index : kernel/git/torvalds/linux.git

master switch

Linux kernel source tree — Linus Torvalds

about summary refs log tree commit diff stats — log msg search

author	Linus Torvalds <torvalds@linux-foundation.org>	2016-10-13 13:07:36 -0700
committer	Linus Torvalds <torvalds@linux-foundation.org>	2016-10-18 14:13:29 -0700
commit	19be0eaffa3ac7d8eb6784ad9bdbc7d67ed8e619 (patch)	
tree	9ed601a5726b067beb3e29414c469f88c499a63b	
parent	6b25e21fa6f26d0f0d45f161d169029411c84286 (diff)	
download	linux-19be0eaffa3ac7d8eb6784ad9bdbc7d67ed8e619.tar.gz	

diff options

context:	3
space:	include
mode:	unified

mm: remove gup_flags FOLL_WRITE games from __get_user_pages()

```
This is an ancient bug that was actually attempted to be fixed once
(badly) by me eleven years ago in commit 4ceb5db9757a ("Fix
get_user_pages() race for write access") but that was then undone due to
problems on s390 by commit f33ea7f404e5 ("fix get_user_pages bug").

In the meantime, the s390 situation has long been fixed, and we can now
fix it by checking the pte_dirty() bit properly (and do it better).  The
s390 dirty bit was implemented in abf09bed3cce ("s390/mm: implement
software dirty bits") which made it into v3.9.  Earlier kernels will
have to look at the page state itself.

Also, the VM has become more scalable, and what used a purely
theoretical race back then has become easier to trigger.

To fix it, we introduce a new internal FOLL_COW flag to mark the "yes,
we already did a COW" rather than play racy games with FOLL_WRITE that
is very fundamental, and then use the pte dirty flag to validate that
the FOLL_COW flag is still valid.

Reported-and-tested-by: Phil "not Paul" Oester <kernel@linuxace.com>
Acked-by: Hugh Dickins <hughd@google.com>
Reviewed-by: Michal Hocko <mhocko@suse.com>
Cc: Andy Lutomirski <luto@kernel.org>
Cc: Kees Cook <keescook@chromium.org>
Cc: Oleg Nesterov <oleg@redhat.com>
Cc: Willy Tarreau <w@1wt.eu>
Cc: Nick Piggin <npiggin@gmail.com>
Cc: Greg Thelen <gthelen@google.com>
Cc: stable@vger.kernel.org
Signed-off-by: Linus Torvalds <torvalds@linux-foundation.org>
```

Diffstat

-rw-r--r--	include/linux/mm.h	1
-rw-r--r--	mm/gup.c	14

2 files changed, 13 insertions, 2 deletions

```
diff --git a/include/linux/mm.h b/include/linux/mm.h
index e9caec6a51e97..ed85879f47f5f 100644
```

```
--- a/include/linux/mm.h
+++ b/include/linux/mm.h
@@ -2232,6 +2232,7 @@ static inline struct page *follow_page(struct vm_area_struct *vma,
 #define FOLL_TRIED     0x800   /* a retry, previous pass started an IO */
 #define FOLL_MLOCK     0x1000  /* lock present pages */
 #define FOLL_REMOTE    0x2000  /* we are working on non-current tsk/mm */
+#define FOLL_COW       0x4000  /* internal GUP flag */

 typedef int (*pte_fn_t)(pte_t *pte, pgtable_t token, unsigned long addr,
                         void *data);

diff --git a/mm/gup.c b/mm/gup.c
index 96b2b2fd0fbd1..22cc22e7432f6 100644
--- a/mm/gup.c
+++ b/mm/gup.c
@@ -60,6 +60,16 @@ static int follow_pfn_pte(struct vm_area_struct *vma, unsigned long address,
        return -EEXIST;
 }

+/*
+ * FOLL_FORCE can write to even unwritable pte's, but only
+ * after we've gone through a COW cycle and they are dirty.
+ */
+static inline bool can_follow_write_pte(pte_t pte, unsigned int flags)
+{
+       return pte_write(pte) ||
+               ((flags & FOLL_FORCE) && (flags & FOLL_COW) && pte_dirty(pte));
+}
+
 static struct page *follow_page_pte(struct vm_area_struct *vma,
                unsigned long address, pmd_t *pmd, unsigned int flags)
 {
@@ -95,7 +105,7 @@ retry:
        }
        if ((flags & FOLL_NUMA) && pte_protnone(pte))
                goto no_page;
-       if ((flags & FOLL_WRITE) && !pte_write(pte)) {
+       if ((flags & FOLL_WRITE) && !can_follow_write_pte(pte, flags)) {
                pte_unmap_unlock(ptep, ptl);
                return NULL;
        }
@@ -412,7 +422,7 @@ static int faultin_page(struct task_struct *tsk, struct vm_area_struct *vma,
         * reCOWed by userspace write).
         */
        if ((ret & VM_FAULT_WRITE) && !(vma->vm_flags & VM_WRITE))
-               *flags &= ~FOLL_WRITE;
+               *flags |= FOLL_COW;
        return 0;
 }
```

generated by cgit 1.2.3-1.el7 (git 2.26.2) at 2021-02-05 20:31:44 +0000