
Subject: Re: [RFC] [PATCH] memory controller statistics
Posted by KAMEZAWA Hiroyuki on Fri, 07 Sep 2007 05:02:34 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hi,

On Fri, 7 Sep 2007 12:39:42 +0900 (JST)
yamamoto@valinux.co.jp (YAMAMOTO Takashi) wrote:

```
>+enum mem_container_stat_index {  
>+/*  
>+ * for MEM_CONTAINER_TYPE_ALL, usage == pagecache + rss  
>+ */  
>+MEMCONT_STAT_PAGECACHE,  
>+MEMCONT_STAT_RSS,  
>+  
>+/*  
>+ * redundant; usage == charge - uncharge  
>+ */  
>+MEMCONT_STAT_CHARGE,  
>+MEMCONT_STAT_UNCHARGE,  
>+  
>+/*  
>+ * mostly for debug  
>+ */  
>+MEMCONT_STAT_ISOLATE,  
>+MEMCONT_STAT_ISOLATE_FAIL,  
>+MEMCONT_STAT_NSTATS,  
>+};  
>+
```

please add comments on each statistics name. It's uneasy to catch the meaning of ISOLATE/ISOLATE_FAIL without comments.

```
>+static const char * const mem_container_stat_desc[] = {  
>+[MEMCONT_STAT_PAGECACHE] = "page_cache",  
>+[MEMCONT_STAT_RSS] = "rss",  
>+[MEMCONT_STAT_CHARGE] = "charge",  
>+[MEMCONT_STAT_UNCHARGE] = "uncharge",  
>+[MEMCONT_STAT_ISOLATE] = "isolate",  
>+[MEMCONT_STAT_ISOLATE_FAIL] = "isolate_fail",  
>+};  
>+  
>+struct mem_container_stat {  
>+ atomic_t count[MEMCONT_STAT_NSTATS];  
>+};  
>+  
>+static void mem_container_stat_inc(struct mem_container_stat * stat,
```

```

> + enum mem_container_stat_index idx)
> +{
> +
> + atomic_inc(&stat->count[idx]);
> +}
> +
> +static void mem_container_stat_dec(struct mem_container_stat * stat,
> + enum mem_container_stat_index idx)
> +{
> +
> + atomic_dec(&stat->count[idx]);
> +}
> +

```

Can we do this accounting as mod_zone_page_state()(in mm/vmstat.c) ?
 (use per-cpu data for accounting.)

```

> +/* XXX hack; shouldn't be here. it really belongs to struct page_container. */
> +#define PAGE_CONTAINER_CACHE_BIT 0x1
> +#define PAGE_CONTAINER_CACHE (1 << PAGE_CONTAINER_CACHE_BIT)
> +

```

Is this used for remebering whether a page is charged as page-cache or not ?

```

> + page_assign_page_container_flags(page,
> +   is_cache ? PAGE_CONTAINER_CACHE : 0, pc);
> +
> + stat = &mem->stat;
> + if (is_cache) {
> +   mem_container_stat_inc(stat, MEMCONT_STAT_PAGECACHE);
> + } else {
> +   mem_container_stat_inc(stat, MEMCONT_STAT_RSS);
> + }

```

nitpick,in linux style, one-sentence block shouldn't have braces {}.

```

==  

if (is_cache)  

  mem_cont...  

else  

  mem_cont...  

==
```

```

> + mem_container_stat_inc(stat, MEMCONT_STAT_CHARGE);
>
```

```

> spin_lock_irqsave(&mem->lru_lock, flags);
> list_add(&pc->lru, &mem->active_list);
> @@ -377,6 +454,12 @@ err:
>     return -ENOMEM;
> }
>
> +int mem_container_charge(struct page *page, struct mm_struct *mm)
> +{
> +
> +    return mem_container_charge_common(page, mm, 0);
> +}
> +
> /*
> * See if the cached pages should be charged at all?
> */
> @@ -388,7 +471,7 @@ int mem_container_cache_charge(struct pa
>
>     mem = rcu_dereference(mm->mem_container);
>     if (mem->control_type == MEM_CONTAINER_TYPE_ALL)
> -        return mem_container_charge(page, mm);
> +        return mem_container_charge_common(page, mm, 1);
>     else
>         return 0;
>     }
> @@ -411,15 +494,29 @@ void mem_container_uncharge(struct page_
>     return;
>
>     if (atomic_dec_and_test(&pc->ref_cnt)) {
> +        struct mem_container_stat *stat;
> +        int is_cache;
> +
> +        page = pc->page;
> +        lock_page_container(page);
> +        mem = pc->mem_container;
> +        css_put(&mem->css);
> +        /* XXX */
>     }

```

This kind of comment is bad.

```

> +    is_cache = (page->page_container & PAGE_CONTAINER_CACHE) != 0;
>     page_assign_page_container(page, NULL);
>     unlock_page_container(page);
>     res_counter_uncharge(&mem->res, 1);
>
> +    stat = &mem->stat;
> +    if (is_cache) {
> +        mem_container_stat_dec(stat, MEMCONT_STAT_PAGECACHE);
> +    } else {
> +        mem_container_stat_dec(stat, MEMCONT_STAT_RSS);

```

```
> + }
> + mem_container_stat_inc(stat, MEMCONT_STAT_UNCHARGE);
> +
>   spin_lock_irqsave(&mem->lru_lock, flags);
> + BUG_ON(list_empty(&pc->lru));
```

Why this BUG_ON() is added ?

Thanks
-Kame

Containers mailing list
Containers@lists.linux-foundation.org
<https://lists.linux-foundation.org/mailman/listinfo/containers>
