
Subject: Re: [PATCH 4/5] blk-throttle: track buffered and anonymous pages

Posted by [Andrea Righi](#) on Wed, 23 Feb 2011 08:37:40 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Tue, Feb 22, 2011 at 07:07:19PM -0500, Vivek Goyal wrote:

> On Wed, Feb 23, 2011 at 12:05:34AM +0100, Andrea Righi wrote:

> > On Tue, Feb 22, 2011 at 04:00:30PM -0500, Vivek Goyal wrote:

> > > On Tue, Feb 22, 2011 at 06:12:55PM +0100, Andrea Righi wrote:

> > > > Add the tracking of buffered (writeback) and anonymous pages.

> > > >

> > > > Dirty pages in the page cache can be processed asynchronously by the

> > > > per-bdi flusher kernel threads or by any other thread in the system,

> > > > according to the writeback policy.

> > > >

> > > > For this reason the real writes to the underlying block devices may

> > > > occur in a different IO context respect to the task that originally

> > > > generated the dirty pages involved in the IO operation. This makes

> > > > the tracking and throttling of writeback IO more complicate respect to

> > > > the synchronous IO from the blkio controller's point of view.

> > > >

> > > > The idea is to save the cgroup owner of each anonymous page and dirty

> > > > page in page cache. A page is associated to a cgroup the first time it

> > > > is dirtied in memory (for file cache pages) or when it is set as

> > > > swap-backed (for anonymous pages). This information is stored using the

> > > > page_cgroup functionality.

> > > >

> > > > Then, at the block layer, it is possible to retrieve the throttle group

> > > > looking at the bio_page(bio). If the page was not explicitly associated

> > > > to any cgroup the IO operation is charged to the current task/cgroup, as

> > > > it was done by the previous implementation.

> > > >

> > > > Signed-off-by: Andrea Righi <arighi@develer.com>

> > > > ---

> > > > block/blk-throttle.c | 87 ++++++

> > > > include/linux/blkdev.h | 26 ++++++

> > > > 2 files changed, 111 insertions(+), 2 deletions(-)

> > > >

> > > > diff --git a/block/blk-throttle.c b/block/blk-throttle.c

> > > > index 9ad3d1e..a50ee04 100644

> > > > --- a/block/blk-throttle.c

> > > > +++ b/block/blk-throttle.c

> > > > @@ -8,6 +8,10 @@

> > > > #include <linux/slab.h>

> > > > #include <linux/blkdev.h>

> > > > #include <linux/bio.h>

> > > > +#include <linux/memcontrol.h>

> > > > +#include <linux/mm_inline.h>

> > > > +#include <linux/pagemap.h>

```

>>>> +#include <linux/page_cgroup.h>
>>>> #include <linux/blktrace_api.h>
>>>> #include <linux/blk-cgroup.h>
>>>>
>>>> @@ -221,6 +225,85 @@ done:
>>>>     return tg;
>>>> }
>>>>
>>>> +static inline bool is_kernel_io(void)
>>>> +{
>>>> + return !!(current->flags & (PF_KTHREAD | PF_KSWAPD | PF_MEMALLOC));
>>>> +}
>>>> +
>>>> +static int throtl_set_page_owner(struct page *page, struct mm_struct *mm)
>>>> +{
>>>> + struct blkio_cgroup *blkcg;
>>>> + unsigned short id = 0;
>>>> +
>>>> + if (blkio_cgroup_disabled())
>>>> +     return 0;
>>>> + if (!mm)
>>>> +     goto out;
>>>> + rcu_read_lock();
>>>> + blkcg = task_to_blkio_cgroup(rcu_dereference(mm->owner));
>>>> + if (likely(blkcg))
>>>> +     id = css_id(&blkcg->css);
>>>> + rcu_read_unlock();
>>>> +out:
>>>> + return page_cgroup_set_owner(page, id);
>>>> +}
>>>> +
>>>> +int blk_throtl_set_anonpage_owner(struct page *page, struct mm_struct *mm)
>>>> +{
>>>> + return throtl_set_page_owner(page, mm);
>>>> +}
>>>> +EXPORT_SYMBOL(blk_throtl_set_anonpage_owner);
>>>> +
>>>> +int blk_throtl_set_filepage_owner(struct page *page, struct mm_struct *mm)
>>>> +{
>>>> + if (is_kernel_io() || !page_is_file_cache(page))
>>>> +     return 0;
>>>> + return throtl_set_page_owner(page, mm);
>>>> +}
>>>> +EXPORT_SYMBOL(blk_throtl_set_filepage_owner);
>>>>
>>> Why are we exporting all these symbols?
>>
>> Right. Probably a single one is enough:

```

```
> >  
> > int blk_throtl_set_page_owner(struct page *page,  
> > struct mm_struct *mm, bool anon);  
>  
> Who is going to use this single export? Which module?  
>
```

I was actually thinking at some filesystem modules, but I was wrong, because at the moment no one needs the export. I'll remove it in the next version of the patch.

Thanks,
-Andrea

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