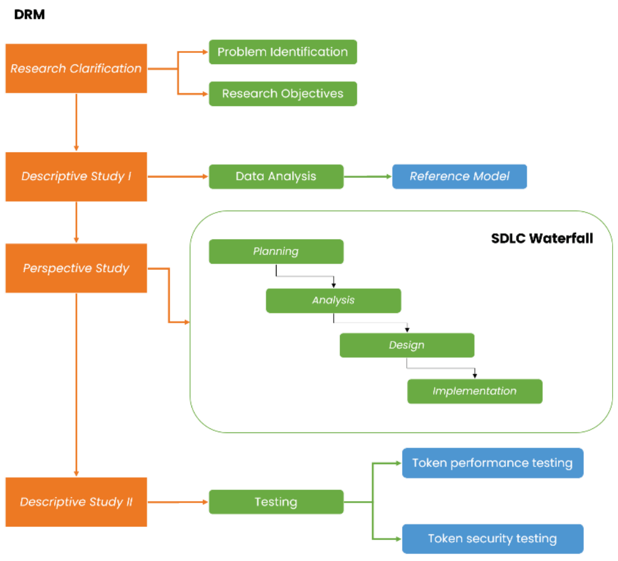

Figure 1. Research Design

1. *Research Clarification* (RC)

The Research Clarification (RC) stage is a phase where researchers gather evidence from existing theories as indications that support the hypothesis in order to formulate realistic and beneficial research objectives. In this stage, two things are produced: problem identification and research objectives.

2. *Descriptive Study 1*

The Descriptive Study I (DS-I) stage is aimed at gaining a better understanding of the existing situation by conducting identification, creating detailed descriptions through data analysis, and providing a more detailed explanation of the factors that influence the success of the research. In this stage, success factors, measurable success factors, and key factors relevant to the research are determined to create a reference model that will be used in this study.

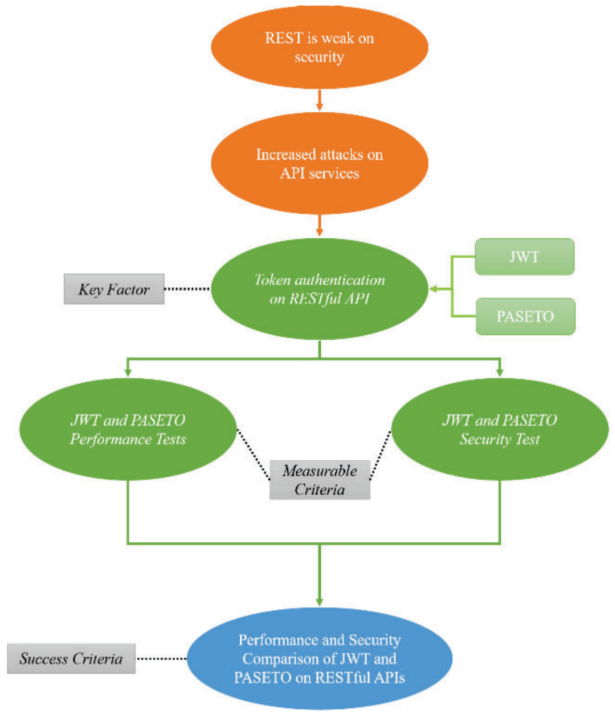
REST is weak on security
Increased attacks on API services
Key Factor
Token authentication on RESTful API
JWT
PASETO
JWT and PASETO Performance Tests
JWT and PASETO Security Test
Measurable Criteria
Success Criteria
Performance and Security Comparison of JWT and PASETO on RESTful APIs

Figure 2. *Reference Model*

3. *Prespective Study* (PS)

The Prescriptive Study (PS) is the stage of problem-solving aimed at correcting and describing the expected objectives. In this stage, planning and development of the RESTful API with JWT and PASETO token authentication will be carried out. The process of building the RESTful API with token authentication is done using the SDLC Waterfall method, which consists of planning, analysis, design, and implementation.

1) Planning
Literature review serves as an initial guide for the subsequent stages. In this stage, the researcher will conduct a literature search on Json Web Token (JWT), Platform Agnostic Security Token (PASETO), and RESTful API from various sources such as books, papers, thesis, the internet, and final projects..

2) Analysis
The analysis stage is conducted by performing problem analysis, program analysis, and requirement analysis, which includes functional requirements, non-functional requirements, software requirements, and hardware requirements. The purpose of requirement analysis is to identify the necessary functions that should be present in the application based on literature review.

3) Design
The design stage involves designing the RESTful API with token authentication, depicted in the form of a business process and Data Flow Diagram..

4) Implementation
The implementation phase involves building the RESTful API with token authentication using the Node.js runtime. Two separate servers are created for JWT and PASETO, and they are run in a Docker Container environment with identical characteristics for both JWT and PASETO implementations.

The environment used can be seen in Table 1. Once the application is developed, it will proceed to the performance and security testing phase.

Table 1. *Environment System*

No	Item	Deskripsi	Versi
1.	Node.js	*JavaScript runtime environment.*	18.12.1 LTS
2.	Docker Container	Virtualisasi sistem operasi berbasis kontainer	20.10.9
3.	Burp Suite	*Network Testing Tools*	2020.9.1
4.	Postman	*API Testing Tools*	v9.4

4. *Descriptive Study II (DS-II)*

This stage is the final phase to conduct testing on token authentication in the created RESTful API. The testing phase is performed based on the performance and security of the token.

a. Token performance testing

The performance testing samples were taken based on the research [13][14], consisting of fifty trial runs for HMAC-SHA384 and V3.l. The performance testing parameters include token generation time, which is measured by capturing the start and end times of the token signing function, and token transfer time, which is measured using the Postman application, as described in the research [17].

To analyze the performance differences (token generation time and token transfer time) using the independent sample t-test method, the researcher utilized the SPSS application. This was done to test the performance differences between the JWT algorithm and the PASETO protocol version. The formulated hypothesis to be tested are as follows:

H0	:	There is no significant difference in performance between JWT and PASETO tokens.
H1	:	There is a significant difference in performance between JWT and PASETO tokens.

The basis for decision-making is:

- If the Sig. (2-tailed) value < significance level = 0.05, then there is a significant difference between the token generation time or token transfer time of JWT and PASETO tokens..
- If the Sig. (2-tailed) value > significance level = 0.05, then there is no significant difference between the token generation time or token transfer time of JWT and PASETO tokens.

b. Token security testing.

Security testing is divided into several projects based on the type of attack carried out. The security testing parameters conducted include Object Level Authorization, Broken User Authentication, and Excessive Data Exposure based on the standard scenarios outlined in the OWASP API Security Top Ten 2019. Additionally, penetration testing is performed using the CSRF method. Security testing is carried out on both tokens, JWT token and also on the PASETO token. The details of the testing are explained in Table 2.

Table 2. Token Testing Explanation

No.	Scenario/Project	Testing	Description
1.	Token performance testing	Token Generation Time Testing, Token Quantity Testing, and Token Transfer Time Testing	The data obtained from each token is subjected to independent sample t-test.
2.	Access to sales data	*Broken Object Level Authorization*	Accessing store sales data (which should not be accessible to them).
	PATCH X-User-Id		Modifying the http header X-User-Id.
3.	Header alg none	*Broken User Authentication*	Performing forged token and changing the alg in the header to none.
	Decode payload token		Decoding the payload from the access token.
4.	Sensitive data access through comments	*Excessive Data Exposure*	Accessing sensitive user data through comments.
5.	Penetration testing CSRF	CSRF	*CSRF penetration testing on JWT and PASETO tokens stored in Cookie Storage.*

Specifically for CSRF penetration testing, it differs from the OWASP API Security Top 10 2019 security testing. In this specific testing, the token is stored in cookies. The CSRF phase is explained in Figure 3.

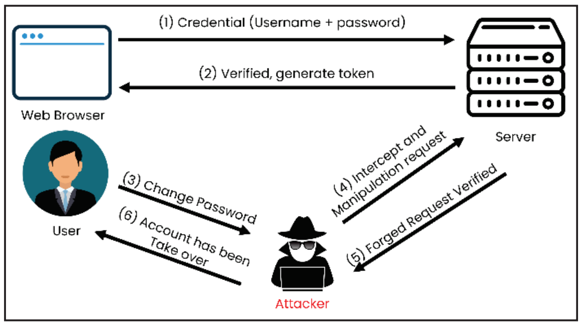
(1) Credential (Username + password)
(2) Verified, generate token
Web Browser
Server
User
(3) Change Password
(6) Account has been Take over
(4) Intercept and Manipulation request
(5) Forged Request Verified
Attacker

Figure 3. CSRF Penetration Testing Scheme

IV. Result And Analysis

This chapter discusses planning, analysis, design, implementation, and testing.

1. *Planning*

The planning phase is carried out by initiating five server-side website application development projects, each project encompassing two security authentication methods using JWT and PASETO tokens. These five projects are created based on the conducted testing, with one project dedicated to testing the performance of each token, specifically testing the token generation time, token size, and token transfer time. The remaining four projects are focused on security testing, specifically evaluating parameters such as Broken Object Level Authorization, Broken User Authentication, and Excessive Data Exposure based on the OWASP API Security 2019 standard, and conducting CSRF penetration testing.

2. *Analysis*

a. Program Analysis

In this study, the application is focused on testing its performance and security, hence the application is developed on the server-side specifically for token generation processes of JWT and PASETO. Both token authentications and all the testing procedures are conducted using the JavaScript programming language with the Node.js runtime. The following illustrates the system architecture for token authentication security in both JWT and PASETO.

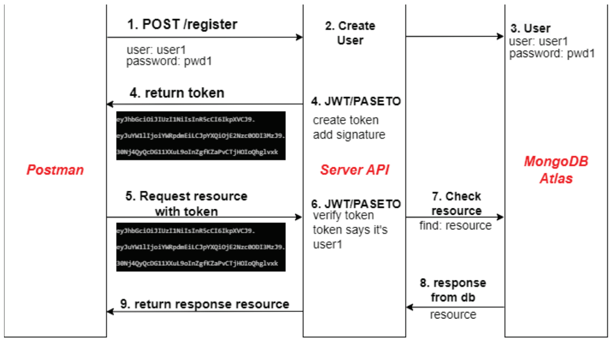
Postman
1. POST /register
user: user1
password: pwd1
2. Create User
3. User
user: user1
password: pwd1
4. return token
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
eyJuYW1lIjoiYWRpdmEiLCJpYXQiOjE2NzcwODI3MzJ9.
30Nj4QyQcDG11XXuL9oInZgfKZaPvCTjHOIoQhglvxk
4. JWT/PASETO
create token
add signature
Server API
MongoDB Atlas
5. Request resource with token
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
eyJuYW1lIjoiYWRpdmEiLCJpYXQiOjE2NzcwODI3MzJ9.
30Nj4QyQcDG11XXuL9oInZgfKZaPvCTjHOIoQhglvxk
6. JWT/PASETO
verify token
token says it's user1
7. Check resource
find: resource
8. response from db
resource
9. return response resource

Figure 4. Token Authentication System Architecture

b. Requirement Analysis

The requirements analysis in this study consists of functional and non-functional requirements, which are described as follows:

Functional Requirements, The following are the functional requirements:

- Program should verify token signatures: Ensure that the signatures of JWT and PASETO tokens are properly verified. Tokens should be considered invalid if the signature does not match or if there are any modifications to the token.
- Program should verify token expiration: Ensure that JWT and PASETO tokens are not used after the specified expiration time. Tokens should be considered invalid if the expiration time has passed.
- Program should verify access rights: Ensure that JWT and PASETO tokens provide the correct access rights to users. Ensure that users can only access the resources they are supposed to have.

Non-functional requirements, The following are the non-functional requirements:

- Security: This requirement is related to the ability of token authentication to protect user data from security threats such as injection attacks, broken authentication and session management, and broken object-level authorization. Security testing should be conducted to ensure that both token authentications can meet high-security standards.

- Performance: This requirement is related to the ability of token authentication to provide good performance in terms of token generation time, token count, and token transfer time between the server and clients. Performance testing should be conducted to ensure that both token authentications can generate efficient tokens.

3. *Design*

Design Phase explains the process of designing an application that implements JWT and PASETO token authentication mechanisms, illustrated in the form of a business process and Data Flow Diagram (DFD).

a. Business Process

This section describes the business process in the application that implements JWT and PASETO token authentication mechanisms, illustrated through a business process diagram.

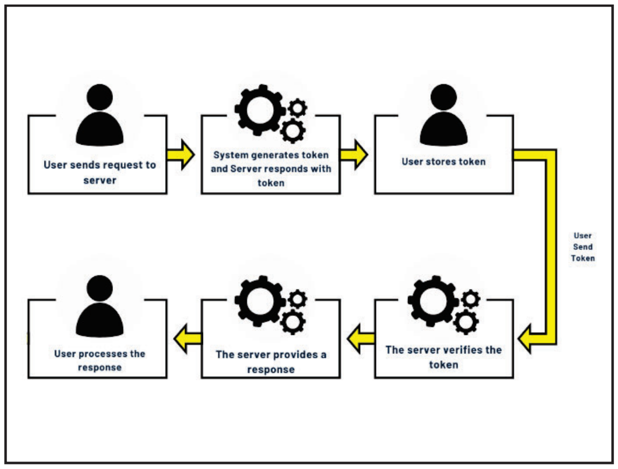
User sends request to server
System generates token and Server responds with token
User stores token
User Send Token
The server verifies the token
The server provides a response
User processes the response

Figure 5. Business Process

b. Data Flow Diagram

There are two DFDs created: DFD level 0 and level 1. The DFD level 0 provides an overview of the entire system. It includes one main process, the API that uses JWT and PASETO token authentication. The main actor involved in this system is the user. There are five inputs from the actor to the system, and five outputs generated by the system. DFD level 0 can be seen in Figure 6.

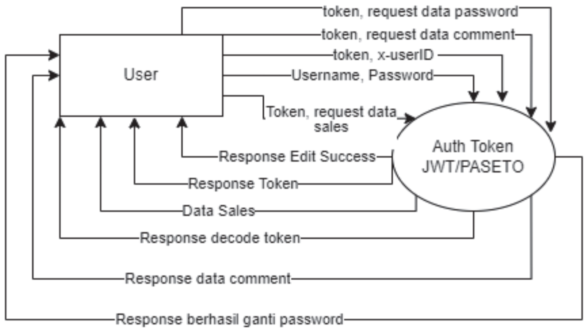

Figure 6. DFD level 0

DFD level 1 is more detailed than DFD level 0 and shows how the main system is decomposed into smaller parts. At this level, the main process in level 0 is divided into five smaller and more detailed processes: API Login, API Sales Data Access, API Account Editing, API Get Comment, and API Change Password. Additionally, level 1 also shows the data input and output of each process and the data flow between processes. DFD level 1 can be seen in Figure 7.

4. *Implementation and Testing*

In this phase, the implementation results of the design process created in the previous phase will be explained.

a. Database

In the application development process, a database is required as a data exchange medium to be used for token testing. The database used is a NoSQL database, MongoDB. Testing is conducted with five projects using five different databases. The databases for each application are illustrated in Table 3.

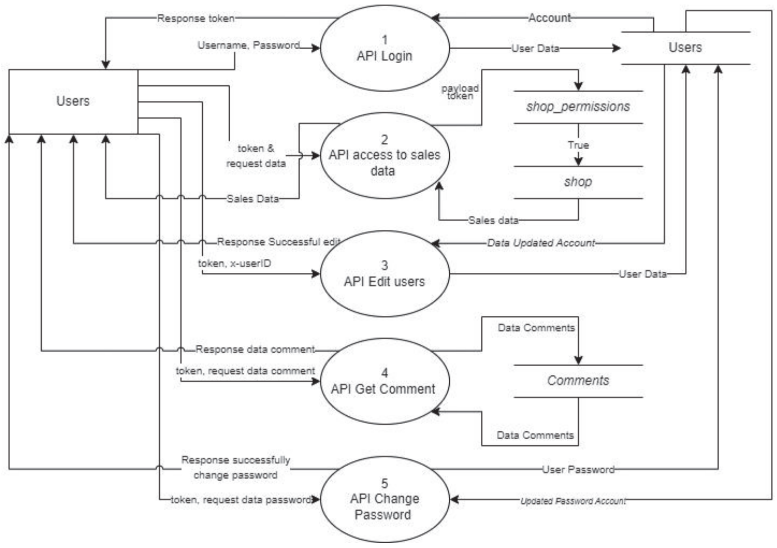

Figure 7. DFD level 1

Table 3. Database

Projeck	Database	*Collection*	*Field*
Token performance Testing	Pengujiantoken	Posts	username,title
		User	username, password, email, imageUrl, userId
		Refreshtokens	_id, token
Broken Object Level Authorization	skenario_satu_a	Users	username, password, email, imageUrl, userId
		Shops	Id, name, owner_id
		Shop_permissions	id, user_id, shop_id
		refreshTokens	id, token
Broken User Authentication	skenario_dua	Users	userId, username, email, password, role
		refreshTokens	Id, token
Excessive Data Exposure	skenario_tiga	Users	id, name, email, role, username, password
		refreshTokens	Id, token
		comments	Id, articleId, content, authorId
CSRF	Skenario_empat	Users	userId, username, email, password, role
		refreshTokens	Id, token

b. Token Performance Testing

This section focuses on testing the performance of tokens with parameters such as token generation time, token size, and token transfer time. All testing is conducted using 50 samples for both JWT and PASETO tokens.

a. *Token Generation Time*

The token generation time parameter is measured using the performance.now() function in the generateToken function to obtain accurate results in milliseconds (ms) [19]. Figure 8 shows that the average token generation time for PASETO is longer compared to the token generation time for JWT.

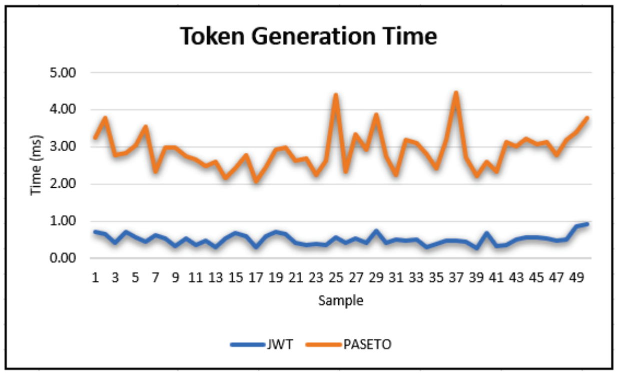

Figure 8. Token Generation Time

The token generation time parameter was also subjected to an independent sample t-test hypothesis to compare the token generation time between JWT and PASETO. The output of the independent sample t-test indicates that there is a significant difference in the token generation time between the two. The Sig (2-tailed) value is 0.000, which indicates that H0 is rejected and H1 is accepted. Therefore, it can be concluded that the hypothesis test results show a significant difference in the token generation time between JWT and PASETO. The output of the independent sample t-test for token generation time can be seen in Table 4.

Table 4. Output *Independent* sample *t-test* Token Generation Time

		Levene's Test for Equality of Variances		T-test for Equality of Means		
		F	Sig.	t	df	Sig. *(2-tailed)*
Token generation time	*Equal variances assumed*	27.52	.000	-26.6	98	.000
	Equal variances not assumed			-26.6	57.741	.000

b. Token Size

In this study, an analysis of the token count was conducted for JWT and PASETO, using 50 samples. From the analysis, it was found that the token count for JWT was 186, while for PASETO it was 320.

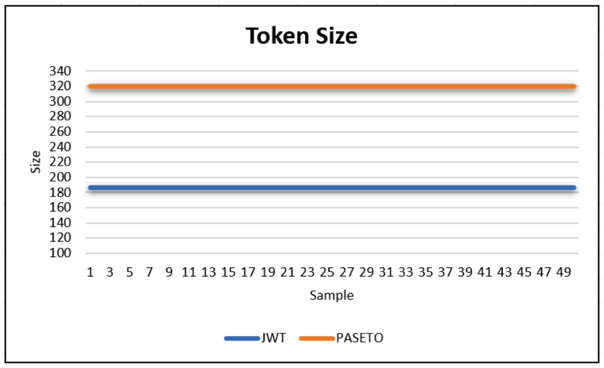

Figure 9. Token Size

c. Token Transfer Time

The data for the token transfer time parameter used in this study was obtained from the Postman application when successfully logging into the API. Figure 10 displays the 50 samples of token transfer time obtained for JWT and PASETO. The results from the graph show that the average token transfer time for PASETO is longer compared to JWT.

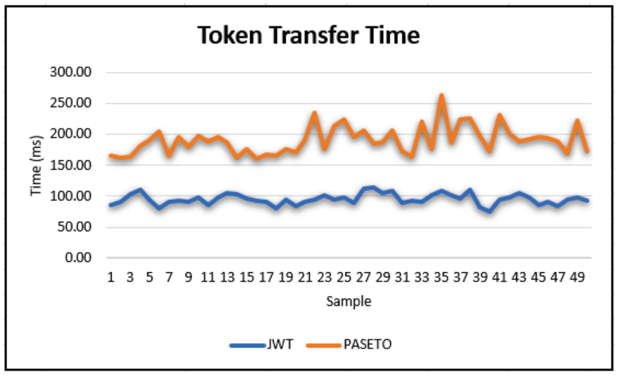

Figure 10. Token Transfer Time

The calculated value of Sig (2-tailed) is 0.000, indicating that H0 is rejected and H1 is accepted. Therefore, it can be concluded that the hypothesis test results show a significant difference in the token transfer time between JWT and PASETO. The output of the independent sample t-test for token transfer time can be seen in Table 5.

Table 5. Output *Independent sample t-test* Token Transfer Time

		Levene's Test for Equality of Variances		T-test for Equality of Means		
		F	Sig.	t	df	Sig. *(2-tailed)*
Token transfer time	*Equal variances assumed*	23.182	.000	-27.62	98	.000
	Equal variances not assumed			-27.62	63.805	.000

c. Token Security Testing

This section focuses on testing the security of the token with several test scenarios that have been described in the previous section. At this testing stage is done by testing using the black box testing method. This test aims to ensure that the security tested is in accordance with the three highest OWASP API Security 2019 and Penetration Testing CSRF on Cookie Storage. All scenarios in this test are written in the test case as shown in Table 6.

Table 6. Results of Token Security Testing Test Cases

No.	Test Scenario Description	Test Case ID	Test Cases	Hasil
1	Broken Object Level Authorization - API Data Penjualan	FT_001	Valid API access to sales data - JWT	Secure
		FT_002	Invalid API access to sales data - JWT	Secure
		FT_003	Valid API access to sales data - PASETO	Secure
		FT_004	Invalid API access to sales data - PASETO	Secure
	Broken Object Level Authorization - PATCH X-User-Id	FT_005	X-User-Id Header Tampering Attack - JWT	Secure
		FT_006	X-User-Id Header Tampering Attack – PASETO	Secure
2	Broken User Authentication - Header alg none	FT_007	Exploiting token header algorithm - JWT	Not Secure
		FT_008	Exploiting token header algorithm - PASETO	Secure
	Broken User Authentication - Decode payload Token	FT_009	Decode payload token - JWT	Not Secure
		FT_010	Decode payload token - PASETO	Secure
3	Excessive Data Exposure – Access Sensitive Data through comments	FT_011	Sensitive data access through comments - JWT	Secure
		FT_012	Sensitive data access through comments - PASETO	Secure
4	CSRF Token stored on Cookie	FT_013	Penetration testing CSRF - JWT	Not Secure
		FT_014	Penetration testing CSRF - PASETO	Not Secure

V. Conclusion

Based on the results of the research conducted, several conclusions were obtained to answer the problem formulation in this study:

a. Of the 50 samples of JWT and PASETO performance testing, there is a significant difference in the token generation time parameter with the average time of token generation. significant difference in the token generation time parameter with an average time of 0.5068 ms for JWT and 2.5068 ms for PASETO. In addition, there is a significant difference in the token transfer time parameter with the average time for JWT being 95.4604 ms and for PASETO 190.4344 ms. The token size generated for JWT is 186, while for PASETO it is 320. This shows that PASETO generates more tokens than JWT. Overall, based on the performance testing results, it can be concluded that JWT has better performance than PASETO in terms of token generation time, token size, and token transfer time.

b. Based on the results of security testing on 14 test cases conducted on JWT and PASETO tokens, the test results show that JWT tokens are vulnerable to Broken User Authentication which focuses on exploiting header algorithms and decoding payloads. In addition, PASETO is safe from the top three of OWASP API Security 2019. Based on the security testing results, it can be concluded that PASETO has security advantages over JWT. PASETO is proven to be secure against the tested vulnerabilities, while JWT shows vulnerabilities in some aspects, such as header algorithm exploitation and token payload decode.

REFERENCE

[1] A. Soni and V. Ranga, “API features individualizing of web services: REST and SOAP,” *Int. J. Innov. Technol. Explor. Eng.*, vol. 8, no. 9 Special Issue, pp. 664–671, 2019, doi: 10.35940/ijitee.I1107.0789S19.

[2] A. Tarkowska, D. Carvalho-Silva, C. E. Cook, E. Turner, R. D. Finn, and A. D. Yates, “Eleven quick tips to build a usable REST API for life sciences,” *PLoS Comput. Biol.*, vol. 14, no. 12, pp. 1–8, 2018, doi: 10.1371/journal.pcbi.1006542.

[3] de Rosal Ignatius Moses Setiadi, A. F. Najib, E. H. Rachmawanto, C. A. Sari, M. K. Sarker, and N. Rijati, “A comparative study MD5 and SHA1 algorithms to encrypt REST API authentication on mobile-based application,” *2019 Int. Conf. Inf. Commun. Technol. ICOIACT 2019*, pp. 206–211, 2019, doi: 10.1109/ICOIACT46704.2019.8938570.

[4] I. G. Anugrah and M. A. R. I. Fakhruddin, “Development Authentication and Authorization Systems of Multi Information Systems Based REst API and Auth Token,” *Innov. Res. J.*, vol. 1, no. 2, p. 127, 2020, doi: 10.30587/innovation.v1i2.1927.

[5] E. Cabezas, “Leveraging the OWASP API Security top 10 to build secure web services,” 2020, [Online]. Available: https://www.sans.org/white-papers/39935/

[6] D. Kennedy, “The 2022 API Security Trends Report,” no. April, pp. 5–9, 2022.

[7] The OWASP Foundation, “OWASP API Security Top 10 2019”, [Online]. Available: https://owasp.org

[8] Y. Balaj, “Token-Based vs Session-Based Authentication : A survey,” no. September, pp. 1–6, 2017.

[9] S. Lee, J. Y. Jo, and Y. Kim, “Authentication system for stateless RESTful Web service,” *J. Comput. Methods Sci. Eng.*, vol. 17, no. S1, pp. S21–S34, 2017, doi: 10.3233/JCM-160677.

[10] B. Satria, A. Kusyanti, and W. Yahya, “Implementasi Algoritme Blake2s pada JSON Web Token (JWT) sebagai Algoritme Hashing untuk Mekanisme Autentikasi Layanan REST-API,” *J. Pengemb. Teknol. Inf. dan Ilmu Komput. Univ. Brawijaya*, vol. 2, no. 12, pp. 6269–6276, 2018, [Online]. Available: https://www.researchgate.net/publication/338188718_Implementasi_Authentication_dengan_JSON_Web_Token_JWT_pada_Laravel_Single_Page_Application

[11] jwt.io, “Jwt.io,” 2022. https://jwt.io/ (accessed Oct. 10, 2022).

[12] Scott Arciszewski, “No Way, JOSE! Javascript Object Signing and Encryption is a Bad Standard That Everyone Should Avoid,” 2017. https://paragonie.com/blog/2017/03/jwt-json-web-tokens-is-bad-standard-that-everyone-should-avoid (accessed Oct. 18, 2022).

[13] A. Rahmatulloh, R. Gunawan, and F. M. S. Nursuwars, “Performance comparison of signed algorithms on JSON Web Token,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 550, no. 1, 2019, doi: 10.1088/1757-899X/550/1/012023.

[14] A. Rahmatulloh, H. Sulastri, and R. Nugroho, “Keamanan RESTful Web Service Menggunakan JSON Web Token (JWT) HMAC SHA-512,” *J. Nas. Tek. Elektro dan Teknol. Inf.*, vol. 7, no. 2, 2018, doi: 10.22146/jnteti.v7i2.417.

[15] R. Gunawan and A. Rahmatulloh, “JSON Web Token (JWT) untuk Authentication pada Interoperabilitas Arsitektur berbasis RESTful Web Service,” vol. 5, no. 1, pp. 74–79, 2019.

[16] I. Darmawan, A. Pratama Abdul Karim, Aditya Rahmatulloh, R. Gunawan, and D. Pramesti, “JSON Web Token Penetration Testing on Cookie Storage with CSRF Techniques,” *2021 International Conference Advancement in Data Science, E-Learning and Information Systems, ICADEIS 2021*. 2021. doi: 10.1109/ICADEIS52521.2021.9701965.

[17] N. F. Sitorus, A. Kusyanti, and A. Bhawiguya, “Implementasi Autentikasi Berbasis Token Menggunakan Platform-Agnostic Security Tokens (PASETO) Sebagai Mekanisme Autentikansi RESTful API,” vol. 4, no. 11, 2020.

[18] S. Ebneyamini, “Towards Developing a Framework for Conducting Management Studies Using Design Research Methodology,” *Int. J. Qual. Methods*, vol. 21, pp. 1–7, 2022, doi: 10.1177/16094069221112245.

[19] Saransh Kataria, “How to Measure JavaScript Execution Time,” *wisdomgeek.com*, 2021. https://dev.to/saranshk/how-to-measure-javascript-execution-time-5h2#:~:text=The easiest way to track,the difference of the two (accessed Apr. 16, 2023).