Vulnerability Assessment Report

==========

SUMMITED BY:

CYBERNITY

code: CYCYBSIAS12

1. SQL Injection in Login Functionality (login bypass)

Summary: A critical SQL injection vulnerability has been identified in the login functionality of the web application hosted at http://testphp.vulnweb.com/login.php. This vulnerability allows an attacker to bypass authentication by injecting a malicious payload in the username and password fields.
Severity: Critical
Confidence: High
Host: http://testphp.vulnweb.com
Path: /login.php

1. Issue in Detail:

The identified vulnerability is a classic SQL injection in the login functionality of the web application. An attacker can manipulate the input fields by injecting the payload ' or 1=1 -- in both the username and password fields. This payload exploits the SQL query logic, allowing unauthorized access to the system.

2. Issue Background:

SQL injection is a common web application security vulnerability that occurs when an attacker can manipulate an application's SQL query by injecting malicious SQL code. In this case, the vulnerability arises from insufficient input validation in the login.php script, enabling an attacker to craft a payload that alters the logic of the SQL query, leading to unauthorized access.

3. Issue Remediation:

To mitigate this vulnerability, the following actions are recommended:

- Implement proper input validation and parameterized queries to prevent SQL injection attacks.
- Regularly update and patch the web application and associated components to address any security vulnerabilities.
- Conduct thorough code reviews and security audits to identify and remediate potential security flaws.

4. References:

- OWASP: SQL Injection Prevention Cheat Sheet (https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html)
- CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection') (https://cwe.mitre.org/data/definitions/89.html)

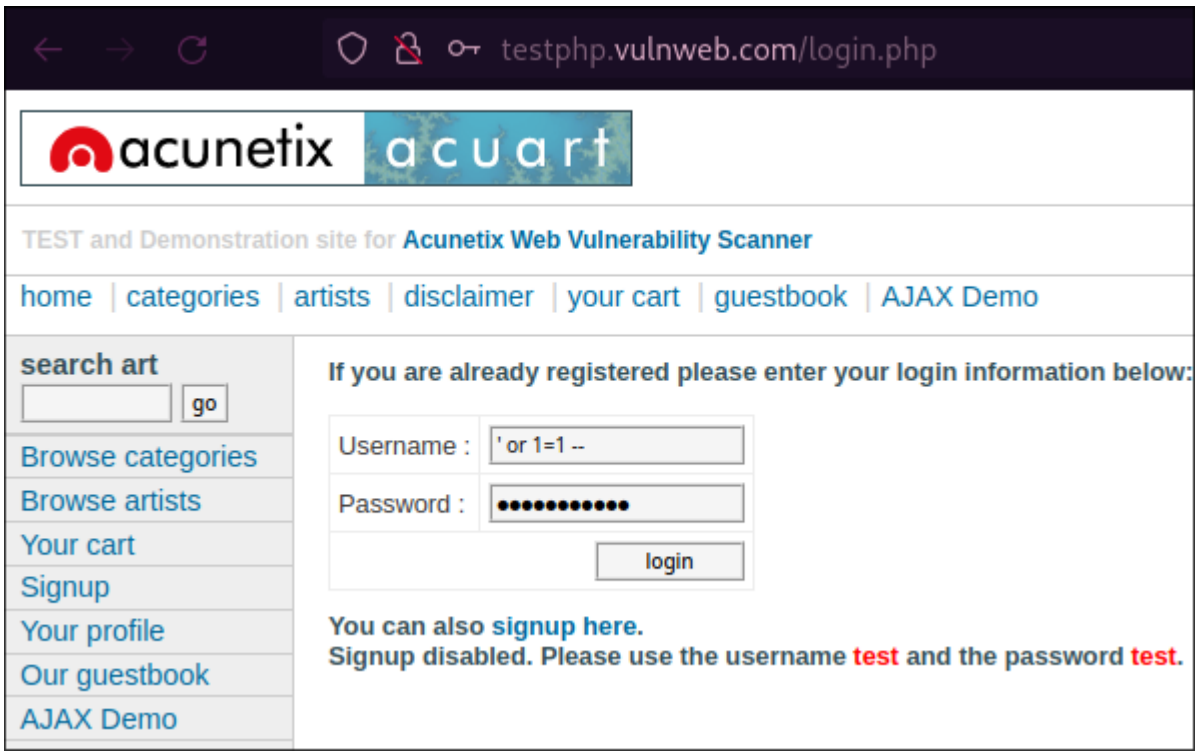
5. POC (Proof of Concept) of the Vulnerability:

The following payload demonstrates the vulnerability:

```
Username: ' or 1=1 --
Password: ' or 1=1 --
```

Entering this payload into the login fields successfully bypasses authentication, granting unauthorized access to the application.

Note: This proof of concept is provided for educational purposes only. It is crucial to address and remediate the vulnerability promptly to secure the application against potential exploitation.


testphp.vulnweb.com/login.php
acunetix
acuart
TEST and Demonstration site for Acunetix Web Vulnerability Scanner
home | categories | artists | disclaimer | your cart | guestbook | AJAX Demo
search art
go
Browse categories
Browse artists
Your cart
Signup
Your profile
Our guestbook
AJAX Demo
If you are already registered please enter your login information below:
Username : ' or 1=1 --
Password : ••••••••••••
login
You can also signup here.
Signup disabled. Please use the username test and the password test.


testphp.vulnweb.com/userinfo.php
acunetix
acuart
TEST and Demonstration site for Acunetix Web Vulnerability Scanner
home | categories | artists | disclaimer | your cart | guestbook | AJAX Demo
search art
go
Browse categories
Browse artists
Your cart
Signup
Your profile
Our guestbook
AJAX Demo
Links
Security art
PHP scanner
PHP vuln help
Fractal Explorer
John Smith (test)
On this page you can visualize or edit you user information.
Name: John Smith
Credit card number: 1234-5678-2300-9000
E-Mail: convert(int, cast(0x5f21403264696c656d6d
Phone number: 2323345
Address: 21 street
update

2. Critical SQL Injection and Database Dump in Web Application

Summary:
A severe SQL injection vulnerability has been discovered in the web application hosted at http://testphp.vulnweb.com. This vulnerability enables an attacker to bypass authentication and subsequently dump sensitive information from the database using tools like sqlmap.
Severity: Critical
Confidence: High
Host: http://testphp.vulnweb.com
Path: /listproducts.php?cat=1

1. Issue in Detail:

The identified vulnerability allows an attacker to execute arbitrary SQL queries through crafted input parameters. By utilizing tools like sqlmap with the command `sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -T artists -C aname --dump`, an attacker can extract sensitive data from the database, posing a significant security threat.

2. Issue Background:

SQL injection is a prevalent attack vector, and in this case, it allows an attacker to manipulate the application's SQL queries, leading to unauthorized access and extraction of sensitive data. The ability to dump database contents exacerbates the severity of the vulnerability.

3. Issue Remediation:

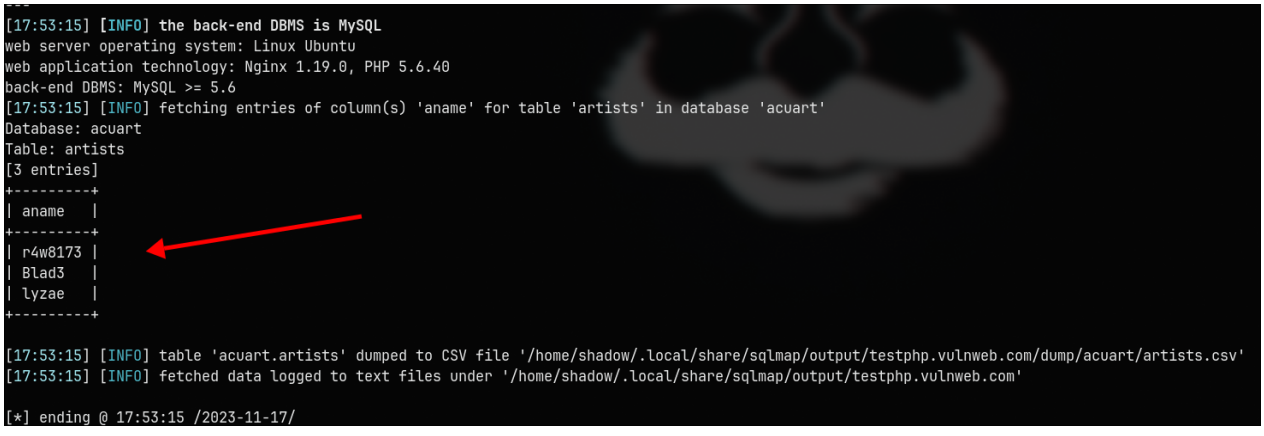
To address this critical vulnerability:

- Immediately apply proper input validation and parameterized queries to mitigate SQL injection risks.
- Conduct a thorough security audit to identify and remediate vulnerabilities in the application code.
- Regularly update and patch the web application and associated components to maintain a secure environment.

4. References:

- OWASP: SQL Injection Prevention Cheat Sheet
- CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')

5. POC (Proof of Concept) of the Vulnerability:



The provided sqlmap command `sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -T artists -C`

`aname --dump` demonstrates the ability to extract data from the database. This emphasizes the urgency of addressing the SQL injection vulnerability to prevent unauthorized access and data leakage.

3. Cross-Site Scripting (XSS) in Web Application

Summary:
A critical Cross-Site Scripting (XSS) vulnerability has been identified in the web application hosted at http://testphp.vulnweb.com/guestbook.php. This vulnerability allows an attacker to inject malicious scripts, leading to the execution of arbitrary code in the context of other users' browsers.
Severity: Critical
Confidence: High
Host: http://testphp.vulnweb.com
Path: /guestbook.php

1. Issue in Detail:

The XSS vulnerability enables an attacker to inject a script, such as `<script>alert("hacked")</script>`, into user input fields. When other users view the affected pages, the injected script executes in their browsers, potentially leading to unauthorized access, session theft, or other malicious activities.

2. Issue Background:

Cross-Site Scripting (XSS) is a common web application vulnerability that arises when user input is not properly validated or sanitized. In this case, the lack of input validation in the guestbook.php script allows an attacker to inject and execute malicious scripts in the context of other users' browsers.

3. Issue Remediation:

To address this critical XSS vulnerability:

- Implement input validation and output encoding to ensure user input is sanitized.
- Apply context-aware output encoding to prevent script execution in user-generated content.
- Regularly perform security reviews and testing to identify and remediate potential vulnerabilities.

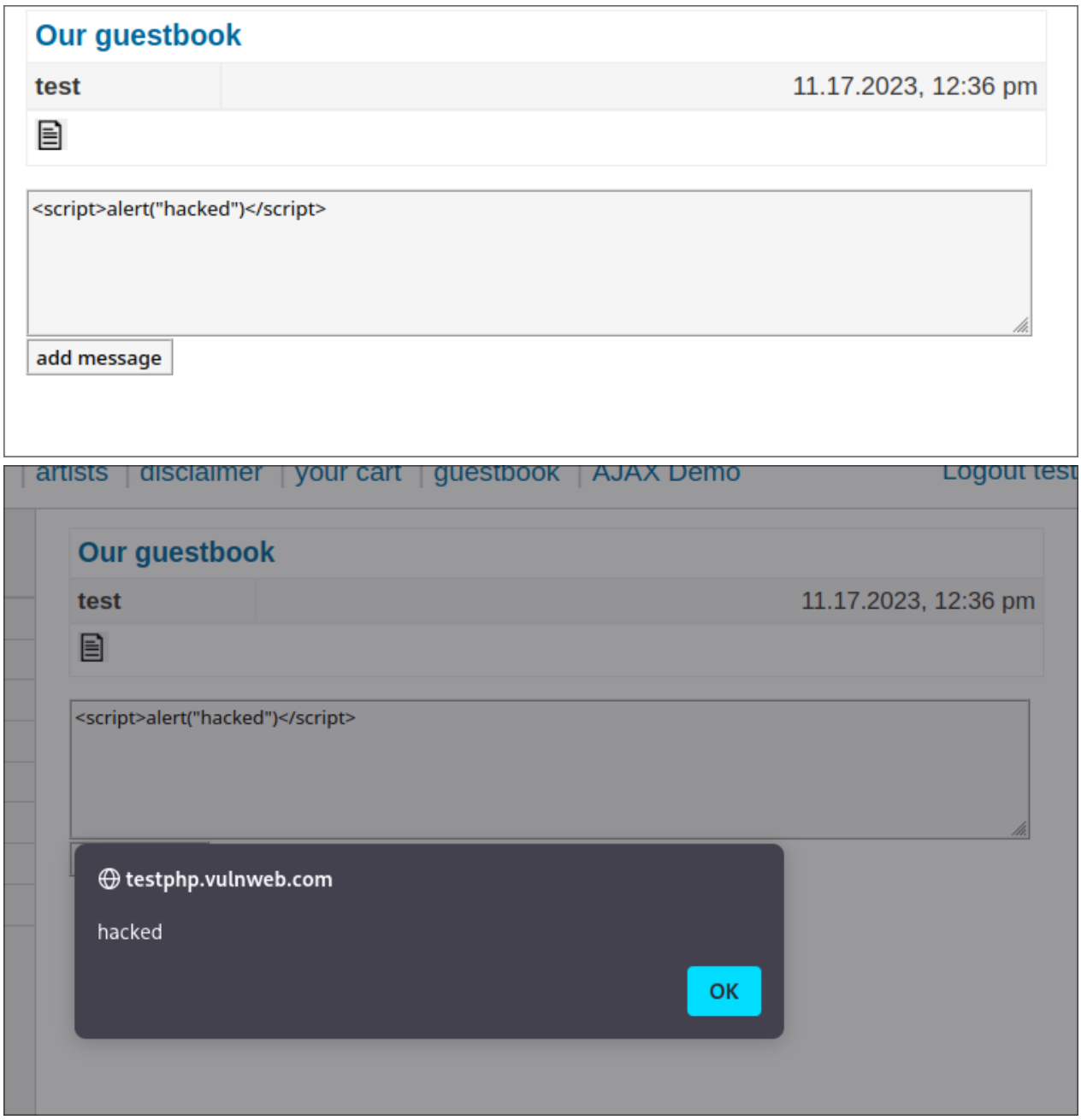
4. References:

- OWASP: Cross-Site Scripting (XSS)
- CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')

5. POC (Proof of Concept) of the Vulnerability:

The following payload demonstrates the XSS vulnerability:

- Payload: `<script>alert("hacked")</script>`



When this payload is injected into user input fields on the guestbook.php page, it triggers a JavaScript alert, indicating successful exploitation of the vulnerability.

Note: This proof of concept is provided for educational purposes only. Immediate action is recommended to remediate the vulnerability and secure the application against potential exploitation.