SQL Injection Assignment

Purpose: To attack a web application with SQL injection, and then secure the website against SQL injection.

Preparation: You must have successfully completed the Secure Database Setup Assignment.

Assignment Instructions:

1. Create a database for the web application.

   The database you will create will replace the database you used for the Secure Database Setup Assignment. You will use the same application user account that you established as part of the Secure Database Setup Assignment.

   Download the attached SQL file **my_guitar_shop2A.sql**. Load this file into MySQL Workbench (or phpMyAdmin). Run the SQL file in MySQL Workbench to create the database. This should proceed without any errors. If you don't already have a user created, the SQL file will create one with the password "pa55word". Otherwise, the user and password you established previously will continue to be used.

2. Install the Guitar Shop PHP web application on the web server that uses the database.

   Download and unzip the attached file **GUITAR.zip**. Paste the unzipped "GUITAR" directory and all of its subdirectories in the "htdocs" directory of your Apache server.

   Edit the file **GUITAR/model/database.php** and set the database access password to the password you used for the Secure Database Setup Assignment.

   In the Apache configuration file (**apache/conf/httpd.conf**) find the **DirectoryIndex** directive and add **index.php** as shown below:

   ```
   <IfModule dir_module>
      DirectoryIndex index.html index.php
   </IfModule>
   ```

   Restart the Apache server after making the change to **httpd.conf**.

   Test the Guitar Shop application by typing the following URL in your browser address bar:

   http:/localhost/GUITAR

   Click on the various links to verify the application is working, and retrieving product information from the database. The navigation code for this site is a little quirky, so use the links in the site for navigation. Typing in something like **localhost/GUITAR/Index.php** won't work (it works for the first page you see, but the pages after that won't work correctly). Note: only manipulate URLs after you have seen the URLs generated by the site's own links.

3. Attack the web application with SQL injection.

   Develop four different SQL injection attacks that delete all the contents of the "products" table in the web application's database. The vulnerable pages are any page that generates a GET request or a POST request

that causes the web application to access the database. In the case of GET requests, your SQL injection can be done in a URL parameter. In the case of POST requests, the SQL injection can be typed into form fields.

As you browse this website, keep an eye on the URLs that occasionally appear (for the GET requests within the site). Those URLs can give you useful information. Also, if you inspect some of the form pages (using your browser inspector), the hidden fields that you can discover can also help you design your exploits. You can also try submitting a form with a GET request through URL manipulation (hint: as an alternative to the product "Delete" button on the website). Who knows, it might work.

4. Secure the website against SQL injection and URL manipulation.

   The Guitar Shop website has many problems, and a multilayer defense can be established. **Your security approach should include the use of prepared statements when accessing the database**. However, there may be opportunities to restrict the use of GET requests (as in the case of the "Delete" button) and to filter input before it is used in a query. Files that you may need to edit include the following:

   - **GUITAR/model/product_db.php** – this file contains all of the database queries.
   - **GUITAR/admin/product/index.php** – this file processes all the requests for admin functions that access the database.
   - **GUITAR/catalog/index.php** – this file processes all the non-admin requests that access the database.

   Demonstrate that you have prevented all of the exploits developed in Step 3.

5. Prepare your group presentation.

   Each group will give a 5 to 8 minute presentation that includes the following:

   - Briefing slides explaining the SQL injection attacks you used
   - Demonstration of the SQL injection attacks
   - Briefing slides explaining how you secured the website against your SQL injection attacks
   - Demonstration of the secure website