

# Vulnerability Assessment and Penetration Testing Report for Jeansrs Web Application

<b>Project Name</b>	Jeansrs
<b>Start &amp; End Dates</b>	12/Oct/2023 - 13/Oct/2023
<b>Analyst Name</b>	InfoSec Team
<b>Email</b>	<a href="mailto:secure@orangemantra.in">secure@orangemantra.in</a>
<b>Website</b>	<a href="https://www.jeansrs.com/">https://www.jeansrs.com/</a>
<b>Place</b>	Gurugram, Haryana
<b>Website Health State (Security)</b>	High
<b>Suggested Remediation Time</b>	As Soon As Possible

## SUMMARY OF FINDINGS

Risk	Count
Critical	1
High	4
Medium	4
Low	4
<b>Total</b>	<b>13</b>

Vulnerability or Condition Type		Risk	Severity	Retest Status
1	Session Hijacking via PHPSESSID Manipulation	Critical	Critical	
2	Input Validation Bypass	High	High	
3	Browser Back Button Vulnerability	High	High	
4	Directory Indexing	High	High	
5	Open Redirection	High	High	
6	MISSING SECURITY HEADERS	Medium	Medium	
7	Banner Grabbing	Medium	Medium	
8	CORS	Medium	Medium	
9	DMARC Record Not Found	Medium	Medium	
10	Additional Security Measures	Low	Low	
11	Known Vulnerable components	Low	Low	
12	TLS Weak ciphers	Low	Low	

13	Missing HTTP only and secure flag in cookie (Session Id)	Low	Low	
----	--	-----	-----	--

DETAILED FINDINGS

Finding #1: Session Hijacking via PHPSESSID Manipulation

Risk
Critical
Severity
Critical
Affected Components
Website
Details
<p>This bug report addresses a critical vulnerability in the session management system of our application, which allows for session hijacking through PHPSESSID manipulation. This report provides a detailed overview of the bug, its potential impact, and recommended remediation measures.</p> <p>The identified vulnerability poses significant risks and can have the following impacts:</p> <ul style="list-style-type: none"><li>a) <b>Unauthorized Access:</b> Attackers can exploit the bug by manipulating the PHPSESSID value, gaining unauthorized access to other user accounts. This can result in unauthorized actions, data exposure, and potential misuse of user accounts.</li><li>b) <b>Data Breach:</b> Session hijacking can lead to the exposure of sensitive user data, including personal information, financial details, and confidential records. This can have severe consequences for both users and our organization, including legal and regulatory implications.</li><li>c) <b>Reputational Damage:</b> Failing to protect user sessions and allowing unauthorized access undermines user trust in our application. The resulting reputational damage can have long-lasting negative effects on our brand and business.</li></ul>
Suggested Remediation
<p>To address the session hijacking vulnerability, the following remediation measures are recommended:</p> <ul style="list-style-type: none"><li>a. Implement Strong Session Management:</li></ul>

- Generate random, unique session IDs that are resistant to guessability or manipulation.
- Associate session IDs with user IDs and validate their consistency during authentication and authorization processes.
- Consider implementing session regeneration after successful authentication.
- Enforce session expiration to limit the lifespan of user sessions.

b. Ensure Secure Transmission and Storage:

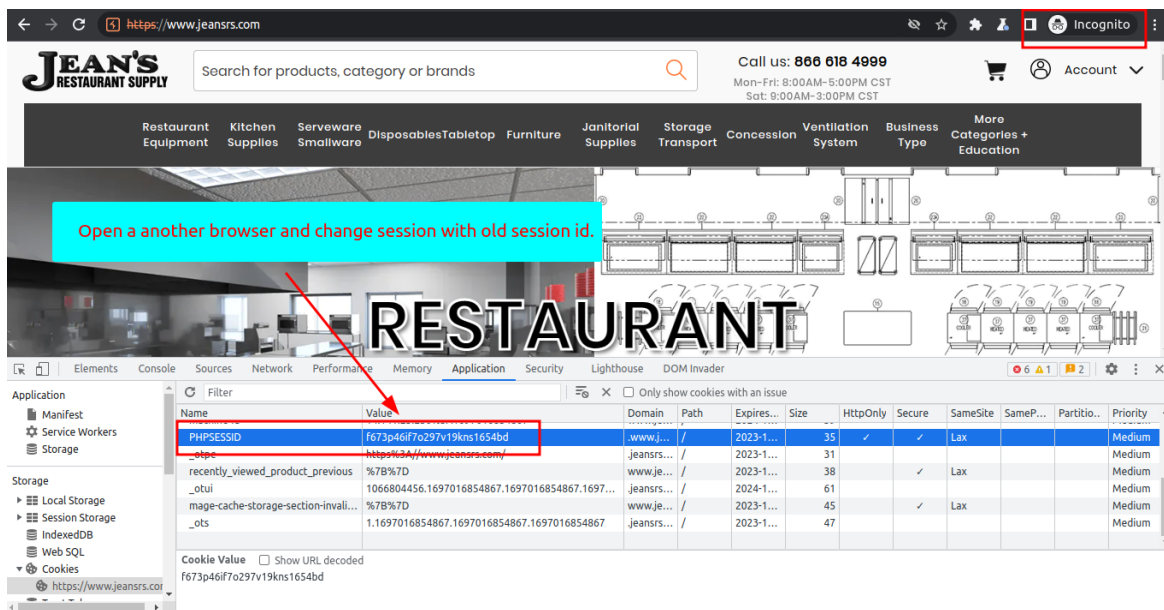
- Transmit session IDs securely over HTTPS to protect them from interception.
- Encrypt or hash session IDs when storing them in databases or other persistent storage to prevent unauthorized access.

Screenshot 1:-

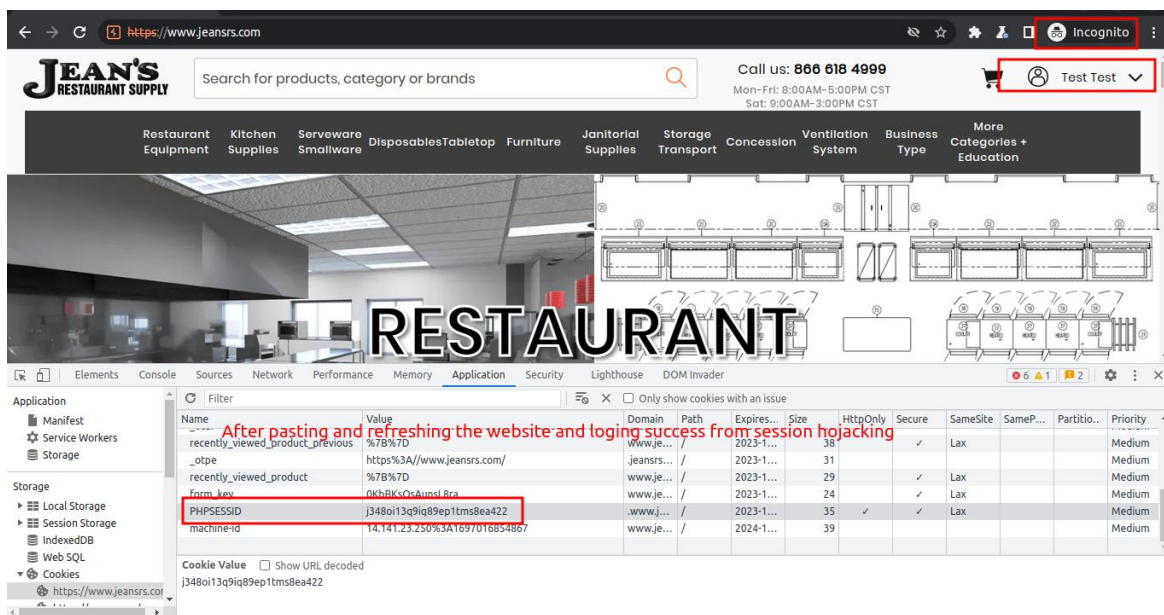
The screenshot shows the Jean's Restaurant Supply website. The Chrome DevTools Application tab is open, displaying the 'Cookies' section. A table of cookies is shown, with the 'PHPSESSID' cookie highlighted. A red box surrounds the 'PHPSESSID' cookie, and a red arrow points to its value, 'j348oi13q9iq89ep1tms8ea422', with the text 'Copy the session id' written above it.

Name	Value	Domain	Path	Expires	Size	HttpOnly	Secure	SameSite	SameP...	Partitio..	Priority
mage-cache-storage-section-invali...	%7B%7D	www.je...	/	2023-1...	45			Lax			Medium
mage-cache-storage	%7B%7D	www.je...	/	2023-1...	24			Lax			Medium
mage-cache-session	true	www.je...	/	2023-1...	21			Lax			Medium
private_content_version	be2a305e2bb2346c5f...	www.je...	/	2024-1...	55			Lax			Medium
PHPSESSID	j348oi13q9iq89ep1tms8ea422	www.je...	/	2023-1...	35			Lax			Medium
machine-id	14.141.23.250%3A1697015789782	www.je...	/	2024-1...	39						Medium

Screenshot 2:



Screenshot 3:



## Finding #2: Input Validation Bypass

Risk
High
Severity
High
Occurrences

Multiple

## Details

It was Observed that Form inputs are not filtering user inputs, An input validation attack occurs when an attacker deliberately enters malicious input with the intention of confusing an application and causing it to carry out some unplanned action. Malicious input can include code, scripts and commands, which if not validated correctly can be used to exploit vulnerabilities.

## Suggested Remediation

It is recommended to implement filter special characters such as <> , ( ) , / , SCRIPT tags as these are not currently filtered so as to prevent Injection Attacs

### Screenshot: 1

The screenshot shows the contact form on the Jean's Restaurant Supply website. The form fields are filled with the following data:

- Your Name: <script>alert(1)</script>
- Company Name: <script>alert(2)</script>
- Email: wegon30401@mugadget.com
- Mobile Number: 9876543211
- Zip Code: 110094
- What's on your mind?: <script>alert(1)</script>
- Enter Captcha: 3K53

The form is submitted, and a 400 error message is displayed on the right side of the page. The error message reads: "400. That's an error. Your client has issued a malformed or illegal request. That's all we know." The "Talk to us" button is visible at the bottom of the form.

### Screenshot 2:-

The screenshot shows the thank-you page on the Jean's Restaurant Supply website. The page displays the following content:

- Header: Jean's Restaurant Supply logo and navigation menu.
- Section: Thank You from Jean's
- Text: Thank you for contacting us. Our representative will get back to you as soon as possible. We aim to respond to all queries within 24 hours. If your enquiry requires urgent attention or you would like to speak to someone immediately, please contact 1.866.618.4999.
- Footer: 400. That's an error.

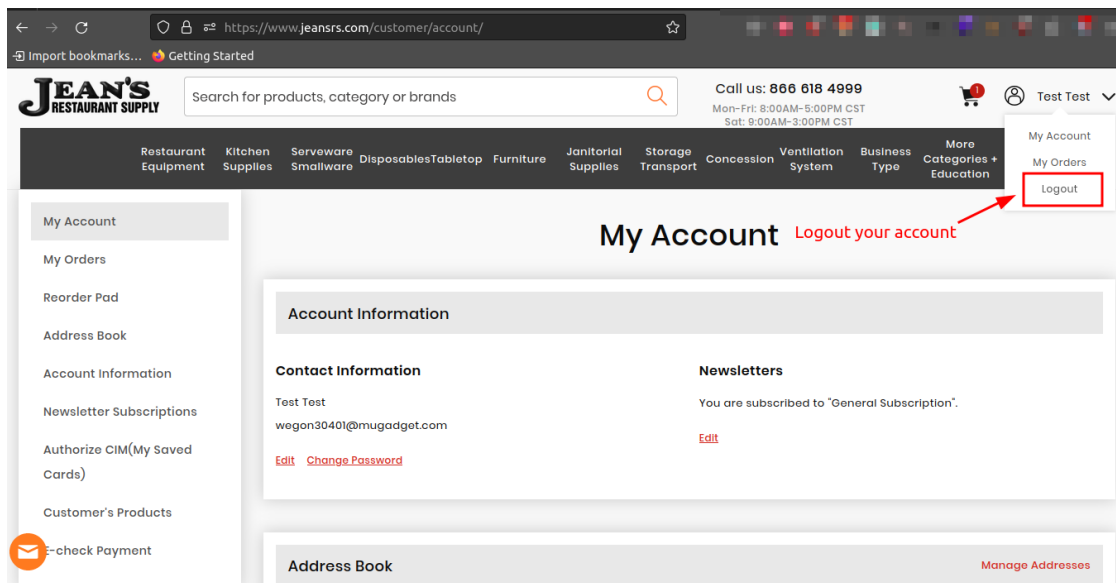
The page is titled "Thank You from Jean's" and includes a message from the company. The error message "400. That's an error." is visible in the bottom right corner.

## Finding #3: Browser Back Button Vulnerability

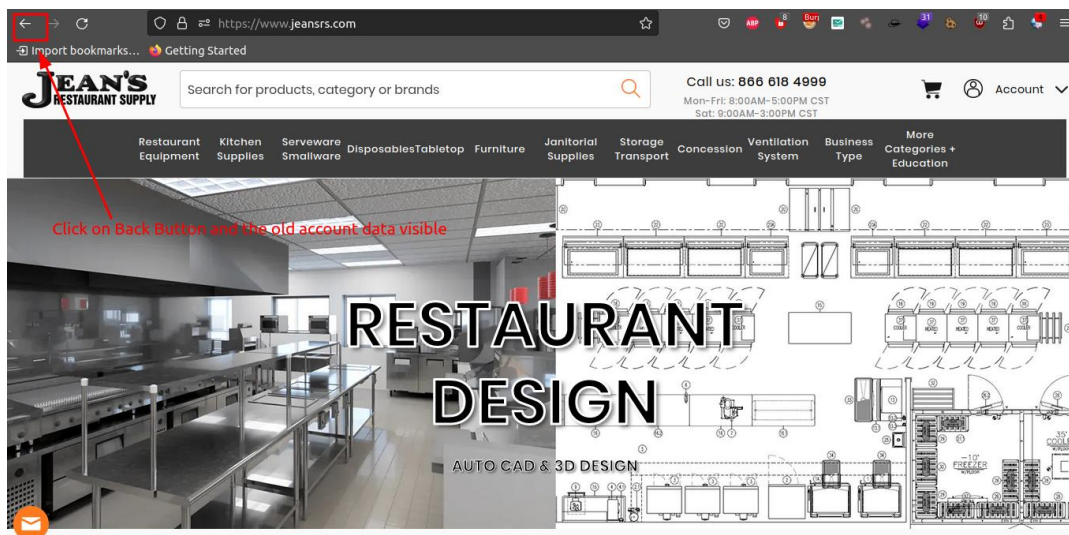
Risk
High
Severity
High
Occurrences
Multiple
Details
<p>The browser back button vulnerability, also known as "session fixation" or "session riding," is a type of web application security issue where an attacker can manipulate a user's session and gain unauthorized access to an account or perform actions on behalf of the user.</p> <p>This vulnerability occurs when a web application fails to properly manage session tokens or lacks adequate security controls. Attackers can exploit this vulnerability in the following ways:</p> <ul style="list-style-type: none"> <li>• <b>Session Fixation Attack:</b> An attacker can set a session token and trick a victim into using that token, potentially gaining access to the victim's account.</li> <li>• <b>Unwanted Actions:</b> Attackers can use a victim's session to perform unwanted actions, potentially altering data or settings.</li> </ul>
Suggested Remediation
<p>To mitigate the browser back button vulnerability, follow these remediation steps:</p> <ul style="list-style-type: none"> <li>• <b>Generate Secure Session Tokens:</b> <ul style="list-style-type: none"> <li>❖ Implement a robust session token generation mechanism that ensures unpredictability and randomness.</li> </ul> </li> <li>• <b>Session Expiration:</b> <ul style="list-style-type: none"> <li>❖ Set a session timeout and automatically invalidate sessions after a period of inactivity. This limits the window of opportunity for attackers.</li> </ul> </li> <li>• <b>Regenerate Session Tokens:</b> <ul style="list-style-type: none"> <li>❖ Generate new session tokens upon critical actions (e.g., login) or when privilege levels change. This prevents attackers from using previously captured tokens.</li> </ul> </li> <li>• <b>Session Management:</b> <ul style="list-style-type: none"> <li>❖ Implement secure session management practices and controls. Always regenerate the session ID upon login or privilege elevation.</li> </ul> </li> <li>• <b>Secure Transmission:</b> <ul style="list-style-type: none"> <li>❖ Ensure session tokens are transmitted over secure, encrypted channels (HTTPS) to prevent eavesdropping.</li> </ul> </li> <li>• <b>Logout Functionality:</b></li> </ul>

- ❖ Implement a proper logout functionality that invalidates the session token and prevents its reuse.
- **Session Fixation Protection:**
  - ❖ Protect against session fixation attacks by regenerating the session token upon login, changing privilege levels, or any other critical actions.
- **Client-Side Security Headers:**
  - ❖ Implement security headers like the **SameSite** attribute in cookies to restrict cross-origin access.

Screenshot: 1

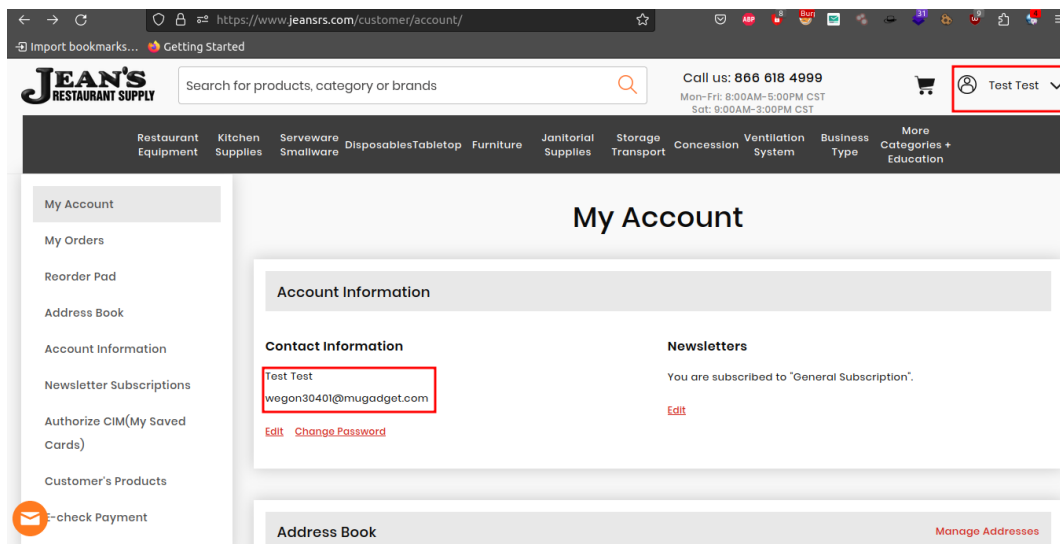


Screenshot 2:



Screenshot 3:





## Finding #4: Directory Indexing

Risk
High
Severity
High
Occurrences
Multiple
Details
<p>Web servers can be configured to automatically list the contents of directories that do not have an index page present. This can aid an attacker by enabling them to quickly identify the resources at a given path and proceed directly to analyzing and attacking those resources. It particularly increases the exposure of sensitive files within the directory that are not intended to be accessible to users, such as temporary files and crash dumps. Directory listings themselves do not necessarily constitute a security vulnerability. Any sensitive resources within the web root should in any case be properly access-controlled and should not be accessible by an unauthorized party who happens to know or guess the URL. Even when directory listings are disabled, an attacker may guess the location of sensitive files using automated tools.</p>
Suggested Remediation
<p>There is not usually any good reason to provide directory listings and disabling them may place additional hurdles in the path of an attacker.</p> <p>This can normally be achieved in two ways:</p> <ol style="list-style-type: none"> <li>1. Configure your web server to prevent directory listings for all paths beneath the web root.</li> <li>2. Place into each directory a default file (such as index.htm) that the web server will display instead of returning a directory listing.</li> </ol>

## Screenshot: 1

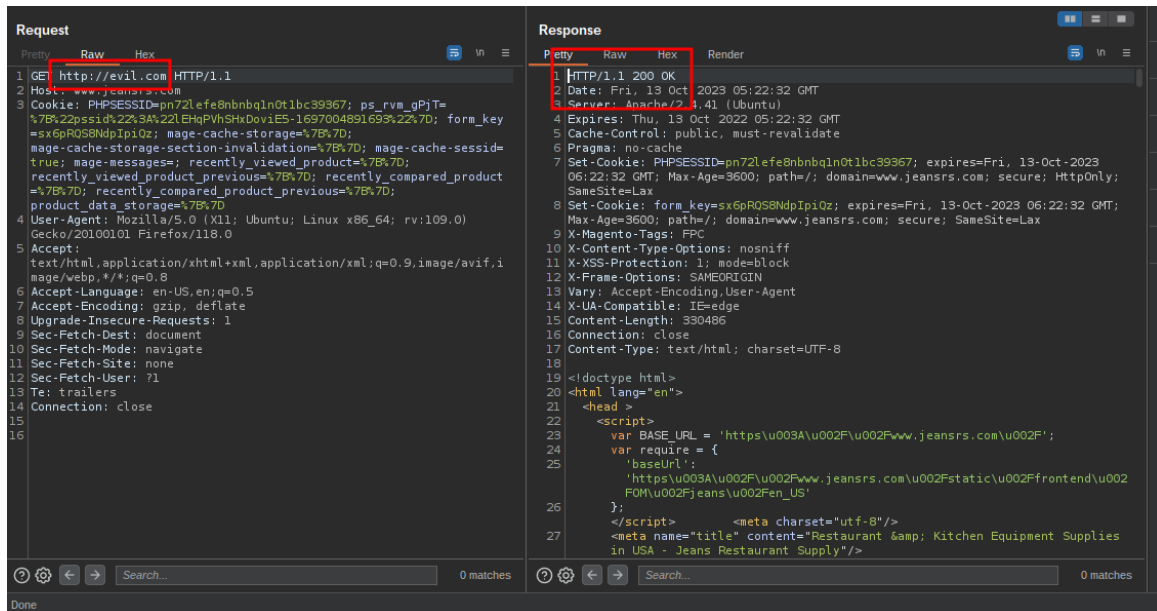


## Finding #5: Open Redirection

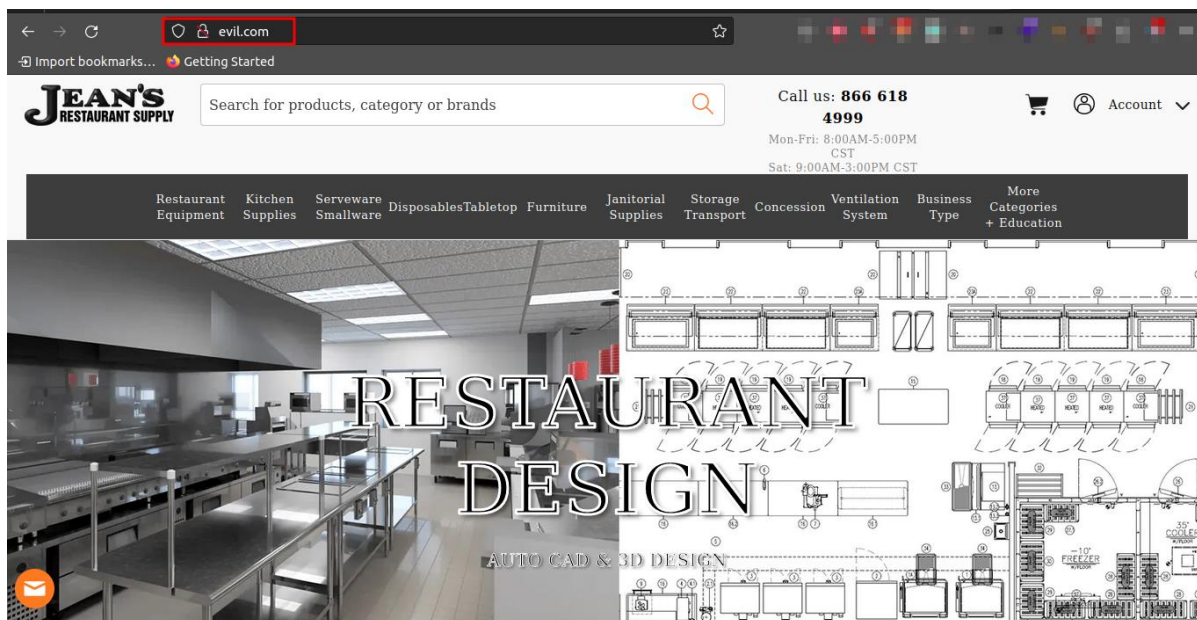
Risk
High
Severity
High
Occurrences
Multiple
Details
<p>An open redirection vulnerability can be exploited by attackers to craft phishing attacks, deceive users into visiting malicious websites, and potentially compromise their security. Although the impact is limited in this scenario, it is crucial to fix the vulnerability to prevent potential risks.</p>
<p><b>Steps to Reproduce:</b></p> <ul style="list-style-type: none"><li>• Access the website and initiate a legitimate request.</li><li>• Observe the request, particularly the URL, and locate the parameter that can be modified.</li><li>• Modify the parameter's value from <b>GET / HTTP/1.1</b> to <b>GET <a href="http://evil.com">http://evil.com</a> HTTP/1.1</b>.</li><li>• Submit the modified request.</li></ul>
Suggested Remediation

I recommend addressing this vulnerability by implementing proper input validation and security controls on the affected parameter. Ensure that any redirects are restricted to internal or trusted domains and do not allow redirection to arbitrary external sites. Furthermore, consider implementing output encoding to prevent any injected malicious code from executing.

Screenshot: 1



Screenshot 2:

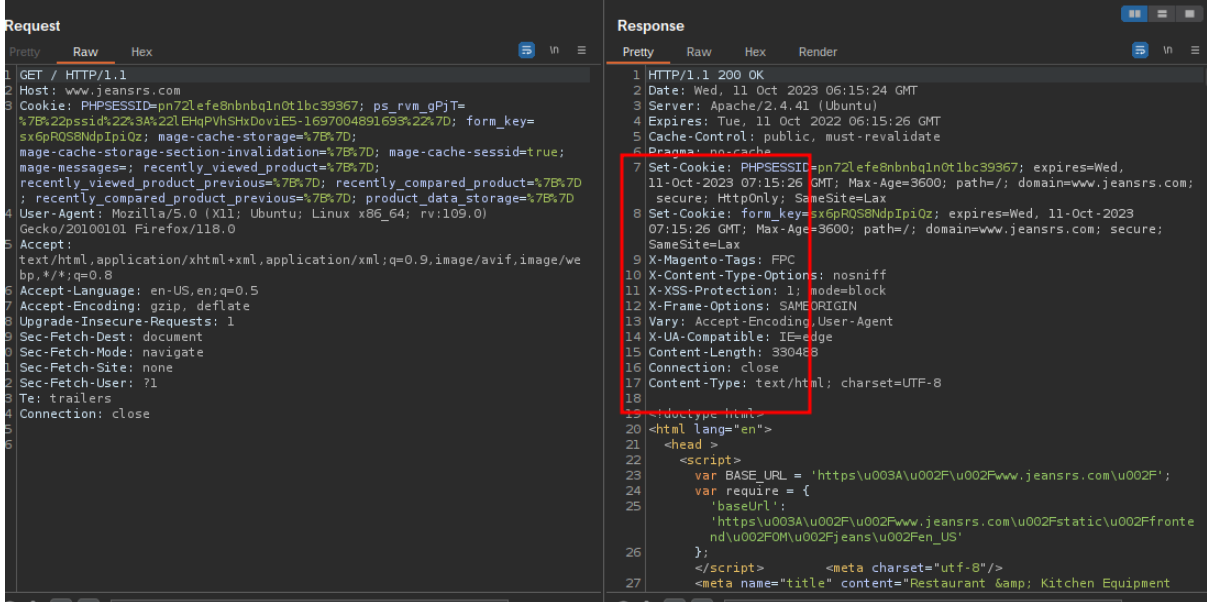


## Finding #6: Missing Security Headers

Risk

Medium
<b>Severity</b>
Medium
<b>Details</b>
<p>There are some headers that protect the application against attacks. These headers are not implemented in the application.</p> <ol style="list-style-type: none"><li>1. HTTP Strict Transport Security</li><li>2. Content Security Policy</li><li>3. Access-Control-Allow-Origin</li><li>4. X-XSS-Protection</li><li>5. Referrer policy</li></ol>
<b>Suggested Remediation</b>
<p>There are some headers that protect the application against attacks. These headers are not implemented in the application.</p> <ol style="list-style-type: none"><li>1. HTTP Strict Transport Security</li><li>2. Content Security Policy</li><li>3. Access-Control-Allow-Origin</li><li>4. X-XSS-Protection</li><li>5. Referrer policy</li></ol>

Screenshot 1: -

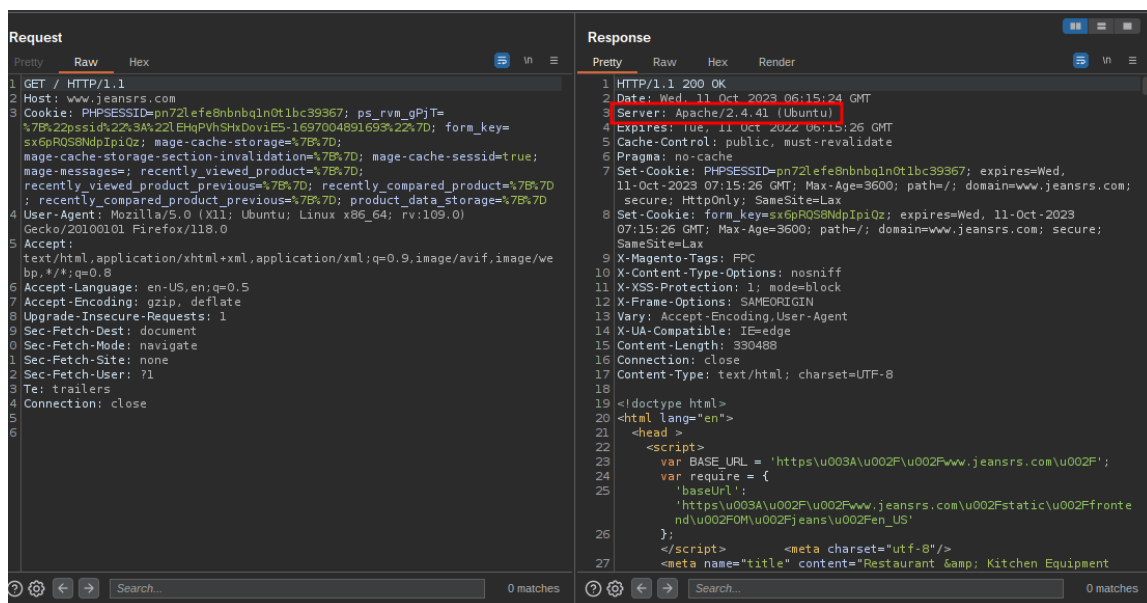


## Finding #7: Banner Grabbing

Risk
------

Medium
<b>Severity</b>
Medium
<b>Details</b>
It is observed that Apache version and Operating system has been disclosed in the response headers. This will help Attacker to Gather Information on the application server to launch Attacks from the public available exploits from Vulnerability Database.
<b>Suggested Remediation</b>
It is recommended to Hide Apache and Os version from /etc/apache2/conf-enabled/security.conf.

#### Screenshot: 1



## Finding #8: CORS (Cross-Origin Resource Sharing)

<b>Risk</b>
Medium
<b>Severity</b>
Medium

## Details

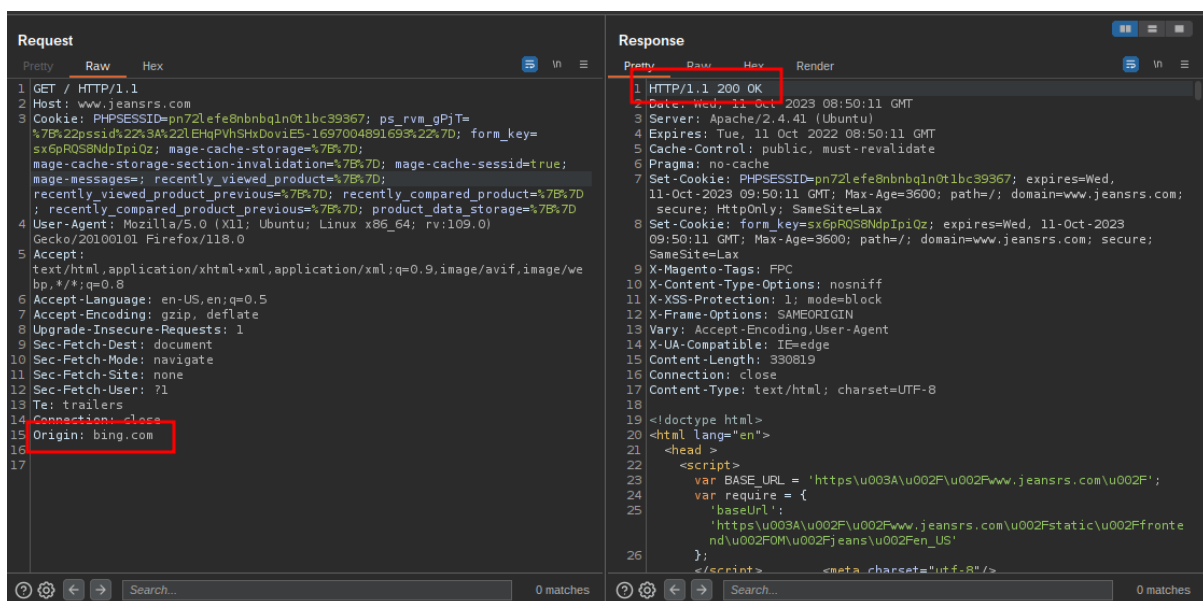
Cross-origin resource sharing (CORS) can be understood as a controlled relaxation of the same-origin policy. CORS provides a controlled way to share cross-origin resources. The CORS protocol works with specific HTTP headers that specify which web origins are trusted and their associated properties, such as whether authenticated access is permitted. These parameters are expressed in HTTP header exchanges between a browser and the cross-origin website it's attempting to access.

## Suggested Remediation

The solution is to prevent the vulnerabilities from arising in the first place by properly configuring your web server's CORS policies.

1. Specify the allowed origins
2. Only allow trusted sites
3. Don't whitelist "null"
4. Implement proper server-side security policies

### Screenshot 1: -



## Finding #9: DMARC Record Not Found

### Risk

Medium

### Severity

Medium

Affected Components
Whole Website
Details
It was observed that this domain does not email security implemented
Suggested Remediation
Please implement DMARC for mail security

#### Screenshot: 1

The screenshot shows a web interface for a DMARC lookup tool. At the top, there is a search bar containing 'jeansrs.com' and a 'DMARC Lookup' button. Below this, the tool displays the domain 'dmarc:jeansrs.com' with two buttons: 'Find Problems' and 'Solve Email Delivery Problems'. A 'dmARC' logo is also present. The main content area is a table with two columns: 'Test' and 'Result'. The table contains one row with a red 'x' icon in the 'Test' column, indicating a failure. The 'Test' column text is 'DMARC Record Published' and the 'Result' column text is 'No DMARC Record found'. A 'More Info' link is visible in the bottom right corner of the table.

Test	Result
DMARC Record Published	No DMARC Record found

## Finding #10: Additional Security Measures

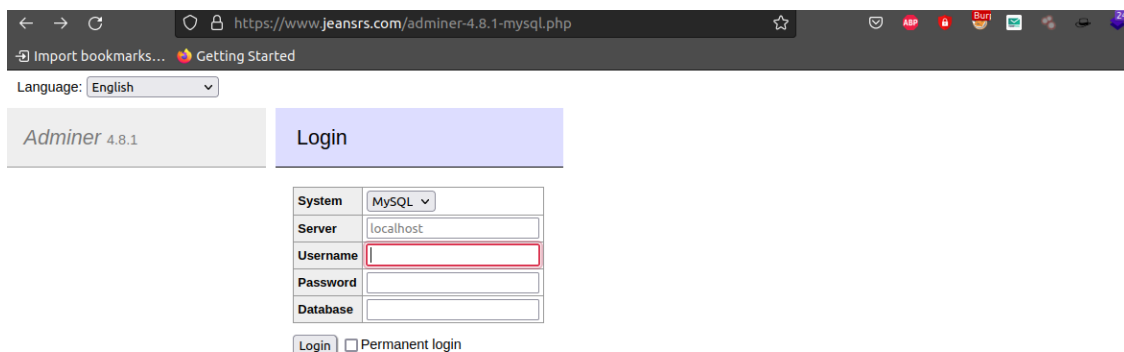
Risk
Low
Severity
Low
Details
<p>Exposing web applications like phpMyAdmin, Adminer, and Swagger UI to the public can lead to various security risks and vulnerabilities, including:</p> <ul style="list-style-type: none"> <li>• <b>Disclosure of Sensitive Information:</b> Phpinfo pages reveal details about the PHP installation, potentially aiding attackers in identifying vulnerabilities.</li> <li>• <b>Brute-Force Attacks:</b> Publicly accessible phpMyAdmin pages become targets for automated brute-force attacks to gain unauthorized access.</li> <li>• <b>Exploitation of Known Vulnerabilities:</b> Attackers may exploit known vulnerabilities in older versions of phpMyAdmin and related software.</li> <li>• <b>Unauthorized Access:</b> Successful exploitation of vulnerabilities or weak authentication mechanisms can result in unauthorized access to your database, potentially compromising sensitive information.</li> </ul>

## Suggested Remediation

To enhance the security of your web applications, consider implementing the following additional measures:

- **Restricted Access:**
  - ❖ **Place Behind Secure Login:** Limit access to phpMyAdmin, Adminer, and Swagger UI by placing them behind a secure login page.
  - ❖ **IP Restriction:** Restrict access to specific IP addresses or IP ranges to ensure only authorized users can access these applications.
- **Strong Authentication:**
  - ❖ **Complex Passwords:** Configure strong and complex passwords for login credentials of phpMyAdmin and Adminer.
  - ❖ **Two-Factor Authentication (2FA):** Enable 2FA if supported by the applications for an additional layer of security.
- **Update to Latest Versions:**
  - ❖ Regularly update phpMyAdmin, Adminer, and PHP to the latest stable versions to patch known vulnerabilities, reducing the risk of exploitation.

## Screenshot: 1



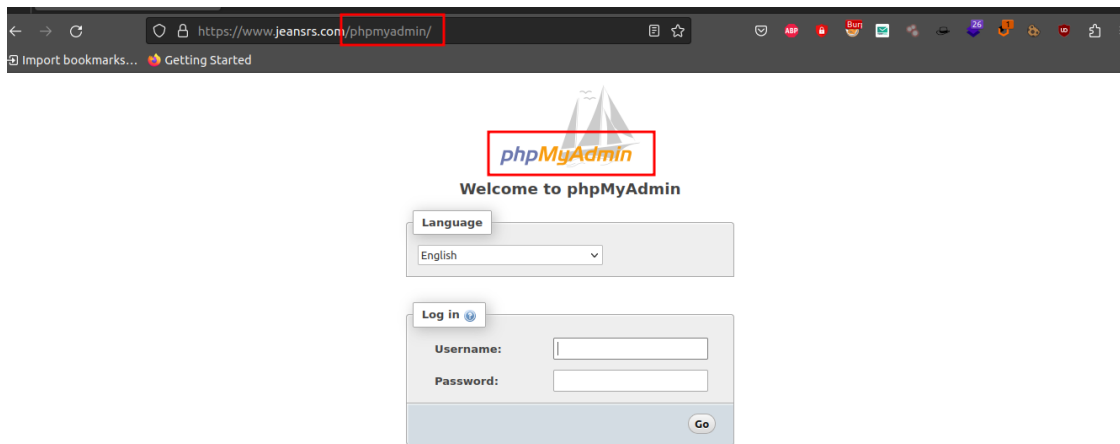
The screenshot shows the Adminer 4.8.1 login interface. The browser address bar displays <https://www.jeansrs.com/adminer-4.8.1-mysql.php>. The page has a language dropdown set to English. The main content area features a 'Login' button and a form with the following fields:

System	MySQL
Server	localhost
Username	
Password	
Database	

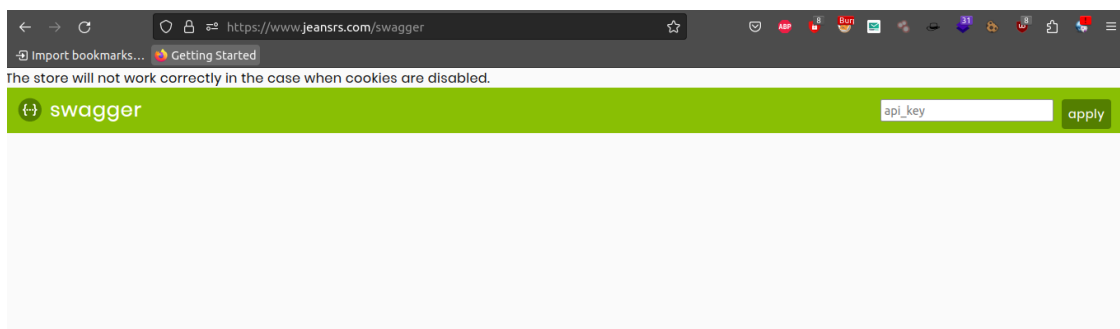
Below the form, there is a 'Login' button and a checkbox for 'Permanent login'.

## Screenshot 2:





Screenshot 3:

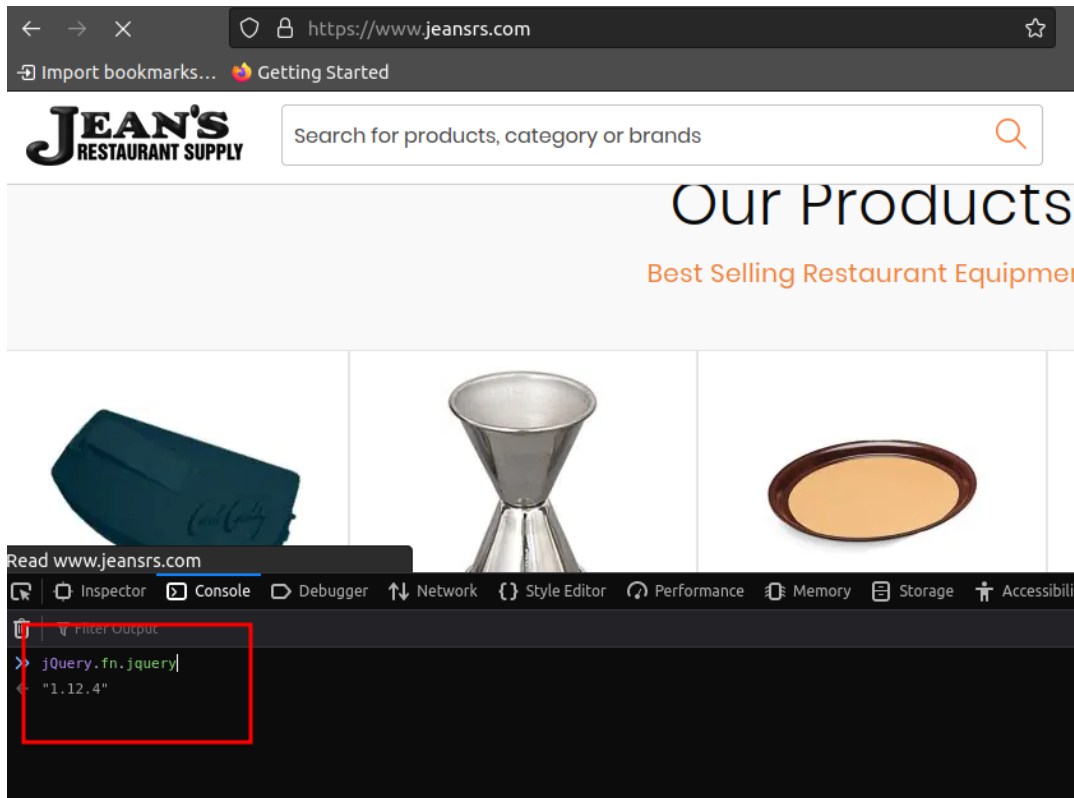


## **Finding #11: Known Vulnerable components**

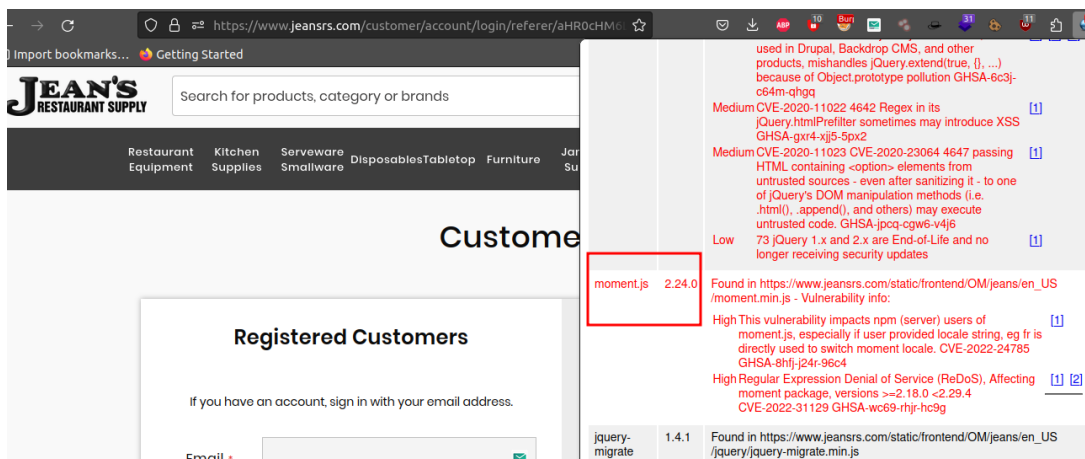
Risk
Low
Severity
Low
Affected Components
Angular js, Moment js and jQuery is affected
Details
It was observed that jQuery Ui version, angular js and moment js version is outdated
Suggested Remediation

It is recommended to update Angular js, Moment js and jQuery version to the latest to prevent future attacks on the application

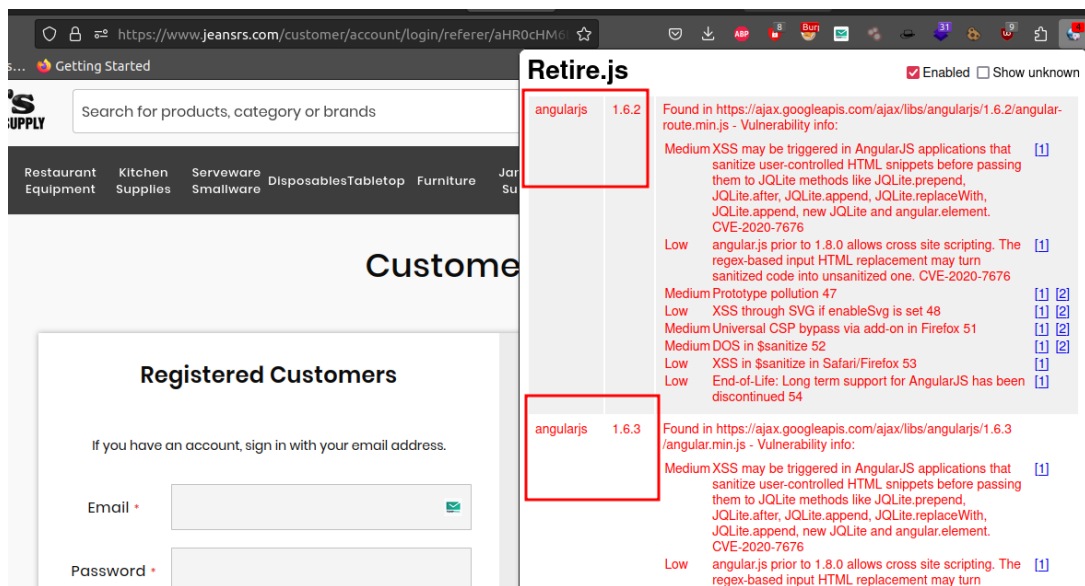
Screenshot: -



Screenshot 2 :




Screenshot 3:



## Finding #12: TLS Weak Ciphers

<b>Risk</b>
Low
<b>Severity</b>
Low
<b>Details</b>
It was observed that website has using weak ciphers
<b>Suggested Remediation</b>
It is recommended to implement the strong ciphers instead of weak ciphers

Screenshot 1: -

 <b>Cipher Suites</b>		
# TLS 1.3 (server has no preference)		
TLS_AES_128_GCM_SHA256 (0x1301)	ECDH x25519 (eq. 3072 bits RSA) FS	128
TLS_AES_256_GCM_SHA384 (0x1302)	ECDH x25519 (eq. 3072 bits RSA) FS	256
TLS_CHACHA20_POLY1305_SHA256 (0x1303)	ECDH x25519 (eq. 3072 bits RSA) FS	256
# TLS 1.2 (server has no preference)		
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (0xc027)	ECDH secp521r1 (eq. 15360 bits RSA) FS <b>WEAK</b>	128
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)	ECDH secp521r1 (eq. 15360 bits RSA) FS	128
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xc028)	ECDH secp521r1 (eq. 15360 bits RSA) FS <b>WEAK</b>	256
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)	ECDH secp521r1 (eq. 15360 bits RSA) FS	256
TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (0xcca8)	ECDH secp521r1 (eq. 15360 bits RSA) FS	256

## Finding #13: Missing HTTP only and secure flag in cookie

Risk
Low
Severity
Low
Affected Components
Session ID
Details
HTTP Only is an additional flag included in a Set-Cookie HTTP response header. Using the HttpOnly flag when generating a cookie helps mitigate the risk of client side script accessing the protected cookie
Suggested Remediation
Set-Cookie: <name>=<value>; <Max-Age>=<Age> `[]; expires=<date>[]; domain=<domain_name> []; path=<some_path>[]; secure[]; HttpOnly

Screenshot 1:

Request

PrettyRawHex

1 GET / HTTP/1.1

2 Host: www.jeansrs.com

3 Cookie: PHPSESSID=pn72lefe8nbnbqln0t1bc39367; ps\_rvm\_gPjT=%7B%22psid%22%3A%22LEHqPVhSHxDoviES-169700489169%22%7D; form\_key=sx6pRQs8NdpIpi0z; mage-cache-storage=%7B%7D; mage-cache-storage-section-invalidation=%7B%7D; mage-cache-session=true; mage-messages=; recently\_viewed\_product=%7B%7D; recently\_viewed\_product\_previous=%7B%7D; recently\_compared\_product=%7B%7D; recently\_compared\_product\_previous=%7B%7D; product\_data\_storage=%7B%7D

4 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86\_64; rv:109.0) Gecko/20100101 Firefox/118.0

5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,\*/\*;q=0.8

6 Accept-Language: en-US,en;q=0.5

7 Accept-Encoding: gzip, deflate

8 Upgrade-Insecure-Requests: 1

9 Sec-Fetch-Dest: document

0 Sec-Fetch-Mode: navigate

1 Sec-Fetch-Site: none

2 Sec-Fetch-User: ?1

3 Te: trailers

4 Connection: close

5

6

Response

PrettyRawHexRender

1 HTTP/1.1 200 OK

2 Date: Thu, 12 Oct 2023 06:10:47 GMT

3 Server: Apache/2.4.41 (Ubuntu)

4 Expires: Wed, 12 Oct 2022 06:10:47 GMT

5 Cache-Control: public, must-revalidate

6 Pragma: no-cache

7 Set-Cookie: PHPSESSID=pn72lefe8nbnbqln0t1bc39367; expires=Thu, 12-Oct-2023 07:10:47 GMT; Max-Age=3600; path=/; domain=www.jeansrs.com; secure; HttpOnly; SameSite=Lax

8 Set-Cookie: form\_key=sx6pRQs8NdpIpi0z; expires=Thu, 12-Oct-2023 07:10:47 GMT; Max-Age=3600; path=/; domain=www.jeansrs.com; secure; SameSite=Lax

9 X-Magento-Tags: TFS

10 X-Content-Type-Options: nosniff

11 X-XSS-Protection: 1; mode=block

12 X-Frame-Options: SAMEORIGIN

13 Vary: Accept-Encoding,User-Agent

14 X-UA-Compatible: IE=edge

15 Content-Length: 330743

16 Connection: close

17 Content-Type: text/html; charset=UTF-8

18

19 <!doctype html>

20 <html lang="en">

21 <head >

22 <script>

23 var BASE\_URL = 'https\u003A\u002F\u002Fwww.jeansrs.com\u002F';

24 var require = {

25 'baseUrl':

26 'https\u003A\u002F\u002Fwww.jeansrs.com\u002Fstatic\u002Ffrontend\u002FQM\u002Fjeans\u002Fen\_US'

27 };

28 </script>

29 <meta charset="utf-8"/>

30 <meta name="title" content="Restaurant & Kitchen Equipment Supplies in USA - Jeans Restaurant Supply"/>

0 matches

0 matches