A SURVEY ON SQL INJECTION ATTACK: DETECTION AND PREVENTION

[1]**Anita Joy, [2]Dr.Renu Mary Daniel**

[1]Student, [2]Supervisor
Department of Computer Science and Engineering
APJ Abdul Kalam Technological University
Kerala, India

Abstract: **Web applications play an important role in the lives of millions of users across the world. Consequently, they have become a primary target of cyber-attacks. SQL injection attacks cause serious security threats to web application. It allows the attacker to interfere with the queries that an application makes to its database. Through these SQL injection attacks; attackers can gain unauthorized access to the database and get the confidential information of users and use it for illegal purpose. It will affect the confidentiality, authentication, and integrity of a web application. Sometimes it can go unnoticed to some period leading to a long-term compromise. This paper evaluates different types of SQL injection detection and prevention techniques.**

Keywords: **SQL Injection, Web application.**

I. INTRODUCTION

With the increasing use of web application, the data stored in the database also increases. Specifically, the data related to financial transaction, health information, personal information must be secured. According to the reports of Web Application Security Consortium [30], SQL injection attack poses the most serious threat to web application security. SQL Injection Attacks (SQLIAs) are directed towards web applications with back-end databases. The database can be accessed by the attacker by injecting malicious statements into the login field for execution. Subsequently the attacker can gain access to the resources or make changes to data stored in the databases. Such unauthorized interventions lead to data loss and destroy the functionality of a web application. A typical example includes a website login form, where an attacker can push malicious SQL commands to the connected database. There are different types of SQL injection attacks, and each has its own consequences on the application. Preventing SQL injection can be done through the discovery and repairing of the vulnerability. In 2016, hackers executed SQL injection attack against Qatar's National Bank. The attackers extracted 1.4GB data, which contain sensitive information of thousands of users like religious leaders, intelligence officials. A recent SQL injection attack on an e-commerce website led to an estimated loss of about 1 million dollars. SQL injection attack can also cause severe security issues in android applications, intelligent transportation system [4]. SQLIA are the most common attack and are becoming more complex in these fields. A successful SQL injection attack can steal credentials, access databases, alter data, delete data, and access networks. The SQL injection may happen through user input, cookies and through server variables. It compromises the safety, security, and trust of user data and the company's competitiveness or even the ability to stay in business. There are different ways to prevent SQL injection attack by using Prepared statements, Stored procedures etc. Still, research is being carried out to find an effective way for preventing SQL injection attack and its variants. This paper is organized as follows: Section I "Introduction" provides an overview of SQLIAs. Section II "Background" highlights the relevance of SQLIAs. Section III "Types of SQL injection attack" provides an overview of the different types of SQLIAs. Section IV "SQL Injection Detection and Prevention Techniques" gives the review of the existing tools and techniques that are used for detecting and preventing SQLIAs. Section V "Comparative Study" provides a comparison of the techniques used for the detection and prevention of SQLIAs. Section VI "Conclusion" depicts the conclusion of the survey.

II. RELATED WORKS

According to Open Web Application Security Project (OWASP), SQLIAs consists of the insertion of SQL commands through the input data from the user to the application. The process of SQLIA is shown in figure 1. When SQL injection becomes successful, it allows the attacker to spoof identity, alter existing data, cause repudiation issues such as voiding transactions. It allows the complete disclosure of all data on the system. Sometimes, the attacker becomes the administrators of the database server. SQLIAs are common type of injection attack, in which SQL commands are injected into the input, to modify the execution of the defined SQL commands. Even though this attack has been a known issue for years, there are numerous factors causing the risk to increase. First reason is due to the improper development of application and the use of insecure development architecture. Secondly, as more hackers gain skills in SQL injection, they are finding more applications and services that are prone to attack. The result is a dramatic increase in the opportunities to do this attack method. In this survey, SQLIA is acknowledged as an intentional use of the software functionality, caused by the execution of malicious input data.

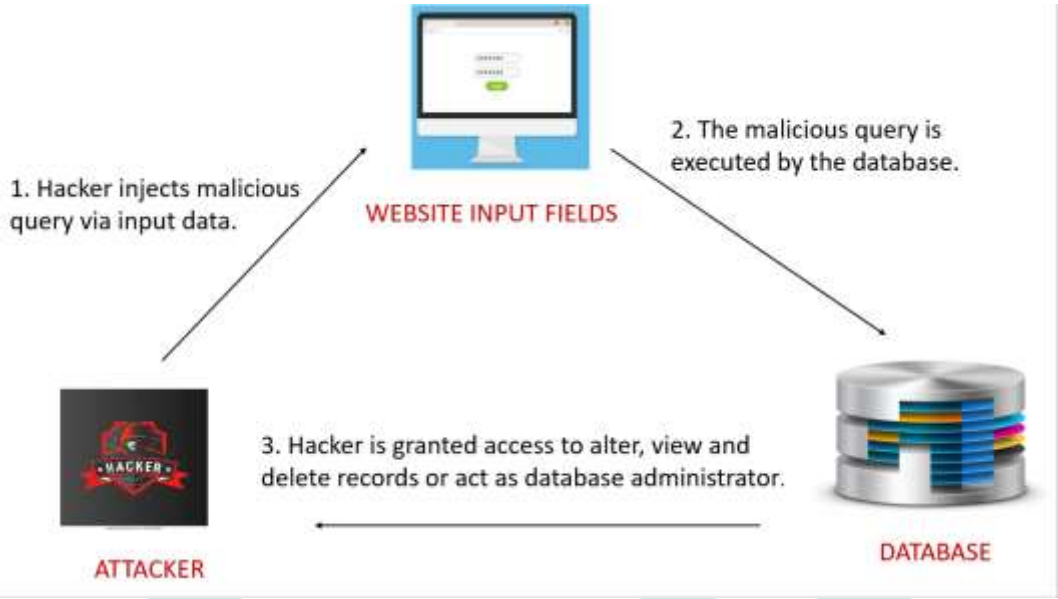

Figure 1: Illustration of SQL injection attack.

III. TYPES OF SQL INJECTION ATTACK

Classification of SQL injection attack is based on the damage potential and the method it uses to access the backend data. SQLIAs can come under three categories. They are In-band SQLIA, Inferential SQLIA and Out of-band SQLIA.

A. In-band SQLIA

Here the attacker uses the same communication channel to launch the attacks and to gather the results. It is one of the most common types of SQLIA attack because of its simplicity and efficiency. The two variants of this method are:

1) *Error based SQLIA:* It uses the error messages produced in the database. The data in the error message can be used to manipulate the data inside the database.

2) *Union based SQLIA:* It takes advantage of certain operators like SQL UNION to combine multiple SELECT statements to get a single HTTP response. This response contains data which can be used by the attacker.

B. Inferential (Blind) SQL injection

It is commonly referred to as blind SQLIA. Here the attacker learns about the structure of the server by sending payloads to the server and observing the response made by the server. It is called blind SQLIA because no data is transferred from the database to the server. It can be classified as follows:

1) *Boolean:* In Boolean based SQL injection, the attacker sends a SQL query to the database. The result depends on whether the query is TRUE or FALSE. The information within the HTTP response will get altered or remains unchanged based on the result.

2) *Time-based:* In time based SQLIA, the attacker observes the time the database takes to respond to SQL query. Response time indicates whether the result of the query is TRUE or FALSE. So based on this response time, attacker can determine whether the injected query is true.

C. Out-of-band SQLi

Out-of-band SQL injection occurs when an attacker cannot use the same channel to launch the attack and gather results.

IV. SQL INJECTION DETECTION AND PREVENTION

A. Non-Machine Learning Methods

Karis D'silva at [5] proposed an efficient technique for the prevention of SQLIAs. It uses hashing technique. It is like a unique digital fingerprint. For every user a unique fingerprint is created using hashing algorithm which is based on the credentials. This hash digest is matched with the hash digest stored in the database. The user is given access permission only if two hash digest matches. When an SQLIA happens, the hash digest will not match, thus preventing the attack. Rathod Mahesh Pandurang at [6] proposed a mapping model for the detection and prevention SQLIAs and Cross Site Scripting Attack. This method can be applied to static and dynamic web applications. For static web application, they stored valid download query as true mapping. For dynamic web application, they categorize the activity of each valid user. Based on these, each web request can be categorized. It can be used to differentiate between malicious and the legit mapping. This technique shows reduced execution time and server overhead. Maksy Sending at [8] implemented PDO (Php data object) for the minimization of SQL injection attack. PDO is used because, it ensures the security and flexibility when an application is connected to the database. PDO can be used when we need an application that support multiple databases. This technique is applied to the scheduling application for high school of Indonesia. It prevents union

and tautology attacks only. In [9], hashing technique is used to prevent SQL injection attacks. Instead of the unique fingerprint in [5], here the username and password are validated using a cryptographic Hash function. In the database two additional attributes are stored one hash value for username and the other hash value for the password. When the user creates an account, the hash value is automatically generated using Hash function algorithm and it is stored in the database. When user needs to login, the automatically generated hash value is compared with the one stored in the database. If it matches, the user can login. Otherwise, the request is rejected. In [10], a three-tier system is used for the detection and mitigation of SQL injection attacks. The three tiers operate on a database server, and it is associated with a cloud environment. It makes sure that the requested queries are running securely by preventing the hacking on the database. The architecture is composed of client logic, access, and data server. It removes the malicious query and makes sure that the system is prepared for an environment that is secure. It functions on a large-scale database. In order to implement this method, various technical attributes such as OS, RAM and hard disk are required.

In [12], a mechanism called SEPTIC is introduced for the prevention of DBMS attack. It identifies the attacks inside the DBMS. Mainly it uses two concepts. One is quarantine and the other one is aging. These concepts reduce the false negative and false positive rate. It shows a low performance overhead. One of the disadvantages of SEPTIC is that it needs some manual effort by the administrator to start the training. In [13], the proposed a technique is used for the detection and prevention of SQLIAs in Java applications. The framework contains certain critical variables, that accept user input from the external world. It then finds all the legal path the critical variable can take during its lifetime. Runtime monitors are used to compare the path taken by the critical variable with the predefined legal path. If any of the critical variable identified violates the legal path, it implies that the critical variable contains malicious input. It then notifies the administrator. The proposed method can detect and prevent tautology based SQLIAs only. Detnom at [26] proposed a technique for anomaly detection named DetAnom, which keeps a signature for every query that has been submitted and certain constraints that the application program should assure while submitting the query. The query is marked malicious, if there is a mismatch in the signature and if there is a violation in the constraint in the context of the application. In [27] the author used regular expression as well as finite automata for determining SQL injection attacks. The query will be scanned against some operators like '=', * etc. In [28] the author uses genetic programming approach and software testing technique to identify SQL injection vulnerabilities. Whereas in [29] author proposed a reversed insertion algorithm to prevent SQLIAs. This technique does need much programming work. Here all the input data are reversed, and a special character is inserted between them, then the data is stored in the database. Whenever the user login, it undergoes the same process, and it is compared with the data stored in the database.

The syntax of Uniform Resource Identifier (URI) in RFC 2616 [16] defines the malicious characters that should not appear in benign queries. A filter should be developed to remove the unsafe queries, such a filter, is proposed by Dong et al [17]. It focuses on the most common code injection attacks like SQLIA, XSS. The initial detection and the training of the model are done with a small set of labelled queries. Mitropoulos et al. [21] done a survey, that classifies 41 defense methods against XSS, SQL injection and other web application attacks, most of them do not use machine learning.

B. Machine Learning Methods

Qi Li at [1] proposed an AdaBoost algorithm based deep forest model (ADF). It has greater performance than the classical machine learning models and deep learning models. It has several advantages compared to the existing machine learning methods like better detection accuracy. One of the reasons is that the model parameters can be automatically adjusted during the training phase by using ADF. It has got high flexibility and robustness. The detection result depends on the hyper parameter setting and neural network selection. In practical implementation it is difficult to set and optimize hyper parameter. This becomes one of the main disadvantages of using ADF. In [14] the proposed method analyze the malicious query, and patterns of this query are mined by association rules. Using the concept of parse tree, it extracts the SQL features and construct feature vectors, and it is given as the input to the classifier. It uses random forest classifier for the detection. It can prevent and detect all differential attacks and cloning attacks. XIN XIE at [2] introduces a method based on Elastic Pooling CNN(EP-CNN). It detects the SQL injection attacks which are not recognized by traditional methods. CNN can extract features that cannot be observed by human and the features that are not obvious. For the training of CNN, it takes massive real web logs. The model is trained by randomly choosing the negative and positive sample. It selects some negative and positive sample as training and test set. The testing of the model is done using the test set. If the error is within an acceptable range, then the training can be stopped. This method extracts the hidden common features of SQL injection. It can also identify the network traffic. Anamika Joshi at [3] uses Naive Bayes and Role Based Access Control mechanism for the detection of SQL injection attack. It uses a dataset of malicious and non-malicious query. The smaller representation of such queries can be used as the test set. The algorithm uses two probabilities prior probabilities and posterior probability. The prior probability is calculated from the training set. The likelihood of a query being malicious is calculated from number of features that are matching the given test case and the total number of malicious queries. Similarly, the probability of non-malicious query is calculated. The final classification is done by combining both the prior and the likelihood, to form a posterior probability using the so-called Bayes' rule. Finally, we can classify the query as malicious or non-malicious depending on both prior probability and likelihood. This classifier also uses the role of user as a parameter for evaluation. Based on the user role (like administrator), the actions they can perform will be limited. Cai et al [18] uses Convolutional Neural Network (CNN) and Recurrent neural networks (RNN). It combines the deep learning techniques as well as the traditional query parsing technique. While Yan et al [19] put forward a Hybrid Deep Neural Network HDNN) to detect code injection vulnerability in hybrid applications. This hybrid method uses features that are extracted from Abstract Syntax Tree (AST) of JavaScript [20]. In [22], the authors proposed a model, to detect and prevent

SQLIAs at the runtime. It was developed based on SQL syntax-aware at the web application layer and negative taint at the database layer. Negative taint in database layer helps to identify untrusted data at the database layer. This technique prevents all classical types of SQLIAs. One of the drawbacks of this model is that it imposes very low overhead on the system. Bockermann et al. [15] put forward machine learning methods like clustering to detect the malicious behavior at the database level. In order to correlate SQL queries with application and distinguishing malicious and non-malicious queries, it incorporates the parse tree structure of SQL queries as features.

C. Reinforcement Learning

It is important to detect SQL injection vulnerabilities, to make the system secure. In [23] they proposed Q-learning reinforcement learning agents for the simulation of SQL injection vulnerabilities. It relies on reinforcement learning algorithm to exploit SQLIAs. Lopez-Martin et al. [24] state that for the code injection detection, reinforcement has to be limited to only one numeric reinforcement value, i.e., successful detection. They presented a survey on the application of reinforcement learning to address the issues of network intrusion detection.

D. Pattern matching

Oluwakemi Christiana Abikoye at [7] proposed Knuth Morris-Patt string matching algorithm for detecting and preventing SQL injection and XSS attacks. It uses PHP scripting language and xampp server. It formulates the different SQL injection string pattern. Using KMP string matching algorithm, it compares the input string with the patterns that have been formulated. For the prevention of SQL injection attack, a filter function is formulated. If the function returns true, an injected string is found. It then provides some warning messages and blocks the user. Otherwise, the permission for access is granted. Whereas Aho–Corasick in [25] proposed pattern matching algorithm for the detection and prevention for SQLIA. Then the user entered SQL queries are checked by running static pattern matching algorithm. If any form of anomaly occurs, then the new pattern will be added to the existing static pattern list.

V. COMPARATIVE STUDY

This section provides a comparative study of all the detection and prevention techniques against SQLIAs mentioned in Section 4. In Table 1 the comparison contains a (✓) tick symbol to indicate that the code modification is required and (×) cross symbol to indicate that the code is not modified. In Table 2 (✓) tick symbol to indicate that the technique will be able to deal with the attacks and (×) cross symbol indicates that the technique will not be able to detect or prevent the attacks.

Techniques	Protection Type	Additional Infrastructure	Code modified	Detection Time	Detection Location	Percentage of SQLIAs controlled
[8]	Prevention	Not Required	×	Coding Time	Client Side	60%
[27]	Detection	Not Required	×	compile Time/Runtime	Client/Server Side	40%
[28]	Detection	Required	×	Runtime	Server Side	50%
[29]	Prevention	Required	✓	Runtime	Client Side	50%

Table1: Comparison Of The Different Technique For SQL Injection Detection And Prevention Based On Specific Evaluation Parameters.

Attacks → Techniques ↓	Tautologies	Illegal Queries	Blind SQL Injection	Piggy Backed Queries	Union Query	Timing Attack
[8]	✓	✓	✓	✓	✓	×
[22]	✓	✓	✓	✓	✓	✓
[27]	✓	✓	×	✓	×	×
[28]	✓	✓	✓	✓	✓	×
[29]	✓	✓	✓	✓	✓	×

Table 2: Comparison of The Different Technique for SQL Injection Detection And Prevention With Respect To The Types Of Attacks.

Techniques	Learning	Language
[19]	Hybrid	JavaScript
[18]	CNN, RNN	SQL
[15]	Clustering	SQL
[3]	Bayesian	SQL
[29]	CNN	SQL

Table 3: Comparison SQL Injection Detection and Prevention Techniques Based on Machine Learning.

VI. CONCLUSION

It is evident from the analysis that SQLIAs cause a serious threat to web application security, and its consequence are severe. In this paper the survey explored the different type of SQLIAs, and its detection and prevention methods. In real time, passively detecting SQL injection is not helpful, for preventing it. Hence the efficiency of some techniques needs to be improved. This paper analyzed at least 26 different methods of SQL injection detection and prevention using various types of non- machine learning and machine learning algorithms.

REFERENCES

[1] Qi Li, Weishi Li, Junfeng Wang ,"A SQL Injection Detection Method Based on Adaptive Deep Forest", vol.7, pp.145385-145394, 2019.

[2] Xin Xie, Chunhui Ren, Yusheng Fu , Jie Xu , And Jinhong Guo," SQL Injection Detection for Web Applications Based on Elastic-Pooling CNN", vol. 7, pp.151475-151481, 2019.

[3] Anamika Joshi Geetha," SQL Injection Detection using Machine Learning", pp. 1111-1115,2014.

[4] Qi Li, Fang Wang Junfeng Wang and Weishi," LSTM-Based SQL Injection Detection Method for Intelligent Transportation System", vol.68, pp. 4182-4191, 2019.

[5] Karis D," An Effective Method for Preventing SQL Injection Attack and Session Hijacking",pp. 697-201, 2017

[6] Rathod Mahesh Pandurang,"A Mapping-based Podel for Preventing Cross Site Scripting and SQL Injection Attacks on Web Application", pp. 414-418, 2015.

[7] Oluwakemi Christiana Abikoye,"A novel technique to prevent SQL injection and cross-site scripting attacks using Knuth-Morris-Pratt string match algorithm", pp. 14, 2020.

[8] M. Sendiang, A. Polii, J. Mappadang, "Minimization of SQL Injection in Scheduling Application Development", 2017.

[9] S. P. Singh, U.N.Tripathi, M. Mishra "Detection and prevention of SQL injection attack using hashing technique", vol: 2, pp. 27-31, 2014.

[10] Wahid Rajeh,Alshreef Abed, "A Novel Three-Tier SQLi Detection and Mitigation Scheme for Cloud Environments", 2017 .

[11] I. Lee, S. Jeong, S. Yeo, and J. Moon, "A novel method for SQL injection attack detection based on removing SQL query attribute values," Mathematical and Computer Modelling", vol.55, pp.58-68, 2012.

[12] Ib´eria Medeiros, Miguel Beatriz, "SEPTIC: Detecting Injection Attacks and Vulnerabilities Inside the DBMS", vol. 68, pp.1168-1188, 2019.

[13] Ramya Dharam and Sajjan G. Shiva, "Runtime Monitors for Tautology based SQL Injection Attacks", pp. 253-258, 2012.

[14] Jianguo Zheng,**"**Pattern Mining and Detection of Malicious SQL Queries on Anonymization Mechanism", vol.9, pp.15015-1507,2021.

[15] Bockermann Christian, Martin Apel, Meier Michael, "Learning SQL for database intrusion detection using context-sensitive modelling", vol. 5587, pp. 196–205, 2009.

[16] Fielding Roy, Jim Gettys, Jeffrey Mogul, Henrik Frystyk, Larry Masinter, Paul Leach, Berners-Lee Tim. Hypertext transfer protocol–HTTP/1.1. 1999.

[17] Dong Ying, Zhang Yuqing, Ma Hua, Wu Qianru, Liu Qixu, Wang Kai, Wang Wenjie, "An adaptive system for detecting malicious queries in web attacks", vol. 61, pp. 16, 2018.

[18] Cai Ruichu, Xu Boyan, Zhang Zhenjie, Yang Xiaoyan, Li Zijian, Liang Zhihao,"An encoder-decoder framework translating natural language to database queries", vol.18, pp. 3977–3983, 2018

[19] Yan Ruibo, Xiao Xi, Hu Guangwu, Peng Sancheng, Jiang Yong, "New deep learning method to detect code injection attacks on hybrid applications", vol.137, pp. 67–77, 2018.

[20] Xi Xiaoa, Ruibo Yana, "Detection and Prevention of Code Injection Attacks on HTML5-based Apps", pp. 254-261, 2015.

[21] Mitropoulos Dimitris, Louridas Panos, Polychronakis Michalis, Dennis Keromytis Angelos, "Defending against web application attacks: approaches, challenges, and implications", vol. 16, pp. 188–203, 2019.

[22] A. Alazab , A. Khresiat , " New Strategy for Mitigating of SQL Injection Attack", International Journal of Computer Applications (IJCA), Vol, 154, pp.11, 2016.

[23] Laszlo Erdodi, "Simulating SQL injection vulnerability exploitation using Q-learning reinforcement learning agents", pp.20,2021.

[24] Lopez-Martin Manuel, Carro Belen, Sanchez-Esguevillas Antonio,"Application of deep reinforcement learning to intrusion detection for supervised problems", vol. 141, pp. 15, 2020.
[25] Nency Patel, Narendra Shekokar," Implementation of pattern matching algorithm to defend SQLIA", International Conference on Advanced Computing Technologies and Applications, vol. 45, pp. 453 – 459, 2015.
[26] Bossi, L., Bertino, E., Hussain, S.R," A system for profiling and monitoring database access patterns by application programs for anomaly detection", IEEE Transactions on software engineering, vol. 43, pp. 415-431, 2017.
[27] Qbea'h, M., Alshraideh, M., Sabri, K.E,"Detecting and preventing sql injection attacks: a formal approach. In: Cybersecurity and Cyberforensics Conference", pp. 123-129, 2016.
[28] Aziz, B., Bader,, M., Hippolyte, C, "Search based sql injection attack testing using genetic programming", In: European Conference on Genetic Programming (EuroGP). pp. 183-198,2016.
[29] Raj, Shaji.N., Sherly Elizabeth," An SQL injection defensive mechanism using reverse insertion technique",In: International Conference on Next Generation Computing Technologies, pp. 335-346, 2017.
[30] **https://resources.infosecinstitute.com/topic/sql-injection-vulnerability/**