```
Date: Tue, 25 Feb 2020 19:04:01 +0100
From: Florian Weimer <fweimer@...hat.com>
To: oss-security@...ts.openwall.com
Subject: CVE-2020-9391: Ignoring the top byte of addresses in brk causes heap corruption (AArch64)

AArch64 has an architectural feature where the top byte of a 64-bit
pointer is ignored.  Therefore, applications can use this as storage
space for colored pointers without having to mask those bits.  Recent
Linux kernels (starting with 5.4) ignore the top byte in certain system
call arguments as well.  This was also done for the brk system call, but
there it can result in moving the brk in the wrong direction (downward
instead of upward).

Here's a test program that shows the problem with the glibc allocator on
AArch64:

#include <err.h>
#include <stdlib.h>
#include <stdio.h>
#include <stddef.h>
#include <string.h>

int
main (void)
{
  enum { size = 4096, count = 128 };
  void *array[count];
  for (int i = 0; i < count; ++i)
    {
      array[i] = malloc (size);
      if (array[i] == NULL)
        err (1, "malloc (%d)", size);
    }

  void *p = malloc ((2ULL << 56) - size * count);
  printf ("p = %p\n", p);
  if (p != NULL)
    explicit_bzero (p, 1024);

  for (int i = 0; i < count; ++i)
    explicit_bzero (array[i], size);

  for (int i = 0; i < count; ++i)
    free (array[i]);

  free (p);

  return 0;
}

With Fedora's 5.6.0-0.rc1.git0.1.fc32.aarch64 kernel, this is the result.

brk(NULL)                               = 0x2c490000
brk(0x2c4b1000)                         = 0x2c4b1000
brk(NULL)                               = 0x2c4b1000
brk(NULL)                               = 0x2c4b1000
brk(0x2c4d2000)                         = 0x2c4d2000
brk(NULL)                               = 0x2c4d2000
brk(0x2c4f3000)                         = 0x2c4f3000
brk(NULL)                               = 0x2c4f3000
brk(0x2c514000)                         = 0x2c514000
mmap(NULL, 144115188075335680, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = -1 ENOMEM (Cannot allocate memory)
brk(NULL)                               = 0x2c514000
brk(0x20000002c4b1000)                  = 0x2c4b1000
mmap(NULL, 144115188075466752, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = -1 ENOMEM (Cannot allocate memory)
--- SIGSEGV {si_signo=SIGSEGV, si_code=SEGV_MAPERR, si_addr=0x2c510a98} ---
+++ killed by SIGSEGV (core dumped) +++

The last brk call moved the break down because the top byte has been
ignored by the kernel.  glibc detects the brk result as a failure, but
at that point, the damage is already done, and the heap is corrupted.

Originally reported by Victor Stinner as
<https://bugzilla.redhat.com/show_bug.cgi?id=1797052>.  Additional
analysis by DJ Delorie.

The upstream fix is here:

commit dcde237319e626d1ec3c9d8b7613032f0fd4663a
Author: Catalin Marinas <catalin.marinas@....com>
Date:   Wed Feb 19 12:31:56 2020 +0000

    mm: Avoid creating virtual address aliases in brk()/mmap()/mremap()

    Currently the arm64 kernel ignores the top address byte passed to brk(),
    mmap() and mremap(). When the user is not aware of the 56-bit address
    limit or relies on the kernel to return an error, untagging such
    pointers has the potential to create address aliases in user-space.
    Passing a tagged address to munmap(), madvise() is permitted since the
    tagged pointer is expected to be inside an existing mapping.

    The current behaviour breaks the existing glibc malloc() implementation
    which relies on brk() with an address beyond 56-bit to be rejected by
    the kernel.

    Remove untagging in the above functions by partially reverting commit
    ce18d171cb73 ("mm: untag user pointers in mmap/munmap/mremap/brk"). In
    addition, update the arm64 tagged-address-abi.rst document accordingly.

    Link: https://bugzilla.redhat.com/1797052
    Fixes: ce18d171cb73 ("mm: untag user pointers in mmap/munmap/mremap/brk")
    Cc: <stable@...r.kernel.org> # 5.4.x-
    Cc: Florian Weimer <fweimer@...hat.com>
    Reviewed-by: Andrew Morton <akpm@...ux-foundation.org>
    Reported-by: Victor Stinner <vstinner@...hat.com>
    Acked-by: Will Deacon <will@...nel.org>
    Acked-by: Andrey Konovalov <andreyknvl@...gle.com>
    Signed-off-by: Catalin Marinas <catalin.marinas@....com>
    Signed-off-by: Will Deacon <will@...nel.org>

<https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=dcde237319e626d1ec3c9d8b7613032f0fd4663a>

Thanks,
Florian
```