SKYLINE WEB AGENCY

Healthcare Security Assessment Report

White-Box Penetration Test with Source Code Review

Report ID	HSEC-ANON-2025-002
Date	March 2025
Classification	CONFIDENTIAL — SAMPLE REPORT
Industry	Healthcare Technology
Client	[Anonymized Healthcare SaaS Platform]
Assessment Type	White-Box 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 Healthcare SaaS platform identified **14 validated vulnerabilities**, including **2 critical issues** requiring immediate remediation to prevent unauthorized access to protected health information (PHI) and maintain HIPAA compliance.

Severity	Count	Action Required
CRITICAL	**2**	Immediate (24 hours)
HIGH	**5**	Short-term (1-2 weeks)
MEDIUM	**4**	Scheduled (30 days)
LOW	**3**	Planned improvement

Immediate Risks

- Complete patient database compromise via SQL injection (Critical)
- Unauthorized access to any patient record via authentication bypass (Critical)
- Mass PHI exposure through broken access controls

Engagement Details

Duration	**48 hours**
Methodology	**White-box testing with source code review and manual validation**
Scope	**Web application, patient portals, API endpoints, authentication systems**
Environment	**Staging with anonymized test data**

02 — CRITICAL FINDINGS

C1: SQL Injection — Patient Search

Severity	**Critical (CVSS 9.8)**
Location	**/api/patients/search**
Authentication	**Not required (public search)**
Complexity	**Low**

Description

Unsanitized user input is concatenated directly into SQL queries, enabling arbitrary command execution against the patient database. This vulnerability requires no authentication and can be exploited remotely to extract all patient records including protected health information.

Proof of Concept

```
curl -X POST https://[redacted]/api/patients/search \
-H "Content-Type: application/json" \
-d '{"search": "test' UNION SELECT patient_name,ssn,date_of_birth FROM patients--"}'
```

Data Exposed

- Patient names, SSNs/National IDs, dates of birth
- Medical record numbers, contact information, insurance details

Impact

- Complete patient database access
- Mass PHI compromise
- **Direct HIPAA violation**

Remediation

Vulnerable:

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

Secure:

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

Priority: Immediate — patch within 24 hours

C2: Authentication Bypass — JWT Implementation Flaw

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

Description

The application accepts JWT tokens with "alg": "none", allowing attackers to forge valid tokens and access any patient account without credentials. This completely bypasses the authentication layer for the patient portal.

Proof of Concept

```
import jwt

payload = {
"patient_id": "12345",
"role": "patient",
"email": "any.patient@example.com"
}

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

Impact

- Access to any patient's medical records, prescriptions, appointments, and billing information

Remediation

Vulnerable:

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

Secure:

```
try {
const decoded = jwt.verify(token, process.env.JWT_SECRET, {
algorithms: ['HS256']
});
req.user = decoded;
next();
} catch (error) {
return res.status(403).json({ error: 'Invalid token' });
}
```

Priority: Immediate — patch within 24 hours

03 — HIGH SEVERITY FINDINGS

ID	Finding	CVSS	Location
H1	Insecure Direct Object Reference	**7.5**	/api/patients/{patientId}/records
H2	Stored Cross-Site Scripting	**7.3**	Clinical notes
H3	Missing Rate Limiting	**7.3**	Authentication endpoints
H4	Weak Password Policy	**7.1**	Registration systems
H5	Insecure Session Cookies	**7.0**	Session management

H1: Insecure Direct Object Reference — Medical Records

Sequential patient IDs in URLs lack authorization verification. Patient 1001 can access Patient 1002's complete medical history by simply changing the ID parameter in the request. Server-side authorization checks must verify the requesting user owns or is authorized to access the requested resource.

H2: Stored XSS — Clinical Notes

Provider-entered clinical notes execute JavaScript in other providers' browsers when viewed. This enables session hijacking and unauthorized PHI access across provider accounts. All output must be HTML-encoded.

H3: Missing Rate Limiting

Authentication endpoints accept unlimited login attempts, enabling brute-force attacks against both patient and staff accounts. Implement a 5-attempt lockout per 15-minute window with progressive delays.

H4: Weak Password Policy

Current 6-character minimum with letters only permits easily guessed credentials. Enforce a minimum of 10 characters with complexity requirements including uppercase, lowercase, numbers, and special characters.

H5: Insecure Session Cookies

Session cookies are missing critical security flags, exposing sessions to theft via XSS and man-in-the-middle attacks. Implement the following cookie configuration:

```
Set-Cookie: sessionId=xxx; HttpOnly; Secure; SameSite=Strict; Max-Age=7200
```

04 — MEDIUM SEVERITY FINDINGS

M1 **Information Disclosure**

Verbose API error messages expose database schema, table names, and internal file paths to end users. Configure production error handling to return generic messages while logging full details server-side.

M2 **Missing Security Headers**

Critical security headers including Content-Security-Policy, X-Frame-Options, X-Content-Type-Options, and Strict-Transport-Security are not implemented, leaving the application vulnerable to clickjacking and content injection.

M3 **Outdated Dependencies**

Multiple npm packages (express, jsonwebtoken, lodash) contain known vulnerabilities with published CVEs and available patches. Implement automated dependency scanning in the CI/CD pipeline.

M4 **Predictable Reset Tokens**

Password reset tokens are generated using timestamp-based seeds, making them predictable. An attacker can calculate valid reset tokens and take over patient or staff accounts. Use cryptographically secure random token generation.

05 — LOW SEVERITY FINDINGS

L1 **HTML Comments Exposing Internal Endpoints**

HTML source contains developer comments referencing internal API endpoints and infrastructure details. Remove all comments from production builds.

L2 **Missing security.txt**

No /.well-known/security.txt file exists for responsible disclosure. Implement per RFC 9116 to provide security researchers a clear reporting channel.

L3 **Server Version Disclosure**

HTTP response headers expose web server version information, aiding attacker reconnaissance. Configure the server to suppress version headers in production.

06 — REMEDIATION ROADMAP

Phase	Timeline	Actions
EMERGENCY	**Week 1**	Patch SQL injection (C1) and JWT bypass (C2). Deploy WAF rules as temporary mitigation. Validate fixes in staging before production deployment.
HIGH	**Weeks 2-3**	Fix IDOR (H1) and XSS (H2). Implement rate limiting (H3). Strengthen password policy (H4). Secure session cookies (H5).
SYSTEMATIC	**Month 2**	Update dependencies. Add security headers. Fix error disclosure. Secure reset tokens. Clean HTML comments. Add security.txt. Address all low findings.

07 — HIPAA COMPLIANCE IMPACT

Security Rule	Status	Finding
§164.312(a) — Access Control	AT RISK	JWT bypass enables unauthorized PHI access
§164.312(c) — Integrity	AT RISK	SQL injection permits data modification
§164.312(e) — Transmission Security	AT RISK	Missing HTTPS enforcement mechanisms
§164.308(a) — Risk Analysis	UPDATE	New critical risks identified requiring assessment update

COMPLIANCE NOTICE: Remediation of the identified critical and high severity vulnerabilities is required to maintain HIPAA compliance. Failure to address these findings may result in regulatory penalties up to $1.5 million per violation category per year under the HITECH Act.

08 — TESTING METHODOLOGY

Approach

- White-box assessment with full source code access
- Automated scanning with manual validation
- Proof-of-concept exploitation to confirm impact
- Patient data flow analysis

Coverage

- SQL injection and XSS testing across all input vectors
- Authentication and authorization bypass testing
- JWT security implementation review
- IDOR testing of all object references
- Session management and API security validation

Total Duration	**48 hours**
Tools Used	**Burp Suite, Nuclei, SonarQube, custom scripts, manual testing**
Standard	**OWASP Testing Guide v4.2, PTES, HIPAA Security Rule**

09 — CONTACT

Conducted By	**Skyline Web Agency — Security Division**
Email	**security@skylinewebagency.com**
Website	**skylinewebagency.com**

CONFIDENTIALITY NOTICE

SKYLINE WEB AGENCY

Security Assessment Services