Web-Application Penetration Testing Report

For

Website URL: http://www.csl.nic.in/

Date: 9th June 2022

Contact Us :

Application Security Audits and Assessment Division
(Application Security Group)
National Informatics Centre
A – Block,CGO Complex, Lodhi Road
New Delhi - 110003

011-2430-5877
011-2430-5142
011-2430-5934
011-2430-5215

Contributions:

	Name	Role
1.	Mr. Pradeep Kumar Mrs. Alka Upadhyay	Reviewer
2.	Mr. Rajesh Mishra	HOD
3.	Mr. C.J. Antony	HOG

Key Findings

1. Weak Hashing Algorithm – High
2. Open File Upload – High
3. CAPTCHA Bypass – High
4. Critical Information Disclosure – High
5. Possible SQL Injection – High
6. Directory Listing – High
7. Unencrypted Communication – Medium
8. Insecure HTTP Methods – Low
9. CORS Misconfiguration – Low
10. Insecure Cookie Attributes – Low
11. Private IP Disclosure – Low
12. Information Disclosure - Low
13. Vulnerable and Outdated Component - Low
14. Missing Security Headers – Low
15. Admin Page Accessible – Low
16. Email Harvesting – Low

1. Weak Hashing Algorithm

Incident URL: http://csl.nic.in/login/index

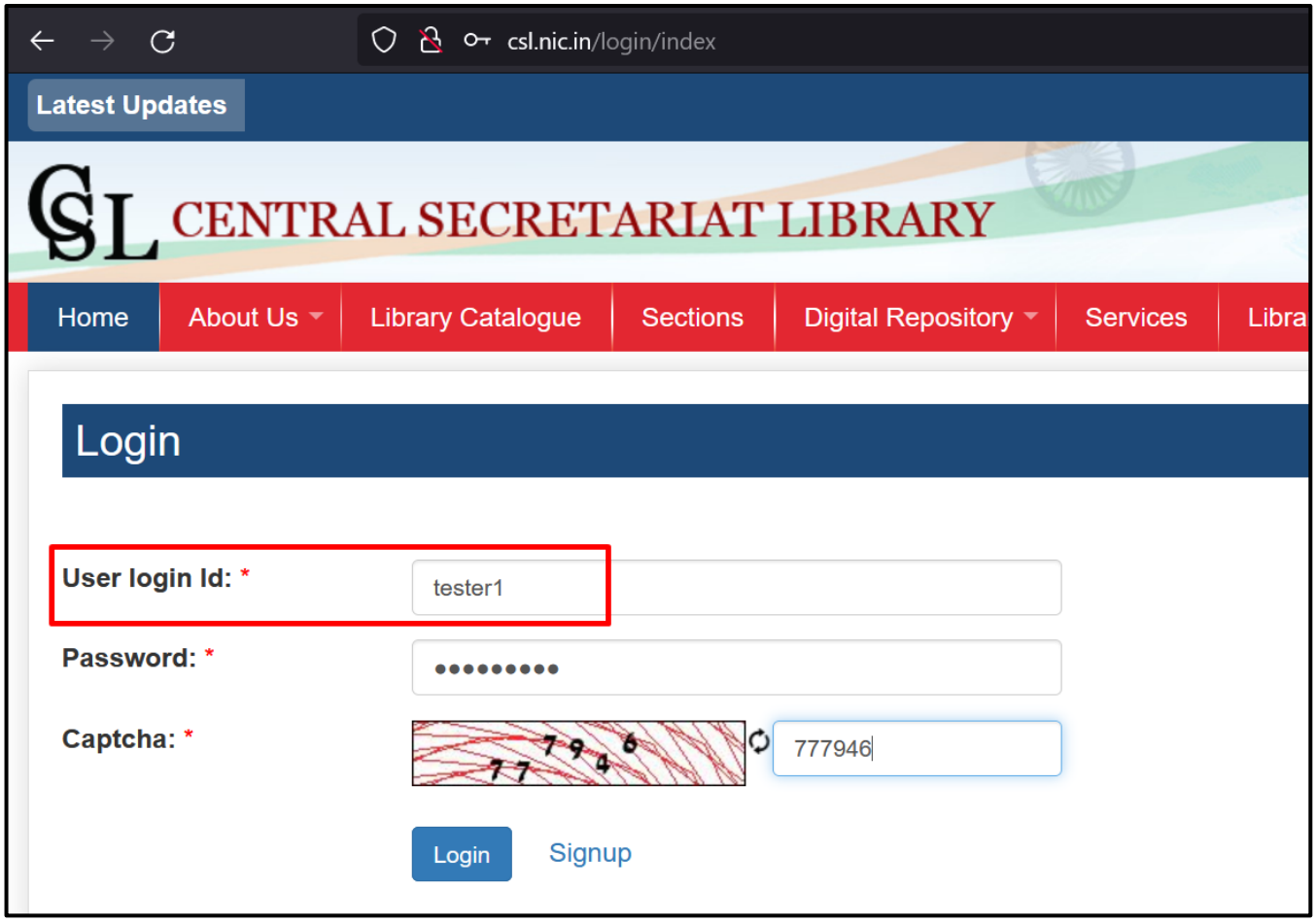
Description: The application has improperly implemented Salted Hashing Technique for transmitting of password from client to server.

Impact: This vulnerability allows an attacker to steal the salted hashed password and can replay to login to the application.

Severity: High

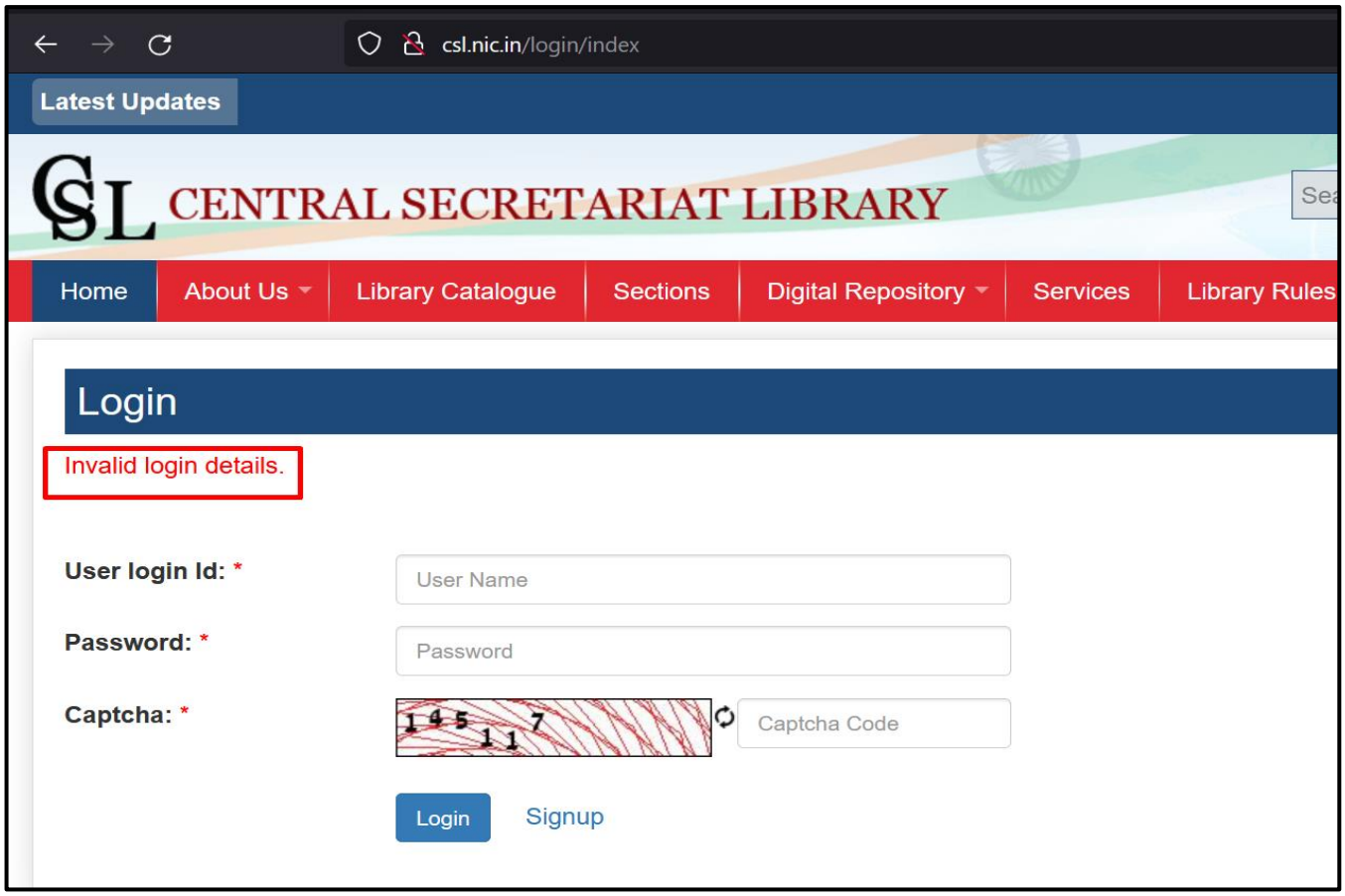
How to Test:

Step#1: A victim user navigates to the application at URL: http://csl.nic.in/login/index and enters invalid username and valid password. He clicks on Login button.

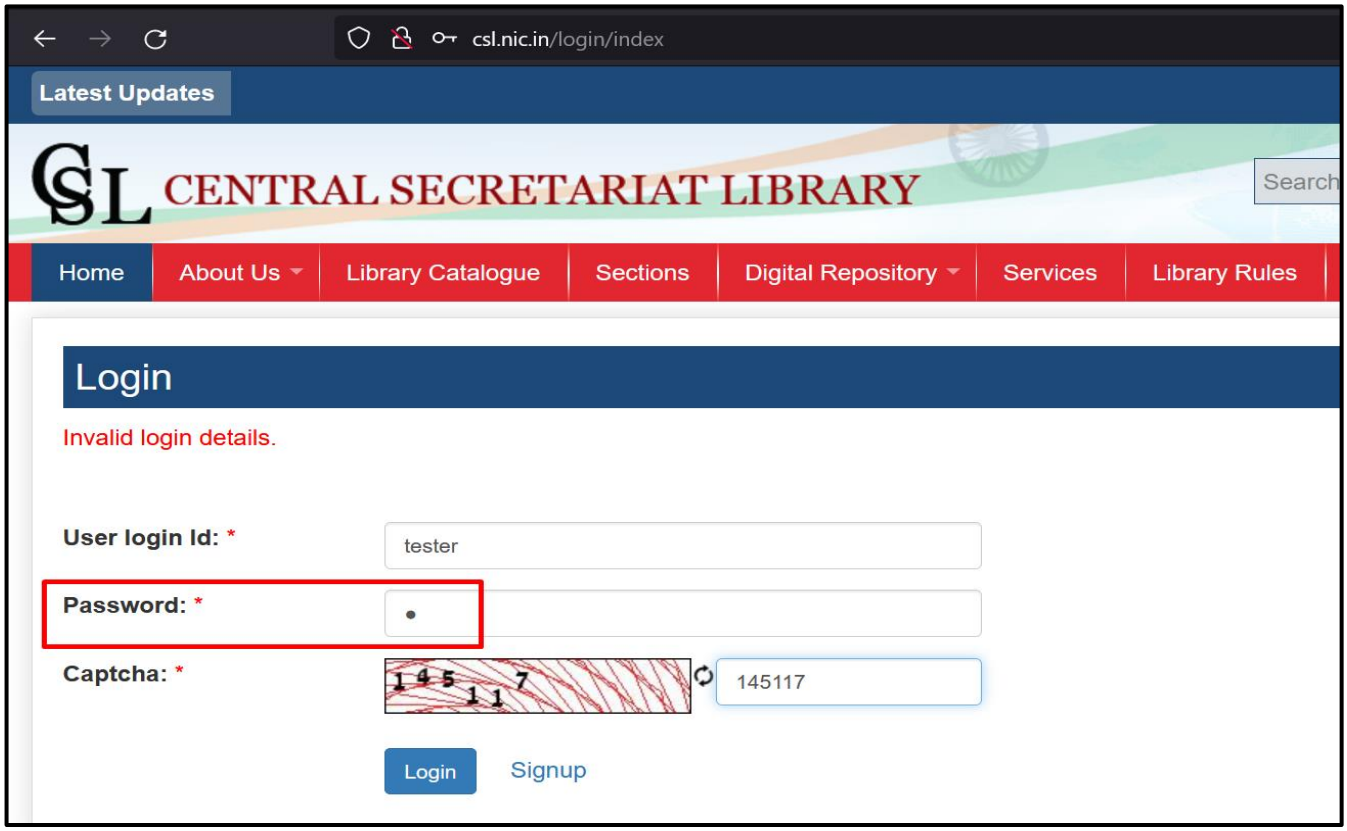


Step#2: An attacker on same network captures the request in HTTP interceptor and copy the salted hashed password as shown below:

Step#3: The request is forwarded, and the victim user is shown a message that the login details are invalid.



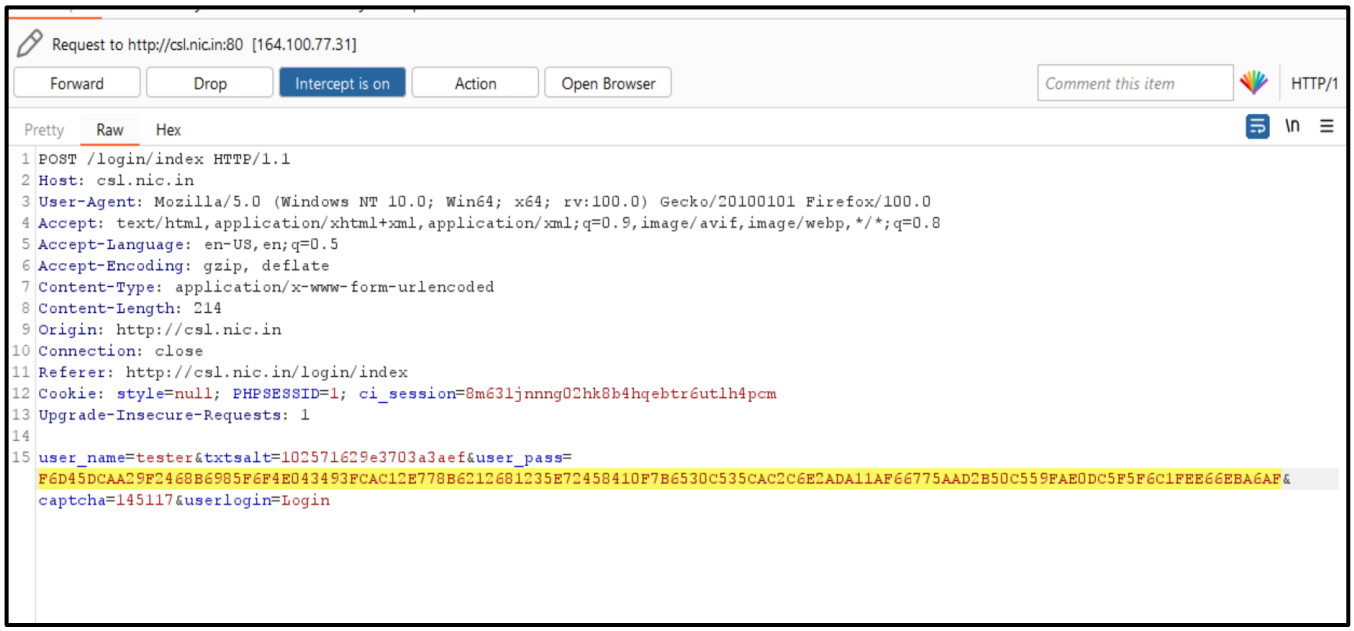
Step#4: The victim user leaves the system keeping the browser open. An attacker having physical access to victim's system enters the valid username and invalid password.



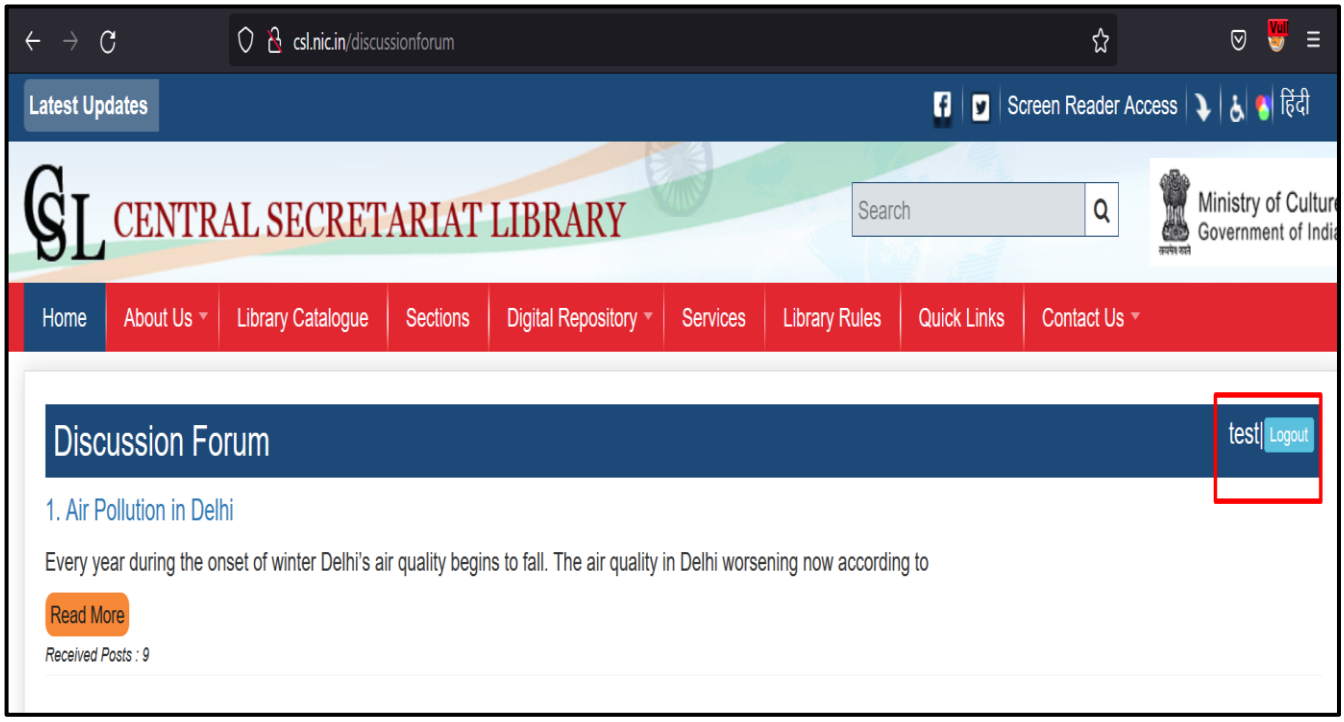
Step#5: The attacker clicks on the Login button and captures the request in an HTTP interceptor.

Original Request: The request contains the salted hashed value of invalid password.

Modified Request: The attacker replaces the salted hashed value with the stolen salted hashed password.



Step#6: The request is forwarded and the attacker is successfully logged into the victim's account.



Same vulnerability is also exists in the URL:

http://csl.nic.in/loginh/index

http://csl.nic.in/admin/

Recommendation(s):

1. Password and Sensitive data should travel in SHA256/512 or encrypted form respectively.
2. Password should be always hashed with random salt and salt should be unique for every request.
3. Salt should be generated at server side and properly validated.

2. Open File Upload

Incident URL: http://csl.nic.in/library-application

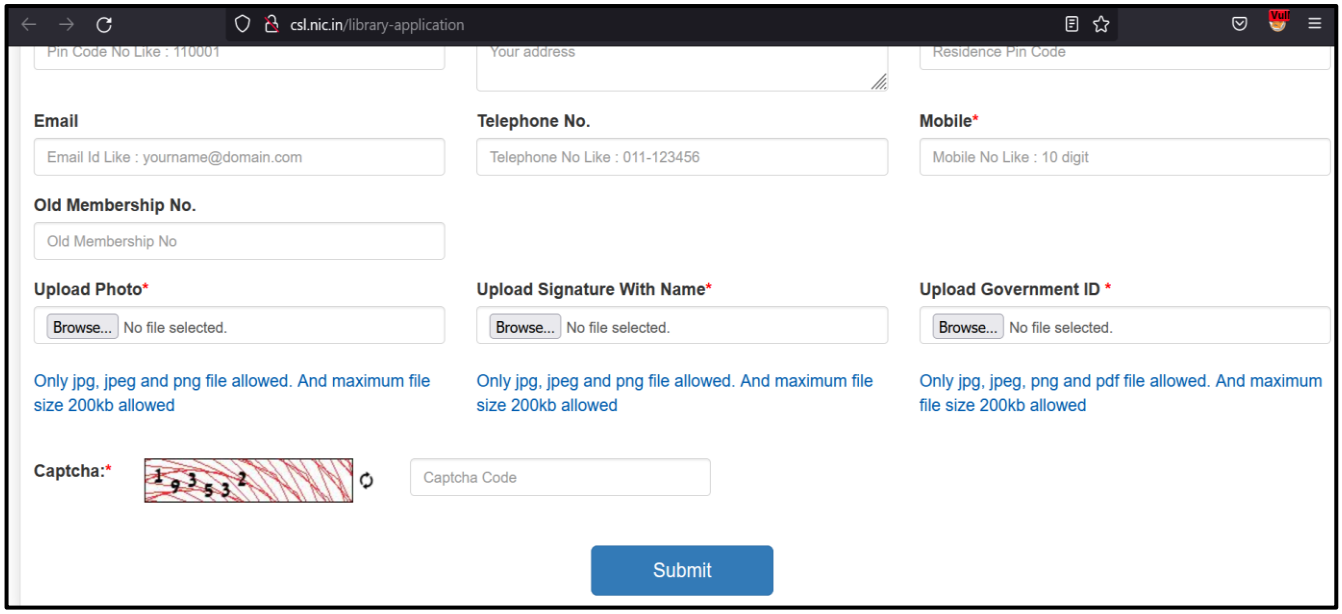
Description: An attacker can directly upload file without any authentication.

Impact: It can take leverage to upload and execute the malicious file into the application server system directly.

Severity: High

How to Test:

Step#1: An attacker navigates to the Library Membership page of the application at URL: http://csl.nic.in/library-application and it is observed that application has an open file upload as shown below:



Same Vulnerability also exists in:
http://csl.nic.in/library-membership-form-hi

Recommendation(s):

1. Application should not allow the public to upload files directly without any restriction.

3. CAPTCHA Bypass

Incident URL: http://csl.nic.in/online-suggestions

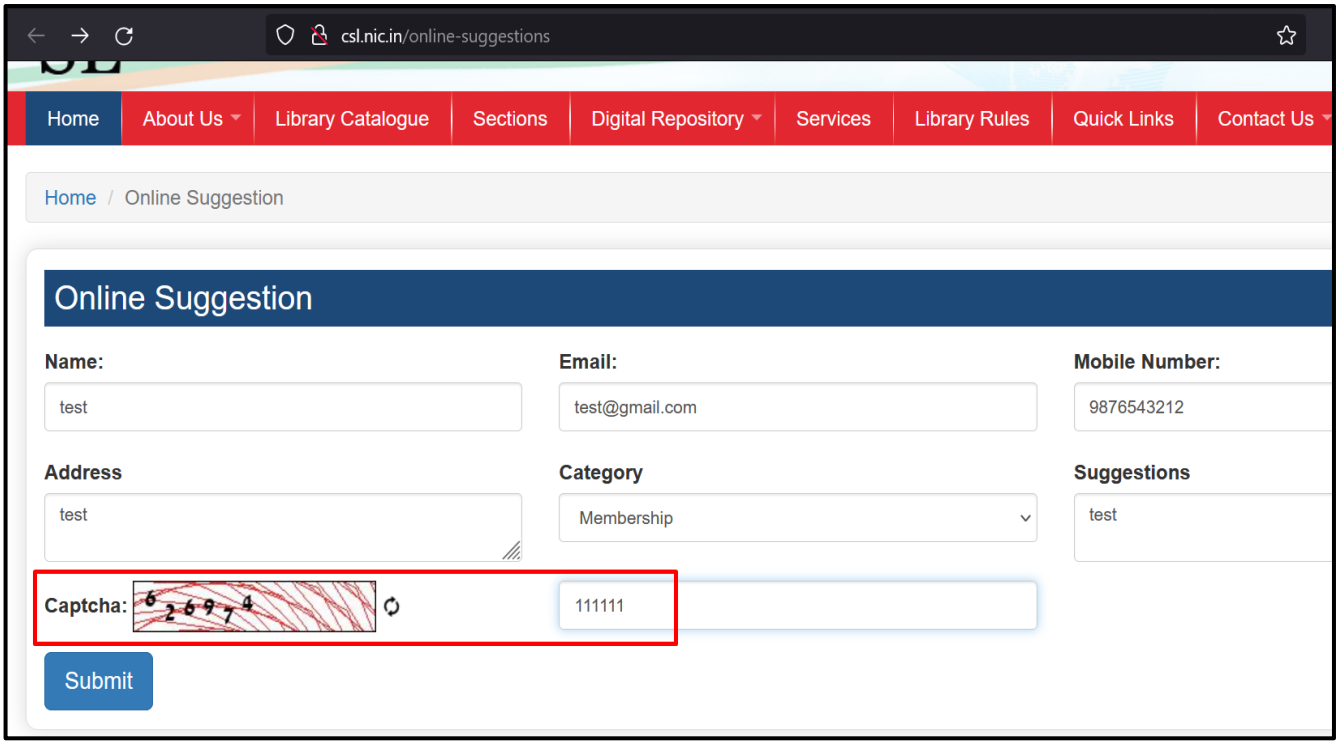
Description: The application has improperly implemented CAPTCHA functionality which can be bypassed.

Impact: An attacker can successfully bypass the CAPTCHA value and perform brute force attack/denial of service attack to the application.

Severity: High

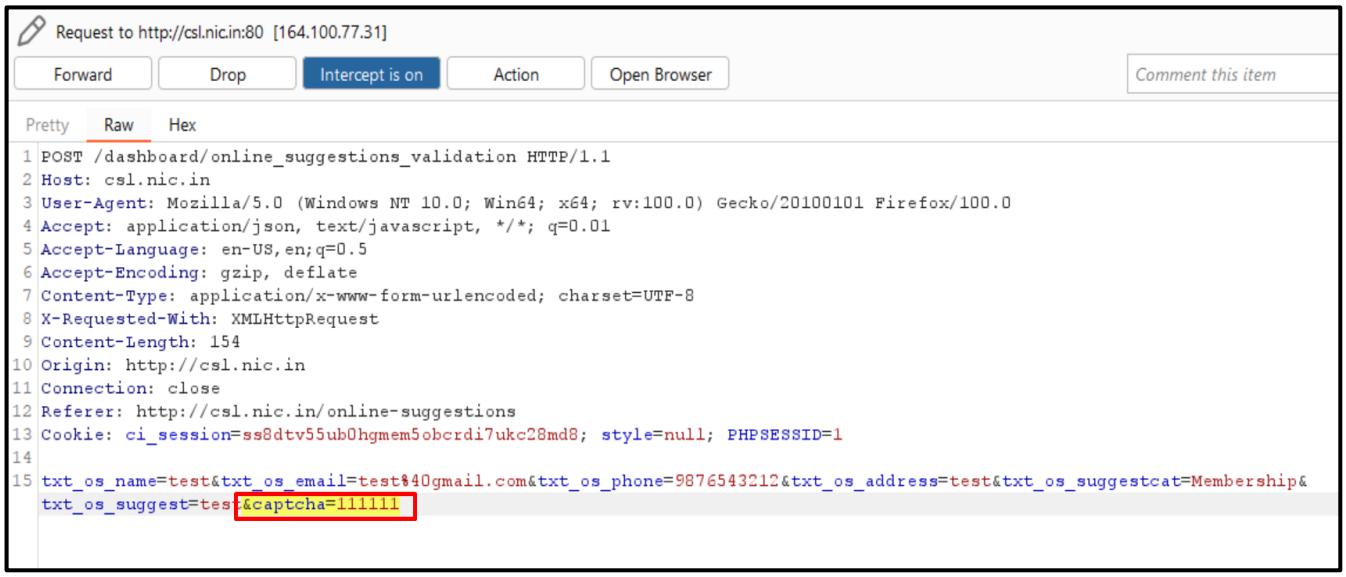
How to Test:

Step#1: An attacker navigates to application URL: http://csl.nic.in/online-suggestions and enters the details with a random CAPTCHA as shown below:



Step#2: The 'Submit' button is clicked and the request is captured in an HTTP interceptor.

Original Request: The request contains the CAPTCHA name value parameter.

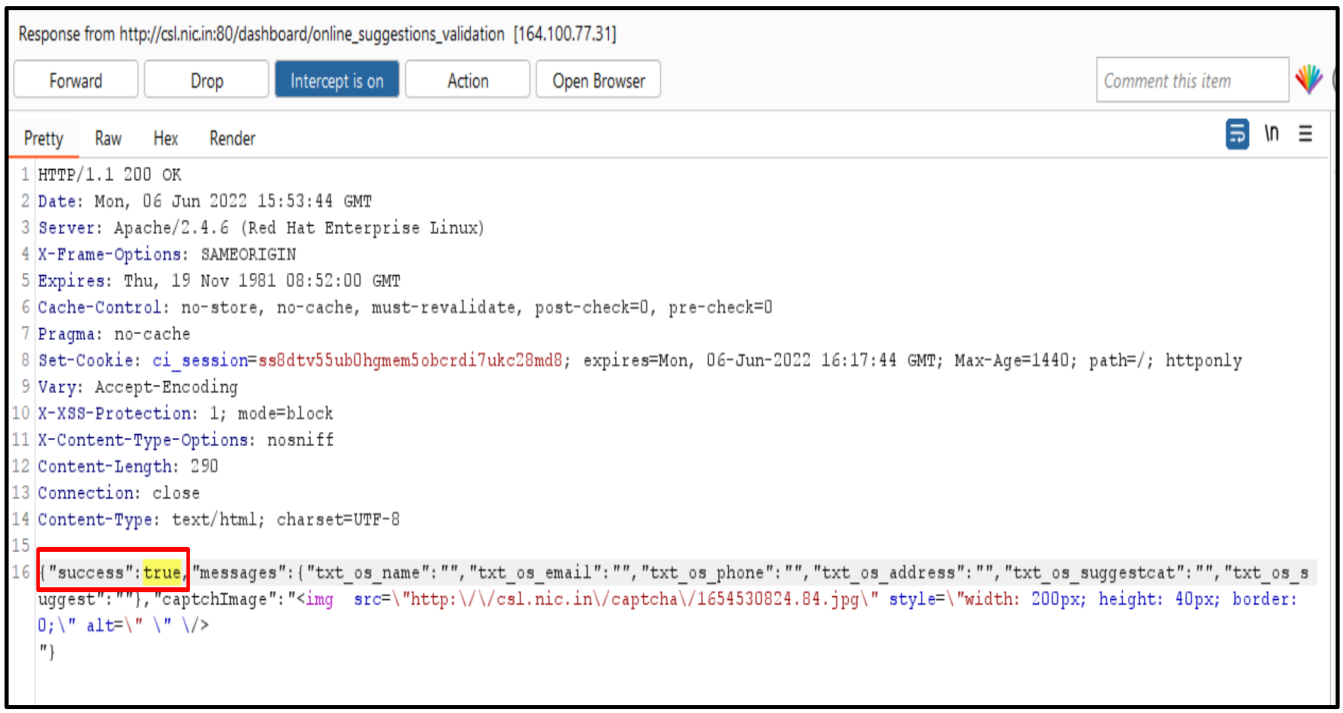


Modified Request: The attacker removes the CAPTCHA name value parameter.

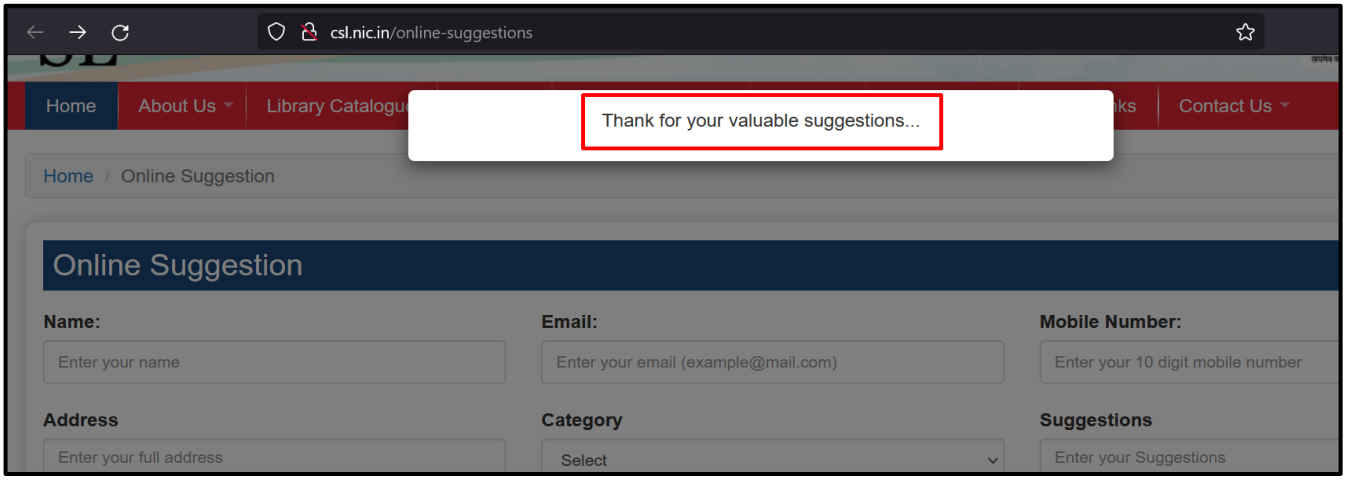
Step#3: The request is forwarded, and the response is captured.

Original Request: The response contains 'false' value.

Modified Request: The attacker modifies the 'false' value to 'true'.



Step#4: The attacker forward the response and form is successfully submitted.



Same vulnerability also exists in the following URL:

http://csl.nic.in/feedback

http://csl.nic.in/enquiry

http://csl.nic.in/ask-librarian

http://csl.nic.in/library-application

http://csl.nic.in/online-suggestions-hindi

http://csl.nic.in/login

http://csl.nic.in/login/signup

http://csl.nic.in/ask-librarian_hi

http://csl.nic.in/feedback/hindi

http://csl.nic.in/library-membership-form-hi

http://csl.nic.in/plan-your-visit-hi

http://csl.nic.in/admin/

Recommendation(s):

1. Kindly implement CAPTCHA on public available forms.
2. **CAPTCHA Specifications:**

a) CAPTCHA should be of 6 characters alphanumeric in length.

b) CAPTCHA should be case-sensitive.

c) CAPTCHA should be image-based.

d) CAPTCHA should be randomly generated from the server and not from client side.

e) After each incorrect user credential, the server should return Login page with a new CAPTCHA

4. Critical Information Disclosure

Incident URL: http://csl.nic.in/test.php

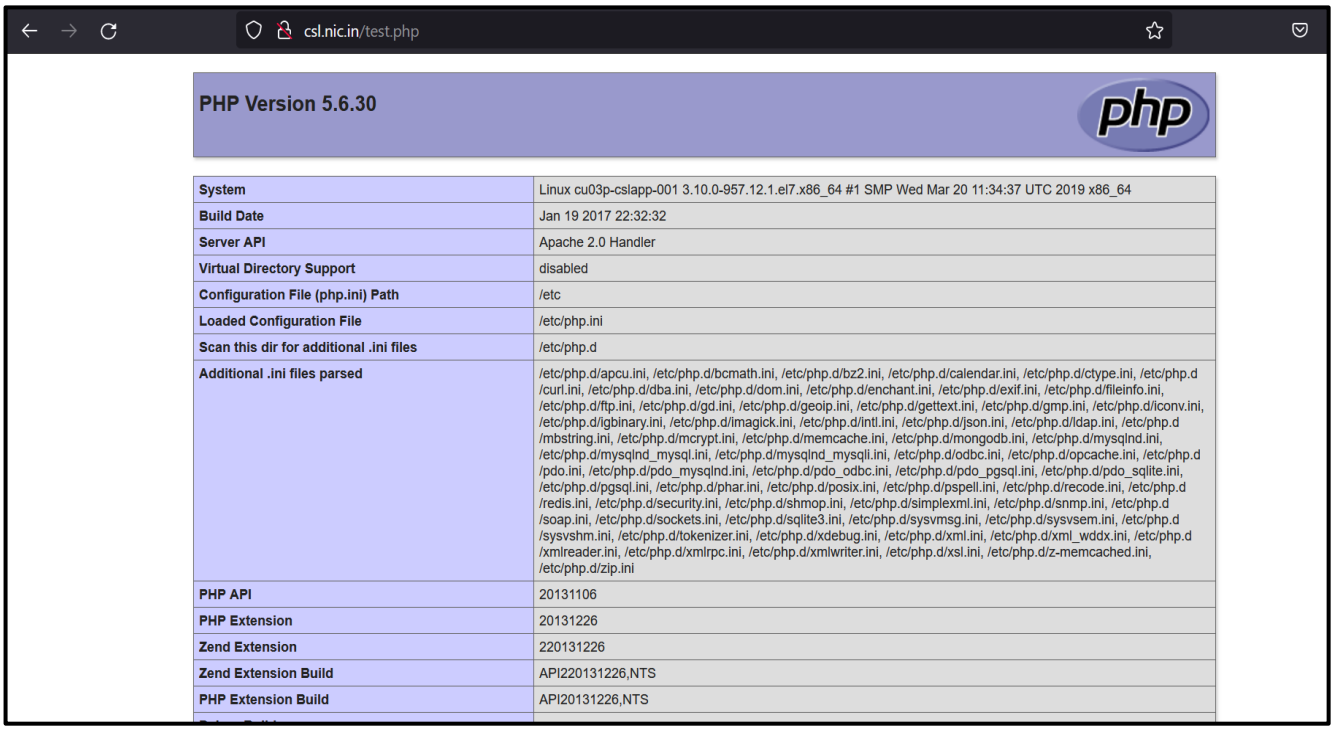
Description: An attacker can gain server-side information like server name, configuration file path from php file.

Impact: An attacker can gain server-side information like server name from php file of the application that may allow an attacker to carry out specific known targeted attacks for remote server.

Severity: High

How to Test:

Step 1: Enter the URL: http://csl.nic.in/test.php



System	Linux cu03p-cslapp-001 3.10.0-957.12.1.el7.x86_64 #1 SMP Wed Mar 20 11:34:37 UTC 2019 x86_64
Build Date	Jan 19 2017 22:32:32
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc
Loaded Configuration File	/etc/php.ini
Scan this dir for additional .ini files	/etc/php.d
Additional .ini files parsed	/etc/php.d/apcu.ini, /etc/php.d/bcmath.ini, /etc/php.d/bz2.ini, /etc/php.d/calendar.ini, /etc/php.d/ctype.ini, /etc/php.d/curl.ini, /etc/php.d/dba.ini, /etc/php.d/dom.ini, /etc/php.d/enchant.ini, /etc/php.d/exif.ini, /etc/php.d/fileinfo.ini, /etc/php.d/ftp.ini, /etc/php.d/gd.ini, /etc/php.d/geoip.ini, /etc/php.d/gettext.ini, /etc/php.d/gmp.ini, /etc/php.d/iconv.ini, /etc/php.d/igbinary.ini, /etc/php.d/imagick.ini, /etc/php.d/intl.ini, /etc/php.d/json.ini, /etc/php.d/ldap.ini, /etc/php.d/mbstring.ini, /etc/php.d/mcrypt.ini, /etc/php.d/memcache.ini, /etc/php.d/mongodb.ini, /etc/php.d/mysqlnd.ini, /etc/php.d/mysqlnd_mysql.ini, /etc/php.d/mysqlnd_mysqli.ini, /etc/php.d/odbc.ini, /etc/php.d/opcache.ini, /etc/php.d/pdo.ini, /etc/php.d/pdo_mysqlnd.ini, /etc/php.d/pdo_odbc.ini, /etc/php.d/pdo_pgsql.ini, /etc/php.d/pdo_sqlite.ini, /etc/php.d/pgsql.ini, /etc/php.d/phar.ini, /etc/php.d/posix.ini, /etc/php.d/pspell.ini, /etc/php.d/recode.ini, /etc/php.d/redis.ini, /etc/php.d/security.ini, /etc/php.d/shmop.ini, /etc/php.d/simplexml.ini, /etc/php.d/snmp.ini, /etc/php.d/soap.ini, /etc/php.d/sockets.ini, /etc/php.d/sqlite3.ini, /etc/php.d/sysvmsg.ini, /etc/php.d/sysvsem.ini, /etc/php.d/sysvshm.ini, /etc/php.d/tokenizer.ini, /etc/php.d/xdebug.ini, /etc/php.d/xml.ini, /etc/php.d/xml_wddx.ini, /etc/php.d/xmlreader.ini, /etc/php.d/xmlrpc.ini, /etc/php.d/xmlwriter.ini, /etc/php.d/xsl.ini, /etc/php.d/z-memcached.ini, /etc/php.d/zip.ini
PHP API	20131106
PHP Extension	20131226
Zend Extension	220131226
Zend Extension Build	API220131226,NTS
PHP Extension Build	API20131226,NTS

Recommendation(s):

1. Application should not disclose the server information from any page or request or response headers.
2. Application should not disclose sensitive files publicly.

5. Possible SQL Injection

Incident URL: http://csl.nic.in/archives

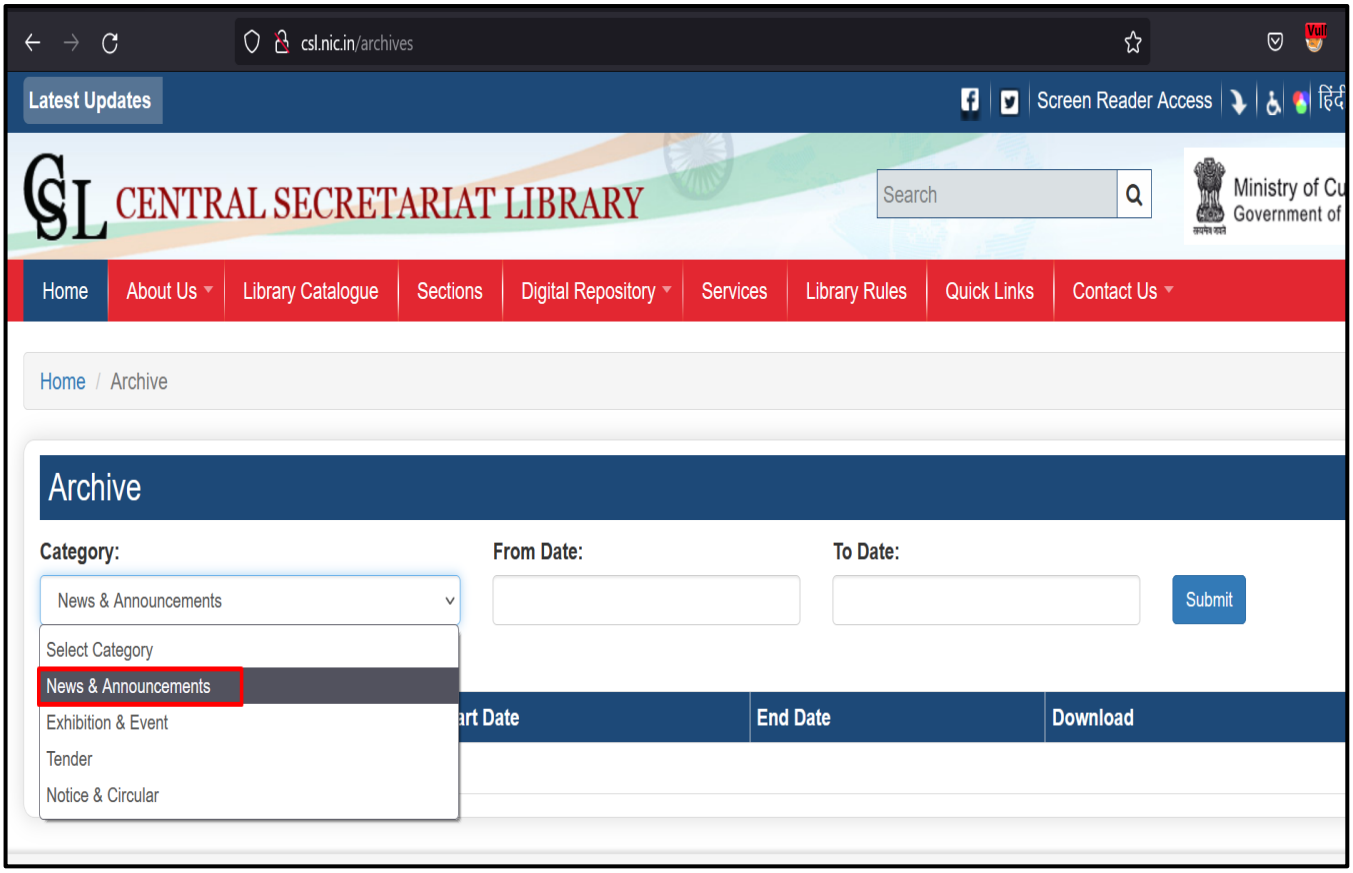
Description: An attacker can perform SQL Injection attack.

Impact: An attacker can perform attack may result in the unauthorized viewing of user lists, the deletion of entire tables and, in certain cases, the attacker gaining administrative rights to a database, all of which are highly detrimental to a business.

Severity: High

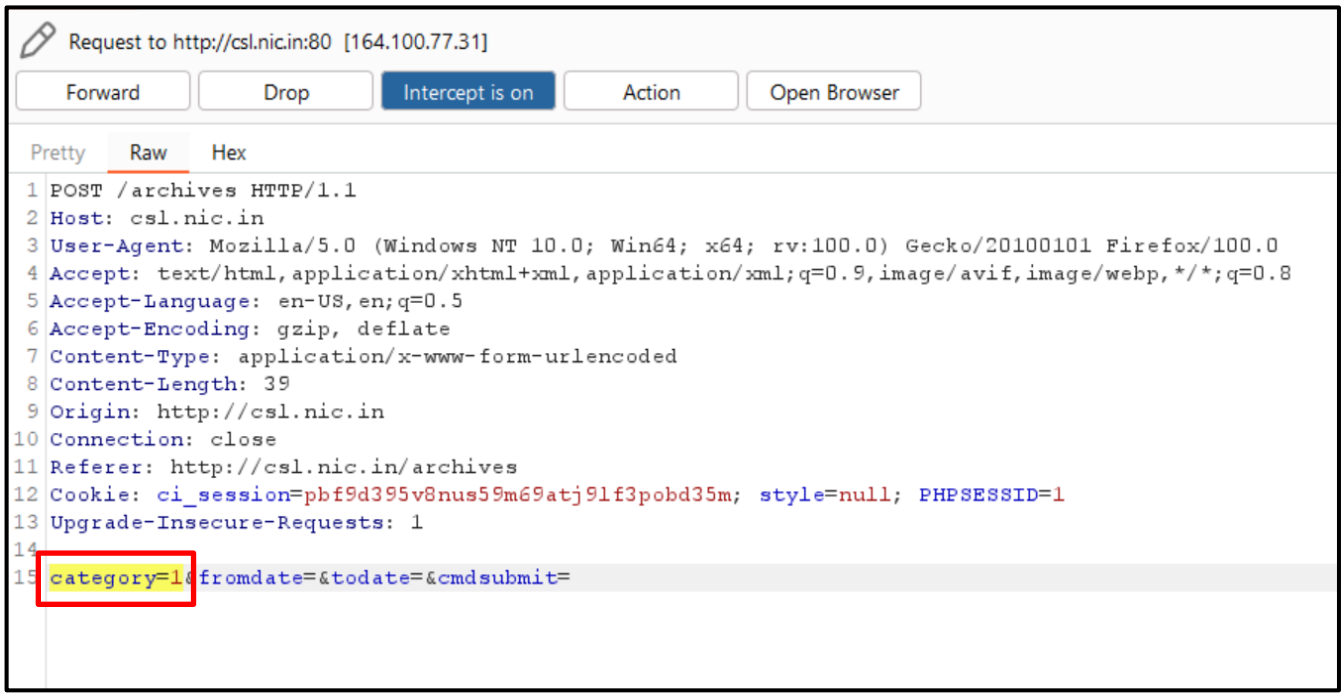
How to Test:

Step#1: An attacker navigates to application URL: http://csl.nic.in/archives/ and select the category shown below:



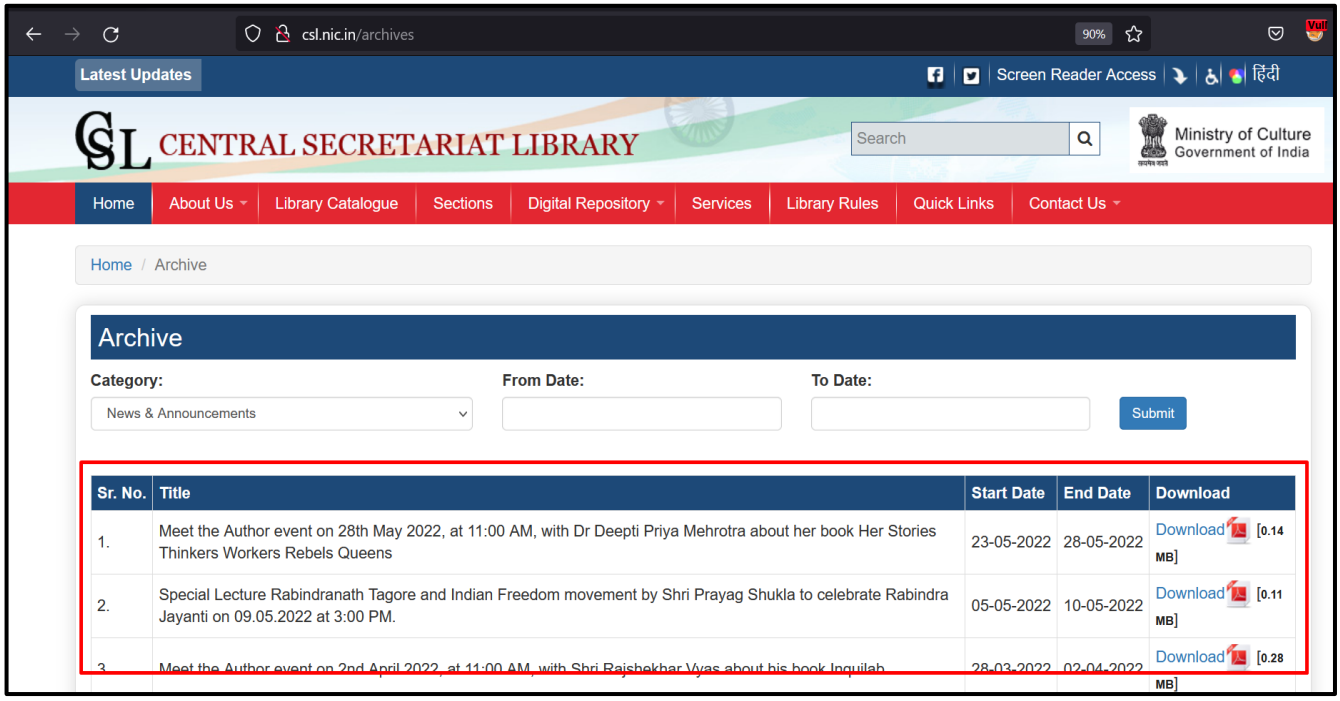
Step#2: The 'Submit' button is clicked and the request is captured in an HTTP interceptor.

Original Request: The request contains the category parameter with value as '1'.



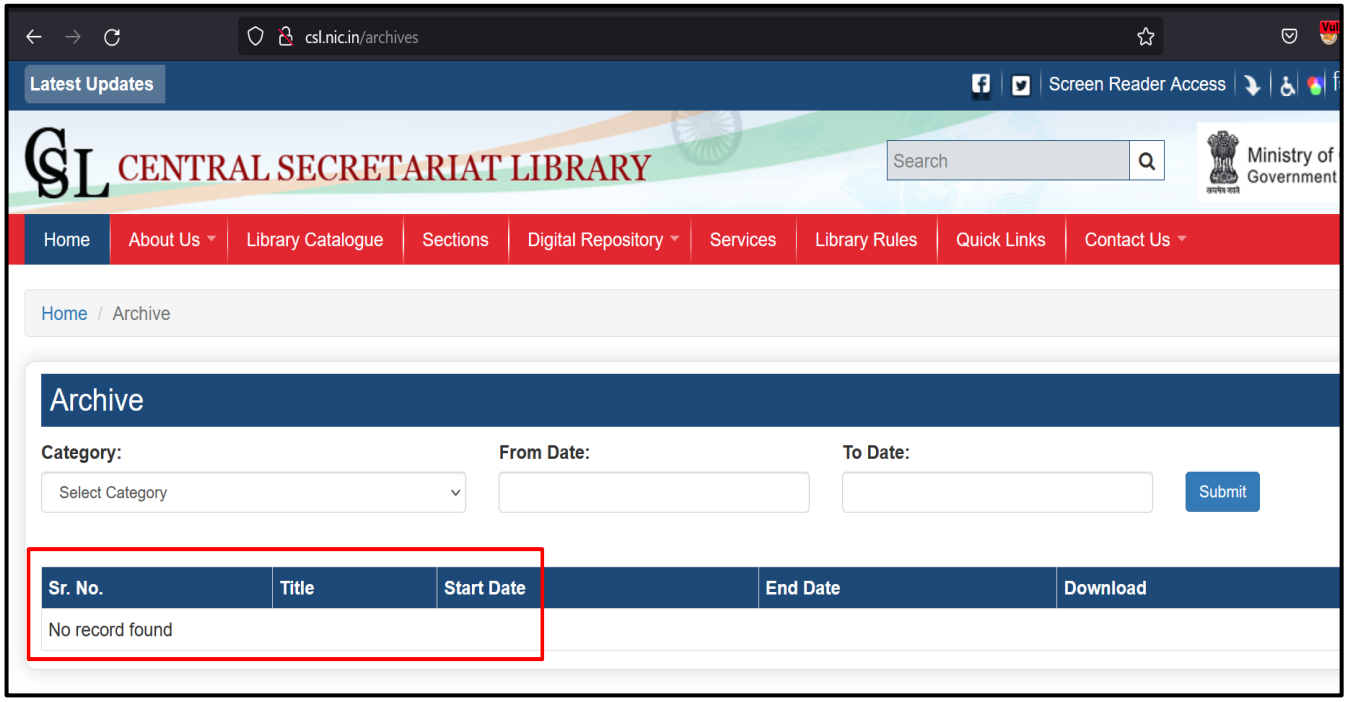
Modified Request#1: The attacker appends a single quote (') to the value of the category parameter.

Step#3: The request is forwarded and the application displays true results.



Modified Request#2: The attacker appends a single quote (‘) to the value of the category parameter.

Step#4: The request is forwarded, and the application displays no result.



Recommendation(s):

1. Server-side validation for all inputs.
2. Use parameterized stored procedure so that all supplied parameters are treated as data, rather than potentially executable queries.
3. The web application should connect to the database using a low privileged database user account.
4. The application should implement error handling for all application generated errors.

6. Directory Listing

Incident URL: http://csl.nic.in/icons/

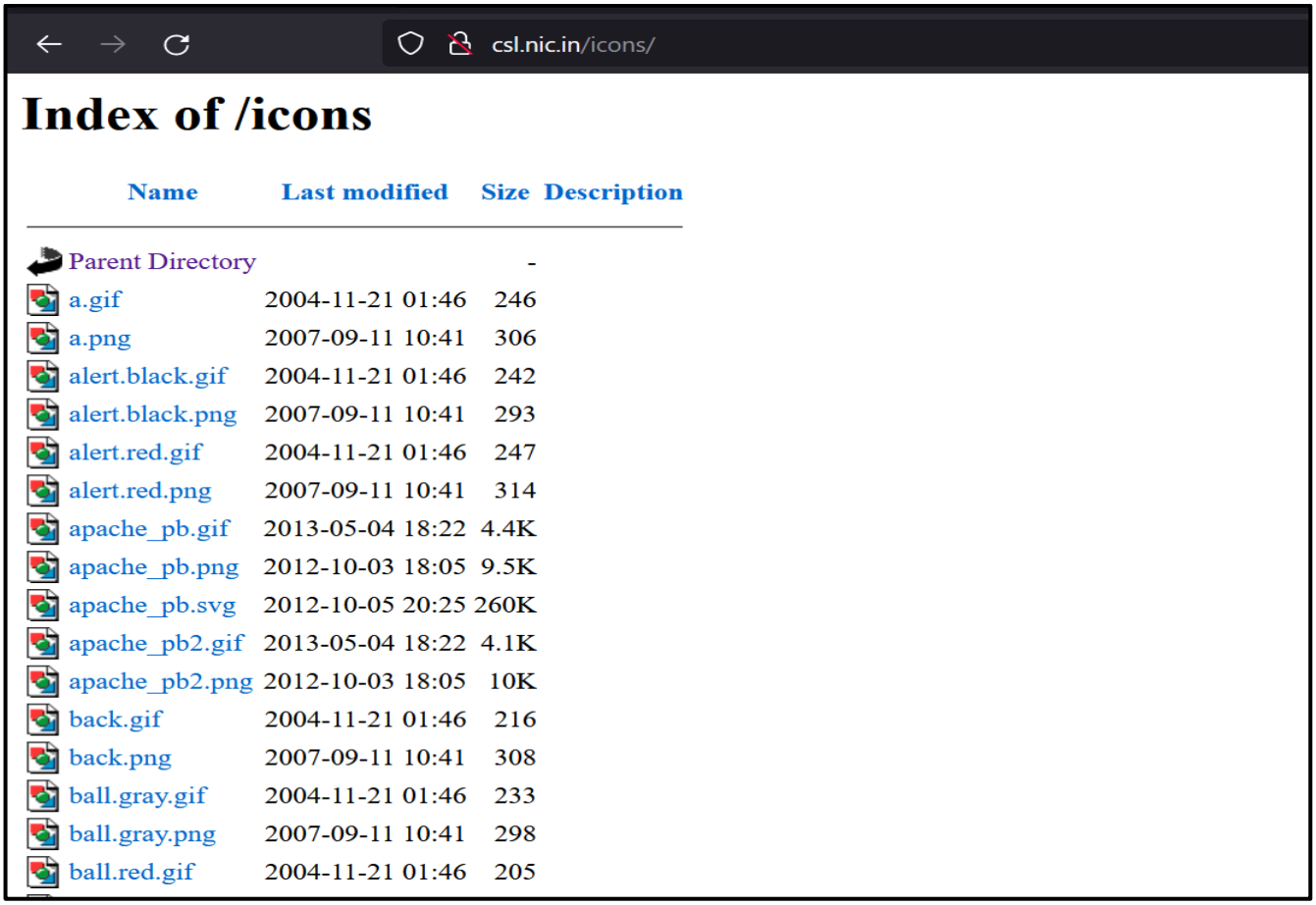
Description: Directory listing is possible in the application.

Impact: It allows an attacker to reveal sensitive information to carry out known attacks for that server.

Severity: High

How to Test:

Step#1: Upon entering the URL: http://csl.nic.in/icons, the application displays the directories.



Recommendation(s):

1. Directory Listing should be disabled for all directories in the website.

7. Unencrypted Communication

Incident URL: http://csl.nic.in/

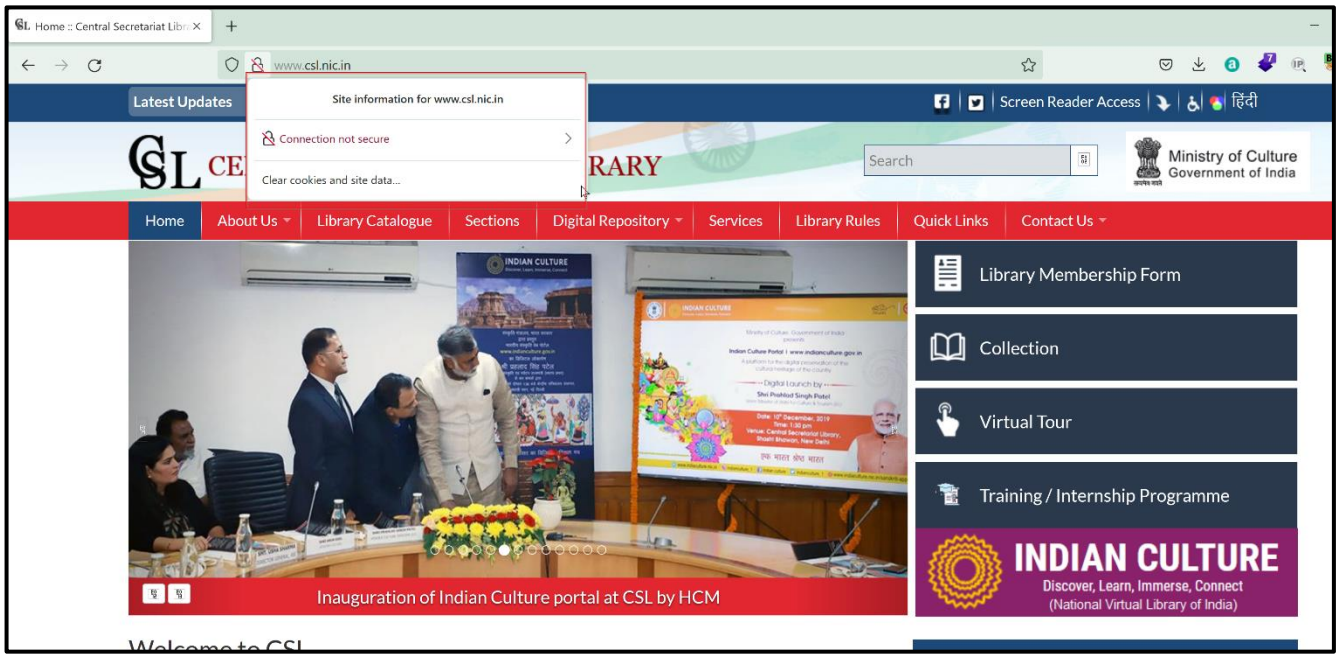
Description: The application is accessible over HTTP.

Impact: The application is working on HTTP as a result an attacker can perform the MITM attack and can steal sensitive information during transmission.

Severity: Medium

How to Test:

Step#1: The attacker navigates to the application URL: http://csl.nic.in/ and it is observed that application is running on HTTP:



Recommendation(s):

1. As per NIC policy, all the website and web application must be deployed on the HTTPS.

8. CORS Misconfiguration

Incident URL: http://csl.nic.in/

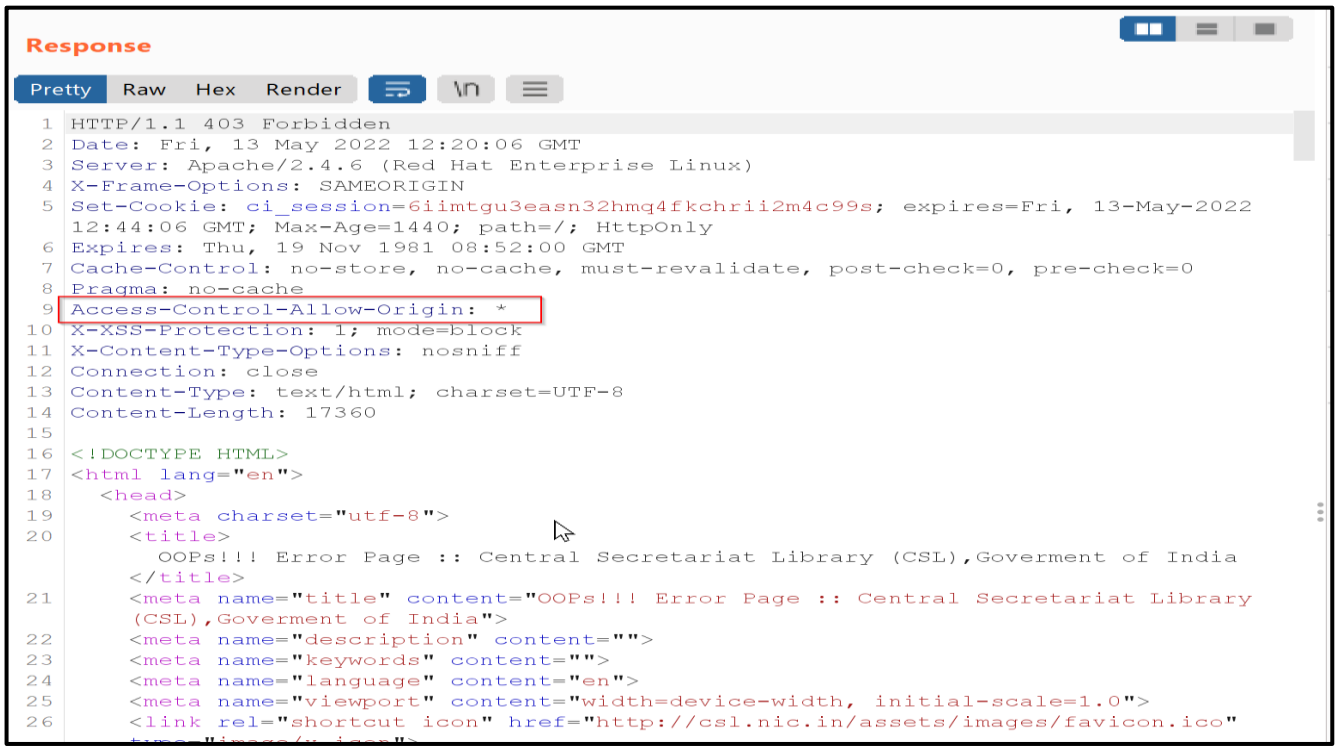
Description: The application is allowing access control requests originating from different domains.

Impact: An attacker can misuse the cross-origin resource sharing policy to carry out privileged actions and retrieve sensitive information.

Severity: Medium

How to Test:

Step#1: An attacker navigates to the main page of the application at URL: https://ayusoft.ayush.gov.in and intercepts the response on the HTTP Interceptor. It is observed that Access-Control-Allow-Origin header is set to * as shown below. This proves the application is allowing access control requests originating from different domains.



Recommendation(s):

1. Avoid setting header Access-Control-Allow-Origin' to *, if the resource contains sensitive information.
2. Allow only selected, trusted domains in the Access-Control-Allow-Origin header.
3. Ensure that URLs responding with *Access-Control-Allow-Origin: ** do not include any sensitive content or information that might aid attacker in further attack.

9. Insecure HTTP Methods

Incident URL: http://csl.nic.in/

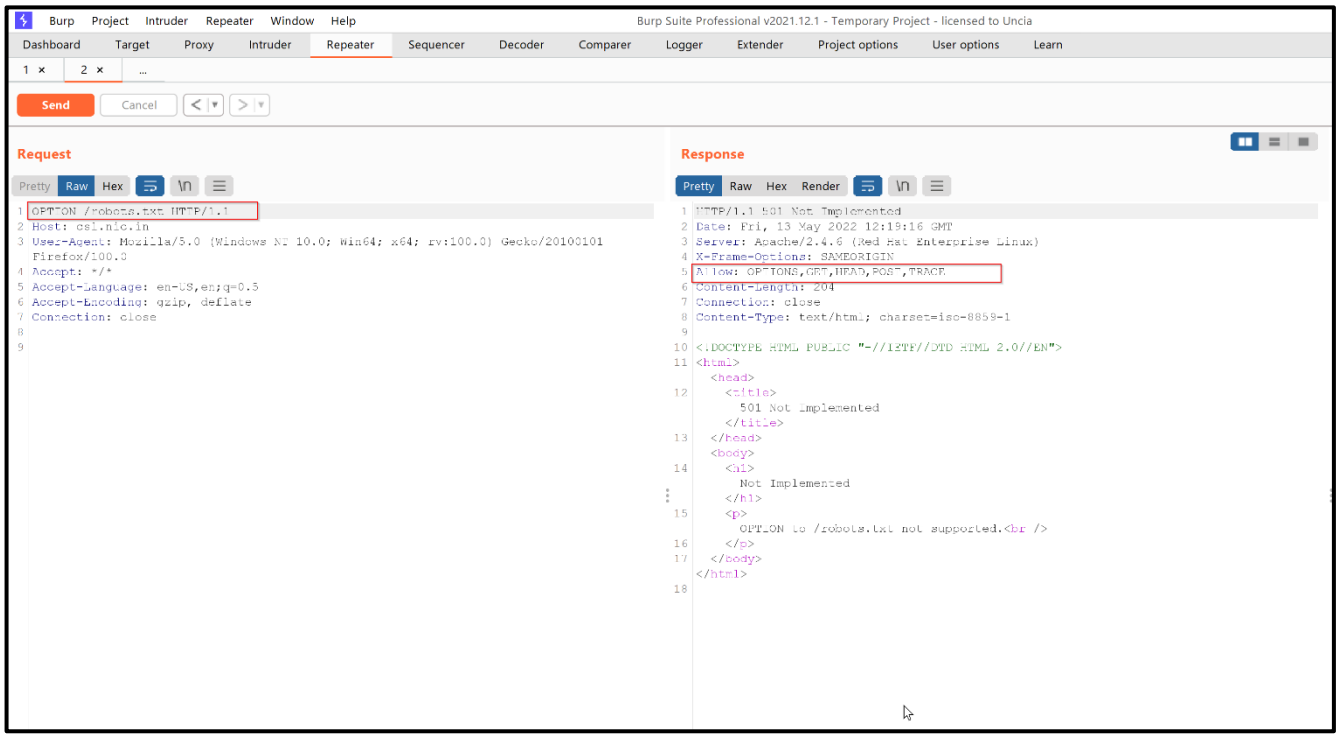
Description: The insecure 'OPTIONS' HTTP method is enabled in the application.

Impact: It allows an attacker to perform further attacks based on the HTTP methods which are enabled.

Severity: Low

How to Test:

Step#1: An attacker browses the application and intercepts the request with the help of HTTP interceptor as shown below.



Recommendation(s):

1. It is recommended to allow only GET, POST and HEAD HTTP methods.

10.Insecure Cookie Attribute

Incident URL: http://csl.nic.in/

Description: The application has not implemented 'Domain' and 'Secure' cookie attributes and 'Path' has been improperly set to root.

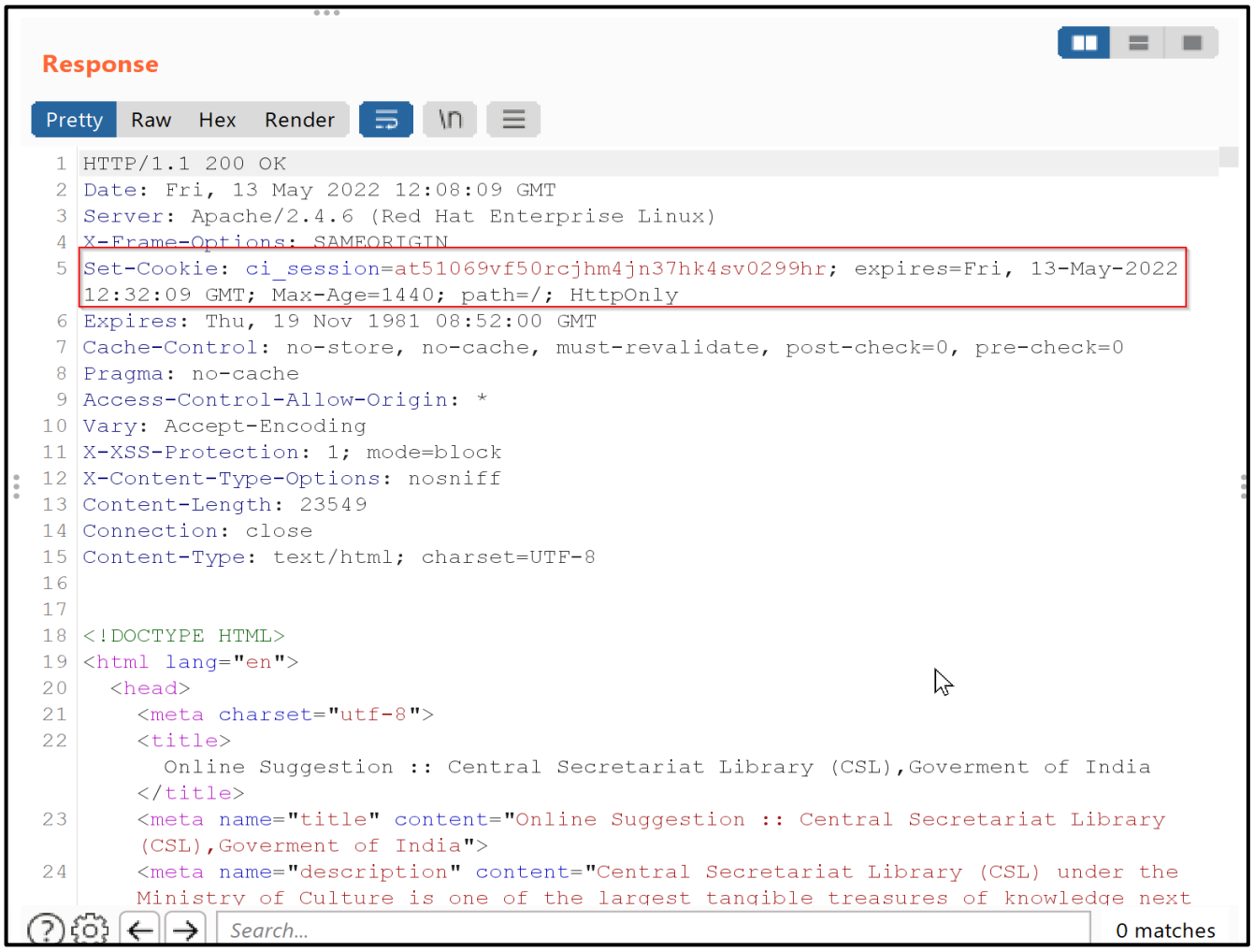
Impact: This vulnerability allows an attacker to steal sensitive information like session token and allows launching further attacks.

Severity: Low

How to Test:

Step#1: The attacker navigates to the application at URL: http://csl.nic.in/ and captures the request.

Step#2: The request is captured and is observed that the only Domain cookie attribute is not set and path is set to root as shown below.



Recommendation(s):

1. Path' should not set to root Instead a sub folder path should be used.
2. Also, the "Domain" cookie attribute should be set as restrictive as possible and the complete application domain should be provided for this attribute.
3. The application should set 'Domain', 'Secure' and 'Path' for both pre authentication Session ID and post authentication Session ID in the application.

11. Private IP Disclosure

Incident URL: http://csl.nic.in/upload/'

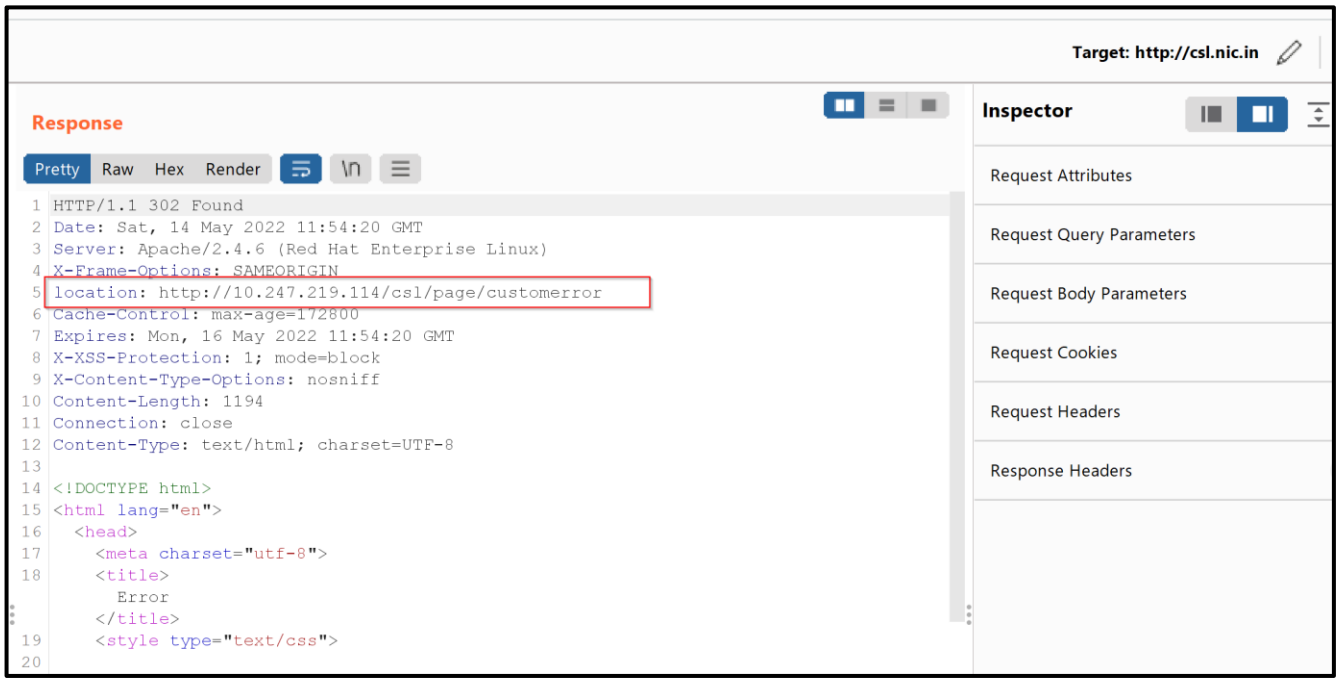
Description: An attacker can gain the critical information as the Internal IP is disclosed in the application.

Impact: It allows an attacker to gain knowledge about the internal IP and launch further attacks.

Severity: Low

How to Test:

Step#1: The attacker navigates to the application at URL: http://csl.nic.in/upload/' and captures the request in HTTP interceptor and observes internal IP in response as shown below.



Recommendation(s):

1. The application should not display server specific information to the application users.
2. Internal IP addresses should never be present in the HTTP requests or responses of the application.

12. Information Disclosure

Incident URL: http://csl.nic.in/

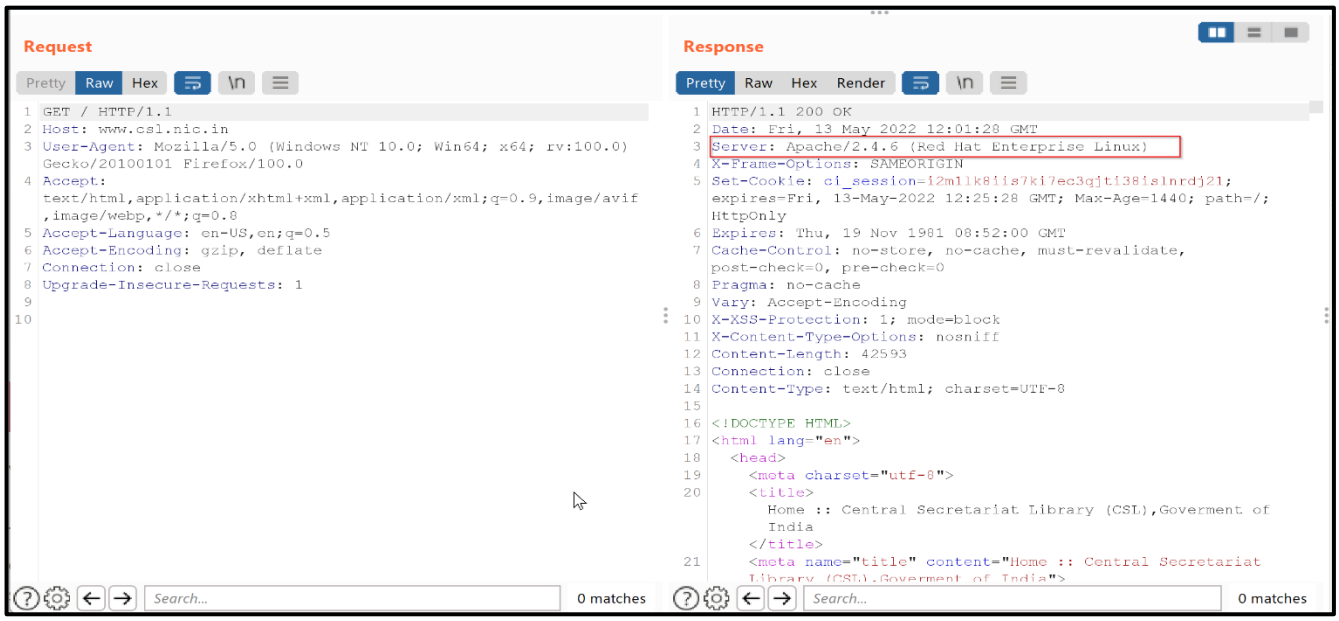
Description: An attacker can gain server-side information like server name from the HTTP response of the application.

Impact: An attacker can gain server-side information like server name in response of the application that may allow an attacker to carry out specific known targeted attacks for remote server.

Severity: Low

How to Test:

Step#1: An attacker navigates to the main page of the application at URL: http://csl.nic.in/ and intercept the request on the HTTP Interceptor. It is observed that server name is disclosing i.e., **Apache/2.4.6 (Red Hat Enterprise Linux)** is disclosed as shown below.



Recommendation(s):

1. Application should not disclose the server information from any page or request or response headers.
2. The application must use latest stable version and update on regular basis.

13. Vulnerable and Outdated Component

Incident URL: http://csl.nic.in/

Description: The application is using a vulnerable version of jQuery v3.1.1, Bootstrap v3.3.7

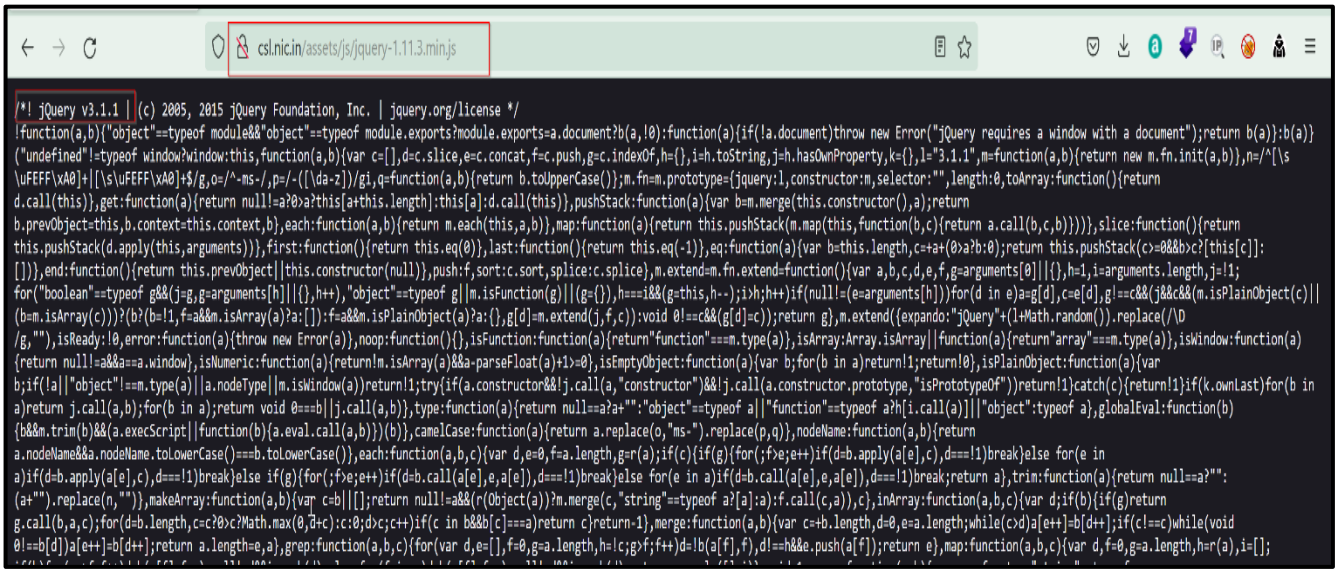
Impact: This vulnerability represents a risk to the application as the application is using vulnerable version of jQuery and Bootstrap.

Severity: Low

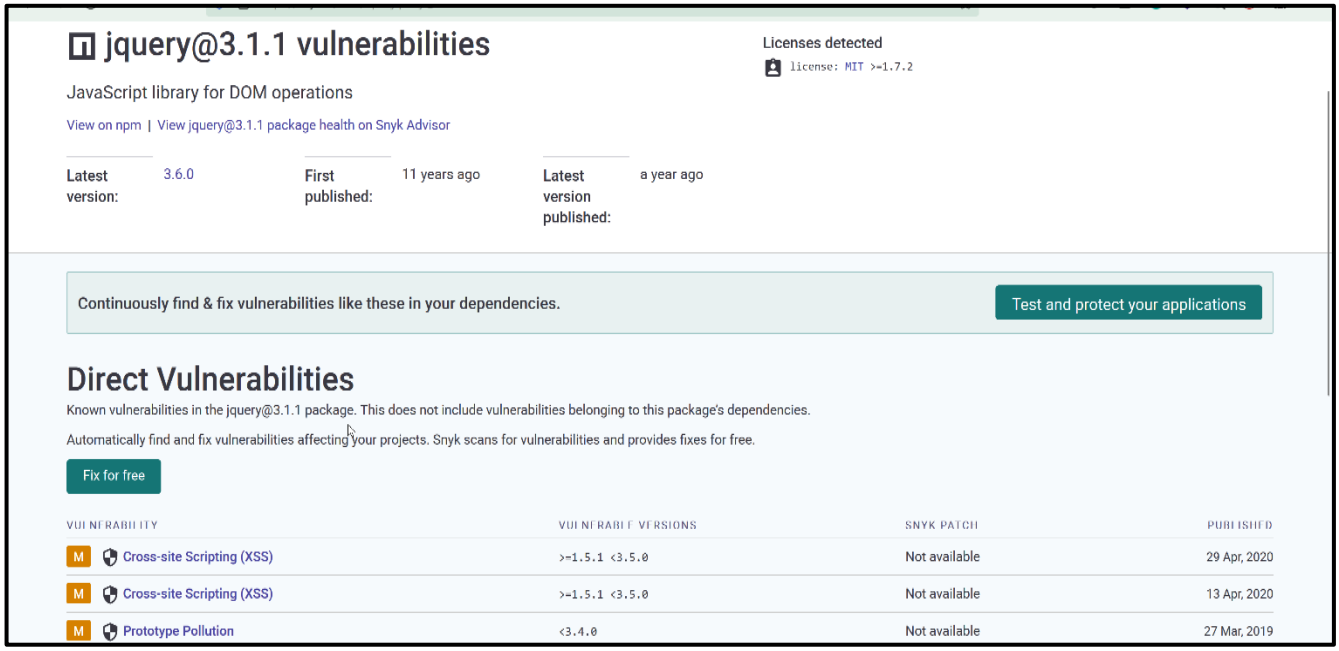
How to Test:

Instance#1:

Step#1: An attacker navigates to the application at URL: http://csl.nic.in/assets/js/jquery-3.1.1.min.js and observes that the application is using the vulnerable version of jQuery as shown below.



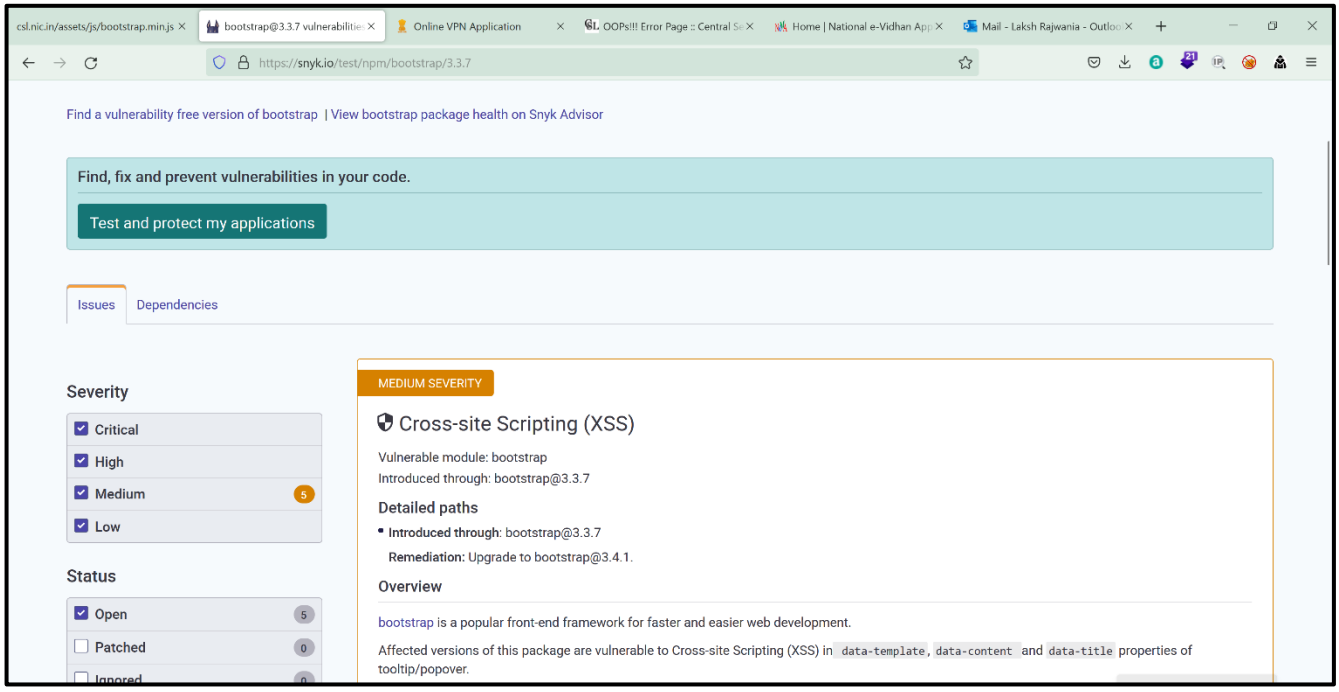
Step#2: From the below screenshot it is observed that jQuery version 3.1.1 is vulnerable.



Instance#2:

Step#1: An attacker navigates to the application at URL: http://csl.nic.in/assets/js/bootstrap.min.js and observes that the application is using the vulnerable version of Bootstrap as shown below.

Step#2: From the below screenshot it is observed that Bootstrap 3.3.7 is vulnerable.



Recommendation(s):

1. Upgrade all components (Applications Frameworks, Content Management Systems (CMS), Web server) to latest stable version.

14. Missing Security Headers

Incident URL: http://csl.nic.in/

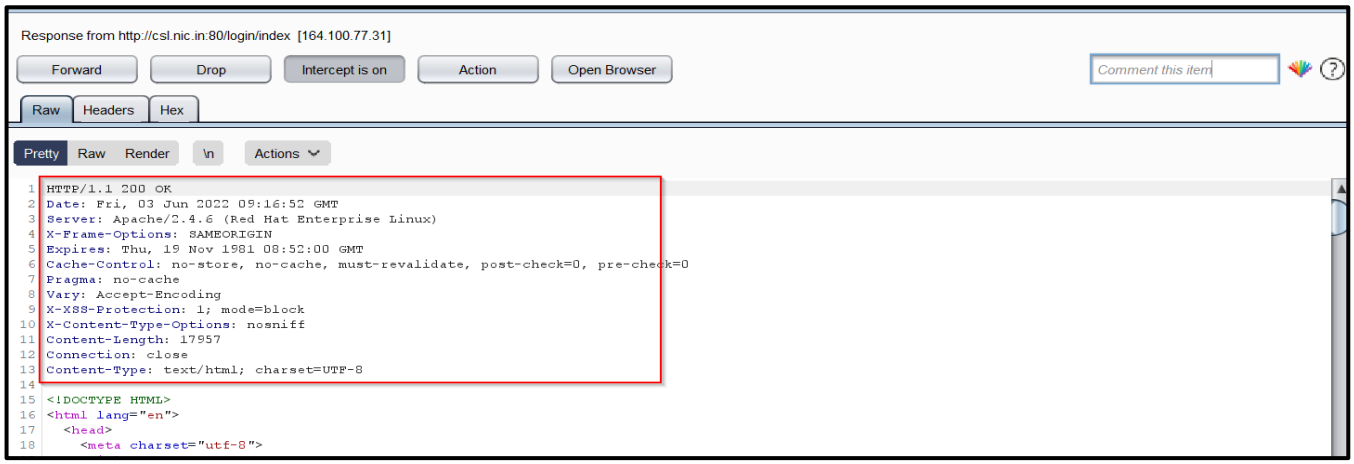
Description: The application security headers provide protection from multiple attack vectors making it difficult for an attacker to compromise or exploit the application or related assets, The application has not configured security headers.

Impact: The application has not implemented CSP and HSTS response header leading it vulnerable to multiple security attacks including Cross Site Scripting (XSS),data injection attacks and SSL-stripping man-in-the-middle attacks, weakens cookie-hijacking protections.

Severity: Low

How to Test:

Step #1: Below screenshot shows that the Security headers (i.e., CSP and HSTS) are missing from the application:



Recommendation(s):

It is recommended that all the necessary security headers should be enabled like:

- X-Content-Type-Options: nosniff
- X-Frame-Options:SAMEORIGIN
- X-XSS-Protection:1;mode=block
- Strict-Transport-Security:max-age=31536000;includeSubDomains;preload
- Application must implement CSP security header.

15. Admin Page Accessible

Incident URL: http://csl.nic.in/admin/

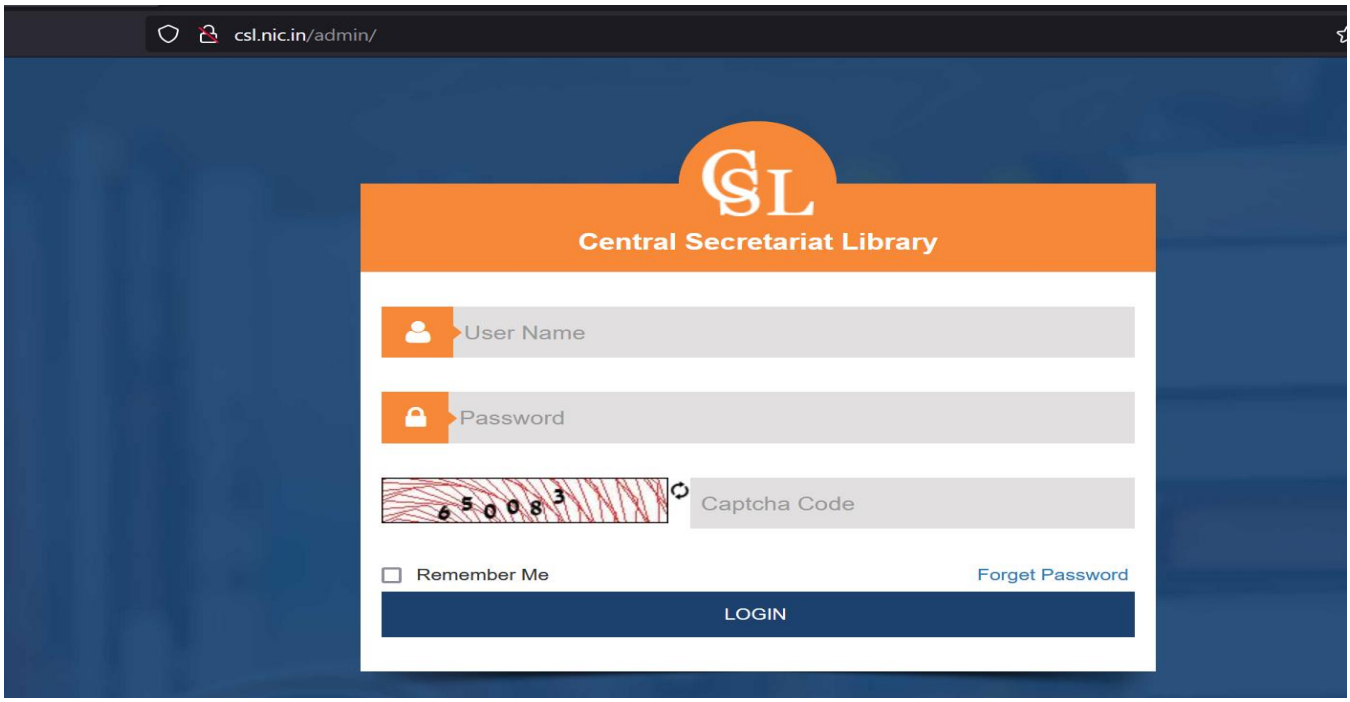
Description: The application's Admin page is directly accessible over the internet.

Impact: An attacker can successfully run an automated/Manual script to crack application user's login credentials, or this information can be used to conduct further attacks.

Severity: Low

How to Test:

Step#1: An attacker navigates to the Admin Login page of the application at the URL: http://csl.nic.in/admin/ and observed it is accessible without any restriction as shown below:



Recommendation(s):

1. There must be **restricted IP access** of admin login or **manager login.**

16. Email Harvesting

Incident URL: http://csl.nic.in/page/innerpage/library-rules.php

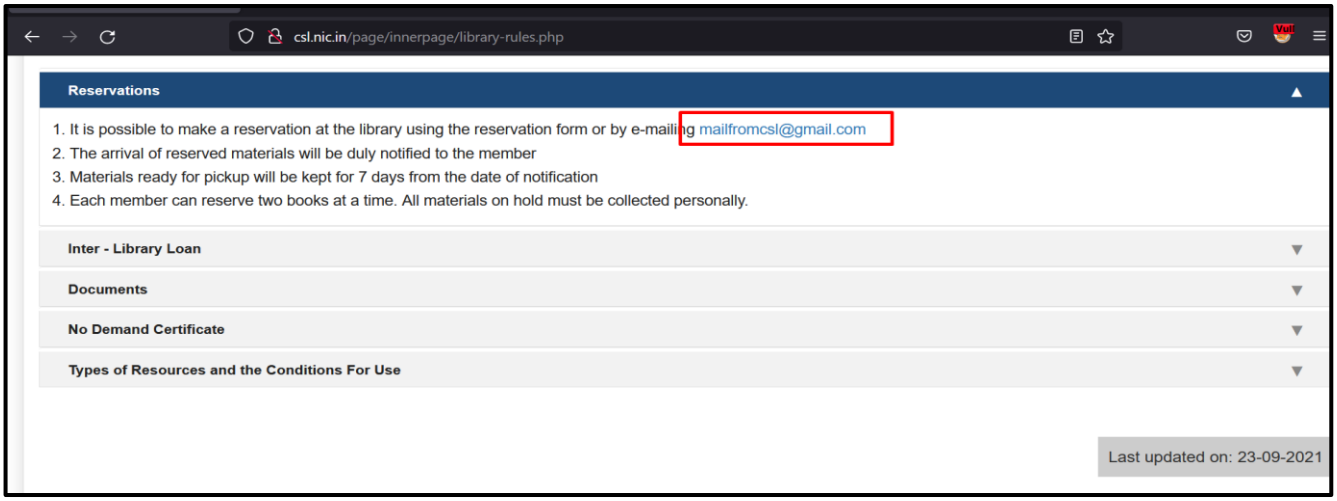
Description: The e-mail address present in the application is in text/hyperlink format thus leading to email harvesting attack.

Impact: It allows an attacker to automate and send multiple spam emails to the harvested email ids present in the application causing denial of service to the victim members.

Severity: Low

How to Test:

Step#1: The attacker navigates to application URL: http://csl.nic.in/page/innerpage/library-rules.php and observes that the application shows e-mail id as text.



Recommendation(s):

1. Email addresses should be posted as an image not as a hyperlink. Alternatively, instead of @symbol, [at] should be used. Similarly the dot character (.) should be replaced by [dot]. For example xyz@gma.com should be written as xyz[at]gma[dot]com.
2. High privilege email addresses should not posted on the website.
3. Email addresses should not be provided as a hyperlink, for example, mailto:xyz[at]gma[dot]com.

Note: Vulnerabilities should be patched throughout the website.