

Re: [PATCH] scsi_lib.c: continue after MEDIUM_ERROR

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2007-02/msg00563.html>

- *From:* Ric Wheeler <ric@xxxxxxx>
 - *Date:* Fri, 02 Feb 2007 07:20:15 -0500
-

James Bottomley wrote:

On Thu, 2007-02-01 at 15:02 -0500, Mark Lord wrote:

I believe you made the first change in response to my prodding at the time, when libata was not returning valid sense data (no LBA) for media errors. The SCSI EH handling of that was rather poor at the time, and so having it not retry the remaining sectors was actually a very good fix at the time.

But now, libata **does** return valid sense data for LBA/DMA drives, and the workaround from circa 2.6.16 is no longer the best we can do. Now that we know which sector failed, we ought to be able to skip over it, and continue with the rest of the merged request.

We can ... the big concern with your approach, which you haven't addressed is the time factor. For most SCSI devices, returning a fatal MEDIUM ERROR means we're out of remapping table, and also that there's probably a bunch of sectors on the track that are now out. Thus, there are almost always multiple sector failures. In linux, the average request size on a filesystem is around 64-128kb; thats 128-256 sectors. If we fail at the initial sector, we have to go through another 128-256 attempts, with the internal device retries, before we fail the entire request. Some devices can take a second or so for each read before they finally give up and decide they really can't read the sector, so you're looking at 2-5 minutes before the machine finally fails this one request ... and much worse for devices that retry more times.

Re: [PATCH] scsi_lib.c: continue after MEDIUM_ERROR

This is not the case on a read error – we commonly see transient errors on reads from disks. What we push our vendors to do is to try to keep the "worst case" response down to tens of seconds as they retry, etc internally with a device. When they take that long (and they do), adding retries up the stack can translate into minutes per sector.

The interesting point of this question is about the typically pattern of IO errors. On a read, it is safe to assume that you will have issues with some bounded numbers of adjacent sectors.

One thing that could be even better than the patch below, would be to have it perhaps skip the entire bio that includes the failed sector, rather than only the bad sector itself.

Er ... define "skip over the bio". A bio is simply a block representation for a bunch of sg elements coming in to the elevator. Mostly what we see in SCSI is a single bio per request, so skipping the bio is really the current behaviour (to fail the rest of the request).

This is really a tricky one – what can happen when we fail merged IO requests is really unpredictable behavior up at the application level since the IO error might not be at all relevant to my part of the request. Merging can produce a request that is much larger than any normal drive error.

I really like the idea of being able to set this kind of policy on a per drive instance since what you want here will change depending on what your system requirements are, what the system is trying to do (i.e., when trying to recover a failing but not dead yet disk, IO errors should be as quick as possible and we should choose an IO scheduler that does not combine IO's).

I think doing that might address most concerns expressed here. Have you got an alternate suggestion, James?

James

Re: [PATCH] scsi_lib.c: continue after MEDIUM_ERROR

—

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to majordomo@xxxxxxxxxxxxxxxxx

More majordomo info at <http://vger.kernel.org/majordomo-info.html>

Please read the FAQ at <http://www.tux.org/lkml/>