

Vulnerability Assessment and Penetration Testing Report

Client / Website Tested
testphp.vulnweb.com
Assessment Type
Vulnerability Assessment and penetration Testing
Intern Name
Rushikesh Borse Cyber Security Intern – Future Interns (2026)
Assessment Date
February 2026

- The assessment was carried out using: Tools

- Browser Developer Tools
- Nmap (basic exposure analysis)
- OWASP ZAP (Passive Scan only)

1. No authentication bypass, brute-force attempts, or denial-of-service activities were performed.

1. Executive Summary

This Vulnerability Assessment was conducted to identify common security weaknesses present on a publicly accessible website using non-intrusive and ethical methods.

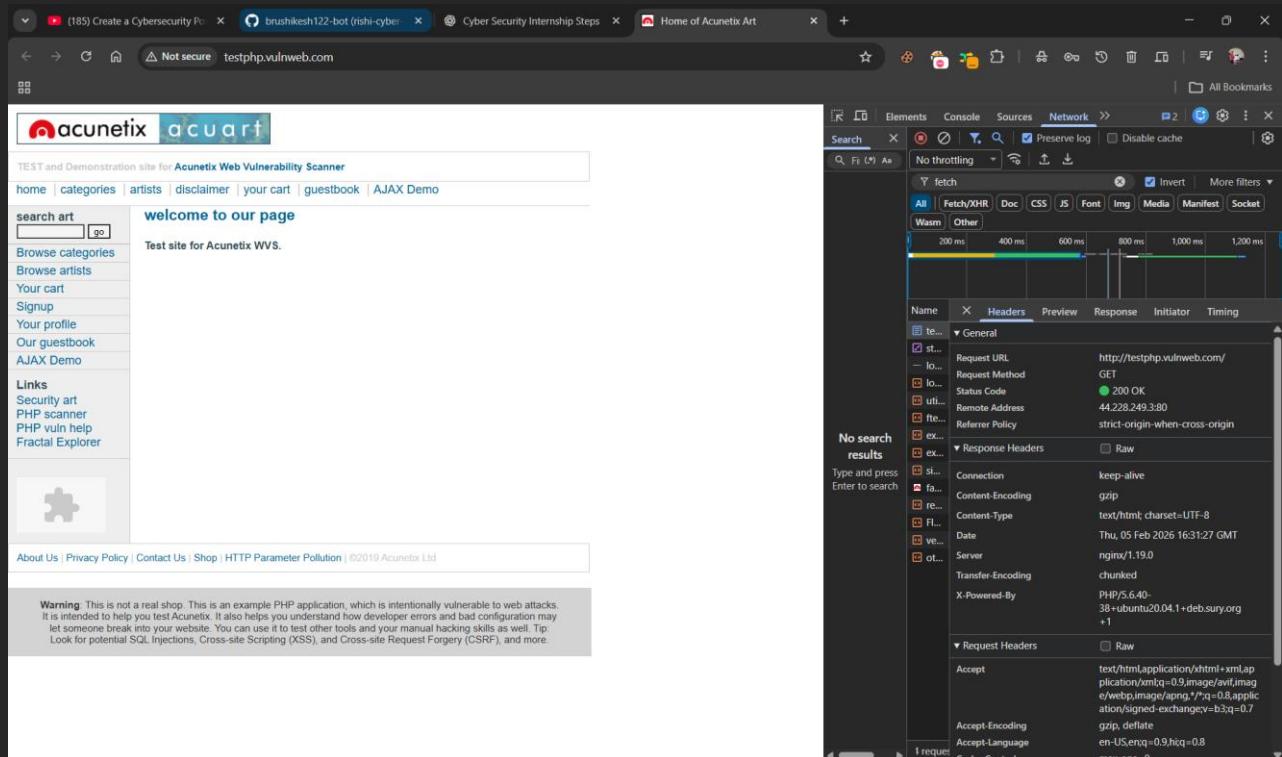
The objective of this assessment is not to exploit or attack the website, but to provide clarity to a business owner about potential security risks and recommend practical remediation steps.

Finding 1: Missing Security Headers

Tool Used: Browser DevTools

What is the issue?

The website is accessible over HTTP, which means data is transmitted without encryption.



Why does it matter?

Without encryption, sensitive information such as login details or user data can be intercepted by attackers. This reduces user trust and may lead to data exposure.

Risk Level :- High

Suggested Remediation (Business-Friendly Fix)

- Enable HTTPS using an SSL/TLS certificate
- Redirect all HTTP traffic to HTTPS

-

X	Headers	Preview	Response	Initiator	Timing
▼ General					
Request URL					
			<code>http://testphp.vulnweb.com/</code>		
Request Method					
			<code>GET</code>		
Status Code					
			● 200 OK		
Remote Address					
			<code>44.228.249.3:80</code>		
Referrer Policy					
			<code>strict-origin-when-cross-origin</code>		

observed Missing Headers:

- Content-Security-Policy (CSP)
- X-Frame-Options
- X-Content-Type-Options

Why This Matters:

Missing headers increase the risk of:

- Clickjacking
- Cross-Site Scripting (XSS)
- Content injection attacks

Risk Level: Medium

Recommended Remediation:

- Implement standard security headers at the server level

Finding 2

Nmap 1 : Open HTTP Port (Port 80)

Tool Used: Nmap

Command Example: **Nmap -F testphp.vulnweb.com**

```
(kali㉿kali)-[~]
└─$ sudo su
[sudo] password for kali:
(root㉿kali)-[~/home/kali]
# nmap -F testphp.vulnweb.com

Starting Nmap 7.95 ( https://nmap.org ) at 2026-02-06 10:44 IST
Nmap scan report for testphp.vulnweb.com (44.228.249.3)
Host is up (0.39s latency).
rDNS record for 44.228.249.3: ec2-44-228-249-3.us-west-2.compute.amazonaws.com
Not shown: 99 filtered tcp ports (no-response)
PORT      STATE SERVICE
80/tcp    open  http

Nmap done: 1 IP address (1 host up) scanned in 8.32 seconds
```

What was checked?

- Scanned the most commonly used ports to quickly identify exposed services.

Identified Issue: - Common web service ports are open and reachable

Why does it matter? - Open ports increase the attack surface if not properly secured

Risk Level :- Medium

Clear Remediation Steps

- Close unused ports
- Restrict access using firewall rules

Nmap 2: Default Nmap Scan

Tool Used: Nmap

Command Example: nmap testphp.vulnweb.com

The screenshot shows a terminal window on a Kali Linux system. The user has run the command `nmap testphp.vulnweb.com`. The output indicates that the host is up and port 80 is open, serving an HTTP service. A security note mentions that the website exposes services publicly without encryption.

```
Starting Nmap 7.95 ( https://nmap.org ) at 2026-02-06 11:58 IST
Nmap scan report for testphp.vulnweb.com (44.228.249.3)
Host is up (0.29s latency).
rDNS record for 44.228.249.3: ec2-44-228-249-3.us-west-2.compute.amazonaws.com
Not shown: 999 filtered tcp ports (no-response)
PORT      STATE SERVICE
80/tcp    open  http

Nmap done: 1 IP address (1 host up) scanned in 22.78 seconds
```

What was checked?

- A standard scan to identify **open ports** and **basic service exposure** on the target server.
- Why does it matter?**
- Port 80 (HTTP) is open and publicly accessible
Web service is exposed without enforced encryption

Identified Issue

- Port 80 (HTTP) is open and publicly accessible
- Web service is exposed without enforced encryption

Risk Level :- Medium

Clear Remediation Steps

- Redirect all HTTP traffic to HTTPS

Nmap 2 : Operating System Detection

Tool Used: Nmap

Command Example: **Nmap -O testphp.vulnweb.com**

The screenshot shows a terminal window titled 'kali@kali: ~'. The command '\$ nmap -O testphp.vulnweb.com' is run, followed by the Nmap scan report. The report indicates the host is up with 0.29s latency, an rDNS record for ec2-44-228-249-3.us-west-2.compute.amazonaws.com, and 999 filtered TCP ports. Port 80 is open and identified as HTTP. OS fingerprinting suggests Linux 4.15 (97%), 2.6.32 (91%), 2.6.32 - 3.13 (91%), 3.10 - 4.11 (91%), 3.2 - 4.14 (91%), 3.4 - 3.10 (91%), 4.15 - 5.19 (91%), 4.19 (91%), 5.0 - 5.14 (91%), and 5.1 - 5.15 (91%). No exact OS match is found due to non-ideal testing conditions. The scan took 32.23 seconds.

```
(kali㉿kali)-[~] $ nmap -O testphp.vulnweb.com
Starting Nmap 7.95 ( https://nmap.org ) at 2026-02-06 12:08 IST
Nmap scan report for testphp.vulnweb.com (44.228.249.3)
Host is up (0.29s latency).
rDNS record for 44.228.249.3: ec2-44-228-249-3.us-west-2.compute.amazonaws.com
Not shown: 999 filtered tcp ports (no-response)
PORT      STATE SERVICE
80/tcp    open  http
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Linux 4.15 (97%), Linux 2.6.32 (91%), Linux 2.6.32 - 3.13 (91%), Linux 3.10 - 4.11 (91%), Linux 3.2 - 4.14 (91%), Linux 3.4 - 3.10 (91%), Linux 4.15 - 5.19 (91%), Linux 4.19 (91%), Linux 5.0 - 5.14 (91%), Linux 5.1 - 5.15 (91%)
No exact OS matches for host (test conditions non-ideal).g System Detection (Information Disclosure)

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 32.23 seconds
```

What was checked?

- Attempted to identify the operating system of the server.

- Why does it matter?

OS fingerprinting information may be inferred

Identified Issue :-

- Knowing the operating system helps attackers craft targeted attacks.

Risk Level :- Low

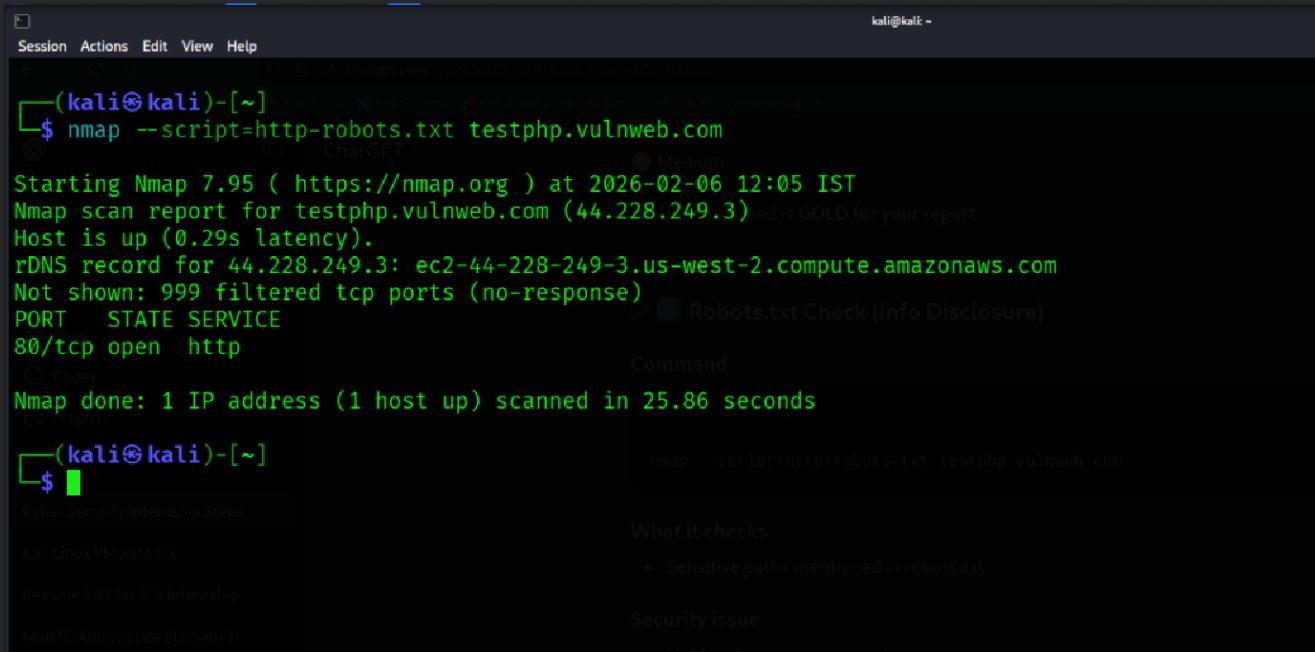
Clear Remediation Steps

- Implement firewall and intrusion prevention rules
- Apply OS hardening measures

Nmap 3 : robots.txt Information Disclosure

Tool Used: Nmap

Command Example: **nmap --script=http-robots.txt testphp.vulnweb.com**



```
kali㉿kali:[~] $ nmap --script=http-robots.txt testphp.vulnweb.com
Starting Nmap 7.95 ( https://nmap.org ) at 2026-02-06 12:05 IST
Nmap scan report for testphp.vulnweb.com (44.228.249.3)
Host is up (0.29s latency).
rDNS record for 44.228.249.3: ec2-44-228-249-3.us-west-2.compute.amazonaws.com
Not shown: 999 filtered tcp ports (no-response)
PORT      STATE SERVICE
80/tcp    open  http
Nmap done: 1 IP address (1 host up) scanned in 25.86 seconds
```

What was checked?

- Reviewed the robots.txt file for exposed paths.
- Why does it matter?
robots.txt file is publicly accessible and reveals internal paths

Identified Issue

- Attackers can use these paths to locate hidden or sensitive sections of the website.

Risk Level :- Low

Clear Remediation Steps

- Avoid listing sensitive directories in robots.txt
- Use authentication and access control instead

Finding: - 3

OWASP ZAP Vulnerability Assessment (Passive Scan)

- **Tool Used:** OWASP ZAP (Zed Attack Proxy)

The screenshot shows the OWASP ZAP interface in Standard Mode. The top menu bar includes File, Edit, View, Analyse, Report, Tools, Import, Export, Online, and Help. The title bar indicates "Untitled Session - 20260208-160615 - ZAP 2.17.0". The main window has tabs for Header, Text, Body, and Response. The Header tab shows an HTTP response with status code 200 OK, version 1.1, and various headers including Server, Date, Content-Type, and X-Powered-By. The Body tab displays the raw HTML source code of the page. The left sidebar lists "Contests" and "Sites", with "testphp.vulnweb.com" selected. The bottom navigation bar includes History, Search, Alerts, Output, and a "+" button. The Alerts panel is open, showing a single critical alert: "Absence of Anti-CSRF Tokens (Systemic)". The alert details are as follows:

URL	http://testphp.vulnweb.com/categories.php
Risk	Medium
Confidence	Low
Parameter	Attack
Evidence	<form action="search.php?test=query" method="post">
CWE ID	352
WASC ID	9
Source	Passive (10202 - Absence of Anti-CSRF Tokens)
Input Vector	Description
No Anti-CSRF tokens were found in a HTML submission form.	
A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The underlying cause is application functionality using predictable URL/form actions in a repeatable way. The nature of the attack is that CSRF exploits the trust that a web site has for other info.	
No known Anti-CSRF token [anticrf, CSRFToken, __RequestVerificationToken, csrfmiddlewaretoken, authencity_token, OWASP_CSRFTOKEN, anonrcrf, csrf_token__crlfSecret, _crlf_magic, CSRF_token, csrf_token, csrfToken] was found in the following HTML form: [Form 1: "goButton" "searchFor"]	

- **Scan Type:** Passive Scan (Read-Only)
- **Target Website:** testphp.vulnweb.com
- **Testing Approach:** Non-intrusive, ethical security analysis

Methodology

OWASP ZAP was used as a **passive security analysis tool** to monitor traffic between the browser and the website.

The tool identified common web security misconfigurations and weaknesses **without sending malicious requests**.

Owasp zap 1 : Missing Content Security Policy (CSP)

Tool Used: Owasp zap

Command Example: **nmap --script=http-robots.txt**
testphp.vulnweb.com

The screenshot shows the Owasp Zap interface. At the top, there's a menu bar with File, Edit, View, Analyse, Report, Tools, Import, Export, Online, Help. Below it is a toolbar with various icons. The main window has tabs for Header, Text, Body, Text, Request, Response, and Requester. On the left, there's a tree view of 'Sites' and 'Contents'. The 'Contents' section shows a single item: 'Default Context'. The 'Sites' section shows a single item: 'http://testphp.vulnweb.com'. In the center, there's a large text area showing an HTTP response header and its corresponding HTML content. The header includes fields like Server, Date, Content-Type, Content-Length, and X-Powered-By. The HTML content is a simple form page with a title, a search input field, and some descriptive text. At the bottom, there's a detailed alert list titled 'Absence of Anti-CSRF Tokens'. It lists several findings under this category, such as 'Content Security Policy (CSP) Header Not Set (Systemic)', 'Missing Anti-clickjacking Header (Systemic)', and 'Server Leaks Version Information via "Server" HTTP Response Header (Systemic)'. Each finding has details like URL, Risk level (Medium), Confidence (Low), Attack type (Cross-site request forgery), Evidence (HTML code snippet), CWE ID (352), WASC ID (9), Source (Passive), Input Vector, and Description (explaining the attack). There are also sections for 'Other Info' and 'Known Anti-CSRF token'. The bottom right corner shows current status metrics.

What was checked?

- The website does not implement a Content Security Policy (CSP) header.
- Why does it matter?

Without CSP, attackers may inject malicious scripts into the website, potentially leading to data theft or page manipulation.

Identified Issue

- Attackers can use these paths to locate hidden or sensitive sections of the website.

Risk Level: - medium

Clear Remediation Steps

- Review and update CSP rules during application changes

Finding 2: Missing X-Frame-Options Header

The screenshot shows the ZAP interface with the following details:

- Header Text:** Shows the raw HTTP response headers:

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Fri, 06 Feb 2020 18:36:26 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
Content-Length: 6189
```
- Body Text:** Shows the raw HTML response body:

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html><!-- InstanceBegin template="/Templates/main_dynamic_template.dwt.php" codeOutsideHTMLIsLocked="false" -->
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-2">
<!-- InstanceBeginEditable name="document_title_rgn" -->
<title>picture categories</title>
<!-- InstanceEndEditable -->
<link rel="stylesheet" href="style.css" type="text/css">
<!-- InstanceBeginEditable name="headers_rgn" -->
<!-- here goes headers headers -->
<!-- InstanceEndEditable -->
```
- Alerts:** A detailed alert for "Absence of Anti-CSRF Tokens (Systemic)" is expanded, showing the following information:
 - URL: <http://testphp.vulnweb.com/categories.php>
 - Risk: Medium
 - Confidence: Low
 - Parameter: Attack
 - Evidence: <form action="search.php?test=query" method="post">
 - CWE ID: 352
 - WASC ID: 9
 - Source: Passive (10202 - Absence of Anti-CSRF Tokens)
 - Input Vector:
 - Description: No Anti-CSRF tokens were found in a HTML submission form. A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The underlying cause is application functionality using predictable URL/form actions in a repeatable way. The nature of the attack is that CSRF exploits the trust that a web site has for a user. By contrast, cross-site scripting (XSS) attacks exploit the trust that a web site has for a user's input.
 - Other Info: No known Anti-CSRF token [anticarf, CSRFToken, _RequestVerificationToken, csrfmiddlewaretoken, authenticity_token, OWASP_CSRFTOKEN, antonicsrf, csrf_token, _csrf, _csrfSecret, __csrf_magic, CSRF_token, __csrf_token, __csrfToken] was found in the following HTML form: [Form 1: "goButton" "searchFor"]

What is the issue?

The website does not include the **X-Frame-Options** header.

Why does it matter?

This can allow attackers to embed the website inside a malicious page, leading to **clickjacking attacks**.

Risk Level: - Medium

Suggested Remediation

- Set X-Frame-Options to DENY or SAMEORIGIN
- Prevent unauthorized framing of the website