SKYLINE WEB AGENCY

Security Assessment Report

Comprehensive Web Application Penetration Test

Report ID	**SKY-SEC-2025-001**
Date	**March 2025**
Classification	**CONFIDENTIAL — SAMPLE REPORT**
Client	**[Anonymized E-Commerce Platform]**
Assessment Type	**Full Penetration Test**

This sample report demonstrates standard deliverable quality and testing methodology. All identifying information has been anonymized for demonstration purposes.

TABLE OF CONTENTS

01 — EXECUTIVE SUMMARY

A comprehensive security assessment of a modern e-commerce platform identified **15 vulnerabilities**, including **2 critical issues** requiring immediate remediation to prevent system compromise and data breach.

Severity	Count	Action Required
CRITICAL	2	Immediate (48 hours)
HIGH	4	Short-term (1-2 weeks)
MEDIUM	6	Scheduled (30 days)
LOW	3	Planned improvement

Immediate Risks

- Complete database compromise via SQL injection (Critical)
- Administrative privilege escalation via authentication bypass (Critical)

Engagement Details

Duration	**48 hours**
Methodology	**Automated scanning with manual validation**
Scope	**Web application, APIs, authentication systems**
Environment	**Staging (production-equivalent)**

02 — CRITICAL FINDINGS

Finding 1: SQL Injection — Database Compromise

Severity	**Critical (CVSS 9.8)**
Location	**/api/products/search**
Access Required	**None (unauthenticated)**

Description

Unsanitized user input is concatenated directly into SQL queries, enabling arbitrary command execution against the backend database. This vulnerability requires no authentication and can be exploited remotely by any attacker.

Proof of Concept

```
curl -X POST https://[redacted]/api/products/search \
-H "Content-Type: application/json" \
-d '{"search": "test' UNION SELECT email,password FROM users--"}'
```

Impact

- Full database access including all tables and schemas
- Exposure of 20,000+ customer records (PII, payment data)
- Administrative credential compromise

Remediation

Vulnerable:

```
const query = `SELECT * FROM products WHERE name LIKE '%${search}%'`;
db.execute(query);
```

Secure:

```
const query = `SELECT * FROM products WHERE name LIKE ?`;
db.execute(query, [`%${search}%`]);
```

Priority: Immediate — patch within 48 hours

Finding 2: Authentication Bypass — JWT Implementation Flaw

Severity	**Critical (CVSS 9.1)**
Location	**JWT token validation**
Complexity	**Low**

Description

The application accepts JWT tokens with "alg": "none", allowing attackers to forge administrative tokens without knowledge of the signing secret. This completely bypasses the authentication layer.

Proof of Concept

```
import jwt

payload = {
"userId": "1",
"email": "admin@[redacted]",
"role": "administrator"
}

forged = jwt.encode(payload, "", algorithm="none",
headers={"alg": "none", "typ": "JWT"})
```

Impact

- Complete authentication bypass for any user account
- Administrative privilege escalation
- Unauthorized data access and system modification

Remediation

Vulnerable:

```
const decoded = jwt.decode(token); // No verification
```

Secure:

```
const decoded = jwt.verify(token, process.env.JWT_SECRET, {
algorithms: ['HS256'],
issuer: 'your-app',
maxAge: '2h'
});
```

Priority: Immediate — patch within 48 hours

03 — HIGH SEVERITY FINDINGS

#	Finding	CVSS	Location
3	Stored XSS	**8.5**	Product review system
4	Insecure Direct Object Reference	**7.5**	/api/users/{id}
5	Server-Side Request Forgery	**8.6**	Image upload via URL
6	Weak Password Policy	**7.1**	User registration

Stored XSS (Finding 3)

User-submitted product reviews execute persistent scripts without output encoding, enabling session hijacking and credential theft across all visitors to affected pages. Widespread session hijacking is possible through this vector.

Insecure Direct Object Reference (Finding 4)

Users can access other users' data by modifying ID parameters in API requests. Server-side authorization checks are required on all data access endpoints.

Server-Side Request Forgery (Finding 5)

Unvalidated URL fetching in the image upload feature permits internal network access and potential cloud metadata exposure. Implement allow-lists and private IP blocking.

Weak Password Policy (Finding 6)

Current 6-character minimum enables brute-force attacks. Recommend enforcing 10+ character minimum with complexity requirements and implementing account lockout.

04 — MEDIUM SEVERITY FINDINGS

7 **Missing Rate Limiting**

Authentication endpoints lack rate limiting, enabling brute-force attacks against user accounts.

8 **Insecure Session Cookies**

Session cookies are missing the HttpOnly flag, making them accessible to JavaScript and vulnerable to XSS-based theft.

9 **Absent CSRF Protection**

State-changing operations lack Cross-Site Request Forgery tokens, enabling unauthorized actions on behalf of authenticated users.

10 **Verbose Error Messages**

Detailed error messages disclose system information including framework versions, file paths, and database structure.

11 **Missing Security Headers**

Critical headers including Content-Security-Policy, X-Frame-Options, and Strict-Transport-Security are not configured.

12 **Outdated Dependencies**

Multiple third-party libraries contain known vulnerabilities with published CVEs and available patches.

05 — REMEDIATION ROADMAP

Phase	Timeline	Actions
EMERGENCY	Week 1	Patch SQL injection and JWT bypass vulnerabilities. Deploy to production immediately after validation.
HIGH	Weeks 2-3	Remediate XSS, IDOR, and SSRF findings. Strengthen password policy and implement account lockout.
SYSTEMATIC	Month 2	Address all medium findings. Update dependencies. Conduct developer security training program.

06 — BUSINESS IMPACT ANALYSIS

Financial Exposure

- Regulatory fines up to EUR 20 million under GDPR
- Direct financial loss from fraud and unauthorized transactions
- Breach notification, legal counsel, and forensic investigation costs
- Operational downtime and system recovery expenses

Reputational Risk

- Customer trust erosion and user churn
- Brand damage and competitive disadvantage

Compliance Implications

- GDPR / CCPA data protection violations
- PCI DSS non-compliance for payment processing

07 — RECOMMENDATIONS

Immediate Actions

- Deploy critical patches for SQL injection and JWT bypass within 48 hours
- Implement temporary compensating controls (WAF rules, enhanced logging) where patches are delayed

Short-Term Improvements

- Establish continuous security monitoring and alerting
- Schedule regular vulnerability assessments (quarterly recommended)
- Conduct developer security awareness training

Strategic Initiatives

- Adopt Secure Development Lifecycle (SDLC) practices across all projects
- Implement routine third-party penetration testing
- Develop formal incident response plan and capabilities

08 — NEXT STEPS

1. Brief technical leadership on critical and high severity findings
2. Prioritize remediation backlog by business risk and exploitability
3. Schedule validation testing following remediation completion
4. Consider managed security monitoring for ongoing threat detection

09 — ASSESSMENT DETAILS

Methodology	Automated scanning with expert manual validation
Focus Areas	Web application, API, and authentication security
Specialization	Modern web frameworks, e-commerce platforms
Tools Used	Burp Suite, Nuclei, custom scripts, manual testing
Tester	Skyline Web Agency — Security Division

CONFIDENTIALITY NOTICE

This document contains sensitive security information. Distribution is limited to authorized personnel only. All identifying data has been anonymized for demonstration purposes. Unauthorized disclosure of the contents of this report may result in legal liability.

SKYLINE WEB AGENCY

Security Assessment Services