© 2014 IJIRT | Volume 1 Issue 5 | ISSN : 2349-6002

SQL Injection

Vulnerabilities, Implementation and Prevention

Kajal Kanika, Monika Malhotra
Information Technology Branch
Dronacharya College of Engineering, Gurgaon, India

Abstract**—**SQL Injection is a method of injecting SQL commands into SQL statement via input boxes on web pages or web applications. Injected SQL commands can alter the SQL statements and may lead to security compromise in web application. It is one of the oldest techniques used to bypass passwords of the web applications or steal unauthorized information from relational database tables. SQL Injection is based on "is always true" principle. While SQL Injection is not straight forward but it is one of the major attacks to extract/steal information from web servers. Companies and Software developers put huge amount of efforts to prevent SQL Injection attacks by sanitizing passwords and usernames before executing them in SQL commands on the server. In this paper we will explore common mistakes which lead to hackers injecting SQL commands to online systems and will also study ways to prevent it.

Index Terms—*SQL; Injection; Prevention; Implementation;*

I. INTRODUCTION (*SQL INJECTION*)

SQL Injection is an old technique of injecting malicious SQL statements into data field. It is known as an attacker entity for web based applications.SQL Injection exploits the security vulnerability, for example, when user input is not correct or not strongly typed but it is executed. In 2012, security company 'Imperva' study that there are at least 4 attacks that an average web application received in a month. Web applications that are vulnerable to SQL Injection may allow an attacker to get full access. The applications that are in ASP.Net and J2EEE are less likely to have easily exploited by SQL Injections rather than the ones written in PHP and ASP. There are many preventions and solutions that are proposed by many researchers. However there is no one solution that can guarantee complete safety. The main Motive of SQL Injection is to attack the RDBMS running as back-end-systems at web servers, through web applications.

II. HISTORY

SQL Injection got focused in 1998 after the publication of an article in a magazine called "PHRACK". In year 2013 SOLI was rated the number one attack on the OSWAP top 10. In survey between 2007 to 2010 the open web application security project declared the SQLI in top10 web applications of vulnerabilities. In 2012 a representative from Barclaycard claimed that 97% of data breaches are a result of SQL Injection. Over one million pages were affected by the "Lilupophilupop" SQL Injection attack. Even in 2010 the official United Nations website got affected by SQL Injection. In 2011 the United States department of Homeland security, MITRE and SANS institute all named SQL Injection as one of the dangerous security vulnerability.

III. DIFFERENT FORMS OF SQL INJECTION, VULNERABILITIES AND ATTACKS

Type of Attack	***Name***	***Description***
Type I	First Order Attack	In this attacker can inject any malicious strings and cause the modified code to be executed immediately.
Type II	Second Order Attack	Attacker add into fields where data is stored such as rows or columns. An attack is subsequently executed by another activity.
Type III	Lateral Injection	Implicit functions are manipulated.

The Blind attack: Every login has username and password and email us our password link which prove downfall for that system.

When a user clicks on the link 'send my password 'of a web application, the system try to find out that email in user database and send reset information to that email. Our main motive is to check whether the string is without sanitizing and when submitting the form with a quote in an email address it will give error 550(server failure) this tells us that broken input is actually being parsed literally.

For Example

```
Select field list
 From table
Where field = 'abc@unix.net''
```

Note that closing quote mark which yields constructed SQL. This extra quote mark give syntax error and the SQL parser will abort the input query. Now here where clause is that in which data is filled. We will change the nature of where clause in an SQL legal way and see what happens:

Select field list from table

Where field = ‘abc’ or ‘x’=’x’;

In above SQL query where clause is divided into two component and ‘x’=’x’ is always true no matter what the first clause is. Now we know that how to manipulate the query to our own ends though we still don’t know much about the hidden part.

Finding the table name-

Many application’s built in query already have table built into it but we are not aware of it. There are many methods for finding that table names

For example

Select * FROM tablename

This will give the all the records in table name if table name is known.

Now suppose we assume that table name is “customer” so we can write above query like this

Select email, password, email id, full name

From customer

Where email= ‘x’ AND customer. email IS Null;--’

When this returned unknown email, it confirmed that our SQL was well formed and we had guessed properly the table name.

IV. PREVENTION FROM SQL INJECTION

There are many ways to prevent SQL Injection attacks. Prevention means that user input value should be valid and correct. These techniques force the client to enter correct data and can be barred to enter illegal value which is harmful to database server. The prevention can be done from client side as well as from server side. Various tools that are used for detecting and preventing SQL Injections are:

A. SQL Guard

SQL guard is a part of security concern devices that sit in front of databases, monitoring traffic for illegal and malicious activity. It has robust features which helps security manager to protect and alert them at the first sign of security. SQL guard is a combo of three modules such as- Health guard, Policy guard and Audit guard.

1) *Health Guard: Its work is, continuously monitor and assessing the database traffic i.e., input that is given to database. It gathers all information that helps other module to work efficiently. It also rates threat level according to the attack level.*

2) *Policy Guard: Policy guard offers policy generation tools, real time policy tools and automated policy tool. Security managers can access the restricted area of database through these policies.*

3) *Audit Guard: Audits guards offers granular tracking and reporting all the databases activities.*

B. CANDID

A candid is a simple and novel mechanism proposed by Bandhakavi et al for mining programmer intended queries by dynamically evaluating runs over benign candidate inputs. This tool retrofits web applications written in java to defend them against sql Injection attacks.

C. Randomization

This approach is proposed by Boyd and Keromytis in which a query language is used which is randomized, pointing a particular CGI in an application. A proxy server is there between a web server and SQL server. When query is send by SQL server to proxy server with randomized value which is received by client and de-randomized and it to server. This approach has two main advantages security and portability. If randomized value is known then this approach is not secure.

V. CONCLUSION

In this paper we have presented that what is SQL Injection, its past powerful attacks and prevention, detection approaches. Any organization that attempts to create secure database must consider each and every part of database environment including access method. By using hardware and software and the latest technology we can make database system more secure.

REFERENCES

[1] http://download.oracle.com/oll/tutorials/SQLInjection/html/lesson1/les01_tm_attacks.htm

[2] http://www.unixwiz.net/techtips/sql-injection.html