VULNERABILITY ASSESSMENT REPORT

READ-ONLY SECURITY REVIEW OF A PUBLIC WEB APPLICATION

PREPARED BY: DHANUSH KUMAR

INTERNSHIP PROGRAM: FUTURE INTERNS – CYBER SECURITY (2026)

ASSESSMENT DATE: JANUARY 2026

TARGET WEBSITE: TESTPHP.VULNWEB.COM

Executive Summary

This Vulnerability Assessment Report documents the results of a read-only security review conducted on a publicly accessible web application. The purpose of this assessment was to identify common security weaknesses and configuration gaps that could expose the application to potential cyber risks.

The assessment was performed using passive analysis techniques only. No exploitation, authentication bypass, data modification, or denial-of-service activities were conducted. All findings are based on publicly observable information such as HTTP response headers, network exposure, and visible application behavior.

The review identified multiple low to medium risk security issues, primarily related to missing HTTP security headers, insecure transport configuration, and server information disclosure. While no critical vulnerabilities were observed, addressing the identified issues would significantly strengthen the website's security posture, reduce attack surface, and improve user trust.

Scope of Assessment

In Scope

- *Public-facing web pages*
- *Passive traffic observation*
- *HTTP header analysis*
- *Network service exposure review*

Out of Scope

- *Login or authentication testing*
- *Exploitation of vulnerabilities*
- *Brute-force attacks*
- *Denial-of-Service (DoS) testing*
- *Any activity that could disrupt the target website*

METHODOLOGY

THE ASSESSMENT FOLLOWED A STRUCTURED, ETHICAL APPROACH ALIGNED WITH INDUSTRY-STANDARD SECURITY AUDITING PRACTICES:

RECONNAISSANCE (PASSIVE)

IDENTIFICATION OF EXPOSED SERVICES AND BASIC NETWORK CONFIGURATION USING NON-INTRUSIVE SCANNING.

CONFIGURATION ANALYSIS

REVIEW OF HTTP RESPONSE HEADERS AND TRANSPORT SECURITY SETTINGS.

CLIENT-SIDE INSPECTION

ANALYSIS OF BROWSER-VISIBLE SECURITY CONTROLS USING DEVELOPER TOOLS.

RISK CLASSIFICATION

EACH FINDING WAS CLASSIFIED AS LOW, MEDIUM, OR HIGH BASED ON POTENTIAL BUSINESS IMPACT AND LIKELIHOOD.

REPORTING

FINDINGS WERE DOCUMENTED WITH CLEAR EXPLANATIONS AND PRACTICAL REMEDIATION STEPS SUITABLE FOR NON-TECHNICAL STAKEHOLDERS.

TOOLS USED

- *NMAP: BASIC PORT AND SERVICE EXPOSURE ANALYSIS*
- *BROWSER DEVELOPER TOOLS: INSPECTION OF HTTP HEADERS AND CLIENT-SIDE BEHAVIOR*
- *MOZILLA HTTP OBSERVATORY: SECURITY HEADER AND CONFIGURATION ANALYSIS*
- *CANVA: PROFESSIONAL REPORT DESIGN AND PRESENTATION*

FINDINGS AND RISK ANALYSIS

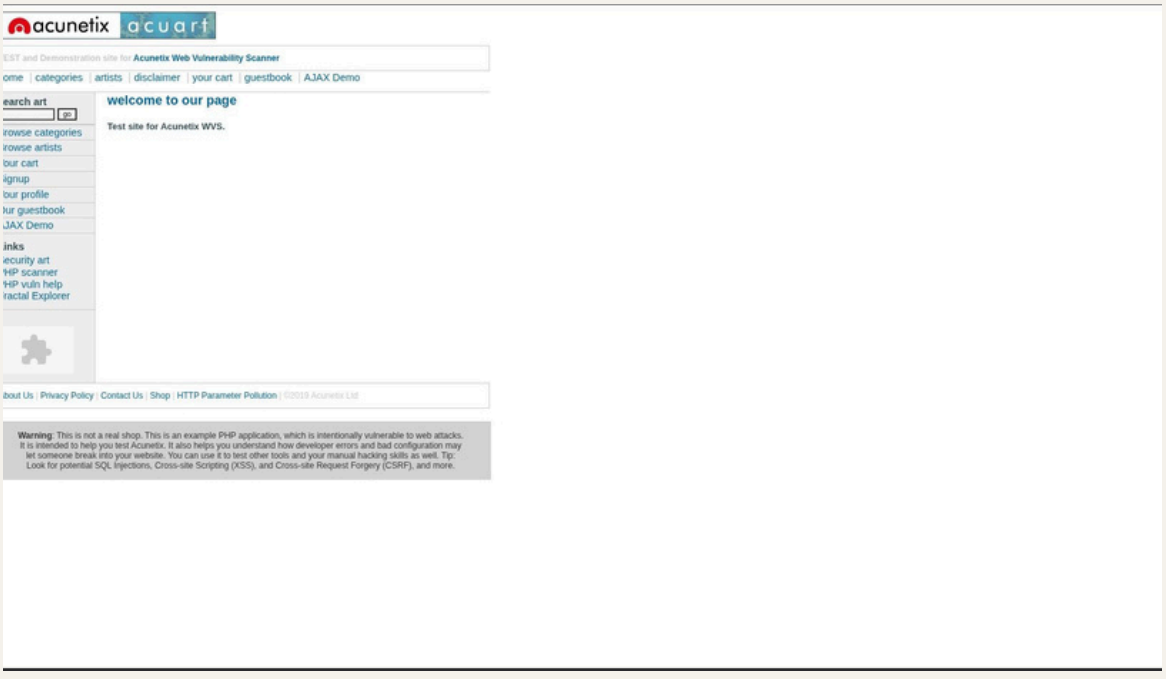
Finding 1: Missing Content Security Policy (CSP)

Risk Level: Medium

Description:
The application does not implement a Content Security Policy (CSP). CSP is a security mechanism that helps prevent cross-site scripting (XSS) and other code injection attacks by defining which sources of content are trusted.

Business Impact:
Without CSP, an attacker may be able to inject malicious scripts that execute in users' browsers, potentially leading to session hijacking, data theft, or defacement.

Recommendation:
Implement a strong Content Security Policy header that restricts script, style, and resource loading to trusted sources only.



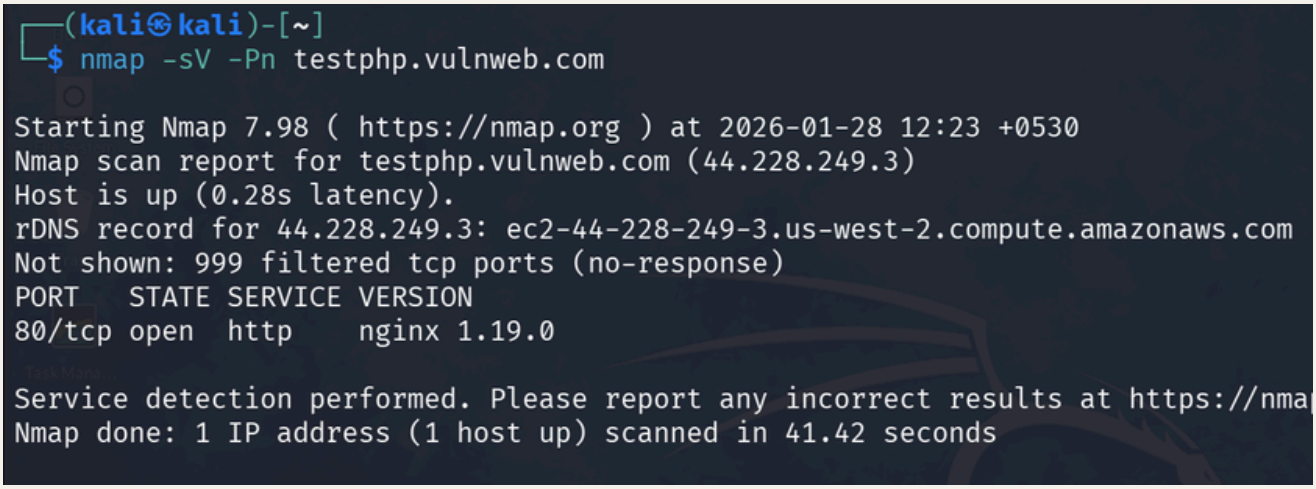
Finding 2: Missing X-Frame-Options Header

Risk Level: Medium

Description:
The application does not include the X-Frame-Options header, which protects against clickjacking attacks by preventing the site from being embedded in malicious frames.

Business Impact:
Attackers could trick users into clicking on hidden elements, potentially leading to unauthorized actions or data exposure.

Recommendation:
Configure the server to include the X-Frame-Options header (DENY or SAMEORIGIN) or define equivalent protections using CSP frame-ancestors.



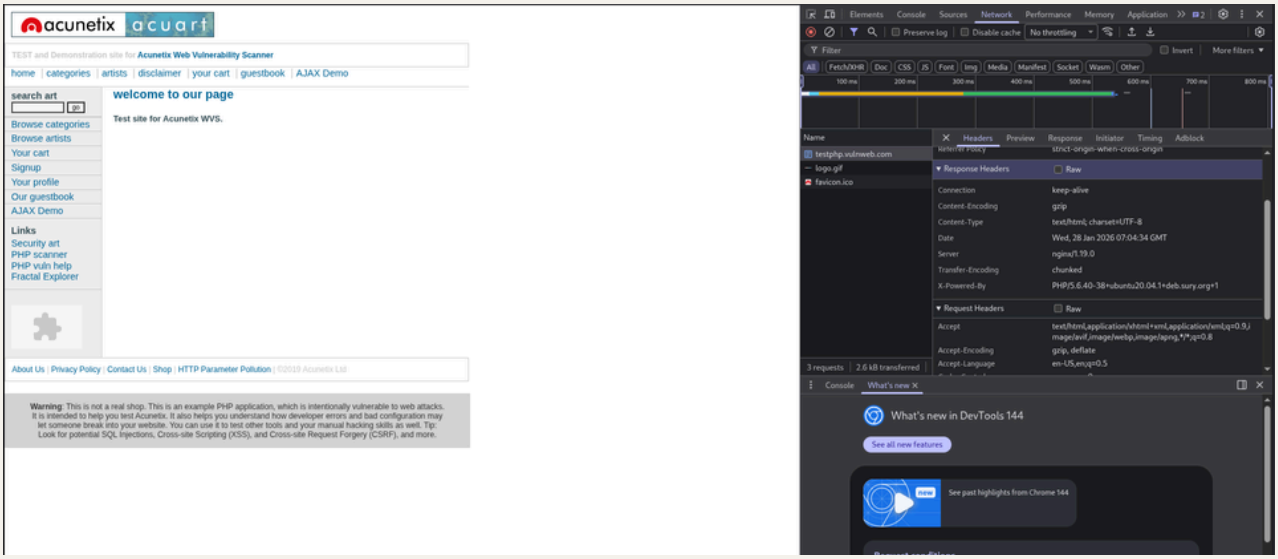
Finding 3: Missing X-Content-Type-Options Header

Risk Level: Low

Description:
The application does not set the X-Content-Type-Options header, allowing browsers to perform MIME-type sniffing.

Business Impact:
This could allow attackers to disguise malicious content as safe file types, increasing the risk of script execution.

Recommendation:
Add the header X-Content-Type-Options: nosniff to all server responses.



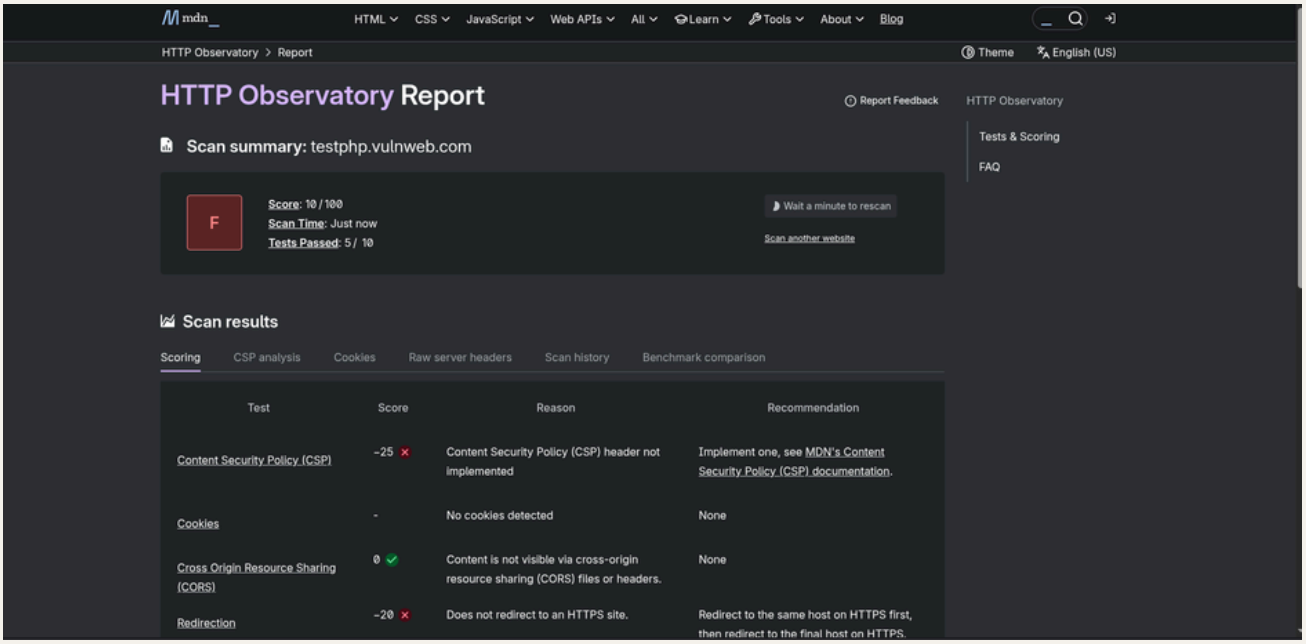
Finding 4: Insecure Transport Configuration (No HTTPS Enforcement / HSTS)

Risk Level: Medium

Description:
The application does not enforce HTTPS connections or implement HTTP Strict Transport Security (HSTS).

Business Impact:
Users may be vulnerable to man-in-the-middle attacks, where attackers intercept or manipulate traffic between the browser and server.

Recommendation:
Enforce HTTPS across the application and enable HSTS to ensure secure connections at all times.



Finding 5: Server Information Disclosure

Risk Level: Low

Description:

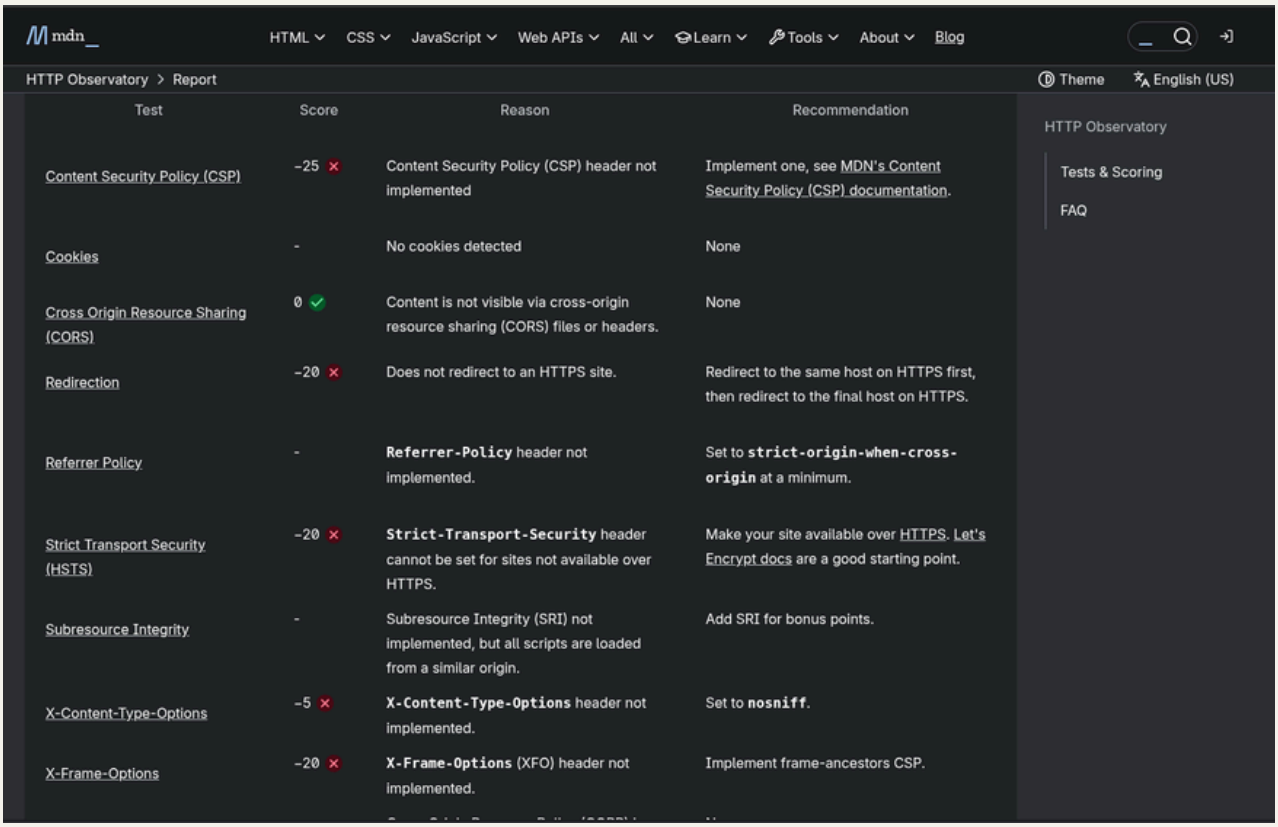
Server headers reveal information about the underlying technology stack.

Business Impact:

Exposed server details can assist attackers during reconnaissance and increase the likelihood of targeted attacks.

Recommendation:

Configure the web server to suppress or generalize server version information in HTTP responses.



Risk Summary Table

Finding	Risk Level	Potential Impact
Missing Content Security Policy	Medium	XSS, script injection
Missing X-Frame-Options	Medium	Clickjacking
Missing X-Content-Type-Options	Low	MIME sniffing
No HTTPS / HSTS	Medium	Man-in-the-middle attacks, data interception
Server Information Disclosure	Low	Reconnaissance and targeted attacks

Conclusion

The vulnerability assessment identified several configuration-level security weaknesses that are commonly found in public web applications. While no critical vulnerabilities were discovered, the identified issues could be exploited if left unaddressed.

By implementing the recommended remediation steps, the organization can significantly reduce its attack surface, enhance user trust, and align the application with modern web security best practices.

Disclaimer

This assessment was conducted for educational and training purposes as part of the Future Interns Cyber Security Internship Program. All testing was limited to passive, read-only techniques on a publicly available demo website. No unauthorized access or harmful activity was performed.