International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 7, Issue 2, pp 101-106, February-2023

https://doi.org/10.47001/IRJIET/2023.702016

SQL Injection Attacks and Defense Mechanisms

[1]Omkar Kamleshwar Kambre, [2]Kabir Kiran Shah, [3]Pankaj Dulabhai Rathod

[1,2]Department of Information Technology, SVKM's Shri Bhagubhai Mafatlal Polytechnic, Mumbai, India
[3]Senior Lecturer, Dept. of Information Technology, SVKM's Shri Bhagubhai Mafatlal Polytechnic, Mumbai, India

***Abstract* - Protection of data is essential in this rapidly evolving technological world. As many of the users rely on the database management system for protection and storage of the sensitive/personal data. Hackers use various methods to invade the user's privacy such as SQL Injection Attack, Denial of Service, Weak Authentication, etc. There are several measures to counter these attacks such as Vulnerability Scanners, Input Validation, Firewall and Intrusion Detection Systems, Two Factor Authentication and Password Complexity, etc. This paper studies the types of Database Attacks, their counter measures and proposes an approach to defend against these attacks.**

Keywords: SQL Injection Attack, Denial of Service, Weak Authentication, Vulnerability Scanners, Input Validation, Firewall and Intrusion Detection Systems, Two Factor Authentication and Password Complexity.

I. INTRODUCTION

SQL injection is the vulnerability that results when you give an attacker the ability to influence the Structured Query Language (SQL) queries that an application passes to a backend database. The attacker can leverage the syntax and the capabilities of SQL itself by influencing what is passed to the database. SQL injection is not a vulnerability that exclusively affects Web applications; any code that accepts input from an untrusted source and then uses that input to form dynamic SQL statements could be vulnerable. One thing that Web applications have in common, regardless of the language in which they were written, is that they are interactive and, more often than not, are database-driven.

Access to databases is via website or APIs using a language called SQL (structured query language). Many of the websites are vulnerable to hacker attacks that may further launch backdoor attacks as a result of bad coding practices that do not comply with coding standards [1].

The main purpose of this paper will be to create an overview of the types of SQL Attacks and its key defense mechanisms to counter those attacks and protect the user's information from injecting SQL attacks.

II. LITERATURE REVIEW

1. Overview of SQL Defense Mechanisms.

The central topic of research in this work is to analyze SQL injection attack methods and to outline the best defense mechanisms to detect and prevent SQL injection attacks. This research shows practical simulation of SQL Injection attack using Kali Linux platform which is used for detecting threats from cyber-attacks.

2. Input-Based Analysis Approach to prevent SQL injection Attacks.

This paper focuses on approach based on input based analysis to detect and prevent SQL Injection Attack. This technique has basically two parts (i) input categorization and (ii) input verifier. This paper provides a detailed description regarding literature on safety and time cost aspects.

3. Research on SQL Injection vulnerabilities and its Detection Methods.

This research paper studies attack principles, detection techniques and ways to prevent those. It proposes an approach and also a tool name SQLliscan.

4. SQL Injection Teaching Based on SQLi-labs.

This paper describes about the SQLi-labs. SQLilabs is teaching assistant software. This helps the tutor to carry a SQL injection attacks which students have to counter it. It shows the practical implementation of the attack example demonstration. To prevent this attacks it also has its defense reinforcement and verification.

5. SQL Injection Countermeasures Methods.

This paper explains SQL Injection and countermeasures and the impacts of SQL Injection attacks. This paper defines different types of SQL Injection attacks like tautologies, piggyback queries, alternate encoding, etc. It researches on countermeasures such as escape functions, custom error messages, etc. to prevent SQL Injection.

6. Research on SQL Injection Attack and Prevention Technology.

ISSN (online): 2581-3048
Volume 7, Issue 2, pp 101-106, February-2023
https://doi.org/10.47001/IRJIET/2023.702016

The characteristics of the SQL injection attack are illustrated in this paper, along with certain safeguards. Input validation and type-safe SQL can stop the SQL injection attack. As a result, a security model is developed to stop SQL injection attacks. This model detects attacks by contrasting the input size of intended queries with the actual ones. The method is applicable for safeguarding online applications.

7. Survey on SQL Injection Attack: Detection and Challenges.

The researchers have provided a thorough analysis in this paper and the overview of the relevant literature on the various SQLi vulnerabilities. They outlined many approaches that may be employed for identifying these SQL assaults. Furthermore, they listed the advantages and drawbacks of each technique utilized for recognizing SQLi attacks. This research could benefit the readers who wished to learn more about the subject or researchers who want to identify all the problems that are still affect the web applications' ability to detect SQLi attacks.

8. Evaluation of SQL Injection and Detection and Prevention measures.

This study first outlined the different categories of SQLIAs. Then it describes about SQL injection detection and avoidance methods. Then it contrasted these methods based on how well they prevented SQLIA. The findings suggest that some present strategies should be upgraded in order to better defend against SQLI attacks. It also contrasted these method’s deployment requirements, which cause users to be inconvenienced.

9. Comparison of SQL Injection Detection and Prevention Techniques.

This research identified the various forms of SQLIAs in this research. Then it looked into SQL injection detection and avoidance methods. Then it describes about the methods based on how well they prevented SQLIA. The findings suggest that the effectiveness of several current strategies for preventing SQL injection attacks has to be enhanced.

10. Network security education: SQL injection attacks.

This paper describes a tool that shows different kinds of SQL injection attacks. The users provided with some extremely helpful practical feedback, such as suggestions to present various defense strategies or to incorporate attacks using single quotations and back quotes (currently, attacks using double quotes are demonstrated). The most popular techniques for SQL injection attacks against vulnerable PHP applications are also covered in the article. In the majority of cases, attack tactics are combined, which strengthens the attack.

11. SQL Injection: Classification and Prevention.

This paper describes what is meant by SQL Injection, what are the impacts of SQL injection and various types of way to detect those attacks. Various types of SQL Injection attacks are listed along with their description. Counter Measures to avoid this attack are described in this paper.

12. Attack Intention Estimation Based on Syntax Analysis and Dynamic Analysis for SQL Injection.

In this research, we presented a technique to recognize the purpose behind SQL injections. The suggested solution uses syntactic and dynamic analysis to separate the four categories of intention from the behavior of attack SQL queries' partial execution. The suggested approach can determine the intention without needing the whole SQL query conducted on the target DB or other information because it only requires a portion of a SQL query contained in an HTTP request as input. According to evaluation results, the aim could be correctly identified with 83.1% accuracy for a synthetic dataset and 71.9% accuracy for a real-world dataset.

13. Impact of SQL Injection in Database Security.

This paper presents a suggested strategy and its applications for safeguarding databases against unauthorized attacks. SQL injection attacks are undoubtedly one of the most destructive and serious security issues or assaults. Every other kind of web based program is extremely vulnerable to SQL injection attacks. This paper describes about ways to overcome these challenges by using strategies. Consequently, databases are protected from SQL injection attacks.

III. TYPES OF SQL INJECTION ATTACKS

SQL injection, also known as SQLI, is a common attack vector that uses malicious SQL code for backend database manipulation to access information that was not intended to be displayed. This information may include any number of items, including sensitive company data, user lists or private customer details.

The types of SQL Injection attacks include following attacks:

1. Error Based SQL Injection Attacks.

It is an injection technique that allows the invader to attack the database using database’s erroneous output. Creation of Database contains some errors that hint the hacker the structure of the database. In this type of attacks, the attacker uses same communication channel for both attack and data retrieval. The error generated by the database is enough for the invader to understand the structure of the database entirely.

Error Based SQL Injection technique forces the database to generate an error giving the attacker information upon which they refine their injection.

2. Out of band SQL Injection Attacks.

Out of band SQL injection is a specific type of SQL injection. The term out of band means that the attacker does not receive a response from the attacked application on the same communication channel but instead is able to cause the application to send data to a remote endpoint that they control.

Out of band SQL injection is only possible if the server that you are using has that trigger DNS or HTTP requests. However, that is the case with all popular SQL servers.

3. Time Based Blind SQL Injection Attacks.

Time Based Blind SQL Injection is a technique that relies on sending an SQL query to the database which forces the database to wait for a specific period of time. Attacker will know from the response time whether the result of the query is true or false [1].

4. Boolean-Based SQL Injection Attacks.

In this type of attacks, a Boolean query causes the application to give a different response for a valid or invalid result in database. It works by enumerating the characters from the text that needs to be extracted (ex. Database name, name, column name, etc.) one by one [1].

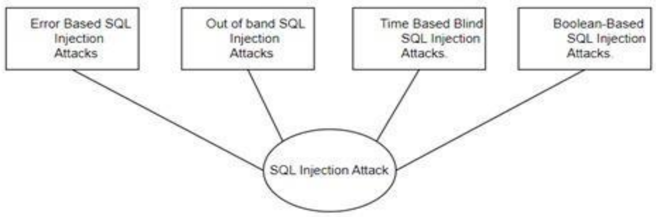

Error Based SQL Injection Attacks
Out of band SQL Injection Attacks
Time Based Blind SQL Injection Attacks
Boolean-Based SQL Injection Attacks
SQL Injection Attack

Figure 1: Types SQL Injection Attack

IV. TYPES OF SQL DEFENCE MECHANISMS

We cannot completely prevent SQL Injection attacks; however, there are some defense mechanisms against SQL Injection attempts.

Below are some defense mechanisms:

1. Filtering

Every input of the user should be filtered before inserting it into the database so that it won’t cause any type of error or maliciousness. For example: email address should be filtered so that only accepted characters should be allowed, phone numbers should be filtered only digits of a phone number should be accepted. HTTPS Usage HTTPS should be used by every website so which will enable them to have encrypted date while sending over the internet. It is “very important” for web applications to be defended against SQL Injection attacks.

2. Encryption

Database Encryption is a process to convert data in the database to cipher text (unreadable text) using an algorithm. A key generated from the algorithm is essential to decrypt the text. Following are the benefits of having encryption in database:

i) It avoids security attacks:

Security attacks are inevitable but with better data encryption intruders might not analyze or decrypt to understand the data in a data breach.

ii) Protecting Sensitive Data:

Encryption key management is must for protecting sensitive data with centralized key management such as Hashicorp Vault (open source) or Public Cloud (proprietary).

3. Monitor SQL Statements:

Monitoring SQL statements helps in identifying rogue SQL statements or vulnerabilities. Monitoring tools using machine learning are especially useful. You can generate information (statistics) such as SQL statements, Start and End Timestamp, Estimated Processing Time, Total Rows in Table Queried and Estimated Number of Joined rows, etc.

You can use these performance statistics to generate various reports. You can include reports that shows queries that:

i. Take a long time to execute.
ii. Create a temporary index during execution.
iii. Did not run because of query governor time limit.

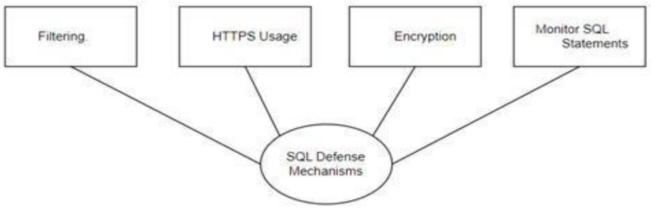


Figure 2: Types SQL Defense Mechanisms

V. DETECTION METHODS

The first step towards preventing an SQL Injection Attack is detecting vulnerabilities. In order to prevent SQL

ISSN (online): 2581-3048
Volume 7, Issue 2, pp 101-106, February-2023
https://doi.org/10.47001/IRJIET/2023.702016

Injection attacks there are some detection methods such as: 1) Static Detection 2) Dynamic Detection 3) Hybrid Detection.

1) Static Detection

Static detection analyzes the web application's SQL queries to detect and prevent SQL Injection attacks, the focus of the static analysis method is validating the user input type to reduce the chances of SQL Injection Methods rather than detecting them. DEKANT, MACKE is the tools used for Static Analysis.

2) Dynamic Detection

The main focus of Dynamic Detection is on verifying the security of program during operation, it the generates test cases for submission collects and analyzes the returned information to determine whether there are vulnerabilities. SecuBat, TaintScope are the tools used for dynamic detection.

3) Hybrid Detection

It can be defined as the combination of both Static and Dynamic Detection; also it inherits the advantages of both Static and Dynamic Detection. High degree of automation and strong applicability are supported by Hybrid detection method. It is specifically designed to detect and prevent SQL Injection attacks. Taint analysis and pattern matching methods are used by GuruWS is a scalable hybrid platform.

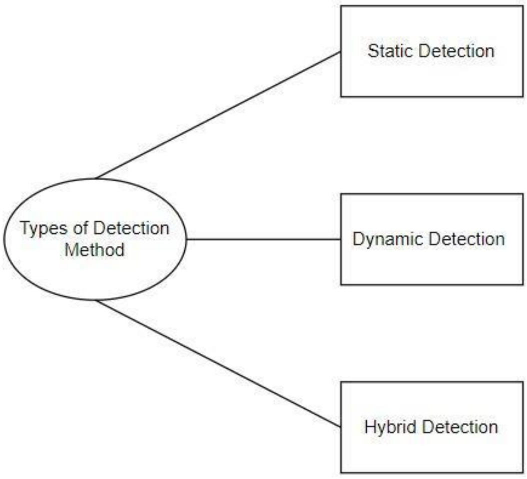


Figure 3: Types of Detection Method

VI. PREVENTION MEASURES

1) Hierarchical Management of User Permissions:

When a database is being designed, the users should be assigned permissions based on their requirements corresponding to their work. In this way even if an SQL Injection attack is initiated on the database, the harm to the database is minimized.

2) SQL statement parameterization:

When a SQL statement, the developers should not write variable values into the statement directly. Instead, they should apply corresponding parameters to pass variables. Such an action reduces the chance of attackers to inject malicious statements.

3) Usage of database security parameters:

There are several security parameters in SQL Server. They should be appropriately being selected for the Server. For example, parameters collection that can automatically verify the length and type of data can be added to the program. After adding this collection, the Web Application will enforcement checks, if the checks exceed a certain range, an exception will be reported to the users. The Web Application will also filter out execution code, which contains malicious statements.

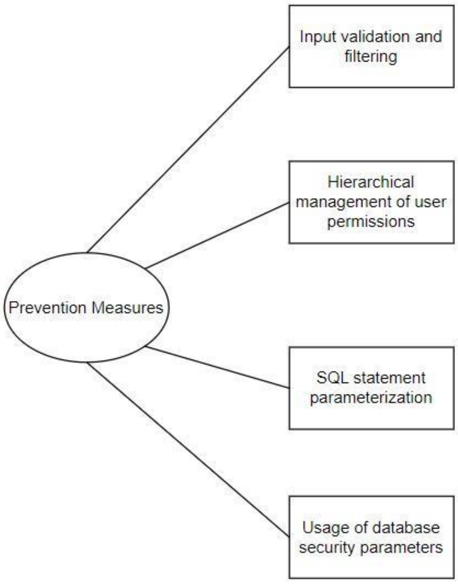

Prevention Measures
Input validation and filtering
Hierarchical management of user permissions
SQL statement parameterization
Usage of database security parameters

Figure 4: Types Prevention Measures

VII. IMPLEMENTATION AND RESULTS

Example of SQL Injection

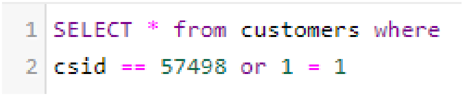
Suppose we have an application based on customer records, any customers can view only his or her own records by entering a unique and private customer ID. Suppose we have a field link below:

Customer ID:

Customer enters the following in the input field:

57498 OR 1=1.

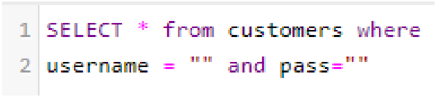
Basically, it translates to:



```
SELECT * from customers where
csid == 57498 or 1 = 1
```

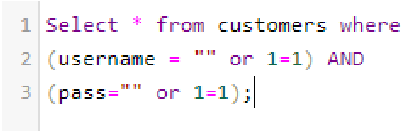
Now this 1=1 will return all records for which this holds true. So basically, all the customer's data is compromised. Now the malicious user can also delete the customer's records in a similar fashion.

Consider the below SQL query:



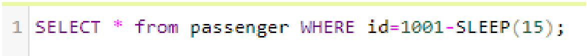
```
SELECT * from customers where
username = "" and pass=""
```

Now the malicious can use the '=' operator in a clever manner to retrieve private and secure user information. So instead of the above-mentioned query the following query when executed, retrieves protected data, not intended to be shown to users.



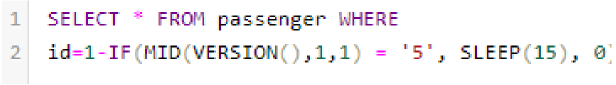
```
Select * from customers where
(username = "" or 1=1) AND
(pass="" or 1=1);
```

Since 1=1 always holds true, user data is compromised. Example of Time-Based Blind SQL Injection Attacker will enter the query in the following manner:



```
SELECT * from passenger WHERE id=1001-SLEEP(15);
```

It is evidence that SQL injection is feasible when the server is using MySQL as a database when an attacker tries to use these functions in the query and is successful in delaying the response. As a result, the attacker might insert a payload that is more intricate.



```
SELECT * FROM passenger WHERE
id=1-IF(MID(VERSION(),1,1) = '5', SLEEP(15), 0
```

Similar to this, SQL Server's WAIT FOR DELAY and WAIT FOR TIME functions let you to pause a query's execution for the length of time you specify before continuing it when the system time reaches the threshold.

VIII. ALGORITHM

The IntCat algorithm takes a user input and decomposes it based on two natures of input i.e. safe/ may unsafe. In this algorithm the input is categorized into Keywords, Special Characters, Alphabets, Numbers as follows:

Keywords:
OR, SELECT, DELETE, WHERE

Special Characters:
/, @, #, $, %, =, ''

Alphabets:
A-Z, a-z

Numbers:
0-9

```
tinyint f=0, c=0;
String str = readInput(); // Accept user input
tinyint l= strlen(str); // Compute length of input
for tinyint i =0 to l-1 do
    if str[i] 2 [A - Z, a - z] then
        f = f+1;
    if str[i] 2 [0 - 9] then
        c = c+1;
    if substr(str) 2 {Keywords} then
        ... do nothing ...
    else
        ... do nothing ...
    if f=l || c=l || (f+c) =l then
        Safe;
    Else
        May unsafe;
    End
    [2]
```

IX. DISCUSSION AND CONCLUSION

In today's world data is of utmost importance, and this data is stored in a database. But this type of storage is not completely secure, intruders can hack into the database using SQL Injection attacks, there are some defenses against these types of attacks. This paper allowed us to present some of the best-known SQL Injection Attacks and their Defense Mechanisms. These SQLi Attacks can make it possible to the intruder to access to the data, modify or even delete it. There are some simple protections that can safeguard our database from being the victim of such harmful attacks.

We have analyzed an algorithm for Input Categorization. We also implemented an example of SQL injection attack by taking customer table as an example. The second thing which we implemented was Time-Based Blind SQL Injection by taking an example of the passenger database; it shows how the intruder will enter the query and use the SLEEP function in the query to delay the response.

REFERENCES

[1] Igor Tasevski and Kire Jakimoski, "Overview of SQl Defense Mechanisms," Serbia, Belgrade, November 24-25, 2020.

[2] Angshuman Jana, Priyam Bordoloi and Dipendu Maity, "Input-Based Analysis Approach to prevent SQL injection Attacks", 5-7 June 2020, Dhaka, Bangladesh.

[3] Tao Zhang and Xi Guo, "Research on SQL Injection vulnerabilities and it's Detection Methods", 2020 4th Annual Conference on Data Science and Business Analytics (ICDSBA).

[4] Chen Ping, Wang Jinshuang, Yang Lanjuan and Pan Lin, "SQL Injection Teaching Based on SQLi-labs", 2020 IEEE 3rd International Conference on Information Systems and Computer Aided Education (ICISCAE).

[5] Hanan Alsobhi, Reen Alshareef, "SQL Injection Countermeasures Methods", 2020 International Conference on Computing and Information Technology, University of Tabuk, Kingdom of Saudi Arabia.

[6] Li Qian, Zhenyuan Zhu, Jun Hu, Shuying Liu, "Research on SQL Injection Attack and Prevention Technology", 2015 International Conference on Estimation, Detection and Information Fusion (ICEDIF 2015).

[7] Zain Marashdeh, Khaled Suwais, Mohammad Alia, "A Survey on SQL Injection Attack: Detection and Challenges", 2021 International Conference on Information Technology (ICIT).

[8] Atefeh Tajpour, Mohammad JorJor zade Shooshtari, "Evaluation of SQL Injection Detection and Prevention Techniques", 2010 Second International Conference on Computational Intelligence, Communication Systems and Networks.

[9] Atefeh Tajpour, Maslin Massrum, Mohammad Zaman Heydari, "Comparison of SQL Injection Detection and Prevention Techniques", 201O 2nd International Conference on Education Technology and Computer (ICETC).

[10] Srdjan Zivanic, Stefan Ruvceski, Ilija Basicevic, "Network security education: SQL injection attacks".

[11] Aditya Rai, MD. Mazharul Islam Miraz, Deshbandhu Das, Harpreet Kaur, Swati, "SQL Injection: Classification and Prevention", 2021 2nd International Conference on Intelligent Engineering and Management (ICIEM).

[12] Kotomi Kuroki∗, Yo Kanemoto∗, Kazufumi Aoki∗, Yasuhiro Noguchi†, Masakatsu Nishigaki†, "Attack Intention Estimation Based on Syntax Analysis and Dynamic Analysis for SQL Injection", 2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC).

[13] Himanshu Gupta, Subhash Mondal, Srayan Ray, Biswajit Giri, Rana Majumdar, Ved P Mishra, "Impact of SQL Injection in Database Security", 2019 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE) December 11–12, 2019, Amity University Dubai, UAE.

Citation of this Article:

Omkar Kamleshwar Kambre, Kabir Kiran Shah, Pankaj Dulabhai Rathod, "SQL Injection Attacks and Defense Mechanisms" Published in *International Research Journal of Innovations in Engineering and Technology - IRJIET*, Volume 7, Issue 2, pp 101-106, February 2023. Article DOI https://doi.org/10.47001/IRJIET/2023.702016

****