
Subject: Re: KVM support on 2.6.18-164.2.1.el5.028stab066.7 (CentOS5.4)

Posted by [jmslkn](#) on Wed, 09 Dec 2009 11:46:11 GMT

[View Forum Message](#) <> [Reply to Message](#)

I have the same panic here, when I start the ksm service. It seems there is a page that is allocated on different path without "charging" it, and when the memory manager try to free that page (free_hot_cold_page), it crashes.

The relevant source lines are here:

/kernel/ub/ub_mem.c

```
/*
 * Pages accounting
 */

int ub_page_charge(struct page *page, int order, gfp_t mask)
{
    struct user_beancounter *ub;
    unsigned long flags;

    ub = NULL;
    if (!(mask & __GFP_UBC))
        goto out;

    ub = get_beancounter(get_exec_ub());
    if (ub == NULL)
        goto out;

    local_irq_save(flags);
    if (ub_kmemsize_charge(ub, CHARGE_ORDER(order),
        (mask & __GFP_SOFT_UBC ? UB_SOFT : UB_HARD)))
        goto err;

    inc_pages_charged(ub, order);
    local_irq_restore(flags);
out:
    BUG_ON(page_ub(page) != NULL);
    page_ub(page) = ub;
    return 0;

err:
    local_irq_restore(flags);
    BUG_ON(page_ub(page) != NULL);
    put_beancounter(ub);
    return -ENOMEM;
}
```

```

void ub_page_uncharge(struct page *page, int order)
{
    struct user_beancounter *ub;
    unsigned long flags;

    ub = page_ub(page);
    if (ub == NULL)
        return;

    BUG_ON(ub->ub_magic != UB_MAGIC);
    dec_pages_charged(ub, order);
    local_irq_save(flags);
    ub_kmemsize_uncharge(ub, CHARGE_ORDER(order));
    local_irq_restore(flags);
    put_beancounter(ub);
    page_ub(page) = NULL;
}

```

ksm/ksm_main.c

```

static int try_to_merge_two_pages_alloc(struct mm_struct *mm1,
                                         struct page *page1,
                                         struct mm_struct *mm2,
                                         struct page *page2,
                                         unsigned long addr1,
                                         unsigned long addr2)
{
    struct vm_area_struct *vma;
    pgprot_t prot;
    int ret = 1;
    struct page *kpage;

    kpage = alloc_page(GFP_HIGHUSER);
    if (!kpage)
        return ret;
    down_read(&mm1->mmap_sem);
    vma = find_vma(mm1, addr1);
    if (!vma) {
        put_page(kpage);
        up_read(&mm1->mmap_sem);
        return ret;
    }
    prot = vma->vm_page_prot;
    pgprot_val(prot) &= ~_PAGE_RW;

    copy_user_highpage(kpage, page1, addr1);
    ret = try_to_merge_one_page(mm1, vma, page1, kpage, prot);
}

```

```
up_read(&mm1->mmap_sem);  
[.]
```

However the alloc_page calls __alloc_page function, that is "charged". Any idea?

File Attachments

1) [kvm.tar.gz](#), downloaded 398 times
