

---

Subject: Re: [PATCH v3 10/13] memcg: use static branches when code not in use  
Posted by [Michal Hocko](#) on Mon, 01 Oct 2012 12:25:17 GMT

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

---

On Tue 18-09-12 18:04:07, Glauber Costa wrote:

[...]

```
> include/linux/memcontrol.h | 4 +---  
> mm/memcontrol.c          | 26 ++++++-----  
> 2 files changed, 27 insertions(+), 3 deletions(-)  
>  
> diff --git a/include/linux/memcontrol.h b/include/linux/memcontrol.h  
> index 82ede9a..4ec9fd5 100644  
> --- a/include/linux/memcontrol.h  
> +++ b/include/linux/memcontrol.h  
> @@ -22,6 +22,7 @@  
> #include <linux/cgroup.h>  
> #include <linux/vm_event_item.h>  
> #include <linux/hardirq.h>  
> +#include <linux/jump_label.h>  
>  
> struct mem_cgroup;  
> struct page_cgroup;  
> @@ -401,9 +402,10 @@ struct sock;  
> void sock_update_memcg(struct sock *sk);  
> void sock_release_memcg(struct sock *sk);  
>  
> +extern struct static_key memcg_kmem_enabled_key;  
> static inline bool memcg_kmem_enabled(void)  
> {  
> - return true;  
> + return static_key_false(&memcg_kmem_enabled_key);  
> }  
>  
> extern bool __memcg_kmem_newpage_charge(gfp_t gfp, struct mem_cgroup **memcg,  
> diff --git a/mm/memcontrol.c b/mm/memcontrol.c  
> index 720e4bb..aada601 100644  
> --- a/mm/memcontrol.c  
> +++ b/mm/memcontrol.c  
> @@ -467,6 +467,8 @@ struct mem_cgroup *mem_cgroup_from_css(struct  
cgroup_subsys_state *s)  
> #include <net/sock.h>  
> #include <net/ip.h>  
>  
> +struct static_key memcg_kmem_enabled_key;  
> +  
> static bool mem_cgroup_is_root(struct mem_cgroup *memcg);  
> static int memcg_charge_kmem(struct mem_cgroup *memcg, gfp_t gfp, u64 size);  
> static void memcg_uncharge_kmem(struct mem_cgroup *memcg, u64 size);
```

```

> @@ -624,6 +626,16 @@ void __memcg_kmem_uncharge_page(struct page *page, int order)
>     WARN_ON(mem_cgroup_is_root(memcg));
>     memcg_uncharge_kmem(memcg, PAGE_SIZE << order);
> }
> +
> +static void disarm_kmem_keys(struct mem_cgroup *memcg)
> +{
> +    if (memcg_kmem_is_accounted(memcg))
> +        static_key_slow_dec(&memcg_kmem_enabled_key);
> +
> +#else
> +static void disarm_kmem_keys(struct mem_cgroup *memcg)
> +{
> +
> #endif /* CONFIG_MEMCG_KMEM */
>
> #if defined(CONFIG_INET) && defined(CONFIG_MEMCG_KMEM)
> @@ -639,6 +651,12 @@ static void disarm_sock_keys(struct mem_cgroup *memcg)
> }
> #endif
>
> +static void disarm_static_keys(struct mem_cgroup *memcg)
> +{
> +    disarm_sock_keys(memcg);
> +    disarm_kmem_keys(memcg);
> +
> +
> static void drain_all_stock_async(struct mem_cgroup *memcg);
>
> static struct mem_cgroup_per_zone *
> @@ -4131,7 +4149,11 @@ static void memcg_update_kmem_limit(struct mem_cgroup
*memcg, u64 val)
> /*
> mutex_lock(&set_limit_mutex);
> if ((val != RESOURCE_MAX) && memcg_kmem_set_active(memcg)) {
> -
> + /*
> + * Once the static branch is enabled it will only be
> + * disabled when the last reference to memcg is gone.
> + */
> + static_key_slow_inc(&memcg_kmem_enabled_key);

```

I guess the reason why we do not need to inc also for children is that we do not inherit kmem\_accounted, right?

```

>     mem_cgroup_get(memcg);
>
>     for_each_mem_cgroup_tree(iter, memcg) {

```

```
> @@ -5066,7 +5088,7 @@ static void free_work(struct work_struct *work)
>     * to move this code around, and make sure it is outside
>     * the cgroup_lock.
>     */
> - disarm_sock_keys(memcg);
> + disarm_static_keys(memcg);
> if (size < PAGE_SIZE)
>     kfree(memcg);
> else
> --
> 1.7.11.4
>
> --
> To unsubscribe from this list: send the line "unsubscribe cgroups" in
> the body of a message to majordomo@vger.kernel.org
> More majordomo info at http://vger.kernel.org/majordomo-info.html
```

--  
Michał Hocko  
SUSE Labs

---