Assignment Report

DVWA

Prepared by:**R Deepak kumar.**

Date: **11/03/2024**

1. Introduction :

The purpose of this report is to document the assessment, exploitation, and mitigation of vulnerabilities within the Damn Vulnerable Web Application (DVWA). This assignment aims to enhance understanding of common web vulnerabilities and their exploitation techniques, as well as propose effective mitigation strategies to address these issues.

2. Methodology:

2.1 Setup DVWA

The DVWA instance was successfully installed and configured in the local environment. Different security levels and functionalities were explored to gain familiarity with the application.

2.2 Familiarization

The DVWA interface was thoroughly examined, focusing on various functionalities and vulnerable components. Understanding the security levels and their impact on the application's security was a key aspect of this phase.

3. Vulnerability Assessment Results

A comprehensive vulnerability assessment was conducted on DVWA, revealing the following vulnerabilities:

3.1 SQL Injection :

Medium Security Level :

Vulnerability: SQL Injection

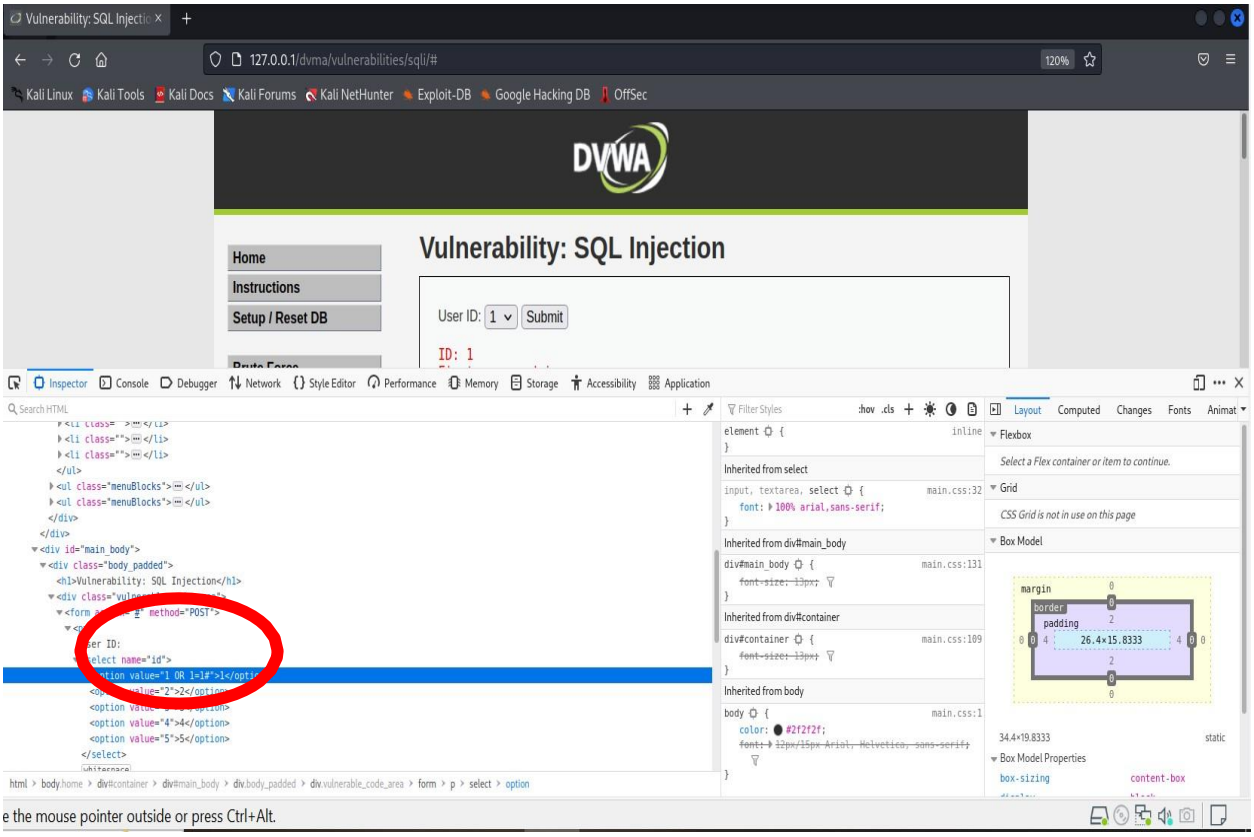
Description: The input fields of the application lack proper input validation and sanitization, allowing for SQL Injection attacks.

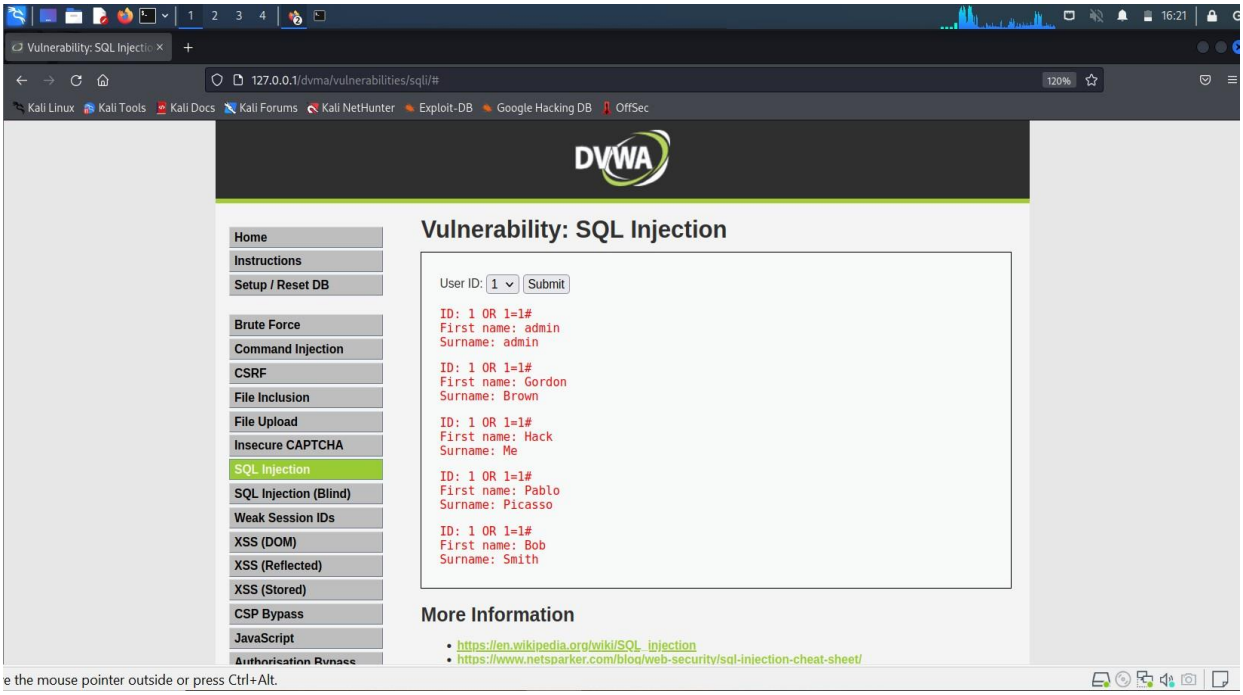
Severity: **Medium**

Impact: Successful exploitation could lead to unauthorized access to sensitive data and potentially compromise the confidentiality of user information.

Exploitation Technique: Altered the value through browser inspect by setting (**OR 1=1#**) to manipulate the application's database. This resulted in the extraction of sensitive information such as user credentials.

Proof of concept :





High Security Level:

Vulnerability: SQL Injection

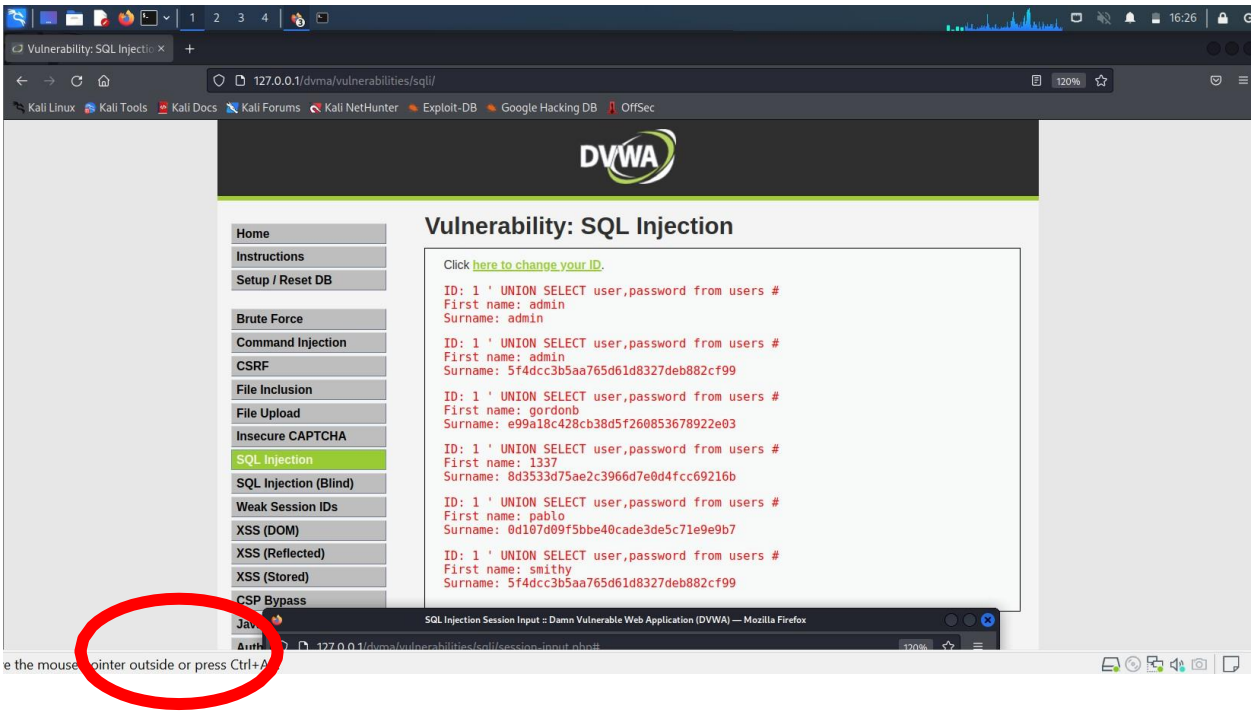
Description: Despite the higher security level, certain input fields remain susceptible to SQL Injection attacks due to insufficient input validation.

Severity: **High**

Impact: Successful exploitation poses a severe risk, potentially leading to unauthorized access, data manipulation, and potential compromise of the application's integrity.

Exploitation Technique: Entered the value (**1 ' UNION SELECT User, password from users #**) directly into the input field. This leveraged union-based SQL Injection to retrieve data from additional database tables.

Proof of concept :



Exploitation Techniques:

Medium Security Level

- Manual SQL Injection:

Utilized crafted SQL queries through browser inspect to manipulate the application's database.

Extracted sensitive information such as user credentials.

High Security Level

- Union-Based SQL Injection:

Exploited the lack of input validation by entering a crafted SQL query directly into the input field.

Successfully retrieved data from additional database tables.

Mitigation Strategies:

Medium and High Security Levels

Input Validation and Parameterized Queries:

- Implement strict input validation to sanitize user inputs effectively.
- Use parameterized queries to prevent SQL Injection attacks.

Least Privilege Principle:

- Restrict database user privileges to the minimum necessary for application functionality.
- Avoid using database users with excessive permissions.

3.2 Cross-Site Scripting (XSS) :

Exploitation Techniques

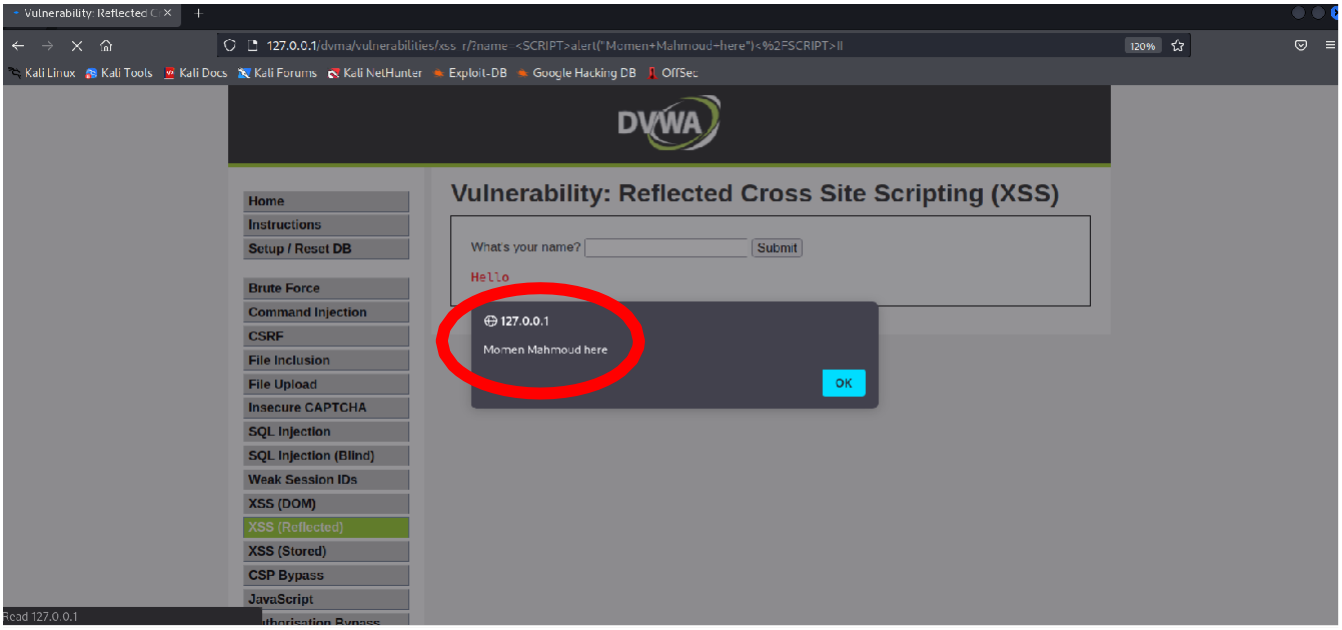
Medium Security Level:

XSS Attack:

Injected the script **<SCRIPT>alert("Momen Mahmoud here")</SCRIPT>** into input fields.

Successfully executed arbitrary JavaScript code within the application, demonstrating the vulnerability.

Proof of concept :



Mitigation Strategies:

Medium Security Level

Input Validation and Output Encoding:

- Implement rigorous input validation to prevent the injection of malicious scripts.
- Employ output encoding to sanitize user inputs before rendering them on the web page.

Content Security Policy (CSP):

- Utilize Content Security Policy headers to control the sources from which certain content can be loaded.
- Restrict the execution of inline scripts to mitigate XSS risks.

Cross-Site Scripting (XSS) - DOM-Based

Low Security Level:

Description: The application at the low security level is susceptible to DOM-Based XSS attacks due to insufficient input validation.

Severity: Low

Impact: Successful exploitation allows an attacker to inject and execute arbitrary JavaScript code in the context of the user's browser, leading to potential unauthorized actions.

Exploitation Technique: Manipulated the URL parameter by injecting the script <script>alert("Momen Mahmoud here")</script> to execute arbitrary JavaScript code within the application.

Exploitation Techniques

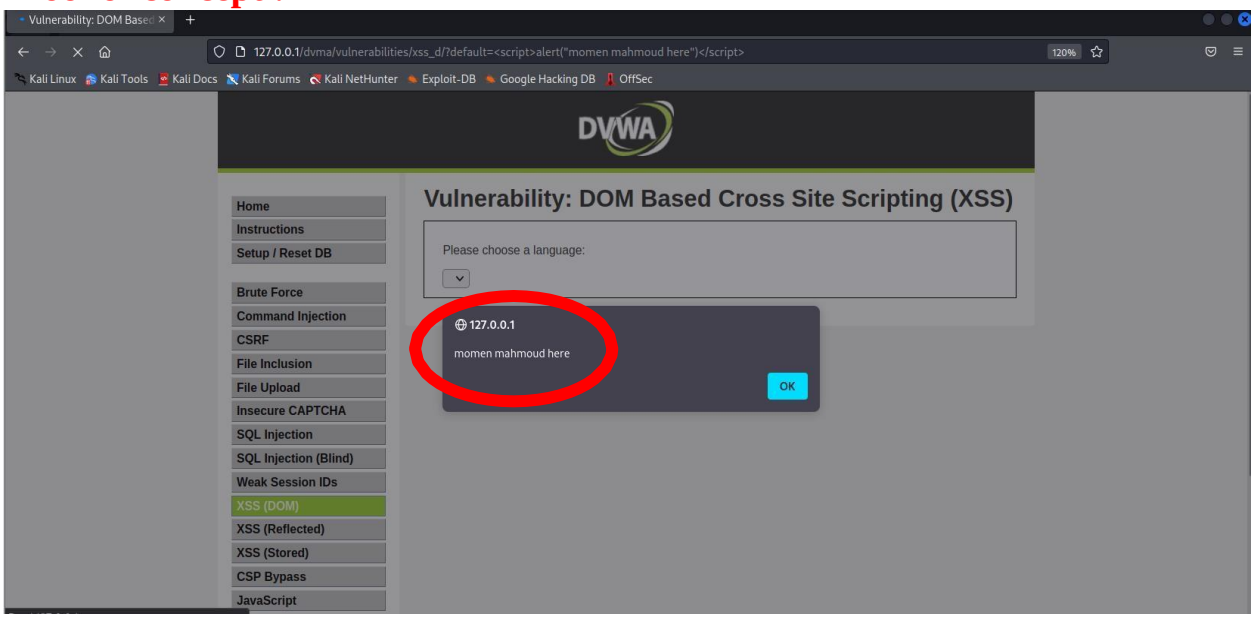
Low Security Level

DOM-Based XSS Attack:

Injected the script <script>alert("Momen Mahmoud here")</script> into the URL parameter.

Successfully executed arbitrary JavaScript code within the application, demonstrating the vulnerability.

Proof of concept :



Mitigation Strategies

Input Validation and Output Encoding:

- Implement rigorous input validation to prevent the injection of malicious scripts.
- Use output encoding to sanitize user inputs before rendering them on the web page.

Content Security Policy (CSP):

- Utilize Content Security Policy headers to control the sources from which certain content can be loaded.
- Restrict the execution of inline scripts to mitigate XSS risks.