



XSS Reflected Attacks on DVWA

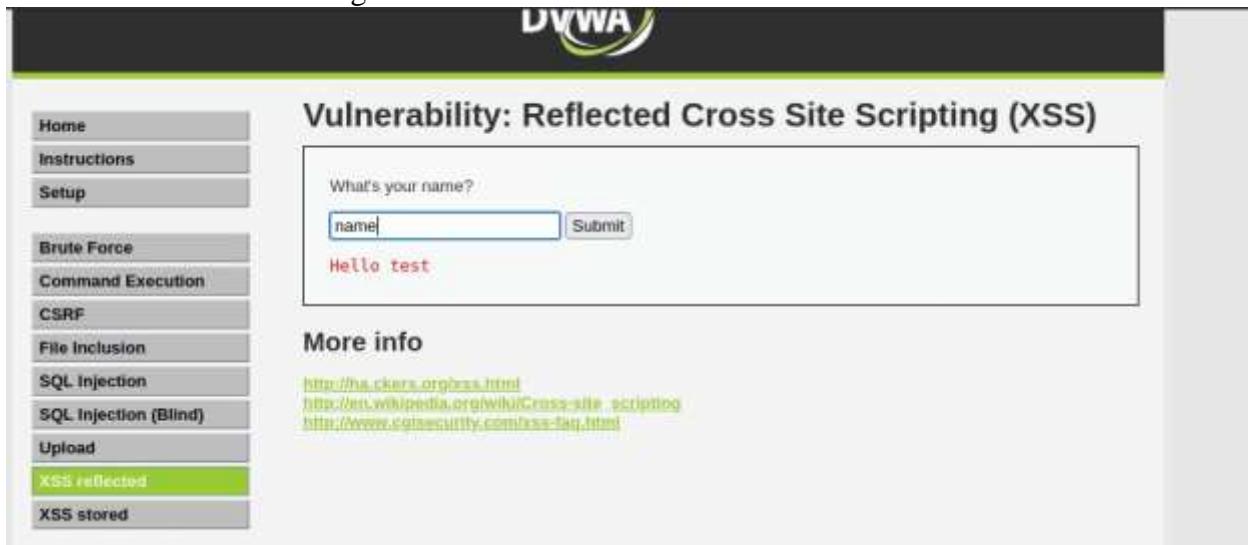
Prepared by Md Jobarul Islam

Md Jobarul Islam
Mirpur, Dhaka
email: jobarulislam1203@gmail.com
linkedIn: <https://www.linkedin.com/in/jobarulislam/>
GitHub: <https://github.com/jobarulislam>
Phone: 01312-678017

LogIn to the dvwa and select XSS reflected :

Low security XSS Reflected Attack

1st Observe how its working :

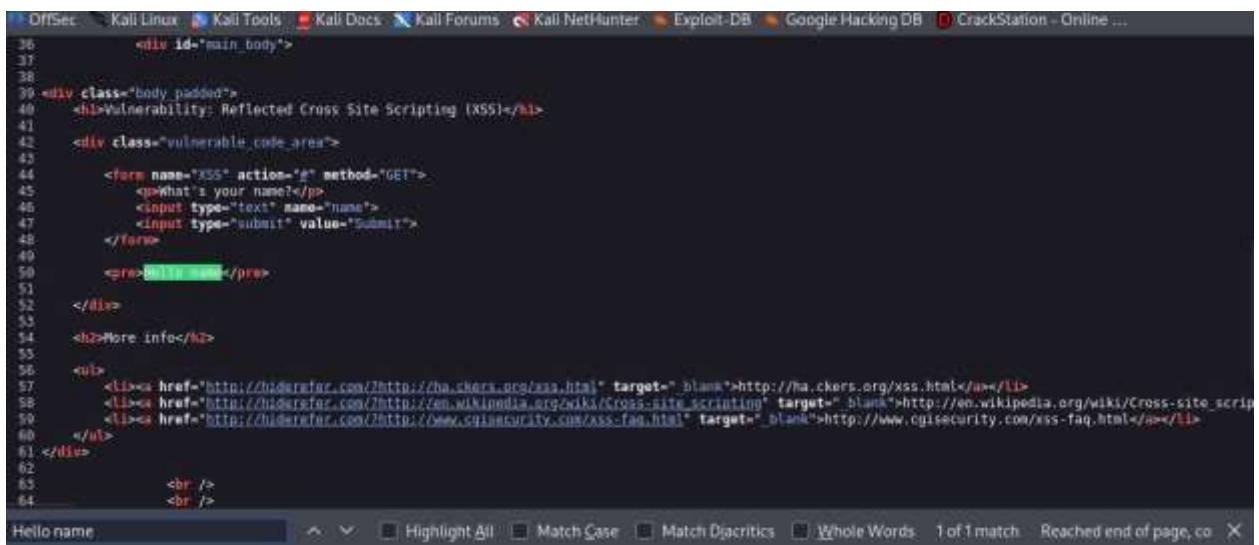


The screenshot shows the DVWA application interface. On the left, a sidebar menu lists various security modules: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected (which is highlighted in green), and XSS stored. The main content area has a title "Vulnerability: Reflected Cross Site Scripting (XSS)". Below the title is a form field labeled "What's your name?" with an input box containing "name" and a submit button. Underneath the form, the text "Hello test" is displayed in red, indicating the reflected XSS payload. A section titled "More info" provides links to external resources: <http://ha.ckers.org/xss.html>, http://en.wikipedia.org/wiki/Cross-site_scripting, and <http://www.cgisecurity.com/xss-faq.html>.

Use (Ctrl + u) for open source code:

(Ctrl + f) for search [hello reflected_text]

Back to Render window by (Ctrl + w)



The screenshot shows a terminal window displaying the source code of the XSS reflected attack page. The code is as follows:

```
OffSec Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB CrackStation - Online ...
36 <div id="main_body">
37
38
39 <div class="body_padded">
40   <h2>Vulnerability: Reflected Cross Site Scripting (XSS)</h2>
41
42   <div class="vulnerable_code_area">
43
44     <form name="XSS" action="/" method="GET">
45       <p>What's your name?</p>
46       <input type="text" name="name">
47       <input type="submit" value="Submit!">
48     </form>
49
50     <pre>Hello name</pre>
51
52   </div>
53
54   <h2>More Info</h2>
55
56   <ul>
57     <li><a href="http://ha.ckers.org/xss.html" target="_blank">http://ha.ckers.org/xss.html</a></li>
58     <li><a href="http://en.wikipedia.org/wiki/Cross-site_scripting" target="_blank">http://en.wikipedia.org/wiki/Cross-site_scripting</a></li>
59     <li><a href="http://www.cgisecurity.com/xss-faq.html" target="_blank">http://www.cgisecurity.com/xss-faq.html</a></li>
60   </ul>
61 </div>
62
63   <br />
64   <br />
```

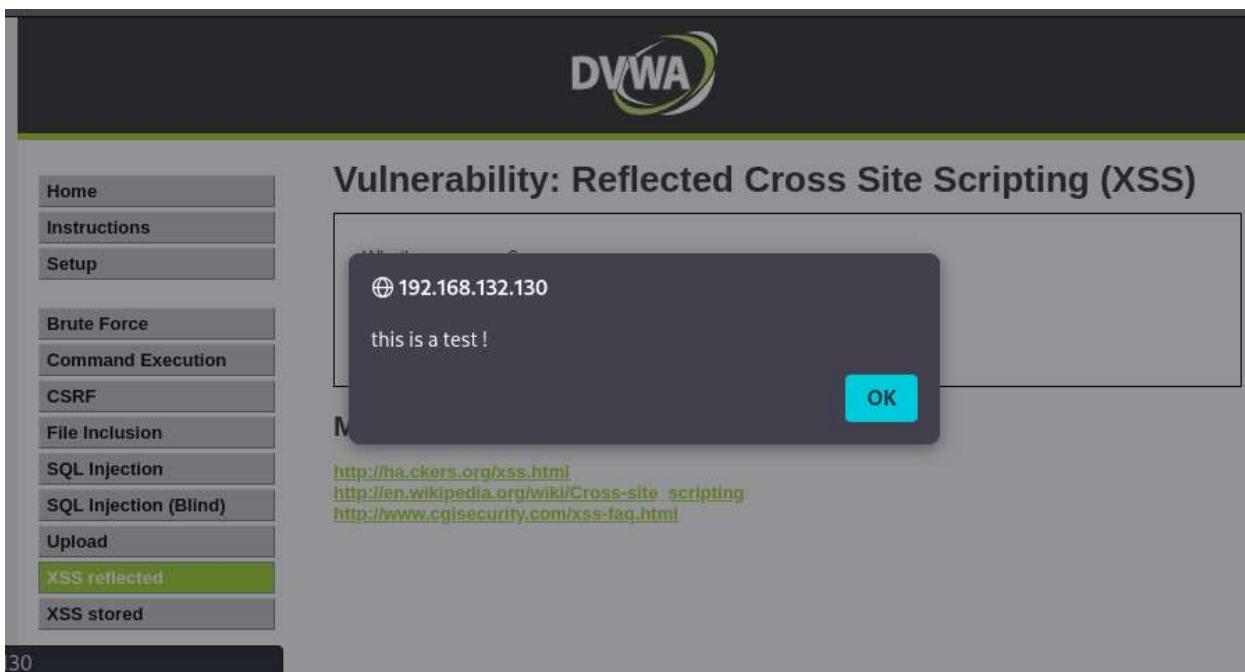
The terminal also shows a search result for "Hello name" in the current file, with 1 of 1 match found.

Submit [<script>alert("this is a test! ")</script>] code in input box for check for XSS valnareblity:



The screenshot shows the DVWA application interface. On the left, a sidebar menu lists various security vulnerabilities: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected (which is highlighted in green), XSS stored, and DVWA Security. The main content area is titled "Vulnerability: Reflected Cross Site Scripting (XSS)". It contains a form field labeled "What's your name?" with the value "<script>alert('your hacked!')</script>". Below the form is a red error message "Hello".

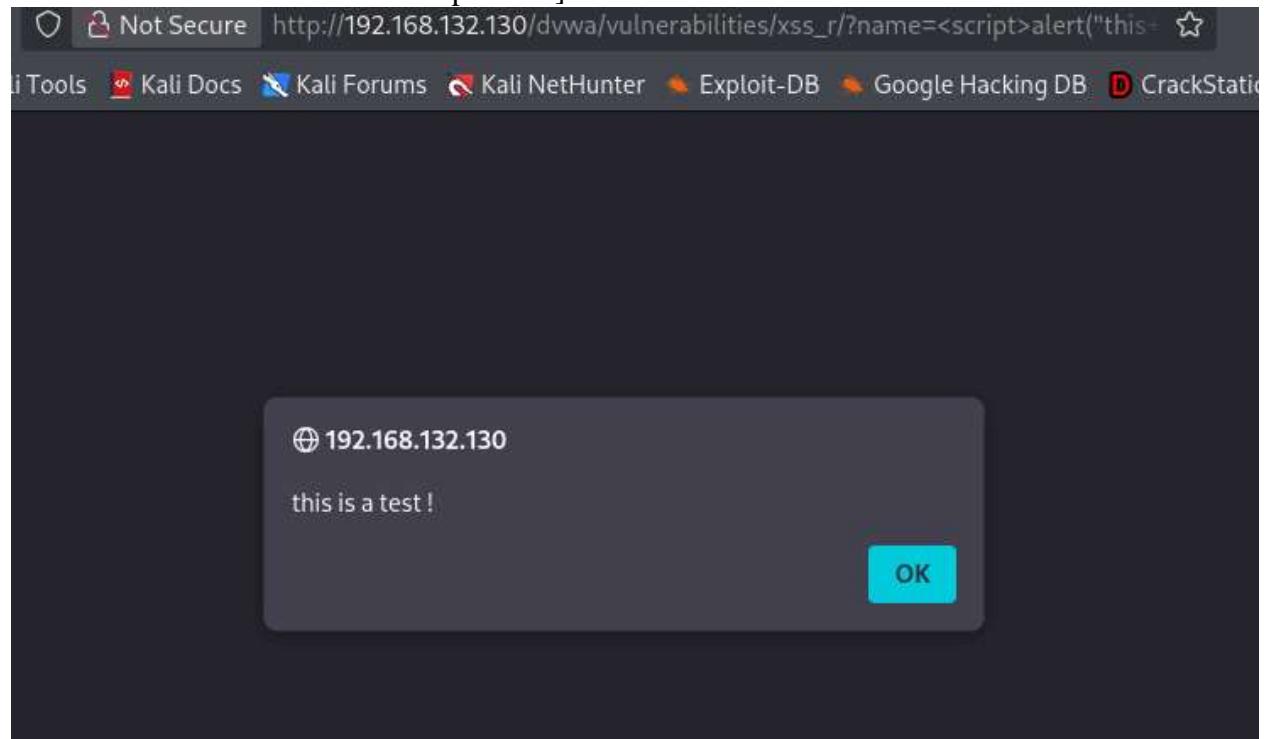
Code Inject work so this is vulnerable :



The screenshot shows the DVWA application interface. The sidebar menu is identical to the previous one. The main content area is titled "Vulnerability: Reflected Cross Site Scripting (XSS)". A modal dialog box is displayed, showing the IP address "192.168.132.130" and the message "this is a test!". An "OK" button is visible in the bottom right corner of the dialog.

Url of this page :

[http://192.168.132.130/dvwa/vulnerabilities/xss_r/?name=%3Cscript%3Ealert%28%22this+is+a+test%21%22%29%3C%2Fscript%3E#]



It seems that this page have vulnerability of XSS Reflected and I found it. So it's can be compromised by attacker.

Medium security XSS Reflected Attack

Submit security level low to medium:

A screenshot of the DVWA (Damn Vulnerable Web Application) interface. The top navigation bar includes "DVWA" with a gear icon, "Home", "Instructions", "Setup", "Brute Force", "Command Execution", "CSRF", "File Inclusion", "SQL Injection", "SQL Injection (Blind)", "Upload", "XSS reflected", and "XSS stored". The main content area is titled "DVWA Security" with a padlock icon. It shows "Script Security" with a note that the security level is currently "low". A dropdown menu shows "medium" selected, and a "Submit" button is nearby. Below this is a section titled "PHPIDS" with a note about PHPIDS v.0.6 being a security layer for PHP based web applications. It shows that PHPIDS is currently "disabled" and provides a link to "[enable PHPIDS]". At the bottom, there are links for "[Simulate attack]" and "[View IDS log]".

Its same as low level. Take look how the page work and the inject malicious code for check the vulnerability.

Code: [<script>alert("this is medium level test !")</script>] this time not working its do what its mean to be:

The screenshot shows the DVWA application interface. The left sidebar contains a navigation menu with various attack types: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected (which is highlighted in green), and XSS stored. The main content area is titled "Vulnerability: Reflected Cross Site Scripting (XSS)". It features a form with the placeholder "What's your name?" and a "Submit" button. Below the form, a red alert message is displayed: "Hello alert('this is medium level test !')".

Look for backend code/ source code : (Ctrl + u)

```
<div class="body_padded">
    <h1>Vulnerability: Reflected Cross Site Scripting (XSS)</h1>

    <div class="vulnerable_code_area">
        <form name="XSS" action="/" method="GET">
            <p>What's your name?</p>
            <input type="text" name="name">
            <input type="submit" value="Submit">
        </form>

        <pre>Hello alert("this is medium level test !")</pre>
    </div>

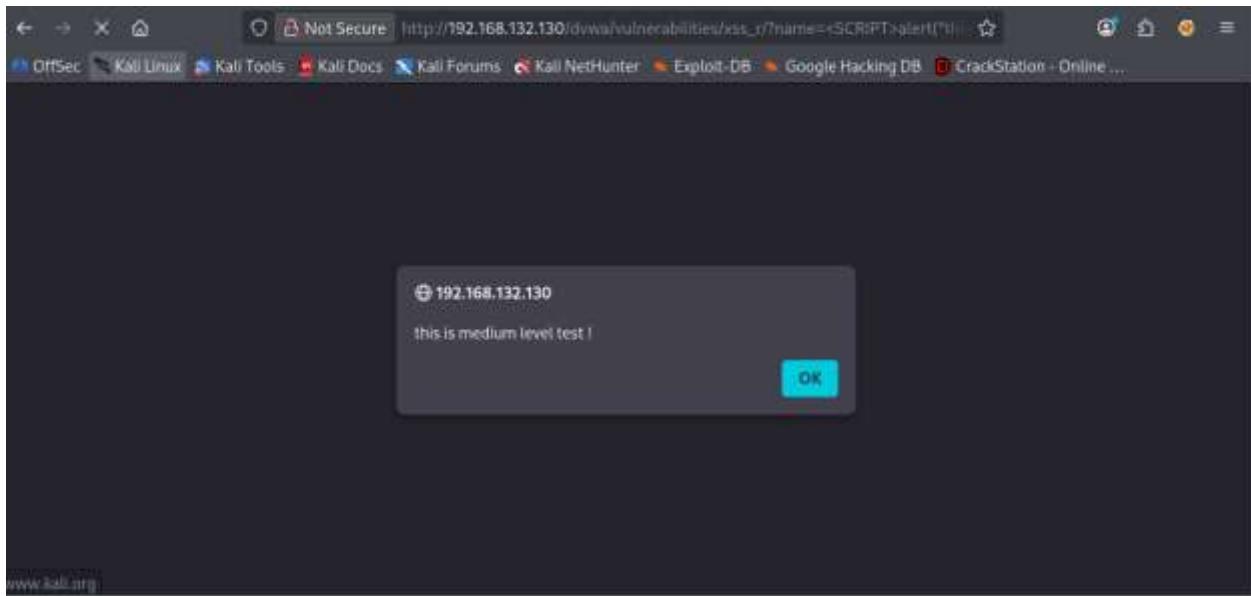
    <h2>More info</h2>

    <ul>
        <li><a href="http://hiderefer.com/?http://ha.ckers.org/xss.html" target="_blank">http://ha.ckers.org/xss.html</a></li>
        <li><a href="http://hiderefer.com/?http://en.wikipedia.org/wiki/Cross-site_scripting" target="_blank">http://en.wikipedia.org/wiki/Cross-site_scripting</a></li>
        <li><a href="http://hiderefer.com/?http://www.cgisecurity.com/xss-faq.html" target="_blank">http://www.cgisecurity.com/xss-faq.html</a></li>
    </ul>
</div>

<br />
<br />
```

In this code seems my <script> tag vanished by [\$name = str_replace('<script>', '', \$_GET['name']) ;] , this piece of JavaScript code. This source code creates a filter, with str_replace() function, that removes the <script> tag in our payload and replaces it with a null value. This renders the payload script ineffective, so the attack failed, and no popup window is displayed. Because this script is only filtering out <script> in lower case, we can try and get around the filter by using a different tag in the payload. We will use <ScRipt>.

This time I use [<SCRIPT>alert("this is medium level test !")</script>] code and its work .This time its work because [str_replace()] function active or null value only when its value is exactly as [<script>] tag . when I use uppercase then the function can not detracted it as a tag so its work.so its also compromised by attack.



Its seems that this medium lavel page have vulnerability of XSS Reflected and I found it. So it's can be compromised by attacker.

High security XSS Reflected Attack

Submit security level medium to high :

A screenshot of the DVWA (Damn Vulnerable Web Application) interface. The top navigation bar has 'DVWA' in the center. On the left is a vertical sidebar with menu items: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, and XSS stored. The main content area has a heading 'DVWA Security' with a lock icon. Below it is a section titled 'Script Security' with the subtext 'Security Level is currently high.' A dropdown menu is set to 'high' with a 'Submit' button next to it. Further down is a section titled 'PHPIDS' with the subtext 'PHPIDS v.0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.' It says 'You can enable PHPIDS across this site for the duration of your session.' Below that, it states 'PHPIDS is currently disabled.' with a link '[enable PHPIDS]' underlined.

Its same as previous two level. Take look how the page work and the inject malicious code for check the vulnerability.

Code: [<script>alert("this is medium level test !")</script>] and [<SCRIPT>alert("this is medium level test !")</script>] this time not working its do what its mean to be:

The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The title bar says "DVWA". The main content area is titled "Vulnerability: Reflected Cross Site Scripting (XSS)". On the left, there's a sidebar with various menu items: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected (which is highlighted in green), and XSS stored. The main form asks "What's your name?" with a text input field and a "Submit" button. Below the input field, the text "Hello [<SCRIPT>alert('this is medium level test !')</script>]" is displayed in red, indicating that the injected script was not executed. Below the form, there's a "More info" section with links to external resources: <http://ha.ckers.org/xss.html>, http://en.wikipedia.org/wiki/Cross-site_scripting, and <http://www.cgisecurity.com/xss-faq.html>.

Lets take look its source code :

The screenshot shows a browser window displaying the source code for a reflected XSS attack. The URL is http://192.168.132.130/dvwa/vulnerabilities/view_source.php?id=xss_r6. The title of the page is "Reflected XSS Source". The source code is as follows:

```
<?php  
if(!array_key_exists ("name", $_GET) || $_GET['name'] == NULL ||  
$_GET['name'] == ''){  
  
$isempty = true;  
  
} else {  
  
echo '<pre>';  
echo 'Hello ' . htmlspecialchars($_GET['name']);  
echo '</pre>';  
  
}  
  
?>
```

At the bottom of the page is a "Compare" button.

Its validation code (!array_key_exists ("name", \$_GET) || \$_GET['name'] == NULL || \$_GET['name'] == '')AMD(echo 'Hello '. htmlspecialchars(\$_GET['name']));)make all the input value as a string and make all the specialcharacter like alternete . so, those Not respond like tag anymore . so malicious code didn't work as before low and medium levels.

The screenshot shows the DVWA XSS Reflected page. On the left, a sidebar lists various vulnerabilities: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected (highlighted in green), XSS stored, DVWA Security, PHP Info, and About. The main content area has a title "Vulnerability: Reflected Cross Site Scripting (XSS)". It contains a form asking "What's your name?" with a "Submit" button. Below the form, the output is "Hello ". To the right, a Firefox browser window shows the source code of the page, which includes a PHP script that checks if the 'name' parameter is empty or null, and then echo's the input back to the user.

```

<?php
if(!array_key_exists ("name", $_GET) ||
$_GET['name'] == NULL || $_GET['name'] == ''){
    $isempty = true;
} else {
    echo '<pre>';

```

```

<div class="body_padded">
<h1>Vulnerability: Reflected Cross Site Scripting (XSS)</h1>
<div class="vulnerable_code_area">
<form name="XSS" action="#" method="GET">
<p>What's your name?</p>
<input type="text" name="name">
<input type="submit" value="Submit">
</form>
<pre>Hello <img src=ss oError=alert("test3")></pre>
</div>
<h2>More info</h2>
<ul>
<li><a href="http://ha.ckers.org/xss.html" target="_blank">http://ha.ckers.org/xss.html</a></li>
<li><a href="http://ha.ckers.org/xss.html" target="_blank">http://ha.ckers.org/xss.html</a></li>
<li><a href="http://www.cgisecurity.com/xss-faq.html" target="_blank">http://www.cgisecurity.com/xss-faq.html</a></li>
</ul>
<br />
<br />
</div>
<div style="clear:both">

```

It seems that this high level page have no vulnerability of XSS Reflected and I found it. So it's cannot be compromised by XSS attack.

-----End-----