Research Plan & Project Summary

Varun Singh

Preventing SQL Injection to Secure Databases

Problem to be Solved

How can I prevent SQL Injection on a login system using PHP, MySQL, and SQL?

Rationale & Purpose

SQL Injection can cause a database to be destroyed, exposed, or even altered by a malicious user. It can also damage the reputation of a company. However, not all websites will recognize their forms as weak validation, which can make them open to threats. If all web sites could prevent SQL Injection, the web would be safer from hacking. I did this project to examine the potential threats and solutions of SQL Injection.

Research

- PHP Variable names are prepended with `$` and are not initialized
- PHP does not work by simply running in the browser because it needs to run on a web application server
- The MySQL localhost server can be accessed via the command line (mysql client) or phpmyadmin to run SQL commands
- Numerous commands are used for creating, altering, and populating tables in MySQL
    - `SELECT` shows a group of data that is selected
    - `ALTER TABLE` deletes or adds a column to a table
    - `USE *databasename*` selects a database and its tables
    - JOINs are queries that join multiple tables together
- Aggregate functions are SQL Functions that show data based on a group of values
- Before using PHP with MySQL, you use

`$link = mysqli_connect(SERVER_NAME, DB_USERNAME, DB_PASSWORD, DB_NAME);`
to connect to the server

Hypothesis

Based on my research, if I create a login system using PHP, MySQL, and SQL, and I sanitize and validate each input field, then I will have been able to secure my database because the malicious user will be unable to prevent SQL Injection.

Procedure

1. Download a server on computer - WAMP, XAMPP, LAMP, or MAMP
2. Create a Login System using PHP
    a. There should be the following pages:
        i. config.php - connects to the MySQL server and database
        ii. register.php - registers a user and inserts their data into the users table
        iii. login.php - page where user can login if they have an account
        iv. welcome.php - page that user is directed to if the login provided is valid
    b. Create a database for the login system that stores the users' information
        i. Create a users table
3. Create multiple accounts as multiple users - make sure you know all the usernames and passwords
4. Perform SQL Injection on the database
    a. Enter the desired information, a space, and a conditional that is always true, such as `1 = 1`
    b. Use `" or ""="`, another "smart" input and form of SQL Injection that always returns true
5. Validate/Sanitize Inputs in PHP code
6. Repeat steps 4-5 until there is no other way to perform SQL Injection with a regular keyboard and computer and no websites

Risks and Safety

Because everything is being done on the computer, the only potential risk is using the MySQL client incorrectly and destroying the database. Using the command line without experience could result in disastrous effects to the files on the computer.

Data Analysis

To take data, I will perform SQL Injection in different ways (e.g. through the url, input fields). Then, if it works, I will go back into the code and find a way to prevent that from happening. I will repeat this process until I have found no other ways to alter, drop, or expose my database if a malicious user was to attack it.

Works Cited

"Database Security Tips for Preventing SQL Injections." *DevSource*, 10 Dec. 2009. *General OneFile*, http://link.galegroup.com/apps/doc/A217852870/GPS?u=pl2439&sid=GPS&xid=ed50d57e. Accessed 18 Oct. 2018.

Kline, Kevin. "3 free tools that prevent SQL injection attacks: find vulnerabilities in your websites and databases before attackers do." *SQL Server Magazine*, Feb. 2009, p. 9. *Business Collection*, http://link.galegroup.com/apps/doc/A194919509/GPS?u=pl2439&sid=GPS&xid=e3d1f30d. Accessed 18 Oct. 2018.

Matelski, John. "Five best practices for securing databases." *Database Trends & Applications*, Feb.-Mar. 2015, p. 28. *General OneFile*, http://link.galegroup.com/apps/doc/A403588955/GPS?u=pl2439&sid=GPS&xid=9d78f6fc. Accessed 18 Oct. 2018.

"Microsoft Responds To The SQL Injection Problem." *eWeek*, 25 June 2008. *Business Collection*, http://link.galegroup.com/apps/doc/A217902950/GPS?u=pl2439&sid=GPS&xid=12c62d86 . Accessed 18 Oct. 2018.

Refsnes Data. “SQL Injection.” *w3schools*, W3.CSS, 1999, www.w3schools.com/sql/sql_injection.asp.

Ullman, Larry E. *PHP and MySQL for Dynamic Web Sites*. 5th ed., Peachpit Press, 2018. Print. Visual QuickPro Guide.