ISSN 2278-3091
Volume 9, No.3, May - June 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse60932020.pdf
https://doi.org/10.30534/ijatcse/2020/60932020

Implementation of AES-128 and Token-Base64 to Prevent SQL Injection Attacks via HTTP

Muhammad Farras Muttaqin[1], Yaddarabullah[2], Silvester Dian Handy Permana[3]
[1]Universitas Trilogi, Indonesia, mfarrasmuttaqin@trilogi.ac.id
[2]Universitas Trilogi, Indonesia, yaddarabullah@trilogi.ac.id
[3]Universitas Trilogi, Indonesia, handy@trilogi.ac.id

ABSTRACT

SQL Injection is an attack that can be applied to all database servers that support SQL commands. This attack cannot only occur in client-side applications, but can also occur through the process of data communication between client-side applications with server-side to access web services using HTTP GET or POST parameters. There is an alternative solution to create a security system from SQL Injection attacks during data communication process, that is by applying the cryptographic algorithm to the name and value parameters during the data communication process in the GET or POST method. This study proposes the use of the Advance Encryption Standard 128 (AES-128) algorithm combined with Token-Base64. AES-128 algorithm is used in the encryption process and description of name and parameter values, then Token-Base64 for encoding and decoding processes with Token at the binary ciphertext result from the AES-128 encryption process. The testing method used is blackbox testing. SQL Injection Tools used are Web Cruiser which is assisted with the Pentest-Tools URL-Fuzzer web application to find out the URL address of the web service that can be used as a SQL Injection attack gap. The results of this study are that the combination of AES-128 and Token-Base64 algorithms can prevent SQL Injection attacks with a percentage of 100% of 83 attempts on data communication through HTTP GET or POST parameters. This states that the security level obtained is in the application layer of the OSI Model. The implementation of this security process makes the web service load time performance last longer by 31.26% with the file size of the web services being 95%, compared to not using the security process.

Key words : AES-128, Token-Base64, SQL Injection, Web service, Data communication process.

1. INTRODUCTION

The Open Web Application Security Project (OWASP) mentions the highest Application Security Risk or attack risk for web application security from 2013 to 2017 is through Injection. OWASP provides the highest risk score with a total of 8.0 for Injection attacks through calculations using OWASP Risk Rating Methodology [1]**.** There are several types of Injection attacks one of which is SQL Injection. This attack occurs when hackers inject or input malicious code into a web application and the code successfully interacts with database queries [2]**.** SQL Injection attacks can not only occur in client-side applications, but can occur through the process of data communication between client-side applications with server-side to access web services using HTTP GET or POST parameters [3][4]**.** In addition, SQL Injection also used to attack several applications such as steganography and authentication [5][6]. Based on these problems, an alternative solution will be investigated and developed in making a security system during the data communication process, that is by implementing an algorithm that is in cryptography [7]. The data to be encrypted is the name and parameter value during the communication process, either using the GET or POST method. The cryptographic algorithm that will be used is Advance Encryption Standard 128 combined with Token-Base64. Advanced Encryption Standard is a cryptographic algorithm with symmetric keys to carry out the process of encryption and description of data. AES has a constant 128-bit block size with 3 types of key sizes, 128 bits (AES-128), 192 bits (AES-192) and 256 bits (AES-256) [8][9]. The different key sizes used in the AES Algorithm affect the level of data security and time required for the encryption and description of the data. AES-128 has the lowest level of data security compared to AES 192 and AES 256, but AES 128 has the fastest data encryption and description process compared to AES 192 and AES 256 [10]**.** Based on a comparative study of several symmetric cryptographic algorithms namely AES, DES, IDEA and Blowfish conducted by Donzilio Antonio Meko, states that AES can encrypt data with a size of 1508 kb/s and describe data with a size of 1433 kb/s. This declare that AES has the highest data encryption and description speeds, while DES has the lowest data encryption and description speeds [11]. The results of the AES algorithm encryption process are presented in binary ciphertext data. Data in binary form cannot be accepted by the URL Encode during the data communication process via the HTTP protocol, because the URL Encode only translates special characters into forms that can be accepted by the URL in the HTTP protocol**.** Therefore, the AES algorithm is combined with the Base64 Algorithm to translate binary data into Base64 characters, so that data in

the form of Base64 characters can be received by the URL in the HTTP protocol [12]**.** After that, a token is added to the Base64 Algorithm to maintain user authentication from the client-side application. The combination of these two algorithms is used to prevent SQL Injection attacks in the data communication process via HTTP GET or POST parameters and will be applied to a multiplatform application called My Lost Phone.

2. LITERATURE REVIEW

Research conducted in 2018 with the title "Comparison of DES, AES, IDEA and Blowfish Algorithms in Data Encryption and Decryption" by [11], discusses the comparison of several symmetric cryptographic algorithms based on the time required in the process of data encryption and description. The results of this study are the AES Algorithm is relatively much faster than the Blowfish Algorithm, DES and IDEA in terms of speed in the process of data encryption and decryption. The results of a comparative analysis of encryption speed and description can be seen in Table 1.

Table 1: Comparison Analysis Results of Encryption

Algorithm	Encryption	Decryption
DES	402	608
AES	1.508	1.433
IDEA	173	57
Blowfish	1.063	1.057

Table 2.2 explains that the AES algorithm has the highest speed in terms of encryption and data description, with speeds of 1508 kb / s in the encryption process and 1433 kb / s in the description process. Whereas IDEA algorithm has the lowest speed in terms of encryption and data description, with a speed of 173 kb / s in the encryption process and 57 kb / s in the description process. The similarity with this research is AES Algorithm used for data encryption and description. Then, criticism given is AES algorithm has the highest speed in the process of encryption and description of data compared to the DES, IDEA and Blowfish algorithms, but it is not explained the key size of the AES used is 128, 192 or 256 bits. Finally, the idea development is AES algorithm is the best in terms of the process of encryption speed and data description. Therefore, the AES algorithm is very good to be applied in data communication processes that require speed in the security process. In a study entitled "Encrypt And Decrypt Image Using Vigenere Cipher" by [13], describes the process of encryption and image description using the Vigenere Cipher Cryptographic Algorithm. This research requires encoding and decoding algorithms to change the format of digital images in binary form to plaintext, then encrypted into ciphertext and description back to plaintext using Vigenere Cipher. Therefore, researchers used the Base64 Algorithm after making comparisons with similar encoding and decoding algorithms, namely Base8, Base16 and Base32. The advantage of Base64 Algorithm compared to other similar algorithms is that it can produce the number of characters with the smallest size during the encoding and decoding process. It makes the encryption and description process of Vigenere Cipher faster and optimal. The similarity with this research is Use of Base64 Algorithm for encoding and decoding data. Then, criticism given is the ability of the Vigenere Cipher algorithm to encrypt images is not safe, because there is already a method for breaking the data encoded by this algorithm. Finally, the idea development to produce better security, the Base64 algorithm needs to be combined with a more secure encryption algorithm. In 2018 a study entitled "Preventing Exploit URL of Pontianak STMIK Sensitek Website with Blowfish Algorithm" by [14], explains the implementation of Blowfish Algorithm to prevent attacks that utilize unencrypted URLs. The Blowfish algorithm is only used to encrypt the parameter values at the URL that will be taken by the GET method, so the attacker cannot manipulate the parameter values. The weakness of this research is that although it can prevent the attack of SQL Injection through the URL with the GET method, the security of encrypted data is imperfect. Because the Blowfish algorithm is not used to encrypt the name parameter in the URL that will be taken by the GET method. The similiarity with this research is in using of Cryptographic Algorithms to prevent SQL Injection attacks via HTTP GET parameter values. Then, criticism given is the security of encrypted data is imperfect. Because the Blowfish algorithm is not used to encrypt the name parameter in the URL that will be taken by the GET method. Finally, the idea development is The application of the Blowfish Algorithm needs to be developed to not only encrypt parameter values, but also encrypt the parameter name in the URL. In 2017 a study entitled "Implementation of AES (Advanced Encryption Standard) Algorithm for URL Encryption in Web-Based Asset Inventory Applications" by [15], describes the weaknesses of the GET method in asset inventory applications, in sending data from clients and servers to SQL Injection attacks. Because the GET method displays parameters sent from the client to the server at the URL. Therefore in this study the implementation of the AES Algorithm to encrypt the parameter is sent using the GET method in the URL. The results of web security testing against SQL Injection attacks using the Web Cruiser Web Vulnerability Scanner application are URLs with GET parameters that have gone through the encryption process are no longer detected as security holes. Weaknesses in this study are the ciphertext that is produced in the form of Base16 or Hexadecimal and there is still a security hole in the POST parameter. The similarity with this research is using AES Algorithm to prevent SQL Injection attacks. Then, criticism given is The combination of AES and MD5 Algorithm successfully handled the Second Order SQL Injection attack, but this study did not show detailed test results. Finally, the idea development is through the research results obtained, in addition to encrypting and describing data, the AES algorithm is very suitable to be used to prevent SQL Injection attacks. Research conducted in 2017 entitled "SQL Injection

Prevention Technique Using Encryption" conducted by [16], discusses how to prevent SQL Injection attacks using AES and MD5 cryptographic algorithms (Message Digit Algorithm). The combination of AES and MD5 Cryptographic Algorithms successfully handles all four types of SQL Injection attacks, namely Bypass Authentication, Unauthorized knowledge of database, Injected additional query and Second Order SQL Injection. Comparison of the combination of AES and MD5 cryptographic algorithms successfully handles the Second Order SQL Injection attack, where this attack cannot be prevented using the Blowfish cryptographic algorithm as presented in Table 2.

Table 2: Comparison of Cryptographic Algorithms to SQL Injection

SQL Injection Attacks	SQLIA Prevent Techniques			
	Blowfish	MD5	AES	AES + MD5
By pass Authentication	Prevented	Prevented	Prevented	Prevented
Unauthorized knowledge of database	Prevented	Prevented	Prevented	Prevented
Injected additional query	Prevented	Prevented	Prevented	Prevented
Second order SQL Injection	Non Prevented	Prevented	Prevented	Prevented

The similiarity with this research is using AES-128 Algorithm to prevent SQL Injection attacks in the process of data communication between client and server via HTTP GET parameters. Then, criticism given is The combination of the AES-128 Algorithm with the Base16 Algorithm successfully prevents SQL Injection attacks in the process of data communication between the client and server via the HTTP GET parameter. However, there are still security gaps in the HTTP POST parameter. Finally, the idea development is The combination of the AES-128 algorithm with Base16 must also be able to prevent SQL Injection attacks via HTTP POST parameters. The study entitled "Application of Base64 Cryptography for the Security of URLs (Uniform Resource Locator) Websites From SQL Injection Attacks" by [17], explains that URLs can be one aspect that becomes a weakness on a website. Because the URL contains various information that contains protocols, server addresses and file paths that can be used to perform SQL Injection actions. Based on the test results using one of the SQL Injection Tools namely Web Cruiser Web Vulnerability Scanner, it is known that the database can be accessed easily. However, if the Base64 process is applied to a website URL, server information and database information cannot be attacked by SQL Injection. Therefore, this can overcome threats to data security, especially from SQL Injection attacks. Application of Base64 Algorithm on Website URLs has disadvantages, because Base64 Algorithm is not intended to conceal data, but only changes the form or format of data to Base64 character form, so that the attacker can decode the Base64 character to find out the original value of the encoded data. Then the next drawback is not adding tokens to the Base64 Algorithm to maintain user authentication. The similarity with this research is using Base64 Algorithm to prevent SQL Injection attacks. Then, criticism given is Base64 algorithm is not intended to conceal data, but only changes the shape or format of the data to the Base64 character form, so that the attacker can decode the Base64 character to determine the value the original data that has been encoded. Finally, the idea development is the Base64 algorithm needs to be combined with the encryption and description algorithm to keep data confidential, so that data security can be more guaranteed.

3. METHODOLOGY

3.1 Analysis and Design

Token-Base64 algorithm is the result of adding Tokens to the encoding and decoding of the Base64 Algorithm. The token serves to maintain the user authentication of client-side applications, while the Base64 Algorithm functions to change the results of data encryption in binary form by the AES-128 Algorithm in the form of Base64 characters. The process of converting data in binary form to Base64 character form is called the encoding process, otherwise the decoding process functions to change the data in the form of Base64 characters into binary form. In this research, the process of designing a combination of AES-128 Algorithm with Token-Base64 in the encryption and encoding process is illustrated through the flowchart in Figure 1.

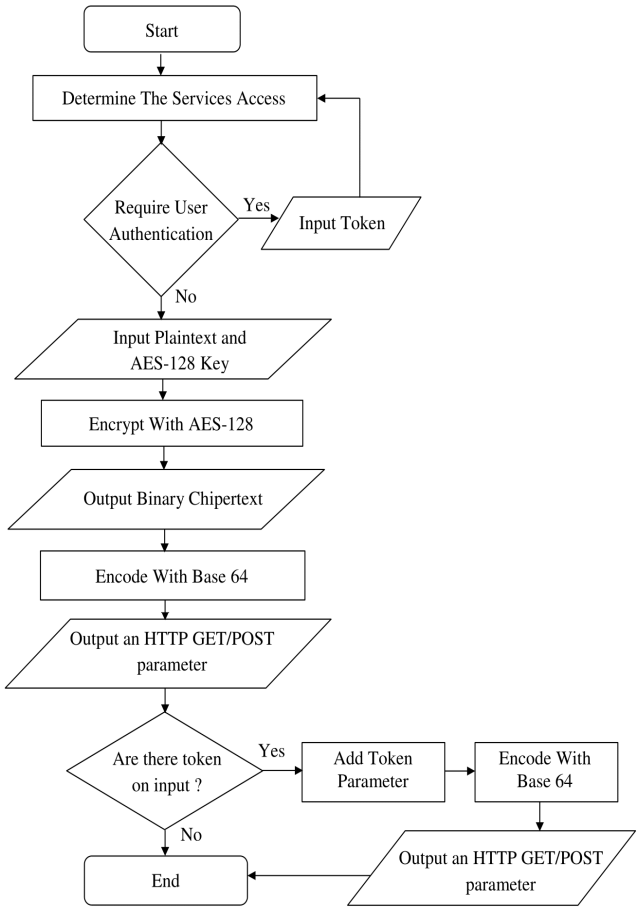

Figure 1: Encryption and Encoding Flowchart

In Figure 1. the combination of the AES-128 encryption process and Base64 encoding that uses Tokens starts from determining services that are accessed, if 'YES' accesses services that require user authentication that is services that can be accessed before logging in, Token input is added and added with plaintext input in the form of HTTP GET or POST parameters and AES-128 key, but if 'No' accesses services that require user authentication then directly plaintext input in the form of HTTP GET or POST parameters and AES-128 key. Then do the encryption process on the input data in the form of plaintext using AES-128 key. The result of the encryption process in the plaintext with the AES-128 algorithm is ciphertext in the form of binary bytes. The Binary Ciphertext will be encoded using the Base64 algorithm to convert the form of bytes to the Base64 character form. After the Encryption and Encoding process uses a combination of AES-128 and Token-Base64, then the description and decoding process are illustrated through the flowchart in Figure 3.

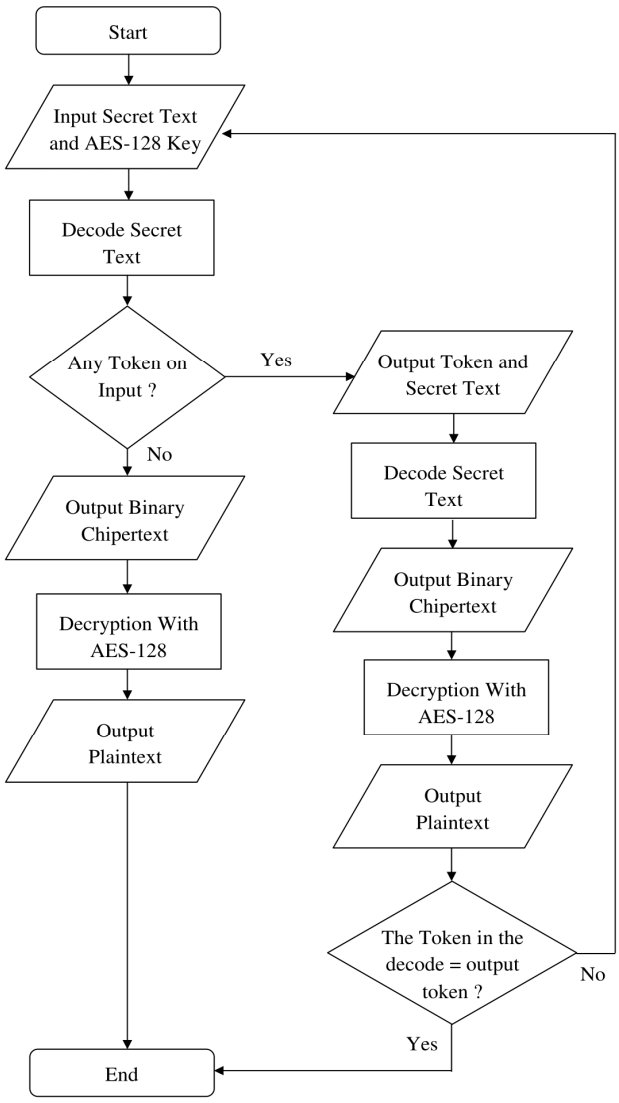

Figure 2: Description and Decoding Flowchart

In Figure 2. the combination of the AES-128 decryption process and Base64 decoding that uses Tokens starts from the AES-128 key input and secret text in the form of the Base64 character. Then decode the secret text in the form of Base64 characters, if the decoding process does not produce a Token parameter, the result of the decoding process is binary ciphertext. Then do the description process in binary using AES-128, to produce a plaintext in the form of HTTP GET or POST parameters. However, if the decoding process in secret text produces Token parameters, then 2 parameters are obtained, namely Token and Secret Text. Then decode the secret text to produce a binary ciphertext. After that, the description process uses AES-128, to produce a plaintext in the form of a Token parameter and other parameters sent using the GET or POST method. Then if the Token in the description results is not the same as the Token from the secret text decoding process for the first time, then user authentication or user authentication is "wrong" and goes to the initial process, whereas if the Token value is "the same" then the user authentication is "true" and a combination of AES-128 description process and Token-Base64 decoding are complete.

3.2 Implementation

The combination of the AES-128 Algorithm with Token-Base64 will be designed in the form of a library program and implemented in the Data Communication process between Client-side and Server-side in one application example, My Lost Phone. Then the Wireshark version 3.0.2 application is used to capture or capture the value of the encrypted and encoded data request parameters using a combination of AES-128 and Token-Base64 algorithms. Next Figure 3. is the coding of the encryption and encoding process flow in the combination of AES-128 and Token-Base64.

```
BEGIN
// Input_parameter encryption and token
(if accessing services that require user
authentication) uses AES-128

String plaintext = data_parameter+token;
String key = AES128_key; // 16 bytes key
byte[] state = plaintext.GETBytes();
byte[] key_state = key.GETBytes();

Do the AddRoundKey (state, w[0,3]);
Do the ExpandKey(key_state);

FOR round = 1 step 1 to 9
  Do the SubBytes(state)
  Do the ShiftRows(state)
  Do the MixColumns(state)
  Do the
  AddRoundKey(state,w[round*4,(round*4)+3])
END FOR

Do the SubBytes(state)
Do the ShiftRows(state)
Do the AddRoundKey(state,w[40,43])
ciphertext = state

//The combination of encryption results with
Token-Base64 Encoding
```

```
Count byte(ciphertext) length

IF (byte(ciphertext) length able to divided
    by 3)
THEN
  Count binary(ciphertext) length
  IF (the addition of padding 1 byte can be divided
      by 3)
  THEN
    Count binary(ciphertext) length
  ELSE
    Add 2 bytes on plaintext
    Count binary(ciphertext) length
  END IF
END IF

Create Grouping each 6 bits
Secret_text = Encode_result

IF (there is token on plaintext)
THEN
  Secret_text =
  Encode_result+"&access_token="+token;
  Secret_text = Base64Encode(Secret_text);
ELSE
  Secret_text = Encode_result
END IF
END
```

Figure 3: AES-128 and Token-Base64 Encryption and Encoding Process

In Figure 3. Shows, the combination of encryption and encoding process of AES-12 and Token-Base64 explained from the initial stages (declaration) and (initialization) plaintext and key to produce a Secret_text which becomes a parameter value during the process of data communication between client-side and server-side, using either HTTP GET or POST.After encrypting and encoding the HTTP GET or POST parameters and the value obtained from Secret_text, a description and decoding of the encrypted parameters is needed to get the true value back. Next, Figure 4. is a pseudocode of the AES-128 description process and the Token-Base64 decode.

```
BEGIN
String Secret_text = Secret_text;
String key = AES128_key; // 16 bytes key
byte[] key_state = key.GETBytes();

//Decode encryption and encode results (Secret_text)
with Token-Base64

Count padding(=) in Secret_text
Count indeks value in each base64 characters
Convert index values into binary sequences

IF (padding length == 1)
THEN
  delete 2 last bits
ELSE IF(padding length == 2) then
  delete 4 last bits
END IF

Convert every 8 bits into text
Display decode_result

IF (decode_result has token)
THEN
 byte[] ciphertext = Base64Decode(Secret_text)
 String plaintext = AES128.decrypt(ciphertext)
 IF (token decode_result == plaintext token)
 THEN
  // user authentication success
  String data_parameter = state
 ELSE
  // user authentication failed
  String data_parameter = "request gagal"
 END IF
ELSE
 byte[] state = hasil_dekode

 // Combination of decoded results with AES-128
    description

 Do the AddRoundKey(state, w[40,43]);
 Do the ExpandKey(key_state);
 FOR round = 9 step 9 to 1
  Do the InvShiftRows(state)
  Do the InvSubBytes(state)
  Do the AddRoundKey(state,w[round*4,(round*4)+3])
  Do the InvMixColumns(state)
 END FOR

 Do the InvShiftRows(state)
 Do the InvSubBytes(state)
 Do the s AddRoundKey(state,w[0,3])
 String data_parameter = state
END IF
END
```

Figure 4: AES-128 and Token-Base64 Descryption and Decoding Process

In Figure 4. Shows, the description and decoding process has a programming flow from the initial stages (declaration) and (initialization) to produce data_parameter is the actual data request value of the encoding process using a combination of AES-128 and Token-Base64.

3.3 Testing

At this stage, testing is done by the blackbox testing method. This process is carried out to determine the combination of AES-128 and Token-Base64 to prevent various SQL Injection attacks, namely Tautologies, Union, Illegal / logical incorrect queries, Piggyback queries, Inference and Stored procedures, where SQL Injection attacks use the MySQL language which is carried out through the process of data communication between client and server via HTTP GET or POST parameters. After applying the combination of AES-128 and Token-Base64 on the HTTP GET or POST parameters, the web services will be tested for performance through the results of the average load time, file size and performance grade.SQL Injection attack testing of HTTP GET or POST parameters will be performed using one of the SQL Injection Tools, Web Cruiser Web Vulnerability Scanner version 3.5.6, assisted with the Pentest-Tools URL-Fuzzer Web Application to find out the URL address of the web service that can be used as SQL Injection attack loopholes. Then use the Pingdom Website Speed Test application to perform performance testing of web services. An indication of the success of the test in this research is that it can prevent SQL Injection attacks through the process of data communication between client-side and server-side on the HTTP protocol, using either the GET or POST methods.

4. RESULT AND DISCUSSION

There are six types of SQL Injection attacks that will be carried out through communication on HTTP GET or POST, namely Tautologies, Union, Illegal / logical incorrect queries, Piggyback queries, Inference and Stored procedures. The final results of the SQL Injection test will be stated in the security level obtained based on the layer in the OSI Model, which is used as a data communication standard in computer networks. Following are the results of testing each SQL Injection attack in figure 5 and figure 6 below.

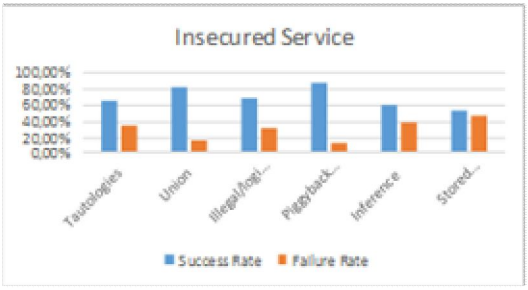

Figure 5: Testing Result in Insecured Service

Figure 6: Testing Result in Secured Service

In the figure 5 and 6 above, shows the results of test state that the combination of AES-128 and Token-Base64 Algorithms can prevent SQL Injection attacks by 100% in the process of data communication via HTTP GET or POST parameters. The combination of these two algorithms can protect the process of data communication between client-side and server-side via HTTP GET and POST parameters which are done through the application layer level as Security Level in the Open System Interconnection Layer. It states that the application of a combination of these two algorithms to the data communication process can prevent SQL Injection attacks carried out through the application layer.

Figure 7: Testing Web Services Performance

In figure 7 shows the implementation of this security function makes the load time performance last longer 31.26%. The grade achieved are the same between before and after the implementation of security functions. After that, the application of this security process also increases the file size on web services, with a total file size of 11.7 KB, which before applying the security process has a total of 6 KB, so the application of this security function increases the file size by 95%.

5. CONCLUSION

Based on the details of the test results, it can be concluded that the combination of AES-128 and Token-Base64 Algorithms can prevent SQL Injection attacks with a percentage of 100% of 83 attempts on data communication via HTTP GET or POST parameters. This states that the security level obtained is at the application layer of the OSI Model. Because the combination of these two algorithms can prevent SQL Injection attacks via the HTTP protocol found at the application layer level. After that, the conclusion of the web service performance test results in Figure 9, namely, the application of data communication security processes using a combination of AES-128 and Token-Base64 makes the web service load time performance last longer by 31.26% with file size web services becoming 95% greater , compared to not using a security process.

REFERENCES

[1] Owasp.org, **OWASP top 10 - 2017**, *Https://Owasp.Org*, vol. 1, no. 1, pp. 1–24, 2017.

[2] B. Gautam, J. Tripathi, and S. Singh, **A Secure Coding Approach For Prevention of SQL Injection Attacks**, *International Journal of Applied Engineering Research*, vol. 13, no. 11, pp. 9874–9880, 2018.

[3] Yaddarabullah, M. F. Muttaqin, and M. Rafiansyah, **Service-Oriented Architecture for E-Marketplace Model Based on Multi-Platform Distributed System**, in *IOP Conference Series: Materials Science and Engineering*, 2019, vol. 662, no. 4. https://doi.org/10.1088/1757-899X/662/4/042028

[4] A. Stasinopoulos, C. Ntantogian, and C. Xenakis, **Commix: automating evaluation and exploitation of command injection vulnerabilities in Web applications**, *International Journal of Information Security*, vol. 18, no. 1, pp. 49–72, 2019. https://doi.org/10.1007/s10207-018-0399-z

[5] S. K. Sonker, S. Kumar, A. Kumar, and P. Singh, **Image Based Authentication Using Steganography Technique**, *International Journal of Advanced Research in Computer Science*, vol. 4, no. 8, pp. 277–282, 2013.

[6] A. W. W. Permana, S. D. H. Permana, and

Yaddarabullah, **Modification Of Least Significant Bit Method With Redundant Pattern Encoding For Protection Of Message Integration From Image Modification**, *International Journal of Scientific and Technology Research*, vol. 9, no. 4, pp. 243–247, 2020.

[7] D. Naidu, S. Tirpude, K. Kalyani, V. Bongirwar, and T. Sharma, **Data Hiding using Meaningful Encryption Algorithm to Enhance Data Security**, *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 9, no. 2, pp. 2408–2413, 2020. https://doi.org/10.30534/ijatcse/2020/226922020

[8] Sailaja, S. Rao, and R. Kumar, **A New Circle based Symmetric key Encryption Technique for Text Data**, *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 8, no. 5, pp. 2573–2576, 2019. https://doi.org/10.30534/ijatcse/2019/106852019

[9] S. Pandey and M. Farik, **Best Symmetric Key Encryption - A Review**, *INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH*, vol. 6, no. 06, pp. 126–128, 2017.

[10] T. B. I. Guy-Cedric and S. R., **A Comparative Study on AES 128 BIT AND AES 256 BIT**, *International Journal of Scientific Research in Computer Science and Engineering*, vol. 6, no. 4, pp. 30–33, 2019. https://doi.org/10.26438/ijsrcse/v6i4.3033

[11] D. A. Meko, **Perbandingan Algoritma DES, AES, IDEA Dan Blowfish dalam Enkripsi dan Dekripsi Data**, *Jurnal Teknologi Terpadu*, vol. 4, no. 1, pp. 8–15, 2018.

[12] S. Wen and W. Dang, **Research on Base64 Encoding Algorithm and PHP Implementation**, in *International Conference on Geoinformatics*, 2018, vol. 26, no. 41661087.

[13] V. Vamshi, K. Reddy, and S. Bhukya, **ENCRYPT AND DECRYPT IMAGE USING VIGENERE CIPHER**, *International Journal of Pure and Applied Mathematics*, vol. 118, no. 24, pp. 1–8, 2018.

[14] Gat, **Mencegah Exploit URL Website Sensitek STMIK Pontianak Dengan Algoritma Blowfish**, *Jurnal Publikasi STMIK Pontianak*, vol. 7, no. 2, pp. 55–66, 2018.

[15] A. S. Putra, **Implementasi Algoritma AES (Advanced Encryption Standard) Untuk Enkripsi URL Pada Aplikasi Inventaris Aset Berbasis Web**, *Program Studi Teknik Informatika FTI-UKSW*, no. 672013214, 2017.

[16] M. Sood and S. Singh, **SQL INJECTION PREVENTION TECHNIQUE USING ENCRYPTION**, *International Journal of Advanced Computational Engineering and Networking*, vol. 5, no. 7, pp. 5–8, 2017.

[17] A. P. Nugraha and E. Gunadhi, **Penerapan Kriptografi Base64 Untuk Keamanan URL (Uniform Resource Locator) Website Dari Serangan SQL Injection**, *Jurnal Algoritma Sekolah Tinggi Teknologi Garut*, vol. 13, no. 1, pp. 491–498, 2016.