

XSS & SQL Injection

Obiettivo

Configurare il laboratorio virtuale per sfruttare con successo le vulnerabilità XSS e SQL Injection sulla Damn Vulnerable Web Application DVWA.

XSS Reflected

Security Low

Andiamo ora ad analizzare nello specifico ciò che concerne un tipo di attacco XSS Reflected. Come possiamo vedere negli screen sottostanti un attaccante potrebbe andare ad inserire del codice JavaScript all'interno del campo di ricerca (In questo caso <script> alert('XSS Attack');</script>)

The screenshot shows the DVWA interface with the 'XSS reflected' menu item selected. In the main content area, there is a form with a text input field containing the value '<script>alert("XSS Attack");</script>'. Below the input field, the word 'Hello' is displayed in red text, indicating that the user's input was successfully reflected back to them.

Il server prende l'input dell'utente e lo inserisce direttamente nel contenuto HTML della pagina dei risultati.

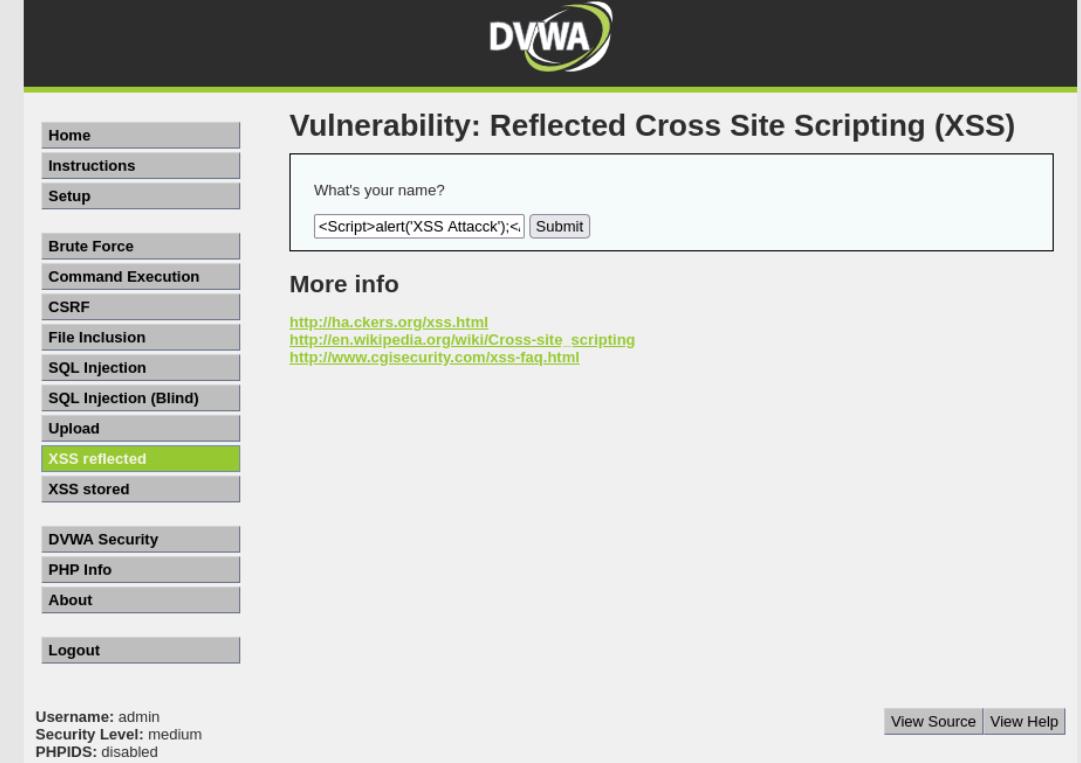
The screenshot shows the DVWA interface with the 'XSS reflected' menu item selected. In the main content area, there is a form with a text input field containing the value '<script>alert("XSS Attack");</script>'. Below the input field, the word 'Hello' is displayed in red text. A confirmation dialog box is overlaid on the page, containing the IP address '192.168.50.3' and the text 'XSS Attack'. A blue 'OK' button is visible in the bottom right corner of the dialog box.

Security Medium

Per quanto riguarda il caso in cui la security impostata su Medium come possiamo vedere viene effettuata una ‘sanificazione’ dell’input (`echo 'Hello' . str_replace('<script>', '', $_GET['name']);`)

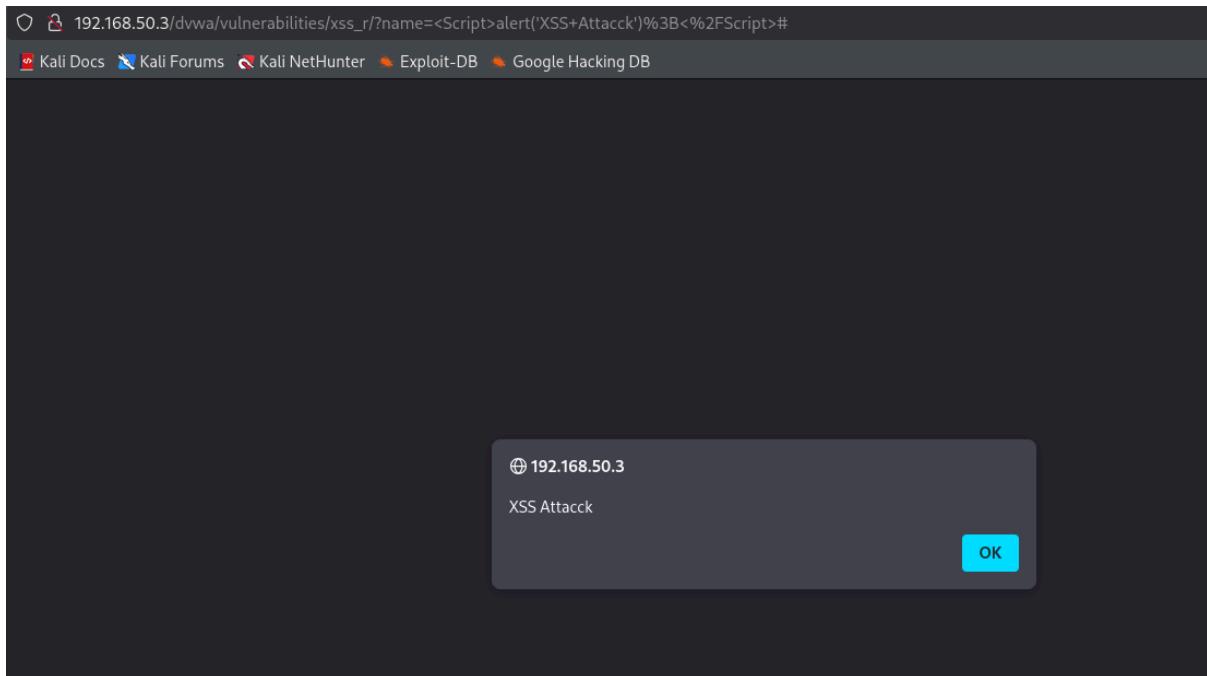
```
Medium Reflected XSS Source
<?php
if(!array_key_exists ("name", $_GET) || $_GET['name'] == NULL || $_GET['name'] == ''){
    $isempty = true;
} else {
    echo '<pre>';
    echo 'Hello' . str_replace('<script>', '', $_GET['name']);
    echo '</pre>';
}
?>
```

Ma questo tipo di sanificazione non è abbastanza in quanto essendo i tag di javascript casesensitive è possibile utilizzare lo stesso comando modificandolo per ottenere lo stesso risultato come mostrato in figura. Quindi modificando il tag con `<Script>alert('testo');`



The screenshot shows the DVWA application interface. The main title is "Vulnerability: Reflected Cross Site Scripting (XSS)". On the left, there's a sidebar menu with various security testing options. The "XSS reflected" option is highlighted with a green background. In the main content area, there's a form field labeled "What's your name?" containing the value "<Script>alert('XSS Attackk');<". Below the form is a "Submit" button. To the right of the form, under "More info", are three links: <http://ha.ckers.org/xss.html>, http://en.wikipedia.org/wiki/Cross-site_scripting, and <http://www.cgisecurity.com/xss-faq.html>. At the bottom of the page, it shows the user information: "Username: admin", "Security Level: medium", and "PHPIDS: disabled". There are also "View Source" and "View Help" buttons at the bottom right.

Otteniamo come aspettato



SQL Injection

Un attacco di tipo SQL injection SQLi permette ad un utente non autorizzato di prendere il controllo sui comandi SQL utilizzati da un'applicazione Web. La query è progettata per estrarre i nomi delle tabelle dal database dvwa e visualizzarli sull'interfaccia dell'applicazione, sfruttando una vulnerabilità SQL Injection di tipo **UNION**.

```
' UNION SELECT table_name,null FROM information_schema.tables  
WHERE table_schema = 'dvwa' -- -
```

Affinché **UNION** funzioni, il numero e il tipo di colonne della query iniettata devono corrispondere al numero e al tipo di colonne della query originale
-- : È il simbolo standard di commento in SQL che dice al server di ignorare tutto ciò che segue fino alla fine della riga.

- : A volte viene aggiunto per assicurare che, se l'applicazione aggiunge automaticamente una virgoletta singola o un punto e virgola dopo l'input, il commento sia comunque valido.

Come possiamo vedere dall'esecuzione della query ci vengono restituiti i nomi delle tabelle del database DVWA.

The screenshot shows the DVWA SQL Injection page. On the left, there is a sidebar with various menu items: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, and SQL Injection. The SQL Injection item is highlighted with a green background. The main area has a title "Vulnerability: SQL Injection". It contains a form with a "User ID:" label and a text input field. Below the input field is a "Submit" button. To the right of the input field, there is a red error message: "ID: ' UNION SELECT table_name,null FROM information_schema.tables WHERE table_schema = 'dvwa' -- - First name: guestbook Surname:". Another line of the message follows: "ID: ' UNION SELECT table_name,null FROM information_schema.tables WHERE table_schema = 'dvwa' -- - First name: users Surname:".