

**[PATCH 16/21] mm: rss = file\_rss + anon\_rss**

**Source:** <http://linux.derkeiler.com/Mailing-Lists/Kernel/2005-09/7352.html>

---

**From:** Hugh Dickins ([hugh\\_at\\_veritas.com](mailto:hugh_at_veritas.com))

**Date:** 09/25/05

Date: Sun, 25 Sep 2005 17:07:12 +0100 (BST)

To: Andrew Morton <[akpm@osdl.org](mailto:akpm@osdl.org)>

I was lazy when we added anon\_rss, and chose to change as few places as possible. So currently each anonymous page has to be counted twice, in rss and in anon\_rss. Which won't be so good if those are atomic counts in some configurations.

Change that around: keep file\_rss and anon\_rss separately, and add them together (with get\_mm\_rss macro) when the total is needed – reading two atomics is much cheaper than updating two atomics. And update anon\_rss upfront, typically in memory.c, not tucked away in page\_add\_anon\_rmap.

Signed-off-by: Hugh Dickins <[hugh@veritas.com](mailto:hugh@veritas.com)>

```

---
 fs/exec.c          | 2 +-
 fs/proc/array.c   | 2 +-
 fs/proc/task_mmu.c | 8 +++-----
 include/linux/sched.h | 4 +++-
 kernel/acct.c     | 2 +-
 kernel/fork.c     | 4 +++-
 mm/fremap.c       | 4 +++-
 mm/hugetlb.c      | 6 +++---
 mm/memory.c       | 31 ++++++-----
 mm/nommu.c        | 2 +-
 mm/rmap.c         | 8 +++-----
 mm/swapfile.c     | 2 +-
12 files changed, 38 insertions(+), 37 deletions(-)
--- mm15/fs/exec.c      2005-09-22 12:31:59.000000000 +0100
+++ mm16/fs/exec.c     2005-09-24 19:29:53.000000000 +0100
@@ -330,7 +330,7 @@ void install_arg_page(struct vm_area_str
     pte_unmap(pte);
     goto out;
 }
-   inc_mm_counter(mm, rss);
+   inc_mm_counter(mm, anon_rss);
   lru_cache_add_active(page);
   set_pte_at(mm, address, pte, pte_mkdirty(pte_mkwrite(mk_pte(
                                     page, vma->vm_page_prot))));
--- mm15/fs/proc/array.c 2005-09-21 12:16:44.000000000 +0100
+++ mm16/fs/proc/array.c 2005-09-24 19:29:53.000000000 +0100
@@ -438,7 +438,7 @@ static int do_task_stat(struct task_stru
     jiffies_to_clock_t(it_real_value),
     start_time,
     vsize,

```

## Linux-Kernel: [PATCH 16/21] mm: rss = file\_rss + anon\_rss

```

-         mm ? get_mm_counter(mm, rss) : 0, /* you might want to shift this left 3 */
+         mm ? get_mm_rss(mm) : 0,
            rsslim,
            mm ? mm->start_code : 0,
            mm ? mm->end_code : 0,
--- mm15/fs/proc/task_mmu.c      2005-09-22 12:32:00.000000000 +0100
+++ mm16/fs/proc/task_mmu.c      2005-09-24 19:29:53.000000000 +0100
@@ -29,7 +29,7 @@ char *task_mem(struct mm_struct *mm, cha
        "VmPTE:\t%8lu kB\n",
        (mm->total_vm - mm->reserved_vm) << (PAGE_SHIFT-10),
        mm->locked_vm << (PAGE_SHIFT-10),
-       get_mm_counter(mm, rss) << (PAGE_SHIFT-10),
+       get_mm_rss(mm) << (PAGE_SHIFT-10),
        data << (PAGE_SHIFT-10),
        mm->stack_vm << (PAGE_SHIFT-10), text, lib,
        (PTRS_PER_PTE*sizeof(pte_t)*mm->nr_ptes) >> 10);
@@ -44,13 +44,11 @@ unsigned long task_vsize(struct mm_struct
int task_statm(struct mm_struct *mm, int *shared, int *text,
int *data, int *resident)
{
-   int rss = get_mm_counter(mm, rss);
-
-   *shared = rss - get_mm_counter(mm, anon_rss);
+   *shared = get_mm_counter(mm, file_rss);
+   *text = (PAGE_ALIGN(mm->end_code) - (mm->start_code & PAGE_MASK))
                                                >> PAGE_SHIFT;
+   *data = mm->total_vm - mm->shared_vm;
-   *resident = rss;
+   *resident = *shared + get_mm_counter(mm, anon_rss);
    return mm->total_vm;
}

--- mm15/include/linux/sched.h  2005-09-22 12:32:03.000000000 +0100
+++ mm16/include/linux/sched.h  2005-09-24 19:29:53.000000000 +0100
@@ -243,6 +243,8 @@ extern void arch_unmap_area_topdown(stru
#define add_mm_counter(mm, member, value) (mm)->_##member += (value)
#define inc_mm_counter(mm, member) (mm)->_##member++
#define dec_mm_counter(mm, member) (mm)->_##member--
+#define get_mm_rss(mm) ((mm)->_file_rss + (mm)->_anon_rss)
+
typedef unsigned long mm_counter_t;

struct mm_struct {
@@ -275,7 +277,7 @@ struct mm_struct {
    unsigned long exec_vm, stack_vm, reserved_vm, def_flags, nr_ptes;

    /* Special counters protected by the page_table_lock */
-   mm_counter_t _rss;
+   mm_counter_t _file_rss;
+   mm_counter_t _anon_rss;

    unsigned long saved_auxv[AT_VECTOR_SIZE]; /* for /proc/PID/auxv */
--- mm15/kernel/acct.c  2005-09-21 12:16:59.000000000 +0100
+++ mm16/kernel/acct.c  2005-09-24 19:29:53.000000000 +0100
@@ -553,7 +553,7 @@ void acct_update_integrals(struct task_s
    if (delta == 0)
        return;
    tsk->acct_stimexpd = tsk->stime;
-   tsk->acct_rss_mem1 += delta * get_mm_counter(tsk->mm, rss);
+   tsk->acct_rss_mem1 += delta * get_mm_rss(tsk->mm);
+   tsk->acct_vm_mem1 += delta * tsk->mm->total_vm;
}

```

## Linux-Kernel: [PATCH 16/21] mm: rss = file\_rss + anon\_rss

```

}
--- mm15/kernel/fork.c 2005-09-24 19:29:40.000000000 +0100
+++ mm16/kernel/fork.c 2005-09-24 19:29:53.000000000 +0100
@@ -321,7 +321,7 @@ static struct mm_struct * mm_init(struct
    INIT_LIST_HEAD(&mm->mmmlist);
    mm->core_waiters = 0;
    mm->nr_ptes = 0;
-   set_mm_counter(mm, rss, 0);
+   set_mm_counter(mm, file_rss, 0);
    set_mm_counter(mm, anon_rss, 0);
    spin_lock_init(&mm->page_table_lock);
    rwlock_init(&mm->ioctx_list_lock);
@@ -499,7 +499,7 @@ static int copy_mm(unsigned long clone_f
    if (retval)
        goto free_pt;

-   mm->hiwater_rss = get_mm_counter(mm, rss);
+   mm->hiwater_rss = get_mm_rss(mm);
    mm->hiwater_vm = mm->total_vm;

good_mm:
--- mm15/mm/fremap.c 2005-06-17 20:48:29.000000000 +0100
+++ mm16/mm/fremap.c 2005-09-24 19:29:53.000000000 +0100
@@ -39,7 +39,7 @@ static inline void zap_pte(struct mm_str
        set_page_dirty(page);
        page_remove_rmap(page);
        page_cache_release(page);
-       dec_mm_counter(mm, rss);
+       dec_mm_counter(mm, file_rss);
    }
} else {
@@ -92,7 +92,7 @@ int install_page(struct mm_struct *mm, s

    zap_pte(mm, vma, addr, pte);

-   inc_mm_counter(mm, rss);
+   inc_mm_counter(mm, file_rss);
    flush_icache_page(vma, page);
    set_pte_at(mm, addr, pte, mk_pte(page, prot));
    page_add_file_rmap(page);
--- mm15/mm/hugetlb.c 2005-09-24 19:26:24.000000000 +0100
+++ mm16/mm/hugetlb.c 2005-09-24 19:29:53.000000000 +0100
@@ -285,7 +285,7 @@ int copy_hugetlb_page_range(struct mm_st
    entry = *src_pte;
    ptepage = pte_page(entry);
    get_page(pte page);
-   add_mm_counter(dst, rss, HPAGE_SIZE / PAGE_SIZE);
+   add_mm_counter(dst, file_rss, HPAGE_SIZE / PAGE_SIZE);
    set_huge_pte_at(dst, addr, dst_pte, entry);
}
    spin_unlock(&src->page_table_lock);
@@ -323,7 +323,7 @@ void unmap_hugepage_range(struct vm_area

    page = pte_page(pte);
    put_page(page);
-   add_mm_counter(mm, rss, - (HPAGE_SIZE / PAGE_SIZE));
+   add_mm_counter(mm, file_rss, - (HPAGE_SIZE / PAGE_SIZE));
}
    flush_tlb_range(vma, start, end);
}
@@ -385,7 +385,7 @@ int hugetlb_prefault(struct address_spac

```

## Linux-Kernel: [PATCH 16/21] mm: rss = file\_rss + anon\_rss

```

                                goto out;
                                }
                                }
-   add_mm_counter(mm, rss, HPAGE_SIZE / PAGE_SIZE);
+   add_mm_counter(mm, file_rss, HPAGE_SIZE / PAGE_SIZE);
    set_huge_pte_at(mm, addr, pte, make_huge_pte(vma, page));
}
out:
--- mm15/mm/memory.c      2005-09-24 19:29:25.000000000 +0100
+++ mm16/mm/memory.c      2005-09-24 19:29:53.000000000 +0100
@@ -397,9 +397,10 @@ copy_one_pte(struct mm_struct *dst_mm, s
    pte = pte_mkclean(pte);
    pte = pte_mkold(pte);
    get_page(page);
-   inc_mm_counter(dst_mm, rss);
    if (PageAnon(page))
        inc_mm_counter(dst_mm, anon_rss);
+   else
+       inc_mm_counter(dst_mm, file_rss);
    set_pte_at(dst_mm, addr, dst_pte, pte);
    page_dup_rmap(page);
}
@@ -581,8 +582,8 @@ static void zap_pte_range(struct mmu_gat
                                set_page_dirty(page);
                                if (pte_young(ptent))
                                    mark_page_accessed(page);
+                               dec_mm_counter(tlb->mm, file_rss);
+                               }
-                               dec_mm_counter(tlb->mm, rss);
                                page_remove_rmap(page);
                                tlb_remove_page(tlb, page);
                                continue;
@@ -1290,13 +1291,15 @@ static int do_wp_page(struct mm_struct *
    spin_lock(&mm->page_table_lock);
    page_table = pte_offset_map(pmd, address);
    if (likely(pte_same(*page_table, orig_pte))) {
-       if (PageAnon(old_page))
-           dec_mm_counter(mm, anon_rss);
-       if (PageReserved(old_page))
-           inc_mm_counter(mm, rss);
-       else
+       inc_mm_counter(mm, anon_rss);
+       else {
            page_remove_rmap(old_page);
+
+           if (!PageAnon(old_page)) {
+               inc_mm_counter(mm, anon_rss);
+               dec_mm_counter(mm, file_rss);
+           }
        }
        flush_cache_page(vma, address, pfn);
        entry = mk_pte(new_page, vma->vm_page_prot);
        entry = maybe_mkwrite(pte_mkdirty(entry), vma);
@@ -1701,7 +1704,7 @@ static int do_swap_page(struct mm_struct
    /* The page isn't present yet, go ahead with the fault. */
-   inc_mm_counter(mm, rss);
+   inc_mm_counter(mm, anon_rss);
    pte = mk_pte(page, vma->vm_page_prot);
    if (write_access && can_share_swap_page(page)) {
        pte = maybe_mkwrite(pte_mkdirty(pte), vma);

```

## Linux-Kernel: [PATCH 16/21] mm: rss = file\_rss + anon\_rss

```

@@ -1774,7 +1777,7 @@ static int do_anonymous_page(struct mm_s
        page_cache_release(page);
        goto unlock;
    }
-    inc_mm_counter(mm, rss);
+    inc_mm_counter(mm, anon_rss);
    entry = mk_pte(page, vma->vm_page_prot);
    entry = maybe_mkwrite(pte_mkdirty(entry), vma);
    lru_cache_add_active(page);
@@ -1887,19 +1890,19 @@ retry:
    /*
    /* Only go through if we didn't race with anybody else... */
    if (pte_none(*page_table)) {
-        if (!PageReserved(new_page))
-            inc_mm_counter(mm, rss);
-
        flush_icache_page(vma, new_page);
        entry = mk_pte(new_page, vma->vm_page_prot);
        if (write_access)
            entry = maybe_mkwrite(pte_mkdirty(entry), vma);
        set_pte_at(mm, address, page_table, entry);
        if (anon) {
+            inc_mm_counter(mm, anon_rss);
+            lru_cache_add_active(new_page);
+            page_add_anon_rmap(new_page, vma, address);
        } else
+        } else if (!PageReserved(new_page)) {
+            inc_mm_counter(mm, file_rss);
+            page_add_file_rmap(new_page);
        }
    } else {
        /* One of our sibling threads was faster, back out. */
        page_cache_release(new_page);
@@ -2192,7 +2195,7 @@ EXPORT_SYMBOL(vmalloc_to_pfn);
void update_mem_hiwater(struct task_struct *tsk)
{
    if (tsk->mm) {
-        unsigned long rss = get_mm_counter(tsk->mm, rss);
+        unsigned long rss = get_mm_rss(tsk->mm);

        if (tsk->mm->hiwater_rss < rss)
            tsk->mm->hiwater_rss = rss;
--- mm15/mm/nommu.c      2005-09-22 12:32:03.000000000 +0100
+++ mm16/mm/nommu.c      2005-09-24 19:29:53.000000000 +0100
@@ -1080,7 +1080,7 @@ void update_mem_hiwater(struct task_stru
    unsigned long rss;

    if (likely(tsk->mm)) {
-        rss = get_mm_counter(tsk->mm, rss);
+        rss = get_mm_rss(tsk->mm);
        if (tsk->mm->hiwater_rss < rss)
            tsk->mm->hiwater_rss = rss;
        if (tsk->mm->hiwater_vm < tsk->mm->total_vm)
--- mm15/mm/rmap.c      2005-09-22 12:32:04.000000000 +0100
+++ mm16/mm/rmap.c      2005-09-24 19:29:53.000000000 +0100
@@ -445,8 +445,6 @@ void page_add_anon_rmap(struct page *pag
{
    BUG_ON(PageReserved(page));

-    inc_mm_counter(vma->vm_mm, anon_rss);
-
    if (atomic_inc_and_test(&page->_mapcount)) {

```

## Linux-Kernel: [PATCH 16/21] mm: rss = file\_rss + anon\_rss

```
struct anon_vma *anon_vma = vma->anon_vma;

@@ -561,9 +559,9 @@ static int try_to_unmap_one(struct page
    set_pte_at(mm, address, pte, swp_entry_to_pte(entry));
    BUG_ON(pte_file(*pte));
    dec_mm_counter(mm, anon_rss);
-   }
+   } else
+       dec_mm_counter(mm, file_rss);

-   dec_mm_counter(mm, rss);
    page_remove_rmap(page);
    page_cache_release(page);

@@ -667,7 +665,7 @@ static void try_to_unmap_cluster(unsigne

    page_remove_rmap(page);
    page_cache_release(page);
-   dec_mm_counter(mm, rss);
+   dec_mm_counter(mm, file_rss);
    (*mapcount)--;
}

--- mm15/mm/swapfile.c 2005-09-24 19:27:19.000000000 +0100
+++ mm16/mm/swapfile.c 2005-09-24 19:29:53.000000000 +0100
@@ -405,7 +405,7 @@ void free_swap_and_cache(swp_entry_t ent
static void unuse_pte(struct vm_area_struct *vma, pte_t *pte,
    unsigned long addr, swp_entry_t entry, struct page *page)
{
-   inc_mm_counter(vma->vm_mm, rss);
+   inc_mm_counter(vma->vm_mm, anon_rss);
    get_page(page);
    set_pte_at(vma->vm_mm, addr, pte,
        pte_mkold(mk_pte(page, vma->vm_page_prot)));
-
To unsubscribe from this list: send the line "unsubscribe linux-kernel" in
the body of a message to majordomo@vger.kernel.org
More majordomo info at http://vger.kernel.org/majordomo-info.html
Please read the FAQ at http://www.tux.org/lkml/
```