DVWA (Damn Vulnerable Web Application) - Attack Documentation

Prepared By: Pawan Kumar Singh
Platform: DVWA (Hosted on VM)
Purpose: To understand and demonstrate common web vulnerabilities in a safe.

<u>Overview and Objective</u>

This report presents the findings of a security audit performed on the Damn Vulnerable Web Application (DVWA), a deliberately insecure web application used for testing and learning web security.

<u>Objective:-</u> To identify and exploit at least three vulnerabilities in DVWA. - To understand how these vulnerabilities work. - To suggest appropriate mitigation techniques for each vulnerability. - To develop hands-on skills in penetration testing and vulnerability assessment.

Tools Used:-

- DVWA (Damn Vulnerable Web Application)
- Web Browser (e.g., Firefox/Chrome)
- Kali Linux.

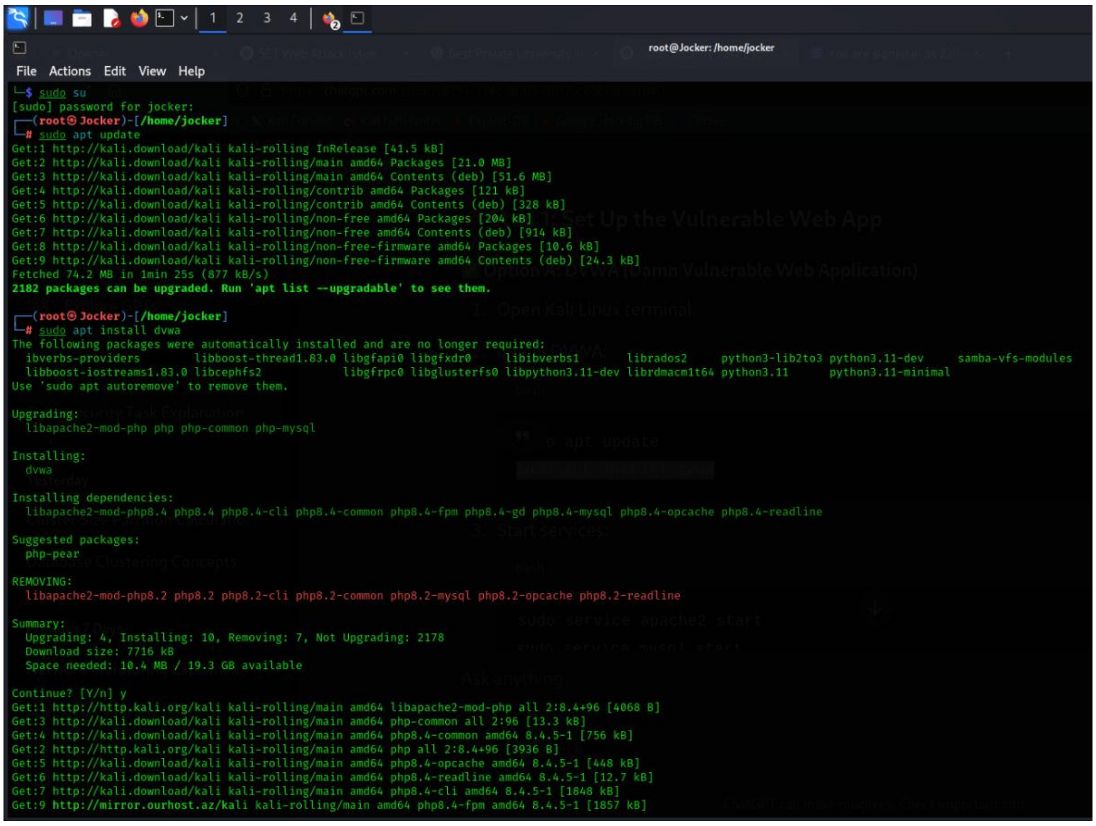
<u>Step-by-Step Guide: To find the Vlunerability in DVWA:-</u>

- **<u>Step 1: Set Up the Environment :-</u>**

  Here I am using Kali linux to make easy in web penteration testing because kali linux is giving lots of predefined tools.

  - First open the kali linux terminal
  - Install DVWA

```
sudo apt update
sudo apt install dvwa
```



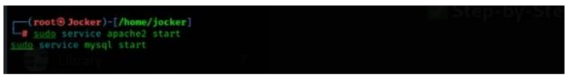
```
└─$ sudo su
[sudo] password for jocker:
┌──(root㉿Jocker)-[/home/jocker]
└─# sudo apt update
Get:1 http://kali.download/kali kali-rolling InRelease [41.5 kB]
Get:2 http://kali.download/kali kali-rolling/main amd64 Packages [21.0 MB]
Get:3 http://kali.download/kali kali-rolling/main amd64 Contents (deb) [51.6 MB]
Get:4 http://kali.download/kali kali-rolling/contrib amd64 Packages [121 kB]
Get:5 http://kali.download/kali kali-rolling/contrib amd64 Contents (deb) [328 kB]
Get:6 http://kali.download/kali kali-rolling/non-free amd64 Packages [204 kB]
Get:7 http://kali.download/kali kali-rolling/non-free amd64 Contents (deb) [914 kB]
Get:8 http://kali.download/kali kali-rolling/non-free-firmware amd64 Packages [10.6 kB]
Get:9 http://kali.download/kali kali-rolling/non-free-firmware amd64 Contents (deb) [24.3 kB]
Fetched 74.2 MB in 1min 25s (877 kB/s)
2182 packages can be upgraded. Run 'apt list --upgradable' to see them.

┌──(root㉿Jocker)-[/home/jocker]
└─# sudo apt install dvwa
The following packages were automatically installed and are no longer required:
  ibverbs-providers       libboost-thread1.83.0 libgfapi0 libgfxdr0     libibverbs1       librados2     python3-lib2to3 python3.11-dev     samba-vfs-modules
  libboost-iostreams1.83.0 libcephfs2            libgfrpc0 libglusterfs0 libpython3.11-dev librdmacm1t64 python3.11      python3.11-minimal
Use 'sudo apt autoremove' to remove them.

Upgrading:
  libapache2-mod-php php php-common php-mysql

Installing:
  dvwa

Installing dependencies:
  libapache2-mod-php8.4 php8.4 php8.4-cli php8.4-common php8.4-fpm php8.4-gd php8.4-mysql php8.4-opcache php8.4-readline

Suggested packages:
  php-pear

REMOVING:
  libapache2-mod-php8.2 php8.2 php8.2-cli php8.2-common php8.2-mysql php8.2-opcache php8.2-readline

Summary:
  Upgrading: 4, Installing: 10, Removing: 7, Not Upgrading: 2178
  Download size: 7716 kB
  Space needed: 10.4 MB / 19.3 GB available

Continue? [Y/n] y
Get:1 http://http.kali.org/kali kali-rolling/main amd64 libapache2-mod-php all 2:8.4+96 [4068 B]
Get:3 http://kali.download/kali kali-rolling/main amd64 php-common all 2:96 [13.3 kB]
Get:4 http://kali.download/kali kali-rolling/main amd64 php8.4-common amd64 8.4.5-1 [756 kB]
Get:2 http://http.kali.org/kali kali-rolling/main amd64 php all 2:8.4+96 [3936 B]
Get:5 http://kali.download/kali kali-rolling/main amd64 php8.4-opcache amd64 8.4.5-1 [448 kB]
Get:6 http://kali.download/kali kali-rolling/main amd64 php8.4-readline amd64 8.4.5-1 [12.7 kB]
Get:7 http://kali.download/kali kali-rolling/main amd64 php8.4-cli amd64 8.4.5-1 [1848 kB]
Get:9 http://mirror.ourhost.az/kali kali-rolling/main amd64 php8.4-fpm amd64 8.4.5-1 [1857 kB]
```

- After the Dvwa installation copy dvwa **share directory** to **Web directory**

```
sudo cp -r /usr/share/dvwa /var/www/html/
sudo chown -R www-data:www-data /var/www/html/dvwa
```

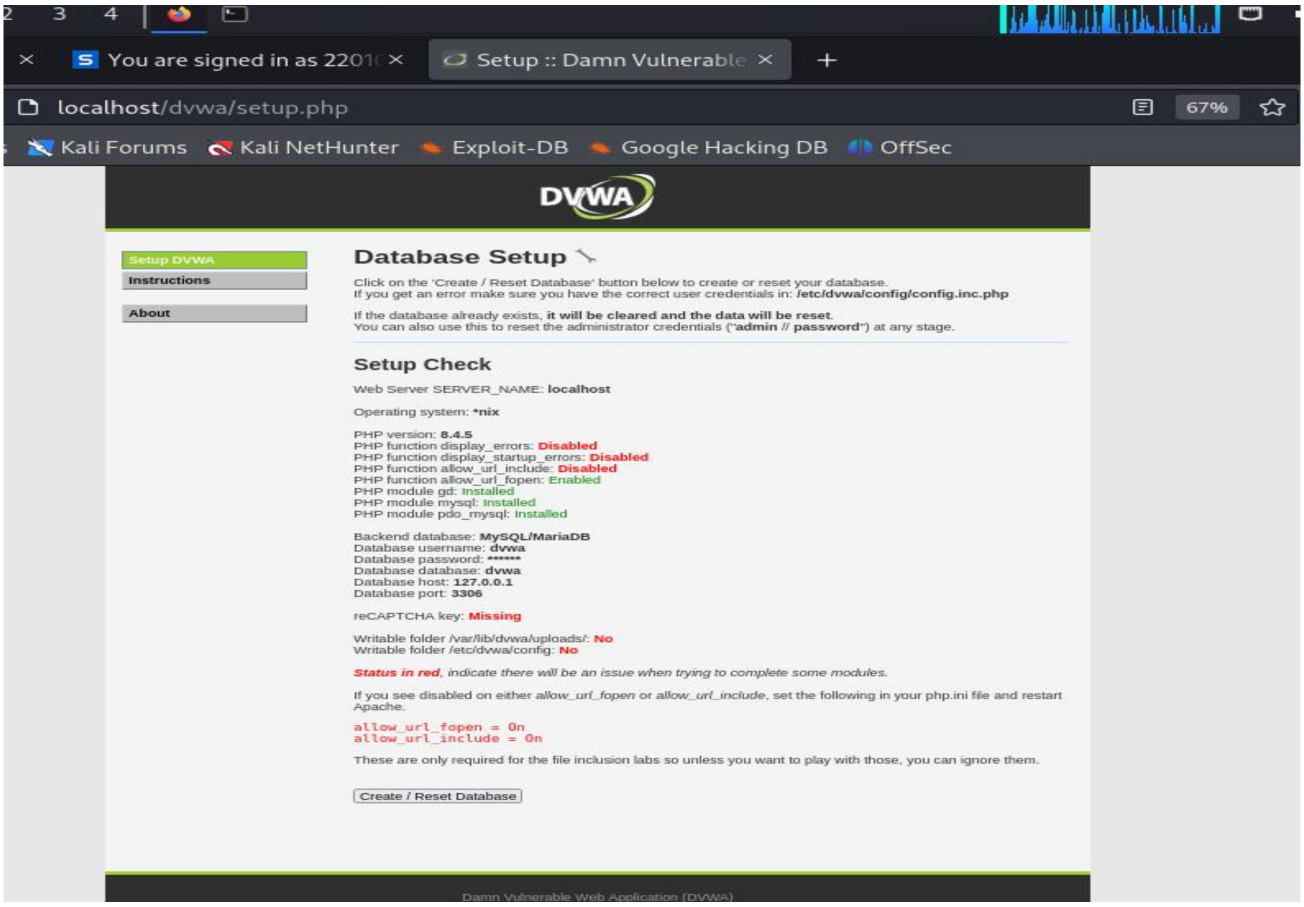
- Start services:

```
sudo service apache2 start
sudo service mysql start
```



```
┌──(root㉿Jocker)-[/home/jocker]
└─# sudo service apache2 start
sudo service mysql start
```

- Now you can try to open “dvwa” in browser.



Database Setup

Click on the 'Create / Reset Database' button below to create or reset your database.
If you get an error make sure you have the correct user credentials in: **/etc/dvwa/config/config.inc.php**

If the database already exists, **it will be cleared and the data will be reset.**
You can also use this to reset the administrator credentials ("**admin // password**") at any stage.

Setup Check

Web Server SERVER_NAME: **localhost**

Operating system: ***nix**

PHP version: **8.4.5**
PHP function display_errors: **Disabled**
PHP function display_startup_errors: **Disabled**
PHP function allow_url_include: **Disabled**
PHP function allow_url_fopen: Enabled
PHP module gd: Installed
PHP module mysql: Installed
PHP module pdo_mysql: Installed

Backend database: **MySQL/MariaDB**
Database username: **dvwa**
Database password: **********
Database database: **dvwa**
Database host: **127.0.0.1**
Database port: **3306**

reCAPTCHA key: **Missing**

Writable folder /var/lib/dvwa/uploads/: **No**
Writable folder /etc/dvwa/config: **No**

Status in red*, indicate there will be an issue when trying to complete some modules.*

If you see disabled on either *allow_url_fopen* or *allow_url_include*, set the following in your php.ini file and restart Apache.

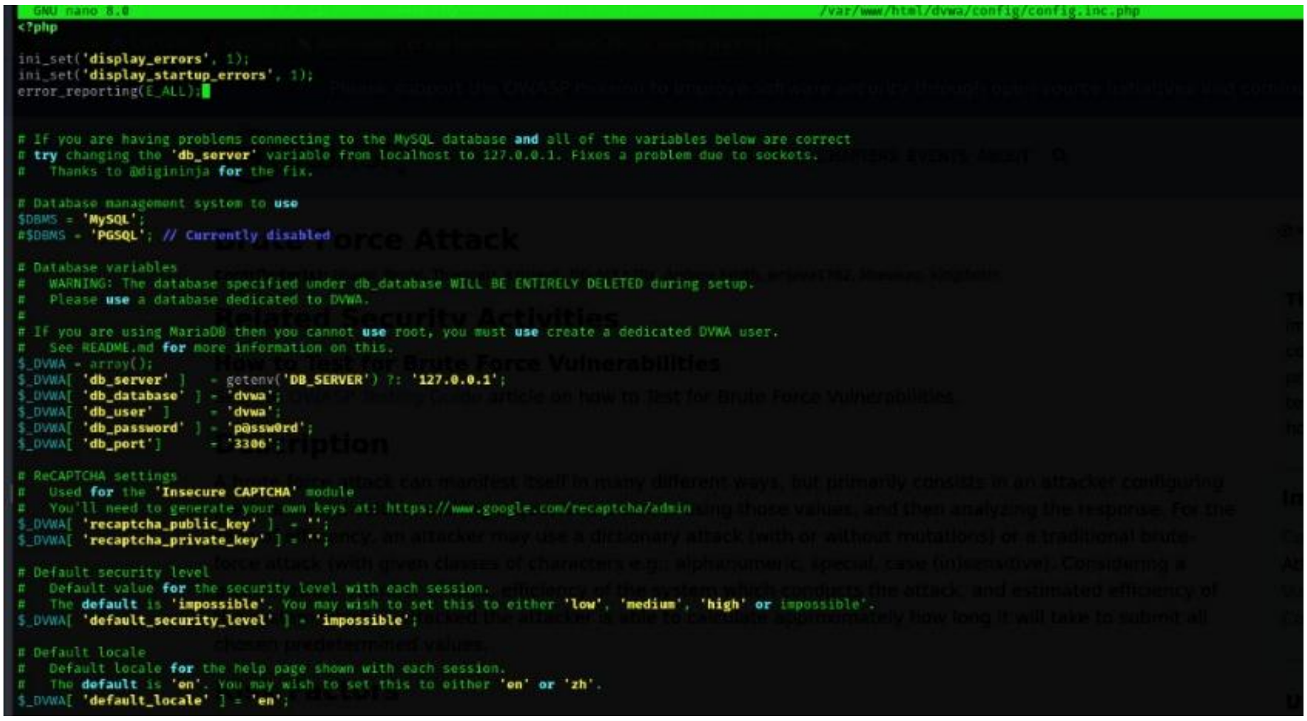
```
allow_url_fopen = On
allow_url_include = On
```

These are only required for the file inclusion labs so unless you want to play with those, you can ignore them.

- When you opening "http://localhost/dvwa/setup.php" and pressing on create or reset button to create the database then it will opening a blanck page. It means there's a **PHP error** or **MySQL issue** behind the scenes.

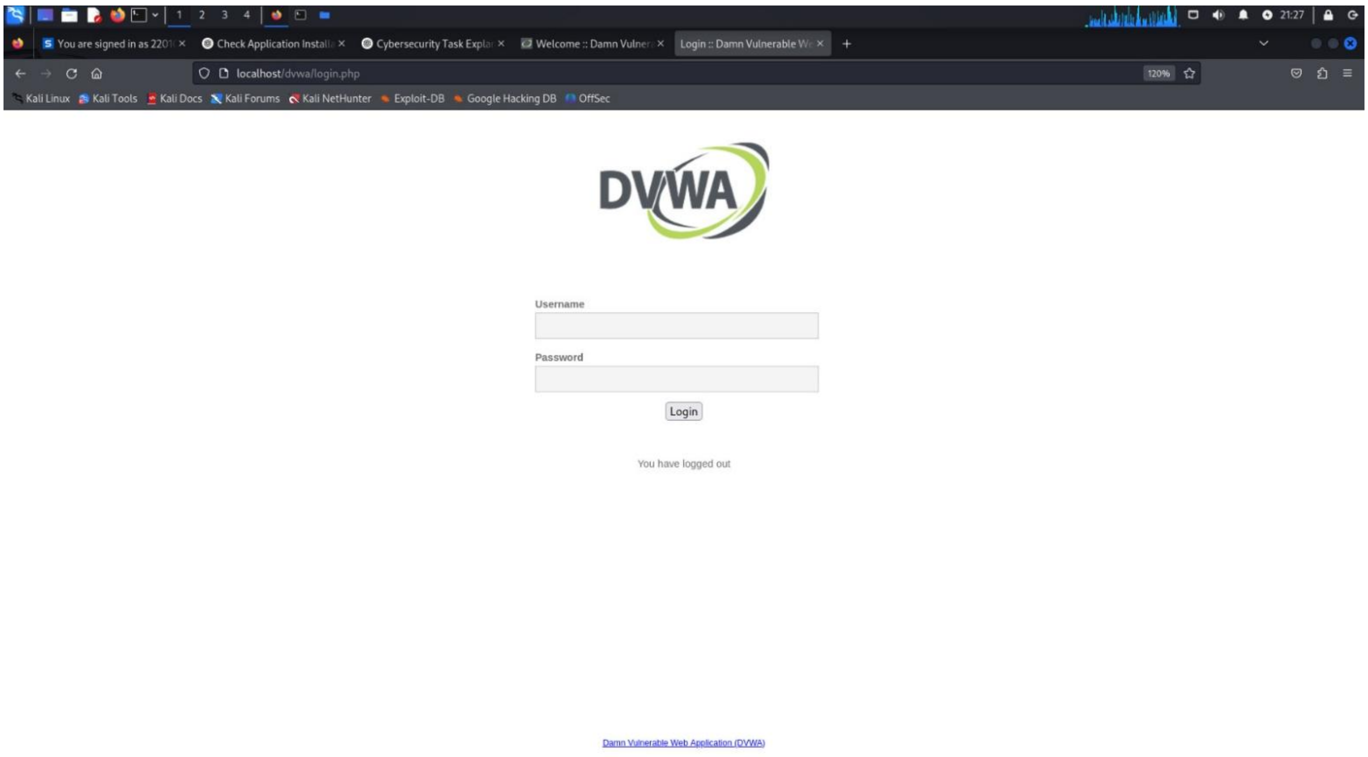
- Let’s troubleshoot it **step-by-step**

- **Enable Error Reporting in PHP**
  Open the DVWA config file:
  ```
  sudo nano /var/www/html/dvwa/config/config.inc.php
  ```



```
  GNU nano 8.0                                                        /var/www/html/dvwa/config/config.inc.php
<?php

ini_set('display_errors', 1);
ini_set('display_startup_errors', 1);
error_reporting(E_ALL);

# If you are having problems connecting to the MySQL database and all of the variables below are correct
# try changing the 'db_server' variable from localhost to 127.0.0.1. Fixes a problem due to sockets.
#   Thanks to @digininja for the fix.

# Database management system to use
$DBMS = 'MySQL';
#$DBMS = 'PGSQL'; // Currently disabled

# Database variables
#   WARNING: The database specified under db_database WILL BE ENTIRELY DELETED during setup.
#   Please use a database dedicated to DVWA.
#
# If you are using MariaDB then you cannot use root, you must use create a dedicated DVWA user.
#   See README.md for more information on this.
$_DVWA = array();
$_DVWA[ 'db_server' ]   = getenv('DB_SERVER') ?: '127.0.0.1';
$_DVWA[ 'db_database' ] = 'dvwa';
$_DVWA[ 'db_user' ]     = 'dvwa';
$_DVWA[ 'db_password' ] = 'p@ssw0rd';
$_DVWA[ 'db_port']      = '3306';

# ReCAPTCHA settings
#   Used for the 'Insecure CAPTCHA' module
#   You'll need to generate your own keys at: https://www.google.com/recaptcha/admin
$_DVWA[ 'recaptcha_public_key' ]  = '';
$_DVWA[ 'recaptcha_private_key' ] = '';

# Default security level
#   Default value for the security level with each session.
#   The default is 'impossible'. You may wish to set this to either 'low', 'medium', 'high' or impossible'.
$_DVWA[ 'default_security_level' ] = 'impossible';

# Default locale
#   Default locale for the help page shown with each session.
#   The default is 'en'. You may wish to set this to either 'en' or 'zh'.
$_DVWA[ 'default_locale' ] = 'en';
```

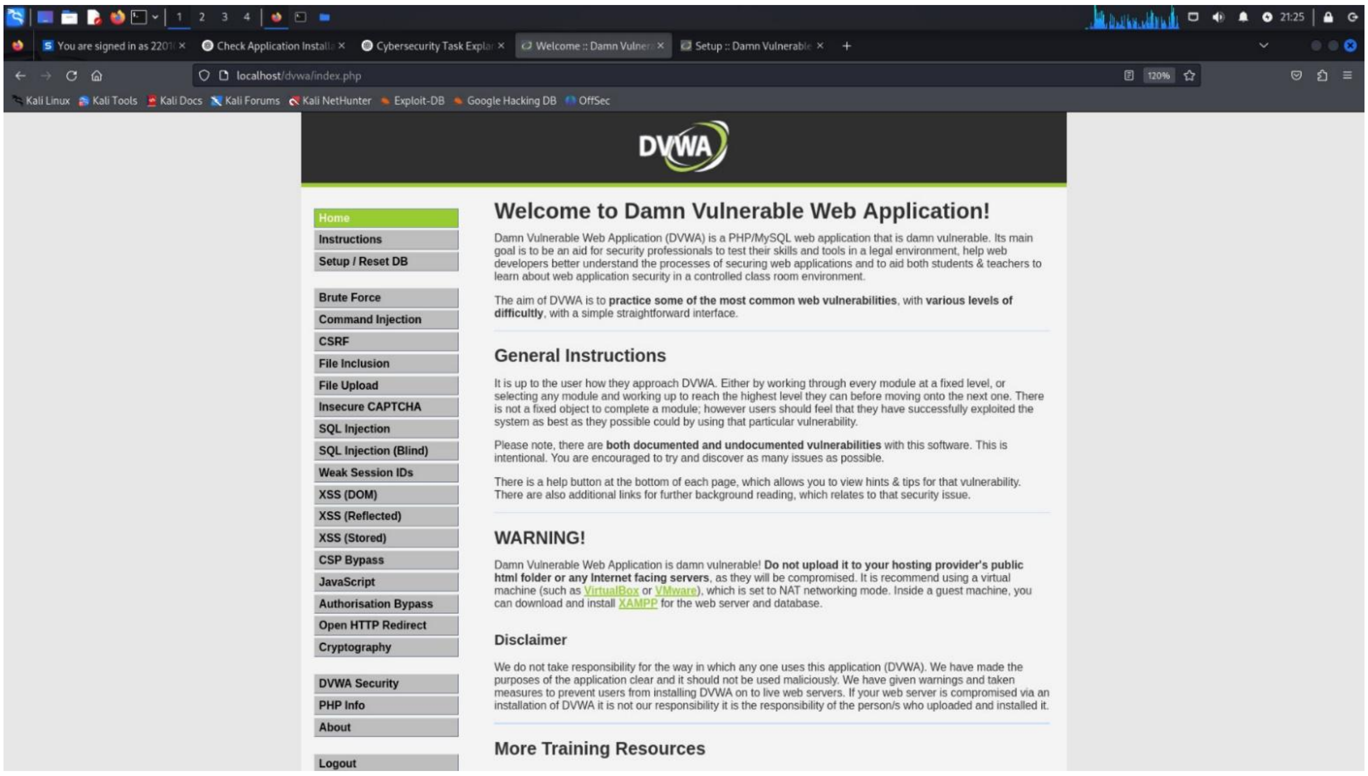
- Now restart the Mysql and Apache server
  ```
  sudo service apache2 restart
  sudo service mysql restart
  ```
- Now restart in the browser and now you can login with the help of user id and password
  USER-ID : admin
  Password: password



You have logged out

Damn Vulnerable Web Application (DVWA)

- After login you will reatched on home page of **DVWA**



Welcome to Damn Vulnerable Web Application!

Damn Vulnerable Web Application (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goal is to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and to aid both students & teachers to learn about web application security in a controlled class room environment.

The aim of DVWA is to **practice some of the most common web vulnerabilities**, with **various levels of difficultly**, with a simple straightforward interface.

General Instructions

It is up to the user how they approach DVWA. Either by working through every module at a fixed level, or selecting any module and working up to reach the highest level they can before moving onto the next one. There is not a fixed object to complete a module; however users should feel that they have successfully exploited the system as best as they possible could by using that particular vulnerability.

Please note, there are **both documented and undocumented vulnerabilities** with this software. This is intentional. You are encouraged to try and discover as many issues as possible.

There is a help button at the bottom of each page, which allows you to view hints & tips for that vulnerability. There are also additional links for further background reading, which relates to that security issue.

WARNING!

Damn Vulnerable Web Application is damn vulnerable! **Do not upload it to your hosting provider's public html folder or any Internet facing servers**, as they will be compromised. It is recommend using a virtual machine (such as VirtualBox or VMware), which is set to NAT networking mode. Inside a guest machine, you can download and install XAMPP for the web server and database.

Disclaimer

We do not take responsibility for the way in which any one uses this application (DVWA). We have made the purposes of the application clear and it should not be used maliciously. We have given warnings and taken measures to prevent users from installing DVWA on to live web servers. If your web server is compromised via an installation of DVWA it is not our responsibility it is the responsibility of the person/s who uploaded and installed it.

More Training Resources

- Now you can **Choose Vulnerabilities to Explore**
- DVWA gives multiple sections (modules) on the left sidebar.
    - Command Injection
    - SQL Injection
    - XSS (Reflected/Stored)

➢ **Step 2: Set Security Level to Low**
Go to:
- **DVWA Security → Set Security Level → Low → Submit**

Why?
Low makes the vulnerabilities easy to exploit. Once We succeed here, We can try Medium or High for learning later.

1. Reflected Cross-Site Scripting (XSS)

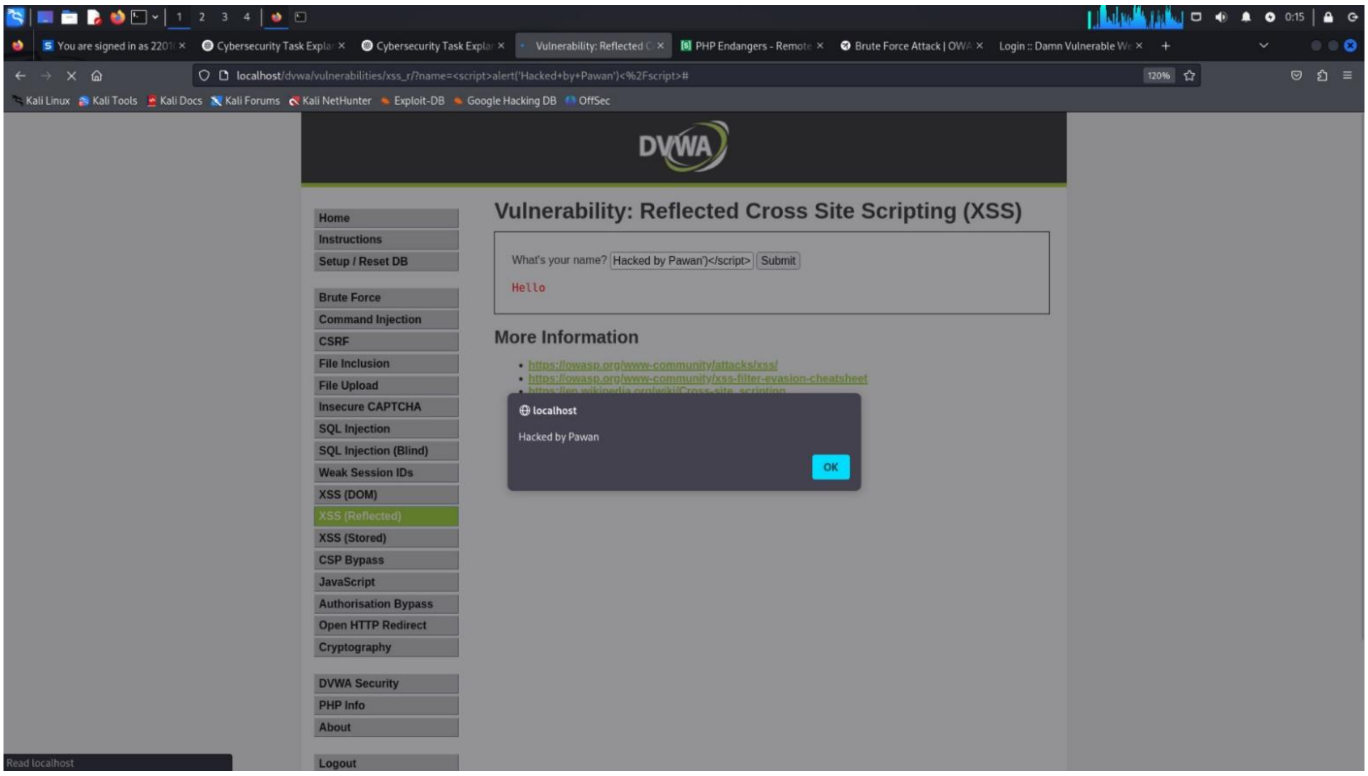
Concept: Reflected XSS occurs when user input is immediately returned by the application without proper sanitization.

DVWA Location: XSS (Reflected)

Payload:

```
<script>alert('XSS by Pawan')</script>
```

Result: Alert box displays. Indicates the script is being executed in the browser.



localhost/dvwa/vulnerabilities/xss_r/?name=<script>alert('Hacked+by+Pawan')<%2Fscript>#

Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name? Hacked by Pawan')</script> Submit

Hello

More Information

- https://owasp.org/www-community/attacks/xss/
- https://owasp.org/www-community/xss-filter-evasion-cheatsheet
- https://en.wikipedia.org/wiki/Cross-site_scripting

localhost

Hacked by Pawan

OK

Real-world Use:

- Stealing cookies using document.cookie
- Redirecting users to malicious sites

Mitigation:

- Input validation
- Output encoding
- CSP (Content Security Policy)

2. Stored Cross-Site Scripting (XSS)

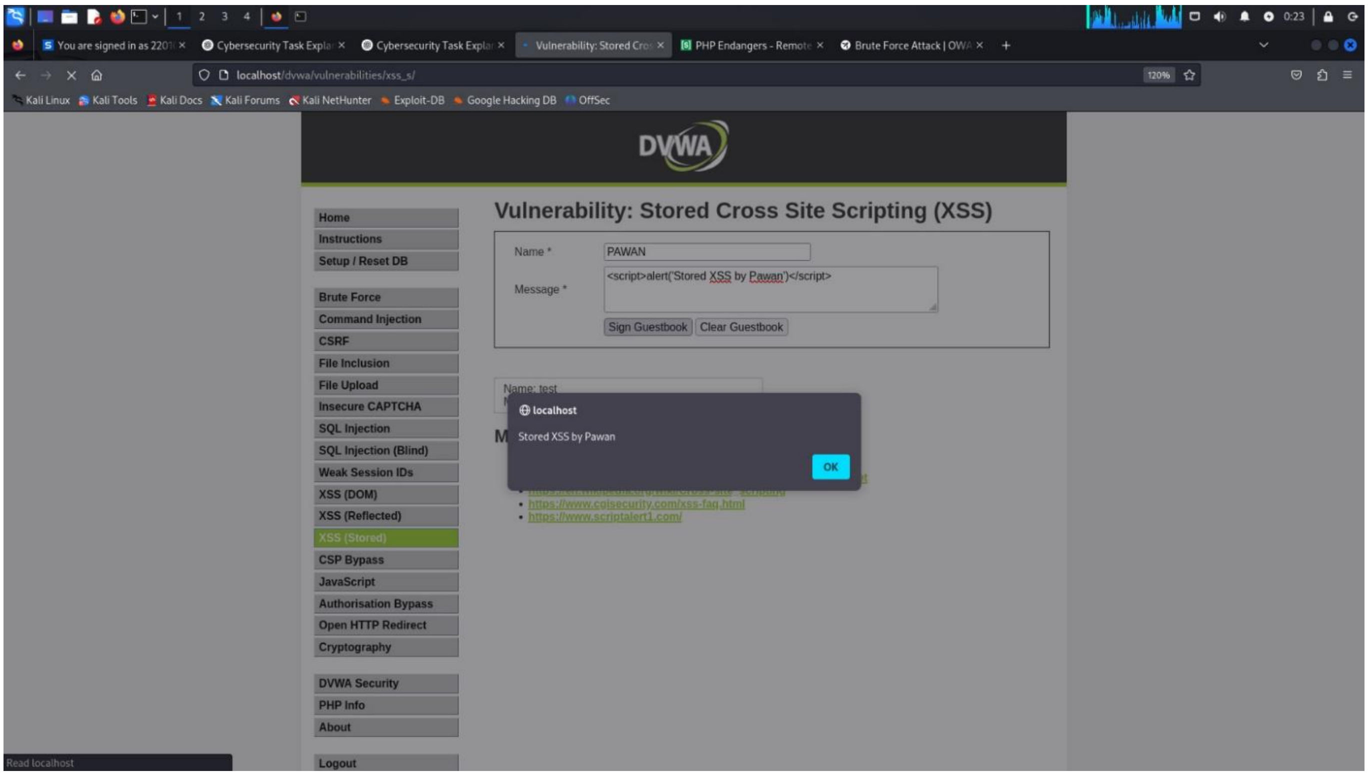
Concept: Stored XSS stores malicious scripts permanently on the server (e.g., in a database).

DVWA Location: XSS (Stored)

Payload:

```
<script>alert('Stored XSS by Pawan')</script>
```

Result: Alert pops up each time the page loads, affecting every viewer.



Vulnerability: Stored Cross Site Scripting (XSS)

Name * PAWAN

Message * <script>alert('Stored XSS by Pawan')</script>

Name: test

localhost

Stored XSS by Pawan

- https://www.cgisecurity.com/xss-faq.html
- https://www.scriptalert1.com/

Real-world Use:

- Persistent malware distribution
- Credential theft via form manipulation

Mitigation:

- Sanitize inputs before storing
- Use encoding when displaying

Real-life Example:

- An attacker posts a comment with malicious JS — every user visiting the comments section gets infected.

<u>3. Command Injection:</u>

Concept: Takes unsanitized user input and runs it as an OS command.

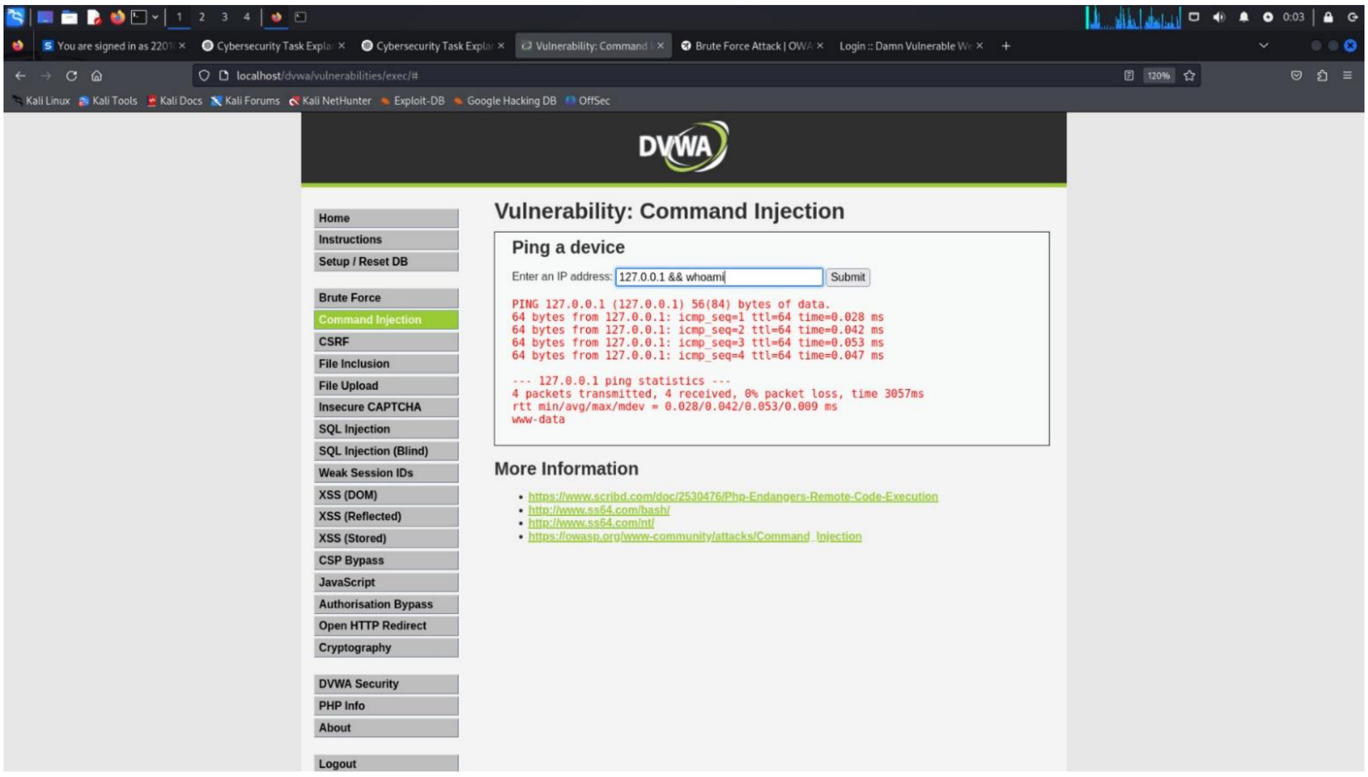
DVWA Location: Command Injection

Payload:

```
127.0.0.1 && whoami
```

Result:

The injected command whoami reveals the user under which the web server is running.


Vulnerability: Command Injection

Ping a device

Enter an IP address: 127.0.0.1 && whoami

```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.028 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.042 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.053 ms
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.047 ms

--- 127.0.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3057ms
rtt min/avg/max/mdev = 0.028/0.042/0.053/0.009 ms
www-data
```

More Information

- https://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution
- http://www.ss64.com/bash/
- http://www.ss64.com/nt/
- https://owasp.org/www-community/attacks/Command_Injection

Real-world Use:

- Remote Code Execution
- Full system compromise

Mitigation:

- Avoid passing user input to system commands
- Use safe APIs

4. SQL Injection

Concept: Injecting SQL queries to manipulate backend databases.

DVWA Location: SQL Injection

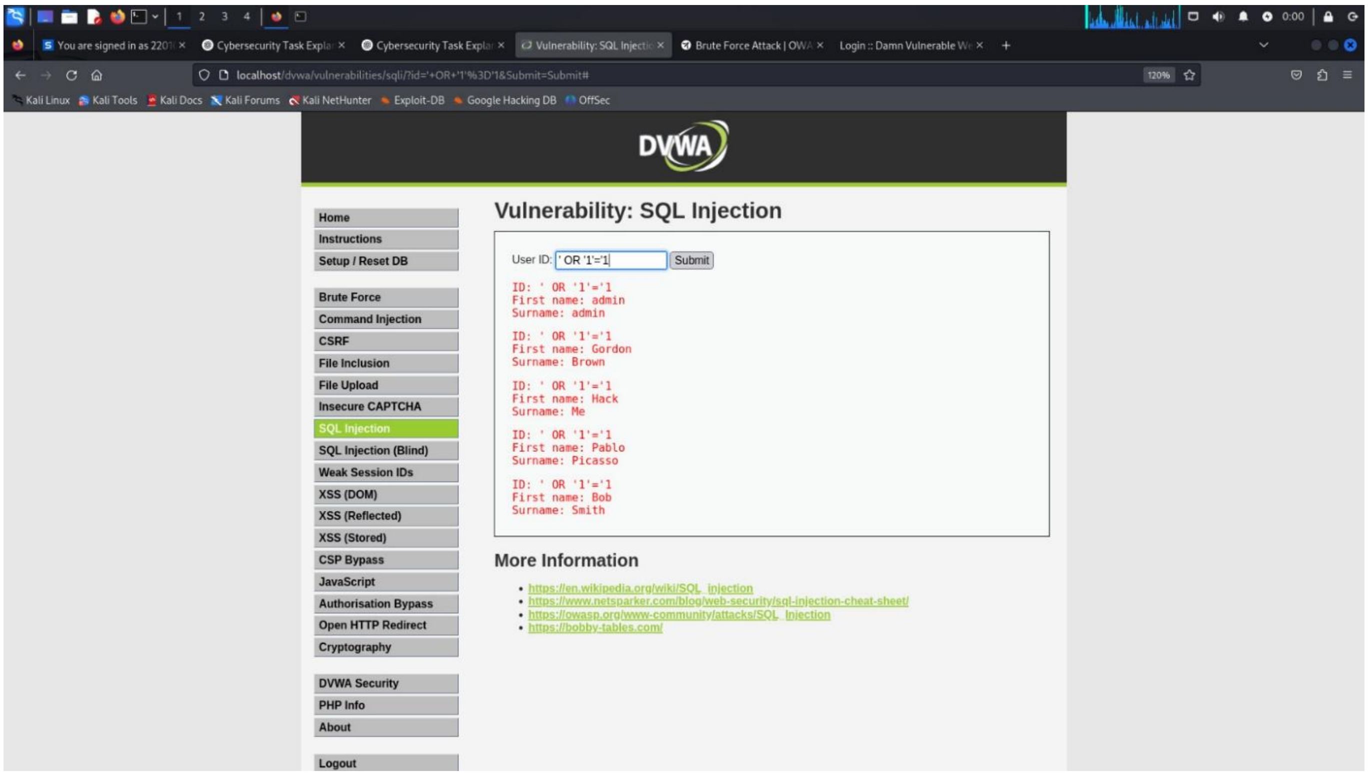
Payload:

```
' OR '1'='1
```

Blind SQLi Test:

```
1' AND SLEEP(5)-- -
```

Result:



Vulnerability: SQL Injection

User ID: ' OR '1'='1

```
ID: ' OR '1'='1
First name: admin
Surname: admin

ID: ' OR '1'='1
First name: Gordon
Surname: Brown

ID: ' OR '1'='1
First name: Hack
Surname: Me

ID: ' OR '1'='1
First name: Pablo
Surname: Picasso

ID: ' OR '1'='1
First name: Bob
Surname: Smith
```

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/
- https://owasp.org/www-community/attacks/SQL_Injection
- https://bobby-tables.com/

Real-world Use:

- Bypass authentication
- Dump database

Mitigation:

- Use prepared statements (parameterized queries)
- Input validation

Conclusion:

These documented vulnerabilities demonstrate real-world attack techniques that can exploit poorly coded web applications. Understanding these attacks is essential for cybersecurity professionals to recognize threats, test defenses, and implement secure coding practices.