Subject: Re: [RFC][PATCH 2/7] RSS controller core
Posted by Herbert Poetzl on Mon, 12 Mar 2007 00:41:52 GMT
View Forum Message <> Reply to Message

On Sun, Mar 11, 2007 at 06:04:28PM +0300, Pavel Emelianov wrote:
> Herbert Poetzl wrote:
> > On Sun, Mar 11, 2007 at 12:08:16PM +0300, Pavel Emelianov wrote:
> >> Herbert Poetzl wrote:
> >>> On Tue, Mar 06, 2007 at 02:00:36PM -0800, Andrew Morton wrote:
> >>>> On Tue, 06 Mar 2007 17:55:29 +0300
> >>>> Pavel Emelianov <xemul@sw.ru> wrote:
> >>>>
> >>>> +struct rss_container {
> >>>> + struct res_counter res;
> >>>> + struct list_head page_list;
> >>>> + struct container_subsys_state css;
> >>>> +};
> >>>> +
> >>>> +struct page_container {
> >>>> + struct page *page;
> >>>> + struct rss_container *cnt;
> >>>> + struct list_head list;
> >>>> +};
> >>>> ah. This looks good. I'll find a hunk of time to go through this
> >>>> work and through Paul's patches. It'd be good to get both patchsets
> >>>> lined up in -mm within a couple of weeks. But..
> >>> doesn't look so good for me, mainly becaus of the
> >>> additional per page data and per page processing
> >>>
> >>> on 4GB memory, with 100 guests, 50% shared for each
> >>> guest, this basically means ~1mio pages, 500k shared
> >>> and 1500k x sizeof(page_container) entries, which
> >>> roughly boils down to ~25MB of wasted memory ...
> >>>
> >>> increase the amount of shared pages and it starts
> >>> getting worse, but maybe I'm missing something here
> >> You are. Each page has only one page_container associated
> >> with it despite the number of containers it is shared
> >> between.
> >>
> >> We need to decide whether we want to do per-container memory
> >> limitation via these data structures, or whether we do it via
> >> a physical scan of some software zone, possibly based on Mel's
> >> patches.
> >>> why not do simple page accounting (as done currently
> >>> in Linux) and use that for the limits, without
> >>> keeping the reference from container to page?
> >> As I've already answered in my previous letter simple

> >> limiting w/o per-container reclamation and per-container
> >> oom killer isn't a good memory management. It doesn't allow
> >> to handle resource shortage gracefully.
> >
> > per container OOM killer does not require any container
> > page reference, you know _what_ tasks belong to the
> > container, and you know their _badness_ from the normal
> > OOM calculations, so doing them for a container is really
> > straight forward without having any page 'tagging'
>
> That's true. If you look at the patches you'll
> find out that no code in oom killer uses page 'tag'.

so what do we keep the context -> page reference
then at all?

> > for the reclamation part, please elaborate how that will
> > differ in a (shared memory) guest from what the kernel
> > currently does ...
>
> This is all described in the code and in the
> discussions we had before.

must have missed some of them, please can you
point me to the relevant threads ...

TIA,
Herbert

> > TIA,
> > Herbert
> >
> >> This patchset provides more grace way to handle this, but
> >> full memory management includes accounting of VMA-length
> >> as well (returning ENOMEM from system call) but we've decided
> >> to start with RSS.
> >>
> >>> best,
> >>> Herbert
> >>>
> >>>> _____
> >>>> Containers mailing list
> >>>> Containers@lists.osdl.org
> >>>> https://lists.osdl.org/mailman/listinfo/containers
> >>> -
> >>> To unsubscribe from this list: send the line "unsubscribe linux-kernel" in
> >>> the body of a message to majordomo@vger.kernel.org
> >>> More majordomo info at  http://vger.kernel.org/majordomo-info.html

> >>> Please read the FAQ at  http://www.tux.org/lkml/
> >>>
> >

_____

Containers mailing list
Containers@lists.osdl.org
https://lists.osdl.org/mailman/listinfo/containers