
Subject: Re: [PATCH v2 04/11] kmem accounting basic infrastructure

Posted by [Michal Hocko](#) on Thu, 16 Aug 2012 15:25:44 GMT

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

On Wed 15-08-12 12:50:55, Ying Han wrote:

> On Tue, Aug 14, 2012 at 9:21 AM, Michal Hocko <mhocko@suse.cz> wrote:

> > On Thu 09-08-12 17:01:12, Glauber Costa wrote:

> > > This patch adds the basic infrastructure for the accounting of the slab

> > > caches. To control that, the following files are created:

> > >

> > > * memory.kmem.usage_in_bytes

> > > * memory.kmem.limit_in_bytes

> > > * memory.kmem.failcnt

> > > * memory.kmem.max_usage_in_bytes

> > >

> > > They have the same meaning of their user memory counterparts. They

> > > reflect the state of the "kmem" res_counter.

> > >

> > > The code is not enabled until a limit is set. This can be tested by the

> > > flag "kmem_accounted". This means that after the patch is applied, no

> > > behavioral changes exists for whoever is still using memcg to control

> > > their memory usage.

> > >

> > > We always account to both user and kernel resource_counters. This

> > > effectively means that an independent kernel limit is in place when the

> > > limit is set to a lower value than the user memory. A equal or higher

> > > value means that the user limit will always hit first, meaning that kmem

> > > is effectively unlimited.

> >

> > Well, it contributes to the user limit so it is not unlimited. It just

> > falls under a different limit and it tends to contribute less. This can

> > be quite confusing. I am still not sure whether we should mix the two

> > things together. If somebody wants to limit the kernel memory he has to

> > touch the other limit anyway. Do you have a strong reason to mix the

> > user and kernel counters?

>

> The reason to mix the two together is a compromise of the two use

> cases we've heard by far. In google, we only need one limit which

> limits u & k, and the reclaim kicks in when the total usage hits the

> limit.

>

> > My impression was that kernel allocation should simply fail while user

> > allocations might reclaim as well. Why should we reclaim just because of

> > the kernel allocation (which is unreclaimable from hard limit reclaim

> > point of view)?

>

> Some of kernel objects are reclaimable if we have per-memcg shrinker.

Agreed and I think we need that before this is merged as I state in other email.

> > I also think that the whole thing would get much simpler if those two
> > are split. Anyway if this is really a must then this should be
> > documented here.

>

> What would be the use case you have in your end?

I do not have any specific unfortunately but I would like to prevent us from closing other possible. I realize this sounds hand wavy and that is why I do not want to block this work but I think we should give it some time before this gets merged.

> --Ying

--

Michal Hocko
SUSE Labs
