Prevention of SQL Injection and Penetrating Attacks

A.S.Anakath* & R.Kannadasan** & S.Ambika***

* School of Computing, E.G.S. Pillay Engineering College (Autonomous), Nagapattinam, TN, India 611002
** School of Computer Science and Engineering VIT University, Vellore, TN, India 632 014
*** Research Scholar in Computer Science and Engineering University College of Engineering Ariyalur, Ariyalur, TN, India 621704

*Correspoding Author

ABSTRACT - In recent times hacking attacks is a very common and huge issue all over the world. Many large MNCs like Google, Apple, Microsoft etc. are investing millions of dollars to stop these types of Hacking and penetration attacks. Among these attacks SQL injection attack is considered as a very serious problem. It results in a very serious security related issues in web applications which handle information like personal bank details, account details, passwords to various accounts and other important data. In order to hack, the hackers are coming up with various new approaches. Researchers and web or cyber security analysts have proposed many methods to prevent these attacks by SQL injection. However, these attacks are on move and leads to vulnerabilities. Only a few techniques those take the advantage of these SQL vulnerabilities are known by the researchers. Hence, the solutions available so far solve only a few SQL related injections. This paper uncover this problem to the research community by means of a chronological review of various SQL injection attacks. We analyse the best ways to protect ourselves from these attacks. We have used an intentionally vulnerable site to demonstrate and understand the various types of SQL injections. We have also analysed and presented the various existing detection and prevention techniques against the SQL injection attacks.

Keywords: Injection, Attacker, Vulnerability, SQL Query, Prevention.

I. Introduction

An SQL injection attack typically means inoculation of a malicious SQL query as input by an attacker to the application. Then an effective SQL injection can lead to vulnerable attack and makes the attacker to read and modify the sensitive data in the database. It also executes the administrative operations on the database. In some cases, attackers inject different commands to operating system to gain access. SQL attacks are type of the injection attacks where the malicious SQL commands are inserted into the input data to result the implementation of the predefine SQL commands.

A. Types of sql injection attacks

- First order attack: Here the attacker through inputting a string of harmful data can modify the code which is to be executed.
- Second order attack: In this attack the attacker injects or penetrates into the storage data in the form of a trusted Source.
- Third order attack: In this attack the attacker is able to alter the implied function to char() by varying the data corresponds to the environment variables, NLS_Date_Format or NLS_Numeric_Characters.

B. Threat modelling

Tampering of the data, spoofing the identity of the person, changing the account balances, changing the passwords related to various web pages etc. can be done with the help of SQL injection attack. The total disclosure of all the sensitive information on the database to the open public is allowed by these attacks. It leads to destruction of the important information and it elevates the privileges to become the administrator of the database server. The SQL injection attacks are very severe problematic based on the attacker’s knowledge and techniques on the basic syntax, defence in depth counter measures like the low priority connections to the database server etc. In reality, considering SQL injection leads to severe problem.

C. Blind SQL injection

Here, the attacker instructs the database on several questions which have true or false as the answer. After each input is sent, there will be a response saying what exactly is processing inside the system. This attack is used whenever the web application is allowed to show a default error messages like there is no database available, user id not matched, password typed is wrong and also have not updated their databases and software in very long time with these patches are vulnerable to SQL injection. When a hacker penetrates or attacks SQL injection, then by default the web server screens an incorrect message from the database saying the typed syntax for the SQL queries is invalid. The blind SQL injection is very same as that of the normal SQL injection attack but the only difference here is that how the data is fetched.
from the database and shown to the end user. But whenever the web application is configured for not showing the error message, this makes exploiting the SQL injection vulnerability very difficult.

**D. Types of the blind SQL injection attacks**

There are two types of attacks in blind SQL injection method, they are explained below.

**Content based attack**

This attack uses a simple site that displays an item with the given ID which as the parameter to be passed. Then the attacker performs few simple tests to find whether the site is unprotected to SQL injections or not. Example of site URL:

http://vit.ac.in/items.php?id=5

Now we are going to send the following command into the database server.

```
SELECT title, description, body FROM items WHERE ID = 9
```

Few times the SQL injection command may send the false information messages which results where the attacker tries for other technique to destroy.

http://vit.ac.in/items.php?id=9 and 1=5

Here the modified SQL injection query looks like the following one.

```
SELECT title, description, body FROM items WHERE ID = 9 and 1=5
```

If the page or site in the Internet is unprotected to SQL injection, then it mostly displays nothing in the browser. This results in the opponent to use other input string to get the result as true.

http://vit.ac.in/items.php?id=9 and 1=5

**Time based attack**

Here in this type of attack, some attackers use selected few SQL statements in order to send the execution of the command to the database. By this technique, an attacker establishes each number of letter using the following logic:

If 'Z' is the first letter of the first database, then wait for 16 seconds.

If 'Y' is the first letter of the first database, then wait for 14 seconds.

Microsoft SQL Server url

http://www.vit.ac.in/attack.php?id=1' wait for delay '00:00:10'--

The rest of the paper is organized as follows. Section II discusses the related works, Section III explains the proposed method, Section IV discusses the results and the paper concludes with Section V.

**II. Related Works**

Many SQL injections and the prevention mechanisms have been developed. All these solutions follow various approaches as these are developed for various technologies which include network, server, and few applications. Other attacks such as XSS (Cross Site Scripting) are possible similarly by using SQL injection attacks. All attacks do not work on all the input forms available in the open source Internet. Some of them are specific to some vulnerabilities, while others are implementation-independent.

The detection of the sql injection attacks is achieved by using the graph of tokens and svm method [1]. The procedure includes the following steps. First, it generates the sequence of tokens from the sql query. Next, it produces a graph with tokens and their interaction. Then it uses a svm classifier in order to identify the queries at the runtime. The sql injection attacks is detected by using query result size [2]. The software used for this purpose consists of svv layer and qce layer. This method mainly focuses to prevent the injection attacks.

The sql injection attacks is detected by using the graph of tokens and svm method [3]. Here in order to evaluate the performance, hqb is used under different environments. This method uses three sqlia detectors, in which each represents different verification methods. The unsafe investigation and related security extensions for securing the NGN applications is dealt in a research work [4]. The main Aim of this paper is to magnify the end-to-end security attacks which are not properly understood by the security mechanisms. This method uses idp intrusion detecting and prevents the system to detect the sql injection.

Huang et al. discuss about the analysis on the vulnerability disclosure platforms in their work [5]. In this work first, the information is gathered then current security issues are analysed. Zineddine discusses on the various unprotected and mitigation techniques in the cloud [6]. Information and communication technology has many challenging security related issues. This paper presents the cloud security framework and coverage and the cost minimization using cuckoo search algorithm.

Scholte et al. studied on the input validation susceptibilities in Internet. This work is done by using the cwe classification system. This work mainly focuses the cross-siting, scripting and sql injection. The drawback is work that this paper hastensand focuses on the web applications [7]. Asghar et al. focus on real
Various types of SQL attacks or the penetrating attacks, unprotected and the prevention methods are presented in the work proposed by Kidny et al. [9]. Also, they have provided the future expectations on counter measures against SQL injection attacks. Aliero et al. proposed the protection technique on the SQL injection attacks [10]. Here, they used the concept of the instruction-set randomization of the SQL, in creating the instances of the data that are unknown to the hacker. MYSQL database is used with a proxy server which translates the SQL queries to its standard language.

Halfond and Orso discuss a technology for the identifying and stopping of the SQL injection attacks [11]. Here they introduced a model-based methodology for investigating the illegitimate or injurious SQL queries well in advance they use the server for execution. They have developed a tool called as AMNESIA. Bau et al. discuss about the automated black-box web application on unsafe testing in order to probe the web applications for security reasons [12]. Here we have used a custom type web application to know and project the vulnerabilities.

Halfond et al. review different types of the SQL injection attacks that are already known to illustrate this problem [13]. For every attack, they provide with descriptions of how attacks are performed. For each technique, they have discussed its strengths and weaknesses. Bandhakavi et al. discussed the prevention of SQL injection attacks [14]. This is achieved by using dynamic candidate evaluations. In this they have exhibited a powerful scheme in order to automatically transform the web applications in order to render them to safety against all SQL injection attacks.

Halfond et al. described that prevention of SQL injection attacks is achieved by using the positive tainting and syntax-aware evaluation method [15]. This method has both practical and conceptual advantages. This method is precise and efficient with minimal deployment requirements. Wei et al. proposed a technique to get protected against the penetrating attacks which are targeted at the stored procedures [16]. In 2005, Halfond and Orso proposed a work on the static-analysis and runtime monitoring to counter the SQL-injection attack [17]. They presented a technique to counter the SQL-injection, which used both the static and runtime monitoring to detect and stop the illegal queries. Fonseca et al. discuss SQL injection attacks as well as XSS attacks in their research work [18]. They proposed a system to evaluate automatic web unprotected scanners using in software fault injection methods.

Liu et al. takes on a modular approach which does not require any source code from the web applications or database in detecting the attack [19]. SQLProb prototype is used in order to measure the technique. Pietraszek and Berge have developed CSSE (context sensitive string evaluation) tool for preventing the SQL injection attacks [20]. A prototype CSSE was implemented for PHP platform and was corrected with PHPBB application. This work prevents many SQL Injection attacks.

In 2012 Lee et al. proposed a work to detect the SQL injection attack which compared SQL queries with the queries dynamically generated without the attribute values [21]. Here for analysis purpose this method removes the attribute values for the queries. This method is not only used in web applications but also used in various databases those are connected to the applications.

III. Proposed Method

In this paper we are focusing on the prevention methods with the use of the encoding techniques like 2 way encoding method. In this method performs encoding on the string before storing it into the database e.g., Base64. This will help in reduction of these attacks. This makes the string with fail proof. The SQL queries which contain the data is then prevented/blocked before the execution. Instruction-set randomization helps in encoding SQL keywords. A proxy is used to decode them, and it blocks the queries which contains the clear-text keywords.

We use Damm Vulnerable Website App (DVWA) technique in this paper to demonstrate the SQL injection attacks. This method is not a safe application in the Internet, which uses PHP/MySQL. The main objective is to provide different techniques and tools to web vulnerability analysts. These tools help to know better about the working of securing the web-based applications.

A. Implementation procedure

Step 1: Start any browser on any operating system including windows, Linux, Ubuntu etc.
Step 2: Place the address below here in the url, http://localhost/DVWA/login.php in order to get the Login page of the DVWA site.
Step 3: Login Details are to be entered in the login page in order to enter the DVWA site. Login: admin; Password: password.
Step 4: Click on the Login button to enter into the application.
Step 5: Click on the DVWA security which appears to the left side of the toolbar and select "low" in order to lower the security of the website, which we never try in reality and click submit.

Step 6: Select the "SQL Injection" from the left bar menu as shown in the figure below:

![Figure 1. SQL injection process window one](image1.png)

Step 7: Now the SQL injection attack begins. We start with simple input of "1" inside the text box and then click the submit button. Webpage is designed in such a way that to display the ID, First name, and Surname on the output device. The statement that is used to reveal, specifically $id is given below.

\[
\text{$get$_id = "SELECT STARTING\_name, FINAL\_name FROM USER WHERE userid = '$id';} \]

![Figure 2. SQL injection process window two](image2.png)

Step 8: Now, we are further going in depth in order to explore the database by inputting the below text into the user ID Textbox as shown in the figure %' or '0'='0 and click the submit button. During this attack we are instructing the system to display those records which are false and true. %' –Is not equal, and the result will be false. '0'='0' - Is equal to true, it is because 0 will always be equal to 0.

\[
\text{SELECT STARTING\_name, FINAL\_name FROM user WHERE user_id = '%'/0'=0';} \]

![Figure 3. SQL injection process window three](image3.png)

Step 9: In order to find the version of the database used we are performing the following. In the userid field %' or 0=0 union select null, version () # and then submit. Figure 4 shows the version of the database and attacker can try different attacks but it should be specific to the particular database.
IV. Results And Discussion

By doing series of attacks, after the input strings are fed with and we instruct the database to respond with any type of error message. Then all these input strings are injected and the desired result will be displayed. Hence userid and password are obtained. By using the rainbow tables, we can crack the password and can login to the system successfully. The injected string and the corresponding results obtained are listed in Table 1.

<table>
<thead>
<tr>
<th>S.No</th>
<th>Injected String</th>
<th>Result Obtained</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><code>%</code> or <code>0</code>='0`</td>
<td>First name,surname</td>
</tr>
<tr>
<td>2</td>
<td><code>%</code> or 0=0 union select null, version() #</td>
<td>It shows all records in Database.</td>
</tr>
<tr>
<td>3</td>
<td><code>%</code> or 0=0 union select null, user() #</td>
<td>Database user that executed the sql command.</td>
</tr>
<tr>
<td>4</td>
<td><code>%</code> or 0=0 union select null, database() #</td>
<td>It shows the Database Name.</td>
</tr>
<tr>
<td>5</td>
<td><code>%</code> or 0=0 union select null, table name from info_schema.tables #</td>
<td>It shows the Tables names</td>
</tr>
<tr>
<td>6</td>
<td><code>%</code> and 1=0 union select null, table name from info_schema.tables where table name like 'user%'#</td>
<td>User table containing the username and passwords are shown.</td>
</tr>
<tr>
<td>7</td>
<td><code>%</code> and 1=0 union select null, table name from info_schema.tables where table name like 'user' #</td>
<td>Different columns in the table are shown.</td>
</tr>
<tr>
<td>8</td>
<td><code>%</code> and 1=0 union select null, concat(table_name,0xa,column_name) from info_schema.columns where table name = 'users' #</td>
<td>Usernames and password information are shown.</td>
</tr>
</tbody>
</table>

A. Prevention techniques

(i) Sanitization

Data validation and sanitization are the important prevention techniques to be implemented strictly. Sanitization refers to testing of any data submitted through form field in the form of a function to make sure that any harmful or irrelevant characters are not present in the SQL query.

(ii) Firewall

Firewall is a software or a hardware on which many large companies are spending millions of dollars in order to prevent or filter the harmful data. This prevents the attacker uses to gain access over the system. In some standardized firewalls, we are able to alter the rules based on our own requirements. Firewall is a very useful application against the attacks in the Internet.

(iii) Using appropriate privileges

Whenever the administrative level message box asks to open the database, then never try to connect to the database unless you have any strong reason or mandatory. This makes the attackers to stop their
privileges easily and can get access to the admin account easily. Therefore, try to use an account which has limited number of privileges that are always controlled by the admin.

(iv) Stored Procedures

A stored procedure can be called from the application point of view which are stored in the database, instead of the user manually entering the data and command.

(v) Prepared Statements

Parameterized queries define the total SQL code in which parameters passed to it. Based on this query the server is permitted to differentiate code and data without bothering about the given input data.

The comparison of various SQL injection detection techniques are presented in the Table 2.

<table>
<thead>
<tr>
<th>Attacking Tools</th>
<th>Tautology</th>
<th>Stored Procedure</th>
<th>Blind Attack</th>
<th>Illegal/Incorrect</th>
<th>Piggy Backed</th>
<th>Union</th>
<th>Alternate Encoding</th>
<th>Timing Attack</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amnesia</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Sqlrand</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Csse</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Candid</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Tautology Checker</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Fgl</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>√</td>
<td>√</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Proposed</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
</tbody>
</table>

× Represents attack cannot be detected, √ represents attack can be detected.

V. CONCLUSION

In this paper, we have exhibited various methods or techniques for detecting and preventing SQL injection attacks. To achieve this, we first found out the various types of known SQL injection attacks. A set of attacks are revealed in this paper, while others are known to the attackers only. Next, we examined the methods, to secure ourselves from the various penetrating attacks and have learned about the different types of prevention methods in order to minimize the SQL injection attacks. Many of the methods suffer in handling these attacks using the alternate encoding techniques. The future scope of this work is to analyse the methods to prevent the blind SQL injection attacks and developing alternate methods for data submission.

REFERENCES


