Progetto - S7/L5

Exploiting di Java RMI con Metasploit

Natalino Imbrogno

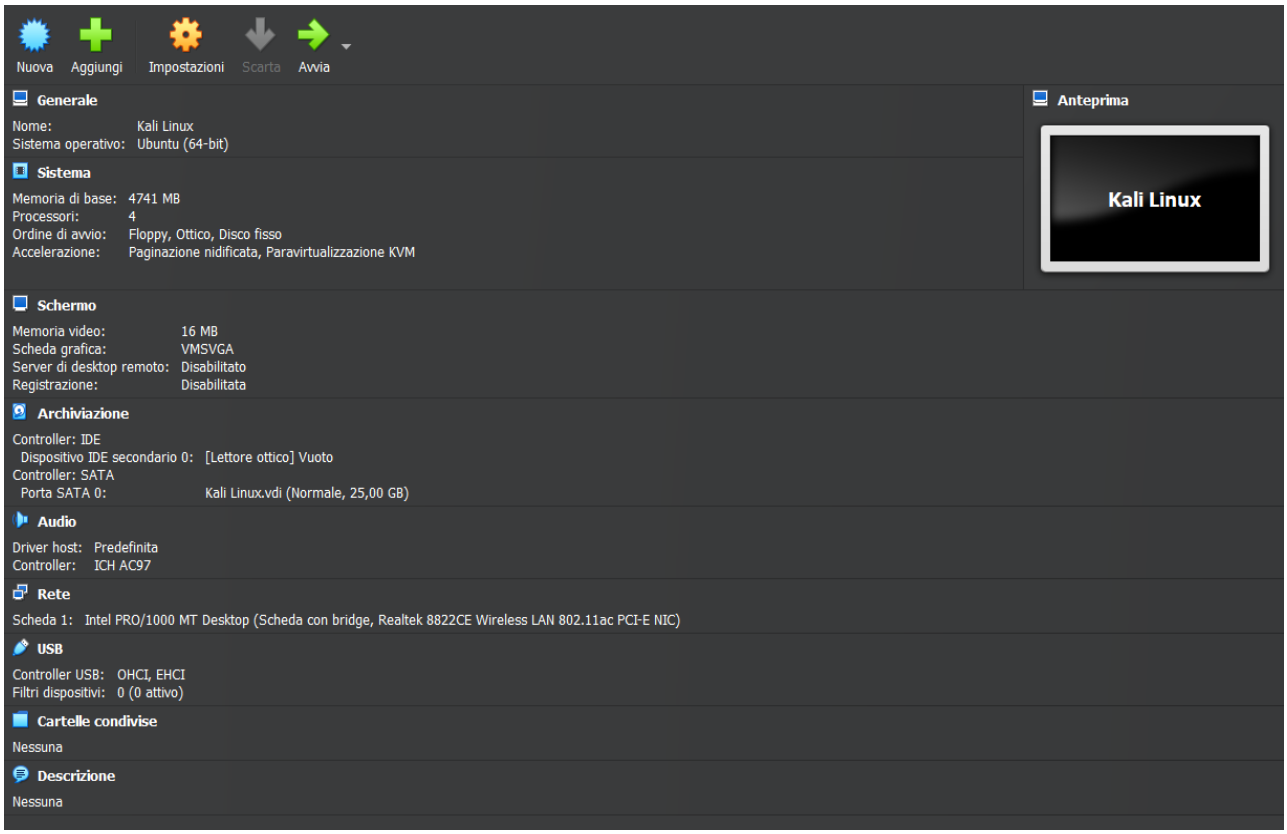
Cybersecurity Specialist
EPICODE

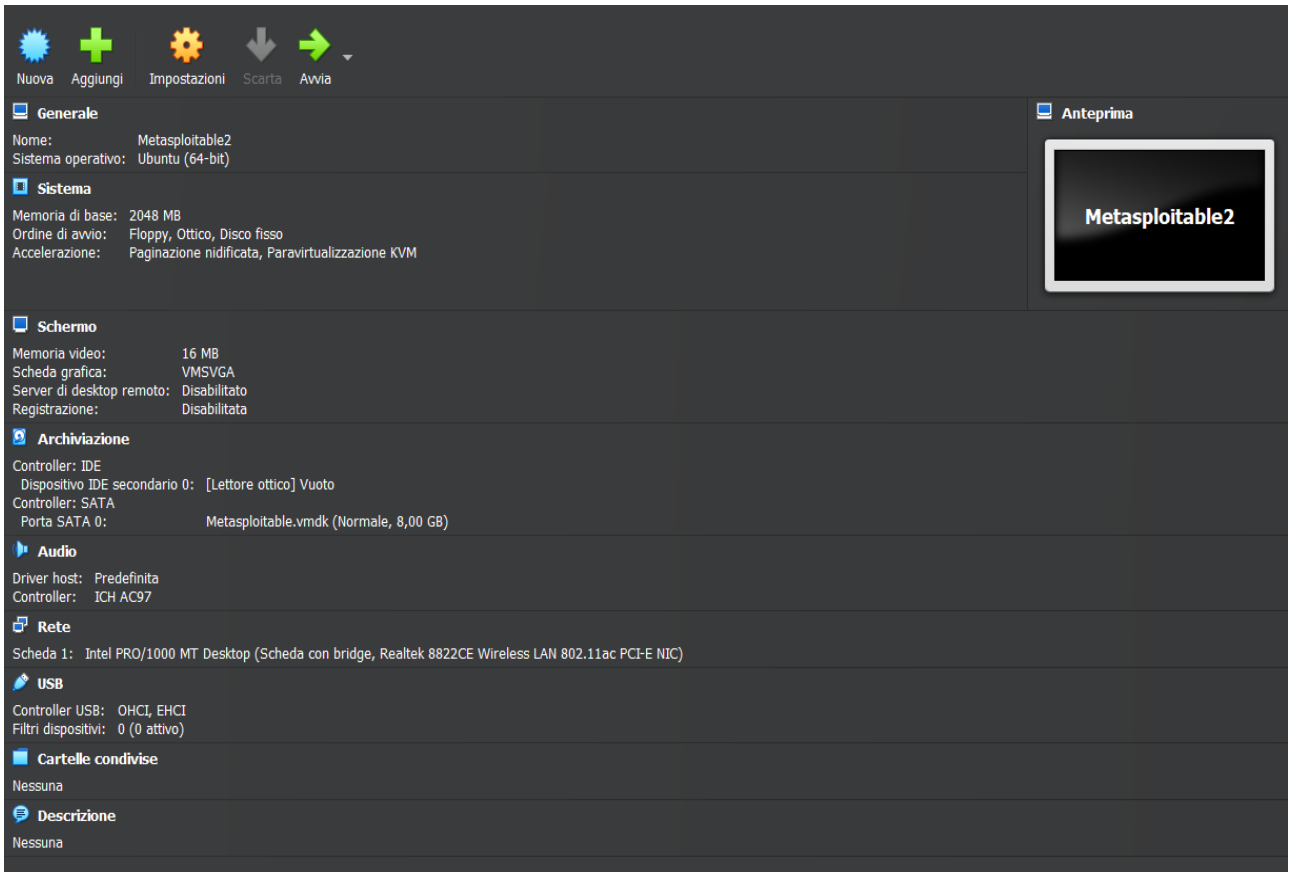
OBIETTIVO

Mi è stato richiesto di sfruttare la vulnerabilità presente sulla macchina Metasploitable 2 relativa ad un servizio vulnerabile sulla porta 1099 che riguarda Java RMI utilizzando Metasploit al fine di ottenere una sessione di Meterpreter sulla macchina remota.

SETUP DELL'AMBIENTE

Per fare ciò, ho operato in un laboratorio configurato all'interno di VirtualBox. Quì ho installato una macchina Kali (lato pentester) e una con Metasploitable 2 (lato target).





SOFTWARE UTILIZZATI

- VirtualBox: software di virtualizzazione
- Kali Linux: distro Linux
- Metasploitable 2: macchina virtuale Linux deliberatamente vulnerabile
- Nmap: tool per la scansione delle reti
- Metasploit: framework che consente di testare e sfruttare le vulnerabilità remote

SCANSIONE DEL TARGET

La prima cosa che vado a fare è assicurarmi che la macchina target

```
To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:4a:ad:67
          inet addr:192.168.1.149  Bcast:192.168.50.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe4a:ad67/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:74 errors:0 dropped:0 overruns:0 frame:0
          TX packets:68 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:6938 (6.7 KB)  TX bytes:6770 (6.6 KB)
          Base address:0xd020 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:34 errors:0 dropped:0 overruns:0 frame:0
          TX packets:34 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:14705 (14.3 KB)  TX bytes:14705 (14.3 KB)
```

sia raggiungibile dal mio host di pentester

```
┌──(natalino㉿kali)-[~]
└─$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.9  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::a00:27ff:fecd:ad9c  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:cd:ad:9c  txqueuelen 1000  (Ethernet)
        RX packets 93  bytes 24342 (23.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 47  bytes 8105 (7.9 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
```

```
┌──(natalino㉿kali)-[~]
└─$ ping 192.168.1.149
PING 192.168.1.149 (192.168.1.149) 56(84) bytes of data.
64 bytes from 192.168.1.149: icmp_seq=1 ttl=64 time=2.47 ms
64 bytes from 192.168.1.149: icmp_seq=2 ttl=64 time=11.9 ms
64 bytes from 192.168.1.149: icmp_seq=3 ttl=64 time=1.08 ms
64 bytes from 192.168.1.149: icmp_seq=4 ttl=64 time=23.3 ms
64 bytes from 192.168.1.149: icmp_seq=5 ttl=64 time=0.877 ms
64 bytes from 192.168.1.149: icmp_seq=6 ttl=64 time=2.66 ms
64 bytes from 192.168.1.149: icmp_seq=7 ttl=64 time=1.68 ms
^C
── 192.168.1.149 ping statistics ──
7 packets transmitted, 7 received, 0% packet loss, time 6008ms
rtt min/avg/max/mdev = 0.877/6.268/23.252/7.784 ms
```

Dopo questa verifica, posso passare alla scansione di rete sul target al fine di determinare quali servizi sono in esecuzione. Utilizzo il comando

nmap -sV 192.168.1.149

```
┌──(natalino㉿kali)-[~]
└─$ nmap -sV 192.168.1.149
Starting Nmap 7.94 ( https://nmap.org ) at 2023-11-10 09:51 CET
Nmap scan report for METASPLOITABLE.station (192.168.1.149)
Host is up (0.039s latency).
Not shown: 978 closed tcp ports (conn-refused)
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?
513/tcp  open  login       OpenBSD or Solaris rlogind
514/tcp  open  tcpwrapped
1099/tcp open  java-rmi    GNU Classpath grmiregistry
1524/tcp open  bindshell   Metasploitable root shell
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
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 83.79 seconds
```

dove:

- *nmap* indica che voglio eseguire una scansione con Nmap;
- l'opzione *-sV* mostra il servizio e la versione del software in esecuzione su un host;
- *192.168.1.149* è l'IP del target.

La porta che a me interessa è, per l'appunto, la *1099/tcp*, la quale risulta aperta ed è associata, come mi aspettavo, al servizio *java-rmi*, ovvero una tecnologia che consente a oggetti Java di comunicare tra loro su reti diverse.

Questa porta di comunicazione può essere utilizzata per scopi legittimi o dannosi.

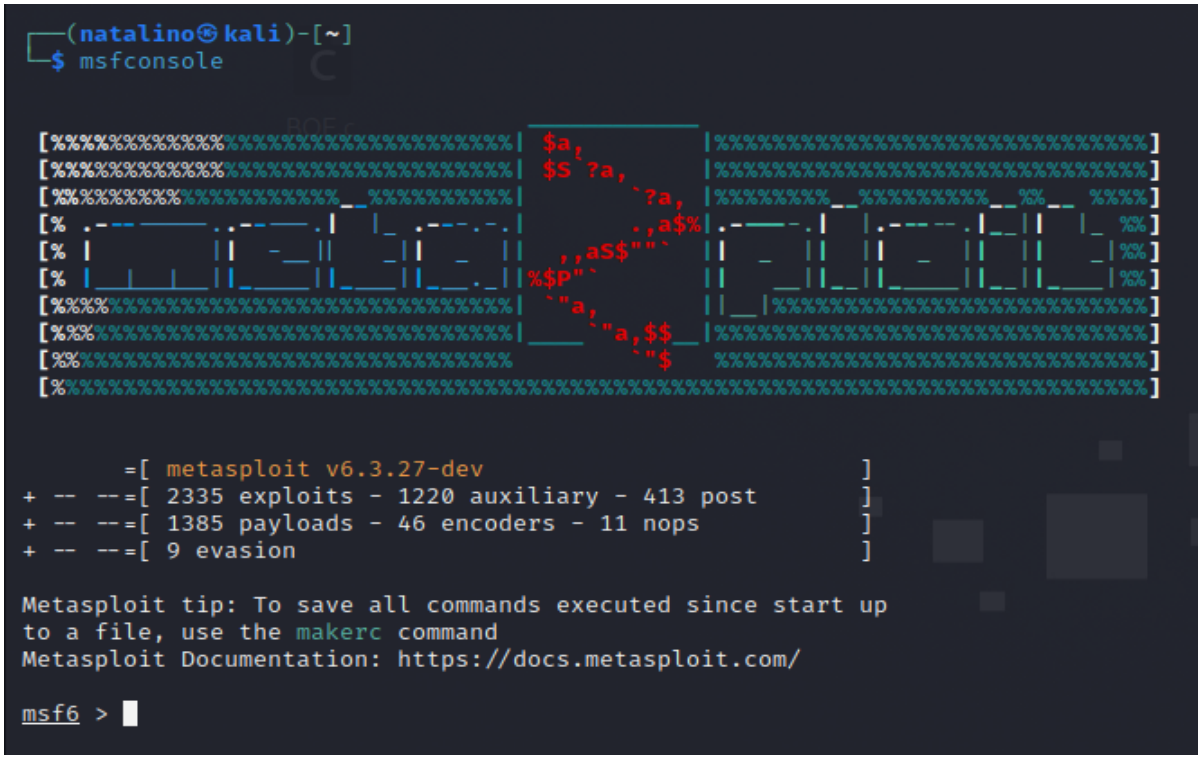
Per scopi legittimi, essa può essere utilizzata con il fine di consentire agli oggetti remoti di comunicare tra loro. Ad esempio, un'applicazione Java può utilizzare questa porta per registrare un oggetto remoto che fornisce un servizio ad un'altra applicazione Java.

Per scopi dannosi, essa può essere utilizzata con il fine di installare malware o ottenere un controllo remoto su un sistema.

EXPLOIT

Avvio ora il tool Metasploit con il comando

msfconsole



Quest'ultimo è un framework open-source usato per il penetration testing e lo sviluppo di exploit. Fornisce una vasta gamma di questi ultimi, i quali sono realizzati dalla comunità, e possiede numerosi vettori di attacco che si possono utilizzare contro diversi sistemi e tecnologie. Inoltre, può essere utilizzato anche per creare ed automatizzare i propri exploit.

Un exploit è un codice malevolo che sfrutta una vulnerabilità già presente all'interno di un software. Si differenzia dal malware perché quest'ultimo è un codice malevolo che per danneggiare un sistema ha bisogno dell'azione dell'utente. Va precisato che, quando si agisce su un software, l'exploit deve essere specifico per quel programma e per la versione dello stesso; al massimo potrebbe funzionare per le versioni precedenti. Affinché l'exploit funzioni, il software target deve essere attivo, ovvero deve risultare tra i processi. L'azione di exploiting si divide in 3 fasi:

- nella prima fase si crea una connessione dall'attaccante verso il target sfruttando una vulnerabilità;
- nella seconda fase si inietta un payload, ovvero un file nocivo che permette di realizzare un ponte tra l'attaccante e il target;
- tale ponte, chiamato shell, va a costituire la terza fase.

Ci sono due tipi di shell:

- shell reverse, ovvero quando il payload crea la connessione dal target verso l'attaccante;
- shell bind, ovvero quando il payload crea la connessione dall'attaccante verso il target.

La tipologia di shell è già implicita all'interno del payload che si andrà ad utilizzare. La shell reverse è la più utilizzata visto che, consentendo connessioni dall'interno verso l'esterno, evita problemi con il firewall, il quale blocca le connessioni che provengono da fuori della LAN. La shell bind può essere usata, invece, quando stiamo già all'interno della rete, o se (cosa molto grave) non c'è un firewall. Un exploit può essere:

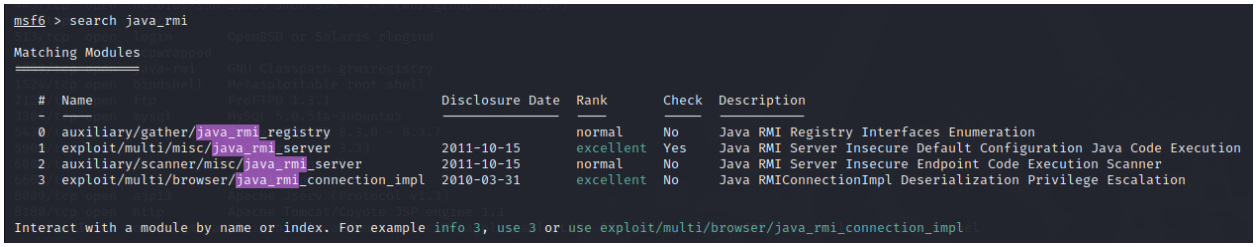
- manuale, quando si usano dei siti che contengono già dei codici malevoli (CVE, ExploitDB). In pratica, si scarica il codice e poi con Netcat (programma di rete che può essere utilizzato per creare una connessione tra due host) lo si usa sul target. C'è da dire che l'exploit manuale è consigliato farlo solo su una rete di piccole dimensioni, dato che richiederebbe molto tempo;
- automatico, quando si va ad utilizzare un software (Metasploit).

Metasploit è organizzato in moduli, ovvero unità di codice che forniscono funzionalità specifiche per l'attacco o l'analisi di un sistema informatico. Tali moduli possono essere:

- normali, quando quasi sempre hanno a che fare con un payload;
- ausiliari, nel caso in cui quasi mai si setta un payload e posseggono informazioni aggiuntive rispetto a quelli normali.

Passo dunque a cercare l'exploit di cui ho bisogno su Metasploit utilizzando il comando

search java_rmi

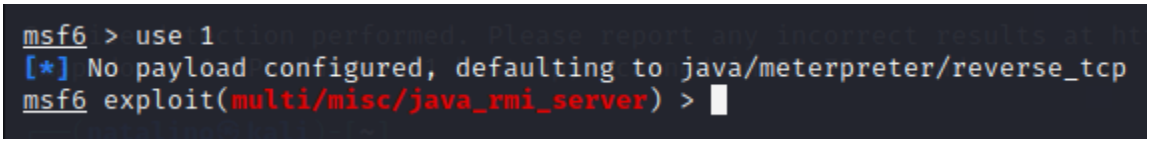


Infatti, come ho già detto, Metasploit contiene codice di exploit e payload ed altre funzionalità, il tutto contenuto nei moduli. Ogni modulo mette a disposizione un vettore di attacco diverso. E' possibile cercare un modulo utilizzando il comando *search* seguito dal termine di ricerca. Se vengono indicati più termini di ricerca, Metasploit esegue una query per ogni termine.

Così facendo, ottengo 4 risultati. Il più interessante per me sembra essere l'exploit 1, che sfrutta una vulnerabilità nel protocollo RMI per eseguire codice arbitrario su un server remoto.

Dopo aver individuato e scelto l'exploit da utilizzare, lo abilito con il comando *use* seguito dal numero dell'exploit (ma si può usare anche il suo path)

use 1

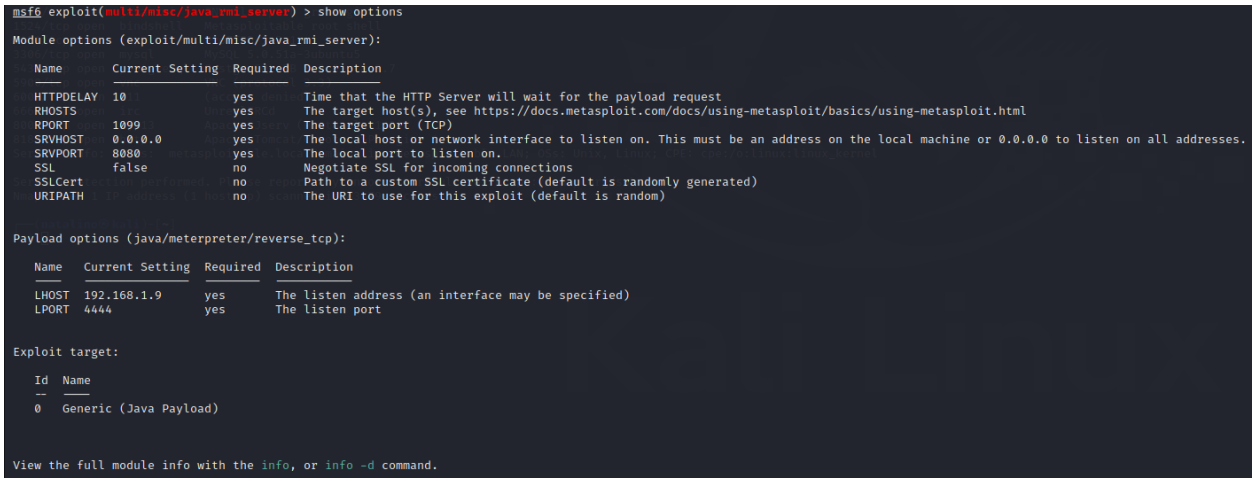


Il prompt dei comandi di MSFConsole cambia quando viene selezionato un exploit. Questo comportamento è dovuto al fatto che Metasploit usa una gerarchia tipo file system per salvare i vari exploit, payload e moduli ausiliari.

Inoltre, di default Metasploit assegna all'exploit un payload specifico, e posso utilizzare questo visto che crea una shell reverse TCP su un dispositivo Java.

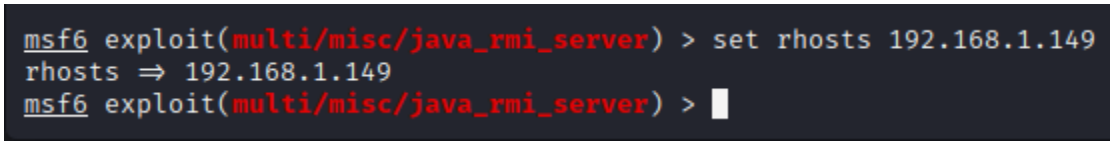
Dopo aver caricato l'exploit, controllo le opzioni di configurazione con il comando

show options



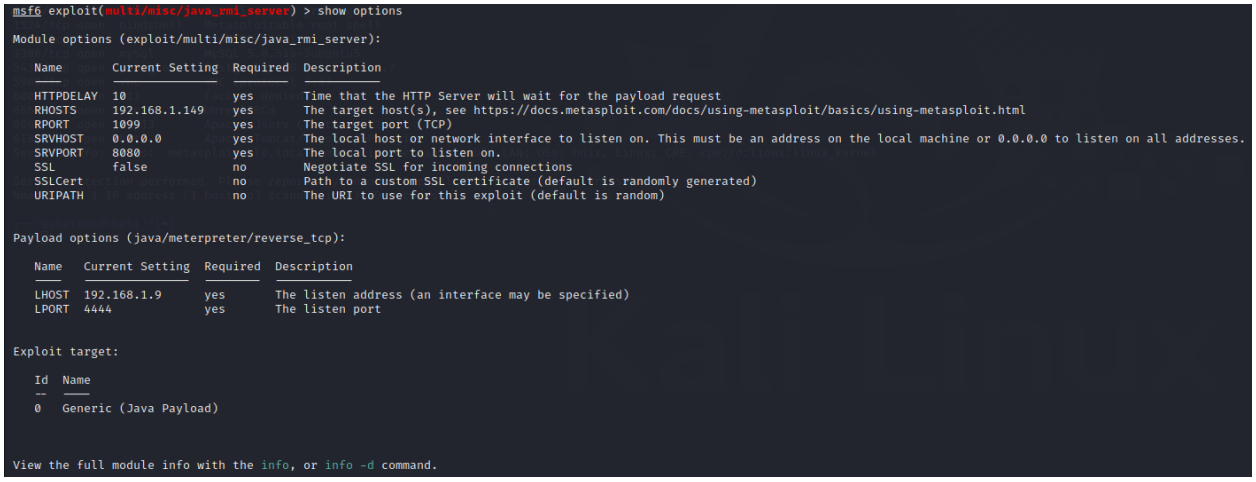
Quì, alcune configurazioni sono *required*, ovvero è obbligatorio inserirle per poter utilizzare l'exploit. Nel mio caso, devo inserire *RHOSTS*, e cioè l'indirizzo IP della macchina target. Utilizzo quindi il comando

set rhosts 192.168.1.149



Vado poi a controllare se l'indirizzo dell'host remoto è stato configurato correttamente

con un altro *show options*



Una volta configurate tutte le impostazioni ed i parametri, posso lanciare l'attacco con il comando

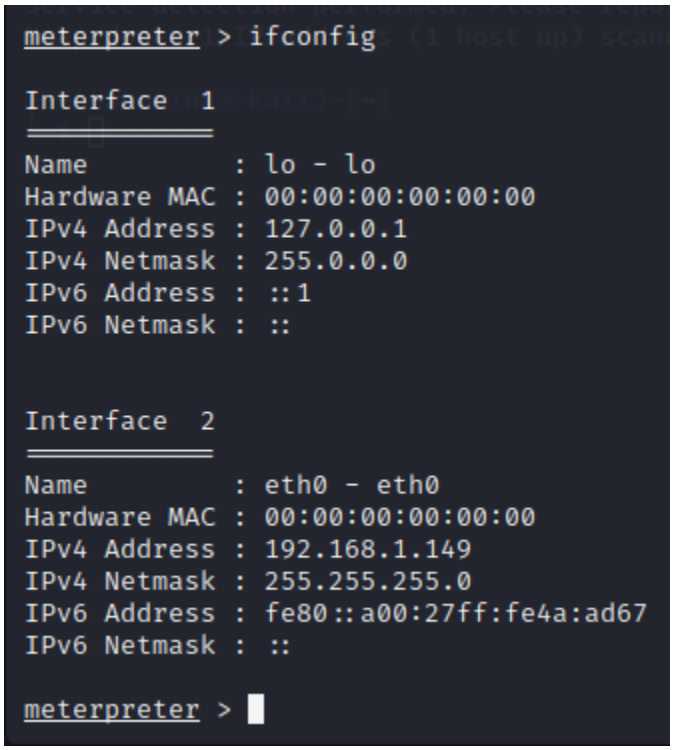
exploit

Ricevo dunque una shell di Meterpreter. Quest'ultimo è un payload avanzato di Metasploit che viene utilizzato per ottenere un accesso completo ad una macchina compromessa. Si tratta di un programma di shell remota che offre un'ampia gamma di funzionalità, tra cui:

- controllo completo del sistema operativo: Meterpreter consente al pentester di eseguire qualsiasi comando sul sistema compromesso, comprese le operazioni di basso livello come l'accesso alla memoria e al registro;
- raccolta di informazioni: Meterpreter può essere utilizzato per raccogliere informazioni sulla macchina compromessa, come le credenziali dell'utente, i file e le applicazioni installate;
- infiltrazione di rete: Meterpreter può essere utilizzato per infiltrarsi in altre macchine sulla rete.

Faccio un test per avere la conferma di essere sulla macchina target, ovvero lancio il comando

ifconfig

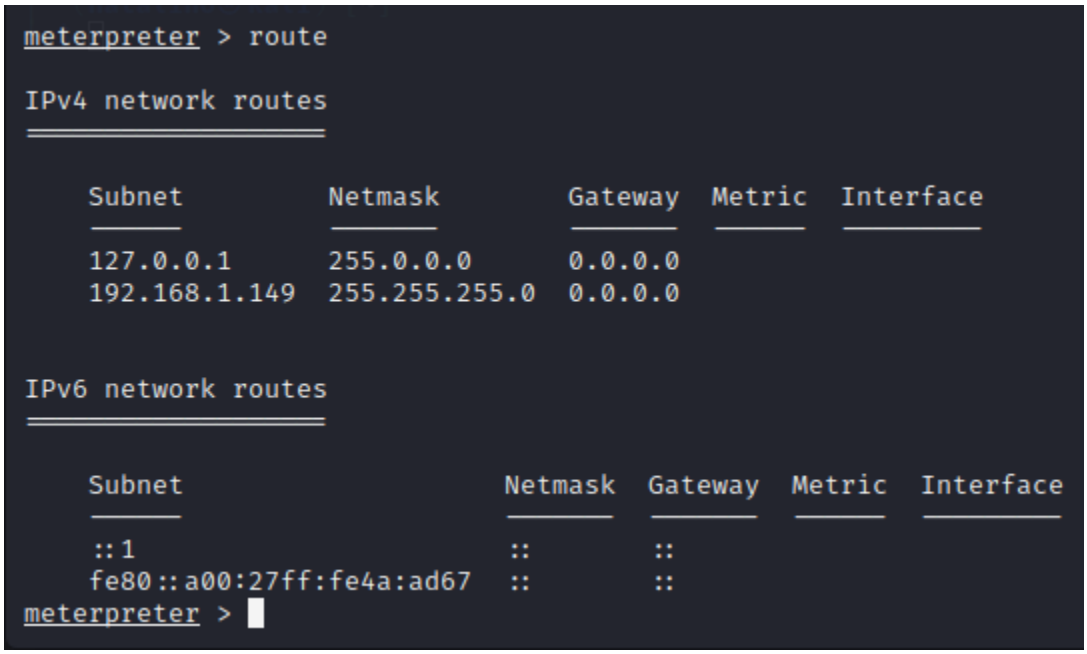


il quale mi restituisce la configurazione di rete che mi aspettavo dato che vedo l'IP *192.168.1.149*. Questa prova è sufficiente a concludere che l'attacco è andato a buon fine e ho sfruttato correttamente la vulnerabilità Java RMI per ottenere l'accesso alla macchina target.

Posso fare un'ultima cosa, ovvero con il comando

route

acquisisco informazioni sulla tabella di routing della macchina target



Una tabella del genere mostra infatti le rotte di rete di un dispositivo, e in genere viene utilizzata dai router con il fine di determinare la migliore rotta per inoltrare i pacchetti di dati. Se un malintenzionato vi accede, può eseguire diverse azioni dannose. Innanzitutto, può utilizzare le informazioni contenute nella tabella per inoltrare i pacchetti di dati verso destinazioni non autorizzate. Ad esempio, può inoltrare i pacchetti di dati di un utente verso un sito web dannoso o verso un server remoto controllato dal malintenzionato. In secondo luogo, può modificare la tabella per cambiare il modo in cui i pacchetti di dati vengono inoltrati. Ad esempio, può modificare la destinazione di una rotta per inoltrare i pacchetti di dati verso un dispositivo non autorizzato. In terzo luogo, può eliminare voci dalla tabella per impedire che i pacchetti di dati vengano inoltrati verso determinate destinazioni. Ad esempio, può eliminare la voce per la rete locale con il fine di impedire agli utenti di accedere ai dispositivi locali.