<aegiscodelabs/>

SAMPLE REPORT

Penetration Testing Report

Web Application Security Assessment

Prepared for: Northbridge Commerce

Assessment window: 5-12 May 2026

Engagement Overview

This sample document demonstrates the reporting structure delivered by AegisCode Labs after a penetration testing engagement. All systems, hosts, and findings shown here are fictional and provided for illustration only.

Scope

Customer portal, authentication flow, REST API

Methodology

OWASP WSTG, OWASP Top 10, manual validation

Outcome

2 High, 3 Medium, 4 Low findings

Confidentiality Notice

Sample only. Real client reports include evidence, affected assets, severity rationale, reproduction steps, and remediation guidance tailored to the tested environment.

Executive Summary

AegisCode Labs identified several weaknesses that could affect confidentiality, integrity, and account security if left unresolved. The most significant issues were an authorization flaw in order management and missing brute-force protection on the login endpoint.

Risk Distribution

High	Medium	Low
2	3	4

Findings Overview

ID	Finding	Severity	Status
AC-01	Broken object level authorization	**High**	Open
AC-02	Missing login rate limiting	**High**	Open
AC-03	Verbose error response	**Medium**	Open
AC-04	Weak cookie attributes	**Medium**	Open
AC-05	Outdated dependency	**Medium**	Open

Recommended Priorities

1. Enforce server-side ownership checks for every order resource.
2. Add throttling, lockout, and alerting to authentication endpoints.
3. Standardize secure cookie flags and sanitize error responses.
4. Retest critical flows after remediation is deployed.

Sample Finding Detail

HIGH

AC-01 Broken Object Level Authorization

Affected Asset
https://portal.example.test/api/orders/{orderId}

Description
A low-privileged authenticated user can retrieve another customer's order details by changing the numeric order identifier in the request path.

Business Impact
Exposure of customer names, shipping addresses, and order history may result in privacy violations, reputational damage, and regulatory impact.

Proof of Concept
1. Log in as User A.
2. Request /api/orders/1042.
3. Change the identifier to /api/orders/1043.
4. Observe that data belonging to User B is returned without an authorization error.

Recommendation
Implement server-side ownership validation for every object access, avoid trusting client-supplied identifiers, and add regression tests for cross-account access attempts.

Retest Deliverable

After remediation, AegisCode Labs validates the fix, updates finding status, and provides closure notes with before/after evidence.