Industrial Engineering Journal
ISSN: 0970-2555
Volume : 52, Issue 7, No. 5, July : 2023

BUILDING AND DETECTING PYTHON SQL INJECTION VULNERABILITIES.

Ms. Monisha M Y, Student, Dept. Of Information Science, National Institute of Engineering, Mysuru.
Sri. N Rajesh, Assistant Professor, Dept. Of Information Science, National Institute of Engineering, Mysuru.

Abstract

SQL injection attack is considered as one of the most dangerous threat in the world of web application security. This attack occurs when an attacker/hacker succeeds to insert the malicious code in any particular web application and these injected codes successfully interacts with database query. The injection of such type of infected queries can be done by attackers through URLs or from web forms. The impact of SQL injection attacks can be very severe if target web application contains very sensitive information in its database such as credit/debit card details, national security policy details or some other information in terms of confidentiality. In this research paper, a secure coding approach is proposed that can be used by web developers and security professionals to secure their application against such type of attacks at the time of coding. To check the accuracy and efficiency of proposed approach, several real time PHP based web applications have been tested and a comparison analysis is done among previous prevention techniques and proposed technique.

Keywords: Cybersecurity, SQL Injection, Attack, Hacking.

I. Introduction:

Generally, people say that SQL injection is not new for them and they know about this terminology but fact is totally different from this and they have only heard about it or experienced only through trivial examples. SQL injection is one of the most devastating vulnerabilities that's impact on business can be very severe as it can expose all sensitive information which are stored in an application's database it may also include handy information such as usernames, passwords, names, addresses, phone numbers, credit & debit card details. Hence, SQL injection is very severe vulnerability that results when application developer gives ability to hacker/attacker that they can influence Structured Query Language (SQL) queries which an application passes to a back-end database. Basically it is the hacking technique which attempts to pass SQL statements through a web application for execution by the backend database. If input data is not sanitized properly, web applications may be victim SQL Injection attacks that allow hackers to view and extract sensitive information from the database and/or even wipe it out. Database applications typically call for user input for data to add to the database. A SQL injection attack occurs when user modifies a SQL statement through a user input field. Therefore, SQL injection attacks can occur when SQL statements are dynamically created using user input. The threat occurs when users enter malicious code that 'tricks' the database into executing unintended commands. If we have to stop SQL injection attack then we have to use input validation for every necessary field. Input validations can be used to prevent SQL injection attack.

A web application, based on the above model, takes text as input from users to retrieve information from a database. Some web applications assume that the input is legitimate and use it to build SQL queries to access a database. Since these web applications do not validate user queries before submitting them to retrieve data, they become more susceptible to SQL injection attacks. For example, attackers, posing as normal users, use maliciously crafted input text containing SQL instructions to produce SQL queries on the 7 web application end. Once processed by the web application, the accepted malicious query may break the security policies of the underlying database architecture

because the result of the query might cause the database parser to malfunction and release sensitive information.

The goal of this project is to build an automated fix generation method to prevent SQL injection vulnerability from plain text SQL statements. In an automated method approach, a server will gather information about previously known vulnerabilities, specifically SQL statements, generate a patch, and apply patch. The process can be completed by someone with no security expertise and secure legacy code, which will allow developers to fix the SQL injection vulnerability.

II. Literature:

"**Cyber Attack Detection and Implementation of Prevention Methods For Web Application**" by Aishwarya Bhalme; Akash Pawar; Aditi Borkar; Pranav Shriram

The internet and web applications are the only things that run the modern world. Today, the biggest concern facing businesses is web security. It is seen as serving as the fundamental framework for the global data society. Security breaches can happen to web

applications. Web security is merely protecting a layer of a web application from attacks by attackers or unauthorized users. A large number of problems with web based applications are mostly the result of incorrect client input. The various facets of web

security are covered in this paper, along with its flaws. This paper also discusses the key components of web security strategies, including encryption, authentication, passwords, and integrity. Additionally described in detail are the attack methods and the anatomy of a web based application attack. This paper explores a number of methods for detection and prevention of vulnerabilities in the web application. This research suggests a more effective method for reducing this category of web vulnerabilities. Additionally, it offers the finest defence against the a for mentioned threats.

"**A Detailed Evaluation of SQL Injection Attacks, Detection and Prevention Techniques**" by Nikita Joshi;Tejas Sheth;Varshil Shah;Jaya Gupta;Sofiya Mujawar

An SQL Injection attack is a database focused attack for programmes that utilise data. It is accomplished by inserting malicious lines of code into the SQL queryto alter and modify its meaning, allowing the attacker to gain access to the database or retrieve sensitive

data. Many strategies for detecting and preventing such assaults have been developed and suggested. This study provides an in depth examination of 38 publications on approaches for detecting SQL Injection in web applications. This offers a foundation for designing and using efficient SQL Injection, detection and prevention techniques.

"**Analysis of SQL Injection Attack Detection and Prevention on MySQL Database Using Input Categorization and Input Verifier**" by Alya Aiman Salsabila Arif; Rahmat Purwoko; Nurul Qomariasih; Hermawan Setiawan

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.

"**Code Injection Assault & Mitigation Model to Prevent Attacks**" by Shalom Akhai; Vincent Balu

The majority of businesses now want to conduct their operations online, and web applications are one of the most popular targets for web application assaults, which are quickly emerging as the biggest security risk facing modern businesses. Denial of services (DOS), malware, and brute force assaults, for instance, are the most frequent cyberattacks on web applications nowadays. Basically, Exploit is a flaw on the web that enables attackers to manipulate the queries that a website relies on. An attacker may actually read, remove, add, and retrieve the stored data with the use of these queries.

"**Clarity: Analysing Security in Web Applications**" by Connor J. Potter;Neetesh Saxena;Soumyadev Maity

The rapid rise in business' moving online has resulted in e-commerce web applications becoming increasingly targeted by hackers. This paper proposes Clarity, a dynamic black box vulnerability scanner capable of detecting Cross-Site Scripting, SQL Injection, HTTP Response Splitting, and Session Management vulnerabilities in web applications. The developed tool employs the use of Mechanize and Selenium to perform the majority of its web scraping requirements. Clarity was tested against 50 e-commerce web applications, uncovering Session Management flaws as the most prevalent vulnerability, with 36 out of the 50 applications being vulnerable. "Protecting User Credentials against SQL Injection through Cryptography and Image Steganography" by Parmit Singh Banga; A. Omar Portillo-Dominguez; Vanessa AyalaRivera [3] As today's world is all about the internet and technology, the whole context of businesses has been shifted digitally. Almost every sector has established itself on a digital

III. Conclusion

In this project, A project focused on SQL injection should aim to address and prevent SQL injection vulnerabilities in a web application that uses a database management system (DBMS) that supports SQL queries.

The project should start with a thorough analysis of the application and its database to identify potential vulnerabilities. This analysis should include an assessment of user inputs and how they are

used in SQL queries, as well as an examination of the application's code to identify any areas that may be susceptible to SQL injection.

Once potential vulnerabilities are identified, the project should focus on implementing appropriate measures to prevent SQL injection attacks. This may include implementing parameterized queries, which can prevent SQL injection by ensuring that user-supplied data is not directly included in SQL statements. Input validation and sanitization techniques can also be used to identify and remove any malicious input data.

The project should also include ongoing monitoring and testing to ensure that the application remains secure against SQL injection attacks. This may involve regular security testing to identify and address any new vulnerabilities that may arise, as well as implementing measures to ensure that any known vulnerabilities are promptly addressed.

In conclusion, a SQL injection project should aim to identify, prevent, and mitigate SQL injection vulnerabilities in a web application. By implementing appropriate measures and conducting ongoing testing and monitoring, it is possible to ensure the security and integrity of the application and its associated database

References:

[1] Wei, K., Muthuprasanna, 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] Merlo, Ettore, Letarte, Dominic, Antoniol & Giuliano. (2007 March 21). Automated Protection of PHP Applications Against SQL-injection Attacks. Software Maintenance and Reengineering, 11th European Conference IEEE CNF. Retrieved November 9, 2007, from http://ieeexplore.ieee.org

[4] Wassermann Gary, Zhendong Su. (2007, June). Sound and precise analysis of web applications for injection vulnerabilities. ACM SIGPLAN conference on Programming language design and implementation PLDI, 42 (6). Retrieved November 7, 2007, from http://portal.acm.org

[5] Friedl's Steve Unixwiz.net Tech Tips. (2007). SQL Injection Attacks by Example. Retrieved November 1, 2007, from http://www.unixwiz.net/techtips/sql-injection.html

[6] Massachusetts Institute of Technology. Web Application S ecurity MIT Security Camp. Retrieved November 1,