

---

Subject: Re: [PATCH v3 09/16] sl[au]b: always get the cache from its page in kfree  
Posted by [Glauber Costa](#) on Fri, 21 Sep 2012 09:30:59 GMT

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

---

On 09/21/2012 01:33 PM, Pekka Enberg wrote:

> On Wed, Sep 19, 2012 at 10:42 AM, Glauber Costa <glommer@parallels.com> wrote:

>>>> index f2d760c..18de3f6 100644

>>>> --- a/mm/slab.c

>>>> +++ b/mm/slab.c

>>>> @@ -3938,9 +3938,12 @@ EXPORT\_SYMBOL(\_\_kmallocc);

>>>> \* Free an object which was previously allocated from this

>>>> \* cache.

>>>> \*/

>>>> -void kmem\_cache\_free(struct kmem\_cache \*cachep, void \*objp)

>>>> +void kmem\_cache\_free(struct kmem\_cache \*s, void \*objp)

>>>> {

>>>> unsigned long flags;

>>>> + struct kmem\_cache \*cachep = virt\_to\_cache(objp);

>>>> +

>>>> + VM\_BUG\_ON(!slab\_equal\_or\_parent(cachep, s));

>>>>

>>> This is an extremely hot path of the kernel and you are adding significant

>>> processing. Check how the benchmarks are influenced by this change.

>>> virt\_to\_cache can be a bit expensive.

>>>

>> Would it be enough for you to have a separate code path for

>> !CONFIG\_MEMCG\_KMEM?

>>

>> I don't really see another way to do it, aside from deriving the cache

>> from the object in our case. I am open to suggestions if you do.

>

> We should assume that most distributions enable CONFIG\_MEMCG\_KMEM,

> right? Therefore, any performance impact should be dependent on whether

> or not kmem memcg is \*enabled\* at runtime or not.

>

> Can we use the "static key" thingy introduced by tracing folks for this?

>

Yes.

I am already using static keys extensively in this patchset, and that is how I intend to handle this particular case.

---