Subject: Re: [PATCH] incorrect error handling inside generic_file_direct_write
Posted by Andrew Morton on Tue, 12 Dec 2006 09:52:32 GMT
View Forum Message <> Reply to Message

On Tue, 12 Dec 2006 15:20:52 +0300
Dmitriy Monakhov <dmonakhov@sw.ru> wrote:

> > XFS (at least) can call generic_file_direct_write() with i_mutex not held.
> > And vmtruncate() expects i_mutex to be held.
> >
> > I guess a suitable solution would be to push this problem back up to the
> > callers: let them decide whether to run vmtruncate() and if so, to ensure
> > that i_mutex is held.
> >
> > The existence of generic_file_aio_write_nolock() makes that rather messy
> > though.
> This means we may call generic_file_aio_write_nolock() without i_mutex, right?
> but call trace is :
>   generic_file_aio_write_nolock()
>   ->generic_file_buffered_write() /* i_mutex not held here */
> but according to filemaps locking rules: mm/filemap.c:77
> ..
> * ->i_mutex   (generic_file_buffered_write)
> *   ->mmap_sem (fault_in_pages_readable->do_page_fault)
> ..
> I'm confused a litle bit, where is the truth?

xfs_write() calls generic_file_direct_write() without taking i_mutex for
O_DIRECT writes.