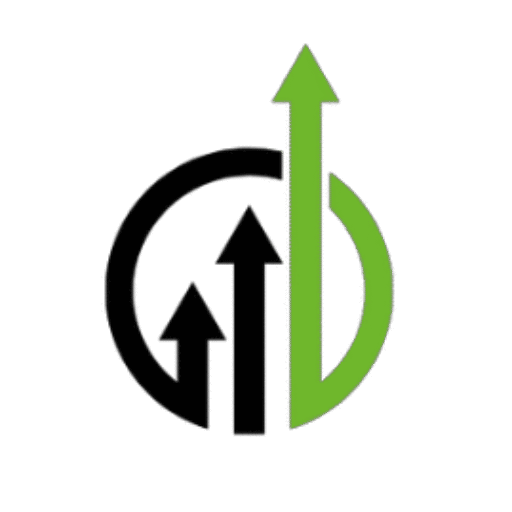
Web Application Security Assessment Report



**Program:** Future Interns Cyber Security Program

**Task 1:** Web Application Security Testing

**Name:** Ilegbemi Damilola Olumide

**Date:** August 2025

**Tools Used:** OWASP ZAP 2.16.1, OWASP Juice Shop, Kali Linux

# Executive Summary

This assessment tested the OWASP Juice Shop application using OWASP ZAP. The goal was to identify common web application vulnerabilities and map them to the OWASP Top 10. The scan revealed several medium-risk findings, mostly related to security misconfigurations, missing headers, and outdated libraries. These issues, if exploited, could allow attackers to steal data, hijack sessions, or compromise the application.

# Methodology

The assessment followed a structured methodology combining automated scanning with manual verification:  
  
1. Target Application: The assessment was conducted against the publicly hosted instance of OWASP Juice Shop at https://juice-shop.herokuapp.com. This is an intentionally insecure application widely used for learning.  
2.Tool Used: OWASP ZAP version 2.16.1 was employed to perform passive and active scanning of the application. ZAP is an open-source tool maintained by OWASP and designed to find common security flaws.  
3. Approach: An automated scan was initiated to crawl the application and identify potential vulnerabilities. The results were then analyzed, and key findings were selected for detailed documentation. Each issue was mapped to the OWASP Top 10 framework for clarity.  
4. Framework Reference: Findings were aligned with the OWASP Top 10 (2021) – the industry standard for categorizing and prioritizing web application risks.

# Findings

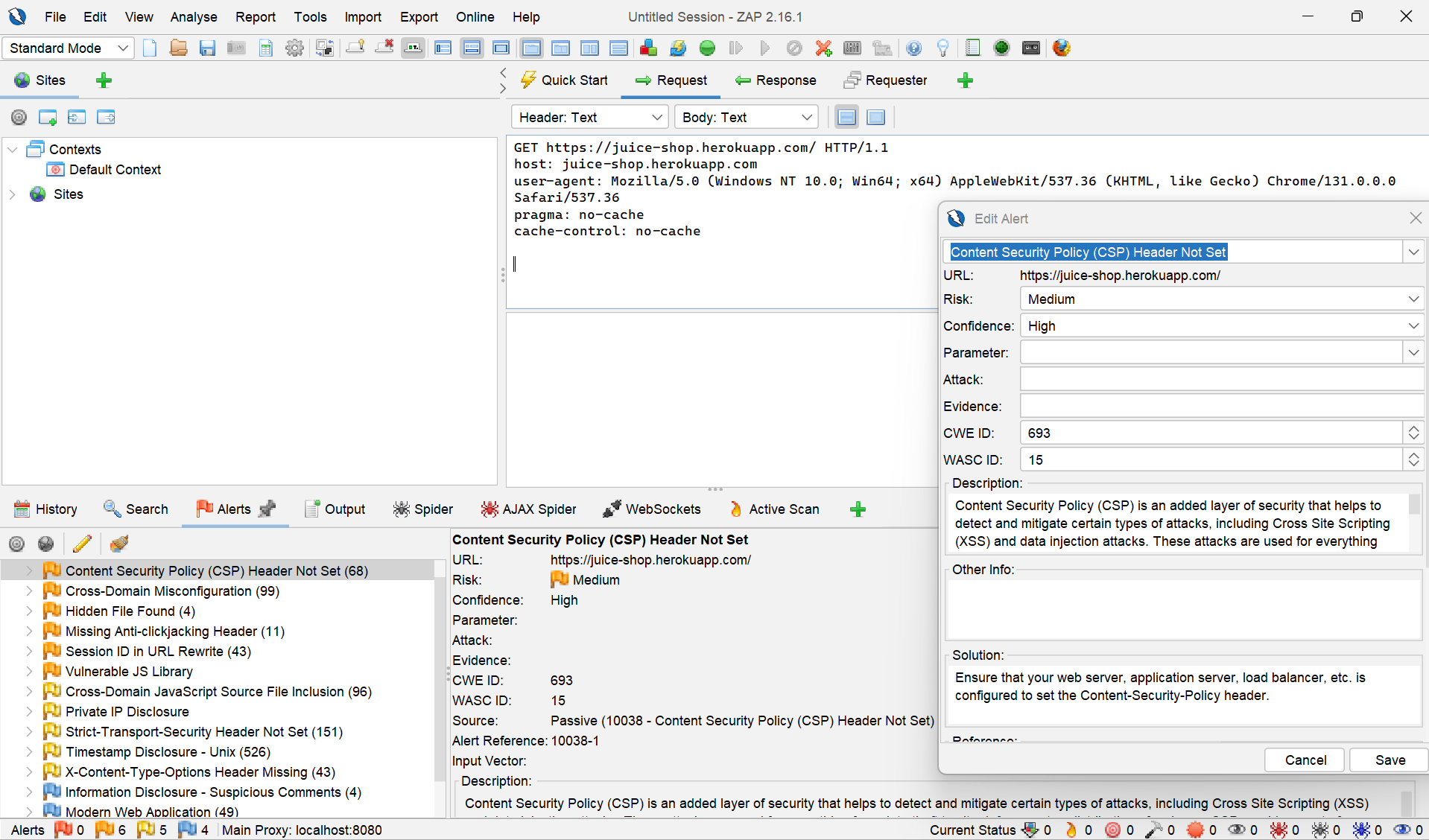
## 1. Content Security Policy (CSP) Header Not Set

Severity: Medium

Description: The application does not implement a Content Security Policy (CSP) header. This weakens protection against cross-site scripting (XSS) and other code injection attacks.

Proof: ZAP flagged missing CSP header on the homepage.

Recommendation: Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.



## 2. Cross-Domain Misconfiguration (CORS)

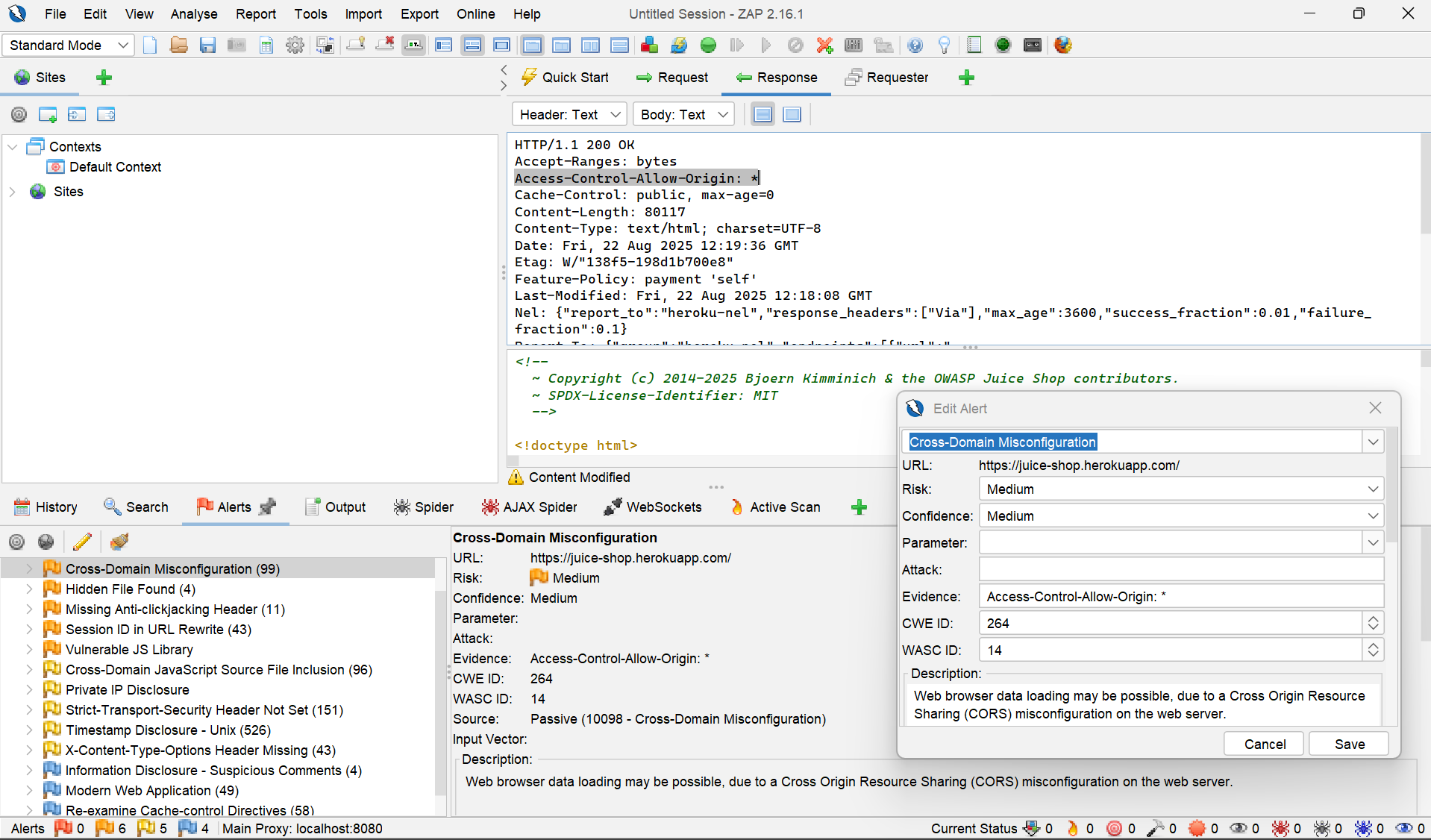
Severity: Medium

Description: The CORS policy is overly permissive, allowing requests from any origin. This can enable malicious sites to perform unauthorized actions on behalf of users.

Proof: ZAP identified overly permissive CORS headers.

Recommendation: Ensure that sensitive data is not available in an unauthenticated manner (using IP address white-listing, for instance).

Configure the "Access-Control-Allow-Origin" HTTP header to a more restrictive set of domains, or remove all CORS headers entirely, to allow the web browser to enforce the Same Origin Policy (SOP) in a more restrictive manner.



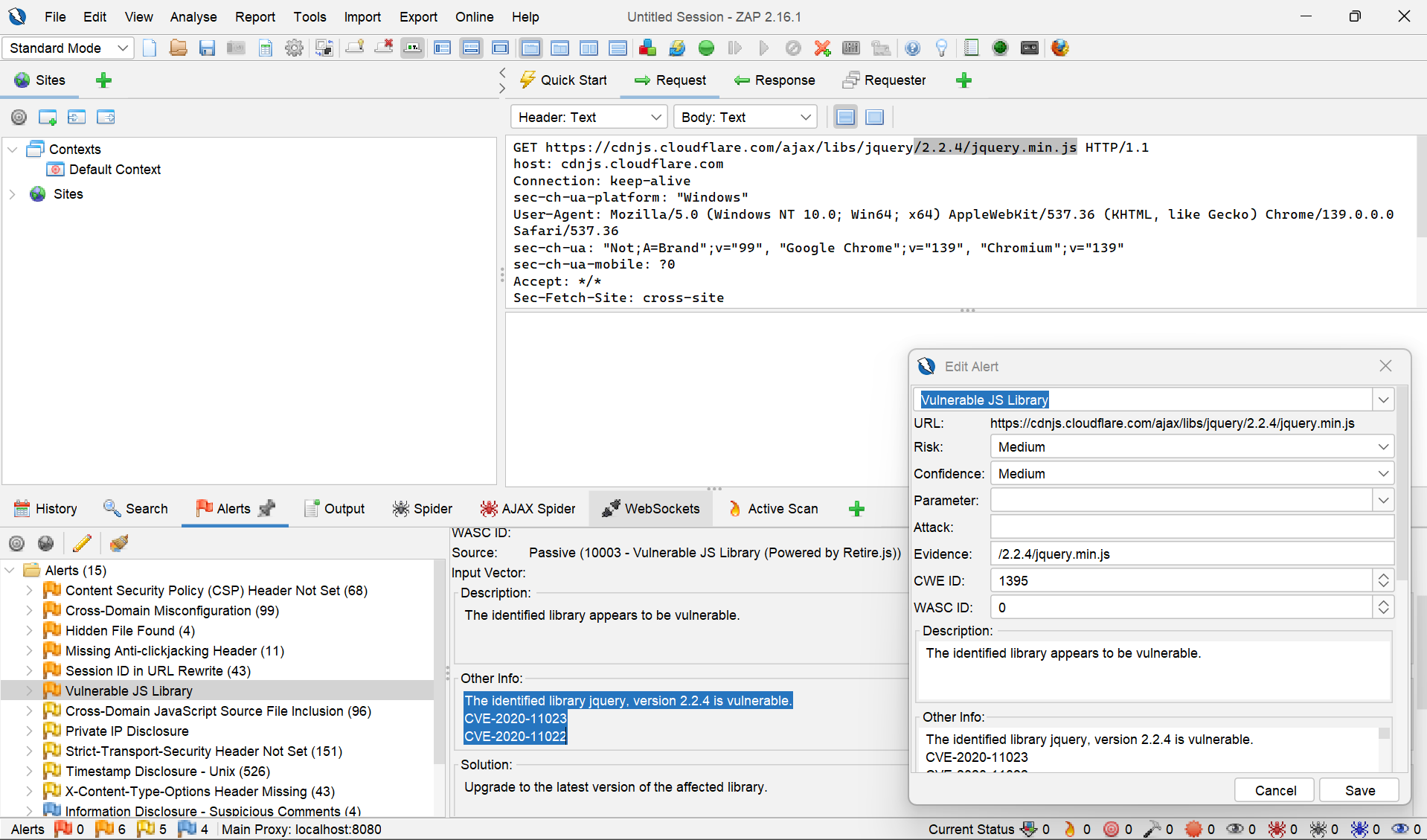
## 3. Vulnerable JavaScript Library

Severity: Medium

Description: The application loads jQuery v2.2.4, which contains known vulnerabilities. Outdated libraries can be exploited to perform XSS, code injection, or denial of service.

Proof: ZAP flagged the outdated jQuery library from cdnjs.cloudflare.com.

Recommendation: Update to the latest stable version of jQuery. Remove unused libraries.



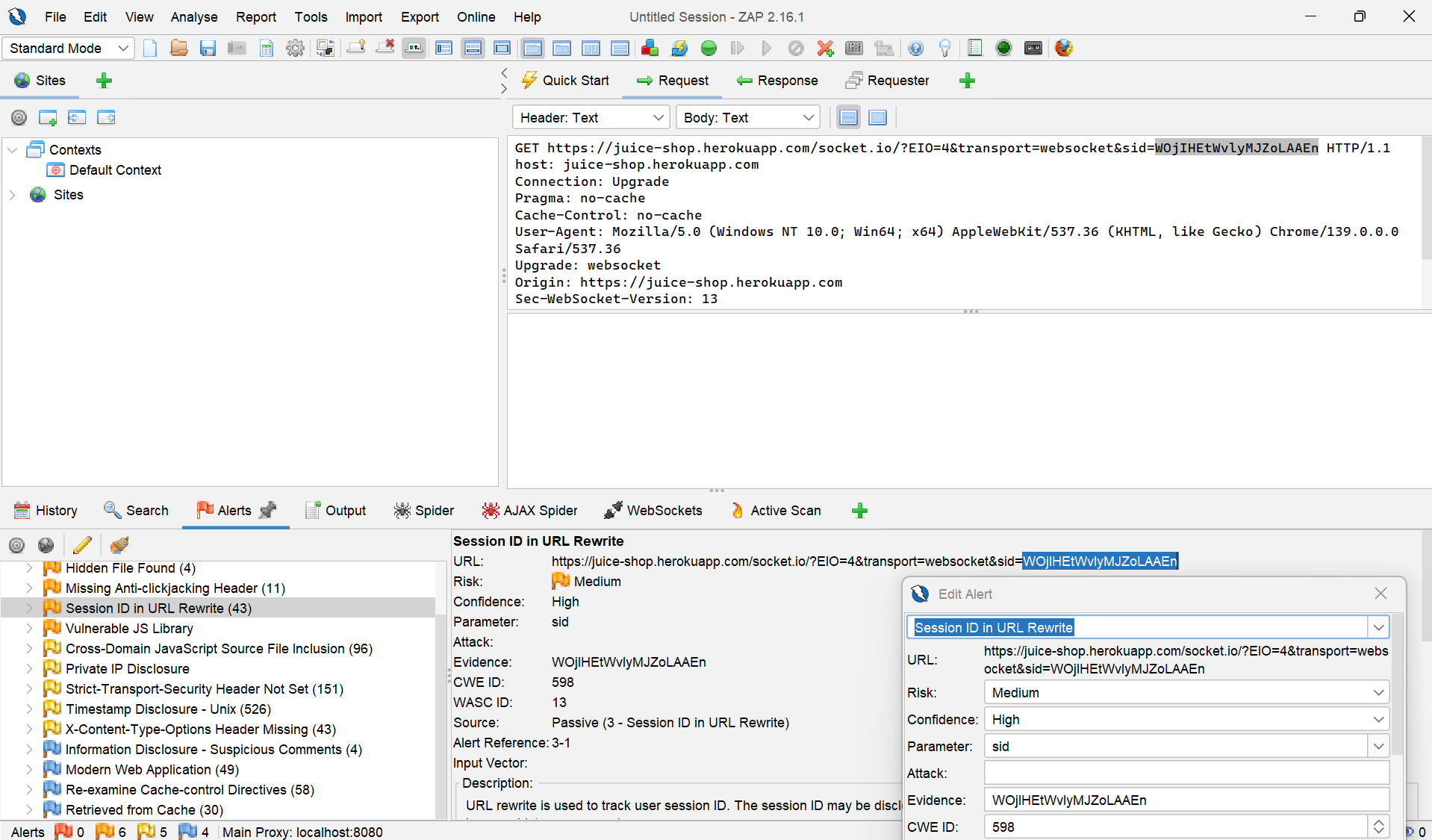
## 4. Session ID in URL Rewrite

Severity: Medium

Description: Session identifiers were observed in the URL. This makes them visible in logs and browser history, which can be intercepted by attackers.

Proof: ZAP flagged session ID in socket.io requests.

Recommendation: For secure content, put session ID in a cookie. To be even more secure consider using a combination of cookie and URL rewrite.



## 5. Missing Anti-Clickjacking Header

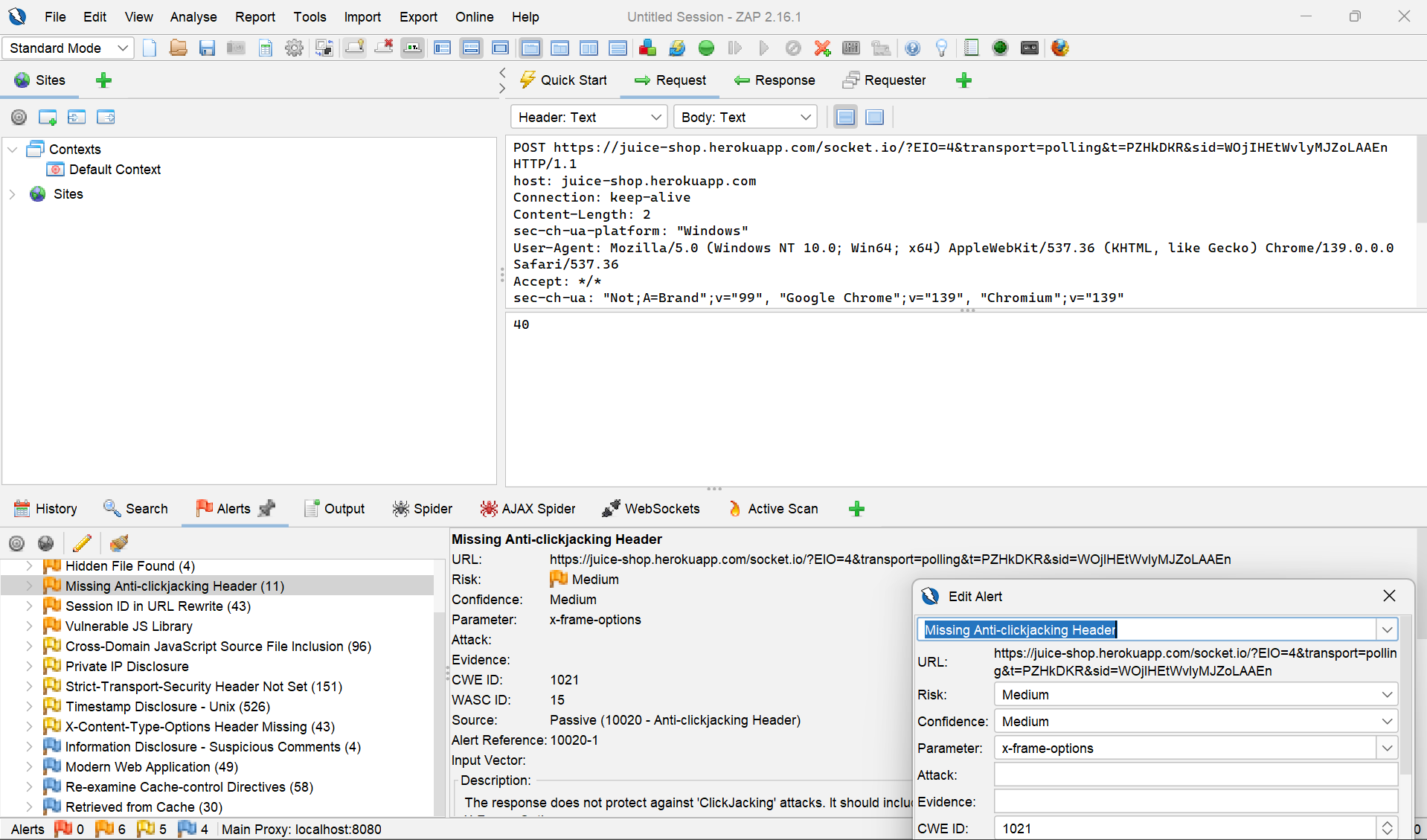
Severity: Medium

Description: The application does not implement the X-Frame-Options header, which protects against clickjacking attacks.

Proof: ZAP flagged missing header in socket.io responses.

Recommendation:Modern Web browsers support the Content-Security-Policy and X-Frame-Options HTTP headers. Ensure one of them is set on all web pages returned by your site/app.

If you expect the page to be framed only by pages on your server (e.g. it's part of a FRAMESET) then you'll want to use SAMEORIGIN, otherwise if you never expect the page to be framed, you should use DENY. Alternatively consider implementing Content Security Policy's "frame-ancestors" directive.



# OWASP Top 10 Mapping

• CSP Header Missing → A05: Security Misconfiguration  
• Cross-Domain Misconfiguration → A05: Security Misconfiguration  
• Vulnerable JS Library → A06: Vulnerable and Outdated Components  
• Session ID in URL → A07: Identification and Authentication Failures  
• Anti-Clickjacking Header Missing → A05: Security Misconfiguration

# Conclusion

The ZAP scan revealed several medium-severity vulnerabilities in the OWASP Juice Shop application. Most issues are related to missing or misconfigured security headers and outdated components. Addressing these findings would improve the security posture of the application and reduce the risk of exploitation by attackers.