Subject: Re: [PATCH] memory.min_usage
Posted by yamamoto on Tue, 04 Dec 2007 07:01:21 GMT
View Forum Message <> Reply to Message

```
> > +int mem_cgroup_canreclaim(struct page *page, struct mem_cgroup *mem1)
> > +{
> > + struct page_cgroup *pc;
> > + int result = 1;
> > +
> > + if (mem1 != NULL)
> > +  return 1;
> > +
> > + lock_page_cgroup(page);
> > + pc = page_get_page_cgroup(page);
> > + if (pc) {
> > +  struct mem_cgroup *mem2 = pc->mem_cgroup;
> > +  unsigned long long min_usage;
> > +
> > +  BUG_ON(mem2 == NULL);
> > +  min_usage = mem2->min_usage;
> > +  if (min_usage != 0) {
> > +   unsigned long flags;
> > +
> > +   spin_lock_irqsave(&mem2->res.lock, flags);
> > +   if (mem2->res.usage <= min_usage)
> > +    result = 0;
> > +   spin_unlock_irqrestore(&mem2->res.lock, flags);
> > +  }
> > + }
> > + unlock_page_cgroup(page);
> > +
> > + return result;
> > +}
> > +
>
> How about adding lock_and_get_page_cgroup(page)/put_page_cgroup() ?
> ==
> struct page_cgroup *pc lock_and_get_page_cgroup(page)
> {
>  struct page_cgroup *pc;
>
>  lock_page_cgroup(page);
>  pc = page_get_page_cgroup(page);
>  if (atomic_inc_not_zero(&pc->ref_cnt))
>   pc = NULL;
>  unlock_page_cgroup(page);
>  return pc;
> }
```

>
> struct page_cgroup *pc put_page_cgroup(struct page_cgroup *pc)
> {
> mem_cgroup_uncharge(pc);
> }
> ==

i'm not sure if it's a good idea given that reference counting (atomic_foo)
has its own costs.

> BTW, how about add a status to res_counter ?
> My (based on your) current patch uses watermark_state.
>
> Maybe we can change it to be resource_state and show following status.
>
> RES_STATE_HIT_LIMIT,
> RES_STATE_ABOVE_HWATER,
> RES_STATE_ABOVE_LWATER,
> RES_STATE_STABLE, ?
> RES_STATE_BELOW_MIN,
>
> Useful ?

how about making res_counter use seq_lock?

> > static int alloc_mem_cgroup_per_zone_info(struct mem_cgroup *mem, int node)
> > {
> >   struct mem_cgroup_per_node *pn;
> > --- linux-2.6.24-rc3-mm2-swappiness/mm/vmscan.c.BACKUP2 2007-12-03
13:52:22.000000000 +0900
> > +++ linux-2.6.24-rc3-mm2-swappiness/mm/vmscan.c 2007-12-03 14:11:42.000000000 +0900
> > @@ -531,6 +531,11 @@ static unsigned long shrink_page_list(st
> >   referenced && page_mapping_inuse(page))
> >   goto activate_locked;
> >
> > +#ifdef CONFIG_CGROUP_MEM_CONT
> > + if (!mem_cgroup_canreclaim(page, sc->mem_cgroup))
> > +   goto activate_locked;
> > +#endif /* CONFIG_CGROUP_MEM_CONT */
> > +
>
> Maybe
> ==
> if (scan_global_lru(sc) && !
>   mem_cgroup_canreclaim(page, sc->mem-cgroup))
>   goto activate_locked:
> ==

i don't think the decision belongs to callers.
(at least it wasn't my intention.)

YAMAMOTO Takashi

_____

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