testphp.vulnweb.com

SECURITY TESTING

ANIEKT KUMAR

REPORT DATED: 5th FEB 2025

Table of Contents

Confidentiality Statement

- This document is the exclusive property of Aniket Kumar. This document contains proprietary and confidential information. Duplication, redistribution, or use, in whole or in part, in any form, requires consent of both parties.
- I may share this document with auditors under non-disclosure agreements to demonstrate penetration test requirement compliance.

Disclaimer

- A penetration test is considered a snapshot in time. The findings and recommendations reflect the information gathered during the assessment and not any changes or modifications made outside of that period.
- Time-limited engagements do not allow for a full evaluation of all security controls. I prioritized the assessment to identify the weakest security controls an attacker would exploit. I recommend conducting similar assessments on an annual basis by internal or third-party assessors to ensure the continued success of the controls.

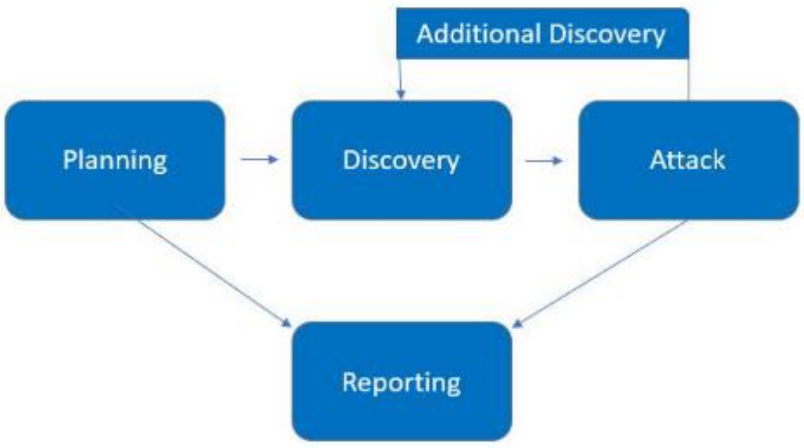
Phases of penetration testing activities include the following:

Planning - Customer goals are gathered and rules of engagement obtained.

Discovery - Perform scanning and enumeration to identify potential vulnerabilities, weak areas, and exploits.

Attack - Confirm potential vulnerabilities through exploitation and perform additional discovery upon new access.

Reporting - Document all found vulnerabilities and exploits, failed attempts, and company strengths and weaknesses.



Finding Severity Ratings

- The following table defines levels of severity and corresponding CVSS score range that are used throughout the document to assess vulnerability and risk impact.

Severity	CVSS V3 Score Range	Definition
Critical	9.0-10.0	Exploitation is straightforward and usually results in system-level compromise. It is advised to form a plan of action and patch immediately.
High	7.0-8.9	Exploitation is more difficult but could cause elevated privileges and potentially a loss of data or downtime. It is advised to form a plan of action and patch as soon as possible.
Moderate	4.0-6.9	Vulnerabilities exist but are not exploitable or require extra steps such as social engineering. It is advised to form a plan of action and patch after high-priority issues have been resolved.
Low	0.1-3.9	Vulnerabilities are non-exploitable but would reduce an organization's attack surface. It is advised to form a plan of action and patch during the next maintenance window.
Informational	N/A	No vulnerability exists. Additional information is provided regarding items noticed during testing, strong controls, and additional documentation.

Scope of Assessment for Penetration Testing on testphp.vulnweb.com

Objective:

The primary objective of this penetration test is to evaluate the security vulnerabilities in testphp.vulnweb.com, a deliberately vulnerable website designed for security training and ethical hacking practice.

In-Scope Components:

The assessment will cover the following areas:

1. Web Application Security

- Identifying injection vulnerabilities (SQLi, XSS, CSRF, LFI, RFI)
- Testing authentication mechanisms (Brute force, credential stuffing)
- Evaluating session management (Session hijacking, cookie manipulation)

2. Server and File System Exposure

- Checking for misconfigured directories and files
- Exploring access control flaws (Directory traversal, privilege escalation)
- Network-Level Vulnerabilities
- Identifying open ports and exposed services (if applicable)
- Data Exposure Risks
- Evaluating how sensitive information (e.g., credentials, database content) is handled

3. Out of Scope:

- Attacks that can cause downtime or disrupt services (e.g., DDoS, resource exhaustion)
- Real-world exploitation of vulnerabilities beyond the test environment
- Social engineering attacks or targeting external users

4. Testing Methodology:

- Manual Testing: Using web browsers, Burp Suite, and command-line tools
- Automated Scans: Limited to non-destructive vulnerability scanning
- Ethical Considerations: No unauthorized access beyond the test environment

5. Deliverables:

- Detailed Report of identified vulnerabilities, risk levels, and remediation steps
- Screenshots & Proof-of-Concepts (PoCs) for successful exploits
- Security Recommendations to mitigate identified risks

No.	Title	Risk Level
1	Man-in-the-Middle (MitM) Attack	High
2	Obtained Login credentials (High)	High
3.	SQL injection	High
4	Clickjacking	Medium
5	CSRF Attack	High
6	Server Version Leakage	Medium
7	Sensitive Information Disclosure	Critical
8	XSS Attack	Low

Man-in-the-Middle (MitM) Attack

Absence of Anti-CSRF Tokens

Description:

No Anti-CSRF tokens were found in the form. CSRF attacks exploit the trust a site has in a user, causing unintended actions without their knowledge. It differs from XSS, which exploits user trust in a site. Other names for CSRF include XSRF, one-click attack, and session riding.

Impacts:

- Unauthorized Actions
- Session Hijacking
- Loss of Trust
- Privilege Escalation

Severity: High

Steps to reproduce:

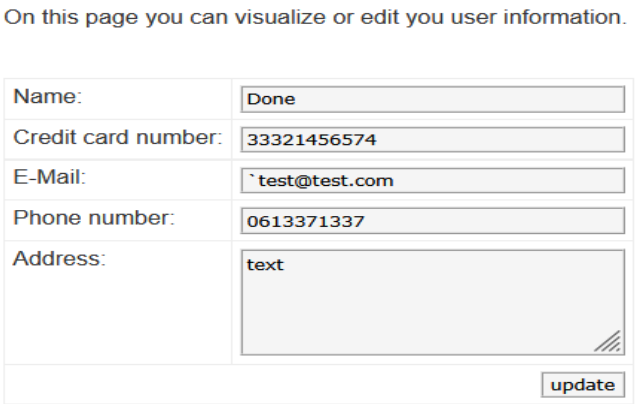
1: After Login we will be landing to userInfo page.

- POST: http://testphp.vulnweb.com/userinfo.php

2: After filling the form click to update button and data will be intercepted.

3: Intercepting the request in middle (before server) and changing the data of user info form.

4: Sending response to client with updated data.



```
POST http://testphp.vulnweb.com/userinfo.php HTTP/1.1
Host: testphp.vulnweb.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:135.0) Gecko/
20100101 Firefox/135.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Content-Type: application/x-www-form-urlencoded
Content-Length: 99
Origin: http://testphp.vulnweb.com
Connection: keep-alive
Referer: http://testphp.vulnweb.com/userinfo.php
Cookie: login=test%2Ftest
```

```
urname=Done&ucc=33321456574&uemail=%60test%40test.com&uphone=0613371337&
uaddress=text&update=update
```

```
POST http://testphp.vulnweb.com/userinfo.php HTTP/1.1
Host: testphp.vulnweb.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:135.0) Gecko/
20100101 Firefox/135.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Content-Type: application/x-www-form-urlencoded
Content-Length: 99
Origin: http://testphp.vulnweb.com
Connection: keep-alive
Referer: http://testphp.vulnweb.com/userinfo.php
Cookie: login=test%2Ftest
```

```
urname=Hello&ucc=33321456574&uemail=%60Hello%40Hello.com&uphone=0613371337
&uaddress=text&update=update
```

On this page you can visualize or edit you user information.

Name:	Hello
Credit card number:	33321456574
E-Mail:	`Hello@Hello.com
Phone number:	0613371337
Address:	text
	update

You have 5 items in your cart. You visualize you cart here.

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

Obtained Login credentials (High)

Description

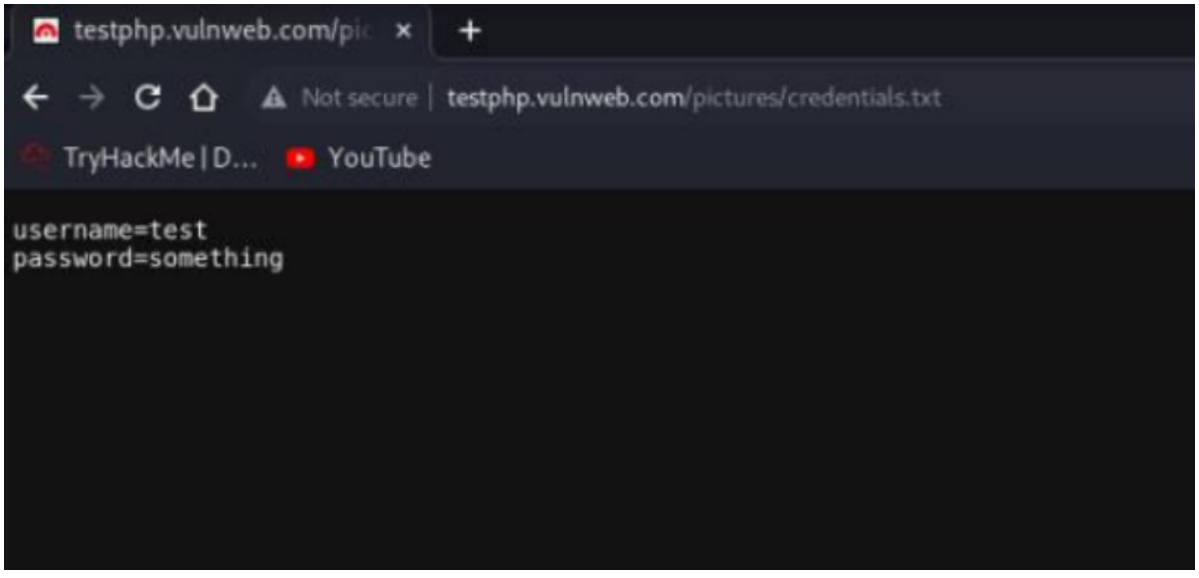
- It was possible to find the login credentials of a user from http://testphp.vulnweb.com/pictures/credentials.txt. This user credentials can be later used to login to any webpages associated with it.

Impact:

- Obtaining login credentials allows attackers to gain unauthorized access, leading to data breaches, privilege escalation, and potential system-wide compromise.

Severity: High

Metigation: Remove the webpage or the contents.



SQL injection

Description:

SQL Injection occurs when an attacker manipulates a website’s database queries by inserting malicious SQL statements in input fields. This can lead to unauthorized access to data, modification of databases, or even complete database control.

Payload:

Username: ' OR '1'='1

Password: ' OR '1'='1

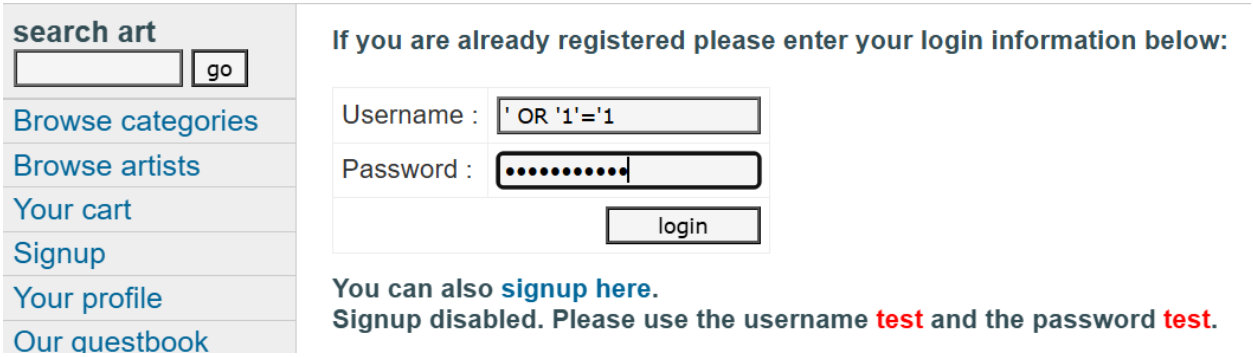
Severity: High

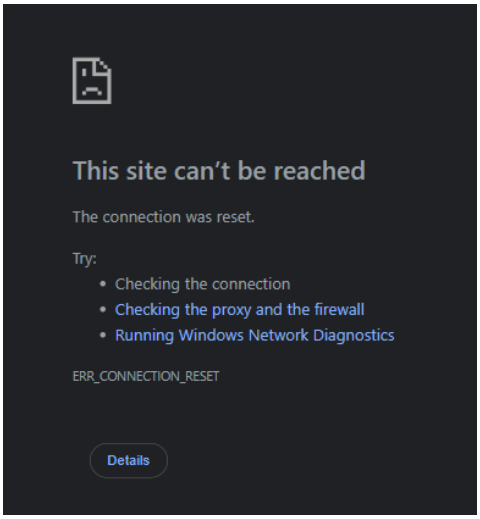
Impact of SQL Injection:

SQL injection can lead to unauthorized data access, manipulation, or deletion, causing severe privacy breaches and system compromise. It can also result in website defacement, downtime, and significant reputation damage.

STEPS

1. Open the test website's login page.
2. Enter ' OR '1'='1 in both Username and Password fields.
3. Click the Login button.
4. If login is successful, the site is vulnerable to SQL Injection.
5. If not, try admin' -- in Username and leave Password blank.
6. Click Login and observe if it bypasses authentication.
7. For blind SQL injection, enter ' AND 1=1 -- in Username and leave Password blank.
8. If successful, repeat with ' AND 1=2 --; failure indicates vulnerability.
9. Document results and responses for each attempt.
10. Use the findings to understand and mitigate SQL injection risks.





Reference:

https://cheatsheetseries.owasp.org/cheatsheets/Input_Validation_Cheat_Sheet.html

Mitigation:

Use prepared statements and input validation to prevent malicious code execution. Limit database access, deploy Web Application Firewalls (WAFs), and perform regular security audits to identify and fix vulnerabilities.

Clickjacking

Description:

- The website is Vulnerable to clickjacking(an attack that tricks a user into clicking a webpage element which is invisible or disguised as another element). This can cause users to unwittingly download malware, visit malicious web pages, provide credentials or sensitive information, transfer money or purchase products online. The webpage was added to the "<iframe>" section of a test code and it was able to show the results successfully.

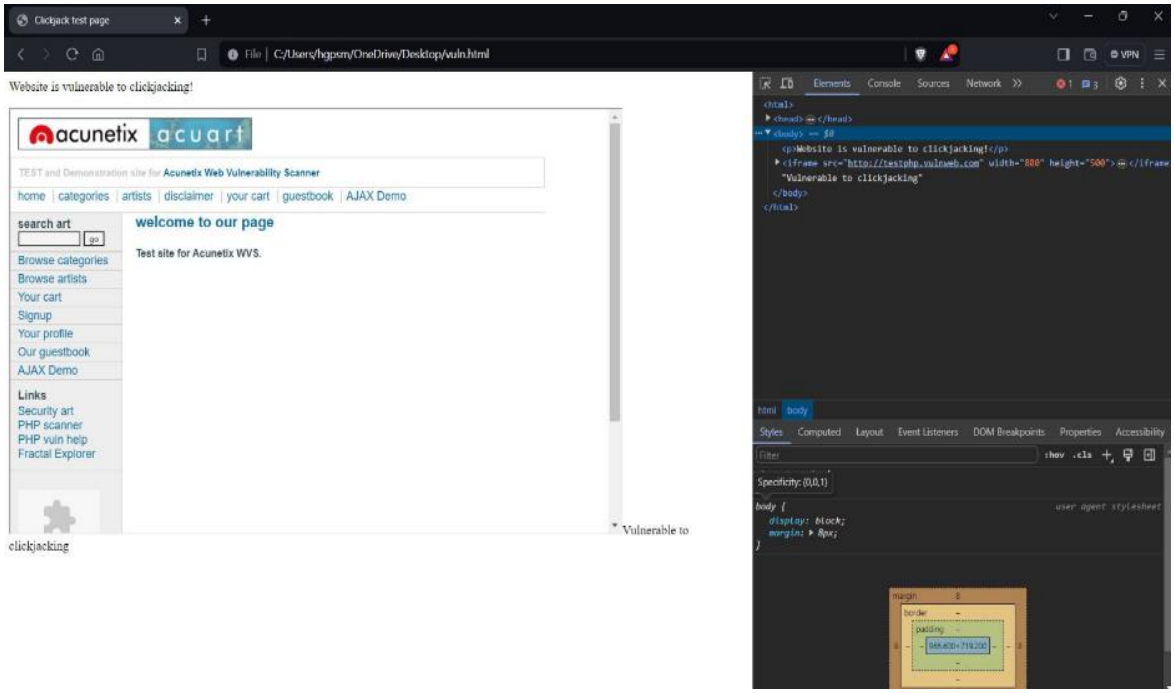
Impact:

- Clickjacking tricks users into clicking hidden elements, leading to unauthorized actions like changing settings, making purchases, or exposing sensitive information.

Severity: Medium

Mitigation:

- Implement X-frame-options header (Used to control whether a page can be placed in an <iframe>)
- The syntax for adding xframe in Apache:
- 1) Edit the httpd.conf file.
- 2) Locate the section for your website or a virtual host.
- 3) Add the following line inside the section: Header always set X-Frame-Options "DENY"



CSRF Attack

Description: CSRF tricks a user into executing unintended actions on a web application in which they are authenticated.

Impact of CSRF:CSRF can lead to unauthorized actions being performed on behalf of an authenticated user, such as changing account settings, transferring funds, or deleting data. It can compromise user accounts without their knowledge, resulting in security breaches.

Severity: High

Steps to Reproduce:

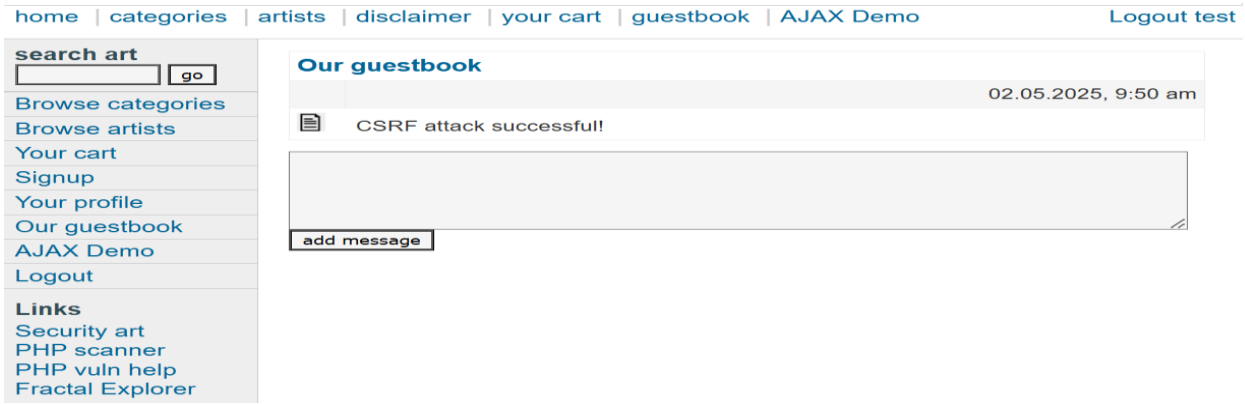
1.Open a New Tab in the Same Browser

Stay logged into the target website (your test site) in one tab.

2.Create a Malicious HTML Form Locally: Open the Developer Tools in your browser (F12 or right-click and select Inspect).

3. Go to the Console tab.: Inject the Malicious Form via JavaScript:

4. Paste this payload code in the console to simulate a CSRF attack.



Mitigation:

Use anti-CSRF tokens to validate requests and ensure they are intentional. Implement same-site cookie attributes and require re-authentication for sensitive actions to protect against CSRF attacks.

Server Version Leakage

Description:

- Server Version Leakage refers to the unintended exposure of a server's version information, which can occur through error messages, HTTP headers, or other means.

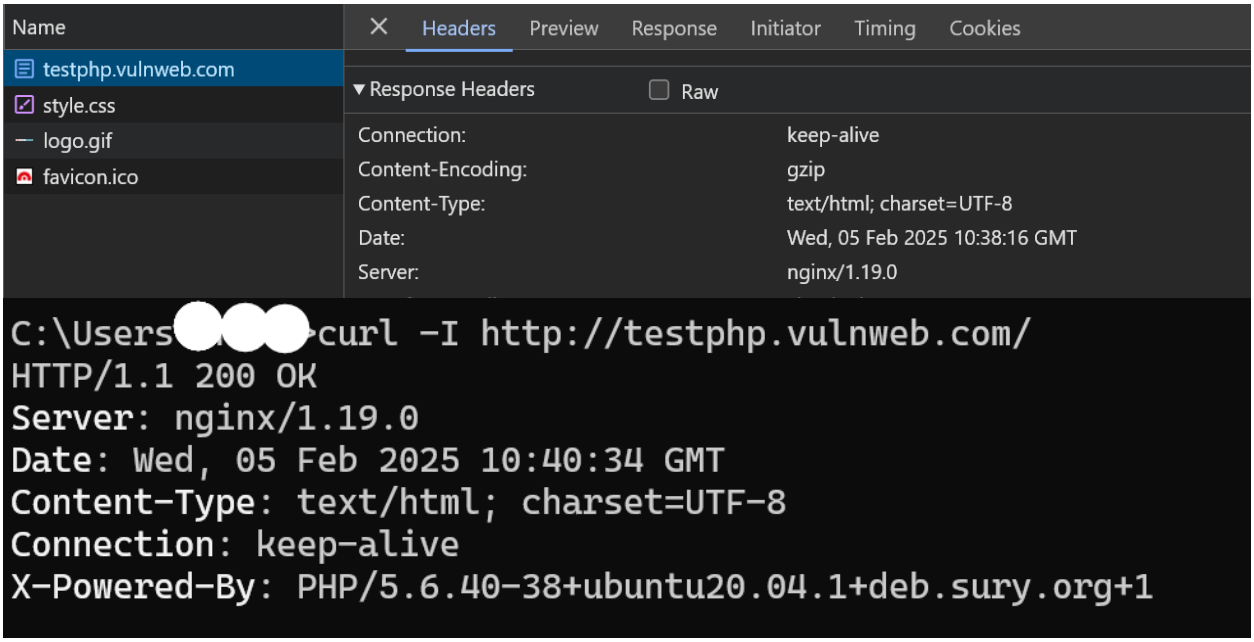
Impact:

- This leakage can allow attackers to identify vulnerabilities specific to that server version, increasing the risk of exploitation. It can lead to unauthorized access, data breaches, or server compromise, affecting the security and integrity of the system.

Severity: High

Steps to Reproduce:

1. Open http://testphp.vulnweb.com/ in your browser.
2. Right-click > Inspect > Network tab > Refresh the page.
4. Click on the first request > Scroll to Response Headers > Check for the Server header.
4. Run curl -I http://testphp.vulnweb.com/ in the terminal to view HTTP headers.
5. Look for the Server header in the curl output, e.g., Server: Apache/2.4.7 (Ubuntu).
6. Run nmap -sV testphp.vulnweb.com to detect service versions.
7. Check the Nmap output for exposed server version details.



Mitigation:
To mitigate server version leakage, disable detailed error messages, remove or obscure version information in HTTP headers, and implement security best practices such as server hardening. Regularly update server software to patch known vulnerabilities and minimize exposure.

Sensitive Information Disclosure

Description:

- The login credentials are visible to other users with username="test" password="test". It is visible to everyone who visits the signup page.

Impact:

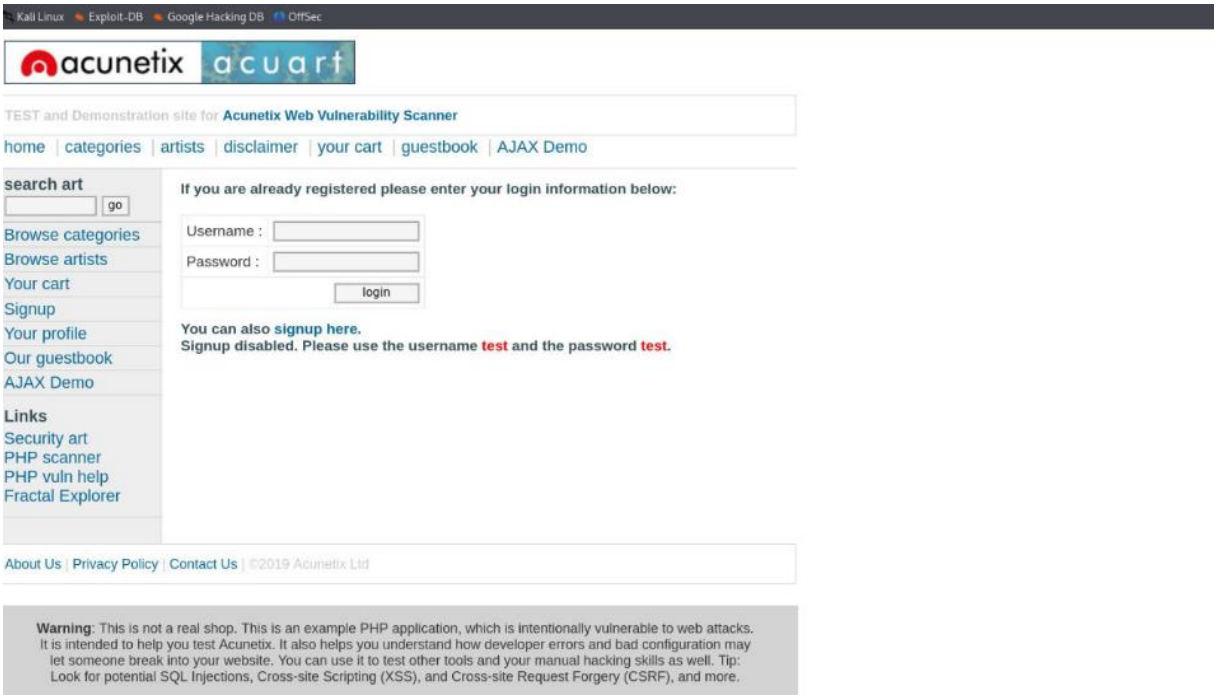
- Sensitive Information Disclosure exposes confidential data, enabling attackers to exploit personal, financial, or system details for fraud, identity theft, or further attacks.

Severity: Critical

Mitigation:

Remove the user credentials

Ask the user to reset the password



XSS Attack

Description:

- XSS allows an attacker to inject malicious JavaScript into a webpage, which then executes in a victim's browser. This can be used to steal cookies, redirect users, or modify the page content.

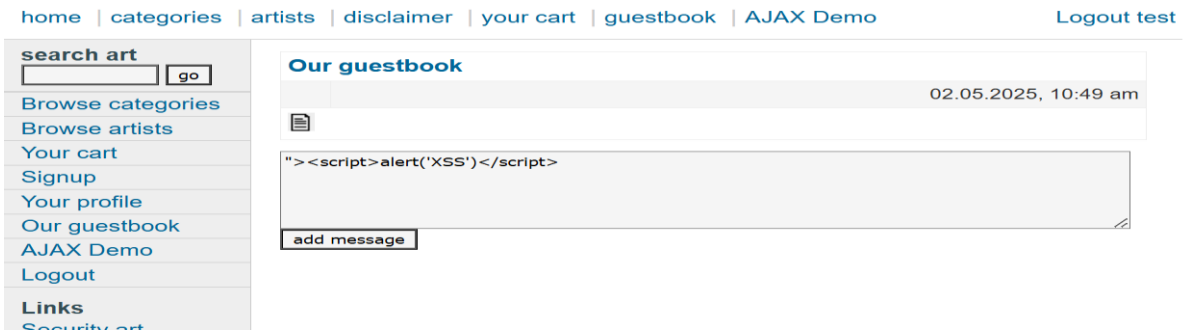
Impact of XSS:

- XSS can lead to stolen cookies, session hijacking, and unauthorized actions on behalf of the user. It may also redirect users to malicious sites or modify web page content, damaging user trust and system integrity.

Severity: High

Steps to Reproduce:

1. Go to http://testphp.vulnweb.com/guestbook.php.
2. Locate the textbox and Add Message button.
3. Enter payload ""><script>alert('XSS')</script> in the textbox.
4. Click the Add Message button.
5. Check if an alert box pops up with 'XSS'.
6. Try alternate payload <img src="x" onerror="alert('XSS')">.
7. Inspect page source (Ctrl+U) or open Developer Tools (F12).



Mitigation:

Sanitize and validate user inputs to prevent script injection. Use Content Security Policy (CSP) and HttpOnly flags for cookies to restrict script access, and implement proper output encoding to safely display data.