Web Application Security Testing of NCSC

Test URL

https://ncscbeta.aeologic.in/

Level-1

15th April, 2025

AAA Technologies P. Ltd

278-280, F Wing, Solaris-1,
Saki Vihar Road, Opp. L & T Gate No. 6,
Powai, Andheri (East),
Mumbai 400 072, INDIA
Tel: + 91 22 28573815 / 16
Fax: + 91 22 40152501
info@aaatechnologies.co.in
www.aaatechnologies.co.in

Web Application Security Test Report For NCSC

Document Version Control			
Data Classification		CLASSIFIED	
Client Name		NICSI	
Document Title		Web Application Security Test Report	
Author		Vikas Kumar	
Version	**Date of Issue**	**Issued by**	**Change Description**
1.0	15-04-2025	AAA Technologies	Level-1

Web Application Security Test Report

Presented by:

Vikas Kumar

Application Testing Conducted On:

08-04-2025 to 15-04-2025

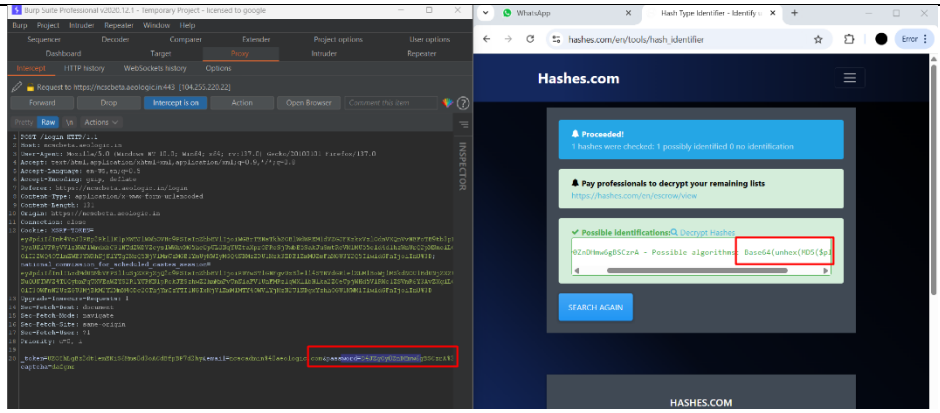
Web Application Security Test Report For NCSC

Table of Contents

High

Web Application Security Test Report For NCSC

1. Weak algorithm MD5 is used

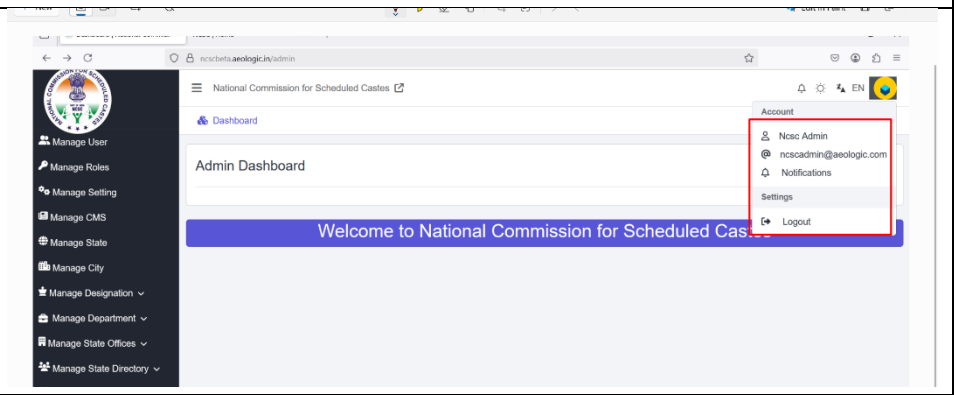
1) Vulnerability Title: Weak algorithm md5 is used	
Risk	**Medium**
Abstract	It was observed that weak algorithm is used in the application
Ease of Exploitation	Easy
Impact	An attacker can construct forged data in a variety of forms that will cause software using the MD5 algorithm to incorrectly identify it as trustworthy.MD5 is vulnerable to Collision Attacks in which the Hashing algorithm takes two different inputs and produce the same hash function.
Recommendations	It is recommended to use SHA-256 for more secured application.
Snapshot	
Affected URLs	**throughout the application**

Web Application Security Test Report For NCSC

MEDIUM

2.Audit Trail is not implemented properly

1) Vulnerability Title: Audit Trail is not implemented in properly	
Risk	**Medium**
Abstract	The application does not maintain the logout action and status of user activity where all user activities have to be logged.
Ease of Exploitation	Easy
Impact	In-case a malicious user tries to attack the application; the application will not be able to trace the attacker
Recommendations	An Audit trail should be incorporated in the application admin module, where all user activities have to be logged. Following points should be considered: • Audits are to be generated at the time of resource access and by the same routines accessing the resource • Information to be logged including the following: IP of the originating client, Date, Time, username if any in addition to other details to be logged in the web server. • These IP, date, time, session details, user details (NO password), referrer, process id to be logged in application logs. • To create audit • logs, use auto numbering so that every logged entry has a log number, which is not editable. Then if one audit entry is deleted a gap in the numbering sequence will appear. • Log entries are to be hashed/ signed so that changes to audit log can be detected. • Audit trails to answer the following • Logging of Authentication Process. Success and failed attempts. • Logging Authentication details and changes. • Software error and failures logged. • Should not be possible to retrieve confidential authentication information from these logs (including passwords) • Is it possible to uniquely identify both client host and user from these logs?

	• What level of information is logged by the application (read/write access, modification data, and copy/paste data) Are log files time sequential and can they positively identify the time of action?
Snapshot	
Affected URLs	**throughout the application**

Web Application Security Test Report For NCSC

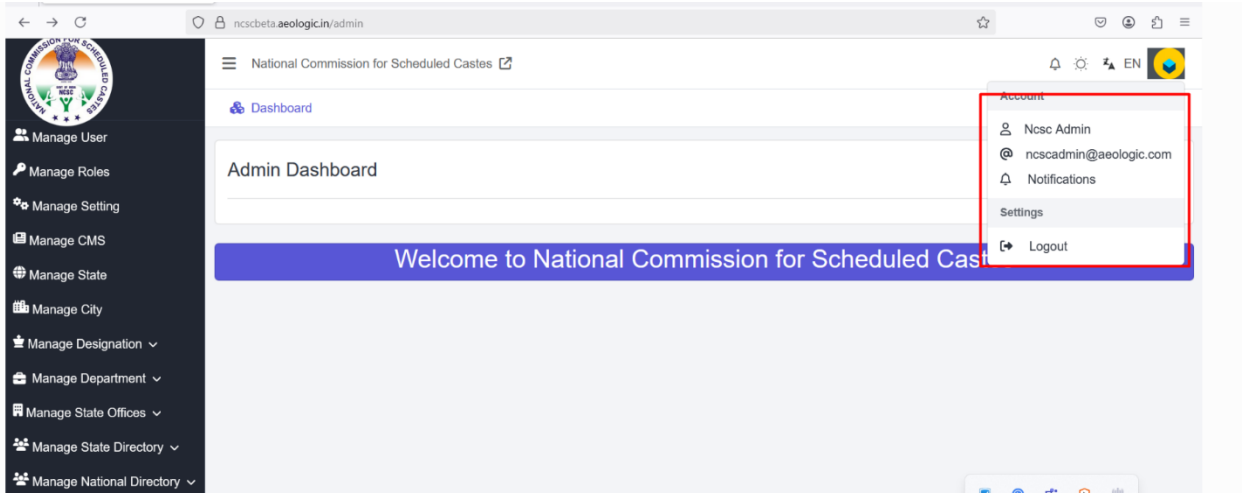
Web Application Security Test Report For NCSC

LOW

3.Change password module is not implemented

Finding: In this application forgot password module is not implemented.
Recommendation: Users may be required to change their password. Users should be provided with a “Change Password” module through which user will change their password whenever required. There is the following conditions should be implemented for Change Password module:
? The password between client and server must be passed in SHA-256 hash technique.
? Passwords should have restrictions that require a minimum size (8-15 characters) and complexity for the password. Complexity typically requires the use of minimum combinations of alphabetic, numeric, and non-alphanumeric characters in a user’s password (e.g. at least one special character ($,@,#,&), one upper case letter, one lower case letter and one number like Test@123).
? Users should be prevented from reusing their current or previous 3 passwords. Password history should ideally be 3.

Screenshot:

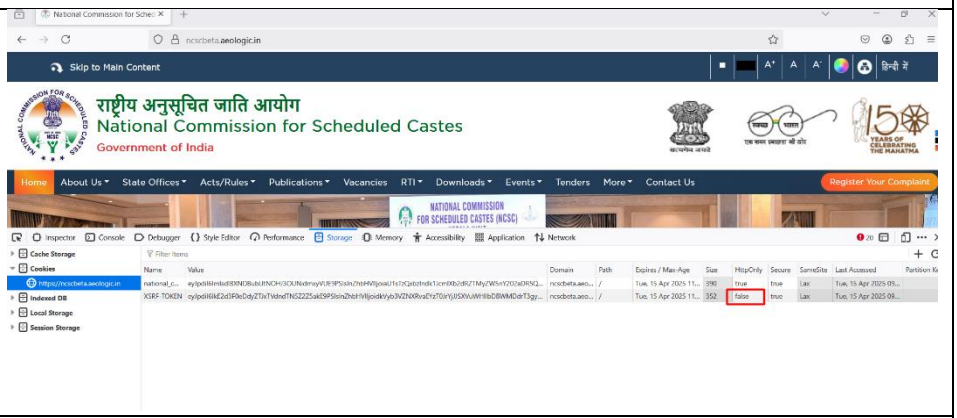


4.SRI Misconfiguration is not implemented in the application

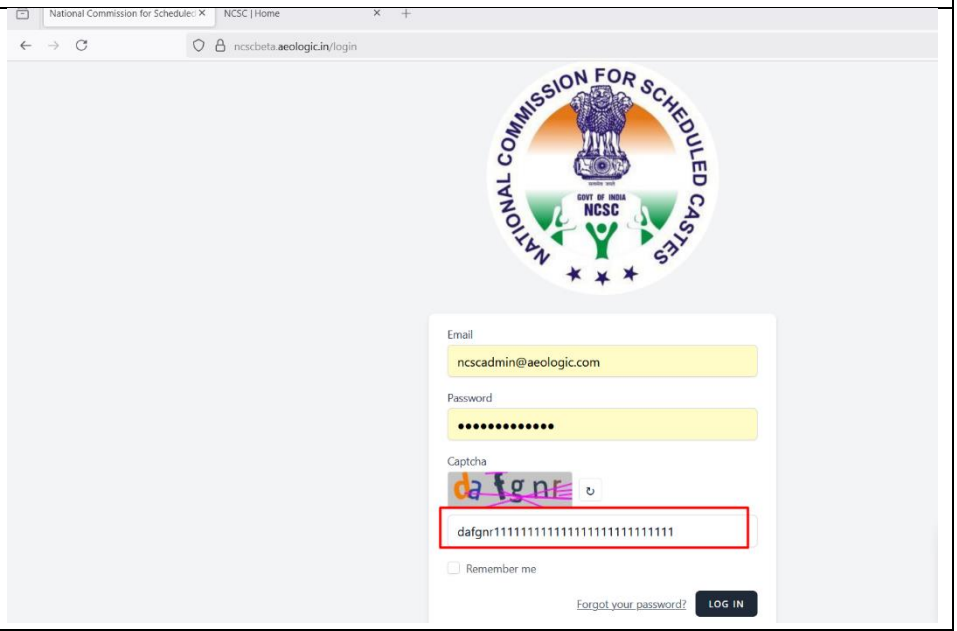
Vulnerability Title: SRI Misconfiguration is not implemented in the application	
Risk	Low
Abstract	Subresource Integrity (SRI) is a security feature that enables browsers to verify that third-party resources they fetch (for example, from a CDN) are delivered without unexpected manipulation. It works by allowing developers to provide a cryptographic hash that a fetched file must match.
Ease of Exploitation	Easy
Impact	An attacker that has access or has hacked the hosting CDN can manipulate or replace the files.
Recommendations	Use the SRI Hash Generator link (from the References section) to generate a <script> element that implements Subresource Integrity (SRI). For example, you can use the following <script> element to tell a browser that before executing the https://example.com/example-framework.js script, the browser must first compare the script to the expected hash, and verify that there's a match.

	<script src="https://example.com/example-framework.js" integrity="sha384-oqVuAfXRKap7fdgcCY5uykM6+R9GqQ8K/uxy9rx7HNQlGYl1kPzQho1wx4JwY8wC" crossorigin="anonymous"></script>
Snapshot	
Affected URLs	Throughout the application

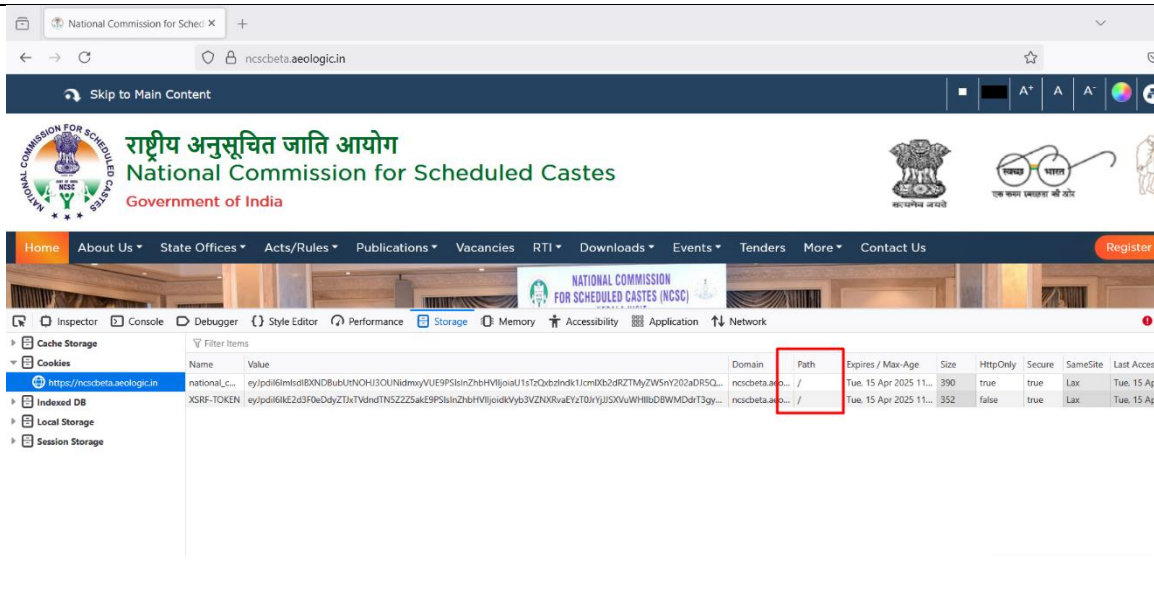
5.HTTP only flag is not set properly in the application

2) Vulnerability Title: Session Cookie without Secure Flag	
Risk	**Medium**
Abstract	It was observed that Session Cookie did not have Secure Flag Set.
Ease of Exploitation	Easy
Impact	This session cookie does not have the Secure flag set. When a cookie is set with the Secure flag, it instructs the browser that the cookie can only be accessed over secure SSL channels. This is an important security protection for session cookies.
Recommendations	HttpOnly flag should be set to “True” in website’s configuration file.
Snapshot	
Affected URLs	Throughout the Application

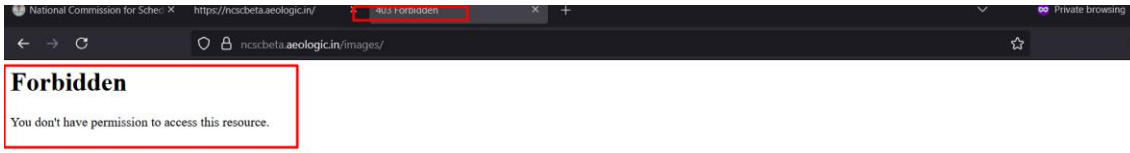
6.Max. Length for input fields is not defined called Buffer overflow.

1) Vulnerability Title: Buffer overflow	
Risk	**Low**
Abstract	It was observed that max. Length for captcha fields is not defined.
Ease of Exploitation	Easy
Impact	This vulnerability can cause a system crash or, worse, create an entry point for a cyberattack.
Recommendations	Length restrictions for every input field should be defined at client as well as at server end.
Snapshot	
Affected URLs	**Throughout the Application**

7.Path Not Set in Cookie Attributes

Vulnerability Title: Path Not Set in Cookie Attributes	
Risk	**Low**
Abstract	It was observed that path is set to default i.e. '/' in the application.
Ease of Exploitation	Easy
Impact	It is difficult to keep track of logged in users in case of any incident theft/fraud.
Recommendations	It is recommended to verify that that the path attribute, just as the Domain attribute, been set too loosely. Even if the Domain attribute has been configured as tight as possible, if the path is set to t directory "/" then it can be vulnerable to less secure applications on the same server.
Snapshot	
Affected URLs	**Throughout the application**

8. Forbidden Resource

Vulnerability Title: Forbidden Resource	
Risk	**Low**
Abstract	It was observed that there is a Forbidden Resource Access to this resource been denied by the web server. This is generally not a security issue, and is reported here for informational purposes.
Impact	This issue is reported as additional information only. There is no direct imp arising from this issue.
Recommendations	It is recommended to restrict such information from being displayed to the
Snapshot	
Affected URLs	**Throughout the application**

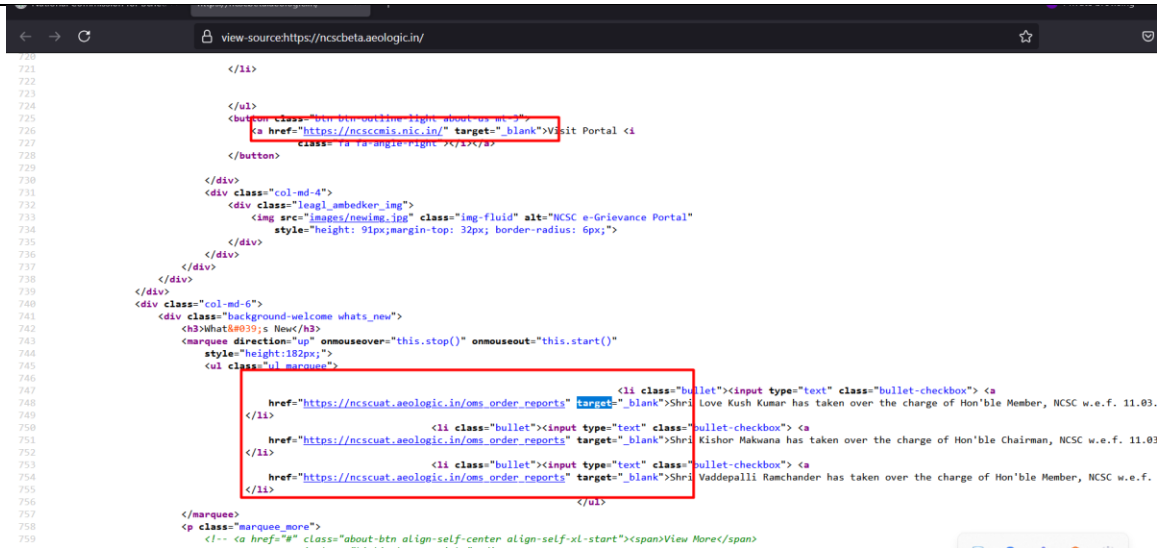
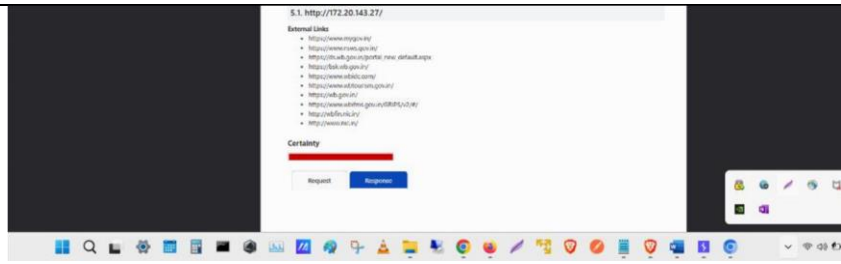
9. Robots.txt

Vulnerability Title: Robots.txt	
Risk	Low
Abstract	It was observed that this page contains an error message that may disclose sensitive information.
Ease of Exploitation	Easy
Impact	The error messages may disclose sensitive information.
Recommendations	It is recommended to implement custom error on the web application.
Snapshot	
Affected URLs	Throughout the Application

10. Phishing by Navigating Browser Tabs

Vulnerability Title: Phishing by Navigating Browser Tabs	
Risk	**Low**
Abstract	It was observed that Phishing by Navigating Browser Tabs is possible on web applicatio
Ease of Exploitation	Easy
Impact	While this vulnerability doesn't allow script execution, it does allow phishing attacks that silently replace the parent tab. If the links lack rel="noopener noreferrer" attribute, a third pa site can change the URL of the source tab using *window.opener.location.assign* and trick users into thinking that they're still in a trusted page and lead them to enter their sensiti on the malicious website.
Recommendations	-Add rel=noopener to the links to prevent pages from abusing *window.opener*. This ens that the page cannot access the *window.opener* property in Chrome and Opera brow -For older browsers and in Firefox, you can add rel=noreferrer which additionally disab Referer header. ...

Web Application Security Test Report For NCSC

Snapshot	
	
Affected URLs	**Throughout the application**