Vulnerability Assessment Report

Security Evaluation of testphp.vulnweb.com

NAME: COLLINS KARURI KING'ORI

CIN ID: FIT/JAN26/CS5913

Executive Summary

This report presents the findings of a vulnerability assessment conducted on testphp.vulnweb.com. The objective of the assessment was to identify potential security weaknesses that may expose the web application to cyber threats. The assessment was performed using Nmap, OWASP ZAP, and browser developer tools. Several vulnerabilities were identified, including insecure cookies and missing security headers. Appropriate recommendations have been provided to enhance the overall security posture of the system.

Scope of Assessment

- Target Website: http://testphp.vulnweb.com
- Assessment Type: Passive Vulnerability Assessment
- Tools Used:1. Nmap 2.OWASP ZAP 3.Browser Developer Tools
- Testing Date: 23/2/2026

METHODOLOGY

- Port scanning using Nmap
- Passive vulnerability scanning using OWASP ZAP
- Manual inspection of HTTP headers and cookies
- Risk classification and documentation

○ **vulnerability**	**risk level**	impact	status
Insecure cookies	Medium	Session hijacking	open
Missing Security Headers	medium	browser-based attacks	open
information disclosure	low	server info exposure	open

FINDINGS

Missing HTTP Security Headers

Description

The vulnerability assessment identified that the web application does not implement important HTTP security headers such as Content-Security-Policy, X-Frame-Options, X-Content-Type-Options, and Strict-Transport-Security. These headers play a critical role in protecting web applications against common client-side attacks.

Risk Level

Medium

Impact

The absence of these security headers increases the risk of browser-based attacks such as Cross-Site Scripting (XSS), clickjacking, and MIME-type sniffing. Attackers may exploit this weakness to inject malicious scripts, trick users into revealing sensitive information, or manipulate how content is displayed in the browser.

Recommendation

The web server should be configured to include the missing security headers. Specifically:

- Implement a strong Content-Security-Policy (CSP).
- Add X-Frame-Options set to DENY or SAMEORIGIN.
- Enable X-Content-Type-Options set to nosniff.
- Enforce HTTPS using Strict-Transport-Security (HSTS).

5.2 Insecure Cookie Configuration

Description

The assessment revealed that cookies set by the application do not have the Secure and HttpOnly attributes enabled. These attributes are essential in protecting session cookies from being accessed or transmitted insecurely.

Risk Level

Medium

Impact

Without the Secure flag, cookies may be transmitted over unencrypted HTTP connections, making them vulnerable to interception through Man-in-the-Middle (MITM) attacks. Without the HttpOnly flag, malicious scripts executed through Cross-Site Scripting (XSS) attacks can access and steal session cookies. This may result in session hijacking and unauthorized access to user accounts.

Recommendation

The application should configure cookies with the following attributes:

- Secure (to ensure transmission over HTTPS only)
- HttpOnly (to prevent access via JavaScript)
- SameSite set to Strict or Lax to mitigate CSRF attacks

Proper cookie configuration significantly reduces the risk of session compromise.

5.3 Information Disclosure Through HTTP Headers

Description

The server exposes information about its underlying software and configuration through HTTP response headers. This includes details such as the web server type and version.

Risk Level

Low

Impact

Although this vulnerability may not directly compromise the system, exposing server information assists attackers in identifying known vulnerabilities associated with specific software versions. This information can be used to plan targeted attacks against the application.

Recommendation

The server should be configured to suppress or obfuscate detailed version information in HTTP response headers. Removing unnecessary header disclosures reduces the amount of information available to potential attackers.

5.4 Use of Unencrypted HTTP Communication

Description

The application allows access over HTTP without enforcing HTTPS redirection. Data transmitted over HTTP is not encrypted and can be intercepted by attackers.

Risk Level

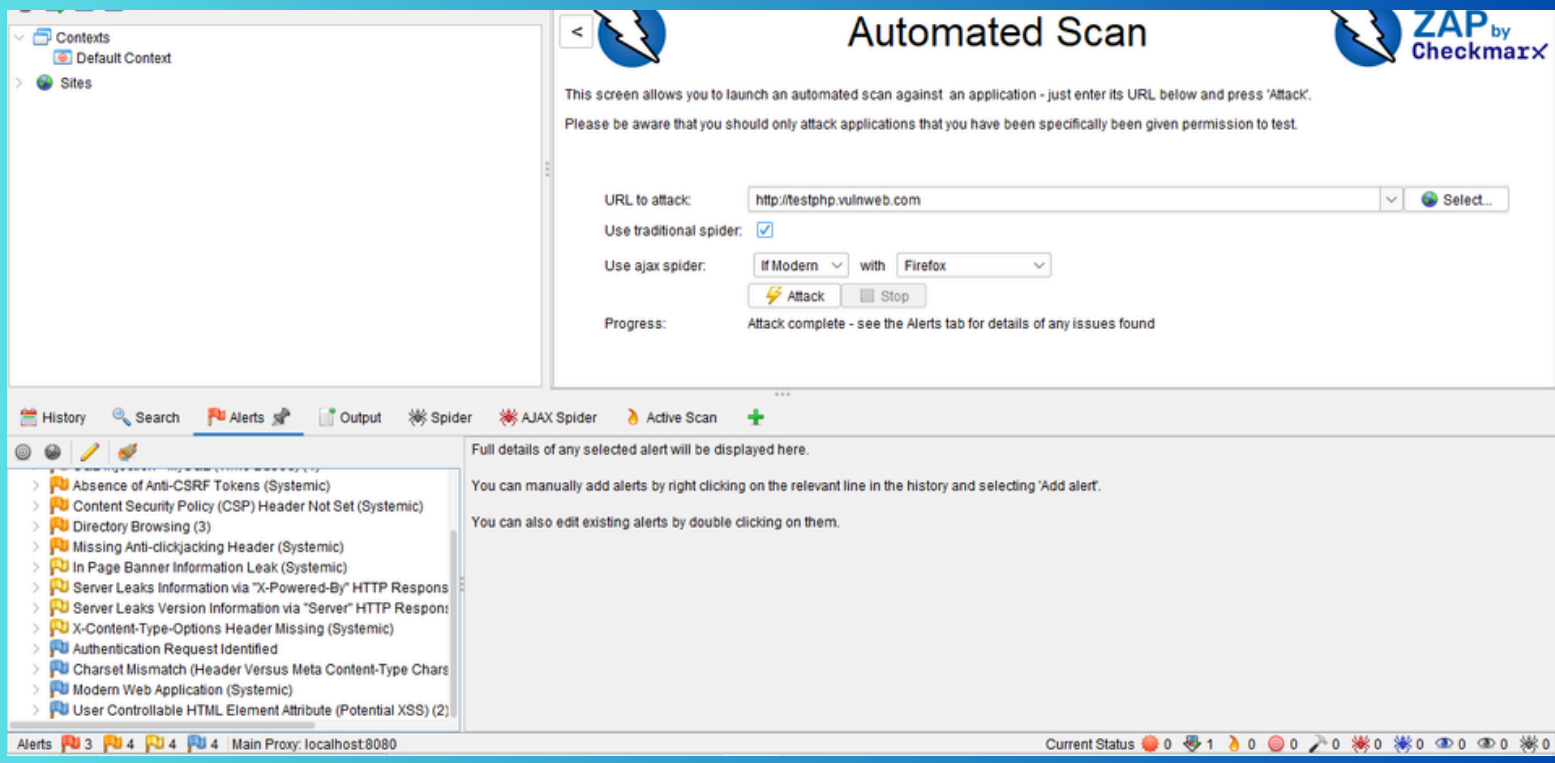
High

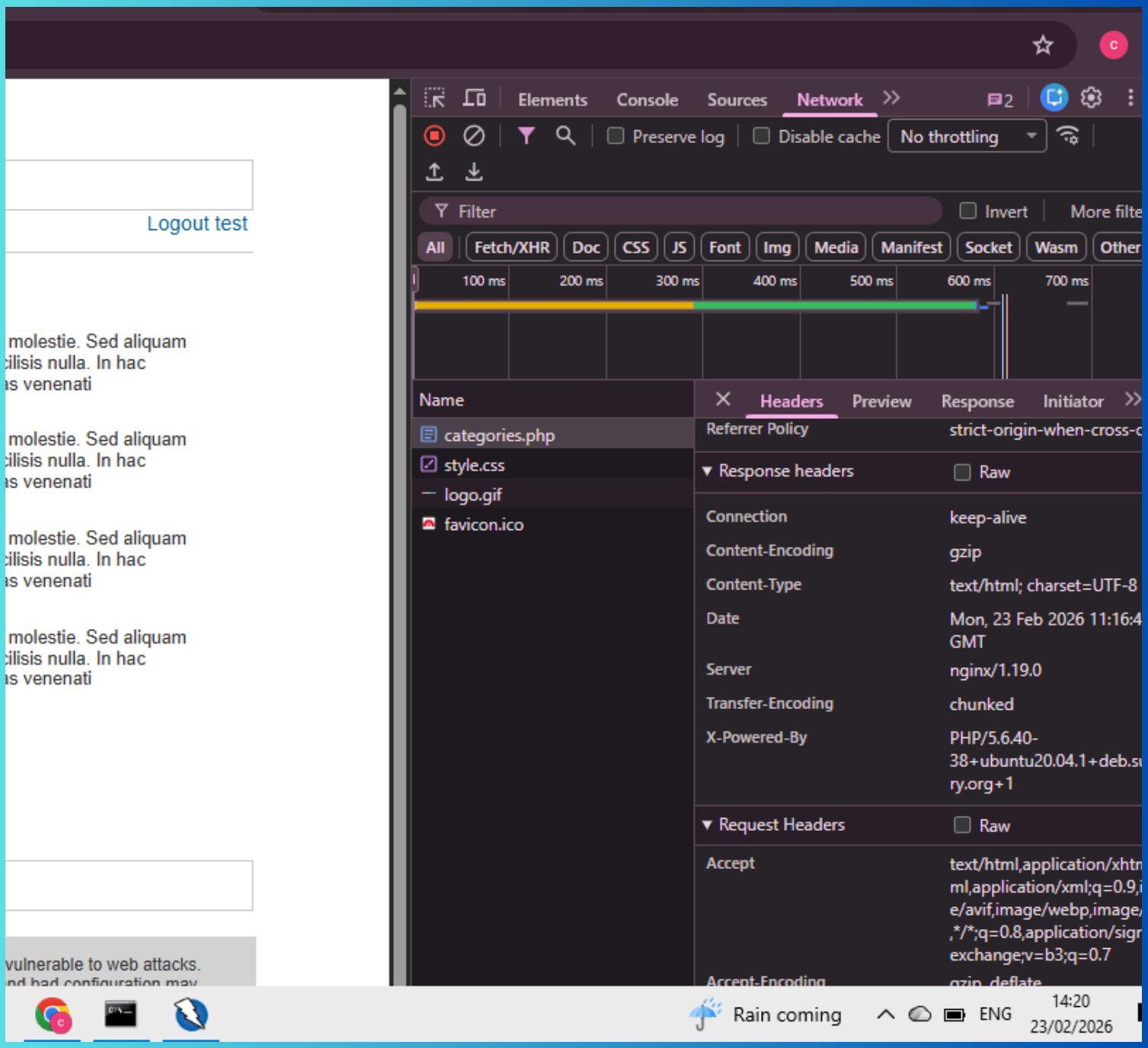
Impact

Unencrypted communication exposes sensitive information such as login credentials, session cookies, and user data to interception. Attackers positioned on the same network can perform Man-in-the-Middle attacks to capture or modify transmitted data.

Recommendation

The application should implement SSL/TLS encryption and enforce HTTPS across all pages. Additionally, HTTP traffic should be automatically redirected to HTTPS, and HTTP Strict Transport Security (HSTS) should be enabled to ensure secure connections.


Contexts
Default Context
Sites
Automated Scan
ZAP by Checkmarx
This screen allows you to launch an automated scan against an application - just enter its URL below and press 'Attack'.
Please be aware that you should only attack applications that you have been specifically been given permission to test.
URL to attack:
http://testphp.vulnweb.com
Select...
Use traditional spider:
Use ajax spider:
If Modern
with
Firefox
Attack
Stop
Progress:
Attack complete - see the Alerts tab for details of any issues found
History
Search
Alerts
Output
Spider
AJAX Spider
Active Scan
Full details of any selected alert will be displayed here.
You can manually add alerts by right clicking on the relevant line in the history and selecting 'Add alert'.
You can also edit existing alerts by double clicking on them.
Absence of Anti-CSRF Tokens (Systemic)
Content Security Policy (CSP) Header Not Set (Systemic)
Directory Browsing (3)
Missing Anti-clickjacking Header (Systemic)
In Page Banner Information Leak (Systemic)
Server Leaks Information via "X-Powered-By" HTTP Respons
Server Leaks Version Information via "Server" HTTP Respons
X-Content-Type-Options Header Missing (Systemic)
Authentication Request Identified
Charset Mismatch (Header Versus Meta Content-Type Chars
Modern Web Application (Systemic)
User Controllable HTML Element Attribute (Potential XSS) (2)
Alerts 3 4 4 4 Main Proxy: localhost:8080
Current Status 0 1 0 0 0 0 0 0 0 0


Logout test
molestie. Sed aliquam
cilisis nulla. In hac
as venenati
vulnerable to web attacks.
Elements
Console
Sources
Network
Preserve log
Disable cache
No throttling
Filter
Invert
All
Fetch/XHR
Doc
CSS
JS
Font
Img
Media
Manifest
Socket
Wasm
100 ms
200 ms
300 ms
400 ms
500 ms
600 ms
700 ms
Name
categories.php
style.css
logo.gif
favicon.ico
Headers
Preview
Response
Initiator
Referrer Policy
strict-origin-when-cross-
Response headers
Raw
Connection
keep-alive
Content-Encoding
gzip
Content-Type
text/html; charset=UTF-8
Date
Mon, 23 Feb 2026 11:16:4
GMT
Server
nginx/1.19.0
Transfer-Encoding
chunked
X-Powered-By
PHP/5.6.40-
38+ubuntu20.04.1+deb.s
ry.org+1
Request Headers
Raw
Accept
text/html,application/xhtm
ml,application/xml;q=0.9,
e/avif,image/webp,image/
,*/*;q=0.8,application/sign
exchange;v=b3;q=0.7
Accept-Encoding
gzip, deflate
Rain coming
ENG
14:20
23/02/2026

Conclusion

The assessment identified several security weaknesses within the tested web application. While most vulnerabilities were classified as medium risk, failure to address them could result in exploitation. Implementing the recommended measures will significantly improve the system’s security posture and reduce exposure to cyber threats.

Overall Security Recommendations

- Implement HTTPS strictly
- Enable Secure and HttpOnly cookie flags
- Configure security headers
- Perform regular vulnerability scans
- Keep server software updated