Subject: [PATCH 2/4] fuse: re-work fuse_get_user_pages() to operate on iovec[]
Posted by Maxim Patlasov on Fri, 20 Jul 2012 11:50:33 GMT
View Forum Message <> Reply to Message

Now fuse_get_user_pages() takes iovec[] as argument and packs so much data
from it to req->pages[] as possible.

Signed-off-by: Maxim Patlasov <mpatlasov@parallels.com>
---
 fs/fuse/file.c | 64 ++----------
 1 files changed, 47 insertions(+), 17 deletions(-)

diff --git a/fs/fuse/file.c b/fs/fuse/file.c
index b321a68..d84416d 100644
--- a/fs/fuse/file.c
+++ b/fs/fuse/file.c
@@ -1020,41 +1020,71 @@ static void fuse_release_user_pages(struct fuse_req *req, int write)
  }
 }

-static int fuse_get_user_pages(struct fuse_req *req, const char __user *buf,
+static int fuse_get_user_pages(struct fuse_req *req,
+   const struct iovec **iov_pp,
+   unsigned long *nr_segs_p,
+   size_t *iov_offset_p,
          size_t *nbytesp, int write)
 {
- size_t nbytes = *nbytesp;
- unsigned long user_addr = (unsigned long) buf;
- 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)) {
+ BUG_ON(*iov_offset_p);
   if (write)
-   req->in.args[1].value = (void *) user_addr;
+   req->in.args[1].value = (*iov_pp)->iov_base;
   else
-   req->out.args[0].value = (void *) user_addr;
+   req->out.args[0].value = (*iov_pp)->iov_base;

+ (*iov_pp)++;
+ (*nr_segs_p)--;
   return 0;
  }

```
- nbytes = min_t(size_t, nbytes, 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;
+ req->iovec = *iov_pp;
+ req->iov_offset = *iov_offset_p;

- req->num_pages = npages;
- req->page_offset = offset;
+ while (nbytes < *nbytesp && req->num_pages < FUSE_MAX_PAGES_PER_REQ) {
+  int npages;
+  unsigned long user_addr = (unsigned long)(*iov_pp)->iov_base +
+       *iov_offset_p;
+  unsigned offset = user_addr & ~PAGE_MASK;
+  size_t frag_size = min_t(size_t,
+      (*iov_pp)->iov_len - *iov_offset_p,
+      *nbytesp - nbytes);
+
+  int n = FUSE_MAX_PAGES_PER_REQ - req->num_pages;
+  frag_size = min_t(size_t, frag_size, n << PAGE_SHIFT);
+
+  npages = (frag_size + offset + PAGE_SIZE - 1) >> PAGE_SHIFT;
+  npages = clamp(npages, 1, n);
+
+  npages = get_user_pages_fast(user_addr, npages, !write,
+          &req->pages[req->num_pages]);
+  if (npages < 0)
+   return npages;
+
+  frag_size = min_t(size_t, frag_size,
+      (npages << PAGE_SHIFT) - offset);
+  nbytes += frag_size;
+
+  if (frag_size < (*iov_pp)->iov_len - *iov_offset_p) {
+   *iov_offset_p += frag_size;
+  } else {
+   (*iov_pp)++;
+   (*nr_segs_p)--;
+   *iov_offset_p = 0;
+  }
+
+  req->num_pages += npages;
+ }

  if (write)
   req->in.argpages = 1;
```

```
  else
   req->out.argpages = 1;

- nbytes = (req->num_pages << PAGE_SHIFT) - req->page_offset;
- *nbytesp = min(*nbytesp, nbytes);
+ *nbytesp = nbytes;

  return 0;
 }
```