Task4_Ans_sytmatic_GPT

Penetration Testing Report

Executive Summary

This penetration testing report documents the process and findings from testing a web application with multiple levels, each with unique vulnerabilities. The goal was to identify security flaws and provide recommendations for enhancing the application's security.

Scope

The scope of the penetration test included all levels of the web application. Each level presented a unique security challenge that had to be solved to advance to the next.

Overview 1

Description

This level, referred to as "The Idiot Test," requires the user to find the password hidden within the webpage's source code.

Key Findings

- **Vulnerability**: Password disclosed in HTML comments.
- **Password**: a13fe647

Recommendations

- Remove sensitive information from HTML comments.
- Use server-side validation for passwords.

Overview 2

Description

The password protection script loads the password from an unencrypted text file that wasn't uploaded.

Key Findings

- **Vulnerability**: Absence of the password file causes the script to match an empty value.
- **Password**: None (Submit without entering anything)

Recommendations

- Ensure all required files are present and properly secured.

- Implement server-side validation to handle missing files gracefully.

Overview 3

Description

The password file is uploaded but referenced within a hidden input field in the HTML form.

Key Findings

- **Vulnerability**: Password file location disclosed via hidden input field.
- **Password**: 990514aa

Recommendations

- Avoid disclosing sensitive file paths in the client-side code.
- Use server-side mechanisms to handle sensitive data.

Overview 4

Description

A script is used to email the password to Sam automatically. The email address is in a hidden input field.

Key Findings

- **Vulnerability**: Email address manipulation via hidden input field.
- **Password**: 7b92d260

Recommendations

- Validate and sanitize input on the server-side.
- Remove or secure hidden fields containing sensitive information.

Overview 5

Description

Sam attempted to make the email program more secure but left the same hidden input vulnerability.

Key Findings

- **Vulnerability**: Persistent hidden input field for email.
- **Password**: e4643a2d

Recommendations

- Implement proper input validation and sanitization.
- Use secure methods for handling email processes.

Overview 6

Description

The password is encrypted, and the encryption algorithm is available.

Key Findings

- **Vulnerability**: Predictable encryption algorithm.
- **Password**: b15a2205

Recommendations

- Use strong, well-tested encryption algorithms.
- Keep encryption algorithms and keys secure.

Overview 7

Description

An obscurely named file contains the password, and a UNIX `cal` command script can be exploited.

Key Findings

- **Vulnerability**: Command injection via `cal` command.
- **Password**: 3624eafe

Recommendations

- Sanitize and validate all user inputs to prevent command injection.
- Limit the exposure of sensitive files and directories.

Overview 8

Description

A vulnerable SSI (Server Side Includes) script can be exploited to retrieve the password.

Key Findings

- **Vulnerability**: SSI Injection.
- **Password**: 27d6aed9

Recommendations

- Disable SSI or implement strict input validation.
- Regularly audit and secure scripts for injection vulnerabilities.

Overview 9

Description

The vulnerability from level 8 still exists, and it can be used to access the password file for level 9.

Key Findings

- **Vulnerability**: Persistent SSI Injection.
- **Password**: a9e3428d

Recommendations

- Fix the existing SSI injection vulnerability.
- Implement robust security controls to prevent exploitation across different levels.

Overview 10

Description

A temporary and hidden approach is used for authentication, which can be bypassed using JavaScript.

Key Findings

- **Vulnerability**: JavaScript-based cookie manipulation.
- **Password**: Change `level10_authorized` cookie value to `yes`

Recommendations

- Avoid relying on client-side security mechanisms like JavaScript for authentication.
- Use secure, server-side session management.

Overview 11

Description

An insecure Apache configuration allows directory listing, leading to the discovery of the password.

Key Findings

- **Vulnerability**: Directory listing and insecure file access.
- **Password**: Varies (Example: short)

Recommendations

- Disable directory listing in the Apache configuration.
- Securely configure web server settings to prevent unauthorized access.

Conclusion

The penetration test identified various vulnerabilities across different levels of the web application. Implementing the provided recommendations will enhance the security and robustness of the application. Regular security audits and adopting best practices for secure coding are essential to maintaining the application's integrity.