VULNERABILITY ASSESSMENT REPORT

Target Website: **OWASP Juice Shop**

Tool Used: **OWASP ZAP (Passive Scan)**

Internship Program: **Future Interns – Cyber Security**

Task: **Task 1**

Prepared By**: Sohel Mujawar**

Role: **Cyber Security Intern**

Date: **04/02/2026**

About the Task

- Every business today relies on a website to represent its brand and services. However, many websites operate with security misconfigurations such as missing security headers, outdated components, or unintentional information exposure.
- Most clients do not ask for hacking or exploitation. Instead, they seek clear answers to important business questions such as whether their website is secure, what risks exist, and which issues should be fixed first.
- This task focuses on identifying common security weaknesses in a professional and ethical manner. The objective is to provide clarity, assess risk, and deliver actionable recommendations, similar to how a real-world security consultant would communicate findings to a business client.

Introduction

- Web applications are frequently exposed to security risks due to misconfigurations and insecure design.
- This report presents a vulnerability assessment performed on a deliberately vulnerable web application using OWASP ZAP.

Objective

The objective of this assessment is to identify common web application vulnerabilities, classify their risk levels, and suggest appropriate remediation measures.

The objective also includes presenting security findings in a business-friendly manner, avoiding excessive technical jargon, and prioritizing risks based on their potential impact on the organization.

The goal is to support informed decision-making rather than technical exploitation.

Scope of Testing

The assessment was limited to non-intrusive testing of the OWASP Juice Shop application using passive scanning and manual exploration.

No intrusive or exploitative testing was performed.

Tools Used

- OWASP ZAP
- Web Browser
- Browser DevTools
- Canva used for designing a clean and professional vulnerability assessment report

Scope & Ethics

This assessment was conducted by strictly following ethical security testing practices.

Allowed Activities:

- Analysis of publicly accessible web pages only
- Passive vulnerability scanning
- HTTP security header inspection
- Configuration and response analysis

Not Allowed Activities:

- Authentication bypass attempts
- Exploitation of vulnerabilities
- Brute force attacks
- Denial-of-Service (DoS) attacks
- Any activity that could disrupt or harm the target website

The assessment was performed from the perspective of a security auditor, focusing on risk identification and mitigation rather than offensive exploitation.

Methodology

The assessment process began with selecting a publicly accessible test website. Passive scanning was performed using OWASP ZAP to identify potential security misconfigurations without actively attacking the application. Manual exploration was conducted using a web browser and developer tools to inspect HTTP headers and client-side behavior.

All identified alerts were reviewed, categorized based on risk severity, and analyzed to understand their potential impact. No exploitation or intrusive testing was carried out during the assessment.

Assessment Approach

The assessment followed a structured and ethical approach focused on identifying common security weaknesses without impacting the availability or integrity of the target application. Emphasis was placed on configuration analysis, security header inspection, and passive observation of application behavior.

This approach ensures accurate risk identification while maintaining compliance with responsible security testing practices. All findings were documented based on observed behavior rather than exploitation.

Target Website Details

- Website Name: OWASP Juice Shop
- Purpose: An intentionally vulnerable web application designed for security training and testing.

Testing Context

OWASP Juice Shop was selected as the target application because it is intentionally designed to simulate real-world security issues in a controlled and legal environment.

This allows security testing techniques to be practiced safely while maintaining realistic assessment conditions.

The findings from this assessment are representative of common issues that may also be present in real-world business websites if proper security configurations are not implemented.

Vulnerability Summary

The following table provides a summary of the identified vulnerabilities and their associated risk levels to give a quick overview of the website's security posture.

Vulnerability Name	Risk Level
Cross-Domain Misconfiguration	Medium
CSP Header Not Set	Medium
Missing Anti-clickjacking Header	Medium
X-Content-Type-Options Missing	Low
Information Disclosure	Informational

Summary of Findings

The vulnerability assessment identified multiple security misconfigurations primarily related to missing or improperly configured HTTP security headers.

The majority of the findings fall under the medium-risk category, indicating areas that require attention but do not represent an immediate threat to the application.

Low and informational findings were also observed, mainly associated with security hardening and information exposure.

While these issues do not directly result in system compromise, they may increase the application's attack surface if left unaddressed.

Overall, the identified vulnerabilities highlight the need for improved security configuration and adherence to industry best practices.

Addressing these issues will help enhance the application's security posture, reduce potential client-side risks, and strengthen user trust.

Detailed Vulnerability Findings

1. Cross-Domain Misconfiguration

Risk Level:
Medium

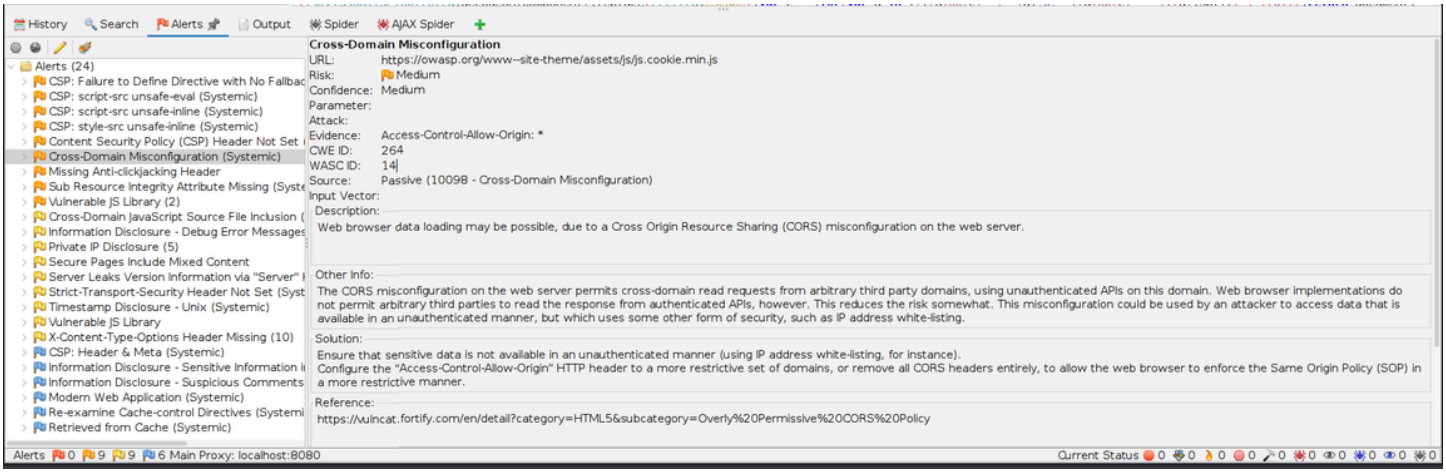
Affected URL:
https://owasp.org/www--site-theme/assets/js/js.cookie.min.js

Description:
Web browser data loading may be possible, due to a Cross Origin Resource Sharing (CORS) misconfiguration on the web server

Impact:
This issue may affect user trust and data integrity if exploited.

Evidence:
Access-Control-Allow-Origin: *

Remediation Steps:
Ensure that sensitive data is not available in an unauthenticated manner (using IP address whitelisting, 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.



2. Content Security Policy (CSP) Header Not Set

Risk Level:
Medium

Affected URL:
https://owasp.org/www--site-theme/assets/js/js.cookie.min.js

Description:
Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including Cross-Site-Scripting (XSS) and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page covered types are Javascript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files

Impact:
This issue may negatively impact user trust and website credibility.

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

3. Missing Anti-clickjacking Header

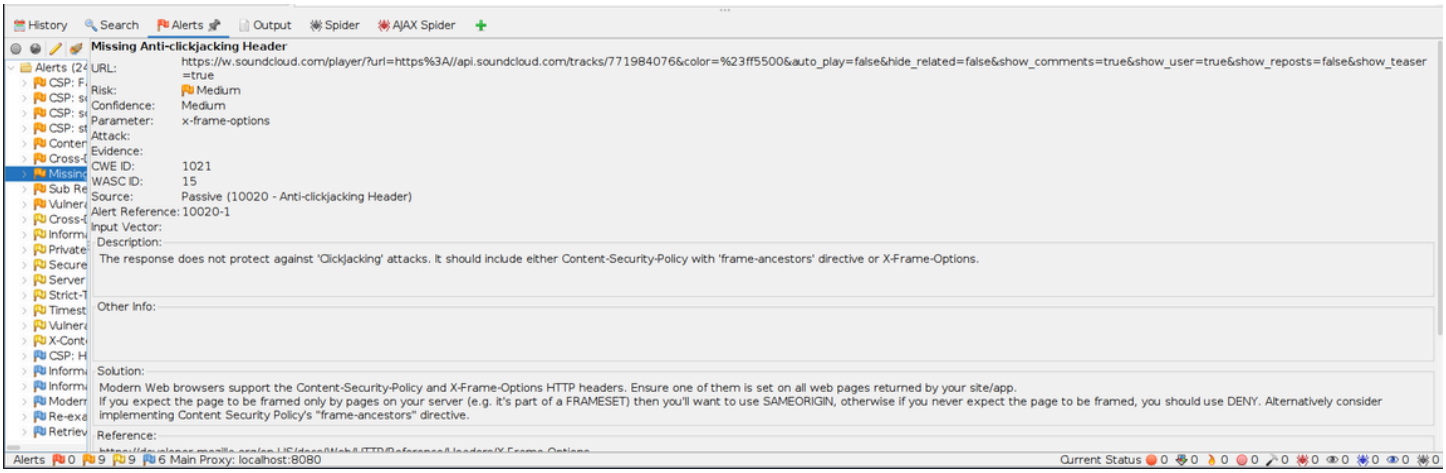
Risk Level:
Medium

Affected URL:
https://w.soundcloud.com/player/?url=https://api.soundcloud.com/tracks/771984076&color=%23ff5500&auto_play=false&hide_related=false&show_comments=true&show_user=true&show_reposts=false&show_teaser=true

Description:
The response does not protect against 'Clickjacking' attacks. It should include either Content-Security-Policy with 'frame-ancestors' directive or X-Frame Options.

Impact:
This vulnerability could expose the organization to compliance and reputational risks.

Remediation Steps:
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.



4. X-Content-Type-Options Header Missing

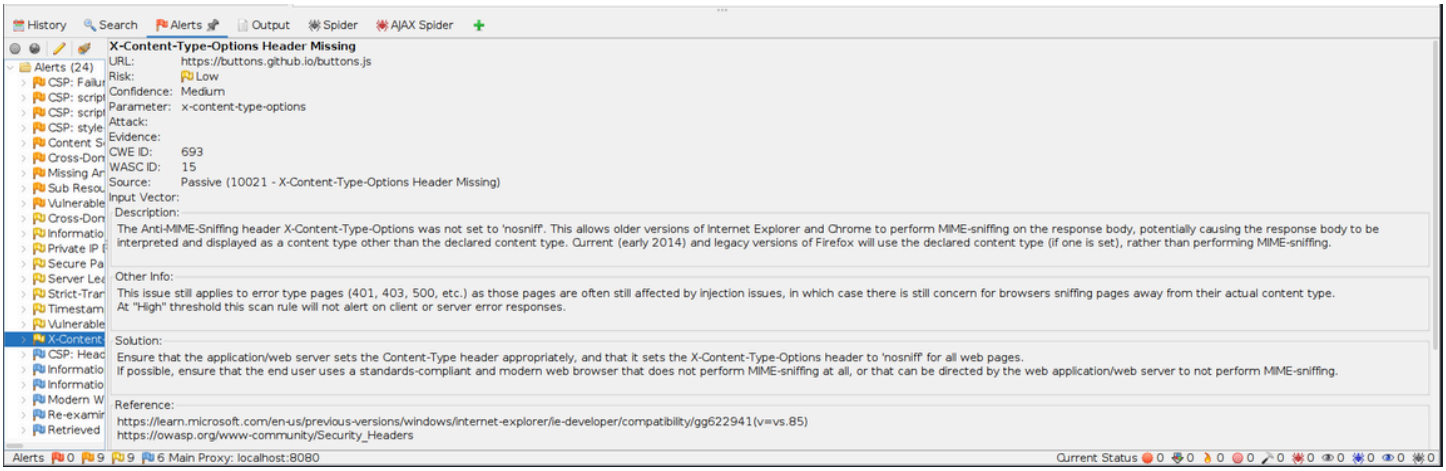
Risk Level:
Low

Affected URL:
https://buttons.github.io/buttons.js

Description:
The Anti-MIME-Sniffing header X-Content-Type-Options was not set to ‘nosniff’. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.

Impact:
If left unresolved, this issue may increase the overall attack surface of the application.

Remediation Steps:
Ensure that the application/web server sets the Content-Type header appropriately and that it sets the X-Content-Type-Options header to ‘nosniff’ for all web pages. If possible, ensure that the end user uses a standard-compliant and modern web browser that does not perform MIME-sniffing at all or that can be directed by the web application/web server to not perform MIME-sniffing.



5. Information Disclosure—Sensitive Information in URL

Risk Level:
Informational

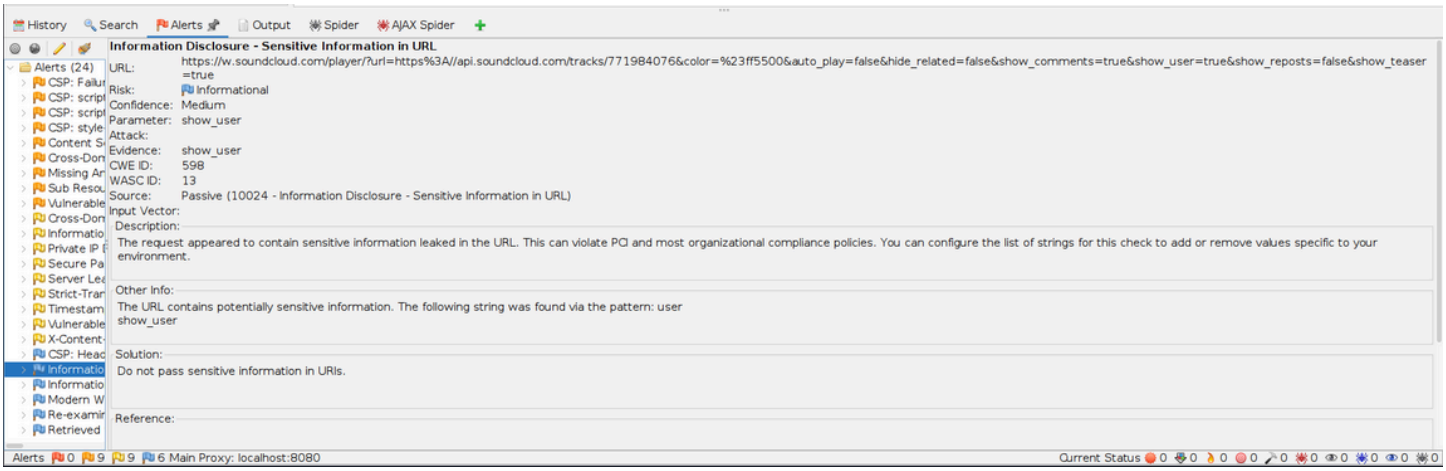
Affected URL:
https://w.soundcloud.com/player/?url=https://api.soundcloud.com/tracks/771984076&color=%23ff5500&auto_play=false&hide_related=false&show_comments=true&show_user=true&show_reposts=false&show_teaser=true

Description:
The request appeared to contain sensitive information leaked in the URL. This can violate PCI and most organizational compliance policies. You can configure the list of strings for this check to add or remove values specific to your environment

Evidence:
show_user

Impact:
This may lead to privacy and compliance concerns if sensitive parameters are exposed.

Remediation Steps:
Do not pass sensitive information in URIs



Conclusion

This vulnerability assessment identified several security misconfigurations primarily related to missing or improperly configured security headers. No high-risk or critical vulnerabilities were observed during the assessment.

Although the identified issues do not indicate immediate compromise, they may increase the risk of client-side attacks and information exposure if left unaddressed. Implementing the recommended remediation steps will improve the overall security posture of the application and align it with industry best practices.

The assessment was limited to passive and non-intrusive testing and should be periodically repeated to ensure continued security as the application evolves.

GitHub Repository Reference

All assessment documentation, screenshots, and the final report have been documented in a public GitHub repository.

Repository Name: FUTURE_CS_01
Repository Link: *https://github.com/sohelmj16/FUTURE_CS_01*

The repository includes the report PDF, supporting evidence, and a README file describing the tested website, scope, and tools used.