IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 2, December 2023

Data Leaks Using SQL Injection

Tahseen Ali[1], Rohan Dipke[2], Vinayak Dhanshetti[3], Nikhil Gadepally[4]
Assistance Professor, Department of Computer Enggineering[1]
Students, Department of Computer Enggineering[2,3,4]
Gramin Technical & Management, Nanded, India

Abstract: *The security of digital data is paramount in today's interconnected world. Among the various cyber threats, SQL injection attacks represent a significant menace to the confidentiality, integrity, and availability of sensitive information stored within databases. SQL injection is a technique employed by malicious actors to exploit vulnerabilities in web applications that interact with databases, allowing unauthorized access to or manipulation of the data. This paper presents an in-depth analysis of SQL injection attacks, their mechanisms, and the potential risks they pose to organizational data. It examines various preventive measures and best practices to mitigate the vulnerabilities that lead to SQL injection. Techniques such as input validation, parameterized queries, and the use of prepared statements are explored as effective defences against these attacks*

Keywords: Parameterized Queries, Escaping Special Characters, Least Privilege Principle, Regular Security Audits, Use of ORM (Object-Relational Mapping) Libraries & Web Application Firewalls (WAFs)

I. INTRODUCTION

SQL injection is a prevalent cybersecurity threat that exploits vulnerabilities in web applications using a Structured Query Language (SQL) database. It occurs when attackers insert malicious SQL statements into input fields, exploiting the system's susceptibility to execute unintended commands. This unethical practice allows unauthorized access to sensitive data, manipulation, or even deletion of crucial information. To prevent data leaks caused by SQL injection, robust security measures are imperative. Employing parameterized queries is an effective defines strategy. Parameterized queries utilize placeholders for user inputs, which are then separately handled from the SQL command. This technique prevents malicious SQL code injection by treating inputs as data rather than executable commands, significantly reducing the risk of exploitation.

II. MOTIVATION

The internet is used by the general population for the purposes such as financial transactions, educational endeavours, and countless other activities. The use of the internet for accomplishing important tasks, such as transferring a balance from a bank account, always comes with a security risk. Today's web sites strive to keep their users' data confidential and after years of doing secure business online, these companies have become experts in information security. The database systems behind these secure websites store non-critical data along with sensitive information, in a way that allows the information owners quick access while blocking break-in attempts from unauthorized users.

III. LITERATURE SURVEY

The X-Log Authentication Technique: is a strategy used to counter SQL injection attacks by logging and monitoring suspicious or potentially malicious activities related to authentication processes in an application or system that interacts with a database. If the data similarity 1s not the same with the model then it shows a potential SOLIA and stopped from executing on the database and reported (Indrani and Ramaraj, 2011).

SQL Rand: This approach utilizes a proxy server placed at the middle of the web server and database server for cracking the queries exchanged between the client and database server receives set of accepted keywords of the de-randomize SQL queries for computation. The error created by the database server which consist of illegitimate queries is hidden to avoid giving the hacker a chance of getting the architecture of the database tables and schema (Kumar and Pateriya, 2012) server.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 2, December 2023

IV. INPUT VALIDATION AND SANITIZATION

- Parameterized Queries/Prepared Statements: Utilize parameterized queries or prepared statements in your database queries. This helps separate SQL code from user inputs, preventing SQL injection attacks.
- Data Validation: Validate and sanitize all user inputs thoroughly. Restrict input lengths, format types, and use whitelistingtechniques to allow only expected inputs.
- *Escaping Special Characters:* Input Filtering: Escape or sanitize special characters (like quotes, semicolons) within userinputs before using them in SQL queries. This prevents these characters from being interpreted as SQL commands.
- *Database Permissions:* Apply the principle of least privilege. Limit database user permissions to only what is necessaryfor their function to mitigate potential damage in case of a successful attack.
- *Software Updates and Patching:* Keep your database management system, frameworks, and all associated software up- to-date with the latest security patches and fixes to address known vulnerabilities.
- *Audit and Log Reviews:* Monitor and review logs for suspicious activities. Implement monitoring systems that alert on unusual database access patterns or unauthorized attempts.
- *Web Application Firewalls (WAFs) and IDS/IPS:* Implement WAFs and IDS/IPS: Employ web application firewalls (WAFs) and intrusion detection/prevention systems (IDS/IPS) to actively monitor and filter incoming traffic, detecting and blocking potential SQL injection attempts.
- *Security Training:* Conduct regular training sessions for developers, emphasizing secure coding practices and awarenessof the risks associated with SQL injection attacks.
- *Regular Security Assessments:* Perform regular security assessments, including vulnerability scans and penetration tests,to identify weaknesses or vulnerabilities before attackers exploit them.
- *Secure Code Reviews:* Encourage and conduct code reviews focusing on security aspects to identify and rectify potentialvulnerabilities, including SQL injection vulnerabilities.

By implementing a comprehensive set of these preventive measures, organizations can significantly reduce the risk of data leaks caused by SQL injection attacks and bolster their overall security posture

V. INTRODUCTION OF TECHNOLOGIES USED

XAMPP is a popular open-source cross-platform web server solution package developed by Apache Friends. It includes various components such as Apache HTTP Server, MySQL (or MariaDB), PHP, and Perl.

- *MySQL or MariaDB*: XAMPP typically comes with MySQL or MariaDB as the default database server. MySQL and MariaDB are both relational database management systems that use SQL (Structured Query Language) to manage and manipulate data within databases.
- Developing Applications with SQL: XAMPP is often used for local development environments. Developers can create web applications using PHP and interact with the MySQL or MariaDB database by writing SQL queries within their PHPcode.
- Security Considerations: When using XAMPP or any local server environment, it's essential to consider security best practices. For instance, setting strong passwords for database users, restricting access to the database server, and avoidingexposing sensitive information in development environments.
- Configuration: XAMPP provides configuration files for each of its components, including MySQL. You can modify these configuration files to adjust settings according to your requirements.

VI. PHPMYADMIN

PhpMyAdmin is an open-source software tool introduced on September 9, 1998, which is written in PHP. Basically, it is a third-party tool to manage the tables and data inside the database. phpMyAdmin supports various type of operations on MariaDB and MySQL. The main purpose of phpMyAdmin is to handle the administration of MySQL over the web. It is the most popular application for MySQL database management. We can create, update, drop, alter, delete, import, and export MySQL database tables by using this software. phpMyAdmin also supports a wide range of operation like managing databases, relations, tables, columns, indexes, permissions, and users, etc., on MySQL and MariaDB. These

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 2, December 2023

operations can be performed via user interface, while we still have the ability to execute any SQL statement. phpMyAdmin is translated into 72 languages and also supports both RTL and LTR languages so that the wide range of people can easily use this software. We can run MySQL queries, repair, optimized, check tables, and also execute other database management commands. phpMyAdmin can also be used to perform administrative tasks such as database creation, query execution.

VII. APACHE TOMCAT

Apache Tomcat is an open-source web server and servlet container developed by the Apache Software Foundation. It's used to deploy Java-based web applications that use servlets and Java Server Pages (JSP). Tomcat implements the Java Servlet and Java Server Pages specifications, providing a platform for running Java web applications. The complete name of Tomcat is "Apache Tomcat" it was developed in an open, participatory environment and released in 1998 for the very first time. It began as the reference implementation for the very first Java-Server Pages and the java servlet API. However, it no longer works as the reference implementation for both of these technologies, but it is
considered as the first choice among the users even after that

VIII. TEST CASES

S. No	input	Login validation	Status
1	Authorized login credentials	Login success	success
2	Unauthorized login credentials	Login fail	failure
3	SQL injection commands (website without defensive code)	Login success	success
4	SQL injection commands (website with defensive code)	Login failed	failure

IX. MODULE DESCRIPTION

- *User Authentication Module:* Description: This module handles user login and authentication functionalities. Potential
- *Vulnerabilities:* SQL injection may occur if input fields for username or password are not properly sanitized.
- *Data Retrieval Module:* Description: Responsible for retrieving data from the database based on user input or queries. Vulnerability Point: Input validation flaws or dynamic query generation might lead to SQL injection vulnerabilities.
- *Search Functionality Module:* Provides search capabilities within the application. Vulnerability Assessment: Dynamicconstruction of search queries might be susceptible to SQL injection if not handled securely.
- *Forms and Input Handling Module*: Description: Manages various forms and user inputs within the application. Vulnerability Analysis: Inadequate input validation or sanitation of form data may expose vulnerabilities to SQL injectionattacks.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 2, December 2023

X. CONCLUSION

In our project, we have tried to discuss the modern SQL Injection attack which are less known to the general world as well as many researchers. The reasons and effects of SQLIA were discussed. The various types of SQLLA techniques were identified and described and also examples of the syntax used for execution were given. Furthermore, the most common detection and prevention techniques for SQLIA were summarized. Finally, an evaluation was done to check the effectiveness of the SQLIA prevention and detection techniques against the SQLIA types. However, the evaluation was based on prevention and detection only without considering the requirements for implementing the technique. The reasons and effects of SQLIA were discussed. The various types of SQLLA techniques were identified and described and also examples of the syntax used for execution were given. Furthermore, the most common detection and prevention techniques for SQLIA were summarized. Finally, an evaluation was done to check the effectiveness of the SQLIA prevention and detection techniques against the SQLIA types

XI. FUTURE ENHANCEMENT

SQL injection remains a prevalent and critical security issue in web applications that interact with databases. Efforts to combat SQL injection continuously evolve to address emerging threats and improve security. To parse the statement, the grammar of that parse statement's language is needed. In this method, by parsing two statements and comparing their parse trees, we can check if the two queries are equal. When attacker successfully injects SQL into a database query, the parse tree of the intended SQL query and the 10 resulting SQL query generated after attacker input do not match. The following figure shows the representation of a parse tree Here attacking code means any modification or changes done to the original query or it can say crafting of user input.

XII. PROPOSED SYSTEM

The solution proposed is a modification to the existing Piggybacked Queries for PHP web applications. In our solution, we modified the existing model by incorporating the logic to number of kinds of SQL injection attack and integrated a separate module to avert reflected cross-site scripting attacks. By preventing the SQL injection attack by adding an additional security mechanism called String Escape As the name suggests that hacker injects additional query with original one by which database gets multiple SQL queries. In this method original query is valid, but another query is attacking query with first one. This type of query is allowed in one query due to miss configuration of a system. Suppose an attacker injects abcd as userid SQL escape is used to escape the quotes so that any single quote characters in the input is considered as a meta-character and not a string delimiter. Replace method is intended to block attacks by preventing an attacker from ending the string and adding SQL injection code. All meta-characters will be removed in process and in username field only 6-15 characters are allowed. Ex: If an attacker passes admin) or 1=1"-- All special characters like(",=,#,!,--) will be deleted and it returns admin or 11 Which tends to the failure of attack with an error "Invalid login credentials"

REFERENCES

[1] Wei, K., Muthu Prasanna, M., & Suraj Kothari. (2006, April 18). Preventing SQL injection attacks in stored procedures. Software Engineering IEEE Conference. Retrieved November 2, 2007, from http://ieeexplore.ieee.org

[2] Thomas, Stephen, Williams, & Laurie. (2007, May 20). Using Automated Fix Generation to Secure SQL Statements. Software Engineering for Secure Systems IEEE CNF. Retrieved November 6, 2007, from http://ieeexplore.ieee.org

[3] Massachusetts Institute of Technology. Web Application Security MIT Security http://web.mit.edu/netsecurity/Camp/2003/clambert-slides.pdf

[4] Martin Bravenboer, Eelco Dolstra, Eelco Visser, Delft University of Technology The Netherlands. Preventing Injection Attacks with Syntax Embeddings A Host and Guest Language Independent Approach. Retrieved October 3, 2007, from http://portal.acm.org

[5] Yuji Kosuga, Kenji Kono, Miyuki Hanaoka Department of Information and Computer Science Keio University. Sania: Syntactic and Semantic Analysis for Automated Testing against SQL Injection. Retrieved November 12, 2007, from IEEE Computer Society. http://ieeexplore.ieee.org

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 2, December 2023

[6] Prithvi B, Madhusudan P, Venkatakrishnan VN (2010) CANDID: dynamic candidate evaluations for automatic prevention of SQL injection attacks. ACM Trans Inf System Security.
[7] Rahul J, Sharma P (2012) Survey on web application vulnerabilities (SQLIA,XSS) exploitation and security engine for SQL injection. In: Proceedings on CSNT 2012 IEEE international conference (978-0-7695-4692-6/1). IEEE, Washington, DC [8] Raju H, Cortesi A (2010) Obfuscation-based analysis of SQL injection attacks. In: ISCC '10 proceedings of the IEEE symposium on computers and communications, 931– 938. IEEE, Riccione