Vulnerability pentesting on DVWA:

Website: http://localhost/dvwa/

Tools used: Acunetix

Host the DVWA website on local machine.

Download and install Acunetix and give the host name as the above website and click on full scan.
It'll show the results with available vulnerabilities (If any).

Vulnerabilities found:

1) **Configuration file source code disclosure:**

A backup/temporary configuration file was found on this directory. It has been confirmed that this file contains PHP source code.
Several popular text editors like Vim and Emacs automatically create backup copies of the files you edit, giving them names like "wp-config.php~" and "#wp-config.php#". If the text editor crashes or the SSH connection drops during editing, then the temporary backup files may not be cleaned up correctly. Also, sometimes developers create this type of files to backup their work or by administrators when making backups of the web server. Most servers, including Apache, will serve the plaintext of .php~ and .php# files without passing them through the PHP preprocessor first, since they don't have the .php file extension.

Remediation:

Remove this file from the web server. As an additional step, it is recommended to implement a security policy within your organization to disallow creation of temporary/backup files in directories accessible from the web.:

2) User credentials are sent in clear text

User credentials are transmitted over an unencrypted channel. This information should always be transferred via an encrypted channel (HTTPS) to avoid being intercepted by malicious users.

A third party may be able to read the user credentials by intercepting an unencrypted HTTP connection.

Remediation:

Because user credentials are considered sensitive information, should always be transferred to the server over an encrypted connection (HTTPS).

3) Possible sensitive files exposure

A possible sensitive file has been found. This file is not directly linked from the website. This check looks for common sensitive resources like password files, configuration files, log files, include files, statistics data, database dumps. Each one of these files could help an attacker to learn more about his target.

Remediation:

This file may expose sensitive information that could help a malicious user to prepare more advanced attacks.

4) Login page password-guessing attack

A common threat web developers face is a password-guessing attack known as a brute force attack. A brute-force attack is an attempt to discover a password by systematically trying every possible combination of letters, numbers, and symbols until you discover the one correct combination that works.

This login page doesn't have any protection against password-guessing attacks (brute force attacks). It's recommended to implement some type of account lockout after a defined number of incorrect password attempts.

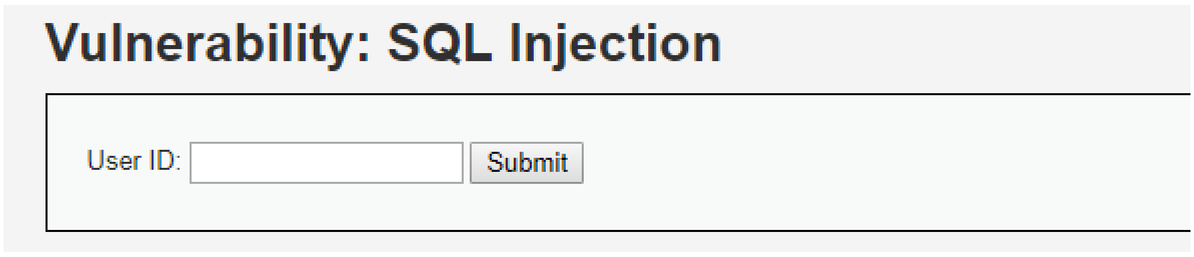
An attacker may attempt to discover a weak password by systematically trying every possible combination of letters, numbers, and symbols until it discovers the one correct combination that works.

Remediation:

It's recommended to implement some type of account lockout after a defined number of incorrect password attempts.

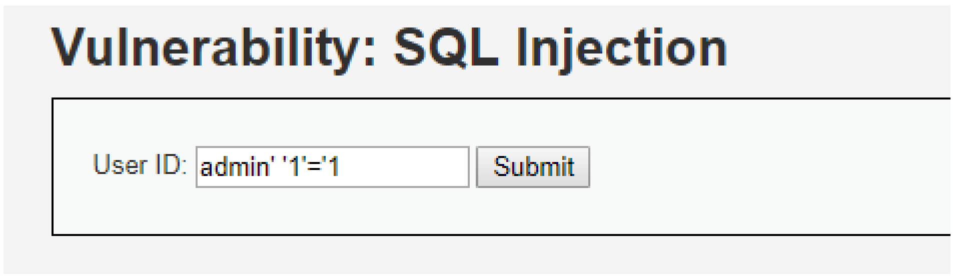
5) SQL injection vulnerability.

->To perform SQL injection go to SQL injection on the left pane in DVWA. It’ll open another page asking to type the user id.


Vulnerability: SQL Injection

User ID:

-> We will perform SQL injection on this page.
-> Type the below mentioned SQLi in order to bypass the login and click on submit.


Vulnerability: SQL Injection

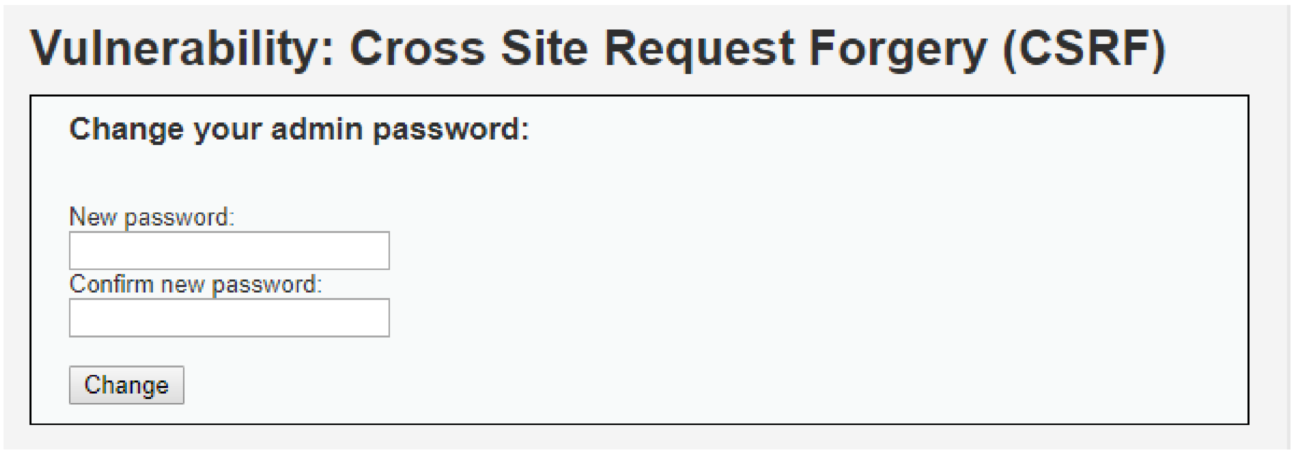
User ID: admin' '1'='1

-> After hitting submit you’ll find that you have logged in without proper authentication.

Remediation:

Use of prepared statements with parameterized queries is a strong control to mitigate an attack. Instead of writing dynamic queries—which fails to differentiate between application code and data—prepared statements force developers to use static SQL query and then pass in the external input as a parameter to query. This approach ensures the SQL interpreter always differentiates between code and data.

6) CSRF vulnerability.
Navigate to CSRF page in DVWA


Vulnerability: Cross Site Request Forgery (CSRF)

Change your admin password:

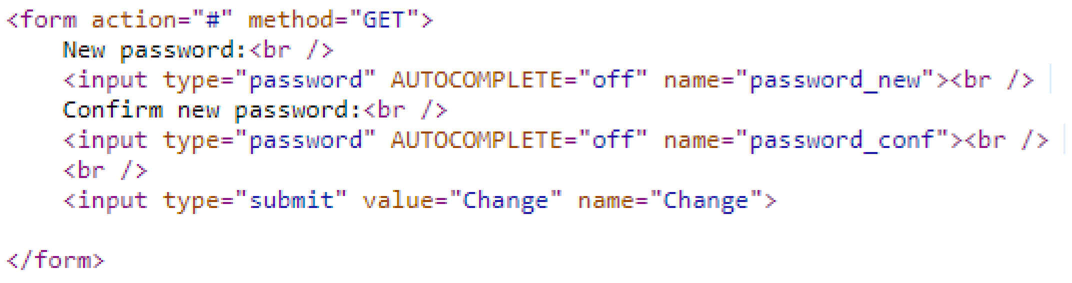
New password:

Confirm new password:

Step 1: Right click anywhere on the screen and click on page source.

Step 2: Navigate to the form tag and copy paste the below code to another file and save it as .html

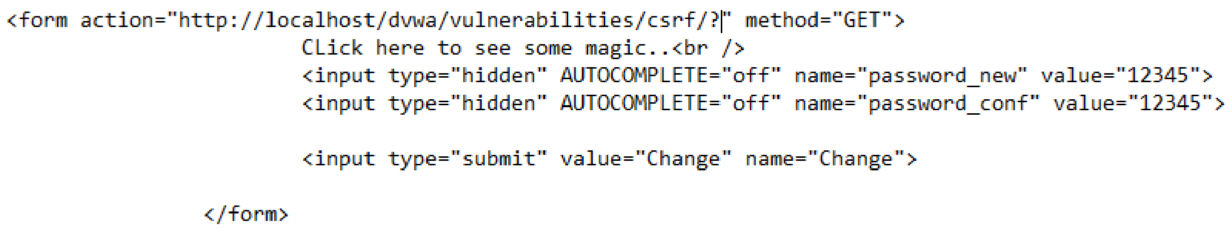
Step 3: Now we will make some changes to the code so that it'll automatically change the password without any user's interference.



```
<form action="#" method="GET">
    New password:<br />
    <input type="password" AUTOCOMPLETE="off" name="password_new"><br />
    Confirm new password:<br />
    <input type="password" AUTOCOMPLETE="off" name="password_conf"><br />
    <br />
    <input type="submit" value="Change" name="Change">

</form>
```

Step 4: Make some changes to the saved code to look something like below.



```
<form action="http://localhost/dvwa/vulnerabilities/csrf/?" method="GET">
                        CLick here to see some magic..<br />
                        <input type="hidden" AUTOCOMPLETE="off" name="password_new" value="12345">
                        <input type="hidden" AUTOCOMPLETE="off" name="password_conf" value="12345">

                        <input type="submit" value="Change" name="Change">

                </form>
```

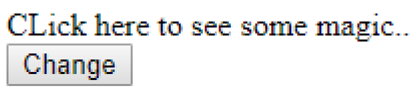
Step 5: Here we have changed the action in the form tag to the URL which will redirect the website to csrf page. We changed the type from password to hidden because we don't want the user to know what's going on.

We added another parameter called value where we declared it with some word or number. Note this value will act as the new password for the CSRF page.

In the same way we changed the value for the next line too

Now save the file with .html format.

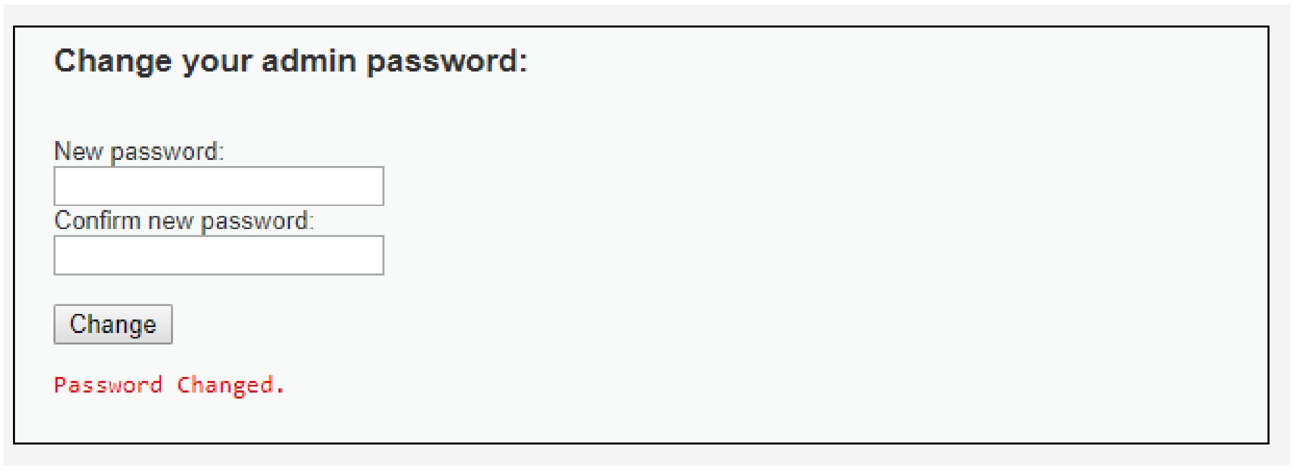
Step 6: Open the saved file in web browser and you'll find something like this.



CLick here to see some magic..

Step 7: When you click on the change button it automatically changes the password of our DVWA to the new password that we mentioned in that text file.

Step 8: New screen opens saying that the password has been changed.


Change your admin password:

New password:

Confirm new password:

Password Changed.

Remediation:

The most common method to prevent Cross-Site Request Forgery (CSRF) attacks is to append CSRF tokens to each request and associate them with the user's session. Such tokens should at a minimum be unique per user session, but can also be unique per request. By including a challenge token with each request, the developer can ensure that the request is valid and not coming from a source other than the user.

The easiest way to check whether an application is vulnerable is to see if each link and form contains an unpredictable token for each user. Without such an unpredictable token, attackers can forge malicious requests. Focus on the links and forms that invoke state-changing functions, since those are the most important CSRF targets.