OWASP TOP 10 (2021) COMPLIANCE ASSESSMENT
Security Testing of OWASP Juice Shop
Assessed by: Ishan Acharya
Assessment Date: December 1, 2025

═══════════════════════════════════════════════════════════

INTRODUCTION
To enhance my cybersecurity training and develop practical penetration testing skills, I carried out an extensive security evaluation of the OWASP Juice Shop application. This document details my findings, aligned with the OWASP Top 10 (2021) framework.
The aim of this assessment was to uncover real-world vulnerabilities and see how they fit within industry-standard security frameworks. Through hands-on testing, I identified several critical security issues that illustrate common web application vulnerabilities.

═══════════════════════════════════════════════════════════

- A01:2021 - BROKEN ACCESS CONTROL

What I Found:
During my testing, I discovered that the admin panel at /administration was accessible without any proper authorization checks. This was concerning because:

My Testing Process:
1. I logged in as a regular user
2. Navigated directly to the /administration endpoint
3. To my surprise, I gained full access to the admin panel
4. I could view all registered user emails and account details

Impact on Real-World Applications: This vulnerability would allow any authenticated user to access sensitive administrative functions. In a production environment, this could lead to data breaches, unauthorized modifications, and complete system compromise.

What I Learned:
This taught me the importance of implementing role-based access control (RBAC) and always verifying user permissions on the server side - not just hiding UI elements.

Remediation Priority: IMMEDIATE
CVSS Score: 8.8 (Critical)

---

- A02:2021 - CRYPTOGRAPHIC FAILURES

What I Discovered:
While exploring the application, I found that the /ftp directory was publicly accessible without any authentication. This directory contained several sensitive files that shouldn't be exposed.

Files I Could Access:
• acquisitions.md - Contains confidential business acquisition plans
• coupons_2013.md.bak - A backup file with potential promotional data
• package.json.bak - Reveals the application's technology stack
• legal.md - Internal legal documents
• suspicious_errors.yml - System error logs

My Thoughts:
I was surprised how easy it was to access this information. In a real company, these files could contain trade secrets, customer data, or system vulnerabilities that attackers could exploit.

Lesson Learned:
Never leave sensitive directories publicly accessible. Always implement authentication, remove backup files from production, and store confidential documents outside the web root.

Remediation Priority: IMMEDIATE
CVSS Score: 7.5 (High)

---

- A03:2021 - INJECTION

My Most Interesting Finding:
This was one of the most exciting discoveries during my assessment. I found two critical injection vulnerabilities:

1.SQL Injection (Authentication Bypass):

What I Did:
•Went to the login page
•Instead of entering a normal email, I tried: admin@juice-sh.op' OR 1=1--
•Used any random password
•The application let me in as the admin user!

Why This Worked:
The application wasn't properly sanitizing my input. My malicious SQL code manipulated the login query to always return true, bypassing the authentication entirely.

Real-World Impact: This could allow attackers to:
- Access any user account without knowing passwords
- Steal sensitive data from the database
- Modify or delete critical information
- Potentially gain complete control of the system

CVSS: 9.8 (Critical)

2. Cross-Site Scripting (XSS):

What I Tested:
• Used the search functionality
• Entered a malicious payload: <iframe src="javascript:alert('XSS')"></iframe>
• The payload was executed and reflected back on the page

Why This Matters:
While this might seem less severe than SQL injection, XSS can be used to steal user sessions, redirect users to phishing sites, or even install malware.

CVSS: 7.3 (High)

Key Takeaway:
Never trust user input! Always validate and sanitize data on both client and server sides. Use parameterized queries for database operations and implement proper output encoding.

Remediation Priority: IMMEDIATE

---

- A04:2021 - INSECURE DESIGN

Status: Not Tested in This Assessment

Reflection: While I focused on implementation vulnerabilities, I recognize that secure design is equally important. This would require a comprehensive architecture review and threat modeling session, which I plan to explore in future assessments.

What I'd Like to Learn:
• How to conduct threat modeling

• Secure design patterns for web applications
• How to evaluate security from the design phase

---

- A05:2021 - SECURITY MISCONFIGURATION

An Interesting Discovery:
When I tried accessing restricted files, the application showed me detailed error messages with full stack traces. While this was helpful for my testing, it's a significant security risk.

What Was Exposed:
• Express.js version (4.21.0)
• Internal file system paths
• Function names and code structure
• Full error stack traces

My Analysis:
This information is like giving attackers a roadmap of the application's internal structure. It helps them identify known vulnerabilities and plan targeted attacks.

Personal Note:
As someone learning cybersecurity, these error messages actually helped me understand the application better - which is exactly why they're dangerous in production!

Remediation Priority: SHORT-TERM
CVSS Score: 5.3 (Medium)

---

- A06:2021 - VULNERABLE AND OUTDATED COMPONENTS

What I Noticed:
The application disclosed its version information (Express 4.21.0), which made me curious about potential known vulnerabilities.

My Recommendation:
Iwould suggest running a full dependency audit using tools like:
•npm audit for Node.js applications
•OWASP Dependency-Check
•Snyk or similar vulnerability scanners

Next Steps I'd Take:
1. Check for CVEs against identified versions
2. Update all dependencies to latest secure versions
3. Implement automated dependency scanning in CI/CD pipeline

Status: Partial Assessment - Needs Further Investigation

---

- A07:2021 - IDENTIFICATION AND AUTHENTICATION FAILURES

Connected to My SQL Injection Finding:
The SQL injection vulnerability I discovered completely bypasses the authentication system. Additionally, I noticed:

• No rate limiting on login attempts (I could try unlimited passwords)
• No multi-factor authentication options
• Session tokens didn't appear to have proper expiration

Real-World Scenario:
In a production application, weak authentication allows attackers to use brute force attacks, credential stuffing, and session hijacking.

Remediation Priority: IMMEDIATE

---

- A08:2021 - SOFTWARE AND DATA INTEGRITY FAILURES

Status: Not Tested

My Reasoning:
Testing this category properly would require access to the CI/CD pipeline and update mechanisms. As an external assessment, I didn't have visibility into these processes.

Future Work:
Ina real engagement, I would want to examine:
•How the application receives and validates updates
•Integrity checks for downloaded dependencies
•Code signing and verification processes

---

- A09:2021 - SECURITY LOGGING AND MONITORING FAILURES

Status: Not Tested

Limitation of This Assessment:
I didn't have access to the application's logging infrastructure to verify if my testing activities were being properly logged and monitored.

Questions I'd Ask in a Real Audit:
• Are failed login attempts being logged?
• Is there alerting for suspicious activities?
• How long are logs retained?
• Who has access to security logs?

---

- A10:2021 - SERVER-SIDE REQUEST FORGERY (SSRF)

Status: Not Tested

My Observation:
During my assessment, I didn't encounter functionality that would be vulnerable to SSRF (like URL fetching, image processing from URLs, or webhook callbacks).

Note: This doesn't mean the vulnerability doesn't exist - it just means I didn't find features that typically exhibit SSRF issues.

---

PERSONAL REFLECTION AND LEARNING OUTCOMES

This Assessment Taught Me:

1. Hands-On Experience is Invaluable
Reading about vulnerabilities in textbooks is one thing, but actually exploiting them gave me a much deeper understanding of web application security.

2. Security is Multilayered
One vulnerability often leads to discovering others. The SQL injection not only bypassed authentication but also could have enabled data exfiltration.

3. Documentation Matters
Creating this detailed report helped me organize my findings and think like a professional security consultant.

4. Ethical Responsibility
Having the ability to find these vulnerabilities comes with the responsibility to report them properly and never use them maliciously.

Skills I Developed:

- Manual penetration testing techniques
- Understanding of OWASP Top 10 framework
- Professional security reporting
- SQL injection and XSS exploitation
- Access control testing
- Security documentation

---

SUMMARY OF FINDINGS

Out of the 10 OWASP categories I assessed:

- Found Vulnerabilities in: 5 categories
- Critical Issues: 3 findings
- High Severity: 2 findings
- Medium Severity: 1 finding
- Not Tested: 4 categories
- Partial Assessment: 1 category

Overall Security Posture: CRITICAL - Immediate Action Required

---

MY RECOMMENDATIONS (Prioritized)

Immediate Actions (This Week):
1. Fix the SQL injection in the login form - this is the most critical issue
2. Implement proper access control for the admin panel
3. Remove or protect the /ftp directory
4. Add input validation and output encoding for XSS prevention

Short-Term Goals (Next Month):
1. Disable detailed error messages in production
2. Implement Content Security Policy headers
3. Add rate limiting on authentication endpoints
4. Conduct code review for similar vulnerabilities
5. Set up security logging and monitoring

Long-Term Strategy (Next Quarter): 1. Provide security training for the development team 2. Implement automated security testing in CI/CD 3. Schedule regular penetration tests 4. Establish secure coding standards 5. Deploy a Web Application Firewall (WAF)

---

CONCLUSION

This hands-on security assessment of OWASP Juice Shop was an incredibly valuable learning experience for me. I successfully identified multiple critical vulnerabilities that demonstrate real-world security risks.

The most concerning findings were:
• SQL Injection allowing complete authentication bypass
• Broken access control exposing sensitive admin functions
• Publicly accessible confidential files

As a cybersecurity student, this project helped me understand not just how to find vulnerabilities, but also how to assess their impact and communicate findings professionally.

I'm grateful for the opportunity to work with OWASP Juice Shop - a safe, legal environment to practice these skills. The experience has motivated me to continue developing my penetration testing abilities and pursue a career in cybersecurity.

Next Steps for My Learning:
• Practice with more vulnerable applications
• Learn advanced exploitation techniques
• Study remediation and secure coding practices
• Prepare for certifications like CEH or OSCP

---

Assessed by: Ishan Acharya
Date: December 4, 2025
Target: OWASP Juice Shop (https://juice-shop.herokuapp.com)
Framework: OWASP Top 10 (2021)
Assessment Type: Educational Penetration Testing