
Subject: Re: [PATCH v4 06/14] memcg: kmem controller infrastructure
Posted by [Michal Hocko](#) on Fri, 12 Oct 2012 09:47:26 GMT
[View Forum Message](#) <> [Reply to Message](#)

On Fri 12-10-12 13:13:04, Glauber Costa wrote:

[...]

> Just so we don't ping-pong in another submission:

>

> I changed memcontrol.h's memcg_kmem_newpage_charge to include:

>

> /* If the test is dying, just let it go. */

> if (unlikely(test_thread_flag(TIF_MEMDIE)

> || fatal_signal_pending(current)))

> return true;

OK

> diff --git a/mm/memcontrol.c b/mm/memcontrol.c

> index c32aaaf..72cf189 100644

> --- a/mm/memcontrol.c

> +++ b/mm/memcontrol.c

> +static int memcg_charge_kmem(struct mem_cgroup *memcg, gfp_t gfp, u64 size)

> +{

> + struct res_counter *fail_res;

> + struct mem_cgroup *_memcg;

> + int ret = 0;

> + bool may_oom;

> +

> + ret = res_counter_charge(&memcg->kmem, size, &fail_res);

> + if (ret)

> + return ret;

> +

> + /*

> + * Conditions under which we can wait for the oom_killer.

> + * We have to be able to wait, but also, if we can't retry,

> + * we obviously shouldn't go mess with oom.

> + */

> + may_oom = (gfp & __GFP_WAIT) && !(gfp & __GFP_NORETRY);

> +

> + _memcg = memcg;

> + ret = __mem_cgroup_try_charge(NULL, gfp, size >> PAGE_SHIFT,

> + &_memcg, may_oom);

> +

> + if (ret == -EINTR) {

> + /*

> + * __mem_cgroup_try_charge() chosed to bypass to root due to

> + * OOM kill or fatal signal. Since our only options are to

> + * either fail the allocation or charge it to this cgroup, do

```
> + * it as a temporary condition. But we can't fail. From a
> + * kmem/slab perspective, the cache has already been selected,
> + * by mem_cgroup_get_kmem_cache(), so it is too late to change
> + * our minds. This condition will only trigger if the task
> + * entered memcg_charge_kmem in a sane state, but was
> + * OOM-killed. during __mem_cgroup_try_charge. Tasks that are
> + * already dying when the allocation triggers should have been
> + * already directed to the root cgroup.
> + */
> + res_counter_charge_nofail(&memcg->res, size, &fail_res);
> + if (do_swap_account)
> +     res_counter_charge_nofail(&memcg->memsw, size,
> +         &fail_res);
> + ret = 0;
> + } else if (ret)
> +     res_counter_uncharge(&memcg->kmem, size);
> +
> + return ret;
> + }
```

OK

--

Michal Hocko
SUSE Labs
