
Subject: [PATCH 10/11] fuse: optimize fuse_get_user_pages() - v2

Posted by [Maxim Patlasov](#) on Mon, 15 Oct 2012 12:03:51 GMT

[View Forum Message](#) <> [Reply to Message](#)

Let fuse_get_user_pages() pack as many iov-s to a single fuse_req as possible. This is very beneficial in case of iov[] consisting of many iov-s of relatively small sizes (e.g. PAGE_SIZE).

Changed in v2:

- renamed local vars in fuse_page_descs_length_init() to be more readable

Signed-off-by: Maxim Patlasov <mpatlasov@parallels.com>

fs/fuse/file.c | 94 ++++++-----
1 files changed, 58 insertions(+), 36 deletions(-)

diff --git a/fs/fuse/file.c b/fs/fuse/file.c

index db9efb5..4d30697 100644

--- a/fs/fuse/file.c

+++ b/fs/fuse/file.c

```
@@ -1047,13 +1047,24 @@ static void fuse_release_user_pages(struct fuse_req *req, int write)
 }
 }
```

```
-static inline void fuse_page_descs_length_init(struct fuse_req *req)
```

```
+static inline void fuse_page_descs_length_init(struct fuse_req *req,
```

```
+         unsigned index, int nr_pages)
```

```
{
```

```
- int i;
```

```
+ while (nr_pages-- > 0)
```

```
+ req->page_descs[index + nr_pages].length = PAGE_SIZE -
```

```
+ req->page_descs[index + nr_pages].offset;
```

```
+}
```

```
- for (i = 0; i < req->num_pages; i++)
```

```
- req->page_descs[i].length = PAGE_SIZE -
```

```
- req->page_descs[i].offset;
```

```
+static inline unsigned long fuse_get_ua(const struct iovec *iov,
```

```
+         size_t iov_offset)
```

```
+{
```

```
+ return (unsigned long)iov->iov_base + iov_offset;
```

```
+}
```

```
+
```

```
+static inline size_t fuse_get_fr_sz(const struct iovec *iov, size_t iov_offset,
```

```
+         size_t max_size)
```

```
+{
```

```
+ return min_t(size_t, iov->iov_len - iov_offset, max_size);
```

```
}
```

```

static int fuse_get_user_pages(struct fuse_req *req,
@@ -1062,14 +1073,12 @@ static int fuse_get_user_pages(struct fuse_req *req,
    size_t *iov_offset_p,
    size_t *nbytesp, int write)
{
- size_t nbytes = *nbytesp;
- size_t frag_size = min_t(size_t, nbytes, (*iov_pp)->iov_len - *iov_offset_p);
- unsigned long user_addr = (unsigned long)(*iov_pp)->iov_base + *iov_offset_p;
- unsigned offset = user_addr & ~PAGE_MASK;
- int npages;
+ size_t nbytes = 0; /* # bytes already packed in req */

    /* Special case for kernel I/O: can copy directly into the buffer */
    if (segment_eq(get_fs(), KERNEL_DS)) {
+ unsigned long user_addr = fuse_get_ua(*iov_pp, *iov_offset_p);
+
        if (write)
            req->in.args[1].value = (void *) user_addr;
        else
@@ -1077,42 +1086,55 @@ static int fuse_get_user_pages(struct fuse_req *req,

        (*iov_pp)++;
        (*nr_segs_p)--;
- *nbytesp = frag_size;
+ *nbytesp = fuse_get_fr_sz(*iov_pp, *iov_offset_p, *nbytesp);
        return 0;
    }

- nbytes = min_t(size_t, frag_size, FUSE_MAX_PAGES_PER_REQ << PAGE_SHIFT);
- npages = (nbytes + offset + PAGE_SIZE - 1) >> PAGE_SHIFT;
- npages = clamp(npages, 1, FUSE_MAX_PAGES_PER_REQ);
- npages = get_user_pages_fast(user_addr, npages, !write, req->pages);
- if (npages < 0)
-     return npages;
+ while (nbytes < *nbytesp && req->num_pages < FUSE_MAX_PAGES_PER_REQ) {
+     int npages;
+     unsigned long user_addr = fuse_get_ua(*iov_pp, *iov_offset_p);
+     unsigned offset = user_addr & ~PAGE_MASK;
+     size_t frag_size = fuse_get_fr_sz(*iov_pp, *iov_offset_p,
+         *nbytesp - nbytes);

- req->num_pages = npages;
- req->page_descs[0].offset = offset;
- fuse_page_descs_length_init(req);
+ int n = FUSE_MAX_PAGES_PER_REQ - req->num_pages;
+ frag_size = min_t(size_t, frag_size, n << PAGE_SHIFT);

```

```

- if (write)
- req->in.argpages = 1;
- else
- req->out.argpages = 1;
+ npages = (frag_size + offset + PAGE_SIZE - 1) >> PAGE_SHIFT;
+ npages = clamp(npages, 1, n);

- nbytes = (req->num_pages << PAGE_SHIFT) - req->page_descs[0].offset;
+ npages = get_user_pages_fast(user_addr, npages, !write,
+     &req->pages[req->num_pages]);
+ if (npages < 0)
+ return npages;

- if (frag_size < nbytes)
- req->page_descs[req->num_pages - 1].length -=
-     nbytes - frag_size;
+ frag_size = min_t(size_t, frag_size,
+     (npages << PAGE_SHIFT) - offset);
+ nbytes += frag_size;

- *nbytesp = min(frag_size, nbytes);
+ if (frag_size < (*iov_pp)->iiov_len - *iiov_offset_p) {
+     *iiov_offset_p += frag_size;
+ } else {
+     (*iov_pp)++;
+     (*nr_segs_p)--;
+     *iiov_offset_p = 0;
+ }

- if (*nbytesp < (*iov_pp)->iiov_len - *iiov_offset_p) {
-     *iiov_offset_p += *nbytesp;
- } else {
-     (*iov_pp)++;
-     (*nr_segs_p)--;
-     *iiov_offset_p = 0;
+ req->page_descs[req->num_pages].offset = offset;
+ fuse_page_descs_length_init(req, req->num_pages, npages);
+
+ req->num_pages += npages;
+ req->page_descs[req->num_pages - 1].length -=
+     (npages << PAGE_SHIFT) - offset - frag_size;
+ }

+ if (write)
+ req->in.argpages = 1;
+ else
+ req->out.argpages = 1;
+

```

```
+ *nbytesp = nbytes;
+
+ return 0;
+ }
```

```
@@ -1954,7 +1976,7 @@ long fuse_do_ioctl(struct file *file, unsigned int cmd, unsigned long
arg,
{
    memcpy(req->pages, pages, sizeof(req->pages[0]) * num_pages);
    req->num_pages = num_pages;
- fuse_page_descs_length_init(req);
+ fuse_page_descs_length_init(req, 0, req->num_pages);

    /* okay, let's send it to the client */
    req->in.h.opcode = FUSE_IOCTL;
```
