
Subject: Re: /proc/*/pagemap BUG: sleeping function called from invalid context
Posted by [Matt Mackall](#) on Tue, 10 Apr 2007 20:07:03 GMT
[View Forum Message](#) <> [Reply to Message](#)

On Mon, Apr 09, 2007 at 12:25:54PM +0400, Alexey Dobriyan wrote:

```
> After
> cat /proc/self/pagemap
>
> BUG: sleeping function called from invalid context at include/asm/uaccess.h:453
> in_atomic():1, irqs_disabled():0
```

This should fix it:

When CONFIG_HIGHPTPE is enabled, use double-buffering in pagemap to avoid calling copy_to_user while preemption is disabled.

Tested on x86 with HIGHPTPE with DEBUG_SPINLOCK_SLEEP and PROVE_LOCKING.

Signed-off-by: Matt Mackall <mpm@selenic.com>

Index: mm/fs/proc/task_mmu.c

```
=====
--- mm.orig/fs/proc/task_mmu.c 2007-04-09 10:54:28.000000000 -0500
+++ mm/fs/proc/task_mmu.c 2007-04-10 14:47:21.000000000 -0500
@@ -535,6 +535,7 @@ struct pagemapread {
     struct mm_struct *mm;
     unsigned long next;
     unsigned long *buf;
+   pte_t *ptebuf;
     unsigned long pos;
     size_t count;
     int index;
@@ -573,6 +574,14 @@ static int pagemap_pte_range(pmd_t *pmd,
     int err;

     pte = pte_offset_map(pmd, addr);
+
+   #ifdef CONFIG_HIGHPTPE
+   /* copy PTE directory to temporary buffer and unmap it */
+   memcpy(pm->ptebuf, pte, PAGE_ALIGN((unsigned long)pte) - (unsigned long)pte);
+   pte_unmap(pte);
+   pte = pm->ptebuf;
+   #endif
+
     for (; addr != end; pte++, addr += PAGE_SIZE) {
         if (addr < pm->next)
             continue;
```



```
out_free:
+#endif
    kfree(page);
out:
    put_task_struct(task);
```

--

Mathematics is the supreme nostalgia of our time.
