SQLi Prevention System

Shreyans Soni
2016CSB1146@iitrpr.ac.in

***Abstract*—My Project is about making such a framework or a PHP filter which prevent SQL injection from happening. In simple ways I want to create such mechanism which blocks the malicious SQL quries from the user side and take appropriate actions.**

I. Introduction

What is a SQL injection in the first place?. Let's understand first the terms than we will dive into the depths of the project.

A. SQL

SQL stands for "Structured Query Language". SQL is a standard language for storing, manipulating and retrieving data in databases.
example of a SQL query is as follows.

```
 SELECT col1, col2, col3...
FROM table_name1, table_name2
WHERE table_name1.col2 = table_name2.col1;
```

which extracts col1,col2,col3 from table 1 and 2 on certain conditions.

B. SQLi

SQL Injection (SQLi) refers to an injection attack wherein an attacker can execute malicious SQL statements (also commonly referred to as a malicious payload) that control a web applications database server (also commonly referred to as a Relational Database Management System RDBMS). Since an SQL Injection vulnerability could possibly affect any website or web application that makes use of an SQL-based database, the vulnerability is one of the oldest, most prevalent and most dangerous of web application vulnerabilities.
a simple example of a SQLi query is as follows.

```
 SELECT id FROM users WHERE username=username
    AND password=password OR 1=1
```

the working of such query grants user access to database see Figure 1

II. Why it is important

The main question is why it is so important and serious issue for a programmer/developers. The power which SQL contains has both sides of coins,an attacker can use this as follows:

- An attacker can use SQL Injection to bypass authentication or even impersonate specific users.
- One of SQLs primary functions is to select data based on a query and output the result of that query. An SQL Injection vulnerability could allow the complete disclosure of data residing on a database server.
- Since web applications use SQL to alter data within a database, an attacker could use SQL Injection to alter data stored in a database. Altering data affects data integrity and could cause repudiation issues, for instance, issues such as voiding transactions, altering balances and other records.
- SQL is used to delete records from a database. An attacker could use an SQL Injection vulnerability to delete data from a database. Even if an appropriate backup strategy is employed, deletion of data could affect an applications availability until the database is restored.
- Some database servers are configured (intentional or otherwise) to allow arbitrary execution of operating system commands on the database server. Given the right conditions, an attacker could use SQL Injection as the initial vector in an attack of an internal network that sits behind a firewall.

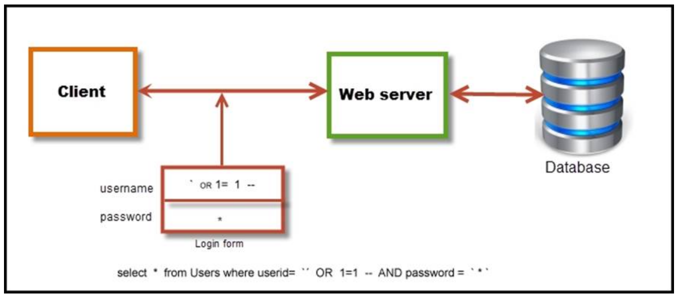
Client
Web server
Database
username
password
` OR 1= 1 --
*
Login form
select * from Users where userid= `` OR 1=1 -- AND password = `*`

Fig. 1. Block diagram of malicious SQL query on a server.

III. Technical Approach

The approach for making such a mechanism is as follows:

- Collecting all the data From security forums for recognizing SQLi pattern
- Selecting that data for a certain security level(as there are many types and levels of SQLi attacks).
- Designing a web based filter which filters all the client side queries.[1]

IV. Further Work

Further plans for this project is that I want to extend this project not only for preventing the SQLi vulnerability but also for other top web based vulnerability such as XSS,CSFR,LFI etc..
Also this project can also befurther extended to the Android Systems as android also uses databases for maintaining the

data which is handled by the SQL lite. I am citing a document for SQL injection. [2]

References

[1] S. W. Boyd and A. D. Keromytis. Sqlrand: Preventing sql injection attacks. In *International Conference on Applied Cryptography and Network Security*, pages 292–302. Springer, 2004.

[2] W. G. Halfond, J. Viegas, A. Orso, et al. A classification of sql-injection attacks and countermeasures. In *Proceedings of the IEEE International Symposium on Secure Software Engineering*, volume 1, pages 13–15. IEEE, 2006.