ASSIGNMENT 5

Web Application Vulnerability Assessment Using OWASP ZAP and OWASP Juice Shop

1. Objective

The objective of this assignment was to gain hands-on experience in web application security by performing a full vulnerability scan using **OWASP ZAP**, documenting findings and remediation steps on **GitHub**, and reflecting on lessons learned from the exercise.

2. Lab Environment

- Operating System: Kali Linux
- Target Application: OWASP Juice Shop
- Security Tool: OWASP ZAP
- Browser: Firefox ESR
- Target URL:http://127.0.0.1:42000

3. Step-by-Step Procedure

Step 1: Install and Run OWASP Juice Shop

OWASP Juice Shop was used as the target because it is an intentionally vulnerable web application designed for security training.

Juice Shop was launched locally and confirmed accessible via browser at:

```
http://127.0.0.1:42000
```

Step 2: Install OWASP ZAP

OWASP ZAP was installed using the APT package manager to avoid snap-related issues.

```
sudo apt update
sudo apt install zaproxy -y
```

ZAP was launched using:

```
zaproxy
```

Step 3: Configure Browser Proxy

To allow ZAP to intercept web traffic:

1. Open Firefox

2. Navigate to **Settings → Network Settings → Settings**
3. Select **Manual proxy configuration**
4. Configure:
    - HTTP Proxy: `127.0.0.1`
    - Port: `8080`
    - Use this proxy for all protocols
5. Save settings

Step 4: Verify Traffic Interception

With the proxy configured, Juice Shop was accessed via:

`http://127.0.0.1:42000`

OWASP ZAP successfully captured HTTP requests, confirmed by the appearance of the target under the **Sites** tree.

Step 5: Add Target to OWASP ZAP

1. In ZAP, navigate to **Quick Start**
2. Select **Manual Explore**
3. Enter the target URL:

   `http://127.0.0.1:42000`

4. Launch browser

Step 6: Spider (Crawl) the Application

To discover application endpoints:

1. Right-click on the target in the **Sites** tree
2. Select **Attack → Spider**
3. Start the scan

Since OWASP Juice Shop is a **single-page Angular application**, manual interaction (clicking links, logging in, browsing products) was performed to ensure complete coverage.

Step 7: Passive Scanning

While browsing the application:

- ZAP performed **passive scanning**
- Identified security misconfigurations such as missing headers and information disclosure
- No intrusive attacks were executed at this stage

Step 8: Active Scanning

To identify exploitable vulnerabilities:

1. Right-click the target URL
2. Select **Attack → Active Scan**
3. Use default scan policy
4. Start scan

ZAP tested for:

- Cross-Site Scripting (XSS)
- Security misconfigurations
- Vulnerable libraries
- Session management weaknesses

Step 9: Review Scan Alerts

After the scan completed:

- Navigate to the **Alerts** tab
- A total of **13 alerts** were identified
- Findings were categorized as:
    - Medium Risk
    - Low Risk
    - Informational

Examples included:

- Missing Content Security Policy
- Vulnerable JavaScript Libraries
- Session ID exposed in URLs
- Missing anti-clickjacking headers

4. Exporting the OWASP ZAP Report

Step-by-Step: Export ZAP Scan Report

1. In OWASP ZAP, click **Report** (top menu)
2. Select **Generate Report**
3. Choose:
    - Report Type: **HTML** (recommended for GitHub)

4. Select:
    - Title: *OWASP ZAP Juice Shop Scan*
5. Choose destination folder
6. Click **Generate Report**

The report was saved as:

`zap-report.html`

PS: VIEW APPENDIX FOR ALL PROCEDURAL SCREENSHOTS

Findings and Remediation Notes

Target Application: OWASP Juice Shop
Scanning Tool: OWASP ZAP
Scan Type: Automated Active Scan
Target URL: http://127.0.0.1:42000

1. CSP: Failure to Define Content Security Policy

Description

The application does not define a **Content Security Policy (CSP)** header. CSP is a critical security control that helps prevent Cross-Site Scripting (XSS), data injection, and other code execution attacks by restricting the sources from which content can be loaded.

Risk

Without CSP, attackers can inject and execute malicious scripts, leading to:

- Session hijacking
- Credential theft
- Defacement or malware injection

Evidence

OWASP ZAP detected missing or improperly configured CSP headers in HTTP responses.

Remediation

- Define a strict `Content-Security-Policy` HTTP header.
- Restrict script, style, image, and frame sources to trusted domains only.

Example:

```
Content-Security-Policy: default-src 'self'; script-src 'self';
```

2. Missing Anti-Clickjacking Header

Description

The application does not include the `X-Frame-Options` or `frame-ancestors` directive, making it vulnerable to **clickjacking attacks**.

Risk

Attackers can embed the application inside an invisible iframe to trick users into performing unintended actions.

Remediation

- Add the `X-Frame-Options` header or CSP `frame-ancestors`.

Example:

```
X-Frame-Options: DENY
```

or

```
Content-Security-Policy: frame-ancestors 'none';
```

3. Session ID in URL Rewrite

Description

Session identifiers were observed in URLs. This practice exposes session tokens through browser history, logs, and referrer headers.

Risk

- Session hijacking
- Replay attacks
- Information leakage

Remediation

- Store session identifiers in **secure cookies**, not URLs.
- Ensure cookies use:
    - `HttpOnly`
    - `Secure`
    - `SameSite` attributes

4. Vulnerable JavaScript Library

Description

OWASP ZAP identified JavaScript libraries with known vulnerabilities.

Risk

Known vulnerable libraries can be exploited to:

- Bypass security controls
- Execute malicious scripts
- Perform client-side attacks

Remediation

- Identify outdated libraries.
- Upgrade to the latest secure versions.
- Regularly audit dependencies using tools like:
    - `npm audit`
    - OWASP Dependency-Check

5. Cross-Domain JavaScript Source Inclusion

Description

The application loads JavaScript resources from external or cross-domain sources without sufficient validation.

Risk

If a third-party domain is compromised, malicious scripts may be served to users.

Remediation

- Avoid unnecessary third-party scripts.
- Use Subresource Integrity (SRI).

Example:

```
<script src="example.js" integrity="sha384-..."
crossorigin="anonymous"></script>
```

6. Application Error Disclosure

Description

Detailed error messages are exposed to users.

Risk

Error messages may reveal:

- Stack traces
- File paths
- Framework versions

This information can help attackers plan targeted attacks.

Remediation

- Disable detailed error messages in production.
- Log errors server-side only.
- Display generic error messages to users.

7. Information Disclosure Issues

Description

Multiple informational alerts were detected, including:

- Private IP address disclosure
- Timestamp disclosure
- Server or framework information exposure

Risk

While low risk individually, combined disclosures can help attackers fingerprint the system.

Remediation

- Remove unnecessary debug information.
- Mask internal IP addresses.
- Minimize response headers.

8. X-Content-Type-Options Header Missing

Description

The `X-Content-Type-Options` header is not set.

Risk

Browsers may perform MIME-type sniffing, which can lead to XSS attacks.

Remediation

Add the following header:

```
X-Content-Type-Options: nosniff
```

9. Modern Web Application Observation

Description

ZAP identified the target as a modern JavaScript-based web application (SPA).

Notes

This is informational and not a vulnerability. However, it highlights the need for:

- Proper API security
- Token protection
- Strong client-side controls

Overall Risk Summary

Severity	Count
High	0
Medium	Multiple
Low	Several
Informational	Multiple

The application demonstrates common security misconfigurations typical of intentionally vulnerable applications like OWASP Juice Shop.

Conclusion

The OWASP ZAP scan of OWASP Juice Shop revealed multiple security issues primarily related to **missing security headers, information disclosure, and insecure configurations**. While Juice Shop is intentionally vulnerable for learning purposes, this exercise demonstrates how automated scanners identify real-world web application risks and emphasizes the importance of secure defaults, proper configuration, and regular security testing.

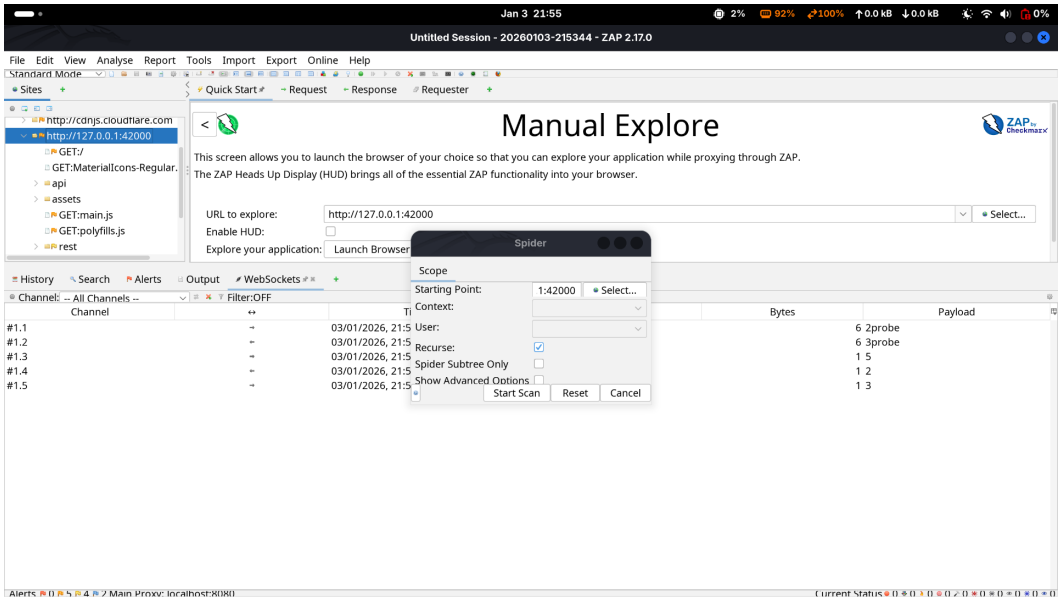
Reflection and Lessons Learned

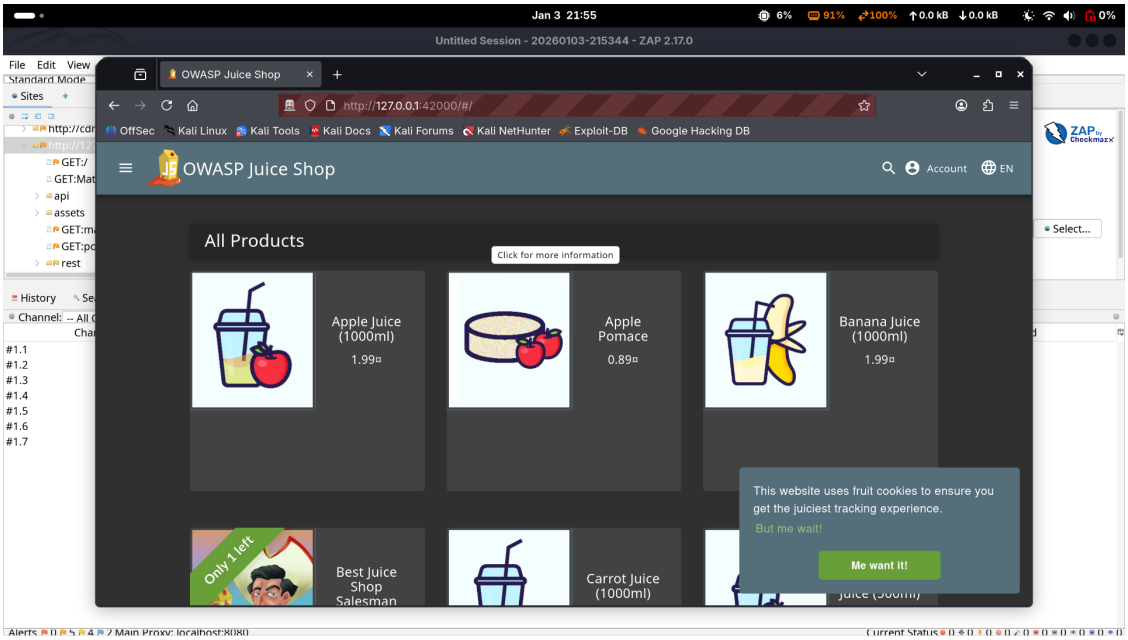
This exercise demonstrated the effectiveness of automated security scanning tools in identifying common web application vulnerabilities. It also highlighted the limitations of automation, especially when scanning modern single-page applications, making manual interaction essential. The assignment strengthened practical skills in vulnerability assessment, risk analysis, and secure configuration.

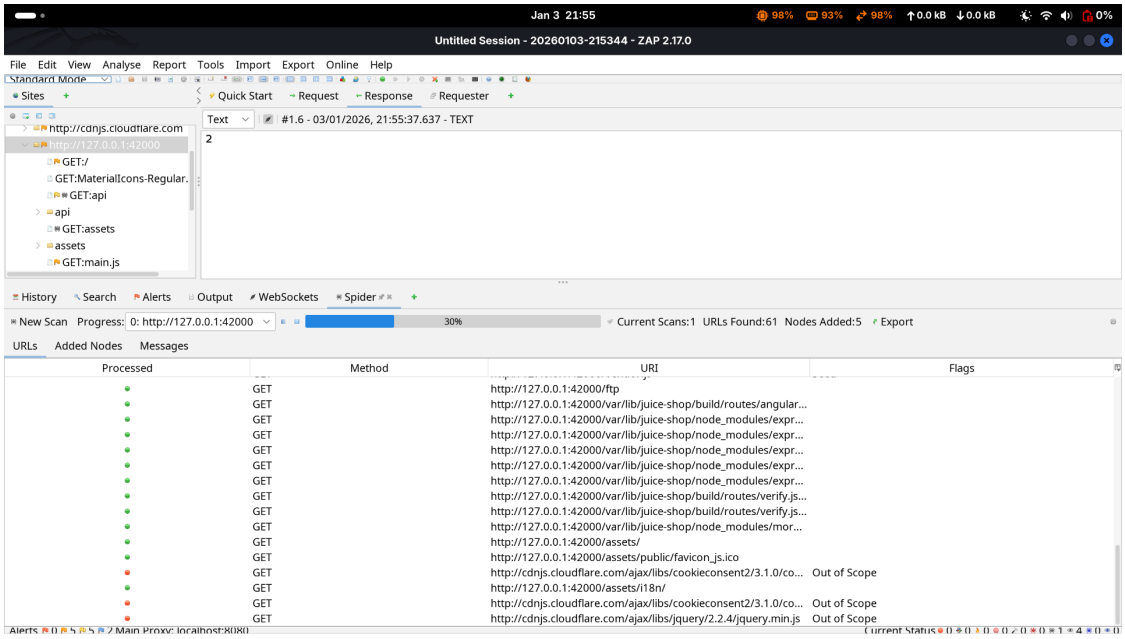
Disclaimer

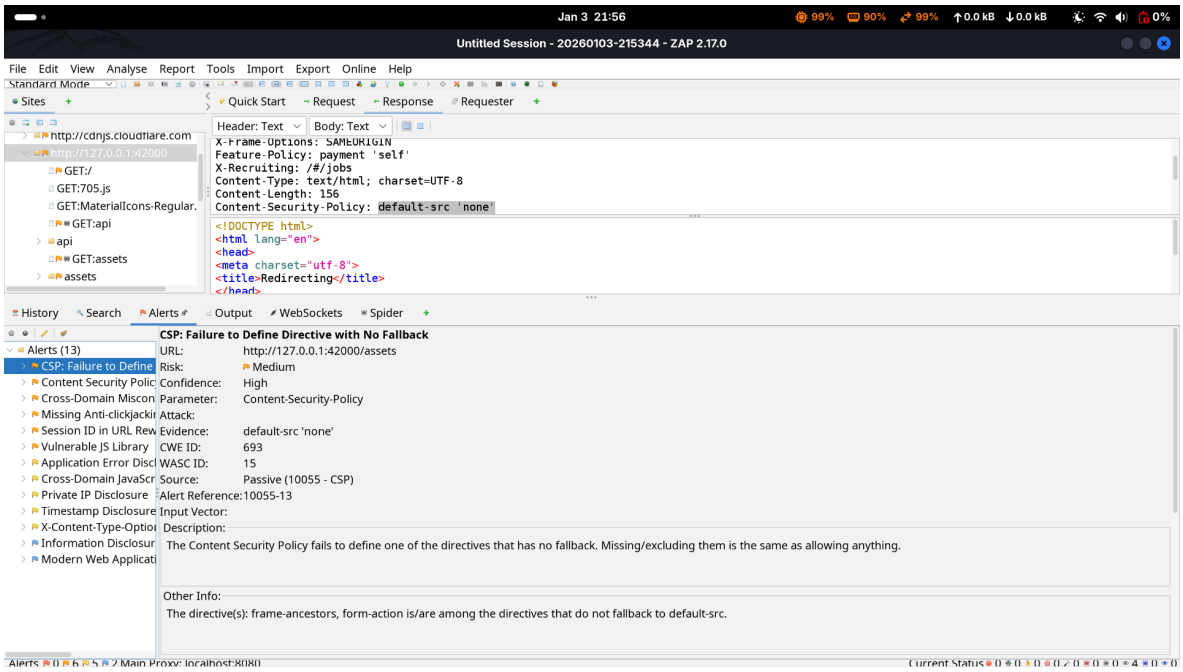
All testing was conducted on **OWASP Juice Shop**, an intentionally vulnerable application, within a controlled and authorized lab environment. These techniques must not be used on production systems without permission.

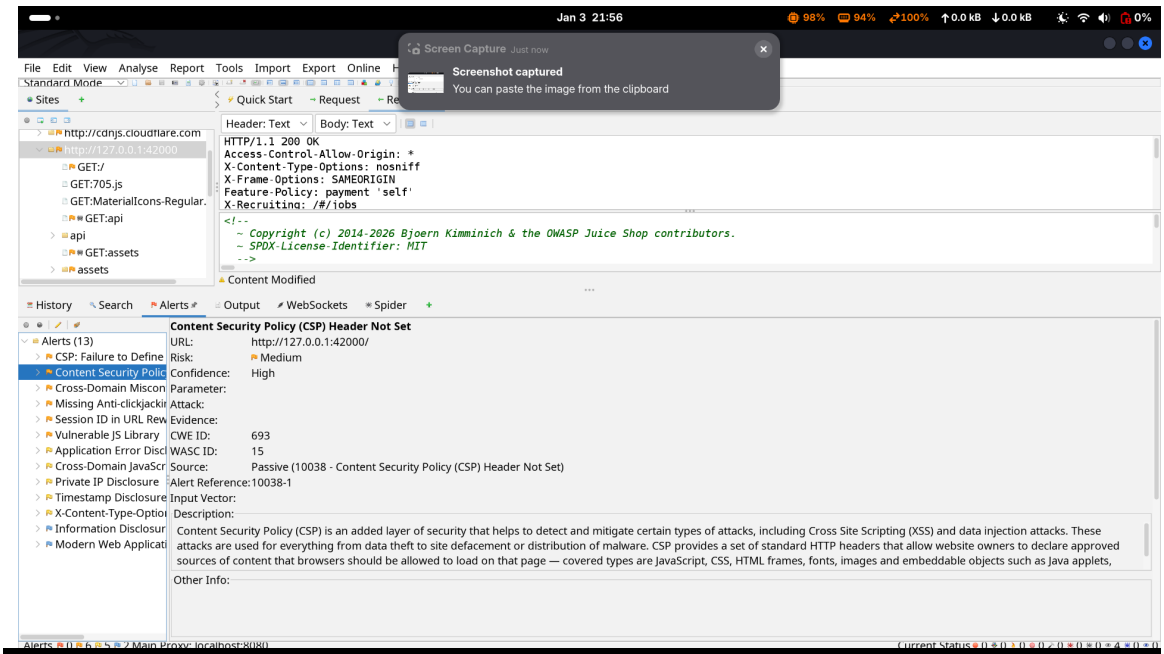
APPENDIX

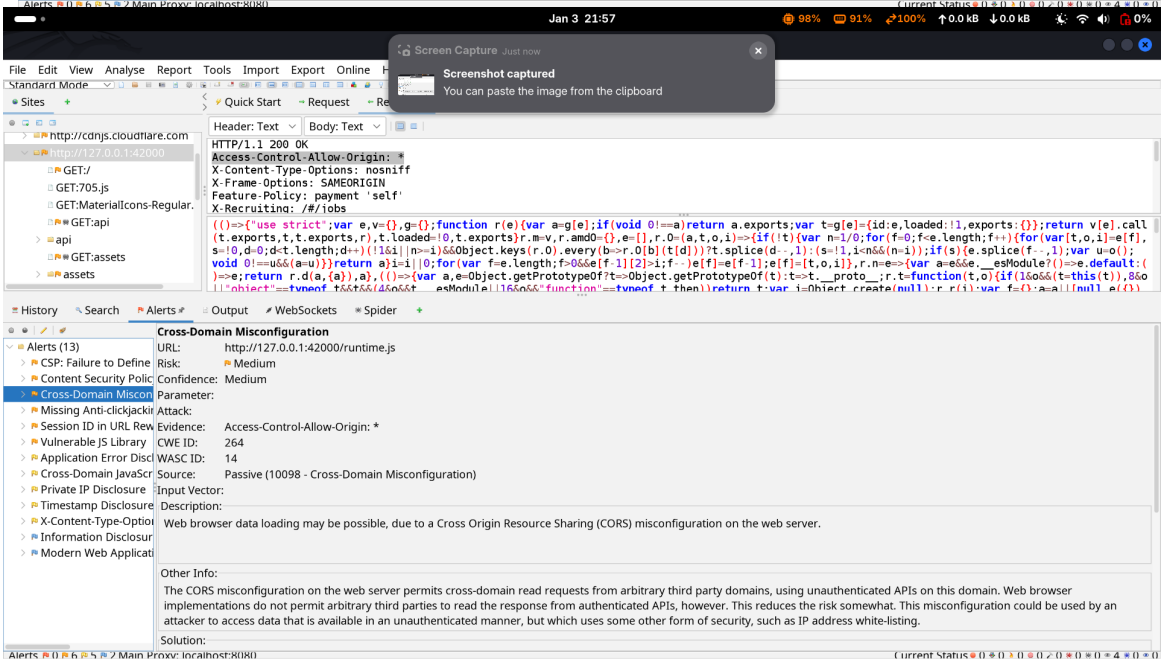

Untitled Session - 20260103-215344 - ZAP 2.17.0
Manual Explore
This screen allows you to launch the browser of your choice so that you can explore your application while proxying through ZAP.
The ZAP Heads Up Display (HUD) brings all of the essential ZAP functionality into your browser.
URL to explore:
http://127.0.0.1:42000
Enable HUD:
Explore your application:
Launch Browser
Spider
Scope
Starting Point:
Context:
User:
Recurse:
Spider Subtree Only
Show Advanced Options
Start Scan
Reset
Cancel

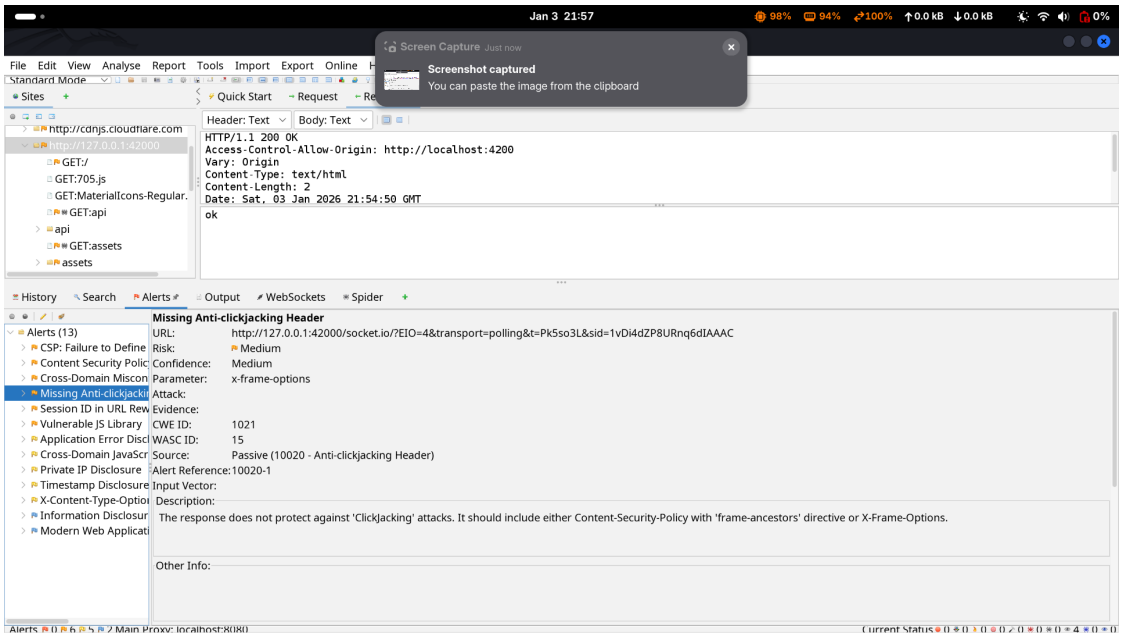

Untitled Session - 20260103-215344 - ZAP 2.17.0
OWASP Juice Shop
http://127.0.0.1:42000/#/
OWASP Juice Shop
Account
EN
All Products
Click for more information
Apple Juice (1000ml)
1.99¤
Apple Pomace
0.89¤
Banana Juice (1000ml)
1.99¤
Only 1 left
Best Juice Shop Salesman
Carrot Juice (1000ml)
This website uses fruit cookies to ensure you get the juiciest tracking experience.
But me wait!
Me want it!

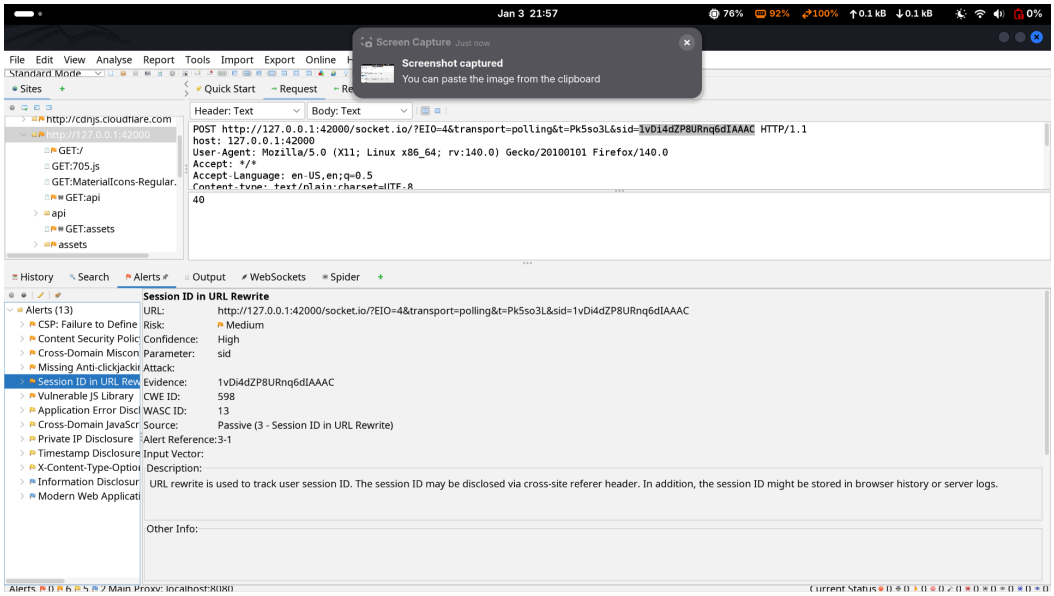

Untitled Session - 20260103-215344 - ZAP 2.17.0
#1.6 - 03/01/2026, 21:55:37.637 - TEXT
2
New Scan
Progress:
0: http://127.0.0.1:42000
30%
Current Scans:1
URLs Found:61
Nodes Added:5
Export
URLs
Added Nodes
Messages
Processed
Method
URI
Flags
GET http://127.0.0.1:42000/ftp
GET http://127.0.0.1:42000/var/lib/juice-shop/build/routes/angular...
GET http://127.0.0.1:42000/var/lib/juice-shop/node_modules/expr...
GET http://127.0.0.1:42000/var/lib/juice-shop/node_modules/expr...
GET http://127.0.0.1:42000/var/lib/juice-shop/node_modules/expr...
GET http://127.0.0.1:42000/var/lib/juice-shop/node_modules/expr...
GET http://127.0.0.1:42000/var/lib/juice-shop/node_modules/expr...
GET http://127.0.0.1:42000/var/lib/juice-shop/build/routes/verify.js...
GET http://127.0.0.1:42000/var/lib/juice-shop/build/routes/verify.js...
GET http://127.0.0.1:42000/var/lib/juice-shop/node_modules/mor...
GET http://127.0.0.1:42000/assets/
GET http://127.0.0.1:42000/assets/public/favicon_js.ico
GET http://cdnjs.cloudflare.com/ajax/libs/cookieconsent2/3.1.0/co... Out of Scope
GET http://127.0.0.1:42000/assets/i18n/
GET http://cdnjs.cloudflare.com/ajax/libs/cookieconsent2/3.1.0/co... Out of Scope
GET http://cdnjs.cloudflare.com/ajax/libs/jquery/2.2.4/jquery.min.js Out of Scope

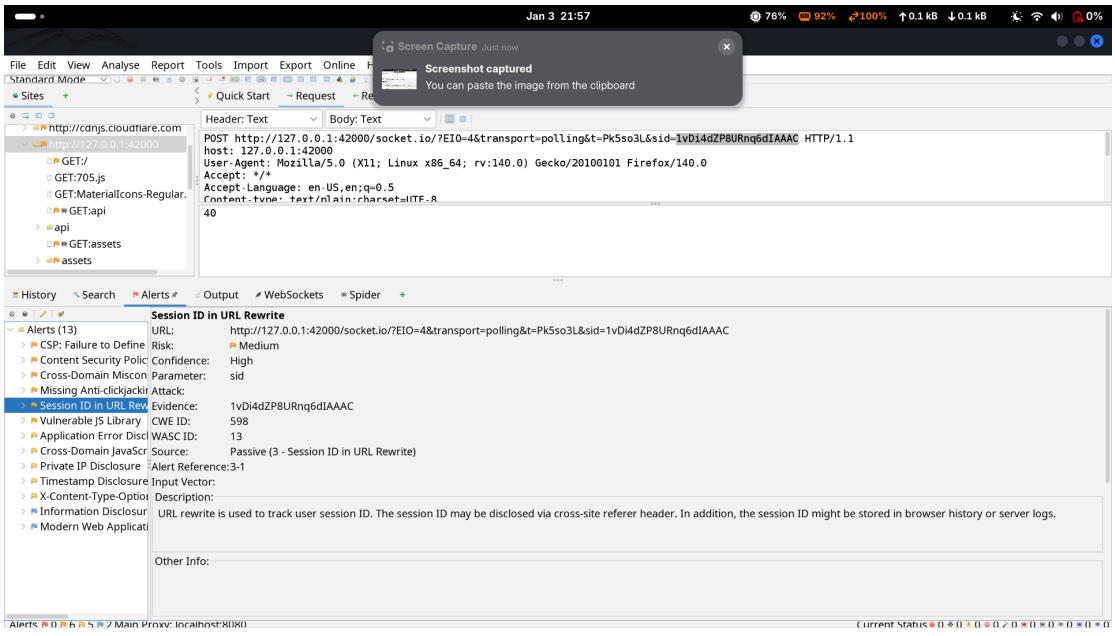

Jan 3 21:56
Untitled Session - 20260103-215344 - ZAP 2.17.0
File Edit View Analyse Report Tools Import Export Online Help
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Content-Type: text/html; charset=UTF-8
Content-Length: 156
Content-Security-Policy: default-src 'none'
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Redirecting</title>
CSP: Failure to Define Directive with No Fallback
URL: http://127.0.0.1:42000/assets
Risk: Medium
Confidence: High
Parameter: Content-Security-Policy
Attack:
Evidence: default-src 'none'
CWE ID: 693
WASC ID: 15
Source: Passive (10055 - CSP)
Alert Reference: 10055-13
Input Vector:
Description:
The Content Security Policy fails to define one of the directives that has no fallback. Missing/excluding them is the same as allowing anything.
Other Info:
The directive(s): frame-ancestors, form-action is/are among the directives that do not fallback to default-src.

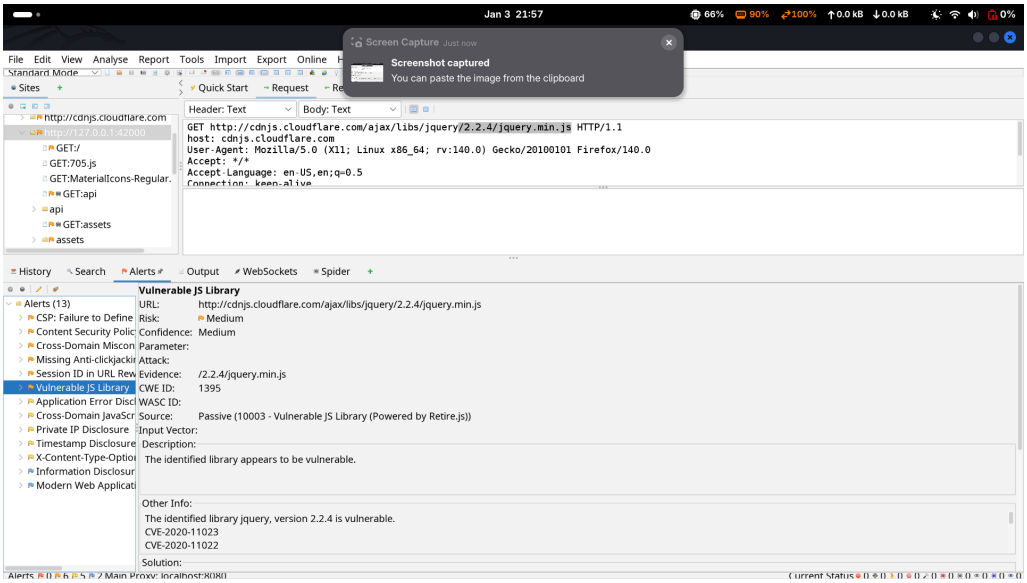

Jan 3 21:56
Screen Capture Just now
Screenshot captured
You can paste the image from the clipboard
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
<!--
~ Copyright (c) 2014-2026 Bjoern Kimminich & the OWASP Juice Shop contributors.
~ SPDX-License-Identifier: MIT
-->
Content Modified
Content Security Policy (CSP) Header Not Set
URL: http://127.0.0.1:42000/
Risk: Medium
Confidence: High
Parameter:
Attack:
Evidence:
CWE ID: 693
WASC ID: 15
Source: Passive (10038 - Content Security Policy (CSP) Header Not Set)
Alert Reference: 10038-1
Input Vector:
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,
Other Info:

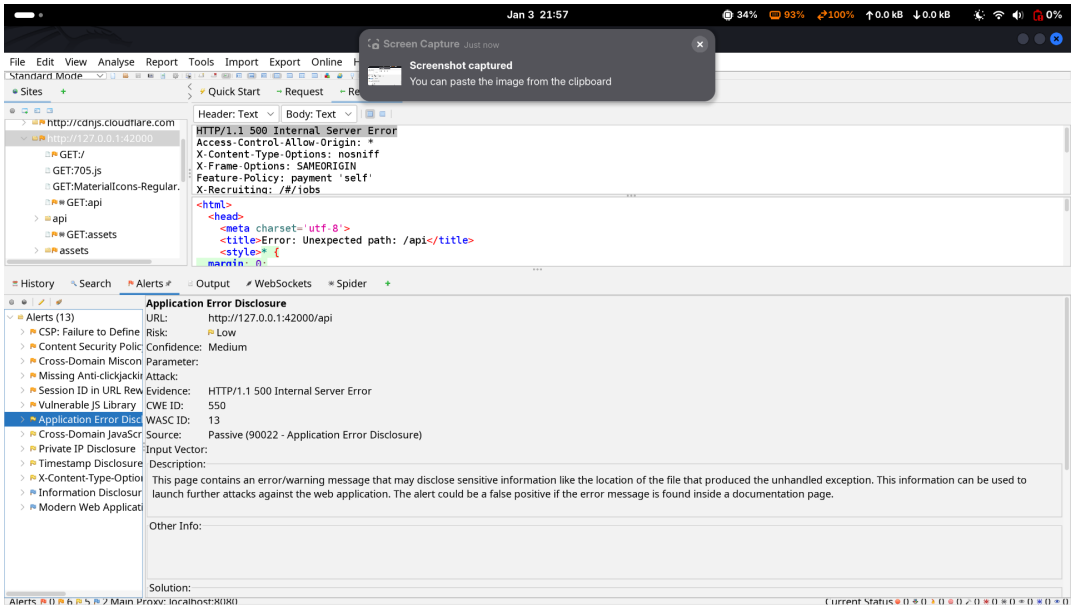

Jan 3 21:57
Screen Capture Just now
Screenshot captured
You can paste the image from the clipboard
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Cross-Domain Misconfiguration
URL: http://127.0.0.1:42000/runtime.js
Risk: Medium
Confidence: Medium
Parameter:
Attack:
Evidence: Access-Control-Allow-Origin: *
CWE ID: 264
WASC ID: 14
Source: Passive (10098 - Cross-Domain Misconfiguration)
Input Vector:
Description:
Web browser data loading may be possible, due to a Cross Origin Resource Sharing (CORS) misconfiguration on the web server.
Other Info:
The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing.
Solution:

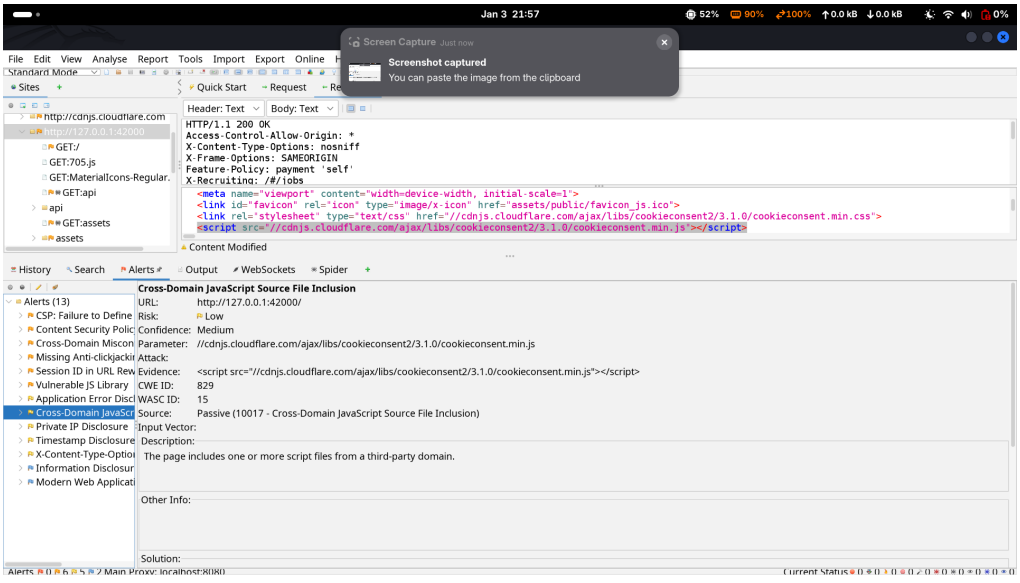

Jan 3 21:57
Screen Capture Just now
Screenshot captured
You can paste the image from the clipboard
File Edit View Analyse Report Tools Import Export Online
Standard Mode
Sites
Quick Start
Request
Header: Text
Body: Text
HTTP/1.1 200 OK
Access-Control-Allow-Origin: http://localhost:4200
Vary: Origin
Content-Type: text/html
Content-Length: 2
Date: Sat, 03 Jan 2026 21:54:50 GMT
ok
History
Search
Alerts
Output
WebSockets
Spider
Alerts (13)
Missing Anti-clickjacking Header
URL: http://127.0.0.1:42000/socket.io/?EIO=4&transport=polling&t=Pk5so3L&sid=1vDi4dZP8URnq6dIAAAC
Risk: Medium
Confidence: Medium
Parameter: x-frame-options
Attack:
Evidence:
CWE ID: 1021
WASC ID: 15
Source: Passive (10020 - Anti-clickjacking Header)
Alert Reference: 10020-1
Input Vector:
Description:
The response does not protect against 'ClickJacking' attacks. It should include either Content-Security-Policy with 'frame-ancestors' directive or X-Frame-Options.
Other Info:

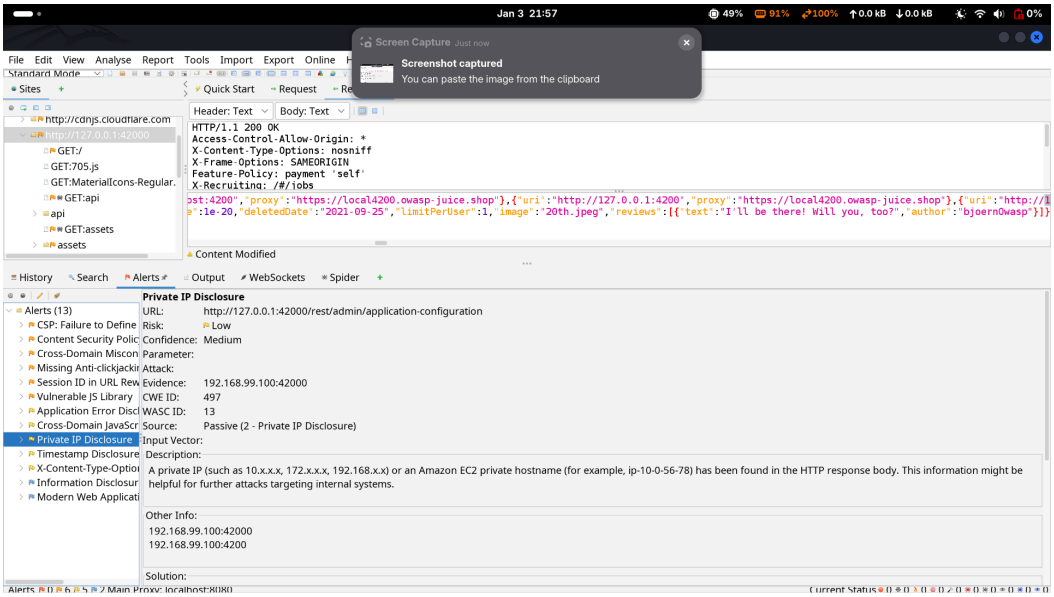

Jan 3 21:57
Screen Capture Just now
Screenshot captured
You can paste the image from the clipboard
POST http://127.0.0.1:42000/socket.io/?EIO=4&transport=polling&t=Pk5so3L&sid=1vDi4dZP8URnq6dIAAAC HTTP/1.1
host: 127.0.0.1:42000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:140.0) Gecko/20100101 Firefox/140.0
Accept: */*
Accept-Language: en-US,en;q=0.5
40
Session ID in URL Rewrite
URL: http://127.0.0.1:42000/socket.io/?EIO=4&transport=polling&t=Pk5so3L&sid=1vDi4dZP8URnq6dIAAAC
Risk: Medium
Confidence: High
Parameter: sid
Attack:
Evidence: 1vDi4dZP8URnq6dIAAAC
CWE ID: 598
WASC ID: 13
Source: Passive (3 - Session ID in URL Rewrite)
Alert Reference: 3-1
Input Vector:
Description:
URL rewrite is used to track user session ID. The session ID may be disclosed via cross-site referer header. In addition, the session ID might be stored in browser history or server logs.
Other Info:

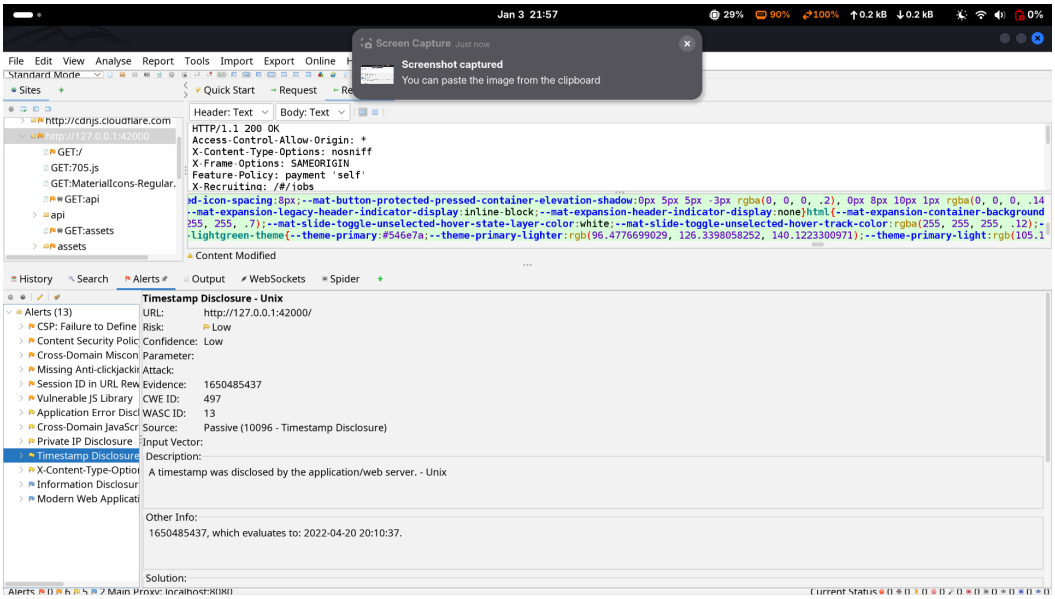

Jan 3 21:57
Screen Capture Just now
Screenshot captured
You can paste the image from the clipboard
POST http://127.0.0.1:42000/socket.io/?EIO=4&transport=polling&t=Pk5so3L&sid=1vDi4dZP8URnq6dIAAAC HTTP/1.1
host: 127.0.0.1:42000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:140.0) Gecko/20100101 Firefox/140.0
Accept: */*
Accept-Language: en-US,en;q=0.5
40
Session ID in URL Rewrite
URL: http://127.0.0.1:42000/socket.io/?EIO=4&transport=polling&t=Pk5so3L&sid=1vDi4dZP8URnq6dIAAAC
Risk: Medium
Confidence: High
Parameter: sid
Attack:
Evidence: 1vDi4dZP8URnq6dIAAAC
CWE ID: 598
WASC ID: 13
Source: Passive (3 - Session ID in URL Rewrite)
Alert Reference: 3-1
Input Vector:
Description:
URL rewrite is used to track user session ID. The session ID may be disclosed via cross-site referer header. In addition, the session ID might be stored in browser history or server logs.
Other Info:

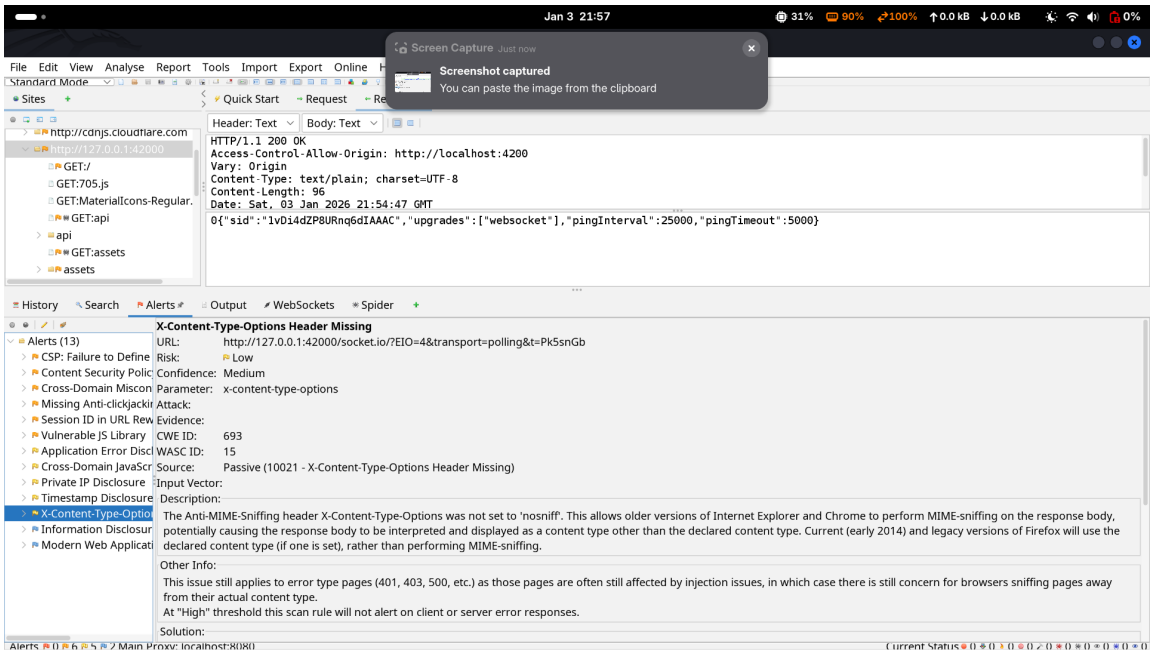

Jan 3 21:57
Screen Capture Just now
Screenshot captured
You can paste the image from the clipboard
GET http://cdnjs.cloudflare.com/ajax/libs/jquery/2.2.4/jquery.min.js HTTP/1.1
host: cdnjs.cloudflare.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:140.0) Gecko/20100101 Firefox/140.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Vulnerable JS Library
URL: http://cdnjs.cloudflare.com/ajax/libs/jquery/2.2.4/jquery.min.js
Risk: Medium
Confidence: Medium
Evidence: /2.2.4/jquery.min.js
CWE ID: 1395
Source: Passive (10003 - Vulnerable JS Library (Powered by Retire.js))
The identified library appears to be vulnerable.
The identified library jquery, version 2.2.4 is vulnerable.
CVE-2020-11023
CVE-2020-11022

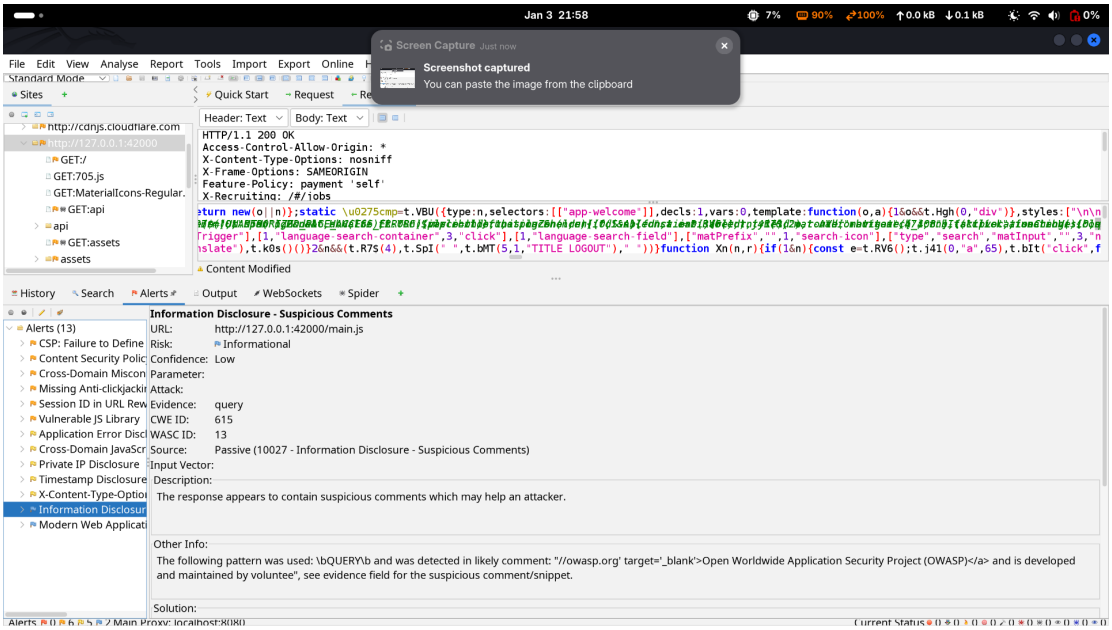

Jan 3 21:57
Screen Capture Just now
Screenshot captured
You can paste the image from the clipboard
HTTP/1.1 500 Internal Server Error
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
<html>
<head>
<meta charset='utf-8'>
<title>Error: Unexpected path: /api</title>
Application Error Disclosure
URL: http://127.0.0.1:42000/api
Risk: Low
Confidence: Medium
Evidence: HTTP/1.1 500 Internal Server Error
CWE ID: 550
WASC ID: 13
Source: Passive (90022 - Application Error Disclosure)
This page contains an error/warning message that may disclose sensitive information like the location of the file that produced the unhandled exception. This information can be used to launch further attacks against the web application. The alert could be a false positive if the error message is found inside a documentation page.

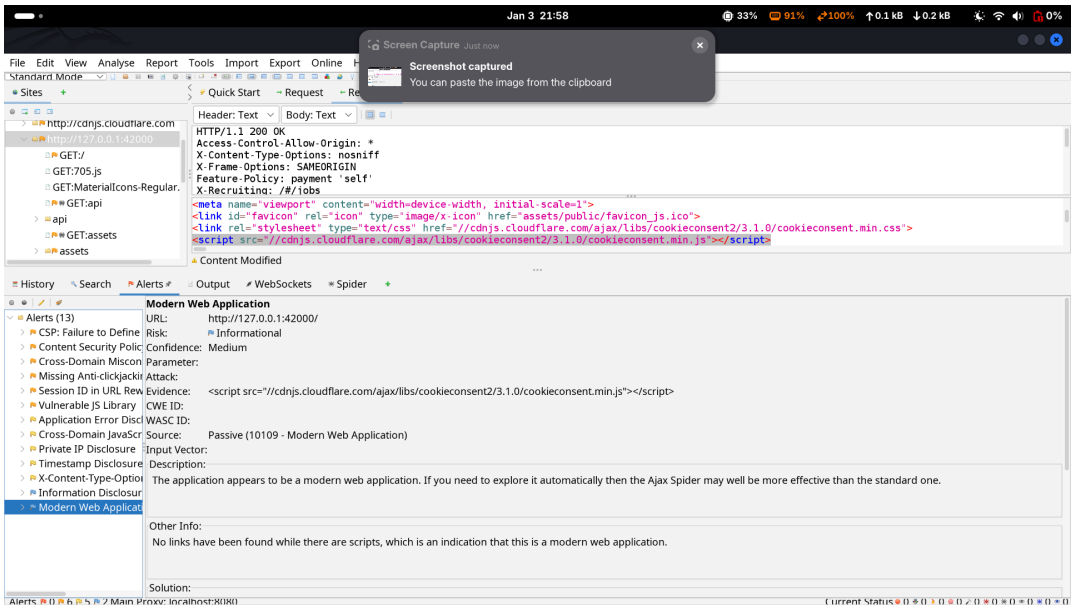

Jan 3 21:57
Screen Capture Just now
Screenshot captured
You can paste the image from the clipboard
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
<meta name="viewport" content="width=device-width, initial-scale=1">
<link id="favicon" rel="icon" type="image/x-icon" href="assets/public/favicon_js.ico">
<link rel="stylesheet" type="text/css" href="//cdnjs.cloudflare.com/ajax/libs/cookieconsent2/3.1.0/cookieconsent.min.css">
<script src="//cdnjs.cloudflare.com/ajax/libs/cookieconsent2/3.1.0/cookieconsent.min.js"></script>
Content Modified
Cross-Domain JavaScript Source File Inclusion
URL: http://127.0.0.1:42000/
Risk: Low
Confidence: Medium
Parameter: //cdnjs.cloudflare.com/ajax/libs/cookieconsent2/3.1.0/cookieconsent.min.js
Evidence: <script src="//cdnjs.cloudflare.com/ajax/libs/cookieconsent2/3.1.0/cookieconsent.min.js"></script>
CWE ID: 829
WASC ID: 15
Source: Passive (10017 - Cross-Domain JavaScript Source File Inclusion)
The page includes one or more script files from a third-party domain.

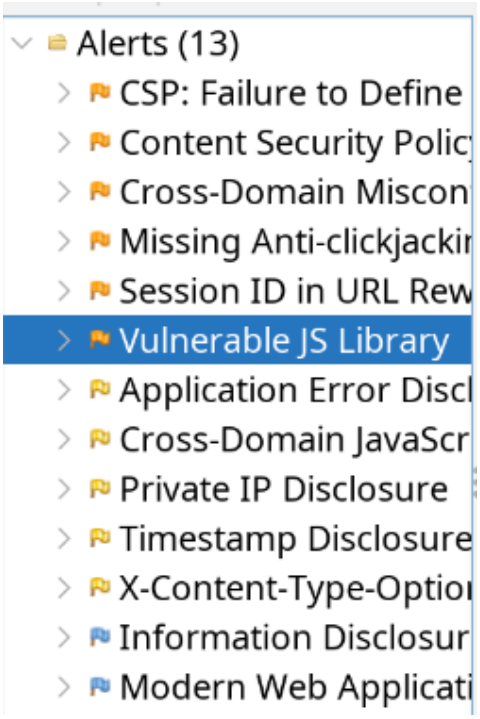

Jan 3 21:57
Screen Capture Just now
Screenshot captured
You can paste the image from the clipboard
Private IP Disclosure
URL: http://127.0.0.1:42000/rest/admin/application-configuration
Risk: Low
Confidence: Medium
Evidence: 192.168.99.100:42000
CWE ID: 497
WASC ID: 13
Source: Passive (2 - Private IP Disclosure)
Description:
A private IP (such as 10.x.x.x, 172.x.x.x, 192.168.x.x) or an Amazon EC2 private hostname (for example, ip-10-0-56-78) has been found in the HTTP response body. This information might be helpful for further attacks targeting internal systems.
Other Info:
192.168.99.100:42000
192.168.99.100:4200
Solution:


Jan 3 21:57
Screen Capture Just now
Screenshot captured
You can paste the image from the clipboard
Timestamp Disclosure - Unix
URL: http://127.0.0.1:42000/
Risk: Low
Confidence: Low
Evidence: 1650485437
CWE ID: 497
WASC ID: 13
Source: Passive (10096 - Timestamp Disclosure)
Description:
A timestamp was disclosed by the application/web server. - Unix
Other Info:
1650485437, which evaluates to: 2022-04-20 20:10:37.
Solution:


Jan 3 21:57
Screen Capture Just now
Screenshot captured
You can paste the image from the clipboard
X-Content-Type-Options Header Missing
URL: http://127.0.0.1:42000/socket.io/?EIO=4&transport=polling&t=Pk5snGb
Risk: Low
Confidence: Medium
Parameter: x-content-type-options
Evidence:
CWE ID: 693
WASC ID: 15
Source: Passive (10021 - X-Content-Type-Options Header Missing)
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 response 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.
Other Info:
This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type.
At "High" threshold this scan rule will not alert on client or server error responses.
Solution:


Jan 3 21:58
Screen Capture Just now
Screenshot captured
You can paste the image from the clipboard
Information Disclosure - Suspicious Comments
URL: http://127.0.0.1:42000/main.js
Risk: Informational
Confidence: Low
Parameter:
Attack:
Evidence: query
CWE ID: 615
WASC ID: 13
Source: Passive (10027 - Information Disclosure - Suspicious Comments)
Input Vector:
Description:
The response appears to contain suspicious comments which may help an attacker.
Other Info:
The following pattern was used: \bQUERY\b and was detected in likely comment: "//owasp.org' target='_blank'>Open Worldwide Application Security Project (OWASP) and is developed and maintained by voluntee", see evidence field for the suspicious comment/snippet.
Solution:


Jan 3 21:58
Screen Capture Just now
Screenshot captured
You can paste the image from the clipboard
Modern Web Application
URL: http://127.0.0.1:42000/
Risk: Informational
Confidence: Medium
Parameter:
Attack:
Evidence: <script src="//cdnjs.cloudflare.com/ajax/libs/cookieconsent2/3.1.0/cookieconsent.min.js"></script>
CWE ID:
WASC ID:
Source: Passive (10109 - Modern Web Application)
Input Vector:
Description:
The application appears to be a modern web application. If you need to explore it automatically then the Ajax Spider may well be more effective than the standard one.
Other Info:
No links have been found while there are scripts, which is an indication that this is a modern web application.
Solution:


Alerts (13)
CSP: Failure to Define
Content Security Polic
Cross-Domain Miscon
Missing Anti-clickjackin
Session ID in URL Rew
Vulnerable JS Library
Application Error Disc
Cross-Domain JavaScr
Private IP Disclosure
Timestamp Disclosure
X-Content-Type-Optio
Information Disclosur
Modern Web Applicati