Vulnerability Name	Reflected Cross-site Scripting
Affected Vendor, Version	OpenKM 6.3
Affected Endpoint	http://localhost:8080/OpenKM/frontend/index.jsp
Product Official Website URL	https://www.openkm.com/
Affected Component:	Affected parameter: Key (Search parameter)

Description: - Reflected Cross-Site Scripting (XSS) is a web application vulnerability in which an attacker injects malicious JavaScript into a request that is immediately reflected by the server in the HTTP response and executed in the victim's browser. The payload is not stored on the server; instead, it is returned through dynamically generated content such as search results, filter fields, or URL parameters. This occurs when user-controlled input is included in the response without proper validation and context-aware output encoding.

Root Cause: -The root cause of this vulnerability is improper handling of user-supplied input due to insufficient input validation and lack of context-aware output encoding. The application reflects untrusted input directly into the HTML response, causing the browser to interpret it as executable script rather than data.

Impact: - Successful exploitation of this reflected XSS vulnerability allows an attacker to execute arbitrary JavaScript in the victim's browser context. This can lead to session token theft, credential harvesting, phishing attacks, or unauthorized actions performed on behalf of the user. If exploited against an authenticated or administrative user, it may result in misuse of privileged functionality and exposure of sensitive data. Such attacks can also cause reputational and security risks for the affected organization.

Mitigation: - Apply strict server-side input validation and reject unexpected or dangerous characters in user-controlled parameters. Sanitize input where appropriate and ensure context-aware output encoding is applied before rendering data in HTML, attribute, URL, or JavaScript contexts. Enforce a strong Content Security Policy (CSP) to restrict script execution and reduce the impact of injection attempts. Limit inline scripts and dangerous DOM sinks where possible.

Remediation: - Update the application code to implement proper context-sensitive output encoding for all reflected user input. Use secure templating frameworks or encoding libraries that automatically escape untrusted data. Introduce centralized input handling and validation controls, and perform secure code reviews focused on injection flaws. Conduct regular security testing and penetration testing to detect XSS issues early. Deploying a Web Application Firewall (WAF) can provide additional detection and blocking of malicious payloads as a defense-in-depth measure.

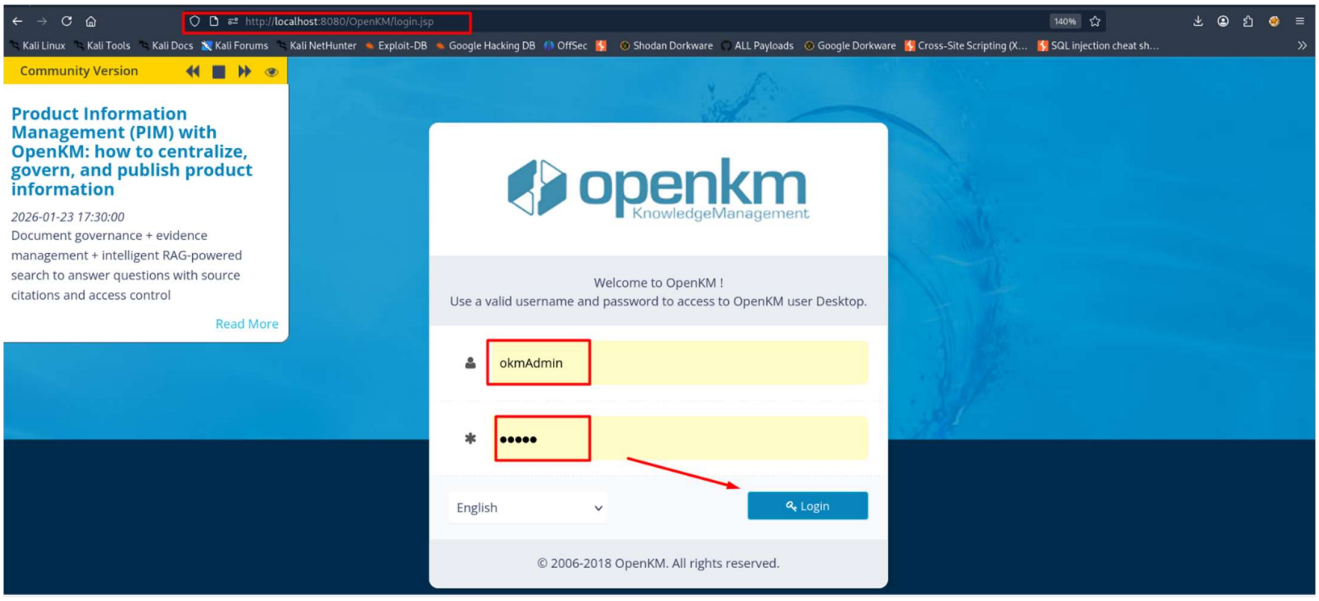
Severity Score

CVSS v3.1 Base Score: 6.1 (Medium)

Vector: AV:N/AC:L/PR:N/UI:R/S:C/C:L/I:L/A:N

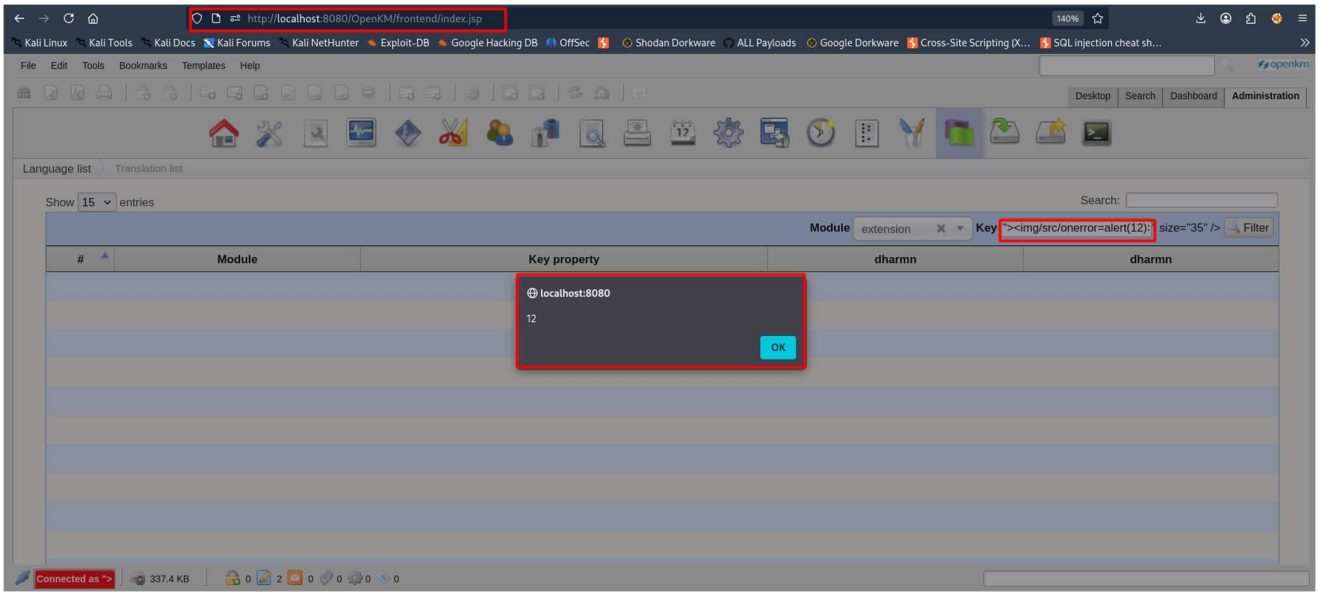
Proof of Concept

Step 1: - First navigate to http://localhost:8080/OpenKM/login.jsp and login with username and Password.



Step 2: - The vulnerability can be reproduced by injecting a crafted payload into the **Key (Search parameter)** ,for example:

"><img/src/onerror=alert(12)>



References

1. OWASP — Cross-Site Scripting (XSS)
   https://owasp.org/www-community/attacks/xss/
2. OWASP — XSS Prevention Cheat Sheet
   https://owasp.org/www-community/xss-prevention

Thankyou