Subject: Re: Linux-VServer example results for sharing vs. separate mappings ...
Posted by Balbir Singh on Sun, 25 Mar 2007 09:50:35 GMT
View Forum Message <> Reply to Message

Andrew Morton wrote:
<snip>
> The problem is memory reclaim. A number of schemes which have been
> proposed require a per-container page reclaim mechanism - basically a
> separate scanner.
>
> This is a huge, huge, huge problem. The present scanner has been under
> development for over a decade and has had tremendous amounts of work and
> testing put into it. And it still has problems. But those problems will
> be gradually addressed.
>
> A per-container recaim scheme really really really wants to reuse all that
> stuff rather than creating a separate, parallel, new scanner which has the
> same robustness requirements, only has a decade less test and development
> done on it. And which permanently doubles our maintenance costs.
>

The current per-container reclaim scheme does reuse a lot of code. As far
as code maintenance is concerned, I think it should be easy to merge
some of the common functionality by abstracting them out as different
functions. The container smartness comes in only in the
container_isolate_pages(). This is an easy to understand function.

> So how do we reuse our existing scanner? With physical containers. One
> can envisage several schemes:
>
> a) slice the machine into 128 fake NUMA nodes, use each node as the
>   basic block of memory allocation, manage the binding between these
>   memory hunks and process groups with cpusets.
>
>   This is what google are testing, and it works.

Don't we break the global LRU with this scheme?

>
> b) Create a new memory abstraction, call it the "software zone", which
>   is mostly decoupled from the present "hardware zones". Most of the MM
>   is reworked to use "software zones". The "software zones" are
>   runtime-resizeable, and obtain their pages via some means from the
>   hardware zones. A container uses a software zone.
>

I think the problem would be figuring out where to allocate memory from?
What happens if a software zone spans across many hardware zones?

> c) Something else, similar to the above. Various schemes can be
>   envisaged, it isn't terribly important for this discussion.
>
>
> Let me repeat: this all has a huge upside in that it reuses the existing
> page reclaimation logic. And cpusets. Yes, we do discover glitches, but
> those glitches (such as Christoph's recent discovery of suboptimal
> interaction between cpusets and the global dirty ratio) get addressed, and
> we tend to strengthen the overall MM system as we address them.
>
>
> So what are the downsides? I think mainly the sharing issue:

I think binding the resource controller and the allocator might be
a bad idea, I tried experimenting with it and soon ran into some
hard to answer questions

1. How do we control the length of the zonelists that we need to
   allocate memory from (in a container)
2. Like you said, how do we share pages across zones (containers)
3. What happens to the global LRU behaviour
4. Do we need a per_cpu_pageset assoicated with containers
5. What do we do with unused memory in a zone, is it shared with
   other zones
6. Changing zones or creating an abstraction out of it is likely
   to impact the entire vm setup core, that is high risk, so
   do we really need to do it this way.

> But how much of a problem will it be *in practice*? Probably a lot of
> people just won't notice or care. There will be a few situations where it
> may be a problem, but perhaps we can address those? Forced migration of
> pages from one zone into another is possible. Or change the reclaim code
> so that a page which hasn't been referenced from a process within its
> hardware container is considered unreferenced (so it gets reclaimed). Or a
> manual nuke-all-the-pages knob which system administration tools can use.
> All doable, if we indeed have a demonstrable problem which needs to be
> addressed.
>
> And I do think it's worth trying to address these things, because the
> thought of implementing a brand new memory reclaim mechanism scares the
> pants off me.
>

The reclaim mechanism proposed *does not impact the non-container users*.
The only impact is container driven reclaim, like every other new feature
this can benefit from good testing in -mm. I believe we have something

simple and understandable to get us started. I would request you to consider merging the RSS controller and containers patches in -mm. If too many people complain or we see the problems that you foresee and our testing, enhancements and maintenance is unable to sort those problems, we know we'll have another approach to fall back upon :-) It'll also teach us to listen to the maintainers when they talk of design ;)

--
Warm Regards,
Balbir Singh
Linux Technology Center
IBM, ISTL

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