Subject: Re: [PATCH v3 04/13] kmem accounting basic infrastructure
Posted by Glauber Costa on Mon, 01 Oct 2012 08:36:37 GMT
View Forum Message <> Reply to Message

On 09/30/2012 11:57 AM, Tejun Heo wrote:
> Hello, Glauber.
>
> On Thu, Sep 27, 2012 at 10:45:01PM +0400, Glauber Costa wrote:
>>> Can you please give other examples of cases where this type of issue
>>> exists (plenty of shared kernel data structure which is inherent to
>>> the workload at hand)? Until now, this has been the only example for
>>> this type of issues.
>>
>> Yes. the namespace related caches (*), all kinds of sockets and network
>> structures, other file system structures like file struct, vm areas, and
>> pretty much everything a full container does.
>>
>> (*) we run full userspace, so we have namespaces + cgroups combination.
>
> This is probably me being dumb but wouldn't resources used by full
> namespaces be mostly independent? Which parts get shared? Also, if
> you do full namespace, isn't it more likely that you would want fuller
> resource isolation too?
>

Not necessarily. Namespaces are pretty flexible. If you are using the network namespace, for instance, you can create interfaces, routes, addresses, etc. But because this deals with the network only, there is nothing unreasonable in saying that your webserver and database lives in the same network (which is a different network namespace), but are entitled to different memory limits - which is cgroups realm. With application-only containers being championed these days by multiple users, I would expect this situation to become non-negligible.

The full container scenario, of course, is very different. Most of the accesses tends to be local.

I will second what Michal said, since I believe this is also very important: User memory is completely at the control of the application, while kernel memory is not, and never will. It is perfectly fine to imagine applications that want its memory to be limited by a very predictable amount, and there are no reasons to believe that those cannot live in the same box as full containers - the biggest example of kmem interested folks. It could be, for instance, that a management tool for containers lives in there, and that application wants to be umem limited but not kmem limited. (If it goes touching files and data inside each container, for instance, it is obviously not local)

>>>> Mel suggestion of not allowing this to happen once the cgroup has tasks
>>>> takes care of this, and is something I thought of myself.
>>>
>>> You mean Michal's? It should also disallow switching if there are
>>> children cgroups, right?
>>
>> No, I meant Mel, quoting this:
>>
>> "Further I would expect that an administrator would be aware of these
>> limitations and set kmem_accounting at cgroup creation time before any
>> processes start. Maybe that should be enforced but it's not a
>> fundamental problem."
>>
>> But I guess it is pretty much the same thing Michal proposes, in essence.
>>
>> Or IOW, if your concern is with the fact that charges may have happened
>> in the past before this is enabled, we can make sure this cannot happen
>> by disallowing the limit to be set if currently unset (value changes are
>> obviously fine) if you have children or any tasks already in the group.
>
> Yeah, please do that.
>

I did already, patches soon! =)
