PENETRATION TESTING REPORT

Scope:	http://3.127.23.232
Test Date:	05/03/2024
Assessment By:	Kancharla Pavan Kumar Reddy

Session Fixation

Severity: High

Description:

Session Fixation is a vulnerability that occurs when an attacker can manipulate session identifiers to gain unauthorized access to another user's account. In this scenario, the attacker intercepts a legitimate user's session identifier (Session ID) and then tricks the user into using it or sets it as the session identifier for the victim's session. As a result, the attacker can effectively impersonate the victim and access their account without needing to authenticate.

Impact:

The impact of Session Fixation can be severe, as it allows attackers to bypass authentication controls and gain unauthorized access to sensitive information or perform malicious actions on behalf of the victim. This can lead to privacy breaches, data loss, financial fraud, and damage to the reputation of the affected organization.

Recommendations:

To mitigate Session Fixation vulnerabilities, it is recommended to implement secure session management practices, including:

- Regenerating session identifiers upon authentication.
- Associating session identifiers with client IP addresses.
- Using secure cookies with the 'HttpOnly' and 'Secure' attributes.
- Implementing session expiration mechanisms to limit the lifespan of sessions.
- Conducting regular security assessments and penetration testing to identify and remediate vulnerabilities in session management.

Steps to Reproduce:

1.) **Log In with Two Separate Accounts:** Sign in to the application using two different user accounts, one as "User 1" and the other as "User 2".
2.) **Use Burp Suite Proxy:** Open Burp Suite, a tool used to intercept and modify web traffic.
3.) **Intercept User 1's Home Page:** While logged in as "User 1", intercept the home page request using Burp Suite and forward it to the Repeater tool for further analysis.
4.) **Copy User 1's Session ID:** Locate the Session ID parameter in the request header. Copy the Session ID associated with "User 1".
5.) **Modify User 2's Request Header:** Switch to "User 2" account. Replace the Session ID in "User 2"'s request header with the copied Session ID of "User 1" and run the modified request for "User 2".

6.) **Observe Unauthorized Access:** Notice that the response grants access to the "User 1" account instead of "User 2", indicating a security vulnerability where one user can access another user's account.

Affected URL's:

http://3.127.23.232

Evidence:

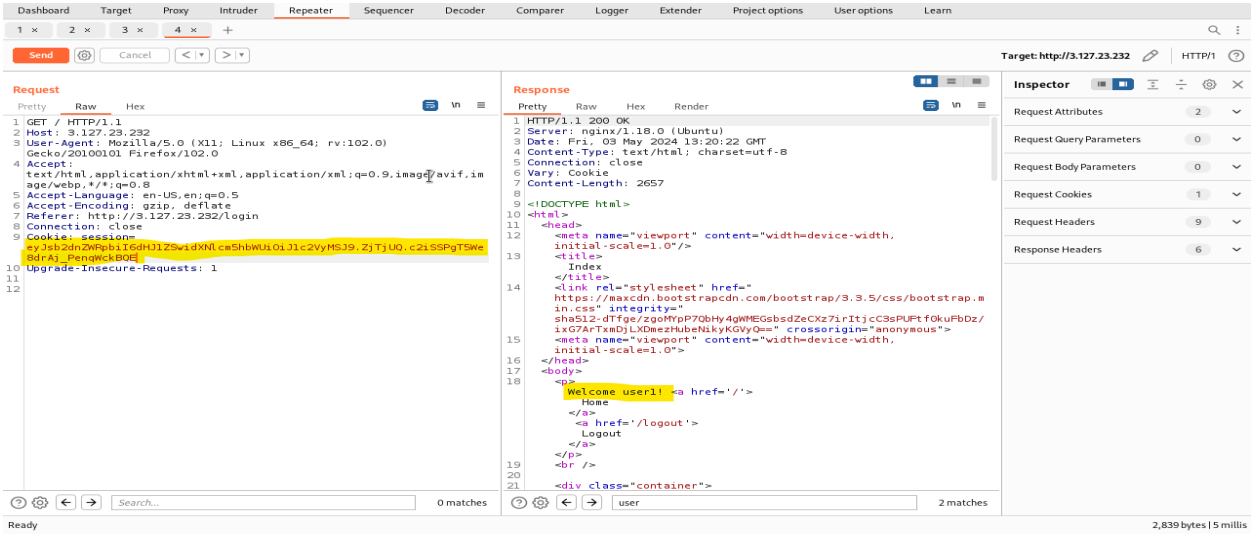

Figure 1Session ID of User 1(Initial Request and Response)

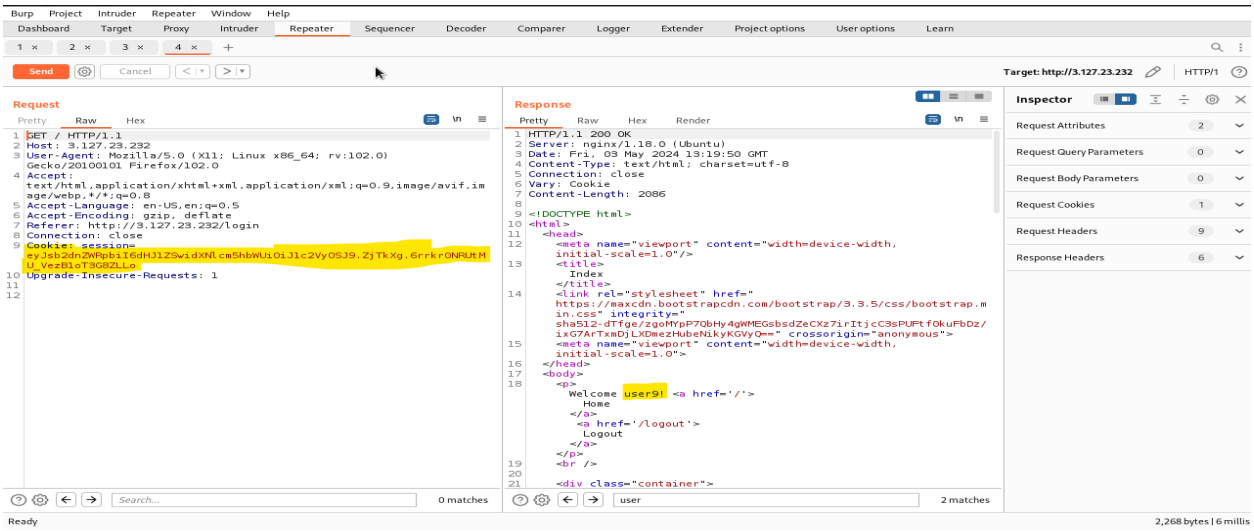

Figure 2 Session Id of User 1 after replacing with User 9

Sensitive Information in Cookie

Severity: Medium

Description:

This vulnerability involves the insecure storage of sensitive information within cookies used by the application. Sensitive data, such as user credentials (e.g., passwords) or personally identifiable information (PII), is stored in cookies without adequate encryption or protection mechanisms. This can lead to unauthorized access to sensitive data by attackers, posing significant risks to user privacy and security.

Recommendations:

To mitigate this vulnerability, it is recommended to review the application's cookie storage mechanisms and ensure that sensitive information is encrypted before being stored in cookies. Additionally, sensitive data should be handled securely, following industry best practices for data protection and compliance with relevant regulations such as GDPR or HIPAA.

Steps to Reproduce:

1.) **Log In:** Sign in to your user account on the website.
2.) **Use a Proxy Tool:** Open a proxy tool like Burp Suite, which helps analyze web traffic.
3.) **Capture the Session ID:** While logging in, capture the request made to the website using Burp Suite or similar tools. Look for the session ID value in the request header.
4.) **Extract a Segment of the Session ID:** From the captured session ID, copy a portion of it for further analysis.
5.) **Decode the Cookie:** Use a proxy tool(Burp Suite) or online decoder to decode the copied portion of the session ID using base64 decryption.
6.) **Inspect the Decoded Information:** After decryption, you may notice sensitive information such as your User ID and possibly your password contained within the decoded session ID.

Affected URL's:

http://3.127.23.232

Complete Session ID:

eyJsb2dnZWRpbiI6dHJ1ZSwidXNlcm5hbWUiOiJ1c2VyMSJ9.ZjTgNg.TnbQ5sr0s8Ayfz6BAE7kSC4ui9c

Partial Session ID(Vulnerable):

eyJsb2dnZWRpbiI6dHJ1ZSwidXNlcm5hbWUiOiJ1c2VyMSJ9

Evidence:

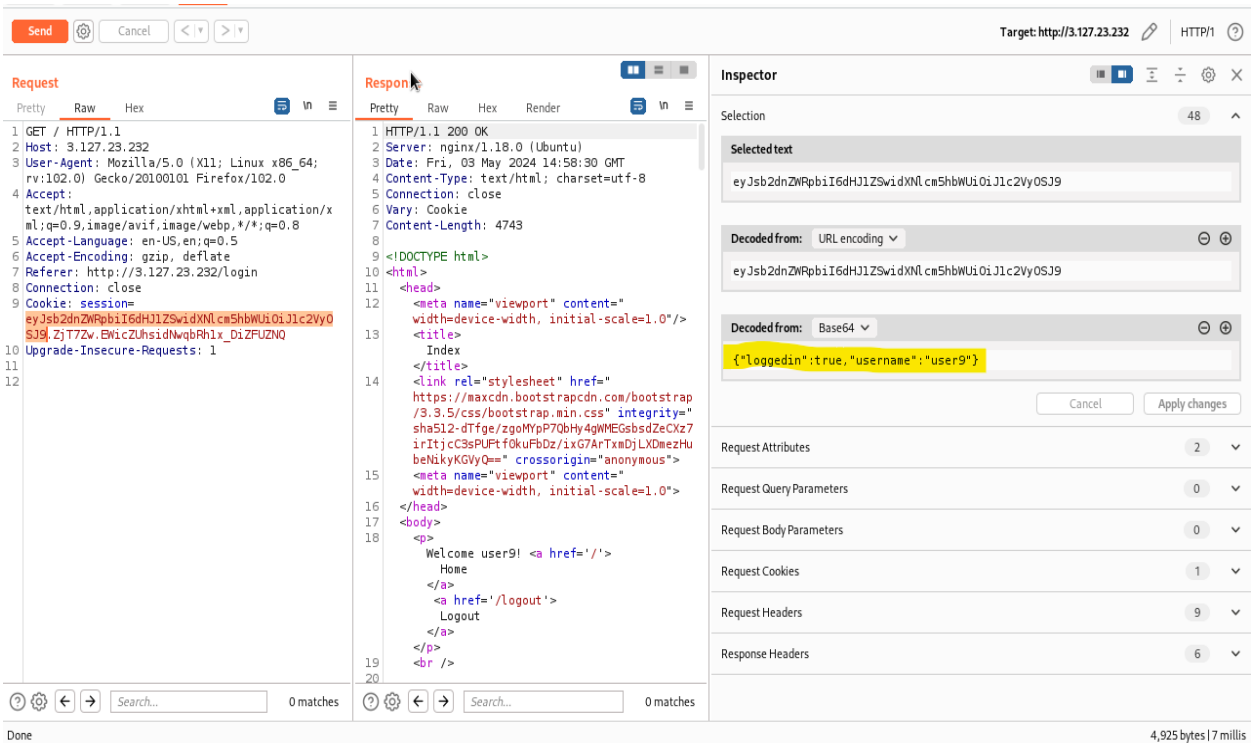

Figure 3 Sensitive Information in Cookie

Clickjacking

Severity: Medium

Description:

Clickjacking is a vulnerability that allows an attacker to trick users into clicking on unintended elements by disguising them as legitimate content. In this scenario, the attacker creates a webpage containing an iframe that overlays the target website. By embedding the target website within the iframe, the attacker can manipulate the user's interactions with the website, potentially leading to unauthorized actions or data theft.

Impact:

Clickjacking can have serious consequences, including:

- **Unauthorized actions:** Users may unknowingly perform actions on the target website, such as making purchases or changing account settings.
- **Data theft:** Attackers can extract sensitive information displayed on the target website, such as login credentials or personal data.
- **Phishing attacks:** Clickjacking can be used to trick users into clicking on malicious links or buttons, leading to further exploitation.

Recommendation:

To mitigate Clickjacking vulnerabilities, it is recommended to implement the following measures:

- Implement X-Frame-Options or Content Security Policy (CSP) headers to prevent the website from being embedded in iframes.
- Use frame-busting scripts to prevent the website from being framed by other pages.
- Educate users about the risks of interacting with embedded content and encourage them to verify the authenticity of websites before performing actions.

Steps to Reproduce:

1. **Log In to the Website:** Access the website using your user credentials.
2. **Create an iframe Script:** Prepare a simple script (Ref: Sample Script) that includes an iframe element.
3. **Paste the Website URL into the Script**: Insert the website's URL into the iframe script.
4. **Open the HTML File in a Web Browser**: Open the HTML file containing the iframe script in any web browser.

5. **Observe the Website in the iframe:** Notice that the website is displayed within the iframe, indicating that it can be embedded into other web pages without restriction.

Sample Script:

```
<html>
  <head>
    <title>Clickjack test page</title>
  </head>
  <body>
    <iframe src=" http://3.127.23.232" width="500" height="500"></iframe>
  </body>
</html>
```

Affected URL's:

http://3.127.23.232

Clear Text Transmission of Credentials over HTTP (Unencrypted Transmission)

Severity: Medium

Description:

The vulnerability involves the transmission of sensitive information, such as usernames and passwords, in clear text over HTTP connections. During the login process on the target website, user credentials are sent as plain text within the HTTP request headers. This exposes the credentials to interception by attackers who may be monitoring network traffic, leading to unauthorized access to user accounts.

Impact:

The impact of transmitting credentials over HTTP can be severe:

- **Credentials Compromise:** Attackers can intercept and capture user credentials, allowing them to impersonate legitimate users and gain unauthorized access to accounts.
- **Privacy Violation:** Sensitive user information, including passwords, is exposed to unauthorized parties, violating user privacy and confidentiality.
- **Credential Reuse:** Compromised credentials obtained through interception can be used by attackers to access other online accounts belonging to the same user, exacerbating the impact of the vulnerability.

Recommendation:

To mitigate the vulnerability of clear text transmission of credentials over HTTP, the following measures are recommended:

- **Implement HTTPS:** Encrypt communication between clients and servers by using HTTPS protocol, which ensures secure transmission of sensitive data, including login credentials.
- **HSTS Implementation:** Enable HTTP Strict Transport Security (HSTS) to enforce the use of HTTPS and prevent downgrade attacks that may attempt to force communication over insecure HTTP.
- **Secure Authentication Mechanisms:** Implement secure authentication mechanisms such as multi-factor authentication (MFA) to add an extra layer of security and mitigate the risk of credential compromise.

Steps to Reproduce:

1. **Log In to the Website:** Enter your username and password on the login page of the target website.
2. **Use a Proxy Tool:** Utilize a tool such as Burp Suite to intercept web traffic.
3. **Capture the Login Request:** Intercept the request made to the login page using Burp Suite or a similar tool.
4. **Inspect the Request Header:** Examine the request header captured in Burp Suite.
5. **Notice Sensitive Information:** Observe that sensitive information such as your username and password is transmitted in clear text within the request header.

Affected URL's:

http://3.127.23.232/auth

Evidence:

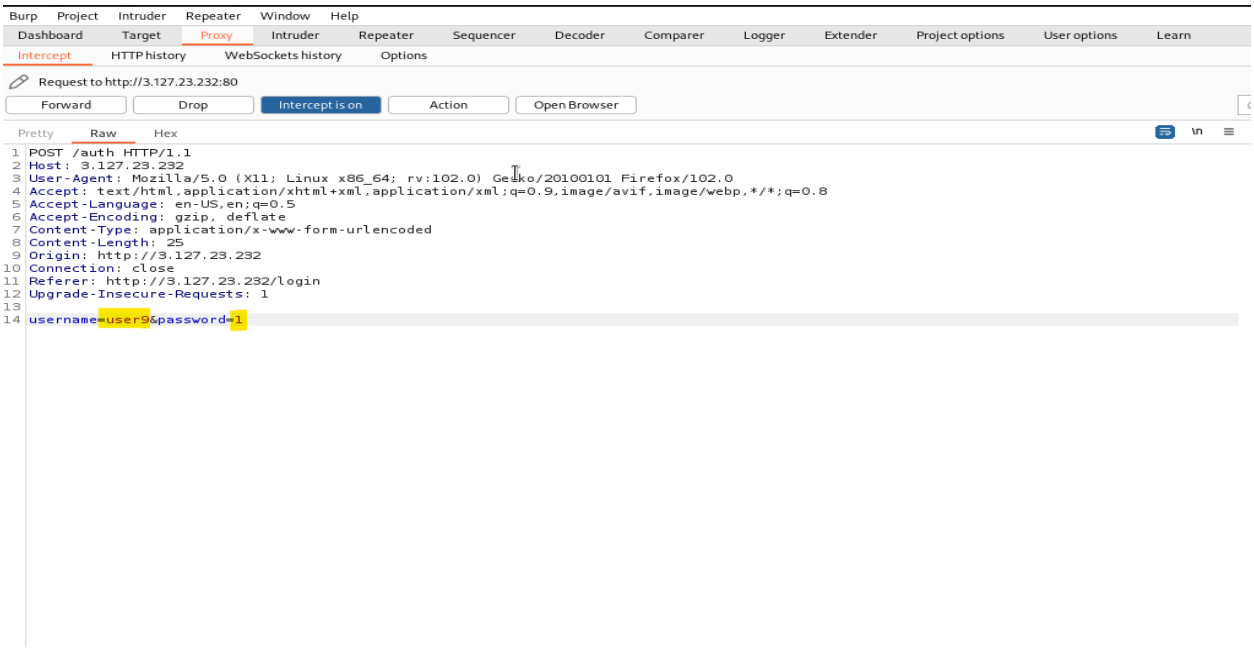

Figure 4 Unencrypted Data Traversal (HTTP)

Sensitive Information Disclosure

Severity: Low

Description:

Server Information Disclosure is a vulnerability that occurs when a web server reveals sensitive details about its software configuration in the response headers. In this scenario, when a client sends a request to the server, the response headers may include information such as the server's name and version (e.g., "nginx/1.18.0 (Ubuntu)"). This information can provide valuable insights to potential attackers, allowing them to exploit known vulnerabilities associated with specific server versions and configurations.

Impact:

The impact of Server Information Disclosure can be significant:

- **Security Risks:** Exposing server details increases the risk of targeted attacks, including exploitation of known vulnerabilities and targeted attacks against specific server versions.
- **Information Leakage:** Server information disclosure can also reveal internal infrastructure details, aiding attackers in reconnaissance activities and facilitating further exploitation.
- **Compliance Concerns**: Server information disclosure may violate security best practices and regulatory requirements, leading to compliance issues and potential legal ramifications.

Recommendation:

To mitigate Server Information Disclosure vulnerabilities, it is recommended to implement the following measures:

- **Server Hardening:** Configure the web server to suppress or obfuscate server information in response headers, preventing unauthorized disclosure of sensitive details.
- **Regular Updates:** Keep server software up-to-date with the latest security patches and updates to mitigate known vulnerabilities associated with specific versions.
- **Security Headers:** Implement security headers such as "ServerTokens" and "ServerSignature" directives to control the amount of server information disclosed in response headers.
- **Security Awareness:** Educate development and operations teams about the risks of server information disclosure and the importance of securely configuring web servers to protect against potential exploitation.

Steps to Reproduce:

1. **Log In to Your User Account:** Access your user account on the website.
2. **Use a Proxy Tool:** Utilize a tool like Burp Suite to intercept web traffic.
3. **Capture the Home Page Request:** Intercept the request made to the home page of the website using Burp Suite.
4. **Inspect the Request in Repeater:** Forward the intercepted request to the Repeater tool for further analysis.
5. **Modify the Request Method:** Change the HTTP method from "GET" to "TRACE" in the Repeater tool and send the modified request to the server.
6. **Observe the Response:** Notice the response from the server, which may include an error message revealing sensitive information such as the server's name and version (e.g., "nginx/1.18.0 (Ubuntu)") in the response body.

Affected URL's:

http://3.127.23.232/login?error=Invalid+char+in+username

Evidence:

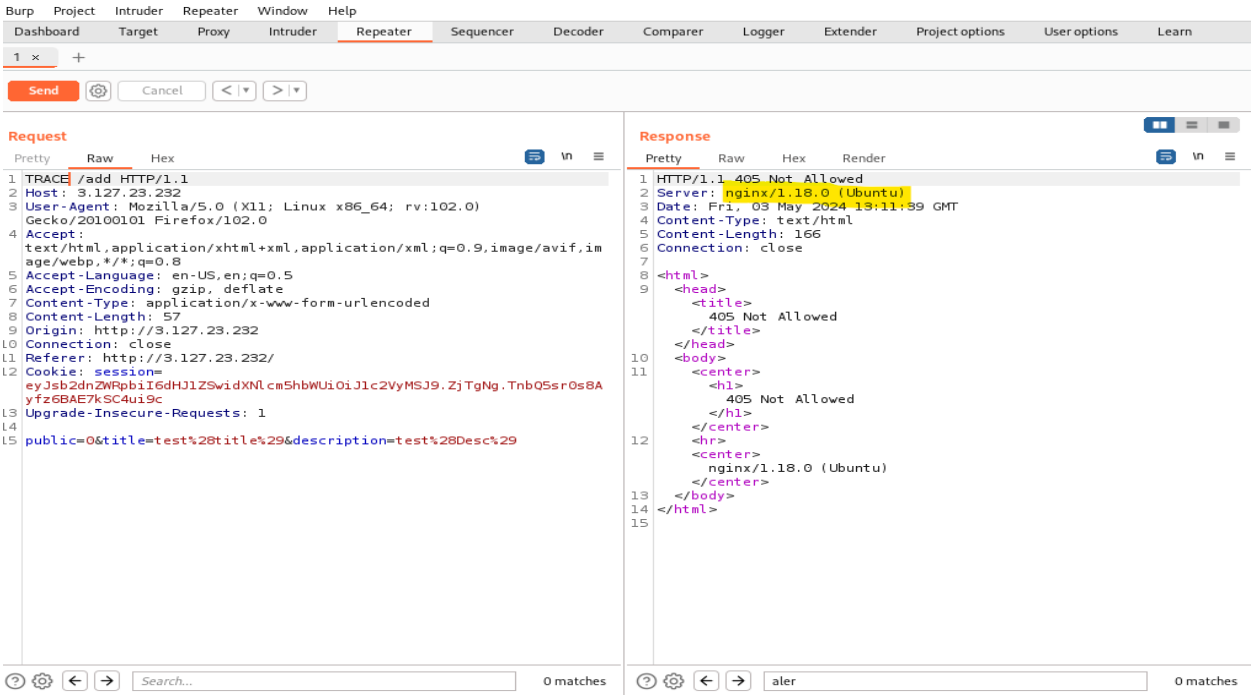

Figure 5 Information Disclosure(Response Header)

Secure and HTTP Only Flag Missing

Severity: Low

Description:

The vulnerability involves the absence of secure and HTTP Only flags on cookies used by the website. When a user authenticates on the website, session cookies are created without these flags, leaving them susceptible to various attacks. The missing "Secure" flag means that cookies can be transmitted over unencrypted HTTP connections, potentially exposing sensitive information to interception by attackers. Additionally, the absence of the "HTTP Only" flag allows client-side scripts to access the cookies, increasing the risk of cross-site scripting (XSS) attacks.

Impact:

The impact of the vulnerability can be significant:

- **Information Disclosure:** Attackers can intercept session cookies transmitted over insecure HTTP connections, leading to the disclosure of sensitive user information, such as session tokens or authentication credentials.
- **Session Hijacking:** With access to session cookies, attackers can hijack user sessions and impersonate legitimate users, gaining unauthorized access to their accounts and performing malicious actions on their behalf.
- **Cross-Site Scripting (XSS):** The absence of the HTTP Only flag increases the risk of XSS attacks, where malicious scripts can steal cookies containing session information and compromise user accounts.

Recommendation:

To mitigate the vulnerability of insecure cookie flags, it is recommended to implement the following measures:

- Set the "Secure" flag on cookies to ensure they are only transmitted over secure HTTPS connections, protecting them from interception by attackers.
- Enable the "HTTP Only" flag on cookies to prevent client-side scripts from accessing them, reducing the risk of XSS attacks and unauthorized cookie access.
- Implement strict cookie security policies and conduct regular security assessments to identify and remediate vulnerabilities in cookie management.

Steps to Reproduce:

1. **Log In to Your User Account:** Access your user account on the website using your credentials.
2. **Access Browser Developer Tools:** While on the homepage of the website, right-click anywhere on the screen and select "Inspect" or "Inspect Element" to open the browser's developer tools/console.
3. **Navigate to the Storage Tab:** Within the developer tools, locate and click on the "Storage" tab. This tab may vary depending on the browser being used (e.g., Application tab in Chrome, Storage tab in Firefox).
4. **Check Secure and HTTP Only Flags**: In the Storage tab, look for the cookies associated with the website. You will notice that the values for the "Secure" and "HTTP Only" flags are set to false for some cookies.

Affected URL's:

http://3.127.23.232/

Evidence:

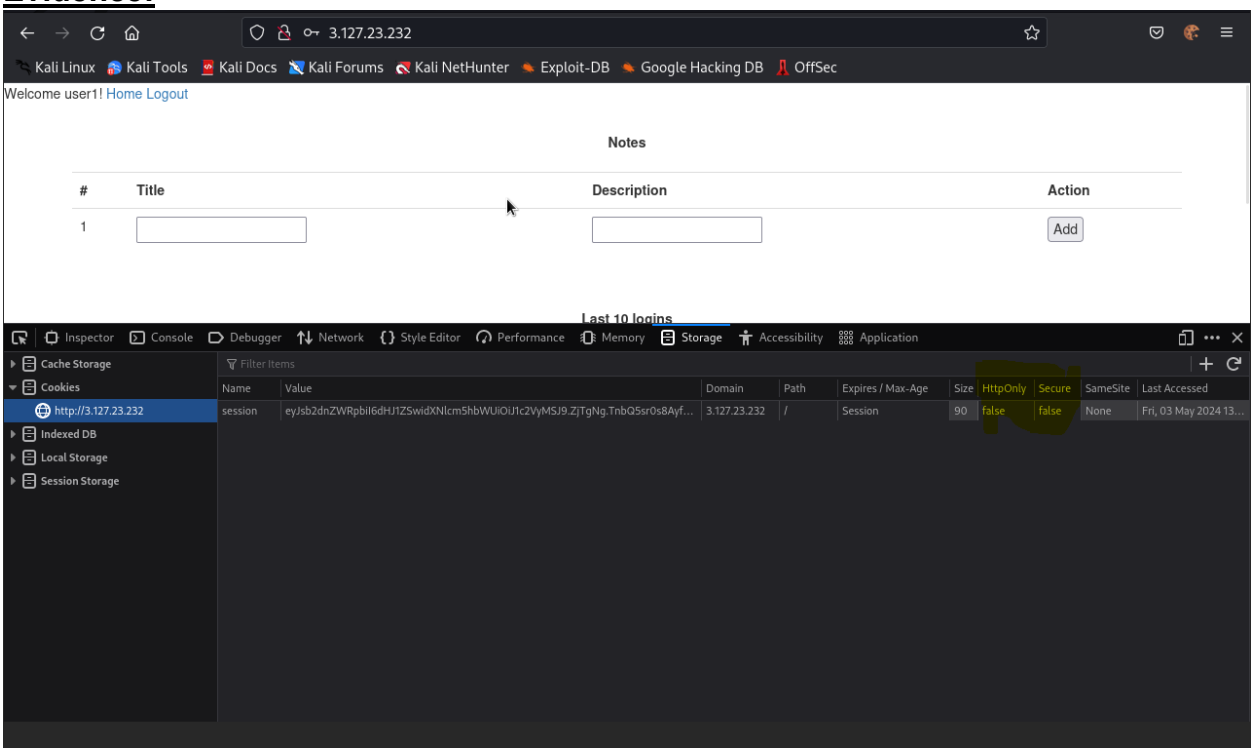

Figure 6 Missing HTTP Only and Secure Flags

User Enumeration via Login Error Messages

Severity: Low

Description:

User Enumeration via Login Error Messages is a vulnerability that occurs when the application provides different error messages for valid and invalid user credentials during the login process. In this scenario, when incorrect user credentials are entered, the application displays a distinct error message indicating that either the username or password is incorrect. This behavior allows attackers to enumerate valid user accounts by observing the response differences between valid and invalid login attempts.

Impact:

The impact of User Enumeration via Login Error Messages can be significant:

- **User Account Enumeration:** Attackers can systematically test various usernames and observe the responses to determine valid user accounts, facilitating further targeted attacks.
- **Privacy Violation:** User enumeration exposes sensitive information about the application's user base, violating user privacy and confidentiality.
- **Credential Stuffing Attacks:** With a list of valid user accounts obtained through enumeration, attackers can launch credential stuffing attacks by attempting to log in using the discovered credentials, potentially leading to unauthorized access to user accounts.

Recommendation:

To mitigate the vulnerability of user enumeration via login error messages, it is recommended to implement the following measures:

- **Provide Generic Error Messages:** Display generic error messages during the login process, such as "Invalid username or password," to prevent attackers from distinguishing between valid and invalid user accounts.
- **Implement Rate Limiting and Account Lockout:** Implement rate limiting and account lockout mechanisms to prevent attackers from performing brute-force attacks or automated user enumeration.
- **Conduct Security Awareness Training:** Educate users and developers about the risks of user enumeration and the importance of implementing secure authentication mechanisms to protect against targeted attacks.

Steps to Reproduce

1. **Enter User Credentials:** Visit the login page of the target website and enter any user credentials, deliberately using incorrect information (either username or password).
2. **Observe Error Message:** After submitting the credentials, a pop-up message will appear, clearly indicating that the provided username or password is incorrect.

Affected URL's:

http://3.127.23.232/login

Evidence:

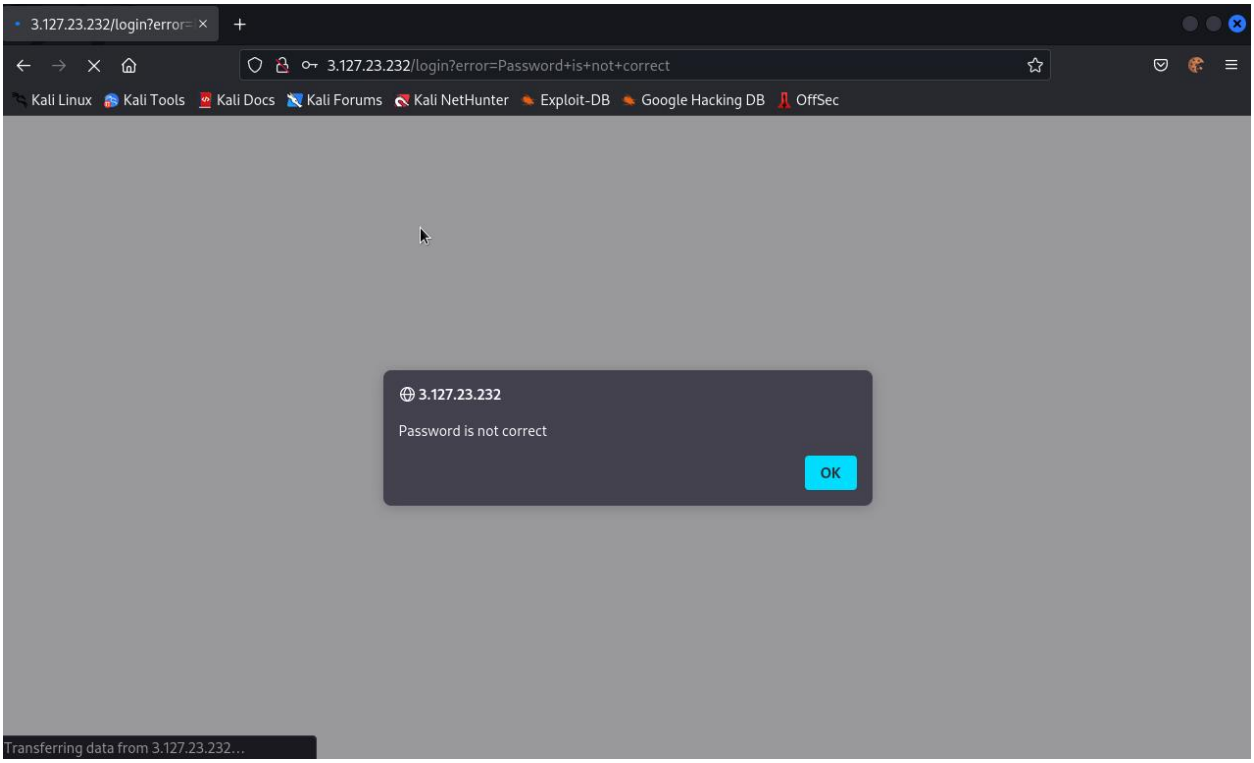

Figure 7 Improper Error Handling