03
Reporting Practice

Executive Summary

This assessment evaluated the security posture of a target web application to identify potential vulnerabilities that could be exploited by malicious users. The testing focused on common web-based threats, including SQL Injection, Cross-Site Scripting (XSS), Command Injection, and Unrestricted File Upload. Each vulnerability was analysed, scored using the CVSS framework, and accompanied by recommended remediation strategies. The findings indicate several high and medium-severity risks that could lead to unauthorized access, data compromise, or system control if left unmitigated.

Technical Findings

1. SQL Injection (F001)

- Description: The application failed to sanitize user inputs in a login form, allowing attackers to manipulate SQL queries.
- Impact: Enables database extraction, authentication bypass, and possible data modification.
- Evidence: Injecting admin' OR '1'='1 in the username field successfully bypassed authentication.
- CVSS Score: 9.1 (Critical)
- Remediation: Implement parameterized queries or prepared statements. Validate and sanitize all user inputs.

2. Cross-Site Scripting (XSS) (F002)

- Description: Reflected XSS vulnerability found in the search input field.
- Impact: Allows injection of malicious JavaScript, leading to session hijacking and user credential theft.
- Evidence: Payload <script>alert('XSS')</script> executed successfully on the client side.
- CVSS Score: 8.2 (High)
- Remediation: Encode output data before rendering. Use Content Security Policy (CSP) and input sanitization.

3. Command Injection (F003)

- Description: The application's system command interface accepts unsanitized parameters.
- Impact: Remote attackers can execute arbitrary OS commands, potentially gaining full control of the host.
- Evidence: Inputting ; whoami in the "ping" field returned the system username.
- CVSS Score: 9.5 (Critical)
- Remediation: Validate and sanitize all command inputs. Use safe API calls instead of direct system commands.

4. Unrestricted File Upload (F004)

- Description: The upload function allows files without validating type or extension.
- Impact: Enables attackers to upload web shells or malicious executables, leading to remote code execution.

- Evidence: Uploaded a .php shell file and accessed it via browser to execute arbitrary commands.
- CVSS Score: 9.8 (Critical)
- Remediation: Implement file type and size restrictions. Use server-side file validation and store uploads outside the webroot.

Findings Table

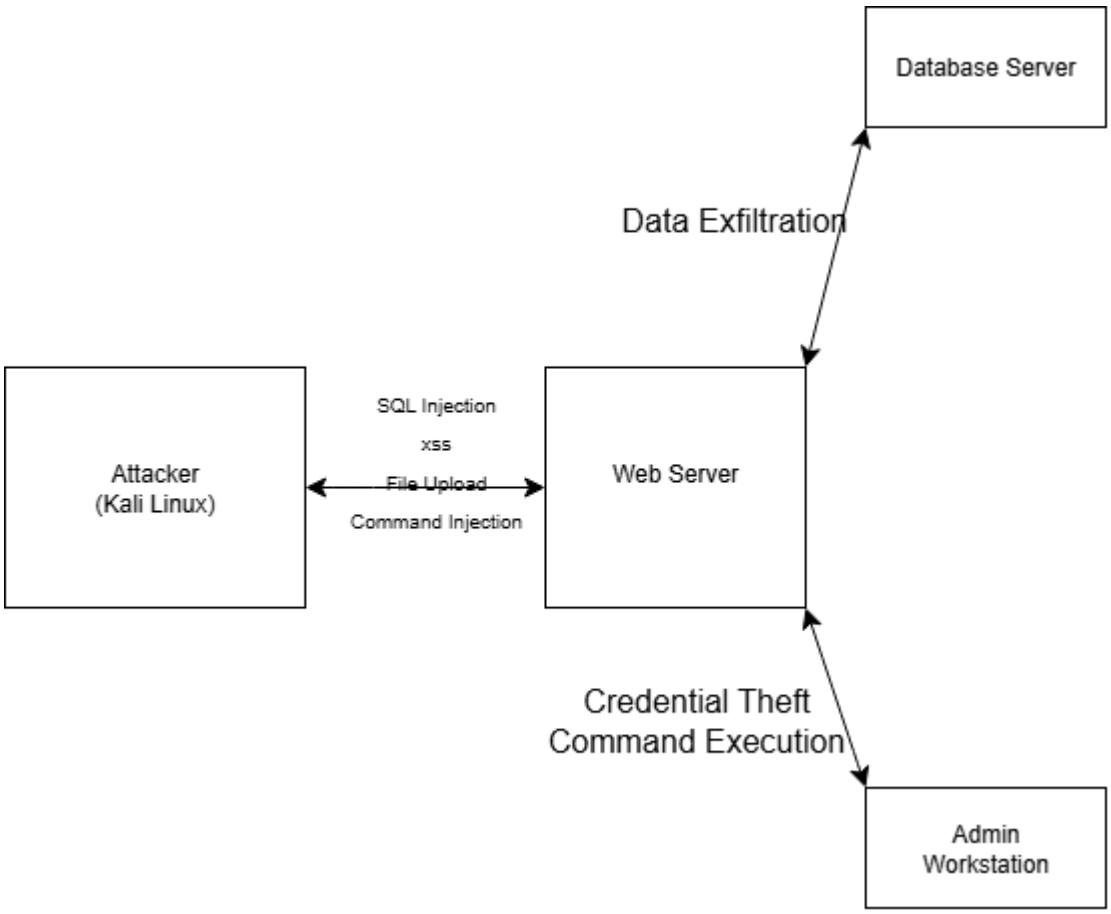
Finding ID	Vulnerability	CVSS Score	Remediation
F001	SQL Injection	9.1	Use parameterized queries
F002	Cross-Site Scripting (XSS)	8.2	Sanitize and encode user input/output
F003	Command Injection	9.5	Use safe APIs and validate input
F004	Unrestricted File Upload	9.8	Restrict file type and validate uploads

Remediation Plan

Priority	Action Item	Responsible Team	Deadline
High	Fix SQL Injection and File Upload flaws	Development	7 days
Medium	Apply XSS protection (input encoding)	Development	14 days
High	Patch Command Injection issue	DevOps	7 days
Low	Perform follow-up penetration test	Security Team	30 days

Visualization

Network Attack Path



Attack Flow:

Attacker → Web App (via SQLi / XSS / File Upload) → Server Command Execution → Database Exfiltration → Admin Credentials Compromise

Non-Technical Summary

The security assessment identified multiple high-risk vulnerabilities in the organization's web application that could allow attackers to steal sensitive data or gain control of internal systems. Weak input validation in forms, unsafe file upload features, and missing output encoding are key issues. If exploited, these flaws could result in data breaches, website defacement, or unauthorized access. It is strongly recommended that immediate remediation steps be taken, such as validating all user inputs, restricting upload functionality, and applying secure coding practices to protect the organization's digital assets.