Dirty COW (CVE-2016-5195)

Linux Kernel vulnerability

Elaheh Toulabi Nejad

OUTLINE

1. What is Dirty COW
2. Paging in Operating System
3. Copy on write
4. Race Condition
5. How Dirty COW Works
6. Impacts and Applications
7. Solutions
8. Let’s Dive Into It

What is Dirty COW?

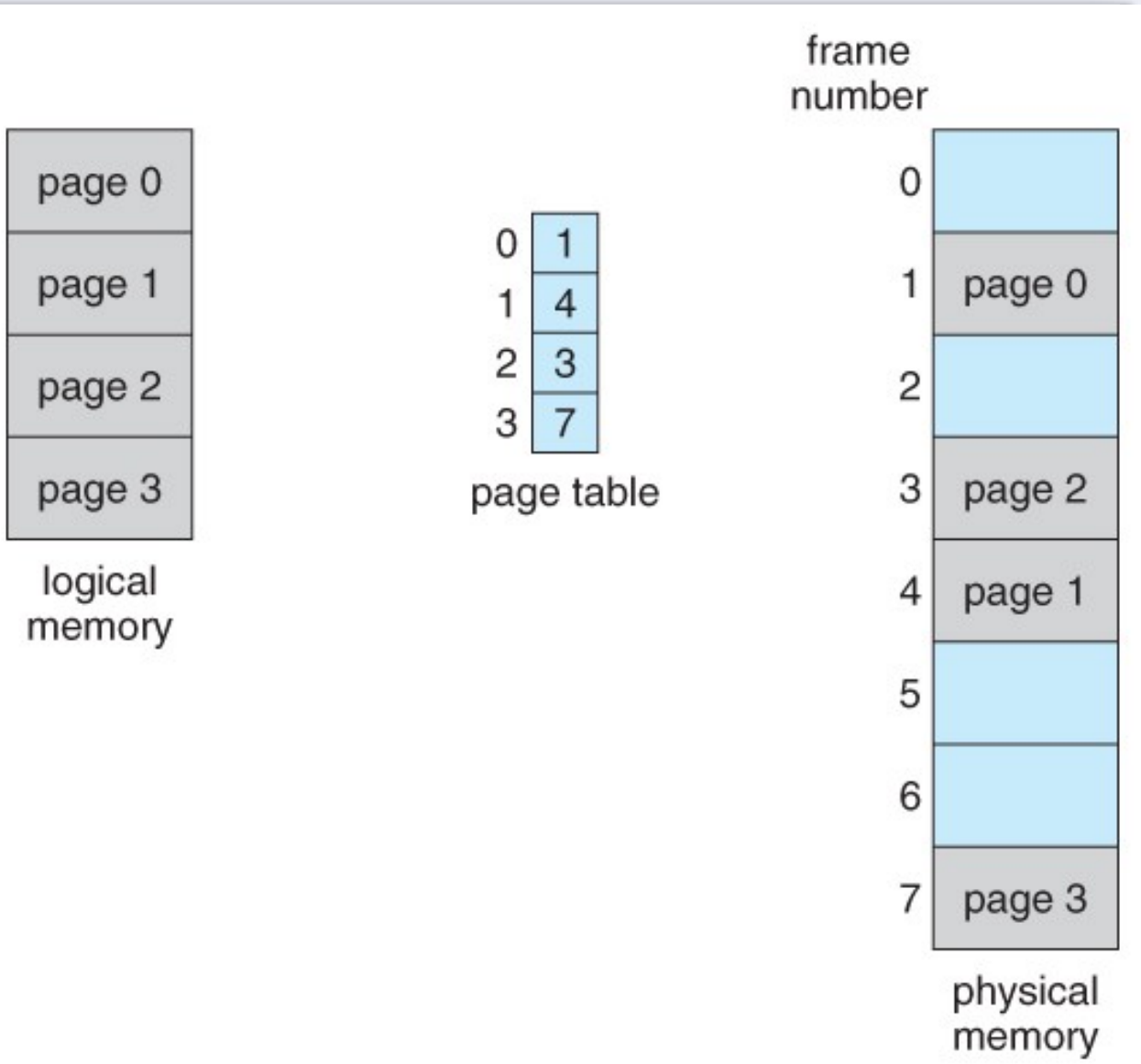
- Was discovered by Phil Oester
- A vulnerability in the Linux kernel since version 2.6.22 released in September 2007
- A local privilege escalation bug that exploits a race condition in the implementation of the copy-on-write mechanism in the kernel's memory-management subsystem

OUTLINE

1. What is Dirty COW
2. **Paging in Operating System**
3. Copy on write
4. Race Condition
5. How Dirty COW Works
6. Impacts and Applications
7. Solutions
8. Let’s Dive Into It

Paging In OS

- Method of writing and reading data from a secondary storage for use in primary storage.
- Main idea behind the paging is to divide each process in the form of pages. The main memory will also be divided in the form of frames
- Pages of the process are brought into the main memory only when they are required

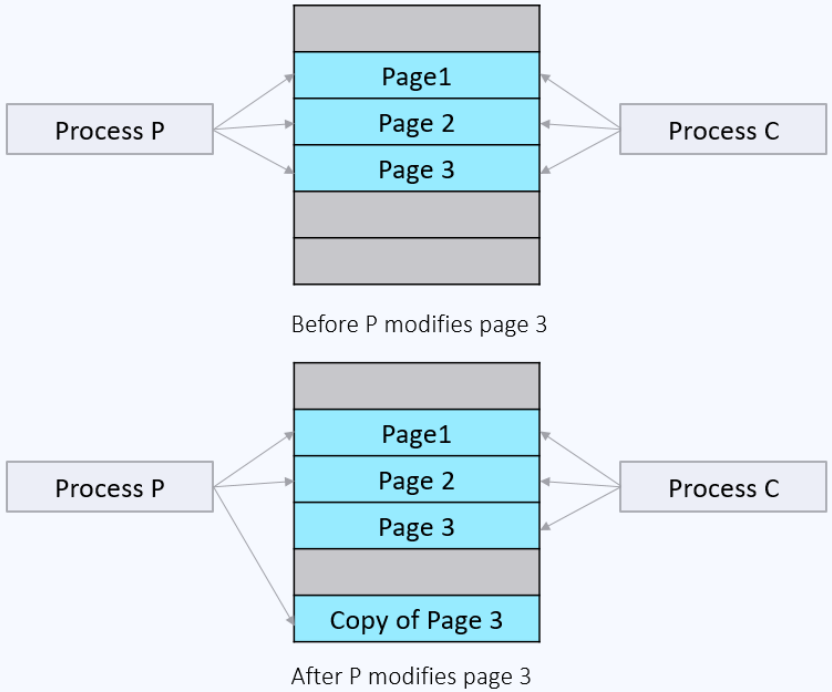


OUTLINE

1. What is Dirty COW
2. Paging in Operating System
3. Copy on write
4. Race Condition
5. How Dirty COW Works
6. Impacts and Applications
7. Solutions
8. Let’s Dive Into It

Copy-on-write

- A resource management technique
- In UNIX like OS, fork() system call creates a duplicate process of the parent process which is called as the child process.
- When a parent process creates a child process then both of these processes initially will share the same pages in memory
- If any of these processes will try to modify the shared pages then only a copy of these pages will be created and the modifications will be done on the copy of pages by that process and thus not affecting the other process.



OUTLINE

1. What is Dirty COW
2. Paging in Operating System
3. Copy on write
4. **Race Condition**
5. How Dirty COW Works
6. Impacts and Applications
7. Solutions
8. Let’s Dive Into It

Race Condition

- A situation that may occur inside a critical section.
- Happens when the result of multiple thread execution in critical section differs according to the order in which the threads execute.
- Critical section in a code segment where the shared variables can be accessed.

OUTLINE

1. What is Dirty COW
2. Paging in Operating System
3. Copy on write
4. Race Condition
5. **How Dirty COW Works**
6. Impacts and Applications
7. Solutions
8. Demo

How Dirty COW Works?

First, we create a private copy (mapping) of a read-only file. Second, we write to the private copy. Since it's our first time writing to the private copy, the COW feature takes place. **The problem** lies in the fact that this write consists of **two non-atomic actions**:

1. locate physical address

2. write to physical address

This means we can get right in the middle (via another thread) and tell the kernel to throw away our private copy — using madvise. This throwing away of the private copy results in the kernel accidentally writing to the original read-only file.

OUTLINE

1. What is Dirty COW
2. Paging in Operating System
3. Copy on write
4. Race Condition
5. How Dirty COW Works
6. **Impacts and Applications**
7. Solutions
8. Demo

Applications

- An unprivileged local user could use this flaw to gain write access to otherwise read-only memory mappings and thus increase their privileges on the system.
- They could change a file, such as /bin/bash, so that it performs additional, unexpected functions, such as a keylogger.
- & etc.

OUTLINE

1. What is Dirty COW
2. Paging in Operating System
3. Copy on write
4. Race Condition
5. How Dirty COW Works
6. Impacts and Applications
7. Solutions
8. Demo

The vulnerability has been patched in Linux kernel versions 4.8.3, 4.7.9, 4.4.26 and newer.

OUTLINE

1. What is Dirty COW
2. Paging in Operating System
3. Copy on write
4. Race Condition
5. How Dirty COW Works
6. Impacts and Applications
7. Solutions
8. Demo

DEMO