IJIRSET
International Journal of Innovative Research in
SCIENCE | ENGINEERING | TECHNOLOGY

e-ISSN: 2319-8753 | p-ISSN: 2347-6710

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

Volume 13, Issue 5, May 2024

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 8.423

9940 572 462 | 6381 907 438 | ijirset@gmail.com | www.ijirset.com

International Journal of Innovative Research in Science, Engineering and Technology (IJIRSET)

|e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Refereed Journal |

|| Volume 13, Issue 5, May 2024 ||

| DOI:10.15680/IJIRSET.2024.1305041 |

Detecting Data Leaks Using SQL Injection

Nazeefullah Shaik, Sriteja Chillarige, Sathish Reddy Janga, Dr. Raja Praveen

U.G. Student, Department of Computer Engineering, Jain University, Kanakpura Road, Karnataka, India

U.G. Student, Department of Computer Engineering, Jain University, Kanakpura Road, Karnataka, India

U.G. Student, Department of Computer Engineering, Jain University, Kanakpura Road, Karnataka, India

Associate Professor, Department of Computer Engineering, Jain University, Kanakpura Road, Karnataka, India

ABSTRACT: The SQL injection is a prevalent web application vulnerability that allows attackers to manipulate database queries by injecting malicious code into user inputs. This technique can lead to unauthorized access to sensitive data, modification of database content, or even the execution of arbitrary commands on the underlying system. To combat this threat, it is crucial to implement effective detection mechanisms that can identify and prevent SQL injection attacks before they can cause damage.This project delves into the intricacies of SQL injection and explores various techniques for detecting potential vulnerabilities. It discusses the significance of proper input validation, pattern matching, and anomaly detection in identifying and thwarting SQL injection attempts. Additionally, it highlights the role of web application security scanners and automated tools in streamlining the detection process. The project also emphasizes the importance of proactive security measures, such as secure coding practices, regular code reviews, and vulnerability assessments, in preventing SQL injection vulnerabilities from arising in the first place. By adopting a comprehensive approach that encompasses both detection and prevention strategies, organizations can significantly enhance their cybersecurity posture and safeguard sensitive data. By embracing this comprehensive approach, organizations can significantly enhance their capability to detect and respond to SQL injection attacks. This, in turn, reduces the risk of data breaches and data leaks, ultimately safeguarding critical data from unauthorized access and compromise. As the digital landscape continues to evolve, adopting such a multifaceted approach becomes increasingly imperative to secure the integrity of web applications and the data they manage.

KEYWORDS: CSP, AWS, EDoS, Dual Access Control, Authorized Keyword Search, Guarded Cloud, Homomorphic Encryption, DDoS

I. INTRODUCTION

SQL Injection Attack is a type of vulnerability that target database connected web applications where by malicious codes are inserted, processed and executed. This vulnerability exists when the web application does not perform a proper input validation. A poorly designed web application can be attacked by inserting malicious code in order to get access to the database. There are several places where users can input data in web applications, each of which can provide a SQL injection attack opportunity that can lead to loss of confidentiality, integrity and market value of an organization.

The broad implementation of security infrastructures such as intrusion prevention systems for protecting enterprise information systems has recently made remote unauthorized access of connected enterprise applications difficult. However, these safeguarding steps are being invaded and beaten without difficulty using simple scripting language. SQL injection is an example of this technique. In such a susceptible application, an SQL command injection attack uses crooked inputs that changes the SQL. query and establish illegal connection to the database. These types of attacks are intensely common and graded as one of the most common form of attack on web applications (Bisht et al., 2010). Many security procedures on the server side exist but they are not helpful in large scale because of the deployment complexity. And on the client side, installing of security applications deteriorates the client systems performance which thus decreases the web browsing satisfaction of the client (Suguria et al., 2014). Detection of SQL injection before it happen is difficult. Several cases of such illegal intrusion is carried out by the attacker using valid user credentials or by using built-in characteristics of database application sucio as virulent modification of current SQL queries of web application that are penetrating the delicate parts of the changed databases (Johari and Sharma, 2012).

The purpose of this study is to review the issue of SQL injection attacks detection and prevention. The review is based on four questions asked and answered.

|e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Refereed Journal |

|| Volume 13, Issue 5, May 2024 ||

| DOI:10.15680/IJIRSET.2024.1305041 |

II. RELATED WORK

The In the Detecting data leaks using SQL injection is a significant challenge in the realm of cybersecurity. SQL injection, a type of injection vulnerability, allows attackers to manipulate SQL statements used by web applications to communicate with databases. By exploiting these vulnerabilities, attackers can gain unauthorized access to sensitive data, modify or delete data, or even execute arbitrary code on the target system.

The primary problem in detecting data leaks using SQL injection stems from the difficulty in identifying and classifying potentially malicious input. Attackers often employ obfuscation techniques to mask their malicious intent, making it challenging for traditional detection methods to distinguish between legitimate user input and malicious code.SQL injection attacks present aserious threats to organizations. A variety of research has been undertaken to address this threat, presenting various artificial intelligence techniques for detection of SQL injection attacks using machine learning and deep learning models. Techniques to facilitate the detection of threats are usually implemented via learning from historical data representing an attack and/or normal data.

Historical data are useful for learning, in order to recognize patterns of attacks, understanding detected traffic, and even predicting future attacks before they occur.This project is to prevent SQL injection while firing queries to database and to make the database secured. This system is online so no need of implementation. It can be accessed through internet from anywhere. The system uses SQL Injection mechanism to keep the data safe and secured.

III. METHODOLOGY

This proposed work focuses on developing a novel anomaly-based technique for detecting SQL injection attacks. The proposed technique will utilize a combination of machine learning and statistical methods to identify unusual patterns in user input. The technique will be evaluated using a dataset of real-world web application traffic. Methodology The proposed technique will consist of the following steps: Data Preprocessing: The data will be preprocessed to remove irrelevant information and to ensure that the data is in a format that can be processed by the machine learning algorithms. Feature Extraction: A set of features will be extracted from the preprocessed data. These features will be used to represent the characteristics of the user input. Model Training: A machine learning model will be trained on the extracted features. The model will be trained to identify patterns in the features that are indicative of an SQL injection attack.Model Evaluation: The trained model will be evaluated on a separate dataset of real-world web application traffic. The evaluation will assess the accuracy of the model in detecting SQL injection attacks.

Development Tools and Tech Stacks:

Web Application Firewalls (WAFs): WAFs act as a barrier between web applications and external traffic, filtering and blocking malicious SQL injection attempts. Popular WAFs include ModSecurity, Cloudflare WAF, and Imperva WAF. Static Application Security Testing (SAST): SAST tools analyze application code to identify potential SQL injection vulnerabilities before deployment. Tools like SonarQube, Veracode, and Fortify can effectively scan source code for injection flaws. Dynamic Application Security Testing (DAST): DAST tools perform real-time testing of web applications by actively scanning and probing for vulnerabilities, including SQL injection. Burp Suite, OWASP ZAP, and WebGoat are widely used DAST tools. Database Activity Monitoring (DAM): DAM tools monitor database activity to detect anomalies and suspicious patterns that may indicate SQL injection attacks. Tools like Oracle Enterprise Manager, Splunk Enterprise, and Datadog can provide valuable insights into database activity. Penetration Testing: Penetration testing involves simulating real-world attacks to identify and exploit vulnerabilities, including SQL injection. Experienced penetration testers can uncover hidden vulnerabilities and provide recommendations for remediation.Tech Stacks:Parameterized Queries: Prepared statements allow developers to separate data from SQL queries, preventing attackers from injecting malicious code directly into database queries.Input Validation and Sanitization: Validating and sanitizing user inputs before they reach the database can effectively block malicious SQL injections. Techniques like whitelisting, input escaping, and type conversion can be employed.Object-Relational Mapping (ORM) Frameworks: ORMs like Hibernate, JPA, and Django provide an abstraction layer between applications and databases, reducing the risk of SQL injection by encapsulating SQL queries within the ORM framework.Database Access Objects (DAOs): DAOs encapsulate database interactions, providing a centralized and secure way to access and manipulate database data, reducing the likelihood of SQL injection vulnerabilities. Secure Coding Practices: Following secure coding practices, such as using secure coding guidelines and avoiding common SQL injection patterns, can significantly reduce the risk of SQL injection vulnerabilities. By utilizing a combination of these development tools and tech stacks, organizations can effectively detect and prevent data leaks caused by SQL injection, safeguarding sensitive information and maintaining the integrity of their web applications.

|e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Refereed Journal |

|| Volume 13, Issue 5, May 2024 ||

| DOI:10.15680/IJIRSET.2024.1305041 |

IV. EXPERIMENTAL RESULTS

Architecture :

Detecting data leaks using SQL injection involves identifying and exploiting vulnerabilities in web applications that allow attackers to inject malicious SQL code into database queries. This can lead to the disclosure of sensitive information, such as user credentials, credit card numbers, and other confidential data. There are several methods for detecting SQL injection vulnerabilities, including: Web Application Vulnerability Scanning (WVS): This involves using automated tools to scan web applications for known SQL injection vulnerabilities. WVS tools can identify vulnerabilities based on patterns in the application's code or by attempting to inject malicious code themselves. Fuzz Testing: This involves bombarding an application with a large amount of random data in an attempt to trigger errors or unexpected behavior. Fuzz testing can be effective in identifying vulnerabilities that are not detected by WVS tools. Manual Penetration Testing: This involves a skilled security professional manually testing an application for vulnerabilities. Penetration testers can use a variety of techniques to identify vulnerabilities, including SQL injection. Code Review: This involves manually reviewing the source code of an application to identify vulnerabilities. Code review can be effective in identifying vulnerabilities that are not detected by other methods. Once a SQL injection vulnerability has been identified, it is important to take steps to remediate it. This may involve: Input Sanitization: This involves scrubbing user input to remove any potentially malicious code. Input sanitization can be done using a variety of techniques, such as regular expressions, URL encoding, and whitelisting. Prepared Statements: This involves using prepared statements to build SQL queries. Prepared statements help to protect against SQL injection by separating the user input from the SQL code.

Web Application Firewalls (WAFs): WAFs can be used to filter out malicious traffic before it reaches the web application. WAFs can be effective in blocking SQL injection attacks, but they are not foolproof. Security Awareness Training: It is important to educate developers and users about SQL injection vulnerabilities so that they can take steps to prevent them. In addition to these technical measures, it is also important to have a process in place for responding to data leaks. This process should include steps for notifying affected individuals, investigating the cause of the leak, and taking steps to prevent future leaks. Here is an example of how SQL injection can be used to detect data leaks:An attacker might try to inject a malicious SQL statement into a web application's login page. If the application is vulnerable to SQL injection, the attacker could use this statement to retrieve the usernames and passwords of all users who have logged in to the application. To prevent this type of attack, the web application should be designed to sanitize user input and use prepared statements. Additionally, a WAF can be used to filter out malicious traffic before it reaches the application. By following the recommendations in this guide, you can help to protect your web applications from SQL injection attacks and data leaks.

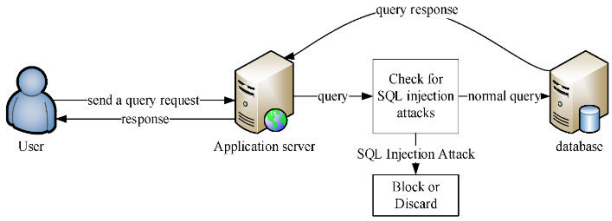

Fig 5a. architecture.

V. CONCLUSION

SQL injection is a common and serious security vulnerability that can allow attackers to gain unauthorized access to sensitive data, modify or delete data, or even execute arbitrary commands on the underlying database server. Detecting and preventing SQL injection attacks is therefore critical for protecting the confidentiality, integrity, and availability of data.

There are a number of different techniques that can be used to detect SQL injection attacks. These include:Signature-based detection: This technique involves looking for known SQL injection patterns in user input. Heuristic-based detection: This technique uses heuristics to identify suspicious user input. Runtime application self-protection (RASP): This technique uses code instrumentation to monitor application behavior and detect SQL injection attacks at runtime. Training-based detection: This technique uses machine learning algorithms to identify SQL injection attacks based on patterns in training data. In addition to these technical methods, there are also a number of organizational practices that can help to prevent SQL injection attacks. These include: Input validation: This involves validating user input to ensure that it is in the expected format and does not contain any malicious code. Output encoding: This involves encoding output from the database to prevent it from being interpreted as SQL code. Using

|e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Refereed Journal |

|| Volume 13, Issue 5, May 2024 ||

| DOI:10.15680/IJIRSET.2024.1305041 |

prepared statements: This involves using prepared statements to avoid constructing SQL queries directly from user input. Keeping software up to date: This involves keeping software up to date with the latest security patches.

REFERENCES

[1]"Detecting Data Leaks Via SQL Injection Prevention on an E-Commerce" (2023) by Nevon Projects: This paper proposes a system for detecting and preventing SQL injection attacks on an e-commerce website

[2]"Analysis of SQL Injection Attack Detection and Prevention on MySQL Database Using Input Categorization and Input Verifier" (2022) by IEEE Xplore: This paper presents a novel approach to detecting and preventing SQL injection attacks on MySQL databases

[3]"Detecting Data Leaks via SQL Injection Prevention" (2022) by SSRN Papers: This paper discusses the use of SQL injection prevention mechanisms to detect and prevent data leaks.

[4]"jayantrane/Secure-Systems-of-SQL-Injection-Attack" (2022) by GitHub: This GitHub repository contains a collection of tools and resources for detecting and preventing SQL injection attacks.

[5]"Detecting SQL Injection Attacks" (2021) by OWASP: This article from the Open Web Application Security Project (OWASP) provides a comprehensive overview of SQL injection attacks.

[6]OWASP SQL Injection Prevention Cheat Sheet: This comprehensive guide from the Open Web Application Security Project (OWASP) outlines various strategies for preventing SQL injection attacks, including input validation, prepared statements, and parameterized queries.
https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html

[7]Detecting SQL Injection Attacks with WAFs: This blog post by Imperva explains how web application firewalls (WAFs) can effectively detect and block SQL injection attempts, providing real-time protection against these attacks. https://docs.imperva.com/bundle/on-premises-knowledgebase-reference-guide/page/sql_injection.htm

[8]Testing for SQL Injection Vulnerabilities: This article by Veracode discusses various methods for testing web applications for SQL injection vulnerabilities, including manual testing, automated tools, and penetration testing
https://community.veracode.com/s/topic/0TO2T000000bzr3WAA/sql-injection

9]SQL Injection Prevention in Web Applications: This comprehensive white paper by Deloitte provides a detailed overview of SQL injection attacks, prevention techniques, and best practices for securing web applications.
https://www.datadoghq.com/product/application-security-management/

[10]SQL Injection Detection and Prevention Tools: This resource from Cigital lists various commercial and open-source tools that can be used to detect and prevent SQL injection attacks, providing a comparison of their features and capabilities. https://research.ijcaonline.org/etcsit/number5/etcsit1036.pdf

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi cross ref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

9940 572 462 6381 907 438 ijirset@gmail.com

www.ijirset.com

Scan to save the contact details