Subject: [PATCH 04/11] fuse: rework fuse_readpages()
Posted by Maxim Patlasov on Wed, 19 Sep 2012 16:32:16 GMT
View Forum Message <> Reply to Message

The patch uses 'nr_pages' argument of fuse_readpages() as heuristics for the number of page pointers to allocate.

This can be improved further by taking in consideration fc->max_read and gaps between page indices, but it's not clear whether it's worthy or not.

Signed-off-by: Maxim Patlasov <mpatlasov@parallels.com>
---
fs/fuse/file.c | 16 ++++++++++++++--
1 files changed, 14 insertions(+), 2 deletions(-)

```
diff --git a/fs/fuse/file.c b/fs/fuse/file.c
index 214e13e..a618371 100644
--- a/fs/fuse/file.c
+++ b/fs/fuse/file.c
@@ -641,6 +641,7 @@ struct fuse_fill_data {
  struct fuse_req *req;
  struct file *file;
  struct inode *inode;
+ unsigned nr_pages;
 };

 static int fuse_readpages_fill(void *_data, struct page *page)
@@ -656,16 +657,25 @@ static int fuse_readpages_fill(void *_data, struct page *page)
       (req->num_pages == FUSE_MAX_PAGES_PER_REQ ||
        (req->num_pages + 1) * PAGE_CACHE_SIZE > fc->max_read ||
        req->pages[req->num_pages - 1]->index + 1 != page->index)) {
+  int nr_alloc = min_t(unsigned, data->nr_pages,
+         FUSE_MAX_PAGES_PER_REQ);
   fuse_send_readpages(req, data->file);
-  data->req = req = fuse_get_req(fc, FUSE_MAX_PAGES_PER_REQ);
+  data->req = req = fuse_get_req(fc, nr_alloc);
   if (IS_ERR(req)) {
    unlock_page(page);
    return PTR_ERR(req);
   }
  }
+
+ if (WARN_ON(req->num_pages >= req->max_pages)) {
+  fuse_put_request(fc, req);
+  return -EIO;
+ }
+
  page_cache_get(page);
```

```
  req->pages[req->num_pages] = page;
  req->num_pages++;
+ data->nr_pages--;
  return 0;
 }

@@ -676,6 +686,7 @@ static int fuse_readpages(struct file *file, struct address_space *mapping,
  struct fuse_conn *fc = get_fuse_conn(inode);
  struct fuse_fill_data data;
  int err;
+ int nr_alloc = min_t(unsigned, nr_pages, FUSE_MAX_PAGES_PER_REQ);

  err = -EIO;
  if (is_bad_inode(inode))
@@ -683,7 +694,8 @@ static int fuse_readpages(struct file *file, struct address_space *mapping,

  data.file = file;
  data.inode = inode;
- data.req = fuse_get_req(fc, FUSE_MAX_PAGES_PER_REQ);
+ data.req = fuse_get_req(fc, nr_alloc);
+ data.nr_pages = nr_pages;
  err = PTR_ERR(data.req);
  if (IS_ERR(data.req))
   goto out;
```