Subject: Re: [RFC][PATCH] another swap controller for cgroup
Posted by yamamoto on Thu, 15 May 2008 06:23:18 GMT
View Forum Message <> Reply to Message

> On Tue, May 13, 2008 at 8:21 PM, YAMAMOTO Takashi
> <yamamoto@valinux.co.jp> wrote:
> > >
> > > Could you please mention what the limitations are? We could get those fixed or
> > > take another serious look at the mm->owner patches.
> >
> > for example, its callback can't sleep.
> >
>
> You need to be able to sleep in order to take mmap_sem, right?

yes.
besides that, i prefer not to hold a spinlock when traversing PTEs
as it can take somewhat long.

> Of course, having lots of datapath operations also take cgroup_mutex
> would be really painful, so it's not practical to use for things that
> can become non-attachable due to a process consuming some resources.
> This is part of the reason that I started working on the lock-mode
> patches that I sent out yesterday, in order to make finer-grained
> locking simpler. I'm going to rework those to make the locking more
> explicit, and I'll bear this use case in mind while I'm doing it.

thanks.

> A few comments on the patch:
>
> - you're not really limiting swap usage, you're limiting swapped-out
> address space. So it looks as though if a process has swapped out most
> of its address space, and forks a child, the total "swap" charge for
> the cgroup will double. Is that correct?

yes.

> If so, why is this better
> than charging for actual swap usage?

its behaviour is more determinstic and it uses less memory.
(than nishimura-san's one, which charges for actual swap usage.)

> - what will happen if someone creates non-NPTL threads, which share an
> mm but not a thread group (so each of them is a thread group leader)?

a thread which is most recently assigned to a cgroup will "win".

> - if you were to store a pointer in the page rather than the

"a pointer"? a pointer to what?

> swap_cgroup pointer, then (in combination with mm->owner) you wouldn't
> need to do the rebinding to the new swap_cgroup when a process moves
> to a different cgroup - you could instead keep a "swapped pte" count
> in the mm, and just charge that to the new cgroup and uncharge it from
> the old cgroup. You also wouldn't need to keep ref counts on the
> swap_cgroup.

PTE walking in my patch is for locking, not for "rebinding".
ie. to deal with concurrent swap activities.
the fact that each page table pages have their own locks (pte_lockptr)
complicated it.

> - ideally this wouldn't actually start charging until it was bound on
> to a cgroups hierarchy, although I guess that the performance of this
> is less important than something like the virtual address space
> controller, since once we start swapping we can expect performance to
> be bad anyway.

i agree.

YAMAMOTO Takashi

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers