COMPREHENSIVE VAPT ANALYSIS ON NETWORK INFRASTRUCTURE

The domain of the Project

Cybersecurity - Web Application Security

Under the guidance of

Mr. Nishchay Gaba (Penetration Tester)

By

Mr. Sanga Venkata Balaji (B.Tech)

Period of the project

NOVEMBER 2024 to JULY 2025

SURE TRUST
PUTTAPARTHI, ANDHRA PRADESH

DECLARATION

The project titled **" *WEB APPLICATION PENTESTING* "** has been mentored by **Mr. Nishchay Gaba** and organized by SURE Trust from November 2024 to june 2025**.** This initiative aims to benefit educated unemployed rural youth by providing hands-on experience in industry-relevant projects, thereby enhancing employability.

I, **Mr. SANGA VENKATA BALAJI,** hereby declare that
I have solely worked on this project under the guidance of my mentor. This project has significantly enhanced my practical knowledge and skills in the domain.

Name **Signature**

Mr.SANGA VENKATA BALAJI

Mr. Nishchay Gaba

Mentor **Signature**

Seal & Signature

Prof.Radhakumari
Executive Director &
Founder

Table of Contents

1. Disclaimer:

This Web Application Penetration Testing Report was prepared as part of an internship project at Sure Trust and is based on the assessment of the provided web applications. The findings, analysis, and recommendations are derived from the testing conducted within the approved scope. While all efforts have been made to identify vulnerabilities accurately, this report does not guarantee the complete security of the tested systems. The author and Sure Trust are not responsible for any misuse, unauthorized actions, or unintended consequences arising from this report. Use this report responsibly and implement necessary security measures as appropriate.

Contact Info :

NAME	TITLE	CONTACT INFO
SANGA VENKATA BALAJI	Intern	Sangabalaji117@gmail.com

2. Executive Summary :

This report presents the findings of a web application penetration test conducted on the target system. The objective was to identify vulnerabilities that could be exploited to compromise the confidentiality, integrity, and availability of the application and its data.

A total of 23 vulnerabilities were discovered, including several critical issues such as SQL Injection, Remote Code Execution, Command Injection, and Broken Authentication mechanisms. These vulnerabilities, if left unaddressed, could allow attackers to gain unauthorized access, execute arbitrary commands on the server, retrieve sensitive data, and manipulate application behavior.

The identified issues vary in severity from Low to Critical, with the majority falling under the High-risk category. Immediate remediation is strongly recommended for all critical and high-severity findings.

The application also lacks secure communication mechanisms (HTTPS), exposes outdated and unsupported technologies, and permits unsafe HTTP methods — further increasing the attack surface.

This report outlines each finding in detail, along with reproduction steps, technical impact, CVSS scores, and recommended mitigations to assist the development team in securing the application.

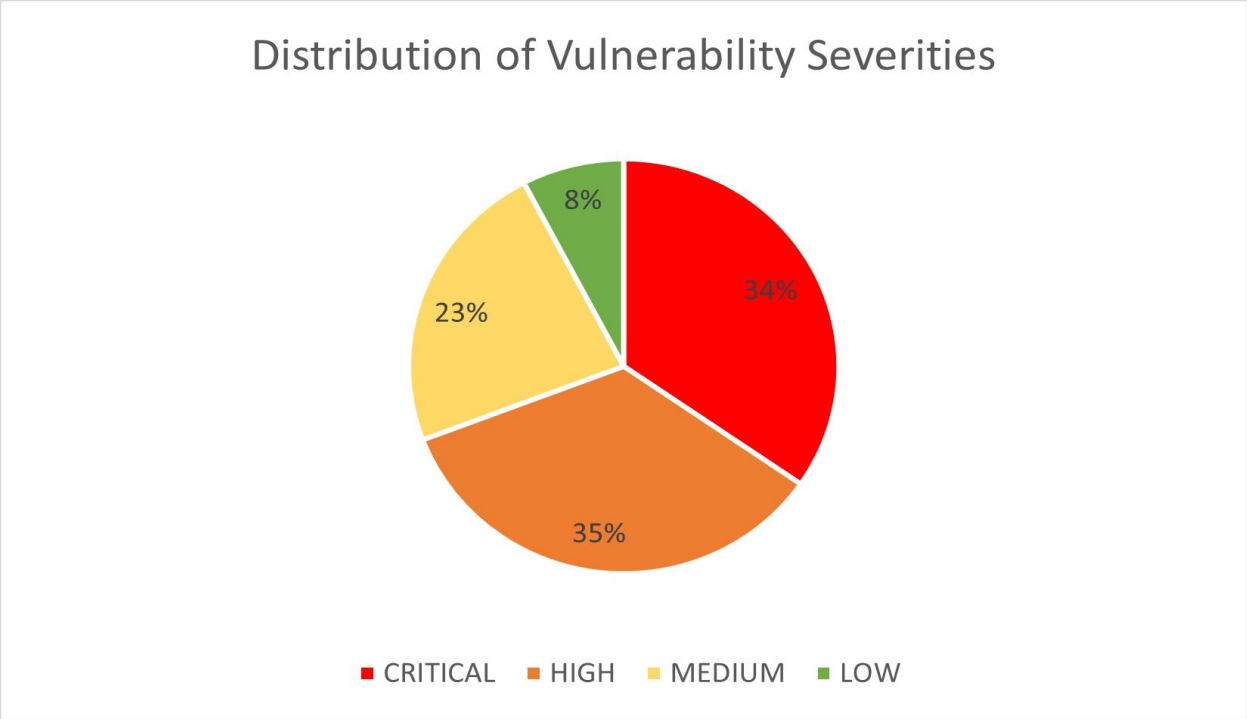


Fig.Pie Chart

3. Scope of Engagement

This penetration testing engagement was conducted to evaluate the security posture of the target web application. The assessment aimed to identify vulnerabilities that could be exploited by malicious actors to compromise the application, its users, and associated data.

The scope of this engagement included:

- Target: http://testphp.vulnweb.com and bWAPP/
- Environment:

The assessment was performed in a controlled lab setup using intentionally vulnerable applications for educational and testing purposes (e.g., bWAPP, DVWA, testphp.vulnweb.com).

- Testing Methodology:
  Manual testing was combined with automated tools to detect security vulnerabilities across various components such as authentication, session management, input validation, file handling, and server configuration
- **Timeframe:**
  Testing was performed during the month of **June and July 2025**.

4. METHODOLOGY :

Methodology

The penetration test was performed using both manual and automated methods based on the **OWASP Web Security Testing Guide v4** and the **OWASP Top 10 2021**. The testing involved:

- Information gathering using reconnaissance tools
- Authentication and session management testing
- Input validation tests including SQLi, XSS, and LFI
- File upload and remote code execution tests
- Misconfiguration analysis (e.g., HTTP methods, outdated software)

Tools used include: Burp Suite, SQLMap, Nmap, Wireshark, and Firefox Developer Tools.

5. TARGET INFORMATION :

Target 1: testphp.vulnweb.com

- Hosted by Acunetix for public testing
- Simulates a real-world e-commerce web application
- Contains common web vulnerabilities (SQLi, XSS, LFI, etc.)
- Accessible over: http://testphp.vulnweb.com

Target 2: bWAPP (Bee-box) & DVWA

- Intentionally vulnerable web application for training purposes

- Installed and hosted in a local virtual machine (http://192.168.102.132)
- Provides over 100 vulnerable scenarios for testing web security
- Contains challenges related to authentication, session management, injection, and misconfigurations

Testing Environment:

- OS: Kali Linux (running on VMware Workstation)
- Browser: Firefox (Developer Edition)
- Network: Host-only or NAT (local lab environment)
- Tools: Burp Suite, SQLMap, Nikto, Nmap, Wireshark, etc.

6. RISK RATING CRITERIA :

SEVERITY	CVSS V3 SCORE RANGE	DEFINITIONS
Critical	9.0–10.0	Exploitation is straightforward and usually results in complete system compromise. Patch as soon as possible.
High	7.0–8.9	May result in elevated privileges, data leaks, or service disruptions. Apply patches or mitigations promptly.
Medium	4.0–6.9	Can pose security risk but not as severe as higher levels. Implement patches or mitigations in a reasonable time.
Low	0.1-3.9	Exploitation is unlikely or difficult in a practical scenario. Consider patching or applying mitigations if necessary.

5.2 Risk Factors & Likelihood

The risk associated with each identified vulnerability was evaluated based on the following factors:

- **Impact**: The potential effect on the confidentiality, integrity, or availability of the system or its data.

- **Likelihood**: The ease with which an attacker can discover and exploit the vulnerability.
- **Exploitability**: Whether the vulnerability can be exploited remotely, without authentication, or with minimal user interaction.

Likelihood Levels

- **High**: The vulnerability is easily exploitable, often remotely, and typically does not require authentication or special conditions.
- **Medium**: Exploitation is possible but may require valid user credentials, a valid session, or some level of user interaction.
- **Low**: Exploiting the issue is complex and may depend on specific configurations, insider access, or chained vulnerabilities.

Each vulnerability's final **severity level** has been determined using the **CVSS v3.1** scoring system, based on the combined assessment of impact, likelihood, and exploitability.

7. Detailed Vulnerability Analysis :


CRITICAL

7.1 VULNERABILITY : Unrestricted File Upload

- **DESCRIPTION :** Allows malicious files (e.g., web shells) to be uploaded and potentially executed.
- **CVSS SCORE:** 9.1 (Critical) AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:H
- **CWE / OWASP REFERENCE:**
  CWE-79
  OWASP A03:2021 – Injection

- **IMPACT :** Remote Code Execution (RCE), privilege escalation, system compromise.
- **MITIGATION :** Enforce extension/MIME checks, move files outside web root, disable script execution in upload dirs.

- **REFERENCES:**

OWASP Unrestricted File Upload: https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload

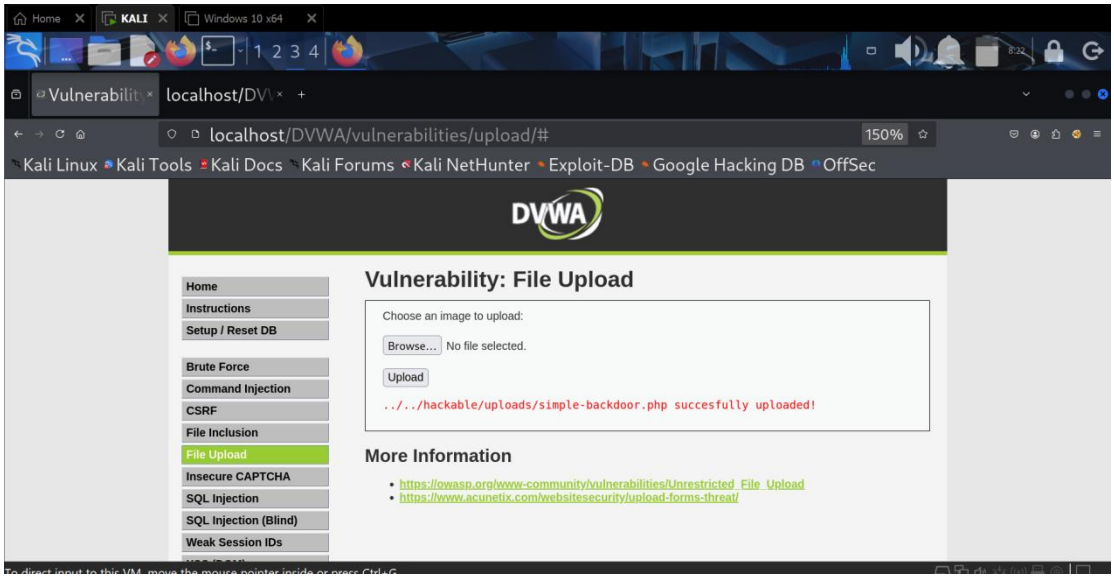
CWE-434: Unrestricted Upload of File with Dangerous Type: https://cwe.mitre.org/data/definitions/434.html

PortSwigger Web Security Academy - File Upload Vulnerabilities: https://portswigger.net/web-security/file-upload

DVWA Official GitHub: https://github.com/digininja/DVWA

• AFFECTED URL'S :
http://192.168.102.129/dvwa/vulnerabilities/fi/?page=include.php

POC :


Home
KALI
Windows 10 x64
localhost/DVWA/vulnerabilities/upload/#
150%
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
Vulnerability: File Upload
Choose an image to upload:
Browse...
No file selected.
Upload
../../hackable/uploads/simple-backdoor.php succesfully uploaded!
More Information
https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload
https://www.acunetix.com/websitesecurity/upload-forms-threat/

7.2 : VULNERABILITY : OS Command Injection

- **DESCRIPTION :** Allows attacker-controlled input to be executed as shell commands on the server
- **CVSS SCORE** :9.8 (Critical 3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H
- **CVE / CWE / OWASP REFERENCE:** CWE-78: Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')
- OWASP Top 10 (2021): A03:2021 - Injection (OS Command Injection is a prime example of an injection vulnerability).
- **IMPACT :** Full server control, file exfiltration or deletion, further lateral movement.

- **MITIGATIONS :** Avoid using system(), exec(); use parameterized functions; sanitize inputs strictly**.**

- **REFERENCES :**

  OWASP Command Injection: https://owasp.org/www-community/attacks/Command_Injection

  CWE-78: Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection'): https://cwe.mitre.org/data/definitions/78.html

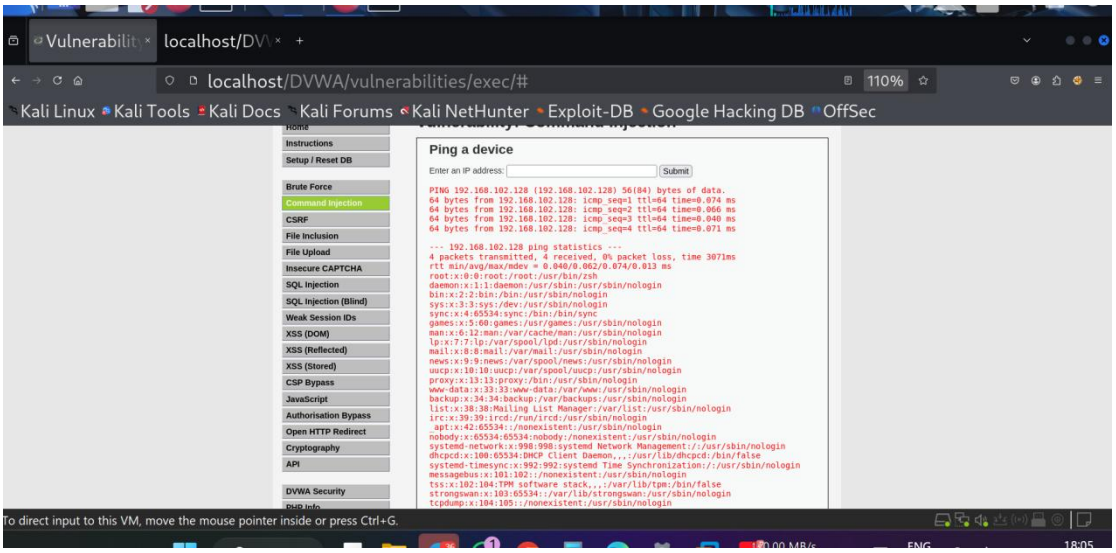
  PortSwigger Web Security Academy - OS command injection: https://portswigger.net/web-security/os-command-injection

  DVWA Official GitHub: https://github.com/digininja/DVWA

  https://www.hackingarticles.in/database-penetration-testing-using-sqlmap-part-1/

- **AFFECTED URL's : http://192.168.102.129/dvwa/vulnerabilities/exec/**

- **POC :**



7.3 VULNERABILITY: Unauthenticated SQL Injection

- **DESCRIPTION:**

  via a login form or search functionality. Unsanitized input is directly concatenated into a SQL query, allowing an attacker to manipulate database logic, extract sensitive data, and potentially gain shell access or escalate privileges.

  In your case, using payloads like:
  ' OR '1'='1 --

- **CVSS SCORE:** 9.8 (Critical) – CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

- **CVE / CWE / OWASP REFERENCE:**

  CWE-89: ImpropeThe Artist VulnHub machine contains an SQL Injection vulnerability in a parameter passed r Neutralization of Special Elements used in an SQL Command ('SQL Injection')
  OWASP Top 10 (2021): A03:2021 – Injection

- **IMPACT:**

- Authentication Bypass
- Credential Dumping (usernames/password hashes)
- Information Disclosure (emails, admin accounts)
- Privilege Escalation (combined with LFI/RCE)
- Full DB compromise (read/write access)
- Shell Upload (via injectable upload logic)

- **MITIGATIONS:**

Use parameterized queries / prepared statements
Apply server-side input validation and escaping
Enforce least privilege for DB accounts
Employ Web Application Firewalls (WAFs)
Disable detailed error messages in production
Use modern frameworks with built-in ORM (e.g., SQLAlchemy, Django ORM)

- **REFERENCES:**

OWASP SQL Injection: https://owasp.org/www-community/attacks/SQL_Injection

CWE-89: https://cwe.mitre.org/data/definitions/89.html

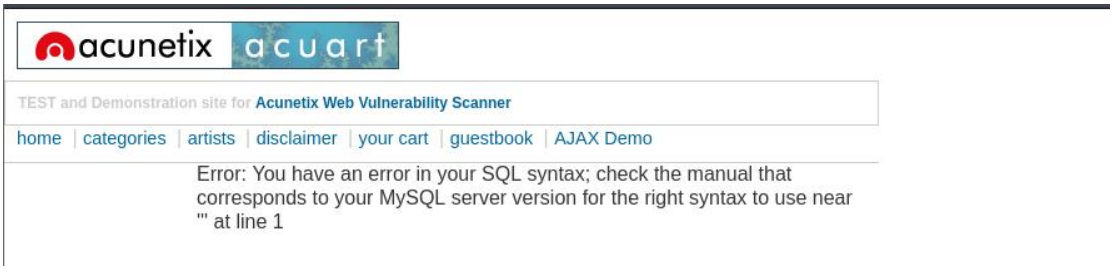
SQLMap: https://sqlmap.org

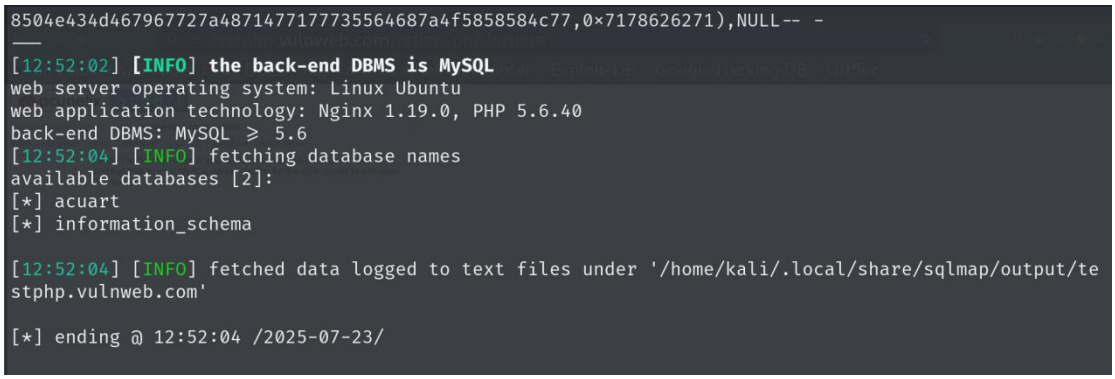
NIST CVSS Calculator: https://www.first.org/cvss/calculator/3.1

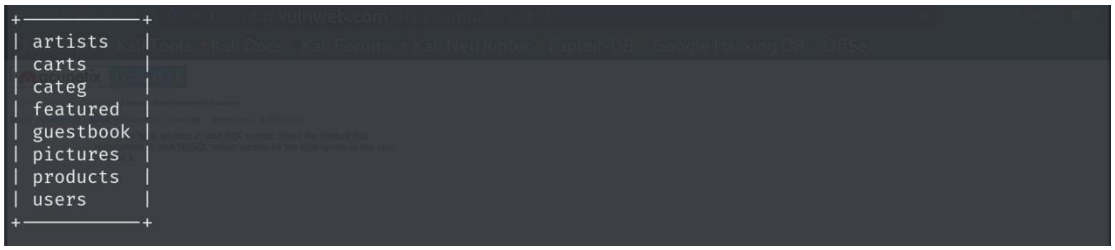
OWASP Top 10 2021: https://owasp.org/www-project-top-ten/

- **AFFECTED URL's :** https://testphp.vulnweb.com/

- **POC:**







7.4 VULNERABILITY: Authentication Bypass via SQL Injection

- **DESCRIPTION:**

  The Artist VulnHub machine has a login page that appears vulnerable to SQL Injection. An attacker can manipulate SQL queries used during the login process to bypass authentication without valid credentials. This indicates broken authentication due to improper validation and insecure coding practices in the login mechanism.

- **CVSS SCORE :**

  Base Score: 9.8 (Critical)Vector:
  CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

- **CVE /CWE / OWASP REFERENCE :**

  • CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')

  • OWASP A01:2021 – Broken Access Control

  • OWASP A03:2021 – Injection

- **IMPACT :**

  • Unauthorized access to restricted areas of the web application.
  • Possible privilege escalation (e.g., access as admin).
  • Exposure of sensitive user data.
  • Total compromise of authentication mechanism.

- **MITIGATION:**

  • Use parameterized queries (prepared statements) to prevent SQL Injection.
  • Implement input validation and escaping of special characters.
  • Avoid displaying detailed error messages to users.
  • Implement multi-factor authentication.
  • Regularly audit authentication and authorization mechanisms.

- **REFERENCES:**

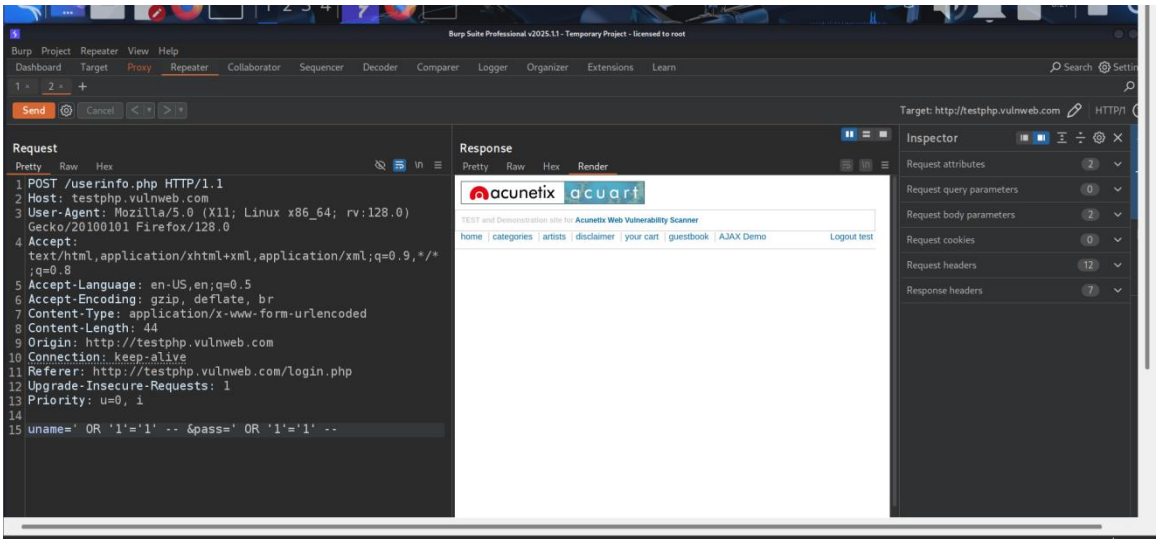
  • CWE-89: https://cwe.mitre.org/data/definitions/89.html

  • OWASP SQLi: https://owasp.org/www-community/attacks/SQL_Injection

  • OWASP Broken Authentication: https://owasp.org/Top10/A07_2021-Identification_and_Authentication_Failure

- **AFFECTED URL's:** https://testphp.vulnweb.com/

- **POC**



7.5 VULNERABILITY : Credential Leakage via Directory Listing

- **DESCRIPTION:**

  The web server at testphp.vulnweb.com allows directory listing for the /pictures/ path. This misconfiguration exposes sensitive files, including credentials.txt, which contains hardcoded login details (username and password). Accessing this file does not require authentication, making it easy for attackers to harvest credentials and attempt unauthorized access.

- **CVSS SCORE**

  CVSS v3.1 Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:L
  Base Score: 9.1 (Critical)

- **CVE / CWE/OWASP REFERENCE :**

  - CWE-200: Exposure of Sensitive Information
  - CWE-532: Insertion of Sensitive Information into Log File
  - CWE-538: Insertion of Sensitive Information into Web Page

- OWASP A01:2021 – Broken Access Control
- OWASP A05:2021 – Security Misconfiguration

- **IMPACT**

- Credentials leakage (username and password visible in plaintext)
- Risk of unauthorized access to login-protected parts of the application
- Increased attack surface via credential stuffing, brute-force attacks, and privilege escalation
- Violation of data handling policies and potential compliance breaches (e.g., GDPR, HIPAA)

- **MITIGATIONS**

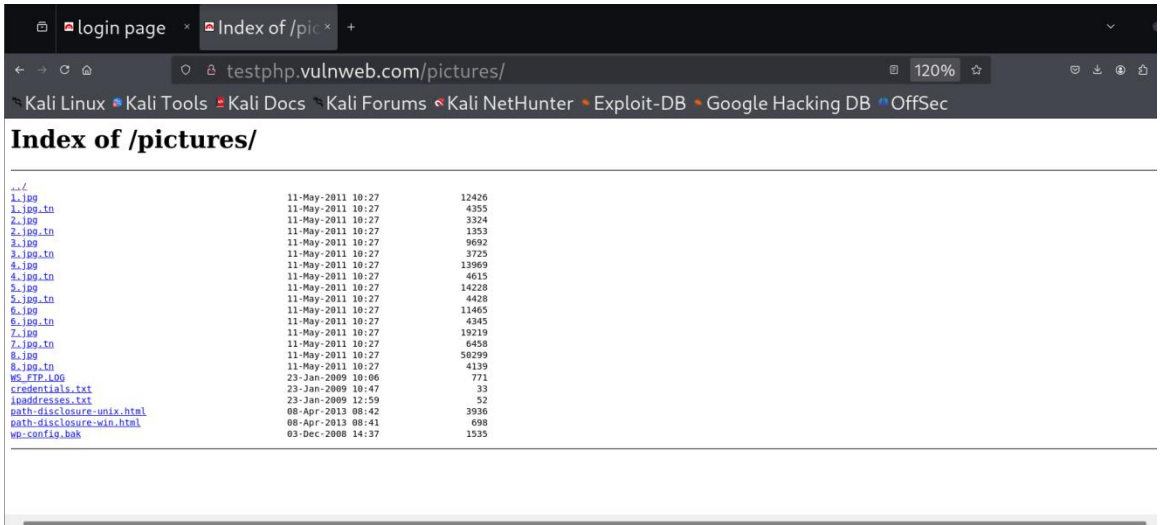
- Disable directory listing on web servers (Apache: Options -Indexes, Nginx: autoindex off;)
- Remove sensitive files like credentials.txt from the web root
- Store secrets in environment variables or secure secrets management tools
- Audit and monitor all files in public directories for accidental exposure
- Implement access control mechanisms to restrict unauthorized access
- Rotate exposed credentials and enforce MFA wherever possible

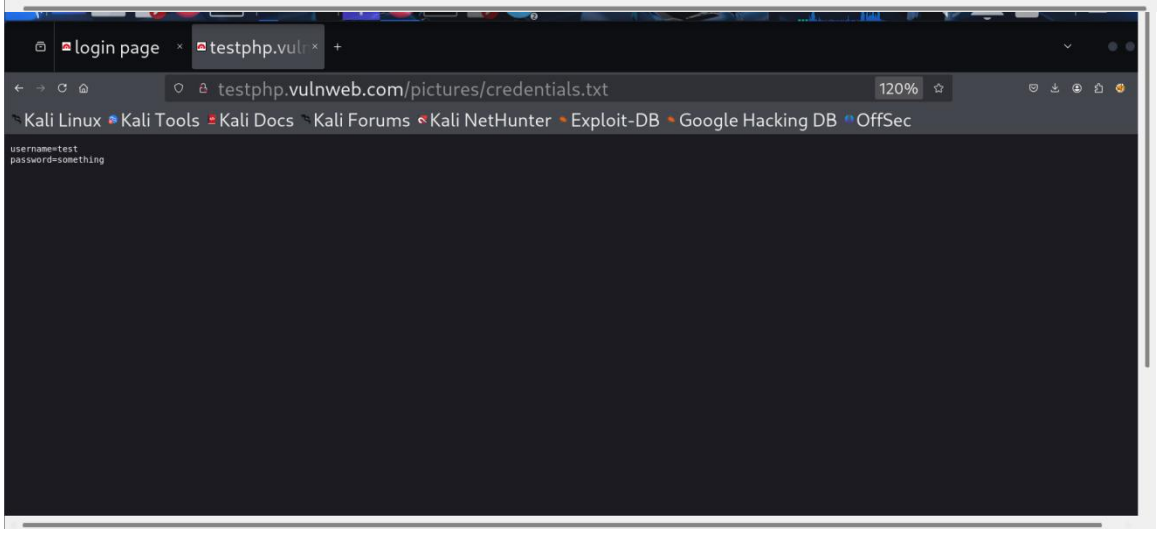
- **REFERENCES**

1. https://cwe.mitre.org/data/definitions/200.html
2. https://cwe.mitre.org/data/definitions/532.html
3. https://cwe.mitre.org/data/definitions/538.html
4. https://owasp.org/Top10/A01_2021-Broken_Access_Control/
5. https://owasp.org/Top10/A05_2021-Security_Misco

- **AFFECTED URL's :** https://testphp.vulnweb.com/

- **POC:**


login page
Index of /pic
testphp.vulnweb.com/pictures/
120%
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
Index of /pictures/
../
1.jpg 11-May-2011 10:27 12426
1.jpg.tn 11-May-2011 10:27 4355
2.jpg 11-May-2011 10:27 3324
2.jpg.tn 11-May-2011 10:27 1353
3.jpg 11-May-2011 10:27 9692
3.jpg.tn 11-May-2011 10:27 3725
4.jpg 11-May-2011 10:27 13969
4.jpg.tn 11-May-2011 10:27 4615
5.jpg 11-May-2011 10:27 14228
5.jpg.tn 11-May-2011 10:27 4428
6.jpg 11-May-2011 10:27 11465
6.jpg.tn 11-May-2011 10:27 4345
7.jpg 11-May-2011 10:27 19219
7.jpg.tn 11-May-2011 10:27 6458
8.jpg 11-May-2011 10:27 50299
8.jpg.tn 11-May-2011 10:27 4139
WS_FTP.LOG 23-Jan-2009 10:06 771
credentials.txt 23-Jan-2009 10:47 33
ipaddresses.txt 23-Jan-2009 12:59 52
path-disclosure-unix.html 08-Apr-2013 08:42 3936
path-disclosure-win.html 08-Apr-2013 08:41 698
wp-config.bak 03-Dec-2008 14:37 1535


login page
testphp.vul
testphp.vulnweb.com/pictures/credentials.txt
120%
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
username=test
password=something

7.6 VULNERABILITY : Local File Inclusion (LFI) via User-Supplied Parameter:

- **DESCRIPTION**: Allows access to internal files using relative paths via vulnerable includes.
- **CVSS SCORE :**CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:L (9.6)
- **CVE / CWE / OWASP REFERENCE:** CWE-98, OWASP A03:2021 – Injection

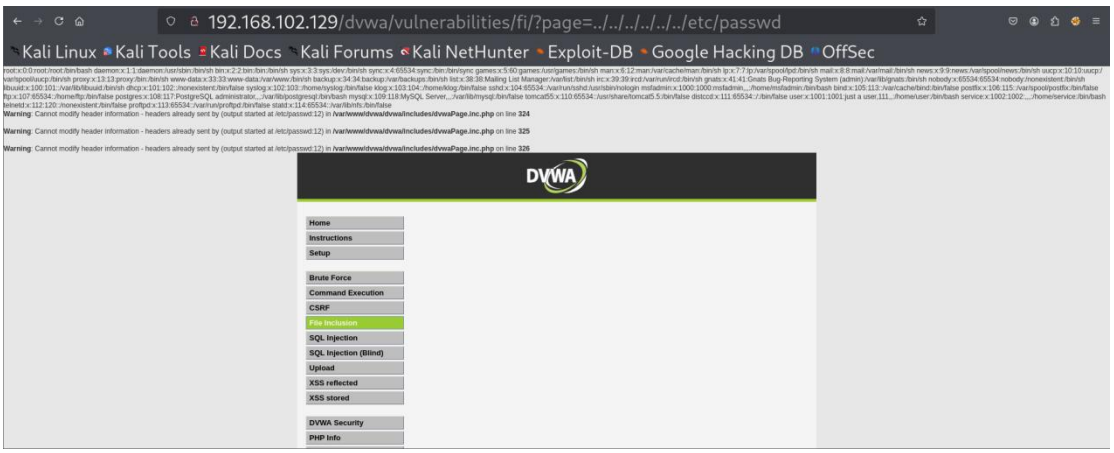
  Impact: Disclosure of sensitive files, escalation to RCE (via log poisoning, upload).

  Mitigation: Sanitize file paths, disable dangerous PHP options, apply strict access controls.

- **MITIGATIONS**: Sanitize file paths, disable dangerous PHP options, apply strict access controls.

- **REFERENCES :CWE-98**
  https://cwe.mitre.org/data/definitions/98.html
  - CWE-22: https://cwe.mitre.org/data/definitions/22.html
  - OWASP LFI Guide: https://owasp.org/wwwcommunity/attacks/Path_Traversal

- **AFFECTED URL's:**
  http://192.168.102.129/dvwa/vulnerabilities/fi/?page=include.php
- **POC :**



7.7 VULNERABILITY : Sensitive File Exposure via Misconfigured Admin Path

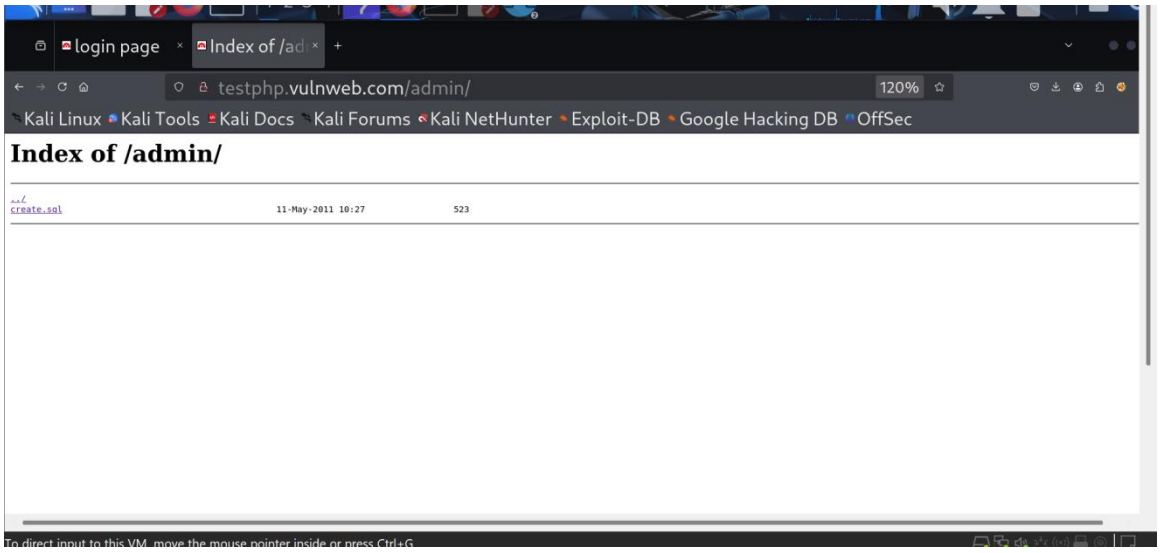
- **DESCRIPTION :** A publicly accessible directory (/admin/) exposed a sensitive create.sql file containing the full database schema. This SQL file was downloadable without authentication due to misconfigured web server settings (directory listing enabled). This allows attackers to gain insight into the internal database structure, which can aid further exploitation such as SQL injection, authentication bypass, or privilege escalation.

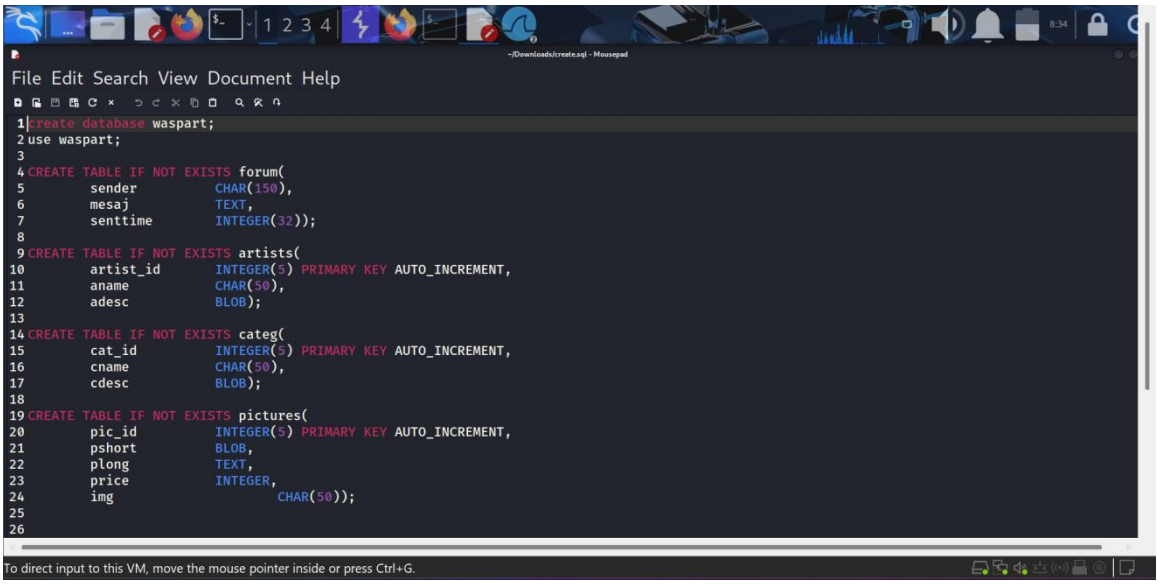
- **CVSS Score:** CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:L (9.1)
- **CVE / CWE / OWASP REFERENCE :**
  - CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
  - CWE-530: Exposure of Backup File
  - OWASP A01:2021 – Broken Access Control
- **MITIGATIONS :**
  - Remove all development and SQL script files from public directories
  - Disable directory listing and apply proper access control on admin paths
  - Configure web server to deny access to sensitive file types
- **REFERENCES :**
  - CWE-538: https://cwe.mitre.org/data/definitions/538.html

- CWE-552: https://cwe.mitre.org/data/definitions/552.html - OWASP A05 – Security Misconfiguration: https://owasp.org/Top10/A05_2021-Security_Misconfiguration/

- **AFFECTED URL'S :** http://testphp.vulnweb.com/admin/

- **POC :**

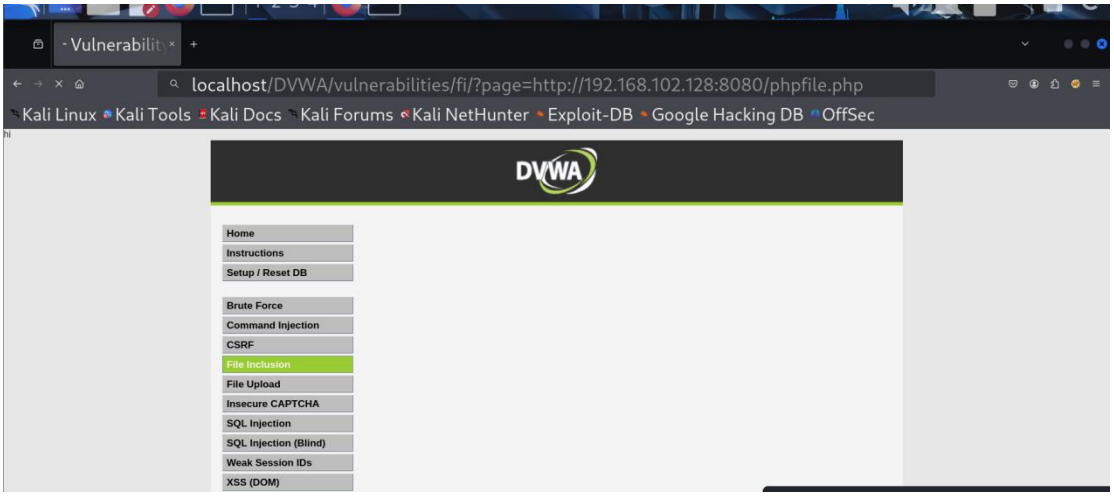


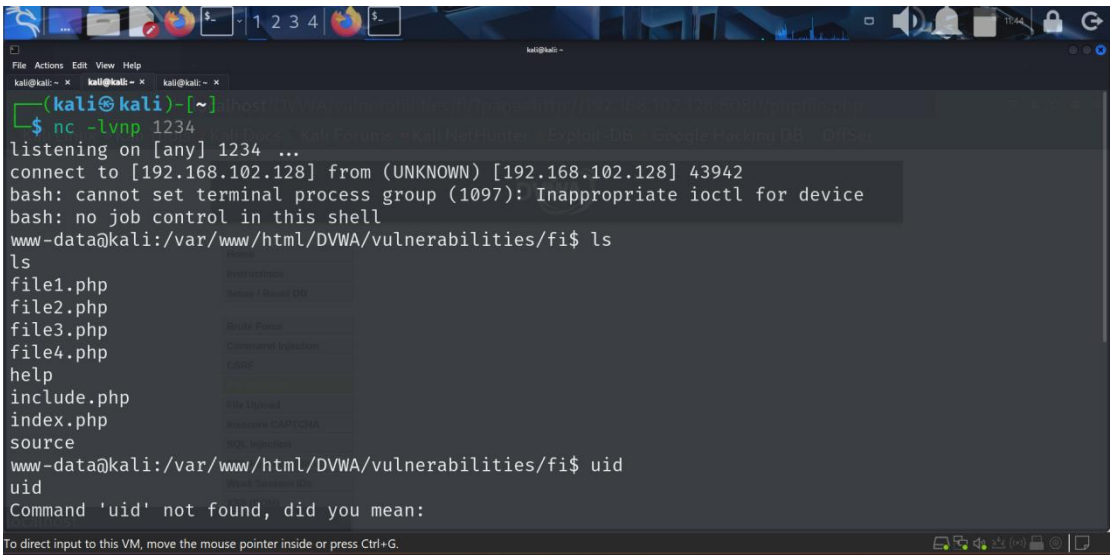


7.8 VULNERABILITY : Remote File Inclusion (RFI) with Remote Code Execution

- **DESCRIPTION :** Attacker includes external malicious code via unsanitized user input in file paths.
- **CVSS SCORE:**
  CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H 9.8
- **CVE / CWE / OWASP REFERENCE :**
  CVE/CWE/OWASP: CWE-98, OWASP A03:2021 – Injection
- **IMPACT :** : Remote Code Execution, remote malware loading.
- **MITIGATIONS :** Disable allow_url_include, validate/whitelist paths, use static includes only.
- **AFFECTED URL's :**
  http://192.168.102.129/dvwa/vulnerabilities/fi/?page=include.php

- **POC :**





I set up a Python server on Kali Linux. Then, I triggered the php-reverse-shell script to establish a reverse shell connection. To listen for the incoming connection, I used Netcat

7.9 VULNERABILITY Credential Exposure via Source Code Disclosure

- **DESCRIPTION:**

  The login page of the bWAPP application contains hardcoded login credentials (e.g., bee / bug) in its HTML source code. This is a serious security flaw, as it allows anyone with basic browser inspection skills to retrieve valid login credentials. This highlights poor credential management and is a critical example of broken authentication.

- **CVSS v3.1 SCORE** : 9.1 (Critical)
  CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N

- **CVE / CWE / OWASP REFERENCE:**

  CWE-798: Use of Hard-coded Credentials

  OWASP A01:2021 – Broken Access Control

  OWASP A02:2021 – Cryptographic Failures (if credentials are not securely stored)

- **IMPACT:** Unauthorized access to the application

  Privilege escalation if admin credentials are exposed

  Full compromise of user sessions and data

- **MITIGATIONS:**

Never hardcode credentials in frontend or client-side code.

Use a secure server-side authentication mechanism.

Enforce proper access controls and session management.

Remove debug/testing comments before pushing to production.

Conduct regular code reviews and static code analysis.

- **REFERENCES:**

OWASP Top 10 – A07:2021 – Identification and Authentication Failures
https://owasp.org/Top10/A07_2021-Identification_and_Authentication_Failures/

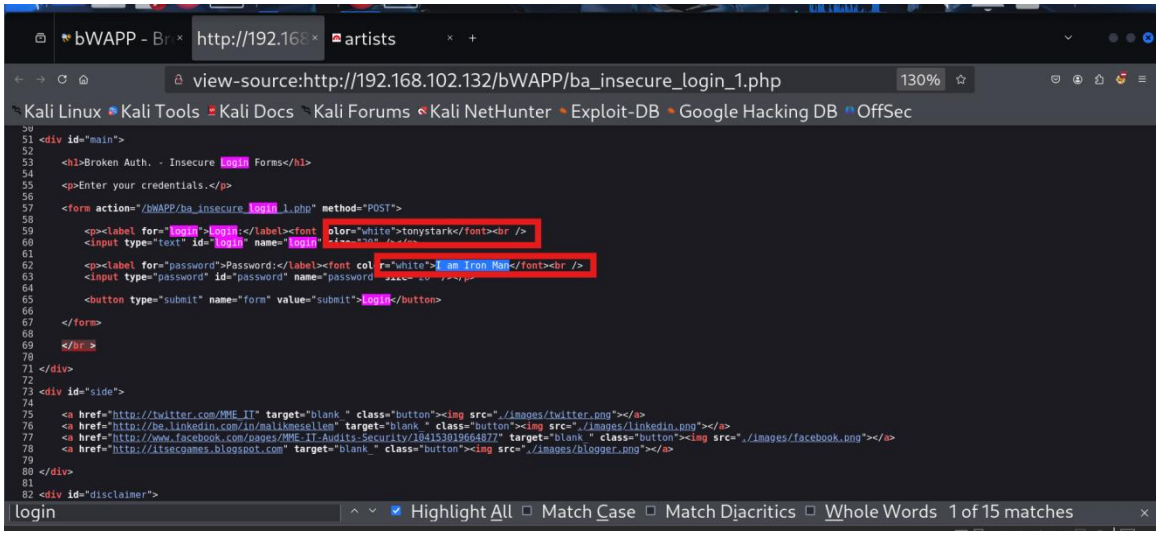
CWE-798: Use of Hard-coded Credentials
https://cwe.mitre.org/data/definitions/798.html

CWE-522: Insufficiently Protected Credentials
https://cwe.mitre.org/data/definitions/522.html

- **AFFECTED URL:**

http://<target-ip>/bWAPP/login.php

- **POC :**




HIGH

7.10 VULNERABILITY : Hardcoded or Exposed Credentials in Client-Side Code

- **DESCRIPTION :**

The login page at http://testphp.vulnweb.com/login.php displays hardcoded credentials openly on the web interface (Username: test, Password: test). This is a serious security misconfiguration and a form of Information Disclosure that can allow unauthorized users to gain access to the application.

- **CVSS SCORE (v3.1):**

Base Score: 7.5 (High)

Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N

- **CVE / CWE / OWASP:**

CWE-798: Use of Hard-coded Credentials OWASP Top 10 (2021): A05:2021 – Security Misconfiguration

- **IMPACT:**

Unauthorized access to user accounts or administrative panels.
Potential pivoting point for deeper application exploitation.
Sensitive data leakage (user information, database contents, etc.).
Undermines authentication mechanism and session integrity.

- **MITIGATIONS:**

Never expose credentials in source code, comments, or front-end interfaces.
Enforce secure authentication and disable default/demo accounts in production.
Implement role-based access control (RBAC) and logging for all login attempts.
Use environment variables or secure vaults for credential management in dev/test environments.
Regularly audit web applications for exposed data and misconfigurations.

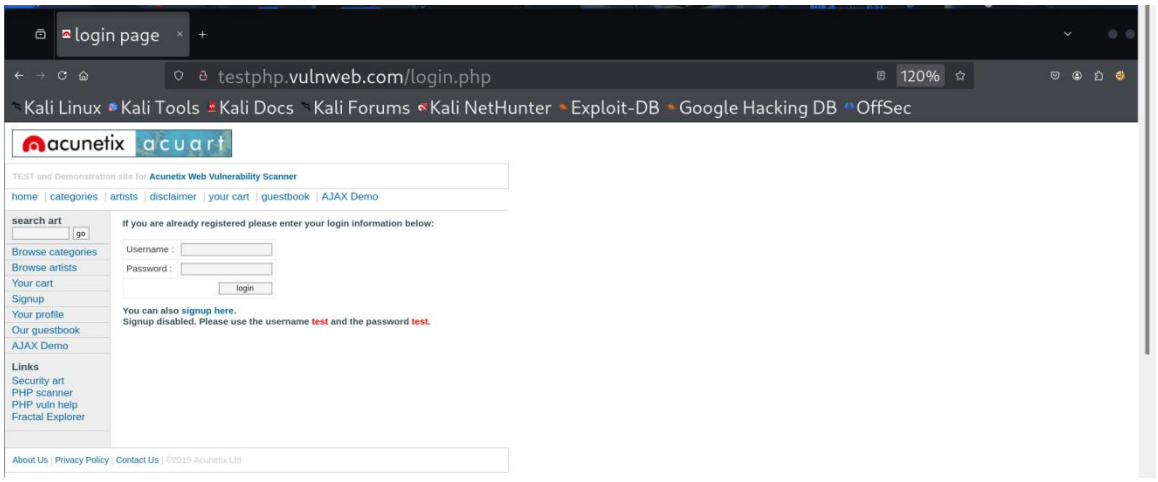
- **REFERENCES**:

CWE-798: https://cwe.mitre.org/data/definitions/798.html

OWASP A05 – Security Misconfiguration: https://owasp.org/Top10/A05_2021-Security_Misconfiguration/

NIST NVD: https://nvd.nist.gov/vuln/detail/CWE-798

- **AFFECTED URL'S:**https://testphp.vulnweb.com/
- **POC :**



7.11 VULNERABILITY : Insecure Business Logic – Price Manipulation

- **DESCRIPTION :** In this vulnerability, an attacker modifies the price of a product while adding it to the cart by intercepting and manipulating HTTP requests.
  The web application trusts client-side inputs for sensitive data like price, allowing attackers to buy products at significantly lower price
- **CVSS SCORE**: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:H/A:N (8.2)
- **CVE / CWE / OWASP REFERENCE:**
  CWE-472: External Control of Assumed-Immutable Web Parameter
  CWE-602: Client-Side Enforcement of Server-Side Security
  OWASP A05:2021 – Security Misconfiguration
- **IMPACT :**

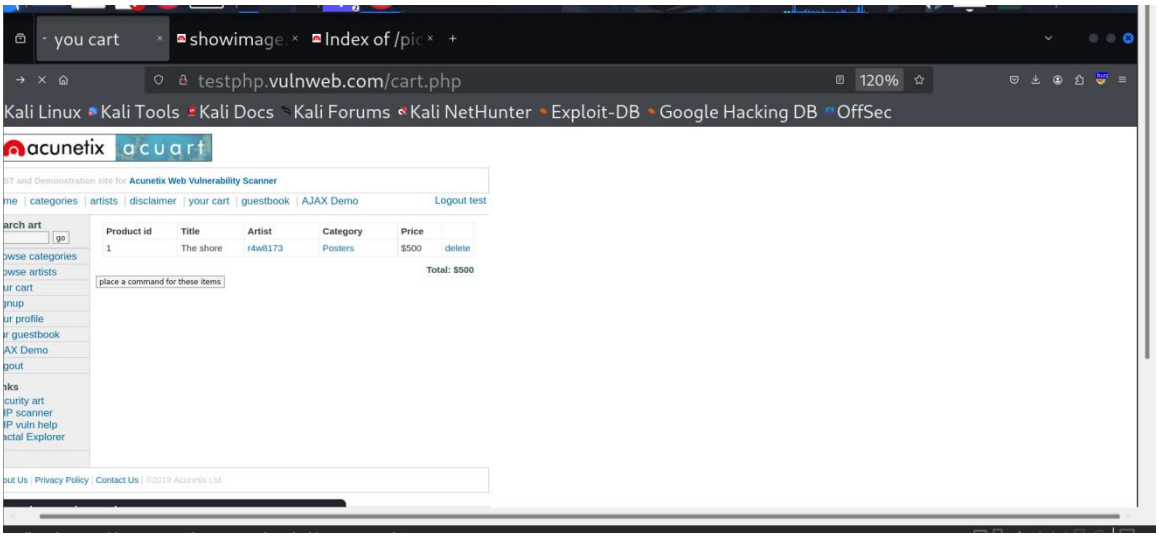
Attackers can purchase high-value items at lower prices, leading to

revenue loss. This can also lead to reputation damage and potential exploitation of other parts of the application.

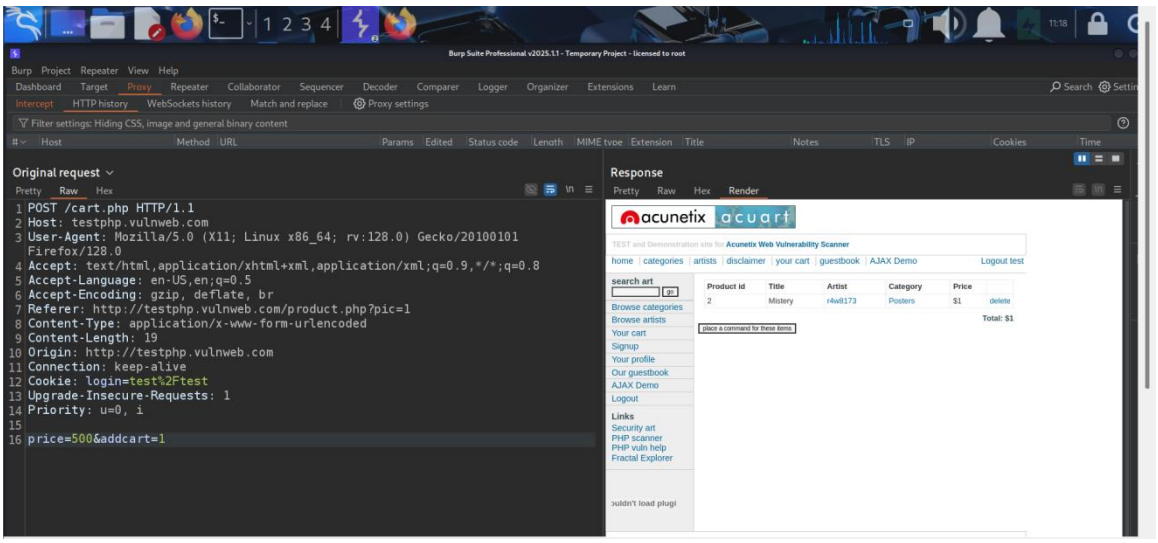
- **MITIGATIONS :**
  - Never trust client-side input for sensitive values like price.
  - Fetch item prices server-side from a secure database.
  - Implement server-side validation for all cart and transaction details.
  - Use signed tokens for pricing or implement checksums.
  - Monitor abnormal transaction patterns.
- **REFERENCES :** - https://owasp.org/Top10/A05_2021-Security_Misconfiguration/
  - https://cwe.mitre.org/data/definitions/472.html
  - https://portswigger.net/web-security/parameter-tampering

- **AFFECTED URL'S :**
  http://testphp.vulnweb.com/cart.php

- **POC :**



Firstly here we have a amount of $500 .Now by capturing the request in burpsuite we can manipulate the price



"After capturing the request, we successfully manipulated the product's cost."

7.12 VULNERABILITY : Insecure Cross-Domain Policy via Misconfigured crossdomain.xml

- **DESCRIPTION :**

  The server hosts a crossdomain.xml file that allows all domains (domain="*"), all ports (to-ports="*"), and disables secure connections (secure="false"). This overly permissive policy enables any external domain to interact with sensitive resources on the server, violating the Same-Origin Policy (SOP).

  This is typically exploited in old Flash-based applications but may still be leveraged in modern hybrid environments or misconfigured backend logic.

- **CVSS SCORE**: CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:C/C:H/I:H/A:N (8.1)
- **CVE / CWE / OWASP:**
  CWE-942: Overly Permissive Cross-domain Policy with Untrusted Domains
  OWASP Top 10: A05 – Security Misconfiguration
- **IMPACT :**

- Sensitive data may be accessed from untrusted domains.
- Facilitates Cross-Site Request Forgery (CSRF) or data theft via Flash/Java.
- Breaks browser isolation, enabling unauthorized interaction with backend APIs or sessions.
- Can be used in multi-stage attacks, like phishing or session hijacking.

- **MITIGATIONS :**

- Restrict the domain attribute to only known, trusted domains.
- Remove support for Flash crossdomain.xml if it's no longer needed.
- Use secure (https) communication and set secure="true" if still required.
- Regularly audit public files for excessive permissions.

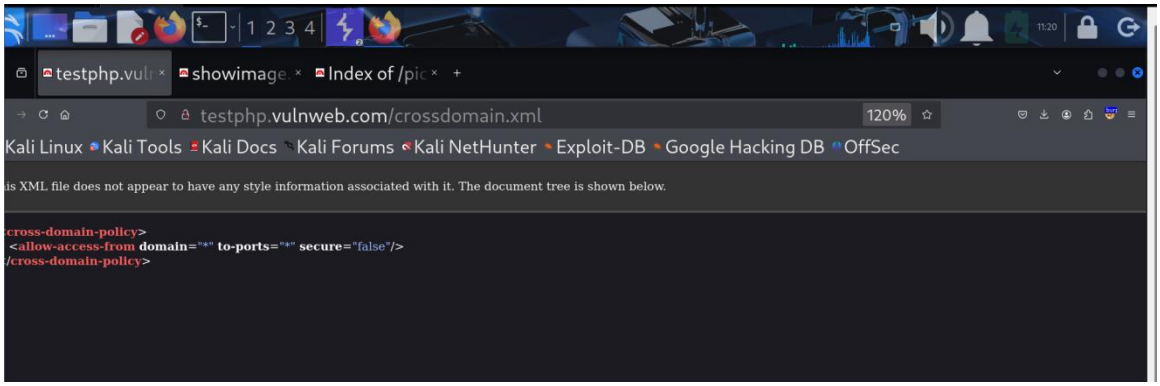
-

- CWE-942: https://cwe.mitre.org/data/definitions/942.html - OWASP Flash Cross-Domain Policy Risk: https://owasp.org/wwwcommunity/attacks/Flash_Cross-Domain_Policy

AFFECTED URL'S :

http://testphp.vulnweb.com/crossdomain.xml

- **POC :**



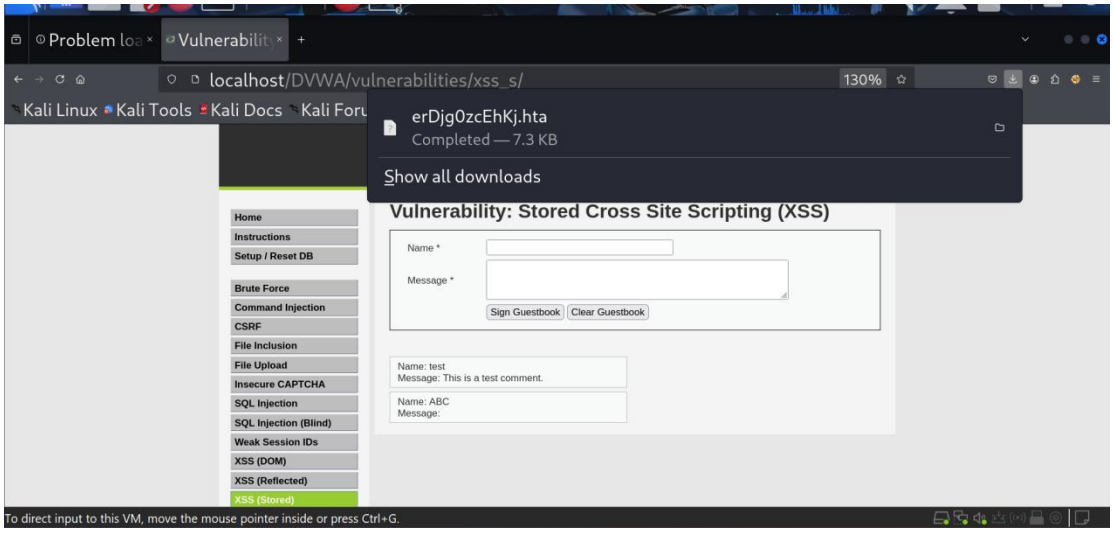
7.13 VULNERABILITY : Stored Cross-Site Scripting (XSS)

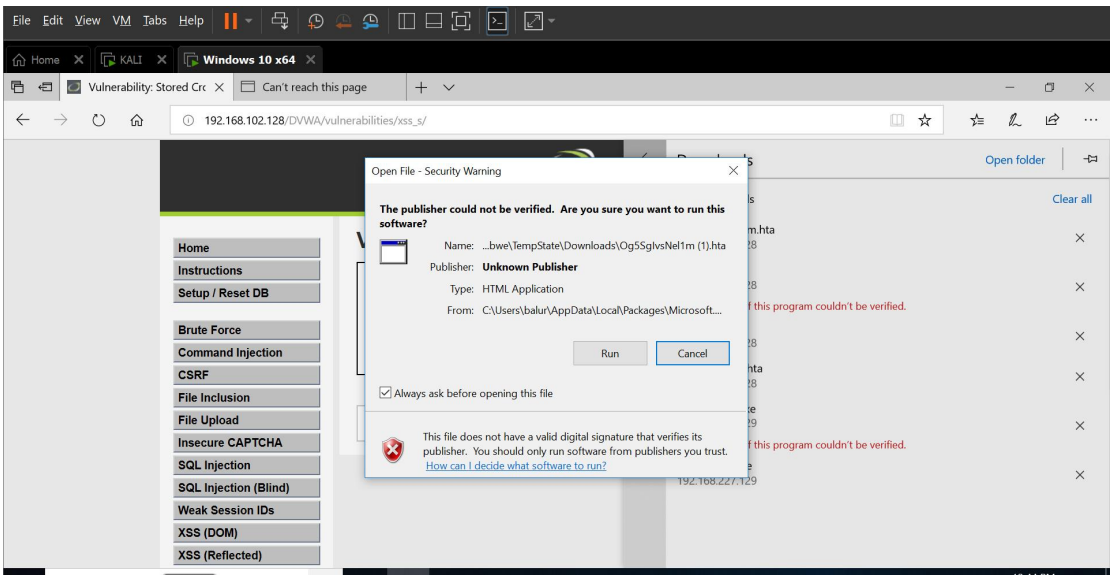
• **DESCRIPTION :** Stored XSS allows attackers to inject JavaScript that is permanently stored and executed in the browser of any user who visits the affected page.

- **CVSS SCORE**: CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:L/I:L/A:N (8.0)
- **CVE / CWE / OWASP REFERENCE :**
  CWE-79, OWASP A03:2021 – Injection
- **IMPACT :** Session hijacking, JavaScript-based attacks, phishing or malware delivery.
- **MITIGATIONS:**
  HTML encode all output, sanitize input, use Content Security Policy (CSP).
- **REFERENCES :**
  PortSwigger Web Security Academy - Stored XSS: https://portswigger.net/web-security/cross-site-scripting/stored
  National Vulnerability Database (NVD) - CVE-2024-21678 (Example Stored XSS with CVSS 8.5): https://nvd.nist.gov/vuln/detail/CVE-2024-21678 (Note: This is an example of a specific real-world CVE for a Stored XSS, not for DVWA itself).
  DVWA Official GitHub (for source code analysis): https://github.com/digininja/DVWA

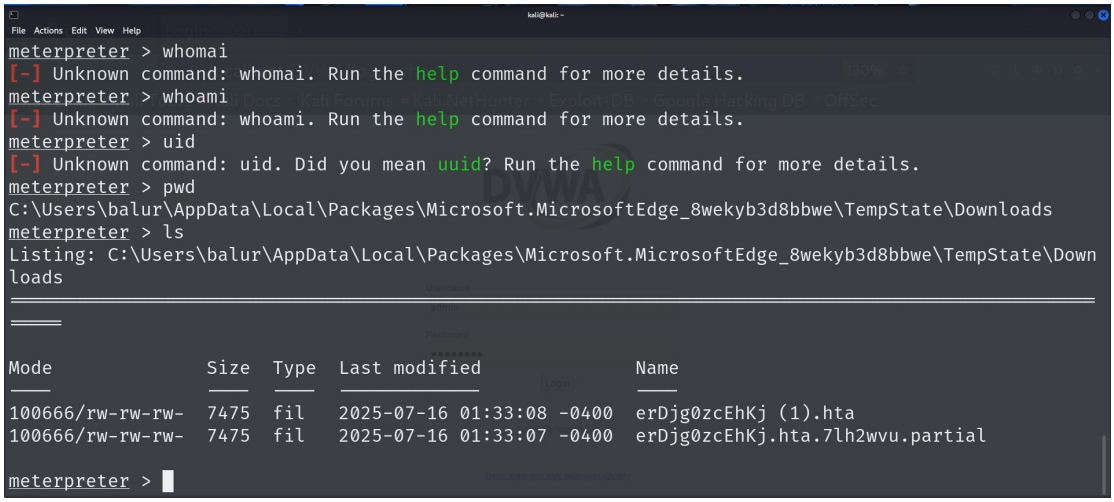
- **AFFECTED URL'S :**http://192.168.102.129/dvwa/vulnerabilities/xss_s/

- **POC:**





I got reverse shell in windows 10 by stored xss



7.14 VULNERABILITY : Insecure HTTP Method: PUT Enabled on Sensitive Endpoint

- **DESCRIPTION :**

  The web application on http://testphp.vulnweb.com accepts HTTP PUT requests on the endpoint /artists.php. This behavior suggests a server-side misconfiguration. Although the PUT method is not functionally used by the application, the server responds with a 200 OK, indicating it is permissively allowing unexpected HTTP methods.

  While no file was actually uploaded or modified, the mere acceptance of PUT requests without restrictions increases the risk of

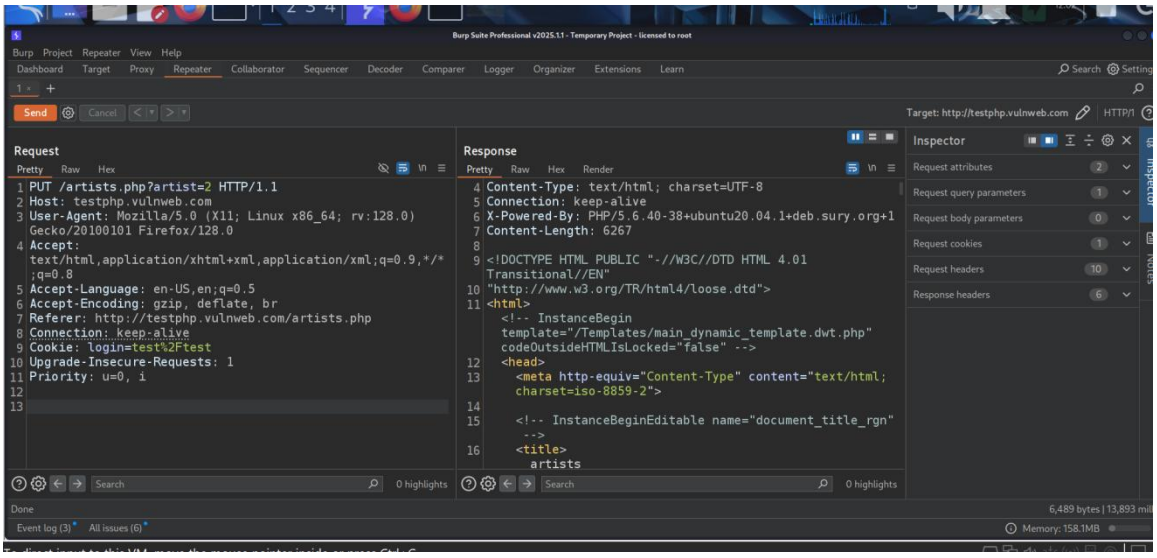
- **CVSS SCORE:** CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:L/I:H/A:L (8.2)

- **CWE / OWASP REFERENCES:**
  - CWE-668: Exposure of Resource to Wrong Sphere
  - CWE-749: Exposed Dangerous Method or Function
  - OWASP A05:2021 – Security Misconfiguration

- **IMPACT :** May allow attackers to probe backend logic with unexpected methods.Can be chained with directory traversal, LFI, or RCE vulnerabilities in rare cases.Exposes attack surface that should be closed by principle of least privilege.

- **MITIGATIONS :** Configure the web server (e.g., Apache, Nginx) to **disable PUT, DELETE, OPTIONS, etc.** unless explicitly required.Only allow GET/POST for public endpoints.Perform method filtering at both application and web server levelsAdd security headers and WAF rules to block unsupported HTTP methods.

- **REFERENCES :**
  - CWE-749: https://cwe.mitre.org/data/definitions/749.html
  - OWASP HTTP Method Hardening: https://owasp.org/www-project-secureheaders/#http-methods
- **AFFECTED URL's :**
  PUT /artists.php?artist=1
- **POC :**



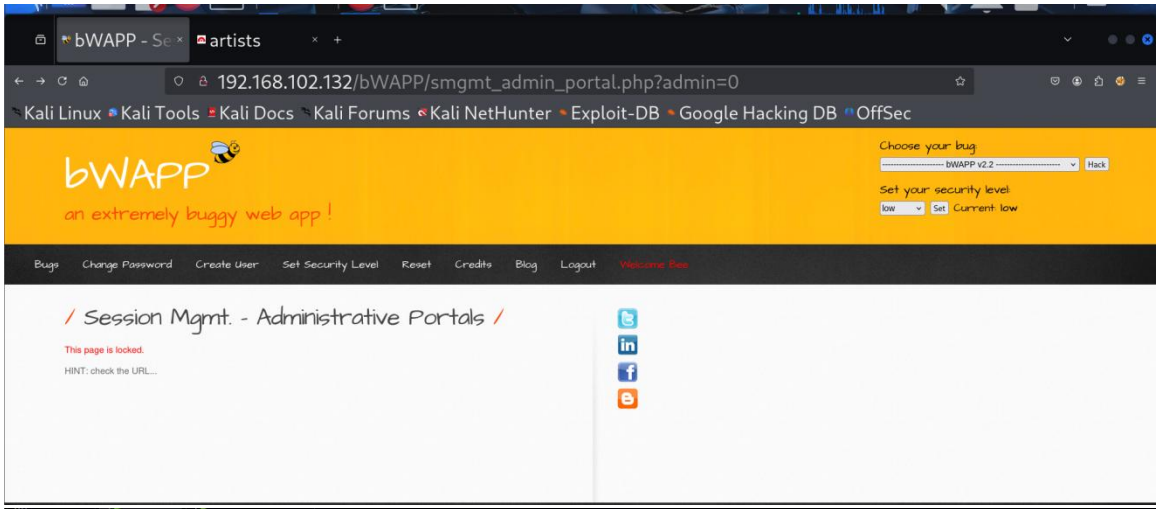
7.15 VULNERABILITY :Privilege Escalation via Insecure Session Management

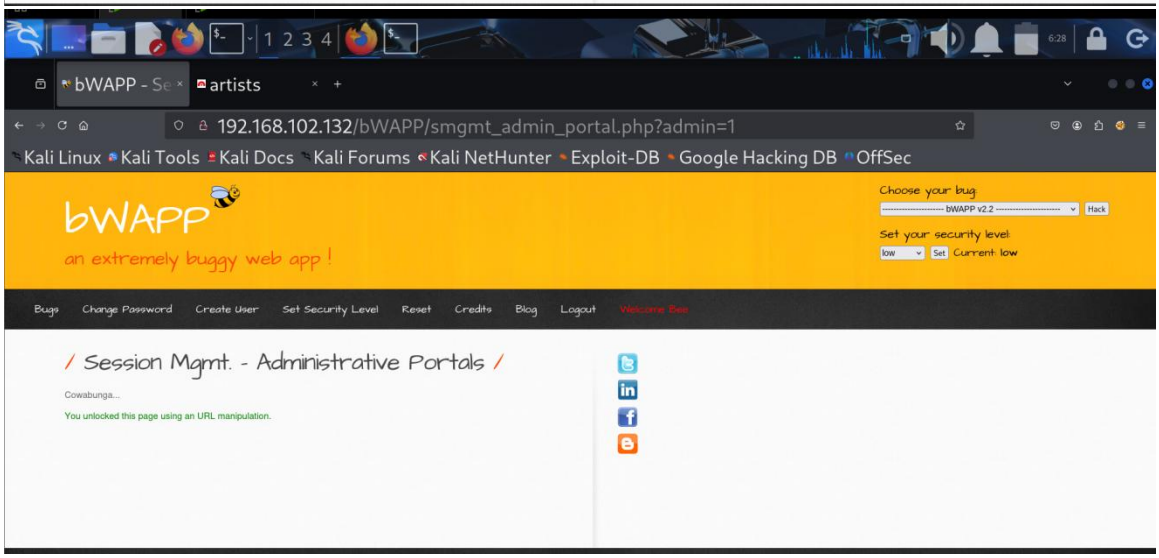
- **Description:** The bWAPP vulnerable application allows unauthorized privilege escalation due to poor session management. By modifying the admin parameter in the URL (e.g., from admin=0 to admin=1), a normal user can gain admin access
- **CVSS Score**: CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N
- **CWE / OWASP REFERENCE :**
  CWE-285: Improper Authorization
  OWASP A01:2021 – Broken Access Control
- **IMPACT:**
  Unauthorized access to admin panel
  Data exposure and manipulation
  Complete takeover of web functionality
- **MITIGATIONS :**
  Never trust user-controlled parameters for authorization
  Use server-side session validation
  Implement role-based access controls (RBAC)
  Secure session tokens and permissions
- **REFERENCES :**
  https://owasp.org/Top10/A01_2021-Broken_Access_Control/
  CWE-285: Improper Authorization
  https://cwe.mitre.org/data/definitions/285.html
  CWE-639: Authorization Bypass Through User-Controlled Key
  https://cwe.mitre.org/data/definitions/639.html
  OWASP – Access Control Cheat Sheet
  https://cheatsheetseries.owasp.org/cheatsheets/Access_Control_Cheat_Sheet.html

AFFECTED URL'S :
http://<target-ip>/bWAPP/admin.php?admin=1

VULNERABLE PARAMETER:
admin (GET method)

POC :





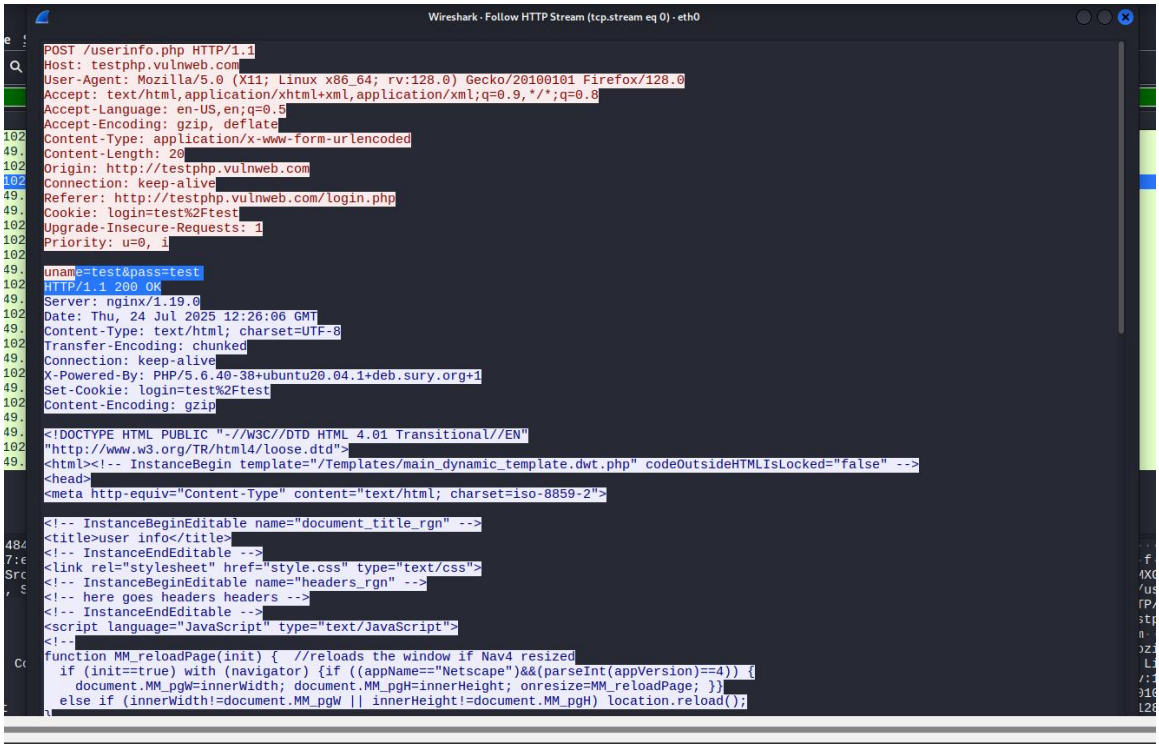
7.16 VULNERABILITY : Credentials Transmitted Over Insecure (Unencrypted) Channel

- **DESCRIPTION :** The login credentials (username and password) on the Artist VulnHub machine are transmitted in plaintext over an unencrypted HTTP connection. This vulnerability allows any attacker on the same network to intercept the credentials using tools like Wireshark or tcpdump.
- **CVSS SCORE :**Base Score: 7.4 (High) Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N
- **CWE / OWASP :** CWE-319: Cleartext Transmission of Sensitive Information

OWASP A02:2021 – Cryptographic Failures

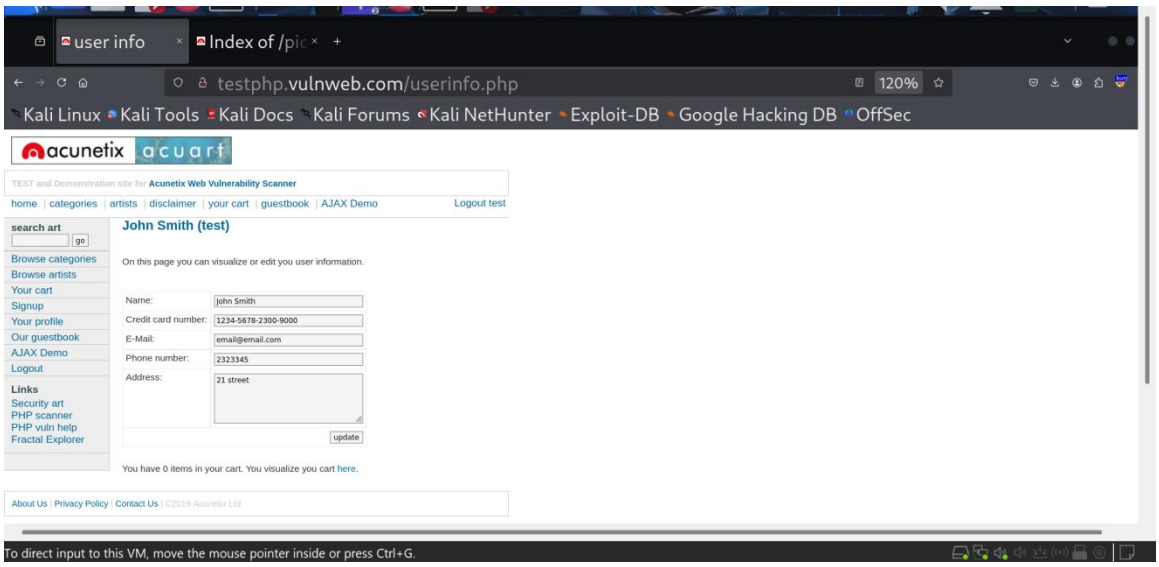
- **IMPACT :**
  Credentials can be stolen by any attacker with network access.
  Leads to unauthorized access and potential data breach.
  Vulnerable to MITM (Man-In-The-Middle) attacks
- **MITIGATIONS :** Enforce HTTPS using TLS on all pages that handle credentials.
  Use HSTS headers to ensure HTTPS is always used.
  Redirect all HTTP traffic to HTTPS.
  •Avoid handling credentials on unencrypted channels.
- **REFERENCES :** CWE-319: https://cwe.mitre.org/data/definitions/319.html
- • OWASP Cryptographic Failures: https://owasp.org/Top10/A02_2021-Cryptographic_Failures/
-

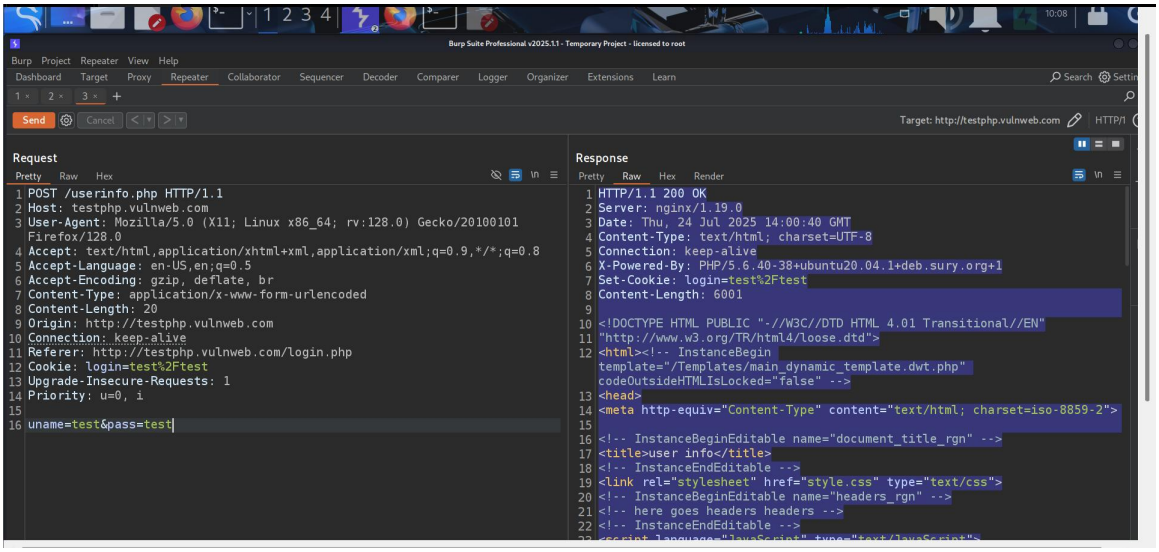
POC :

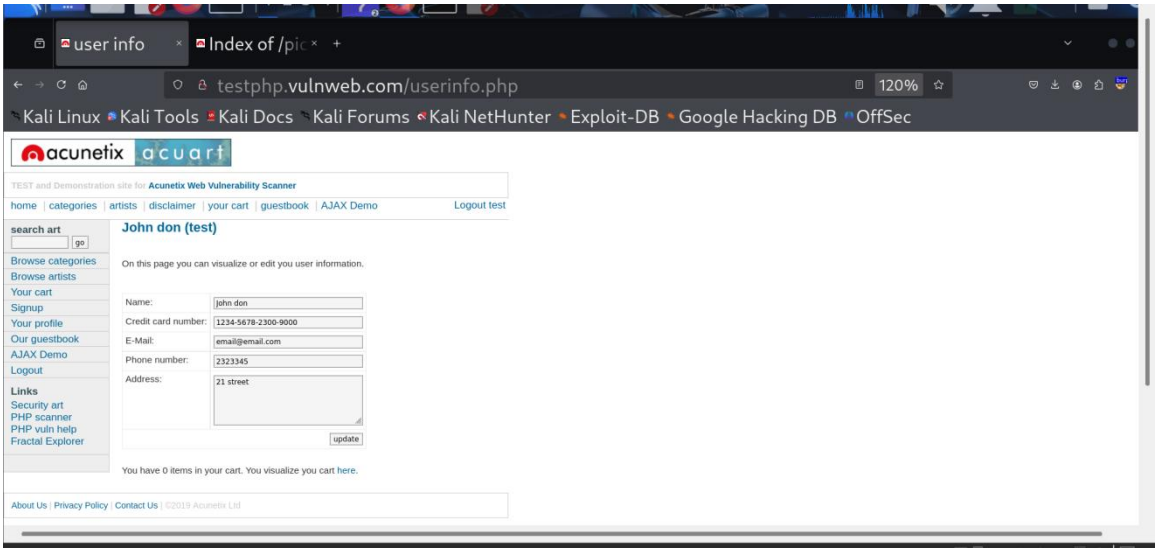


7.17 VULNERABILITY :Authentication Bypass via Client-Side Response Tampering

- **DESCRIPTION :**This vulnerability allows an attacker to bypass authentication checks by modifying the response of an unsuccessful login attempt with that of a successful login. The application fails to properly validate sessions and user credentials server-side, leading to unauthorized access and potential session tampering.
- **CVSS SCORE**:7.1 (High)CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:N/A:
- **CWE / OWASP :**
  - CWE-287: Improper Authentication
  - OWASP A01:2021 - Broken Access Control
  - OWASP A07:2021 - Identification and Authentication Failures
- **IMPACT :**
  An attacker can gain unauthorized access to restricted areas of the web application by crafting a malicious request. This may lead to exposure of sensitive user information, unauthorized actions on behalf of users, and full compromise of the user's session or data.
- **MITIGATIONS :**
  - Implement proper session validation on the server-side.
  - Do not rely on client-side data for login success or failure.
  - Use secure session tokens and regenerate them after login.
  - Validate every session and request with proper authentication checks.
  - Log and alert on unusual login behavior or session hijacking attempts.
- **REFERENCES :**
  - https://owasp.org/Top10/A01_2021-Broken_Access_Control/
  - https://owasp.org/Top10/A07_2021-Identification_and_Authentication_Failures/
  - https://cwe.mitre.org/data/definitions/287.html
  - https://portswigger.net/web-security/authentication
- **AFFECTED URL'S** : http://testphp.vulnweb.com/
- **POC :**


user info
Index of /pic
testphp.vulnweb.com/userinfo.php
120%
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
acunetix acuart
TEST and Demonstration site for Acunetix Web Vulnerability Scanner
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
Logout
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: email@email.com
Phone number: 2323345
Address: 21 street
update
You have 0 items in your cart. You visualize you cart here.
About Us | Privacy Policy | Contact Us | ©2019 Acunetix Ltd
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.


Burp Suite Professional v2025.1.1 - Temporary Project - licensed to root
Burp Project Repeater View Help
Dashboard Target Proxy Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Send Cancel
Target: http://testphp.vulnweb.com
Request
Pretty Raw Hex
1 POST /userinfo.php HTTP/1.1
2 Host: testphp.vulnweb.com
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 20
9 Origin: http://testphp.vulnweb.com
10 Connection: keep-alive
11 Referer: http://testphp.vulnweb.com/login.php
12 Cookie: login=test%2Ftest
13 Upgrade-Insecure-Requests: 1
14 Priority: u=0, i
15
16 uname=test&pass=test
Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Server: nginx/1.19.0
3 Date: Thu, 24 Jul 2025 14:00:40 GMT
4 Content-Type: text/html; charset=UTF-8
5 Connection: keep-alive
6 X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
7 Set-Cookie: login=test%2Ftest
8 Content-Length: 6001
9
10 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
11 "http://www.w3.org/TR/html4/loose.dtd">
12 <html><!-- InstanceBegin
template="/Templates/main_dynamic_template.dwt.php"
codeOutsideHTMLIsLocked="false" -->
13 <head>
14 <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-2">
15
16 <!-- InstanceBeginEditable name="document_title_rgn" -->
17 <title>user info</title>
18 <!-- InstanceEndEditable -->
19 <link rel="stylesheet" href="style.css" type="text/css">
20 <!-- InstanceBeginEditable name="headers_rgn" -->
21 <!-- here goes headers headers -->
22 <!-- InstanceEndEditable -->


user info
Index of /pic
testphp.vulnweb.com/userinfo.php
120%
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
acunetix acuart
TEST and Demonstration site for Acunetix Web Vulnerability Scanner
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
Logout
Links
Security art
PHP scanner
PHP vuln help
Fractal Explorer
John don (test)
On this page you can visualize or edit you user information.
Name: john don
Credit card number: 1234-5678-2300-9000
E-Mail: email@email.com
Phone number: 2323345
Address: 21 street
update
You have 0 items in your cart. You visualize you cart here.
About Us | Privacy Policy | Contact Us | ©2019 Acunetix Ltd

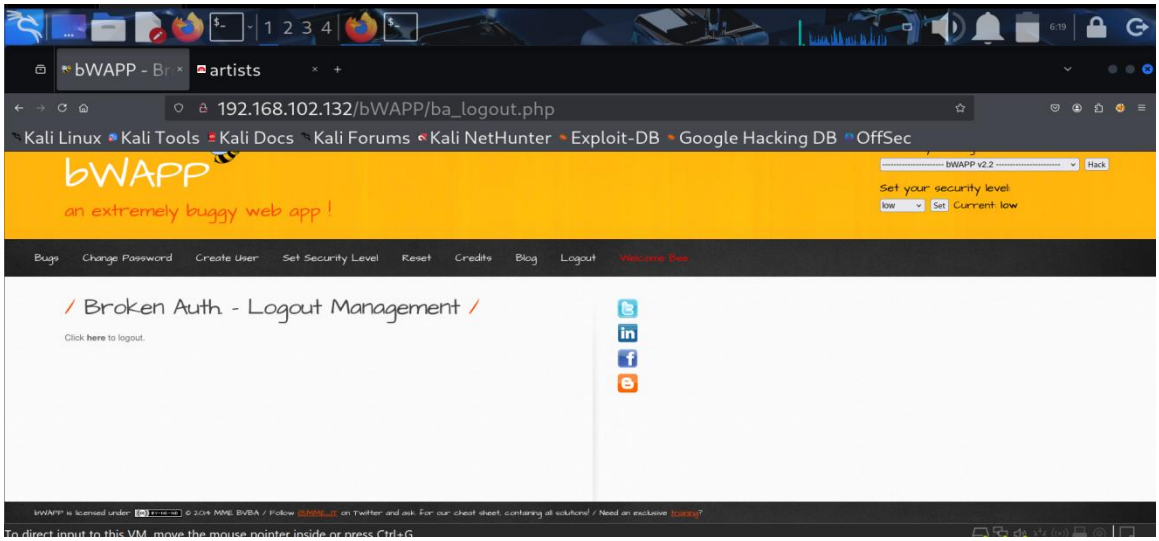
7.18 VULNERABILITY : Insecure Logout Functionality

- **DESCRIPTION :**
  The logout function does not properly destroy the session or invalidate cookies, allowing session reuse.
- **CVSS SCORE**: CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N (8.8)
- **CWE / OWASP :**
  - CWE-613
  - OWASP A07:2021
- **IMPACT :** Session hijacking
  Session fixation.
- **MITIGATIONS :**
  - Call session_destroy() and unset($_SESSION)
  - Use secure flags in cookies
  - Implement CSRF tokens for logout
- **REFERENCES: https://owasp.org/Top10/A07_2021-Identification_and_Authentication_Failures**

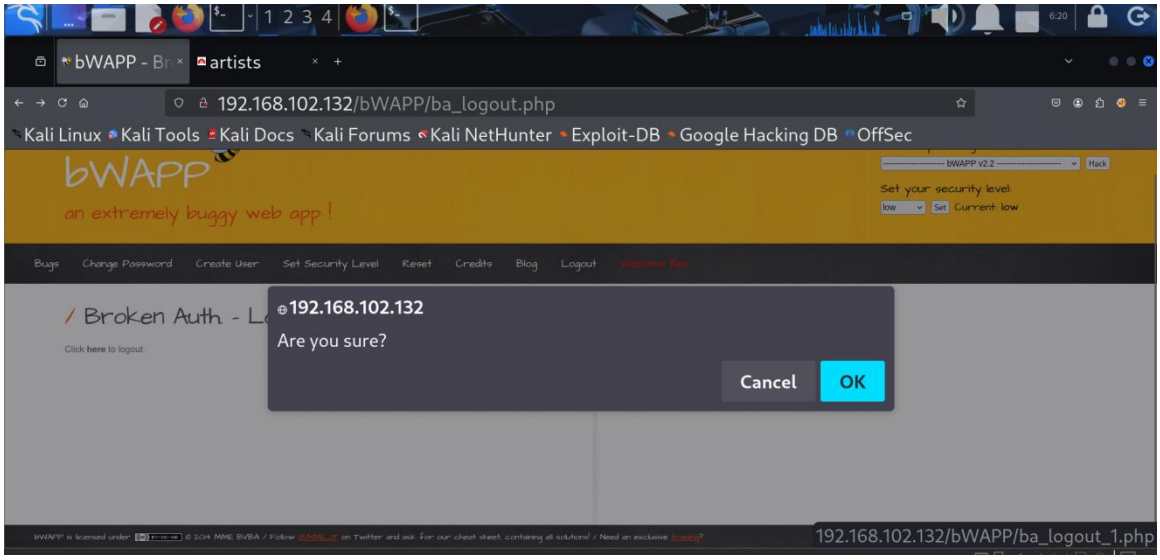
AFFECTED URL'S :

http://192.168.102.132/bWAPP/ba_logout.php

- **POC :**



After logging out if we again click that backarrow we will be able to login into the session again



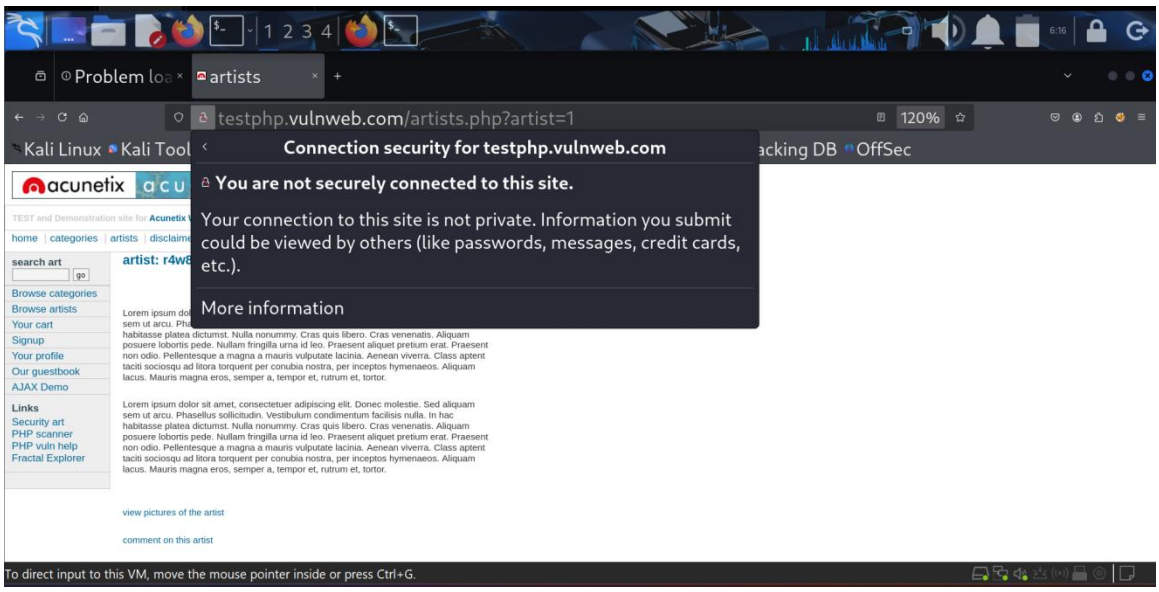

MEDIUM

7.19 Vulnerability : – Insecure Communication over HTTP

- **DESCRIPTION :**
  The site uses HTTP, not HTTPS, exposing data in transit to sniffing or tampering.
- **CVSS SCORE**: CVSS: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:L/I:L/A:N (6.1)
- **CWE AND OWASP REFERENCES :**
  CWE/CVE/OWASP:
  - CWE-319
  - OWASP A02:2021

- **IMPACT :**
  Eavesdropping on credentials and session cookies
  MITM attacks possible on public networks
  Implement SSL/TLS certificates to enforce HTTPS for all communication
  Use HTTP Strict Transport Security (HSTS) to force clients to use HTTPS
- **MITIGATIONS :** - Enforce HTTPS site-wide
  - Set Secure and HttpOnly cookie flags
- **REFERENCES :**
- https://owasp.org/www-project-top-ten/2017/A2_2017-Broken_Authentication
- **AFFECTED URL's :**
  https://testphp.vulnweb.com/

- **POC :**



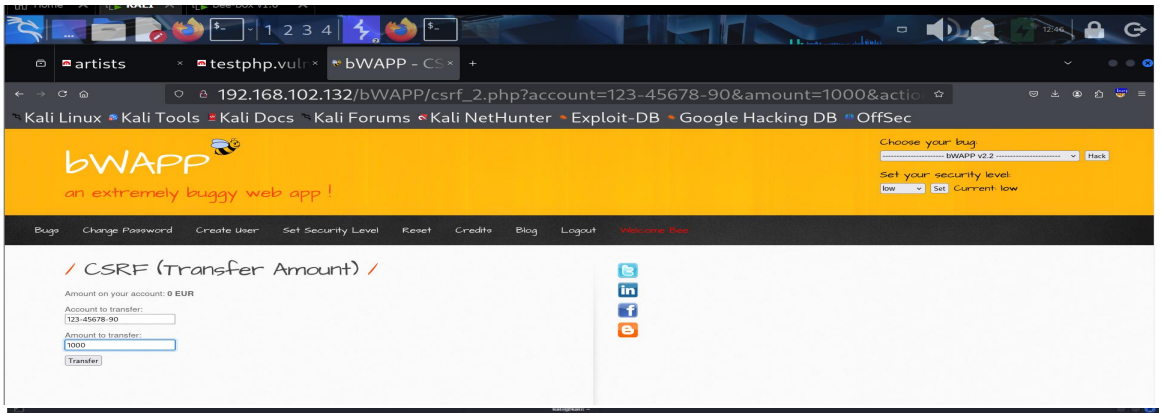
7.20 VULNERABILITY : CSRF in Sensitive Transaction (Transfer Amount)

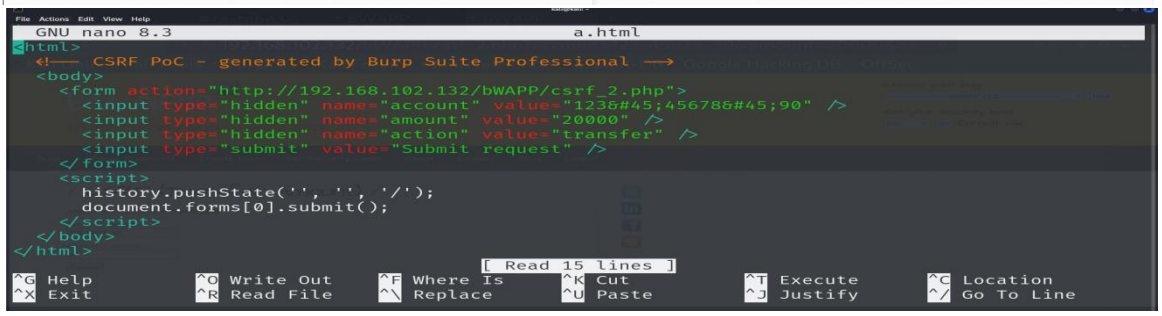
- **DESCRIPTION :**
  The 'Transfer Amount' functionality in the bWAPP vulnerable web application is susceptible to Cross-Site Request Forgery (CSRF). This allows an attacker to trick an authenticated user into submitting a forged request that performs unauthorized actions — such as transferring money to an attacker's account.

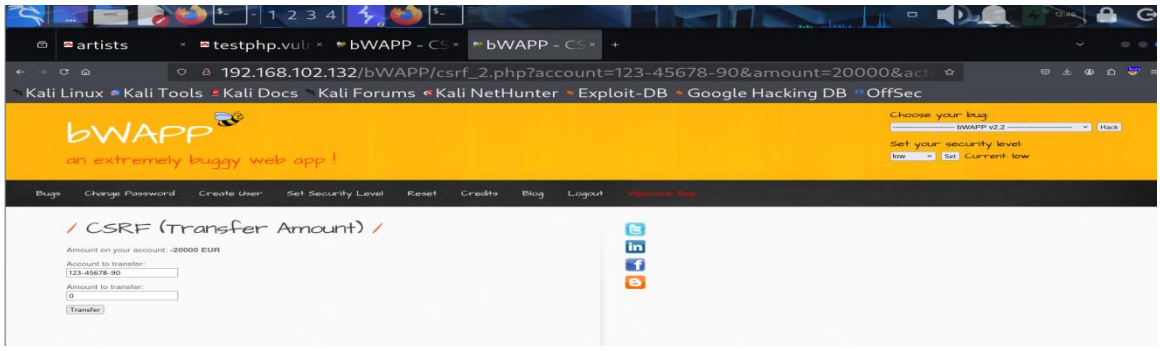
- **CVSS:** 6.5 – Medium
  Vector: AV:N/AC:L/PR:L/UI:R/S:U/C:L/I:L/A:N

- **CWE/OWASP REFERENCES:**
  - CWE-352: Cross-Site Request Forgery (CSRF)
  - OWASP Top 10: A05:2021 – Security Misconfiguration
- **IMPACT :**
  - Unauthorized money transfer to attacker’s account.
  - Victim is unaware the transaction occurred.
  - Exploitable even by just visiting a malicious link while logged in.
  - Can be extended to other critical actions (e.g., password change).

- **MITIGATIONS :**
  - Implement anti-CSRF tokens on all state-changing requests.
  - Use SameSite and Secure cookie attributes.
  - Ensure Referer / Origin header checks are enforced.
  - Set security level to medium/high in bWAPP to activate CSRF protection for lab.
- **REFERENCES** : - https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html
  - https://cwe.mitre.org/data/definitions/352.html
- **AFFECTED URL'S** :http://192.168.102.132/bWAPP/csrf_2.php?account=123-45678-90&amount=1000&action=transfer
- **POC :**



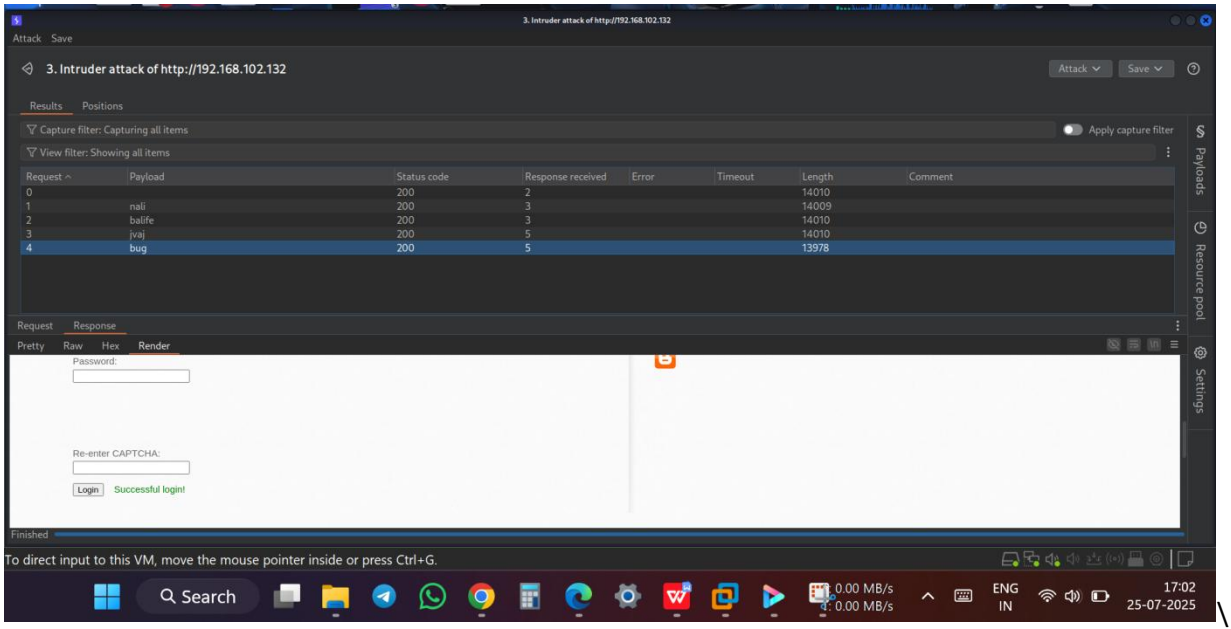




7.21 VULNERABILITY : CAPTCHA Bypass Due to Weak Validation

- **DESCRIPTION :**The CAPTCHA Bypassing vulnerability in the bWAPP web application allows an attacker to bypass CAPTCHA-based login restrictions. CAPTCHAs are designed to prevent automated brute-force attacks and bots. However, in this scenario, the implementation is flawed — the CAPTCHA session variable is only set when a specific script (captcha_box.php) is requested.
  By intercepting and dropping the captcha_box.php request, the CAPTCHA session variable $_SESSION['captcha'] is never set. When the form is submitted without the captcha_user parameter, the server fails to verify the CAPTCHA but still proceeds with login, thereby allowing authentication without human verification.
- **CVSS SCORE :** 6.5 (Medium**):**CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N
- **CWE/OWASP REFERENCES:**
  CWE ID: CWE-287 – Improper Authentication
  A07:2021 – Identification and Authentication Failures
- **IMPACT :-** Allows attackers to bypass CAPTCHA protections
  - Enables automated brute-force attacks on login forms
  - Reduces the overall security of authentication mechanisms
- **MITIGATIONS :-** Never rely solely on client-side CAPTCHA initialization
  - Validate all security inputs server-side
  - Always ensure CAPTCHA value ($_SESSION['captcha']) is required and properly validated
  - Use modern CAPTCHA APIs like Google reCAPTCHA v2/v3
  - Implement rate-limiting and account lockout after failed login attempts
- **REFERENCES: -** OWASP Broken Authentication: https://owasp.org/Top10/A07_2021-Identification_and_Authentication_Failures/
- - bWAPP: https://sourceforge.net/projects/bwapp/
- - Burp Suite: https://portswigger.net/burp
- **AFFECTED URL'S:**
  http://192.168.102.132/bWAPP/ba_captcha_bypass.php

- **POC:**



7.22 VULNERABILITY : Cross-Site Tracing (XST) via HTTP TRACE Method

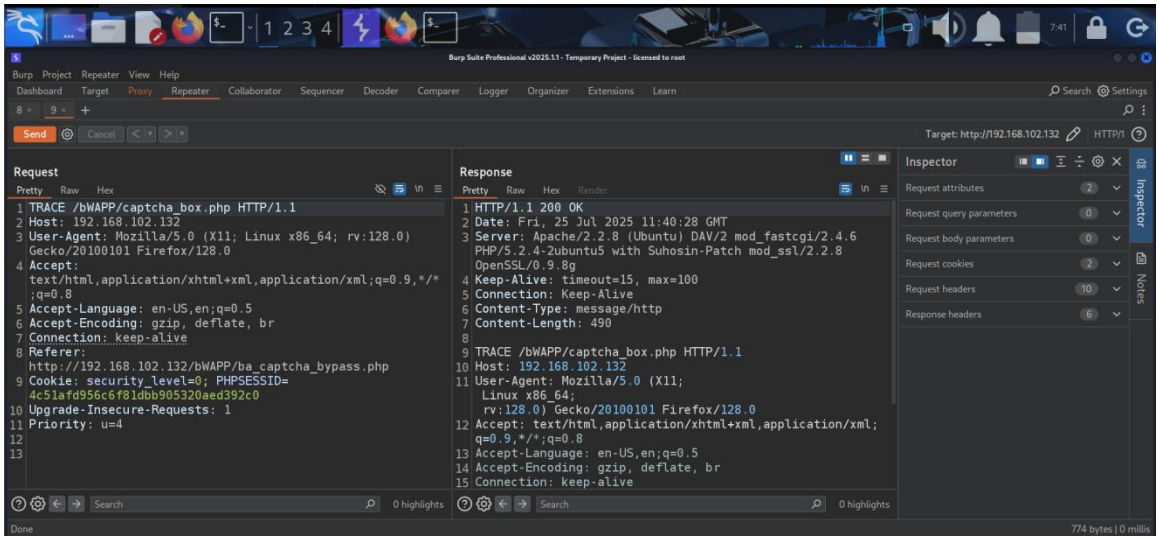
- **DESCRIPTION**:The HTTP TRACE method is a diagnostic feature that echoes back the received request. While useful for debugging, it poses a security risk when enabled in production environments. An attacker can abuse TRACE for Cross Site Tracing (XST), potentially stealing cookies or headers, especially when combined with Cross Site Scripting (XSS). This vulnerability can lead to information disclosure or session hijacking.
- **CVSS SCORE :**5.3 (Medium)CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:N/A:N
- **CWE/OWASP:**

CWE-601: URL Redirection to Untrusted Site ('Open Redirect')

CWE-693: Protection Mechanism Failure

CWE-200: Exposure of Sensitive Information to an Unauthorized Actor

- **IMPACT :-** Possible header and cookie disclosure
  - May enable session hijacking or token stealing in combination with XSS
  - Allows attackers to map internal headers or request structure
- **MITIGATIONS:-** Disable HTTP TRACE method on the server
  - For Apache: Add 'TraceEnable off' to your configuration
  - Use a Web Application Firewall (WAF) to block unsupported methods
  - Harden web servers and reduce attack surface by allowing only necessary HTTP methods
- **REFERENCES:- OWASP XST: https://owasp.org/www-community/attacks/Cross_Site_Tracing**
  - Apache TraceEnable: https://httpd.apache.org/docs/2.4/mod/core.html#traceenable
  - PortSwigger TRACE Testing: https://portswigger.net/web-security/all-translations/en/http-methods#trace
- **AFFECTED URL'S:https://testphp.vulnweb.com/**
- **POC:**



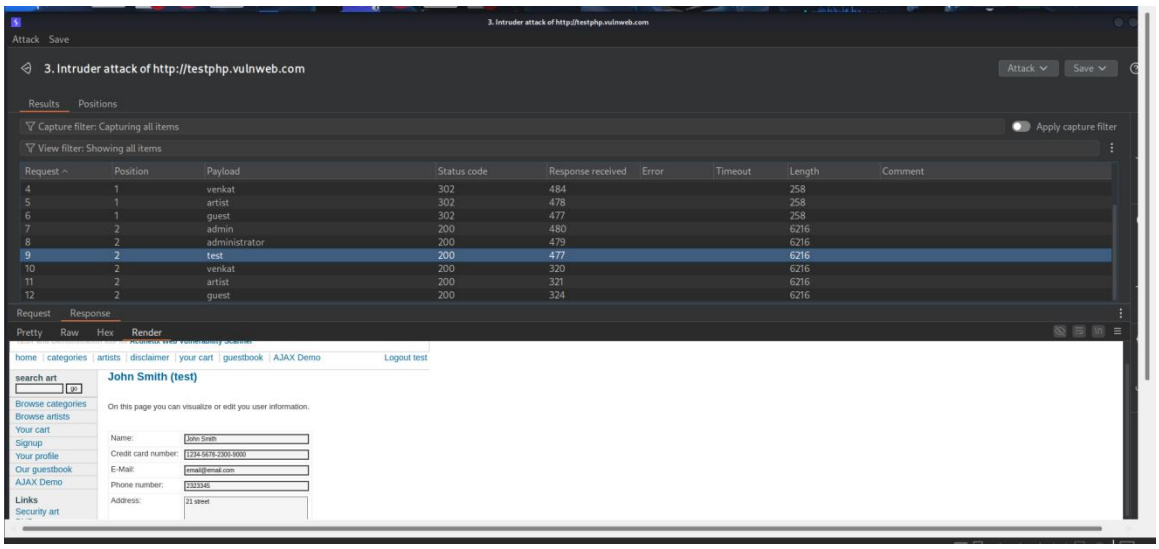
7.23 VULNERABILITY : Brute Force with Username Enumeration

- **DESCRIPTION:**This vulnerability allows attackers to perform brute-force attacks while identifying valid usernames based on different server responses.
  When incorrect usernames and passwords are submitted to the login form,

different responses (e.g., "Invalid username" vs "Invalid password") indicate which part of the credential is incorrect. This enables attackers to enumerate valid usernames and then target those accounts with brute-force password guessing.

- **CVSS SCORE:**CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:N (8.1)

- **CWE/OWASP REFERENCES:-** CWE-204: Response Discrepancy Information Exposure
  - CWE-521: Weak Password Requirements
  - OWASP Top 10: A07 – Identification and Authentication Failures
- **IMPACT :** Username enumeration allows attackers to identify valid accounts.
  - With brute-force, attackers may gain unauthorized access to user/admin accounts.
  - Can lead to privilege escalation and full system compromise.
- **MITIGATIONS:-** Implement generic error messages (e.g., “Invalid credentials”).
  - Enforce account lockout after multiple failed attempts.
  - Use CAPTCHA or rate-limiting to slow down automated attacks.
  - Use strong, unique passwords and MFA (Multi-Factor Authentication)**.**
- **REFERENCES:-** https://owasp.org/Top10/A07_2021-Identification_and_Authentication_Failures/
  - https://nvd.nist.gov/vuln/detail/CVE-2021-22205
  - https://portswigger.net/kb/issues/00600400_username-enumeration
- **AFFECTED URL’S**:https://testphp.vulnweb.com/login.php
- **POC:**



7.24 VULNERABILITY : Reflected Cross-Site Scripting (XSS)

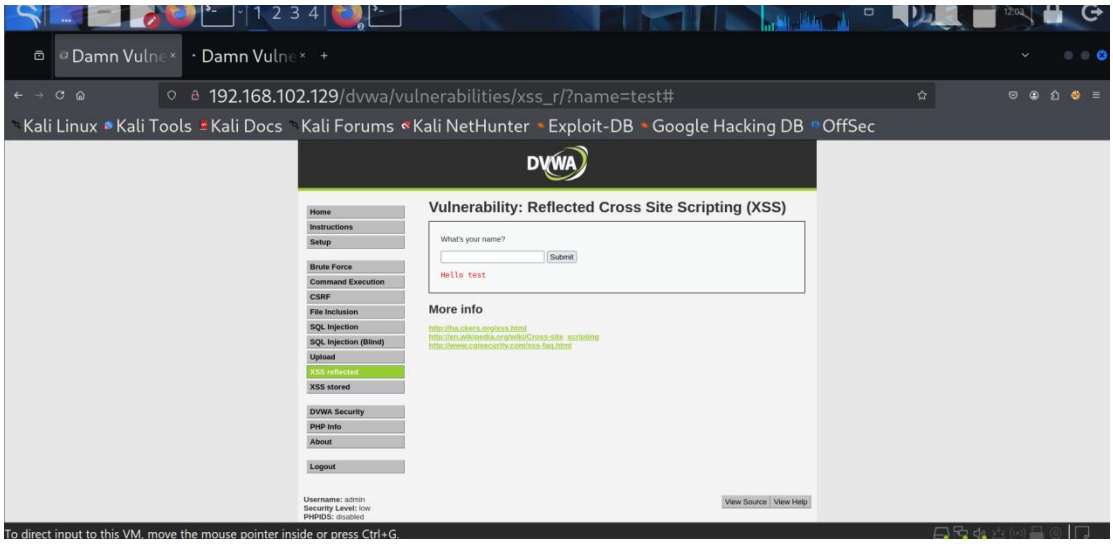
- **DESCRIPTION:** Reflected XSS occurs when malicious scripts are immediately returned in HTTP responses
- **CVSS SCORE:** 6.1 (Medium) – CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:L/I:L/A:N
- **CWE/OWASP REFERENCES:-** CWE-79, OWASP A03:2021 – Injection
- **IMPACT :**Session theft, redirection to malicious sites, phishing**.**
- **MITIGATIONS:**Escape output properly, sanitize GET parameters, use HTTP-only cookies.
- **REFERENCES:**OWASP XSS (Cross Site Scripting) Prevention Cheat Sheet: https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html

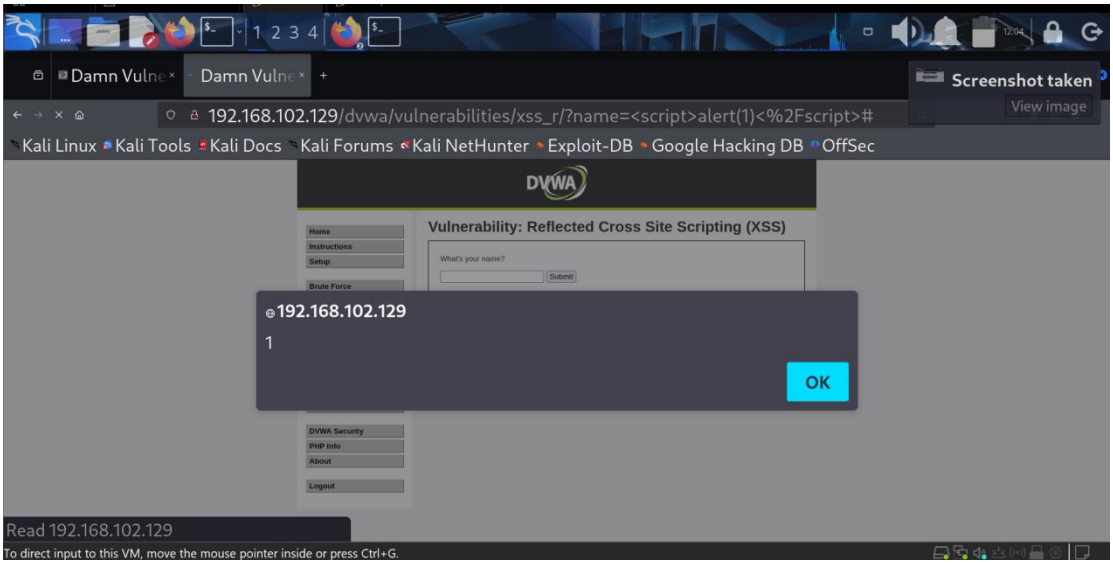
  OWASP Top 10 (2021) - A03:2021 - Injection: https://owasp.org/Top10/A03_2021- Injection/

  DVWA Official GitHub: https://github.com/digininja/DVWA

- **AFFECTED URL'S:** http://localhost/DVWA/vulnerabilities/xss_r/

- **POC:**


Damn Vulne
192.168.102.129/dvwa/vulnerabilities/xss_r/?name=test#
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
DVWA
Vulnerability: Reflected Cross Site Scripting (XSS)
What's your name?
Submit
Hello test
More info
Home
Instructions
Setup
Brute Force
Command Execution
CSRF
File Inclusion
SQL Injection
SQL Injection (Blind)
Upload
XSS reflected
XSS stored
DVWA Security
PHP Info
About
Logout
View Source
View Help
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.


Damn Vulne
192.168.102.129/dvwa/vulnerabilities/xss_r/?name=<script>alert(1)<%2Fscript>#
Screenshot taken
View image
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
DVWA
Vulnerability: Reflected Cross Site Scripting (XSS)
192.168.102.129
1
OK
Read 192.168.102.129
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

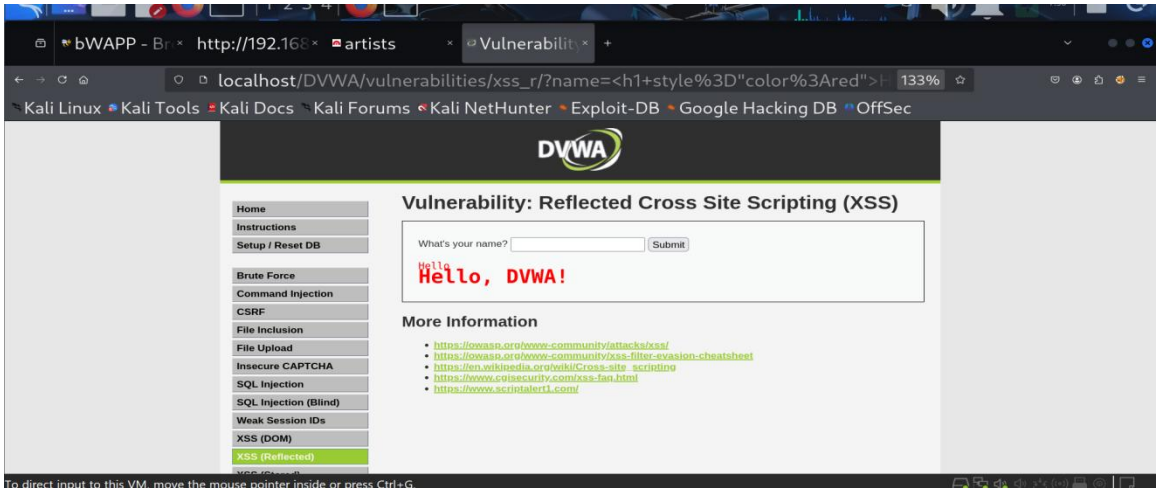

LOW

7.25 VULNERABILITY: Reflected HTML Injection (DVWA)

- **DESCRIPTION :** Reflected HTML Injection occurs when user-supplied input is immediately included in the HTML response without proper sanitization. Unlike XSS, this does not execute JavaScript but manipulates HTML structure.
- **CVSS SCORE**:3.1 (Low)
  Vector: AV:N/AC:L/PR:N/UI:R/S:U/C:N/I:L/A:N
- **CWE/OWASP REFERENCES:**
  CWE-79: Improper Neutralization of Input During Web Page Improper Neutralization of Script-Related HTML Tags in a Web Page (Basic XSS)
  A03:2021 - Injection
- **IMPACT** : An attacker may manipulate the appearance of the page, mislead users, or assist in phishing attacks. It lacks code execution, so the risk is limited.
- **MITIGATIONS :** 1. Encode all user inputs.
  2. Use frameworks or libraries that auto-sanitize inputs.
  3. Employ Content Security Policy (CSP).
- **AFFECTED URL'S:http://localhost/DVWA/vulnerabilities/xss_r/**

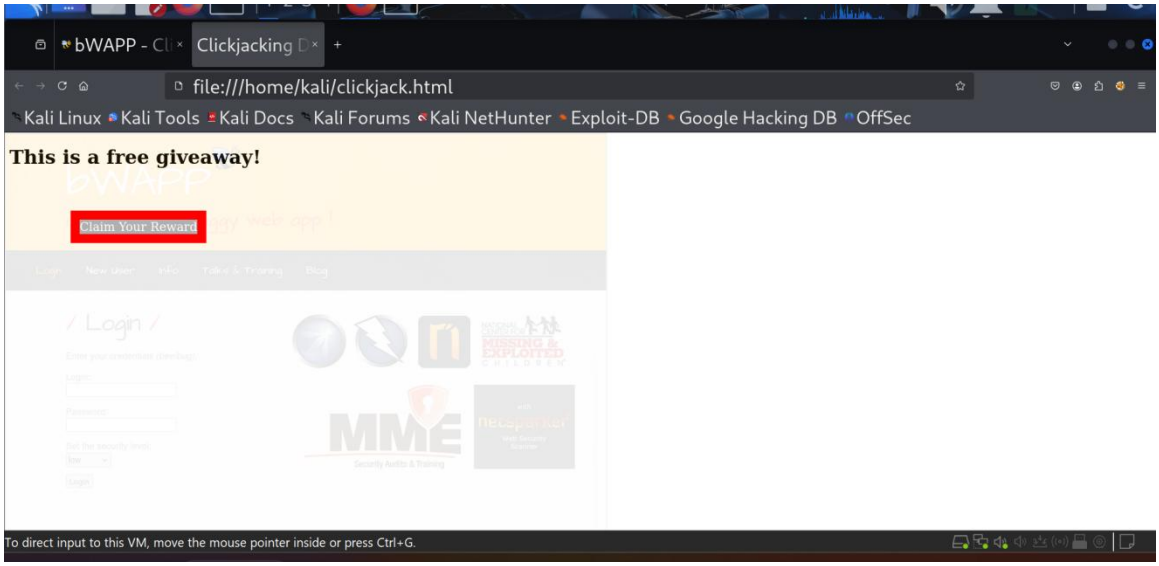
- **REFERENCES :**
  https://owasp.org/www-community/attacks/HTML_Injection
  - https://portswigger.net/web-security/html-injection

- **POC :**



7.26 Vulnerability:Clickjacking (bWAPP)

- **DESCRIPTION :**Clickjacking is a UI redress attack where a user is tricked into clicking on something different from what they perceive. The attacker loads a vulnerable page in an invisible iframe and overlays a fake button.
- **CVSS SCORE:** 3.7 (Low)
  Vector: AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N
- **CWE/OWASP REFERENCES:**
- **IMPACT:**Users may unintentionally click on elements like submit buttons or links, leading to unintended actions. The attacker does not directly gain access but can exploit trust and user interaction.
- **MITIGATIONS:**1. Use X-Frame-Options HTTP header: DENY or SAMEORIGIN.
  2. Use Content-Security-Policy (CSP): frame-ancestors 'none'.
  3. Avoid sensitive actions being triggered via simple clicks.
- **AFFECTED URL'S:**http://192.168.102.132/bWAPP/clickjacking.php
- **REFERENCES:**- https://owasp.org/www-community/attacks/Clickjacking - https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
- **POC:**



- **CONCLUSION :** The Web Application Penetration Testing conducted on the target systems — testphp.vulnweb.com, DVWA, and bWAPP — revealed a total of 26 vulnerabilities. These ranged from critical issues like Remote Code Execution, SQL Injection, and Broken Authentication, to medium and low-risk misconfigurations such as Clickjacking, Reflected HTML Injection, and CAPTCHA Bypass.

- **The breakdown by severity is as follows:**
- Critical: 9 vulnerabilities
- High: 9 vulnerabilities
- Medium: 6 vulnerabilities
- Low: 2 vulnerabilities
- This assessment highlights the importance of:
- Enforcing secure coding practices
- Implementing strong authentication and session management
- Disabling unsafe HTTP methods
- Validating and sanitizing all user inputs
-
- The testing was performed in a controlled lab setup using industry-standard tools such as Burp Suite, SQLMap, Wireshark, and Nmap, adhering to the OWASP Testing Guide v4 and OWASP Top 10 - 2021. All identified vulnerabilities have been documented with their CVSS scores, impacts, and remediation recommendations.
- We recommend immediate mitigation of critical and high-risk findings and the implementation of a continuous security testing strategy to protect the application and its users.