

Re: [RFC] fsblock

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2007-06/msg11582.html>

- *From:* David Chinner <dgc@xxxxxxx>
 - *Date:* Thu, 28 Jun 2007 08:35:48 +1000
-

On Wed, Jun 27, 2007 at 07:50:56AM -0400, Chris Mason wrote:

On Wed, Jun 27, 2007 at 07:32:45AM +0200, Nick Piggin wrote:

On Tue, Jun 26, 2007 at 08:34:49AM -0400, Chris Mason wrote:

On Tue, Jun 26, 2007 at 07:23:09PM +1000, David Chinner wrote:

On Tue, Jun 26, 2007 at 01:55:11PM +1000, Nick Piggin wrote:

[... fsblocks vs extent range mapping]

iomaps can double as range locks simply because iomaps are expressions of ranges within the file. Seeing as you can only access a given range exclusively to modify it, inserting an empty mapping into the tree as a range lock gives an effective method of allowing safe parallel reads, writes and allocation into the file.

The fsblocks and the vm page cache interface cannot be used to facilitate this because a radix tree is the wrong type of tree to store this information in. A sparse, range based tree (e.g. btree) is the right way to do this and it matches very well with a range based API.

I'm really not against the extent based page cache idea, but I

Re: [RFC] fsblock

kind of
assumed it would be too big a change for this kind of generic
setup. At
any rate, if we'd like to do it, it may be best to ditch the idea
of
"attach mapping information to a page", and switch to
"lookup mapping
information and range locking for a page".

Well the `get_block` equivalent API is extent based one now, and I'll
look at what is required in making `map_fsblock` a more generic call
that could be used for an extent-based scheme.

An extent based thing IMO really isn't appropriate as the main generic
layer here though. If it is really useful and popular, then it could
be turned into generic code and sit along side `fsblock` or underneath
`fsblock`...

Lets look at a typical example of how IO actually gets done today,
starting with `sys_write()`:

```
sys_write(file, buffer, 1MB)
for each page:
```