Project Report:

Project Ivory

Contents

Introduction

Project Ivory is a web testing application developed by TITLE TECHNOLOGY. The objective of this project was to create a user-friendly and comprehensive platform for testing various aspects of web application security. The application allows users to perform tests for Cross-Site Scripting (XSS), SQL Injection, Brute-Force Attacks, Session Management, Access Control, and Request Validation. This report provides an overview of the project, including its objectives, features, implementation details, and future enhancements.

2. Project Objectives

The main objectives of Project Ivory were as follows:

- Develop a web testing application with a focus on security testing for web applications.
- Implement features for testing common vulnerabilities such as XSS and SQL Injection.
- Provide functionality for testing session management, access control, and request validation.
- Create an intuitive and interactive user interface.
- Ensure compatibility and functionality across multiple web browsers.
- Implement proper error handling and security measures.
- Provide the ability to generate reports summarizing the testing results.

3. Features and Implementation

Project Ivory includes the following key features:

3.1 Registration and Login:

- Users can create an account and securely log in using a username and password.
- User registration details are stored in memory for demonstration purposes.

3.2 Cross-Site Scripting (XSS) Testing:

- Users can input malicious scripts to test if the application properly sanitizes the input.
- The application includes server-side validation and sanitization of user inputs to prevent XSS attacks.

3.3 SQL Injection Testing:

- Users can input SQL queries to check if the application is vulnerable to SQL injection attacks.
- The application utilizes parameterized queries to prevent SQL injection vulnerabilities.

3.4 Brute-Force Attacks:

- Users can test the login functionality by attempting multiple password guesses or using a brute-force tool.
- The application includes measures to prevent brute-force attacks, such as account lockouts and rate limiting.

3.5 Session Management:

- The application securely handles session tokens, ensuring token expiration and regeneration.
- Session tokens are stored in server-side session storage.

3.6 Access Control:

- The application restricts access to protected resources by enforcing proper access controls.
- Unauthorized access to protected routes, such as the /admin route, is prevented.

3.7 Request Validation:

- The application validates and sanitizes all user inputs, including query parameters, form submissions, and file uploads.
- Server-side validation and sanitization prevent malicious actions and protect against common security vulnerabilities.

3.8 Generate Report:

- Users can input a website link to generate a report summarizing the testing results for the specified website.
- For demonstration purposes, the application displays a success message upon report generation.

3.9 View Report:

- Users can view previously generated reports for their account.
- Dummy report data is displayed for demonstration purposes.

4. Technology Stack

The following technologies were used in the development of Project Ivory:

- Python: The Flask framework was used for developing the web application backend.
- HTML/CSS: HTML was used for creating the structure of the user interface, and CSS was used for styling.
- JavaScript: Some interactive elements and client-side validation may be implemented using JavaScript.
- Database: For demonstration purposes, user account information and generated reports are stored in memory.

5. Conclusion

Project Ivory successfully achieved its objectives of creating a user-friendly web testing application for assessing web application security. The implemented features cover a range of common vulnerabilities and security testing scenarios. The application provides a solid foundation for further enhancements and integration with additional security testing techniques and tools.

6. Future Enhancements

- Integration with additional security testing techniques, such as Cross-Site Request Forgery