

# VULNERABILITY ASSESSMENT REPORT

**Author:** Grace Wambui

**Date:** 21/02/2026

**Assignment:** Future Interns Task 01

**Website tested:** testphp.vulnweb.com

---

## Table of Contents

<b>VULNERABILITY ASSESSMENT REPORT .....</b>	1
<b>EXECUTIVE SUMMARY .....</b>	1
<b>CRITICAL RISKS IDENTIFIED .....</b>	2
<b>SCOPE &amp; METHODOLOGY .....</b>	2
<b>RISK CLASSIFICATION .....</b>	3
<b>BUSINESS IMPACT .....</b>	10
<b>OVERALL RISK ASSESSMENT .....</b>	10
<b>PRIORITIZED REMEDIATION ROADMAP .....</b>	11
<b>RECOMMENDATIONS .....</b>	11
<b>CONCLUSION .....</b>	11
<b>FINAL REMARKS .....</b>	12

---

## EXECUTIVE SUMMARY

### Overview

This report details the findings of a vulnerability assessment conducted on testphp.vulnweb.com. The assessment aimed to identify potential security weaknesses that could be exploited by attackers. This report outlines the identified vulnerabilities, their potential impact, and recommended remediation steps.

### Summary of Key Findings

Risk Level	Number of Findings	Examples
------------	--------------------	----------

High	2	SQL Injection, Default Admin Credentials
Medium	3	Missing Anti-CSRF Tokens, Exposed CVS Directory, Content Security Policy (CSP) Header Not Set

## CRITICAL RISKS IDENTIFIED

### 1. SQL Injection (High Risk)

An attacker can manipulate database queries through the cat parameter in listproducts.php, potentially leading to data theft, authentication bypass, or complete database compromise.

### 2. Default Admin Credentials (High Risk)

The admin panel is accessible using default credentials (test), allowing unauthorized access to sensitive user data including names, email addresses, and credit card information.

### 3. Missing Anti-CSRF Tokens (Medium Risk)

Forms lack CSRF protection, making users vulnerable to having unintended actions performed on their behalf.

### 4. Exposed CVS Directory (Medium Risk)

Version control metadata is publicly accessible, revealing internal file structure and potentially sensitive information about the application's development history.

### 5. Content Security Policy (CSP) Header Not Set (Medium risk)

The application does not define a Content Security Policy (CSP) header, allowing the browser to load resources from untrusted sources.

## SCOPE & METHODOLOGY

- **Target URL:** testphp.vulnweb.com
- **Assessment Type:** External Vulnerability Assessment
- **Date of Assessment:** 21/02/2026
- **Tools Used:** OWASP ZAP, Browser Dev tools
- **Testing Approach:** Combination of automated scanning and manual verification

## **Methodology Overview**

### **Phase 1: Reconnaissance**

- Initial exploration of the application to understand its structure and functionality
- Manual browsing of all accessible pages including homepage, product listings, guestbook, and admin areas
- Identification of input points (forms, URL parameters)

### **Phase 2: Automated Scanning**

- Configuration of OWASP ZAP as a local proxy
- Automated spidering to discover endpoints
- Active scanning for:
  - SQL Injection
  - Cross-Site Scripting (XSS)
  - Cross-Site Request Forgery (CSRF)
  - Security misconfigurations
  - Information disclosure

### **Phase 3: Manual Verification**

- Manual confirmation of automated findings
- Exploitation attempts to assess impact
- Logic and access control testing

---

### **Testing Standards Referenced**

- OWASP Top 10 v2.17.0
- OWASP Testing Guide v4.0
- CWE (Common Weakness Enumeration)

---

## **RISK CLASSIFICATION**

Risk Rating	Description	Impact	Likelihood
-------------	-------------	--------	------------

High	Critical vulnerability that can be easily exploited	Severe data breach, system compromise	Highly likely to be exploited
Medium	Vulnerability that can be exploited under certain conditions	Moderate data breach partial system compromise	Likely to be exploited
Low	Minor vulnerability with limited impact	Minimal data breach, no system compromise	Unlikely to be exploited

---

## Limitations

- Application is deliberately vulnerable
  - No advanced privilege escalation beyond default credentials
  - External testing only
  - No DoS attacks performed
- 

## FINDING I: SQL INJECTION

**Risk Level:** HIGH

**Location:** <http://testphp.vulnweb.com/listproducts.php?cat=1>

**Parameter:** cat

**CWE:** CWE-89

### Description

The application fails to properly validate user input passed through the cat parameter. Injecting a single quote causes a SQL error, confirming unsensitized input.

### Business Impact

- Data breach
- Authentication bypass
- Data manipulation
- Privilege escalation
- Regulatory violations

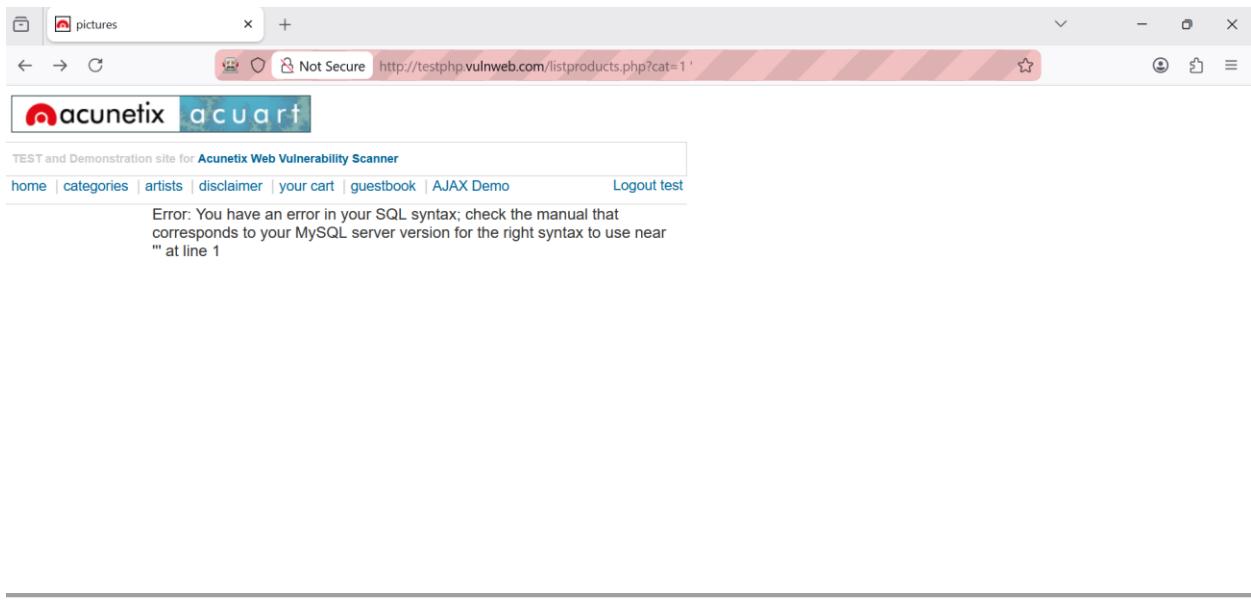
### Remediation

- I. Parameterized queries

2. Input validation
3. Least privilege
4. Error handling
5. Web Application Firewall

### **Evidence:**

Figure 1: SQL error message after injection



## FINDING 2: DEFAULT ADMIN CREDENTIALS

**Risk Level:** HIGH

**Location:** <http://testphp.vulnweb.com/admin/>

**Credentials:** test / test

**CWE:** CWE-798

### **Description**

Admin panel accepts default credentials, granting access to sensitive data.

### **Business Impact**

- Complete system compromise
- Mass data theft
- Identity fraud
- PCI DSS violations

- Legal liability

## Remediation

- Change default credentials immediately
- Implement MFA
- Restrict admin access
- Secure admin panel

## Evidence:

Figure 2: Admin login page with default login credentials

Figure 3: Admin dashboard displaying customer sensitive information

The screenshot shows a web browser window with the URL <http://testphp.vulnweb.com/userinfo.php>. The page is titled "Lg (test)". On the left, there is a sidebar with links like "home", "categories", "artists", "disclaimer", "your cart", "guestbook", "AJAX Demo", "Logout test", "search art", "Browse categories", "Browse artists", "Your cart", "Signup", "Your profile", "Our guestbook", "AJAX Demo", "Links", "Security art", "PHP scanner", "PHP vuln help", and "Fractal Explorer". The main content area displays a form for editing user information. The form fields are: Name (Lg), Credit card number (1234-5678-2300-900), E-Mail (email@email.com), Phone number (2323345), and Address (21 street). Below the form is an "update" button. A message at the bottom of the page states: "You have 2 items in your cart. You visualize you cart [here](#)". At the bottom right of the page is a "Copilot" button.

## FINDING 3: EXPOSED CVS DIRECTORY

**Risk Level:** MEDIUM

**Location:** <http://testphp.vulnweb.com/CVS/>

**CWE:** CWE-548

### Description

CVS version control metadata is publicly accessible.

### Business Impact

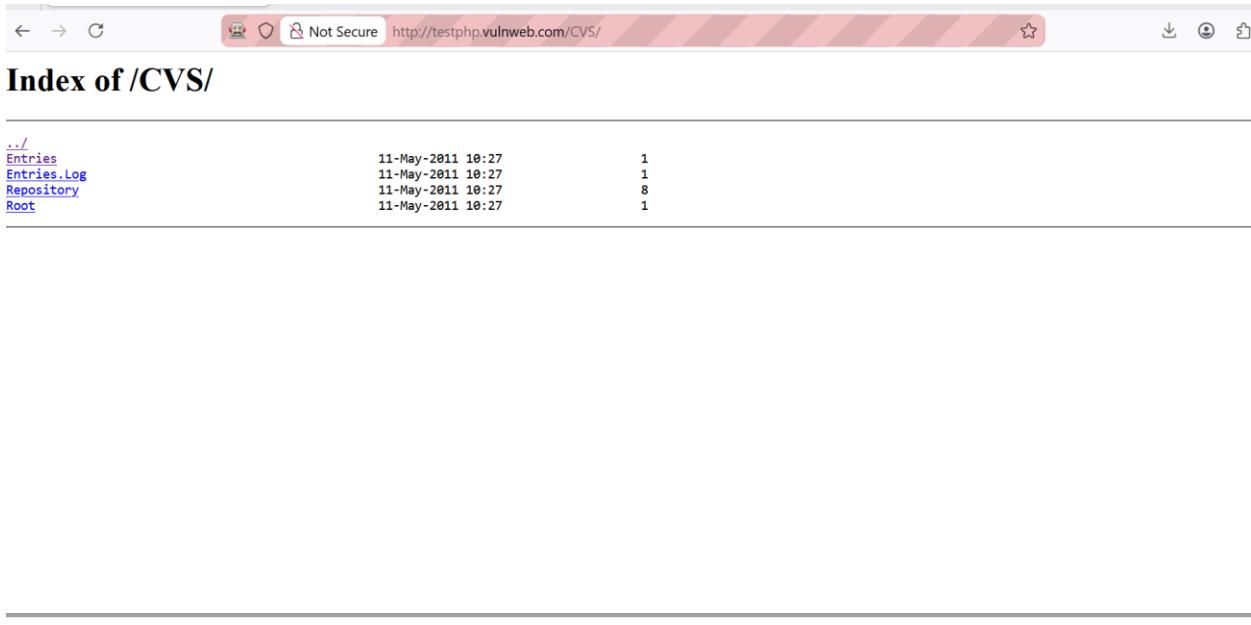
- Attack surface mapping
- Version disclosure
- Social engineering
- Potential source code exposure

### Remediation

- Block /CVS/ access
- Remove VCS directories
- Disable directory listing

## Evidence:

Figure 4: Exposed CVS directory



## FINDING 4: MISSING ANTI-CSRF TOKENS

**Risk Level:** MEDIUM

**Location:** Multiple forms

**CWE:** CWE-352

### Description

Forms lack CSRF protection, allowing forged requests.

### Business Impact

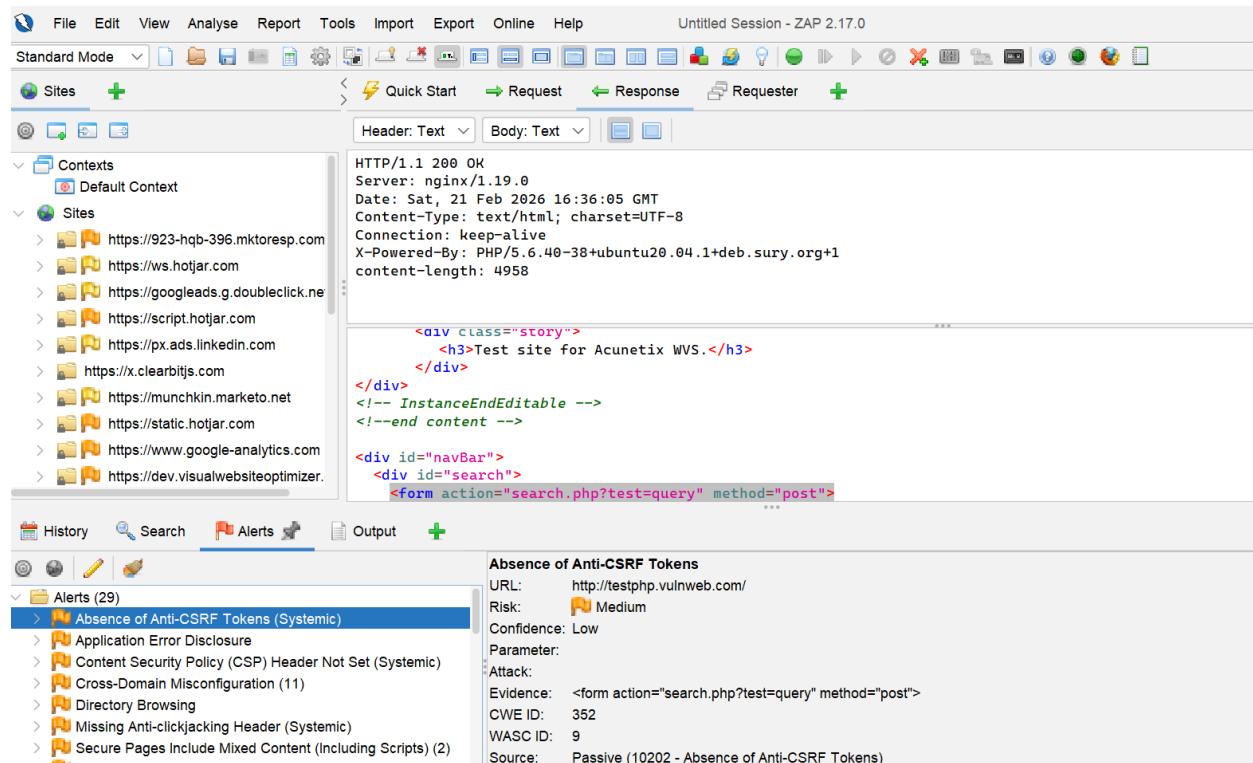
- Unauthorized actions
- Privilege escalation
- Chained attacks

### Remediation

- Implement CSRF tokens
- SameSite cookies
- Framework protections
- Verify Origin headers

## Evidence:

Figure 5: OWASP ZAP alert



## FINDING 5: CONTENT SECURITY POLICY (CSP) HEADER NOT SET

**Risk level:** Medium

**OWASP Category:** A05 – Security Misconfiguration

### Description:

The application does not define a Content Security Policy (CSP) header, allowing the browser to load resources from untrusted sources.

### Business Impact:

This increases the risk of Cross-Site Scripting (XSS) and malicious content injection.

### Recommendation:

Configure a strict Content Security Policy to restrict allowed sources for scripts, styles, and other resources.

## Evidence:

HTTP response headers were observed to be missing the Content-Security-Policy header.

Figure 6: OWASP ZAP Alert

The screenshot shows the OWASP ZAP interface. In the top navigation bar, 'File', 'Edit', 'View', 'Analyse', 'Report', 'Tools', 'Import', 'Export', 'Online', and 'Help' are visible. The title bar says 'Untitled Session - ZAP 2.17.0'. Below the menu is a toolbar with icons for 'Quick Start', 'Request', 'Response', 'Requester', and others. A sidebar on the left lists 'Contexts' and 'Sites', with 'Default Context' and 'Sites' expanded, showing various URLs like https://923-hqb-396.mktoresp.com. The main pane displays a request and response view. The request section shows a standard HTTP response header. The response section shows the raw HTML content of a page from 'testphp.vulnweb.com'. A detailed alert is highlighted in the 'Alerts' section of the sidebar. The alert details are as follows:

Content Security Policy (CSP) Header Not Set	
URL:	http://testphp.vulnweb.com/
Risk:	Medium
Confidence:	High
Parameter:	
Attack:	
Evidence:	
CWE ID:	693
WASC ID:	15
Source:	Passive (10038 - Content Security Policy (CSP) Header Not Set)

## BUSINESS IMPACT

If exploited, these vulnerabilities could lead to:

- **Data Breach:** Exposure of customer personal and financial information
- **Account Takeover:** Attackers gaining administrative access
- **Reputation Damage:** Loss of customer trust and potential regulatory penalties
- **Financial Loss:** Fraudulent transactions or remediation costs

## OVERALL RISK ASSESSMENT

- **Overall Risk Level:** HIGH
- **Confidence:** High
- **Business Impact:** Critical

# PRIORITIZED REMEDIATION ROADMAP

## Immediate (24–48 Hours)

1. Change default credentials
2. Fix SQL injection

## Short-Term (1–2 Weeks)

3. Block CVS directory
4. Implement CSRF tokens
5. Configure a strict Content Security Policy

## Long-Term

- Security training
  - Automated scanning
  - Regular assessments
  - Penetration testing
- 

## RECOMMENDATIONS

Immediate actions should include:

- Fixing the SQL injection vulnerability by implementing parameterized queries
  - Changing default admin credentials and restricting access to the admin panel
  - Implementing anti-CSRF tokens across all forms
  - Restricting public access to version control directories
- 

## CONCLUSION

While this application is intentionally vulnerable for testing purposes, the findings demonstrate common real-world security gaps that must be addressed in production environments. Addressing these issues will significantly improve the application's security posture.

---

## **FINAL REMARKS**

The vulnerabilities identified mirror real-world security issues found in production systems. Security must be treated as an ongoing process involving testing, secure development practices, and continuous improvement.