Web Security Project

By
Manuel Ramirez & Klaus Menendez
CS 476 Spring 2025

Table Of Contents

Using the Program

This project is designed to run locally on any system (Windows, macOS, or Linux) by setting up a standard LAMP environment using Apache, MariaDB, and PHP. The original development environment used WSL2 on Windows, but the setup can be adapted to suit your system.

Clone The Repository:

Clone the repo using the following terminal command -

```
git clone https://github.com/EspressoPlanet/web-application-security-project.git
```

Other Tools:

- **sqlmap** (optional, for SQL injection demo)

- **hashcat** and a wordlist like **rockyou.txt** (downloadable from link)

Platform Specific Instructions:

Windows Users (via WSL2)

1. **Install WSL2 and Ubuntu**:
   - Follow Microsoft's guide.
2. **Launch Ubuntu (WSL2)** and run:

   ```
   sudo apt update
   sudo apt install apache2 php mariadb-server libapache2-mod-php
   ```

3. **Start Services** by running:

   ```
   sudo service apache2 start
   sudo service mysql start
   ```

4. **Access App**: Visit http://localhost in your browser.

macOS Users

1. **Install Homebrew** (if not installed):

   ```
   /bin/bash -c "$(curl -fsSL
   https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
   ```

2. **Install Apache, PHP, and MariaDB**:

```
brew install httpd php mariadb
```

3. **Start Services**:

```
brew services start httpd
brew services start mariadb
```

4. **Clone the Repo** and point Apache's DocumentRoot to your project directory.
5. **Visit** http://localhost

Linux Users

1. **Install Services**:

```
sudo apt update
sudo apt install apache2 php mariadb-server libapache2-mod-php
```

2. **Clone and Move Project**:

```
git clone https://github.com/your-username/web-application-security-project.git
sudo mv web-application-security-project /var/www/html/
```

3. **Start Services**:

```
sudo service apache2 start
sudo service mysql start
```

4. **Access App**: Visit http://localhost

Populating The Local User Database

To set up the MySQL/MariaDB database required for the project, follow these steps:

1. **Ensure MariaDB or MySQL is running on your system.**
   On WSL or Linux-based systems, you can start it with:
   ```
   sudo service mysql start
   ```
2. **Open a MySQL shell and log in as root user:**
   ```
   mysql -u root -p
   ```
3. **Enter your root password when prompted.**

4. **Run the setup SQL file to create and populate the database:**
   Exit the MySQL shell if you're still in it, then from your terminal, navigate to the project directory:
   ```
   cd /path/to/web-application-security-project
   ```

Then run:

```
mysql -u root -p < setup.sql
```

This script will:

- Drop the database bankapp if it already exists.
- Create a new bankapp database.
- Create a users table with username and MD5-hashed password fields.
- Populate it with test users and weak MD5-hashed passwords.

Using The Application

SQL Injection:

1. Click the **Vulnerable** or **Secure** button depending on what you would like to test.
2. Use the following values in the login form:
   a. Username: **‘ or 1=1 --**
      i. Please note: The two dashes at the end have a space afterwards
   b. Password: anything can be typed here
3. Click **Login** to observe how the secure and vulnerable versions function

Reflected XSS:

1. Login via SQL injection or normally using the following:
   a. Username: User1
   b. Password: password123
2. Type a script into the secure and vulnerable forms to see its effect
   a. Sample Script: **<script>alert('XSS');</script>**
3. Submit the form to see a popup after script execution on the vulnerable form or the sanitized note appear on the secure form.

Password Cracking:

1. The password hashes have already been extracted using SQL map and are saved in **md5hashes.txt** and **bcryptHashes.txt**
2. To run the scripts you will need to install HashCat via the link above first.
3. Run the md5 cracking script to crack the md5 hashes
   a. Use the following terminal command: **./crack_md5.sh**
4. Run the Bcrypt cracking script to crack the Bcrypt hashes
   a. Use the following terminal command: **./crack_bcrypt.sh**

5. The terminal will show results for each. Bcrypt will take days, so end the program to view the estimated crack time.

File Upload:

1. To test the vulnerable feature, upload the provided file with dangerous embedded code or a file with a modified name including a script such as: **<script>alert(1)</script>**
2. Accessing the file via the /uploads/ address in the browser will execute the payload.
3. The secure feature will reject any uploads like the ones attempted above.
    a. The secure backend now checks for specific file extensions, malicious keywords, embedded code, and shell command patterns.