Security Vulnerability Remediation Report

DHL Waybill Printing Application

Date: June 18, 2025

This report summarizes the security vulnerabilities identified in the DHL Waybill Printing application and the corresponding remediations implemented. The fixes were based on the findings from the following reports:

- RITM9474447_Homepage Waybill Printing_Pentest_Report_06JUNE2025
- RITM8310099-HK.APP.HomepageWaybillPrinting-v2.0
- vulnerabilities fixing 9.pdf and 10.pdf

Each issue is described with the original problem, the code changes made, the reason for the change, and best practices for future prevention.

---

1. Response Manipulation (Authentication Bypass)

Description:

The application previously relied on client-side redirects or status codes to indicate login success or failure, which could be manipulated to bypass authentication.

Fix Implemented:

Replaced client-side redirects with server-side validation and used `response.sendError(HttpServletResponse.SC_UNAUTHORIZED)` to handle invalid login attempts securely.

Best Practices:

Always validate authentication server-side and avoid exposing authentication logic to the client.

---

2. Missing Authentication on Protected Pages

Description:

Several JSP and CGI pages did not validate user sessions, allowing unauthorized access to protected resources.

Fix Implemented:

Added session validation checks in all JSP files using:

`if (session.getAttribute("emailadr") == null) { response.sendRedirect("../toLogin.html"); return; }`

Added session token validation in all CGI scripts using `is_valid_session()`.

Best Practices:

Ensure all protected endpoints validate user sessions and enforce access control.

---

3. Lack of Rate Limiting

Description:

The login endpoint allowed unlimited login attempts, making it vulnerable to brute-force attacks.

Fix Implemented:

Implemented login attempt tracking using `sf.overAttempt(10, emailadr)` to block excessive login attempts.

Best Practices:

Implement rate limiting or CAPTCHA on authentication endpoints to prevent brute-force attacks.

---

4. Improper Session Management

Description:

Sessions were not invalidated on logout, and session fixation was possible after login.

Fix Implemented:

Added `session.invalidate()` on logout and created a new session after successful login to prevent session fixation.

Best Practices:

Always invalidate old sessions on logout and create new sessions after login to prevent session hijacking.

---

5. Improper Cookie Attributes

Description:

Session cookies were missing the `SameSite` attribute, which could allow CSRF attacks.

Fix Implemented:

Set the `SameSite=Strict` attribute manually using:

`response.setHeader("Set-Cookie", sessionCookie.getName() + "=" + sessionCookie.getValue() + "; HttpOnly; Secure; SameSite=Strict; Path=/");`

Best Practices:

Always set `HttpOnly`, `Secure`, and `SameSite` attributes on session cookies.

---

6. Content-Security-Policy (CSP) Header

Description:

Some pages were missing CSP headers, increasing the risk of XSS attacks.

Fix Implemented:

Added CSP headers to all JSP and HTML pages using:

`Content-Security-Policy: default-src 'self'; object-src 'none'; frame-ancestors 'none'; upgrade-insecure-requests; block-all-mixed-content`

Best Practices:

Use CSP headers to restrict content sources and mitigate XSS risks.
