Java RMI Hacking

RCE, meterpreter
Alessandro Morabito @ Epicode

1 Introduzione

In questo progetto si vuole sfruttare una vulnerabilità nel servizio `Java RMI` in ascolto su porta `1099`. `Java RMI` (`Java Remote Method Invocation` permette l'accesso ad oggetti in un'altra `Java VM`.

2 Scansione con `nmap`

Per velocizzare il processo di scansione, sapendo che vogliamo andare a controllare unicamente i servizi sulla porta `1099`, restringo lo *scope* a quella porta.

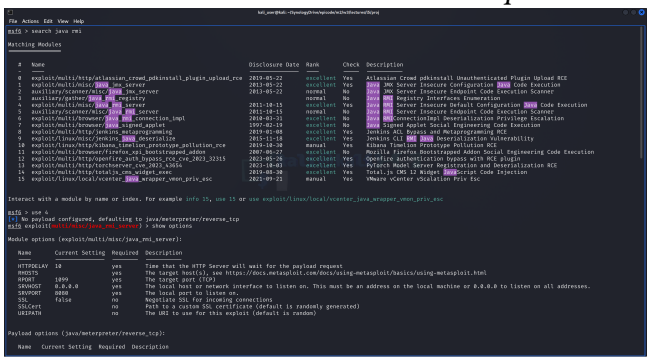
```
PORT     STATE SERVICE  VERSION
1099/tcp open  java-rmi GNU Classpath grmiregistry
```

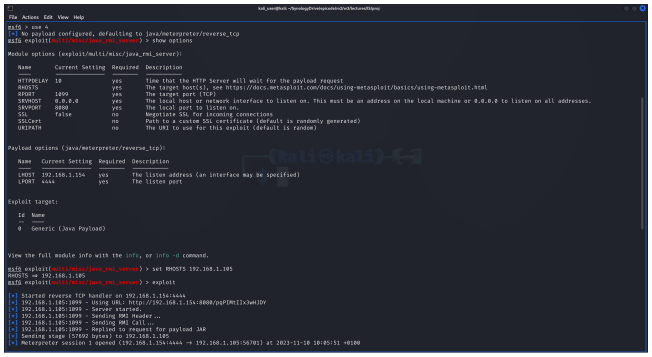
Dal momento che è stato richiesto di utilizzare `nmap` per evidenziare la vulnerabilità, ho aggiunto la flag `--script "rmi-vuln-classloader"` per effettuare l'analisi:

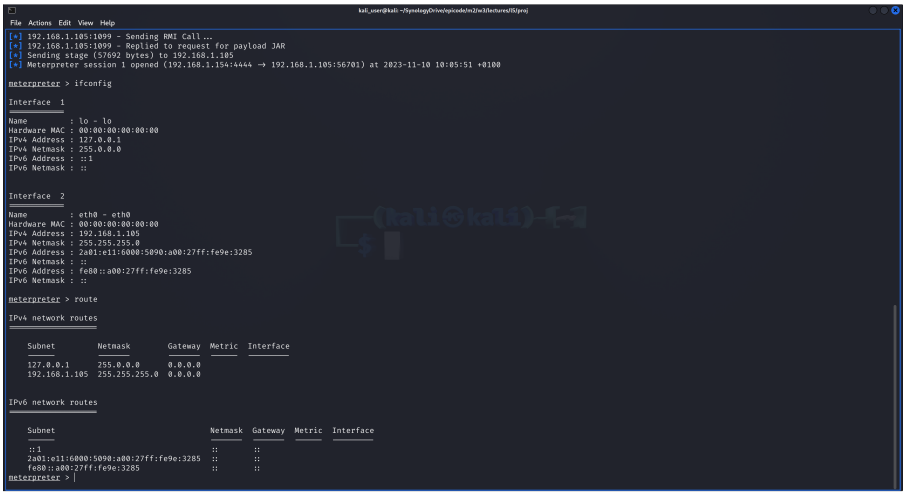
```
PORT     STATE SERVICE  VERSION
1099/tcp open  java-rmi GNU Classpath grmiregistry
| rmi-vuln-classloader:
|   VULNERABLE:
|   RMI registry default configuration remote code execution vulnerability
|     State: VULNERABLE
|       Default configuration of RMI registry allows loading classes from remote
|       URLs which can lead to remote code execution.
```

3 Exploit con `metasploit-framework`

Dopo aver individuato il servizio in ascolto sulla porta ho utilizzato `metasploit` per cercare un exploit per tale servizio. Dopo averlo selezionato ho inizialmente provato il payload `java/meterpreter/reverse_tcp` di default. È bastato inserire l'indirizzo della macchina target ed eseguire l'exploit per ottenere una sessione remota sulla macchina *metasploitable*.



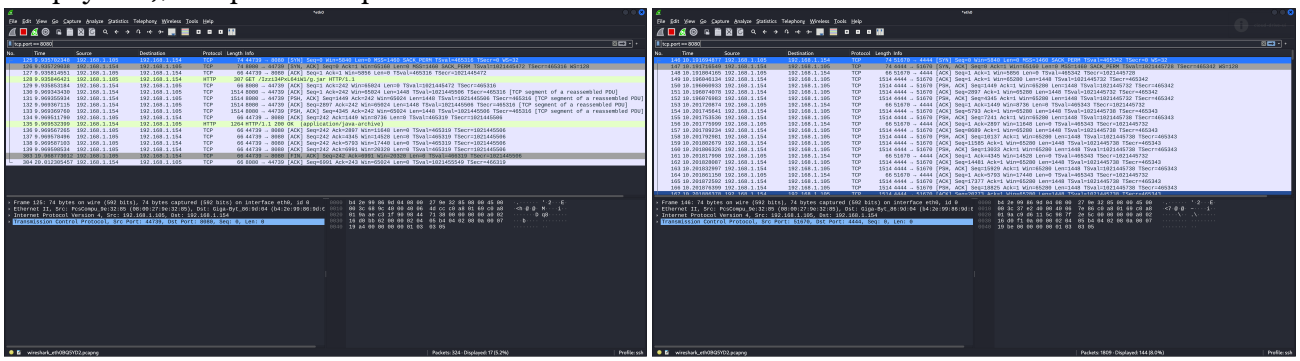




Eseguendo il comando `getuid`, inoltre, possiamo vedere che la shell di meterpreter ottenuta ha permessi di `root`.

4 Controllo con `wireshark`

Ho quindi utilizzato `wireshark` per controllare come avviene lo scambio di pacchetti tra i due dispositivi. In particolare, ho filtrato i pacchetti così intercettati con le porte `8080` e `4444`, che corrispondono alle variabili rispettivamente `SRVPORT` e `LPORT` dell'exploit e del payload. Si può notare come la porta `8080` venga utilizzata solamente al lancio dell'exploit. Probabilmente questa serve quindi per emulare la `Java VM` e quindi sfruttare la vulnerabilità. Solamente dopo entra in gioco la porta `4444` (impostata nel payload), che permette quindi la comunicazione con la backdoor.



5 Conclusioni

In questo progetto abbiamo sfruttato una vulnerabilità nella configurazione di `Java RMI` per ottenere una sessione `meterpreter` con permessi di `root`. La scansione con `nmap` mostra che il problema sta nei file di configurazione di `Java RMI`. Per risolvere questo problema, quindi, un approccio può essere modificare tale file di configurazione e impostare delle opportune regole di firewall sulle porte interessate.