2022 IEEE 8th Information Technology International Seminar (ITIS)
Surabaya, Indonesia, October 19 – 21, 2022

Analysis of SQL Injection Attack Detection and Prevention on MySQL Database Using Input Categorization and Input Verifier

Alya Aiman Salsabila Arif
Crypto Software Engineering
Politeknik Siber dan Sandi Negara
Bogor, Indonesia
alya.aiman@student.poltekssn.ac.id

Rahmat Purwoko
Cryptographic Hardware Engineering
Politeknik Siber dan Sandi Negara
Bogor, Indonesia
rahmat.purwoko@poltekssn.ac.id

Nurul Qomariasih
Crypto Software Engineering
Politeknik Siber dan Sandi Negara
Bogor, Indonesia
nurul.qomariasih@poltekssn.ac.id

Hermawan Setiawan
Crypto Software Engineering
Politeknik Siber dan Sandi Negara
Bogor, Indonesia
hermawan.setiawan@poltekssn.ac.id

2022 IEEE 8th Information Technology International Seminar (ITIS) | 979-8-3503-9819-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/ITIS57155.2022.10010201

***Abstract*—Data leakage affects confidentiality and integrity, which can harm various parties. According to OWASP (Open Web Application Security Project) research, SQL injection attacks rank first in the top web application vulnerabilities. Moreover, the website is directly connected. SQL injection attacks are common on MySQL databases because they are generally more popular than other database systems. One of the efforts to detect and prevent SQL injection attacks is to use input categorization techniques and input verifiers based on input. Application development using SDLC Waterfall. The analysis is obtained from the test results using sqlmap and manually. This paper provides an overview of detection and prevention efforts with input categorization approaches and input verifiers based on the type of SQL injection attack. All applications without prevention and detection can be attacked, while applications with prevention and detection cannot be attacked. This paper designs and develops a web application with and without SQL injection attack detection and prevention using input categorization and input verifier. The results obtained, input categorization, and input verification techniques can detect and prevent SQL injection attacks based on their type, including union-based SQL injection, error-based SQL injection, and blind SQL injection. Input categorization and input verifier can be used in addition to the use of an encrypted database.**

Keywords— SQL Injection, input categorization, input verifier

I. INTRODUCTION

Databases are used in various fields, such as economics, education, and transportation, so it cannot be denied that there is always the possibility of attacks on these databases. These fields have important information databases that should not be known by unauthorized parties. The biggest impact is data leakage which results in data misuse. Data leakage affects confidentiality and integrity, which can be detrimental to various parties. The database breach targets are generally companies, organizations, and also e-commerce.

In May 2020, several e-commerce sites had data leaks, such as Tokopedia and Bukalapak. Tokopedia had a data leak of 91 million data, and Bukalapak had as much as 12.9 million data. Meanwhile, in November 2021 Indonesian National Police also leaked two databases containing important information from the personal data of police personnel [1]. The scattered data can be used for crime by unauthorized parties.

As the use of databases increases and database breaches occur, it is necessary to have a method so that the information in the database is safe from unauthorized parties. One way to secure databases is to use cryptographic algorithms. The cryptographic algorithm encrypts the data entered into a database. The data turns into meaningless or what is often called ciphertext. The ciphertext will replace the original data in the databases. However, cryptographic algorithms have several weaknesses, such as a weak level of security in symmetric algorithms and higher power usage in asymmetric algorithms [2].

SQL injection attacks can cause data leaks. Based on the 2017 OWASP (Open Web Application Security Project) research, SQL injection attacks occupy the first position in the top web application vulnerabilities [3]. The first large-scale SQL injection attack was launched in February 2020. The attack occurred on the Guess.com customer database, which allowed it to retrieve more than 200,000 customer names, credit card numbers, and expiration dates [4]. SQL injection attacks were reentered in 2021 by occupying the third position [5].

Types of SQL injection attacks based on research conducted in [3], include union-based SQL injection, error-based SQL injection, Boolean SQL injection, time-based SQL injection, and out-of-band SQL injection. Boolean-based SQL injection and time-based SQL injection are types of blind SQL injection. In contrast, out-of-band SQL injection is a rare type of SQL injection attack because it relies on features enabled by the database server in the web application [3].

The five types of SQL injection attacks can be narrowed down to the three most common types of attacks: union-based SQL injection, error-based SQL injection, and blind SQL Injection. In this study, the detection and prevention of SQL injection attacks will be analyzed based on these three types. These types are used because they do not depend on

the DNS or HTTP request features enabled by the database server in the web application.

In the Developer Survey in 2021, as many as 50.18% of respondents chose MySQL, so it was placed as the most popular database [6]. MySQL has an API (Application Programming Interface) facility that allows various computer applications written in various programming languages to access the MySQL database. Therefore, SQL injection attacks generally occur on MySQL databases because the use of MySQL databases is more popular than other database systems.

MySQL injection attacks detection and prevention can use input categorization and input verifier techniques[7]. These two techniques serve as a solid foundation for developing input checkers to detect and prevent SQL injection attacks automatically [8]. The input checker functions to determine user input with four categories, including SQL query keywords, special characters, alphabets, and numbers. Both techniques will detect user input and determine if it is safe from SQL injection. The result of these two techniques is to detect and prevent user input in the form of SQL injection attacks.

In this study, input categorization and input verifier were used in previous studies as a technique for the detection and prevention of SQL injection attacks [9]. Currently, SQL injection attack detection and prevention techniques use machine learning, but this research focuses on input-based analysis[10]. However, in this study, the analysis of SQL injection attack detection and prevention is not based on the type of SQL injection attack [9]. Meanwhile, SQL injection attacks generally occur on MySQL databases. Based on the results of previous studies, it is necessary to test using sqlmap and manually to find out the level of security against SQL injection attacks.[11].

So, in this study, an analysis of SQL injection testing was carried out using Sqlmap and manual tools. By comparing the 5 websites with different characteristics as mentioned in table II which use SQL injection attack detection and prevention techniques using input categorization and input verifier and 5 websites which do not use this technique. In addition, we also compare websites that use encryption to obtain input on the impact of attacks on applications based on the type of attack on the MySQL database. Input categorization and input verifier techniques as relatively simple methods can be alternatives for further development.

II. Basic Concepts

In this section, we explain some basic literature related to this study.

A. Database

The database can also be interpreted as a system that functions to store and process a set of data. Based on the previous explanation, it can be concluded that the database is a system that functions to store data so that it can be accessed easily and quickly. An example is in a university database that has entities such as students, faculty, courses, and classrooms. The relationship between these entities such as students who register to take courses, teaching staff who teach courses, and the use of classrooms for a course [12].

B. Database Management System

The database management system is software that was built to help detect and prevent the large amount of data needed in a system with speedy data growth. According to a survey conducted by Stack Overflow in 2021, there are fourteen database management systems used by respondents [6]. The most commonly used database management system is MySQL[13]. Therefore, SQL injection attacks generally occur on MySQL databases because the use of MySQL databases is more popular than other database systems.

C. SQL Injection Attack

Database attacks that are often found are SQL injection attacks. Based on the 2017 OWASP (Open Web Application Security Project) research, SQL injection attacks occupy the first position in the top web application vulnerabilities [14]. Meanwhile, in OWASP research in 2021, SQL injection attacks are in third position [5]. SQL injection attacks are divided into four types, including union-based SQL injection, error-based SQL injection, error-based SQL injection, and blind SQL injection.

D. Input Categorization and Input Verifier

Input categorization aims to detect malicious user input and protect merge operations during query generation [8]. To do this step, it is necessary to define the user input domain in four categories (keywords, special characters, alphabets, and numbers): Keywords consist of OR, UNION, and SELECT. Special Characters that is /, , $, (,), “, //, ?, +, !, @, =, ‘ …. Alphabets is [a – z, A – Z]. And Numbers is [0 – 9][15]. Input verifier aims to identify whether the input set is safe or not against SQL injection attacks.

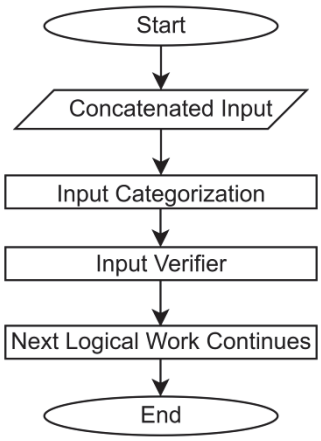

Fig 1. Flowchart Input Categorization and Input Verifier

III. Related Works

In [16] the author proposed query rewriting, encoding-based and assertion-based techniques to detect and prevent SQL injection attacks. This method is efficient for preventing and detecting SQL injection attacks, but they depend on the number of inputs. The proposed approach in [17], which is also used to detect and prevent SQL injection attacks, does not depend on the number of inputs. The authors provide an experimental result on the set benchmark programs. This model serves as a powerful basis to develop an input checker to detect and prevent the SQL injection attack automatically,

but the analysis of SQL injection attacks detection and prevention is not based on the types of SQL injection attacks. In this study, there will be an analysis of SQL injection attack detection and prevention techniques based on the types of attacks on the MySQL database using input categorization and input verifier.

TABLE I. COMPARISON OF PROPOSED TECHNIQUES

Techni-que	Auto-Detec-tion	Auto-Preven-tion	Identifica-tion of all input sources	Modi-fy Code	Time Comp-lexity
Query Rewriting	✓	✓	✓	✓	O(n)
Encoding-based	✓	✓	✓	×	O(n)
Assertion-based	✓	✓	✓	×	O(nm)
Input-based Analysis	✓	✓	✓	✓	O(n)

IV. RESEARCH METHODOLOGY

In this research, the design and development of web applications is carried out using the Waterfall SDLC which consists of planning, analysis, design, and implementation [18]. Web applications are shown in Figures 4 and 5. 15 web applications were designed and developed with 5 different database characteristics as shown in table II and 2 types of applications, both applications with categorization input and input verifier and applications with encrypted databases. Next is testing SQL injection attacks. Web applications will be tested for SQL injection attacks using automated tools such as sqlmap and manual. Manual testing will be based on type, including union-based SQL injection, error-based SQL injection, and blind SQL injection. Manual testing is performed by entering a malicious query into a field. SQL injection attacks can be successful when they can read sensitive data from the database. The test is carried out twice to ensure the first attack test. Testing was carried out with case studies of open-source PHP code with five different encrypted databases. The application runs on localhost. The research methodology is shown in Figure 2.

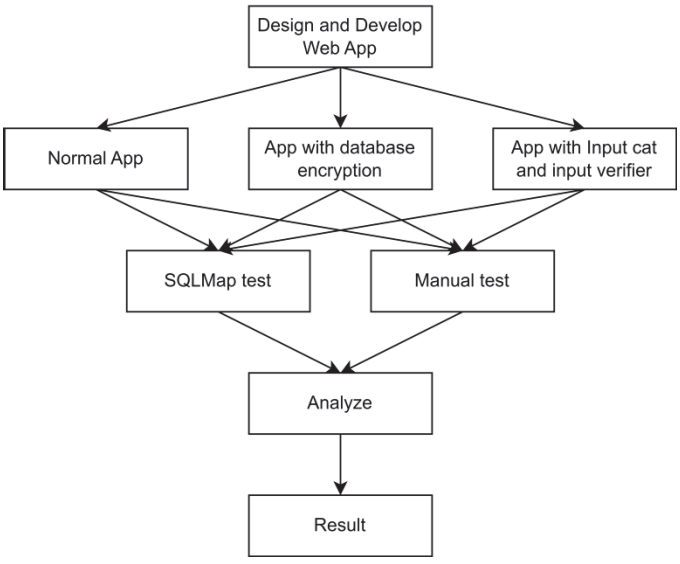

Fig 2. Research Methodology

TABLE II. DATABASES ON APPLICATION

Apps	Description
1	The password length is less than eight characters
2	The password length is at least eight characters
3	The password consists of lowercase and uppercase letters
4	The password consists of lowercase letters, uppercase letters, and numbers
5	The password consists of lowercase letters, uppercase letters, numbers, and symbols

Each application has one database with two tables. The first table is used on the login page containing three attributes, and the second is used on the crud page containing six attributes.

The designed system architecture is shown in Figure 3. A user can run the web application, while an attacker can attack it.

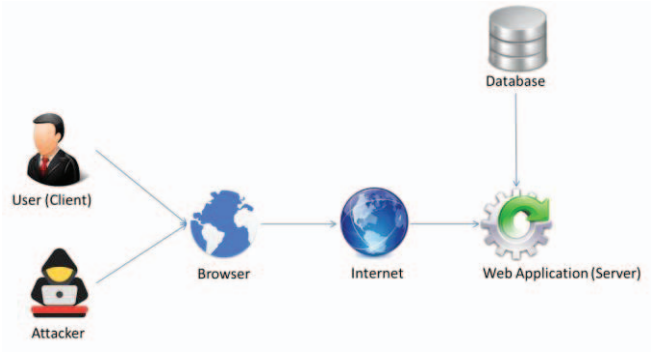

Fig 3. System Architecture

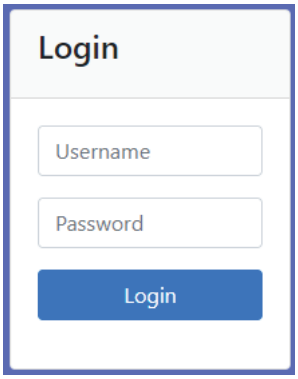

Fig 4. Login Page View

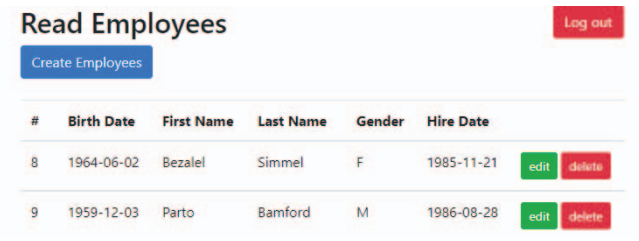

Fig 5. Read.php Page View

V. RESULT AND DISCUSSION

In this first part, the results of testing five web applications with categorization and input verifier techniques and five applications without these techniques are presented. When an application without this technique is attacked, the application will enter read.php in Figure 4. When an application with this technique is attacked, the application will notify of an attack and stop the work process in Figure 6.

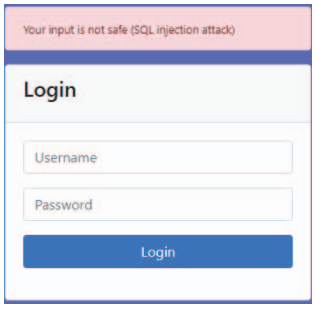

Fig 6. Login Page View after The Attack

The first test was carried out using the sqlmap tool on five applications. Table III contains the results per table, both the page name, the number of lines of code and the test results. from these results, either the login table or the crud data table in applications without detection and prevention can be attacked and applications with detection and prevention otherwise. The results of the SQLMap attack are in Figure 7., which shows an application without detection and prevention can be attacked by SQL injection. On the other hand, in Figure 8, SQLMap shows applications with detection and prevention that cannot be attacked by SQL injection [19].

TABLE III. APPLICATION TESTING DETAILS

Techniques		File	LOC	Test Result
Detection and Prevention	*Encryption*			
No	No	cek_login.php	25	✓
		update.php	87	✓
No	Yes	cek_login.php	25	✓
		update.php	87	✓
Yes	No	cek_login.php	25	×
		update.php	87	×

✓= Injected
× = No injected

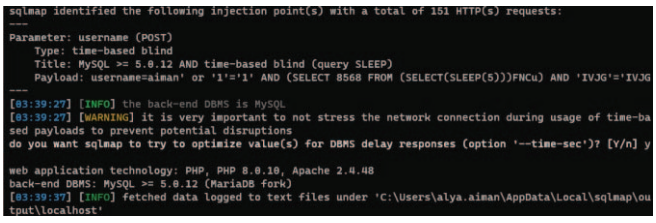

Fig 7. The View of Application that Cannot be Attacked by SQLIA

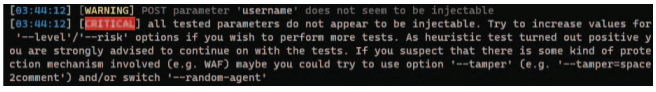

Fig. 8 The View of Application that Can be Attacked by SQLIA

Table IV shows the results of testing SQL injection attacks using sqlmap on each of the fifteen applications.

TABLE IV. TEST RESULTS USING SQLMAP

Techniques		Applications				
Detection and Prevention	Encryption	1	2	3	4	5
No	No	✓	✓	✓	✓	✓
No	Yes	✓	✓	✓	✓	✓
Yes	No	×	×	×	×	×

✓ = Injected
× = No injected

Based on the test results in Table IV, it can be concluded that SQL injection attacks with the help of SQLMap tools cannot inject applications by detecting and preventing SQL injection attacks.

Then a second test was carried out manually on fifteen applications. Blind SQL injection attacks are performed in check_login.php while union-based SQL injection and error-based SQL injection are performed in update.php. The table shows the results of testing SQL injection attacks using sqlmap on fifteen applications.

TABLE V. TEST RESULTS MANUALLY

Attack Type	Techniques		Test Result	Description
	Detection and Prevention	*Encryption*		
Union-based SQL Injection	No	No	✓	Read (no encrypted)
	No	Yes	×	No read (encrypted)
	Yes	No	×	No read
Error-based SQL Injection	No	No	✓	Read (no encrypted)
	No	Yes	✓	Read (no encrypted)
	Yes	No	×	No read
Blind SQL Injection	No	No	✓	Read (no encrypted)
	No	Yes	✓	bypassed
	Yes	No	×	No bypassed

✓= Injected
× = No injected

Table V shows that applications without the detection and prevention of unencrypted SQL injection attacks are mainly subject to injection by SQL injection attacks. While applications with encryption can be injected but cannot read the contents of the database with Union-based SQL Injection. Error-based SQL Injection and Blind SQL Injection cannot inject applications with encryption. And applications with SQL injection attacks cannot be injected by SQL injection attacks at all.

VI. CONCLUSION

In this research, fifteen web applications have been tested, with and without detection and prevention techniques. Injection testing is carried out automatically with the SQLMap tool and manually. The results with SQLMap show that applications with detection and prevention techniques successfully overcome injection attacks and applications without these techniques are successfully attacked by SQL injection. Manually, five applications without detection and prevention but with encryption, SQL injection cannot make SQL injection errors and blind SQL injection but can

perform union-based SQL injection to get information about the database used by the application. Meanwhile, the five applications with SQL injection detection and prevention were completely unable to perform union-based SQL injection, error SQL injection, and blind SQL injection, so they did not get the database information used by these applications. This research focuses on injection attacks in one of the detection and prevention methods, so in future research, the SQL injection attack detection and prevention method can be selected as a comparison so that it is useful as an improvement method in terms of other types of SQL injection attacks in the future.

REFERENCES

[1] H. Hafizhah, "Data Polri Bocor dan Dibagi Gratis oleh 'Stars12n,'" Nov. 18, 2021. https://www.republika.co.id/berita/r2rhcu487/data-polri-bocor-dan-dibagi-gratis-oleh-stars12n (accessed Dec. 02, 2021).

[2] B. E. Hamouda, "Comparative Study of Different Cryptographic Algorithms," *Journal of Information Security*, vol. 11, pp. 138–148, Oct. 2020.

[3] O. P. Voitovych, O. S. Yuvkovetsky, and L. M. Kupershtein, "SQL injection prevention system," in *2016 International Conference Radio Electronics & Info Communications (UkrMiCo)*, pp. 1–4, 2016.

[4] F. Q. Kareem *et al.*, "SQL Injection Attacks Prevention System Technology: Review," *Asian Journal of Research in Computer Science*, vol. 10, no. 3, pp. 13–32, 2021.

[5] OWASP, "OWASP Top 10:2021," 2021. https://owasp.org/Top10/ (accessed Nov. 30, 2021).

[6] Stack Overflow, "Stack Overflow Developer Survey 2021," 2021. https://insights.stackoverflow.com/survey/2021 (accessed Nov. 30, 2021).

[7] A. Jana, P. Bordoloi, and D. Maity, "Input-based Analysis Approach to Prevent SQL Injection Attacks," in *2020 IEEE Region 10 Symposium (TENSYMP)*, pp. 1290–1293, Oct. 2020.

[8] A. Jana, P. Bordoloi, and D. Maity, "Input-based Analysis Approach to Prevent SQL Injection Attacks," in *2020 IEEE Region 10 Symposium (TENSYMP)*, pp. 1290–1293, 2020.

[9] M. S. Aliero, A. A. Ardo, I. Ghani, and M. Atiku, "Classification of Sql Injection Detection And Prevention Measure," 2016. [Online]. Available: www.iosrjen.org

[10] M. al Rubaiei, T. al Yarubi, M. al Saadi, and B. Kumar, "SQLIA detection and prevention techniques," in *Proceedings of the 2020 9th International Conference on System Modeling and Advancement in Research Trends, SMART 2020*, pp. 115–121, Dec. 2020.

[11] M. Dalhatu and M. Hambali, "Manual Testing of SQL Injection Vulnerabilities in an Online Student Database System Using Same Channel Strategy," Oct. 2019.

[12] R. Ramakrishnan and J. Gehrke, *Database Management System*. McGraw Hill Higher Education, 2007.

[13] M. Reichardt, M. Gundall, and H. D. Schotten, "Benchmarking the Operation Times of NoSQL and MySQL Databases for Python Clients," in *IECON Proceedings (Industrial Electronics Conference)*, Oct. 2021, vol. 2021-October

[14] OWASP, "OWASP Top 10 Application Security Risks - 2017," 2017. https://owasp.org/www-project-top-ten/2017/Top_10 (accessed Nov. 30, 2021).

[15] Gupta, Archana, and D. S. Yadav. "An Approach for Preventing SQL Injection Attack on Web Application." International Journal of Computer Science and Mobile Computing (IJCSMC) 5.6, 01-10, 2016.

[16] B. K. Ahuja, A. Jana, A. Swarnkar, and R. Halder, "On Preventing SQL Injection Attacks," *Advances in Intelligent Systems and Computing*, vol. 395, no. November, pp. 65–84, 2016.

[17] A. Jana, P. Bordoloi, and D. Maity, "Input-based Analysis Approach to Prevent SQL Injection Attacks," in *2020 IEEE Region 10 Symposium (TENSYMP)*, pp. 1290–1293, 2020.

[18] V. Bhatnagar, "A compararative study of sdlc model," *IJAIEM*, vol. 4, pp. 23–29, Oct. 2015.

[19] W. Almadhy, A. Alruwaili, and S. Hendaoui, "Using SQLMAP to Detect SQLI Vulnerabilities," *IJCSNS International Journal of Computer Science and Network Security*, vol. 22, no. 1, 2022.