UGC Approved Journal

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified

Vol. 4, Issue 9, September 2017

Detection and Prevention of SQL Injection Vulnerabilities in Web Applications: A Review

Vamsi Mohan V[1], Dr. Sandeep Malik[2]

Department of Computer Science, School of Engineering and Technology, Raffles University, Neemrana, India[1,2]

Abstract: In recent time, web security has been viewed in the context of securing the web application layer from attacks by unauthorized users. The vulnerabilities existing in the web applications are targeted through SQL Injection attacks (SQLIA). In this paper, we explained various method of detection and preventive measures on SQL Injection Attacks through a systematic review process.

Keywords: SQL Injections, Web Application Security, SQLIA, Web Vulnerabilities, Security Flaws.

I. INTRODUCTION

The OWASP Top 10 - 2017 reflects a move towards modern, high-speed software development that we've observed across the IT industry since the last version of the Top 10 in 2013. While many of the vulnerabilities remain the same, and primarily SQL Injection is top 1 in the list from the last 5 years.
The new APIs and attack protection mechanisms couldn't stop the SQL injection attacks. SQLI (SQL injection) is the one of critical web-based database driving application vulnerabilities that presents high risk. SQLI vulnerability results from inappropriate validation of input from user, which enables the attacker to manipulate programmer intended queries by adding new SQL operators, commands, or keys, thereby overriding the authentication mechanism and unauthorized database modification.

II. REVIEW FROM LITERATURE

Atefeh Tajpour, Suhaimi Ibrahim, & Mohammad Sharifi (2012) described SQL injection is a type of attack, which the attacker adds Structured Query Language code to a web form input box to gain access or make changes to data. SQL injection vulnerability allows an attacker to key-in commands directly to a web application's underlying database and destroy functionality or confidentiality. Many researchers have proposed variety of tools to detect and prevent this vulnerability.

In their study, Atefeh Tajpour et al., described in their publication, that SQLIA is a class of code injection attacks that takes advantage of lack of user input validations. Attackers can form illegitimate input as part of the final query string, which operates by the database.

Shaimaa Ezzat Salama, Mohamed I. Marie, Laila M. El-Fangary & Yehia K. Helmy (2012) described Databases in the background of e-commerce applications are more vulnerable to SQL injection attacks, which is considered as one of the most dangerous web attacks.

Makera M Aziz and Dena Rafaa Ahmed (2015) introduced a method in an Annual Conference on Network Security & Distributed Systems, that can be used to prevent SQL injection. They narrated SQL injection represents one of the most important threads for the web applications. Injection attacks are one of the top 10 threads for application security.
Muhammad Saidu Aliero, Abdulhamid Aliyu Ardo, Imran Ghani & Mustapha Atiku (2016) conducted a study on SQL Injections detection and prevention measures. They posted the SQL injection vulnerability is the one of the most common web-based application vulnerabilities, that can be exploited by SQL injection attack to gain access to restricted and confidential data, bypass authentication mechanism, and execute unauthorized data manipulation language. They advised, Defensive coding is a simple and affordable way to tackle this problem, however, there are some issue regarding the use of defensive coding which makes the system ineffective, less resistant and resilient to attack.

III. DETECTION

Atefeh Tajpour et al. propose WAVES, a blackbox technique for testing web applications for SQL injection vulnerabilities. This tool detects the pitfalls of web application that can be used to inject SQLIAs. It is built to target

UGC Approved Journal

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified

Vol. 4, Issue 9, September 2017

these points and monitors the application, how response to the attacks by utilizing machine learning. Also discussed the tools JDBC-Checker, CANDID, SecuriFly which were developed on the open source Java platform. In their paper, they are written about input runtime checkers SQL Guard and SQL Check tools. For static analysis and runtime monitoring, Atefeh Tajpour et al. recommends AMNESIA, WebSSARI.

Shaimaa Ezzat Salama et al., proposed a framework based on misuse and anomaly detection techniques to detect SQL injection attacks. This framework creates a profile for legitimate database behavior extracted from applying association rules on XML file containing queries submitted for application to the database. As a second step in the detection process, the structure of the query under observation will be compared against the legitimate queries stored in the XML file. It helps to minimize false positive alarms.

They also advised a framework that combines the two IDS techniques (Intrusion Detection System), misuse and anomaly detection techniques, to defend against SQLIA. The primary intention of using Web Anomaly Misuse Intrusion Detection (WAMID) framework is to create a profile for web applications that can represent the normal behavior of application users in terms of SQL queries, they submit to the database and collect logs of their usage. Then used an anomaly detection model based on data mining techniques to detect queries that deviate from the profile of normal behavior. The queries retrieved from database log are stored in the format of XML.

Association rules are applied to the XML file, to retrieve relation between each table in the query with each condition in the selection part. These rules represent the profile of normal behavior and any deviation from this profile will be considered an attack. To detect SQLIA and to minimize false alerts, WAMID framework as a second step uses misuse technique to detect any change in the structure of the query.

IV. PREVENTIVE MEASURES

Muhammad Saidu Aliero et al., detailed background of SQLIA (SQL Injection Attack), classified defensive coding to different categories, reviewed existing technique that are related to each technique, state strength and weakness of such technique, evaluate such technique based on number of attacks they could stop and evaluate each category of approach based on its deployment requirements related to inheritance.

In their publication, Muhammad Saidu Aliero et al., stated that one of the objectives of defensive coding is to write secure queries to get the behavior in a predictable manner. They explained that secure codes are being added to the application, after successful completion of web development and functionality testing. However, issues sometimes arise after secure codes are being added to the web application.

Due to lack of knowledge or insufficient testing, it results malfunction of the application. Some defensive coding strategies like Cryptographic Approach, XML Approach, Pattern Matching Approach, Parsing Approach, Machine Learning Approach and Parameterized Query and Stored Procedure writing evaluated and given recommendations.

An interesting thing about Injection through cookies concept explained by Muhammad Saidu Aliero et al. Cookies are structures that maintain persistence of web application by storing state information in the client‘s machine. When a client returns to a Web application, browser cookies can be used to restore the client‘s state information. If a Web application uses the cookie‘s contents to build SQL queries, then attackers can takes this opportunity to modify cookies and submit to the database.

Muhammad Saidu Aliero et al., discussed about the possibility Injection through server variables. Server variables are a collection of variables that contain HTTP, network headers, and environmental variables. Web applications use these server variables in different ways, such as session usage statistics and identifying browsing trends. If these variables are logged to a database without sanitation, this could create SQL injection vulnerability. Because attackers can forge the values that are placed in HTTP requests and network headers-by entering malicious input into the client-end of the application.

In their study, Muhammad Saidu Aliero et al., described about Second order injection, where attackers implant malicious code into a system or database to indirectly trigger an SQLIA, when that code is called later. When the attack occurs, the input that modifies the query to interpret an attack does not come from the user, but from within the system itself.

Makera M Aziz and Dena Rafaa Ahmed experimented and proposed a system to prevent SQL injections. They converted the username as a string (s1) and moved all other usernames to one class (c1). Compares the string(s1) with the other usernames in the class (c1). If it matches the comparison, it is success else it is failed. The same procedure

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified

Vol. 4, Issue 9, September 2017

performed for password also. Post performing the comparisons, they again check whether the username and password are from the same record. If it is yes, the result will be 'Login Successful' else 'Unsuccessful'.

Makera M Aziz and Dena Rafaa Ahmed explained the goal behind converting the input to a string is to make user input as a single unit (one token) that cannot use as a SQL query statement. The system will call the database attribute in such a way in which user cannot access to the SQL statement to do the injection. And the SQL query will be empty from any input tools that can use by user to inject the SQL

V. CONCLUSION

The Vulnerabilities exists naturally in web applications. It can be exploited by SQL injection attacks to get the benefit from the pitfalls of the application. For preventing the SQLIAs, defensive coding has been suggested as a solution. However, it is difficult and not only developers try to put some controls in their source code, but also attackers continue to bring some new ways to override these controls. Implementing the best practice of defensive coding is very difficult and need to have specialized skills. These issues motivate the need for a solution to the SQL injection attacks.

REFERENCES

[1] Atefeh Tajpour , Suhaimi Ibrahim, & Mohammad Sharifi (2012), Web Application Security by SQL Injection DetectionTools. IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 2, No 3, March 2012, ISSN (Online): 1694-0814.Bandhakavi, S., Bisht, P., Madhusudan, P., and Venkatakrishnan V., "CANDID: Preventing sql injection attacks using dynamic candidate evaluations", in the Proceedings of the 14th ACM Conference on Computer and Communications Security, 2007

[2] Bangre, Shruti, and AlkaJaiswal(2012) "SQL Injection Detection and Prevention Using Input Filter Technique." International Journal of Recent Technology and Engineering (IJRTE)145-149.

[3] Das, Debasish, Utpal Sharma, and D. K. Bhattacharyya (2010) "An Approach to Detection of SQL Injection Attack Based on Dynamic Query Matching." International Journal of Computer Applications 28-34.

[4] Kumar, Kuldeep, Debasish Jena, and Ravi Kumar (2013). "A Novel Approach to detect SQL injection in web applications." International Journal of Application or Innovation in Engineering & Management (IJAIEM) Volume 2, Issue ISSN 2319 - 4847

[5] Makera M Aziz and Dena Rafaa Ahmed (2015), Proposed Method to Prevent SQL Injection Attack. The Annual Conference on Network Security & Distributed Systems (NSDS'2015).

[6] Muhammad Saidu Aliero, Abdulhamid Aliyu Ardo, Imran Ghani & Mustapha Atiku (2016), Classification of Sql Injection Detection And Prevention Measure. IOSR Journal of Engineering (IOSRJEN), ISSN (e): 2250-3021, ISSN (p): 2278-8719, Vol. 06, Issue 02 (February. 2016), ||V1|| PP 06-17.

[7] N. Khochare, S. Chalurkar ,S. Kakade, B.B. Meshramm, "Survey on SQL Injection attacks and their countermeasures", International Journal of Computational Engineering & Management (IJCEM), Vol. 14, October 2011

[8] Ritu Gaur, Ravi Bhushan; "Protection against SQL Injection Attack on Web Applications Using AES and Stored Procedure", IJARCSSE, Vol 4, Issue 5, 2014

[9] Shaimaa Ezzat Salama, Mohamed I. Marie, Laila M. El-Fangary & Yehia K. Helmy (2012), Web Anomaly Misuse Intrusion Detection Framework for SQL Injection Detection. (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 3, No. 3, 2012.

[10] W.G.Halfond, J.Viegas, and A.Orso, "A classification of SQL-Injection Attacks and Countermeasures", in proceeding of the International Symposium on Secure Software Engineering (ISSSE), 2006