Mini-Project Proposal JWT Vulnerabilities Laboratory

Team 6

2025/2026

1 Introduction

This mini-project proposes the development of a hands-on laboratory exercise focused on security vulnerabilities in JSON Web Tokens (JWT). The objective is to help students understand how improper implementation of authentication mechanisms can lead to critical security flaws in modern web applications.

The lab follows a progressive and guided approach, combining practical exploitation with secure implementation practices. Students will first exploit vulnerabilities and then learn how to properly fix them.

2 Differentiation from Existing Work

While existing resources such as SEED Labs and PortSwigger Web Security Academy cover JWT vulnerabilities, they typically focus on isolated attack scenarios.

This lab differentiates itself by:

- Combining multiple JWT vulnerabilities into a single progressive learning path
- Including both offensive exploitation and defensive hardening
- Providing a transparent and modifiable vulnerable server implementation

This approach improves clarity and allows students to understand the broader security implications of JWT misuse.

3 Lab Structure

The lab is divided into four progressive tasks, each focusing on a specific vulnerability or security concept.

3.1 Task 1 — `alg:none` and Claim Tampering

Learning Objective: Understand the importance of signature verification and the risks of trusting client-controlled data.

The server accepts unsigned tokens when the header specifies `"alg":"none"`. The student intercepts a valid JWT, decodes it, and modifies the payload by changing `"role":"user"` to `"role":"admin"`. The signature is removed and the algorithm set to `none`.

The manipulated token is accepted by the server, granting access to protected endpoints. This task demonstrates the consequences of failing to enforce signature validation.

3.2 Task 2 — Algorithm Confusion (RS256 → HS256)

Learning Objective: Understand the difference between symmetric and asymmetric cryptography and how improper algorithm validation breaks the security model.

The server uses RSA-based signatures (RS256) but does not properly validate the algorithm field. The student obtains the server's public key and uses it as a secret to sign a forged token using HS256.

Due to incorrect verification logic, the server accepts the forged token. This task highlights a real-world vulnerability caused by misuse of cryptographic primitives.

3.3 Task 3 — `kid` Header Injection (Path Traversal)

Learning Objective: Understand how unsanitized input in security-critical operations can lead to severe vulnerabilities.

The server uses the `kid` field in the JWT header to select a verification key from the filesystem. The student manipulates this field to perform a path traversal attack (e.g., `"../../../../../tmp/attacker.key"`), forcing the server to load a malicious key.

By signing a token with the attacker-controlled key, the student bypasses authentication and gains unauthorized access.

3.4 Task 4 — Fix and Hardening of JWT Validation

Learning Objective: Learn how to securely implement JWT validation and prevent common vulnerabilities.

After exploiting the vulnerabilities, the student must secure the server implementation. The goal is to prevent all previously demonstrated attacks.

The student is required to:

- Reject tokens using `"alg":"none"`
- Enforce a strict whitelist of allowed algorithms (e.g., only RS256)
- Properly separate symmetric and asymmetric verification logic
- Validate and sanitize the `kid` field to prevent path traversal
- Optionally enforce additional checks (expiration, issuer, audience)

Finally, the student verifies that all previous attacks are no longer successful.

4 Technical Implementation

The lab will be implemented as a simple web application with intentionally vulnerable JWT validation logic.

- Backend: Python (Flask)
- Environment: Docker container for reproducibility
- Tools: curl, browser, and optional JWT debugging tools

The source code will be provided to students to ensure transparency and facilitate understanding.

5 Conclusion

This lab provides a comprehensive and practical exploration of JWT vulnerabilities. By combining exploitation and defense in a structured and progressive format, students gain a clear understanding of both how these vulnerabilities arise and how they can be prevented.