

---

Subject: Re: [PATCH v7 00/10] Request for Inclusion: per-cgroup tcp memory pressure

Posted by [Glauber Costa](#) on Mon, 05 Dec 2011 10:28:56 GMT

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

---

On 12/05/2011 07:51 AM, KAMEZAWA Hiroyuki wrote:

> On Mon, 5 Dec 2011 07:09:51 -0200

> Glauber Costa<glommer@parallels.com> wrote:

>

>> On 12/05/2011 12:06 AM, KAMEZAWA Hiroyuki wrote:

>>> On Fri, 2 Dec 2011 16:04:08 -0200

>>> Glauber Costa<glommer@parallels.com> wrote:

>>>

>>>> On 11/30/2011 12:11 AM, KAMEZAWA Hiroyuki wrote:

>>>>> On Tue, 29 Nov 2011 21:56:51 -0200

>>>>> Glauber Costa<glommer@parallels.com> wrote:

>>>>>

>>>>>> Hi,

>>>>>>

>>>>>> This patchset implements per-cgroup tcp memory pressure controls. It did not change  
>>>>>> significantly since last submission: rather, it just merges the comments Kame had.

>>>>>> Most of them are style-related and/or Documentation, but there are two real bugs he  
>>>>>> managed to spot (thanks)

>>>>>>

>>>>>> Please let me know if there is anything else I should address.

>>>>>>

>>>>>>

>>>>>> After reading all codes again, I feel some strange. Could you clarify ?

>>>>>>

>>>>>> Here.

>>>>>> ==

>>>>>> +void sock\_update\_memcg(struct sock \*sk)

>>>>>> +{

>>>>>> + /\* right now a socket spends its whole life in the same cgroup \*/

>>>>>> + if (sk->sk\_cgrp) {

>>>>>> + WARN\_ON(1);

>>>>>> + return;

>>>>>> + }

>>>>>> + if (static\_branch(&memcg\_socket\_limit\_enabled)) {

>>>>>> + struct mem\_cgroup \*memcg;

>>>>>> +

>>>>>> + BUG\_ON(!sk->sk\_prot->proto\_cgroup);

>>>>>> +

>>>>>> + rcu\_read\_lock();

>>>>>> + memcg = mem\_cgroup\_from\_task(current);

>>>>>> + if (!mem\_cgroup\_is\_root(memcg))

>>>>>> + sk->sk\_cgrp = sk->sk\_prot->proto\_cgroup(memcg);

>>>>>> + rcu\_read\_unlock();

```
>>>> ==
>>>>
>>>> sk->sk_cgrp is set to a memcg without any reference count.
>>>>
>>>> Then, no check for preventing rmdir() and freeing memcgroup.
>>>>
>>>> Is there some css_get() or mem_cgroup_get() somewhere ?
>>>>
>>>>
>>>> There were a css_get in the first version of this patchset. It was
>>>> removed, however, because it was deemed anti-intuitive to prevent rmdir,
>>>> since we can't know which sockets are blocking it, or do anything about
>>>> it. Or did I misunderstand something ?
>>>>
>>>>
>>> Maybe I misunderstood. Thank you. Ok, there is no css_get/put and
>>> rmdir() is allowed. But, hmm....what's guarding threads from stale
>>> pointer access ?
>>>
>>> Does a memory cgroup which is pointed by sk->sk_cgrp always exist ?
>>>
>> If I am not mistaken, yes, it will. (Ok, right now it won't)
>>
>> Reason is a cgroup can't be removed if it is empty.
>> To make it empty, you need to move the tasks away.
>>
>> So the sockets will be moved away as well when you do it. So right now
>> they are not, so it would then probably be better to increase a
>> reference count with a comment saying that it is temporary.
>>
>
> I'm sorry if I misunderstand.
>
> At task exit, __fput() will be called against file descriptors, yes.
> __fput() calls f_op->release() => inet_release() => tcp_close().
>
> But TCP socket may be alive after task exit until it gets down to
> protocol close. For example, until the all message in send buffer
> is acked, socket and tcp connection will not be disappear.
>
> In short, socket's lifetime is different from it's task's.
> So, there may be sockets which are not belongs to any task.
>
```

Yeah, you're right. I guess this is one more reason for us to just keep the memcg reference around.

---