

# Vulnerability Assessment Report

**Target Website:** Altoro Mutual (<https://demo.testfire.net>)

**Target IP:** 65.61.137.117

**Assessment Type:** Read-Only Web Security Assessment

**Prepared by:** Sirajudin Seid

**Program:** Cyber Security Task 1 (2026) – Future Interns

**Date::** jan-28-2026

## Introduction

1.1 Executive Summary.....	4
----------------------------	---

## Methodology

2.1 Assessment Scope.....	5
2.2 Assessment Methodology.....	6

## Findings

3.1 Vulnerability Summary .....	6
3.2 Detailed Vulnerability Findings.....	7
3.2.1 Missing / Weak Content Security Policy (CSP).....	7
3.2.2 Missing Security Headers.....	8
3.2.3 Insecure Cookie Configuration.....	9
3.2.4 CSRF Protection Not Implemented.....	10

3.2.5 Insecure Session Management.....	11
3.2.6 Information Disclosure.....	12
3.2.7 Cross-domain JavaScript inclusions and missing Subresource Integrity attributes.....	13
3.2.8 Exposed Services and Open Ports.....	14

## Recommendations & Remediation

4.1 Recommendations Prioritization .....	15
Conclusion.....	16

## Executive Summary

Vulnerability assessment was conducted to evaluate the publicly accessible web application Altoro Mutual (demo.testfire.net) for common security weaknesses using a strictly read-only approach. The assessment identified multiple configuration and exposure-related issues, primarily at the web application and service level. No exploitation or active attacks were performed.

The most significant risks identified include insecure cookie configurations and missing HTTP security headers, which could expose users to session-related and browser-based attacks. While no critical infrastructure weaknesses were observed, remediation of the identified issues is recommended to improve the overall security posture of the application.

Total Alerts Record	Medium Risk level	Low Risk Level	Informational risk level
23	10	8	5

# **Methodology**

## **2.1 Assessment Scope**

OWASP ZAP was configured in Safe Mode to perform passive vulnerability analysis while manually exploring the target application. No active scanning or intrusive testing was conducted.

The assessment was limited to publicly accessible pages of the target website. Authentication mechanisms, exploitation attempts, brute-force attacks, and denial-of-service testing were explicitly excluded from scope.

### **In Scope:**

- . Public web pages
- . HTTP/HTTPS services
- . Passive traffic inspection

### **Out of Scope:**

- . Login bypass attempts
- . Active exploitation
- . Credential attacks

## 2.2 Assessment Methodology

The assessment followed a passive security review approach, consisting of:

- Network and port exposure analysis
- Passive web vulnerability detection
- HTTP header and cookie inspection

Tools used during the assessment include:

- Nmap (Zenmap GUI)
- OWASP ZAP (Passive Scan)
- Browser Developer Tools

## Findings

### 3.1 Vulnerability Summary

Risk Level	Number of Findings
Medium	6
Low	2

## 3.2 Detailed Vulnerability Findings

### 3.2.1 Missing / Weak Content Security Policy (CSP)

Description	Impact	Risk Level	Fix
The web application does not implement a robust Content Security Policy (CSP). Passive analysis revealed that the CSP header is either not set or configured with unsafe directives such as unsafe-inline and wildcard (*) sources.	A weak or missing CSP increases the risk of client-side attacks such as Cross-Site Scripting (XSS). Without proper restrictions, malicious scripts could be executed in users' browsers, potentially leading to data theft or session compromise.	Medium	Implement a strict Content Security Policy that restricts script, style, and resource loading to trusted sources only. Avoid the use of unsafe-inline and wildcard directives where possible.

### Proof of Concept

- > [CSP: Failure to Define Directive with No Fallback \(2\)](#)
- > [CSP: Wildcard Directive \(2\)](#)
- > [CSP: script-src unsafe-inline \(2\)](#)
- > [CSP: style-src unsafe-inline \(2\)](#)
- > [Content Security Policy \(CSP\) Header Not Set \(Systemic\)](#)
- > [Cross-Domain Misconfiguration \(3\)](#)
- > [Missing Anti-clickjacking Header \(Systemic\)](#)
- > [Session ID in URL Rewrite \(3\)](#)
- > [Sub Resource Integrity Attribute Missing](#)
- > [Cookie No HttpOnly Flag \(Systemic\)](#)
- > [Cookie Without Secure Flag \(Systemic\)](#)
- > [Cookie without SameSite Attribute \(Systemic\)](#)
- > [Cross-Domain JavaScript Source File Inclusion](#)

### 3.2.2 Missing Security Headers

Description	Impact	Risk Level	Fix
<p>Several recommended HTTP security headers are missing from server responses, including:</p> <ul style="list-style-type: none"> <li>• X-Frame-Options (anti-clickjacking)</li> <li>• X-Content-Type-Options</li> <li>• Strict-Transport-Security (HSTS)</li> </ul>	<p>The absence of these headers increases exposure to browser-based attacks such as clickjacking, MIME-type sniffing, and downgrade attacks. This weakens the overall client-side security posture of the application.</p>	Medium	<p>Configure the web server to include all recommended security headers according to OWASP best practices, ensuring consistent enforcement across all responses.</p>

## Proof of Concept

> ! Missing Anti-clickjacking Header (Systemic) 1

> ! Session ID in URL Rewrite (3)

> ! Sub Resource Integrity Attribute Missing

> ! Cookie No HttpOnly Flag (Systemic)

> ! Cookie Without Secure Flag (Systemic)

> ! Cookie without SameSite Attribute (Systemic)

> ! Cross-Domain JavaScript Source File Inclusion

> ! Server Leaks Version Information via "Server" HTTP Response Header Field (Systemic)

> ! Strict-Transport-Security Header Not Set (9) 2

> ! Timestamp Disclosure - Unix (Systemic)

> ! X-Content-Type-Options Header Missing (Systemic) 3

> ! Information Disclosure - Suspicious Comments (7)

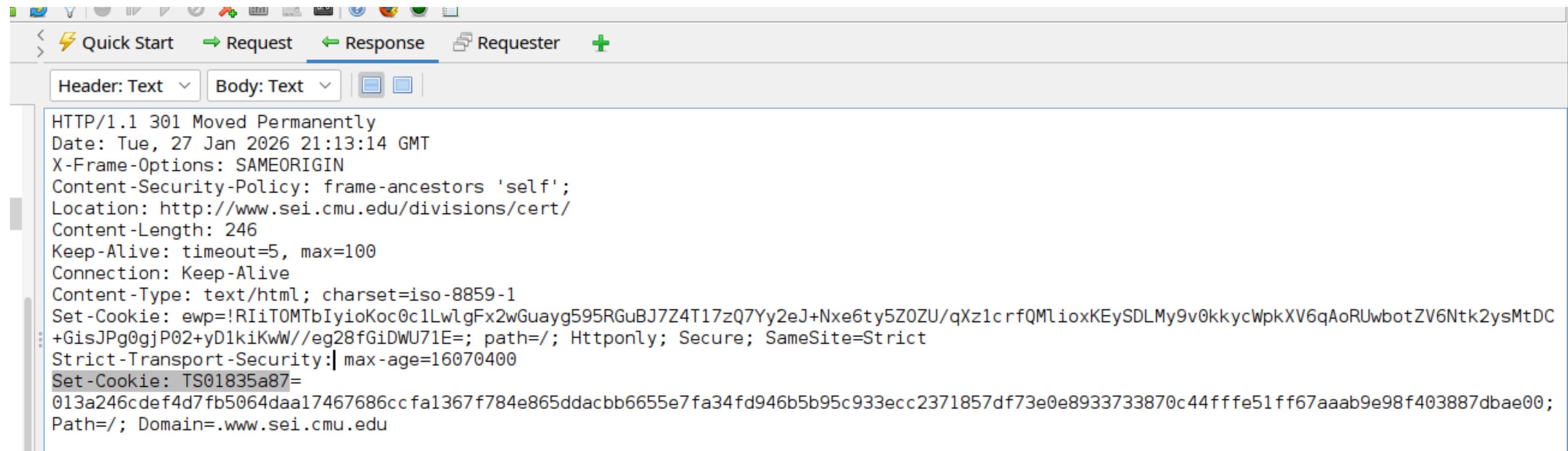
> ! Modern Web Application (3)

Alerts ! 0 ! 10 ! 8 ! 5 Main Proxy: localhost:8080

### 3.2.3 Insecure Cookie Configuration

Description	Impact	Risk Level	Fix
Session cookies used by the application are missing critical security attributes such as Secure, HttpOnly, and SameSite.	Cookies without these attributes may be accessed via client-side scripts or transmitted over insecure connections, increasing the risk of session hijacking, cross-site scripting, and cross-site request forgery (CSRF) attacks.	Medium	Ensure all session cookies are configured with Secure, HttpOnly, and SameSite attributes to protect session data and reduce attack surface.

## Proof of Concept



The screenshot shows a proxy tool interface with tabs for 'Request' and 'Response'. The 'Response' tab is selected, displaying the following HTTP response:

```
HTTP/1.1 301 Moved Permanently
Date: Tue, 27 Jan 2026 21:13:14 GMT
X-Frame-Options: SAMEORIGIN
Content-Security-Policy: frame-ancestors 'self';
Location: http://www.sei.cmu.edu/divisions/cert/
Content-Length: 246
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=iso-8859-1
Set-Cookie: ewp=!RIiTOMTbIyioKoc0c1LwlxFx2wGuayg595RGuBJ7Z4T17zQ7Yy2eJ+Nxe6ty5Z0ZU/qXz1crfQMlioxKEySDLMy9v0kkycWpkXV6qAoRUwbotZV6Ntk2ysMtDC
+GisJPg0gjP02+yD1kiKwW//eg28fGiDWU71E=; path=/; Httponly; Secure; SameSite=Strict
Strict-Transport-Security: max-age=16070400
Set-Cookie: TS01835a87=
013a246cdef4d7fb5064daa17467686ccfa1367f784e865ddacbb6655e7fa34fd946b5b95c933ecc2371857df73e0e8933733870c44ffffe51ff67aaab9e98f403887dbae00;
Path=/; Domain=.www.sei.cmu.edu
```

### 3.2.4 CSRF Protection Not Implemented

Description	Impact	Risk Level	Fix
The application does not appear to implement anti-CSRF tokens for state-changing requests, as identified through passive analysis.	Without CSRF protection, attackers may be able to trick authenticated users into performing unintended actions, potentially compromising user accounts or application data.	Medium	Implement anti-CSRF tokens for all sensitive and state-changing requests and validate them server-side.

proof of concept

The screenshot shows a software interface for security testing. At the top, there's a tree view under 'Alerts (23)' with a single expanded item: 'Absence of Anti-CSRF Tokens (2)'. Below this, two specific alerts are listed: 'GET: http://demo.testfire.net/login.jsp' and 'GET: http://demo.testfire.net/subscribe.jsp'. The second alert is highlighted with a blue selection bar. Below the tree view is a toolbar with icons for 'Quick Start', 'Request', 'Response', 'Requester', and a '+' button. Underneath the toolbar, there are dropdown menus for 'Header: Text' and 'Body: Text', along with some small icons. The main pane displays the raw HTTP response headers for the selected alert:

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: text/html; charset=ISO-8859-1
Date: Tue, 27 Jan 2026 21:14:09 GMT
content-length: 8545
```

### 3.2.5 Insecure Session Management

Description	Impact	Risk Level	Fix
The application was observed rewriting session identifiers within URLs instead of using secure cookies exclusively.	Session IDs included in URLs may be exposed through browser history, logs, or referrer headers, increasing the risk of session fixation or session hijacking.	Medium	Use secure, cookie-based session management and avoid exposing session identifiers in URLs.

proof of concept

The screenshot shows a NetworkMiner capture window. The title bar says "Session ID in URL Rewrite (3)". Below it, there are three captured POST requests to "https://analytics.google.com/g/collect". The main pane displays a detailed view of a single POST request. The "Request" tab is selected, showing the following details:

Method: POST  
URL: https://analytics.google.com/g/collect  
Headers:

- Host: analytics.google.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
- Origin: null
- Connection: keep-alive
- Sec-Fetch-Dest: empty
- Sec-Fetch-Mode: no-cors
- Sec-Fetch-Site: cross-site
- Priority: u=6
- Pragma: no-cache
- Cache-Control: no-cache
- Content-Length: 0

Body:

```
POST https://analytics.google.com/g/collect?v=2&tid=G-87WECW6HCS&gtm=45je61q0v876982250z871720660za20gzb71720660zd71720660&p=1769548405469&_gaz=1&gcd=13l3l3l3l111&npo=0&dma=0&cid=1860796532.1769548412&ul=en-us&sr=1920x1080&frm=0&psc=0&dl=noapi&_eu=AAAAAGA&_s=1&tag_exp=102015666~103116026~103200004~104527906~104528500~104684208~104684211~115616986~115938465~115938468~116185181~116185182~116682875~116988316~117041587~117223564&sid=1769548412&sct=1&seg=0&dl=https%3A%2F%2Fwww.sei.cmu.edu%2Fdivisions%2Fcert%2F&dr=http%3A%2F%2Fdemo.testfire.net%2F&dt=CERT&_tu=QA&en=page_view&_fv=1&_nsi=1&_ss=1&tfd=20382
```

### 3.2.6 Information Disclosure

Description	Impact	Risk Level	Fix
The application discloses internal information through HTTP response headers and page content, including server version details, timestamps, and suspicious comments.	Such information may assist attackers during reconnaissance by revealing technologies, versions, or internal logic that could be targeted in future attacks.	low	Remove or obfuscate unnecessary information from HTTP responses and client-facing content.  Ensure that your web server, application server, load balancer, etc. is configured to suppress the "Server" header or provide generic details.

proof of concept

- > [Cookie without SameSite Attribute \(Systemic\)](#)
- > [Cross-Domain JavaScript Source File Inclusion](#)
- > [Server Leaks Version Information via "Server" HTTP Response Header Field \(Systemic\) ←](#)
- > [Strict-Transport-Security Header Not Set \(9\)](#)
- > [Timestamp Disclosure - Unix \(Systemic\) ←](#)
- > [X-Content-Type-Options Header Missing \(Systemic\)](#)
- > [Information Disclosure - Suspicious Comments \(7\)](#)
- > [Modern Web Application \(3\)](#)

Alerts [0](#) [10](#) [8](#) [5](#) Main Proxv: localhost:8080

### 3.2.7 Cross-domain JavaScript inclusions and missing Subresource Integrity attributes.

Description	Impact	Risk Level	Fix
<p>The application includes JavaScript resources from external domains without sufficient restrictions or validation.</p> <p>The integrity attribute is missing on a script or link tag served by an external server. The integrity tag prevents an attacker who have gained access to this server from injecting a malicious content.</p>	<p>Third-party scripts can introduce supply-chain risks. If an external source is compromised, malicious code may be delivered to users without the application owner's knowledge.</p> <p>Without SRI, the application cannot verify the integrity of third-party resources, increasing the risk of malicious code execution if those resources are altered.</p>	low	<p>Limit the use of third-party scripts to trusted providers and review them regularly for security risks.</p> <p>Provide a valid integrity attribute to the tag.</p>

## proof of concept

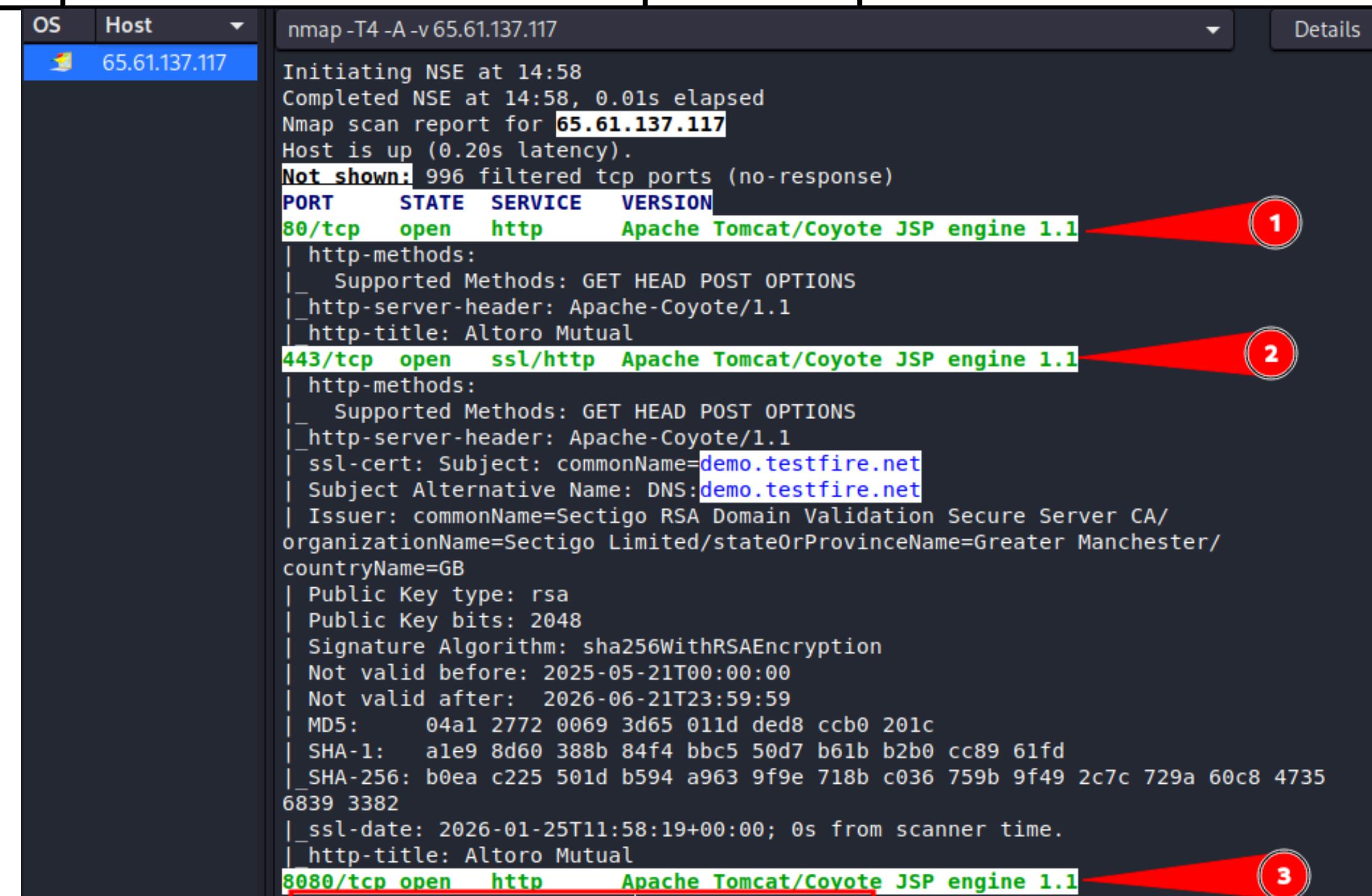
The screenshot shows a browser developer tools interface with the Network tab selected. Two entries are listed:

- Cross-Domain JavaScript Source File Inclusion**:
  - Method: GET
  - URL: https://www.sei.cmu.edu/divisions/cert/
- Sub Resource Integrity Attribute Missing**:
  - Method: GET
  - URL: https://www.sei.cmu.edu/divisions/cert/

### 3.2. 8 Exposed Services and Open Ports

Description	Impact	Risk Level	Fix
A passive port scan using Nmap/Zenmap identified publicly accessible services running on the target system.	Exposed services increase the application's attack surface and may be exploited if misconfigured or not properly maintained.	Medium	Review all exposed services and restrict access to only those required for business operations.

proof of concept



```

OS | Host ▾
  65.61.137.117

nmap -T4 -A -v 65.61.137.117
Initiating NSE at 14:58
Completed NSE at 14:58, 0.01s elapsed
Nmap scan report for 65.61.137.117
Host is up (0.20s latency).
Not shown: 996 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
80/tcp    open  http    Apache Tomcat/Coyote JSP engine 1.1
          | http-methods:
          |_ Supported Methods: GET HEAD POST OPTIONS
          |_ http-server-header: Apache-Coyote/1.1
          |_ http-title: Altoro Mutual
443/tcp   open  ssl/http Apache Tomcat/Coyote JSP engine 1.1
          | http-methods:
          |_ Supported Methods: GET HEAD POST OPTIONS
          |_ http-server-header: Apache-Coyote/1.1
          | ssl-cert: Subject: commonName=demo.testfire.net
          | Subject Alternative Name: DNS:demo.testfire.net
          | Issuer: commonName=Sectigo RSA Domain Validation Secure Server CA/
          | organizationName=Sectigo Limited/stateOrProvinceName=Greater Manchester/
          | countryName=GB
          | Public Key type: rsa
          | Public Key bits: 2048
          | Signature Algorithm: sha256WithRSAEncryption
          | Not valid before: 2025-05-21T00:00:00
          | Not valid after: 2026-06-21T23:59:59
          | MD5: 04a1 2772 0069 3d65 011d ded8 ccb0 201c
          | SHA-1: ale9 8d60 388b 84f4 bbc5 50d7 b61b b2b0 cc89 61fd
          |_ SHA-256: b0ea c225 501d b594 a963 9f9e 718b c036 759b 9f49 2c7c 729a 60c8 4735
6839 3382
          |_ ssl-date: 2026-01-25T11:58:19+00:00; 0s from scanner time.
          |_ http-title: Altoro Mutual
8080/tcp  open  http    Apache Tomcat/Coyote JSP engine 1.1
  
```

# Recommendations & Remediation

## 4.1 Recommendations Prioritization

Medium	<ul style="list-style-type: none"><li>. Missing / Weak CSP</li><li>. Missing Security Headers</li><li>. Insecure Cookie Configuration</li><li>. Absence of Anti-CSRF Tokens</li><li>. Session ID in URL Rewrite</li><li>. Exposed Services &amp; Open Ports</li></ul>
Low	<ul style="list-style-type: none"><li>. Information Disclosure</li><li>. Subresource Integrity Missing</li></ul>

# **Conclusion**

The assessment revealed several configuration-level security weaknesses that could impact user security if left unaddressed. While no exploitation was performed, remediation of the identified issues will significantly enhance the security posture of the Altoro Mutual application.

**THANK-YOU**