Subject: Re: Re: [RFC][PATCH 2/7] RSS controller core
Posted by Kirill Korotaev on Mon, 12 Mar 2007 09:10:22 GMT
View Forum Message <> Reply to Message

Eric,

> And misses every resource sharing opportunity in sight.

that was my point too.

> Except for
> filtering the which pages are eligible for reclaim an RSS limit should
> not need to change the existing reclaim logic, and with things like the
> memory zones we have had that kind of restriction in the reclaim logic
> for a long time. So filtering out ineligible pages isn't anything new.

exactly this is implemented in the current patches from Pavel.
the only difference is that filtering is not done in general LRU list,
which is not effective, but via per-container LRU list.
So the pointer on the page structure does 2 things:
- fast reclamation
- correct uncharging of page from where it was charged
  (e.g. shared pages can be mapped first in one container, but the last unmap
   done from another one).

>>We need to work out what the requirements are before we can settle on an
>>implementation.
>
>
> If you are talking about RSS limits the term is well defined. The
> number of pages you can have mapped into your set of address space at
> any given time.
>
> Unless I'm totally blind that isn't what the patchset implements.

Ouch, what makes you think so?
The fact that a page mapped into 2 different processes is charged only once?
Imho it is much more correct then sum of process' RSS within container, due to:
1. it is clear how much container uses physical pages, not abstract items
2. shared pages are charged only once, so the sum of containers RSS is still
   about physical RAM.

> A
> true RSS limit over multiple processes has a lot of potential to be
> generally useful, is very understandable, doesn't affect kernel cache
> decisions so largely performance should not be affected. There is a
> little more overhead in the fault logic but that is a moderately
> expensive path anyway.

100% agree here.

>>You can actually do a form of overcommittment by allowing multiple
>>containers to share one or more of the zones. Whether that is sufficient
>>or suitable I don't know. That depends on the requirements, and we haven't
>>even discussed those, let alone agreed to them.
>
>
> Another really nasty issue is the container term as the resource guys
> are using the term in a subtlety different way then it has been used
> with namespaces leading to several threads where the participants talked
> past each other. We need a different term to designate the group of
> tasks a resource controller is dealing with.
taskgrp? resgrp?

> The whole filesystem interface also is over general and makes it too
> easy to express the hard things (like move an existing task from one
> group of tasks to another) leading to code complications.
the things which are not supported are easy to disable.

> On the up side I think the code the focus is likely in the right place
> to start delivering usable code.

Thanks,
Kirill
