Subject: Re: [PATCH 0/5] blk-throttle: writeback and swap IO control
Posted by Vivek Goyal on Tue, 22 Feb 2011 19:34:03 GMT
View Forum Message <> Reply to Message

On Tue, Feb 22, 2011 at 06:12:51PM +0100, Andrea Righi wrote:
> Currently the blkio.throttle controller only support synchronous IO requests.
> This means that we always look at the current task to identify the "owner" of
> each IO request.
>
> However dirty pages in the page cache can be wrote to disk 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 perspective.
>
> The same concept is also valid for anonymous pages involed in IO operations
> (swap).
>
> This patch allow to track the cgroup that originally dirtied each page in page
> cache and each anonymous page and pass these informations to the blk-throttle
> controller. These informations can be used to provide a better service level
> differentiation of buffered writes swap IO between different cgroups.
>

Hi Andrea,

Thanks for the patches. Before I look deeper into patches, had few
general queries/thoughts.

- So this requires memory controller to be enabled. Does it also require
  these to be co-mounted?

- Currently in throttling there is no limit on number of bios queued
  per group. I think this is not necessarily a very good idea because
  if throttling limits are low, we will build very long bio queues. So
  some AIO process can queue up lots of bios, consume lots of memory
  without getting blocked. I am sure there will be other side affects
  too. One of the side affects I noticed is that if an AIO process
  queues up too much of IO, and if I want to kill it now, it just hangs
  there for a really-2 long time (waiting for all the throttled IO
  to complete).

  So I was thinking of implementing either per group limit or per io
  context limit and after that process will be put to sleep. (something

like request descriptor mechanism).

If that's the case, then comes the question of what do to about kernel threads. Should they be blocked or not. If these are blocked then a fast group will also be indirectly throttled behind a slow group. If they are not then we still have the problem of too many bios queued in throttling layer.

- What to do about other kernel thread like kjournald which is doing IO on behalf of all the filesystem users. If data is also journalled then I think again everything got serialized and a faster group got backlogged behind a slower one.

- Two processes doing IO to same file and slower group will throttle IO for faster group also. (flushing is per inode).

I am not sure what are other common operations by kernel threads which can make IO serialized.

Thanks
Vivek

> Testcase
> ========
> - create a cgroup with 1MiB/s write limit:
> # mount -t cgroup -o blkio none /mnt/cgroup
> # mkdir /mnt/cgroup/foo
> # echo 8:0 $((1024 * 1024)) > /mnt/cgroup/foo/blkio.throttle.write_bps_device
>
> - move a task into the cgroup and run a dd to generate some writeback IO
>
> Results:
> - 2.6.38-rc6 vanilla:
> $ cat /proc/$$/cgroup
> 1:blkio:/foo
> $ dd if=/dev/zero of=zero bs=1M count=1024 &
> $ dstat -df
> --dsk/sda--
> read writ
> 0 19M
> 0 19M
> 0 0
> 0 0
> 0 19M
> ...
>
> - 2.6.38-rc6 + blk-throttle writeback IO control:

> $ cat /proc/$$/cgroup
> 1:blkio:/foo
> $ dd if=/dev/zero of=zero bs=1M count=1024 &
> $ dstat -df
> --dsk/sda--
> read writ
> 0 1024
> 0 1024
> 0 1024
> 0 1024
> 0 1024
> ...
>
> TODO
> ====
> - lots of testing
>
> Any feedback is welcome.
> -Andrea
>
> [PATCH 1/5] blk-cgroup: move blk-cgroup.h in include/linux/blk-cgroup.h
> [PATCH 2/5] blk-cgroup: introduce task_to_blkio_cgroup()
> [PATCH 3/5] page_cgroup: make page tracking available for blkio
> [PATCH 4/5] blk-throttle: track buffered and anonymous pages
> [PATCH 5/5] blk-throttle: buffered and anonymous page tracking instrumentation
>
> block/Kconfig | 2 +
> block/blk-cgroup.c | 15 ++-
> block/blk-cgroup.h | 335 ----------
> block/blk-throttle.c | 89 ++++++++++-
> block/cfq.h | 2 +-
> fs/buffer.c | 1 +
> include/linux/blk-cgroup.h | 341 ++
> include/linux/blkdev.h | 26 +++-
> include/linux/memcontrol.h | 6 +
> include/linux/mmzone.h | 4 +-
> include/linux/page_cgroup.h | 33 ++++-
> init/Kconfig | 4 +
> mm/Makefile | 3 +-
> mm/bounce.c | 1 +
> mm/filemap.c | 1 +
> mm/memcontrol.c | 6 +
> mm/memory.c | 5 +
> mm/page-writeback.c | 1 +
> mm/page_cgroup.c | 129 ++++++++++++++++--
> mm/swap_state.c | 2 +
> 20 files changed, 649 insertions(+), 357 deletions(-)

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