

---

Subject: Re: Supporting overcommit with the memory controller  
Posted by [KAMEZAWA Hiroyuki](#) on Thu, 06 Mar 2008 03:19:52 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

On Wed, 5 Mar 2008 18:54:52 -0800

"Paul Menage" <menage@google.com> wrote:

> On Wed, Mar 5, 2008 at 5:01 PM, KAMEZAWA Hiroyuki  
> <kamezawa.hiroyu@jp.fujitsu.com> wrote:  
> > > But to make this more interesting, there are plenty of jobs that will  
> > > happily fill as much pagecache as they have available. Even a job  
> > > that's just writing out logs will continually expand its pagecache  
> > > usage without anything to stop it, and so just keeping the reserved  
> > > pool at a fixed amount of free memory will result in the job expanding  
> > > even if it doesn't need to.  
> > It's current memory management style. "reclaim only when necessary".  
> >

>  
> Exactly - if the high-priority latency-sensitive job really needs that  
> extra memory, we want it to be able to automatically squash/kill the  
> low-priority job when memory runs low, and not suffer any latency  
> spikes. But if it doesn't actually need the memory, we'd rather use it  
> for low-priority batch stuff. The "no latency spikes" bit is important  
> - we don't want the high-priority job to get bogged down in  
> try\_to\_free\_pages() and out\_of\_memory() loops when it needs to  
> allocate memory.

>  
In our measurements(on RHEL5), setting dirty\_ratio to suitable value can  
help us to avoid \*long\* latency in most of \*usual\* situation.

(I'm sorry that I can't show the numbers, please try.)

Some mm people are trying to improve the kernel behavior under \*unusual\*  
situation. If you don't want any latency spikes for high priority processes,  
we'll have to try to make global page allocator handle priority of process/pages.

It seems what you really want is priority based file-cache control.  
I have no objectio to using cgroup as controller interface of it.

For avoiding spike, I'm now considering to support dirty\_ratio  
for memcg. (Now, it seems difficut.)

> > >  
> > Can Balbir's soft-limit patches help ?  
> >  
> > It reclamims each cgroup's pages to soft-limit if the system needs.  
> >  
> > Make limitation like this  
> >  
> > Assume 4G server.

> >                   Limit     soft-limit  
> > Not important Apss:     2G     100M  
> > Important Apps   :     3G     2.7G  
> >  
> > When the system memory reaches to the limit, each cgroup's memory usages will  
> > goes down to soft-limit. (And there will 1.3G of free pages in above example)  
> >  
>  
> Yes, that could be a useful part of the solution - I suspect we'd need  
> to have kswapd do the soft-limit push back as well as in  
> try\_to\_free\_pages(), to avoid the high-priority jobs getting stuck in  
> the reclaim code. It would also be nice if we had:  
>  
> - a way to have the soft-limit pushing kick in substantially \*before\*  
> the machine ran out of memory, to provide a buffer for the  
> high-priority jobs.  
>  
Maybe background-reclaim thread can be a help. (I'm now maintaining a patch.)

> - a way to measure the actual working set of a cgroup (which may be  
> smaller than its allocated memory if it has plenty of stale pagecache  
> pages allocated). Maybe refaults, or maybe usage-based information.  
>  
Hmm, current memory resource controller shows

- failcnt
- active/inactive
- rss/cache

I think we have enough infrastructure to account additional parameters.  
But I think support all vmstat members for memcg is a bit overkill.  
We'll have to choice what is necessary.

Thanks,  
-Kame

---

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

---