Subject: Re: [PATCH 02/11] fuse: categorize fuse_get_req()
Posted by Miklos Szeredi on Wed, 24 Oct 2012 16:26:45 GMT
View Forum Message <> Reply to Message

Hi,

Sorry about the long delay in responding.

See comments below.

Thanks,
Miklos

Maxim Patlasov <mpatlasov@parallels.com> writes:

> The patch categorizes all fuse_get_req() invocations into two categories:
> - fuse_get_req_nopages(fc) - when caller doesn't care about req->pages
> - fuse_get_req(fc, n) - when caller need n page pointers (n > 0)
>
> Adding fuse_get_req_nopages() helps to avoid numerous fuse_get_req(fc, 0)
> scattered over code. Now it's clear from the first glance when a caller need
> fuse_req with page pointers.
>
> The patch doesn't make any logic changes. In multi-page case, it silly
> allocates array of FUSE_MAX_PAGES_PER_REQ page pointers. This will be amended
> by future patches.
>
> Signed-off-by: Maxim Patlasov <mpatlasov@parallels.com>
> ---
> fs/fuse/cuse.c | 2 +-
> fs/fuse/dev.c | 11 ++++++-----
> fs/fuse/dir.c | 38 ++++++++++++++++++++++----------------
> fs/fuse/file.c | 30 ++++++++++++++++++++----------
> fs/fuse/fuse_i.h | 17 ++++++++++++++---
> fs/fuse/inode.c | 2 +-
> 6 files changed, 56 insertions(+), 44 deletions(-)
>
> diff --git a/fs/fuse/cuse.c b/fs/fuse/cuse.c
> index 3426521..2f1d5dd 100644
> --- a/fs/fuse/cuse.c
> +++ b/fs/fuse/cuse.c
> @@ -411,7 +411,7 @@ static int cuse_send_init(struct cuse_conn *cc)
>
>   BUILD_BUG_ON(CUSE_INIT_INFO_MAX > PAGE_SIZE);
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req(fc, 1);

>   if (IS_ERR(req)) {
>   rc = PTR_ERR(req);
>   goto err;
> diff --git a/fs/fuse/dev.c b/fs/fuse/dev.c
> index 1b5fc73..3ad5570 100644
> --- a/fs/fuse/dev.c
> +++ b/fs/fuse/dev.c
> @@ -117,7 +117,7 @@ static void fuse_req_init_context(struct fuse_req *req)
>   req->in.h.pid = current->pid;
> }
>
> -struct fuse_req *fuse_get_req(struct fuse_conn *fc)
> +struct fuse_req *fuse_get_req(struct fuse_conn *fc, int npages)

Again, npages should be unsigned, AFAICS.

> {
>   struct fuse_req *req;
>   sigset_t oldset;
> @@ -136,7 +136,7 @@ struct fuse_req *fuse_get_req(struct fuse_conn *fc)
>   if (!fc->connected)
>   goto out;
>
> - req = fuse_request_alloc(FUSE_MAX_PAGES_PER_REQ);
> + req = fuse_request_alloc(npages);
>   err = -ENOMEM;
>   if (!req)
>   goto out;
> @@ -207,13 +207,14 @@ static void put_reserved_req(struct fuse_conn *fc, struct fuse_req
*req)
>   * filesystem should not have it's own file open. If deadlock is
>   * intentional, it can still be broken by "aborting" the filesystem.
>   */
> -struct fuse_req *fuse_get_req_nofail(struct fuse_conn *fc, struct file *file)
> +struct fuse_req *fuse_get_req_nofail_nopages(struct fuse_conn *fc,
> +   struct file *file)
> {
>   struct fuse_req *req;
>
>   atomic_inc(&fc->num_waiting);
>   wait_event(fc->blocked_waitq, !fc->blocked);
> - req = fuse_request_alloc(FUSE_MAX_PAGES_PER_REQ);
> + req = fuse_request_alloc(0);
>   if (!req)
>   req = get_reserved_req(fc, file);
>
> @@ -1563,7 +1564,7 @@ static int fuse_retrieve(struct fuse_conn *fc, struct inode *inode,
>   unsigned int offset;

```
>   size_t total_len = 0;
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req(fc, FUSE_MAX_PAGES_PER_REQ);
>   if (IS_ERR(req))
>    return PTR_ERR(req);
>
> diff --git a/fs/fuse/dir.c b/fs/fuse/dir.c
> index 324bc08..1929bfb 100644
> --- a/fs/fuse/dir.c
> +++ b/fs/fuse/dir.c
> @@ -178,7 +178,7 @@ static int fuse_dentry_revalidate(struct dentry *entry, unsigned int flags)
>     return -ECHILD;
>
>    fc = get_fuse_conn(inode);
> -  req = fuse_get_req(fc);
> +  req = fuse_get_req_nopages(fc);
>    if (IS_ERR(req))
>     return 0;
>
> @@ -271,7 +271,7 @@ int fuse_lookup_name(struct super_block *sb, u64 nodeid, struct qstr
*name,
>   if (name->len > FUSE_NAME_MAX)
>    goto out;
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req_nopages(fc);
>   err = PTR_ERR(req);
>   if (IS_ERR(req))
>    goto out;
> @@ -391,7 +391,7 @@ static int fuse_create_open(struct inode *dir, struct dentry *entry,
>   if (!forget)
>    goto out_err;
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req_nopages(fc);
>   err = PTR_ERR(req);
>   if (IS_ERR(req))
>    goto out_put_forget_req;
> @@ -592,7 +592,7 @@ static int fuse_mknod(struct inode *dir, struct dentry *entry, umode_t
mode,
>  {
>   struct fuse_mknod_in inarg;
>   struct fuse_conn *fc = get_fuse_conn(dir);
> - struct fuse_req *req = fuse_get_req(fc);
> + struct fuse_req *req = fuse_get_req_nopages(fc);
>   if (IS_ERR(req))
>    return PTR_ERR(req);
```

```
>
> @@ -623,7 +623,7 @@ static int fuse_mkdir(struct inode *dir, struct dentry *entry, umode_t
mode)
>  {
>   struct fuse_mkdir_in inarg;
>   struct fuse_conn *fc = get_fuse_conn(dir);
> - struct fuse_req *req = fuse_get_req(fc);
> + struct fuse_req *req = fuse_get_req_nopages(fc);
>   if (IS_ERR(req))
>    return PTR_ERR(req);
>
> @@ -647,7 +647,7 @@ static int fuse_symlink(struct inode *dir, struct dentry *entry,
>  {
>   struct fuse_conn *fc = get_fuse_conn(dir);
>   unsigned len = strlen(link) + 1;
> - struct fuse_req *req = fuse_get_req(fc);
> + struct fuse_req *req = fuse_get_req_nopages(fc);
>   if (IS_ERR(req))
>    return PTR_ERR(req);
>
> @@ -664,7 +664,7 @@ static int fuse_unlink(struct inode *dir, struct dentry *entry)
>  {
>   int err;
>   struct fuse_conn *fc = get_fuse_conn(dir);
> - struct fuse_req *req = fuse_get_req(fc);
> + struct fuse_req *req = fuse_get_req_nopages(fc);
>   if (IS_ERR(req))
>    return PTR_ERR(req);
>
> @@ -696,7 +696,7 @@ static int fuse_rmdir(struct inode *dir, struct dentry *entry)
>  {
>   int err;
>   struct fuse_conn *fc = get_fuse_conn(dir);
> - struct fuse_req *req = fuse_get_req(fc);
> + struct fuse_req *req = fuse_get_req_nopages(fc);
>   if (IS_ERR(req))
>    return PTR_ERR(req);
>
> @@ -723,7 +723,7 @@ static int fuse_rename(struct inode *olddir, struct dentry *oldent,
>   int err;
>   struct fuse_rename_in inarg;
>   struct fuse_conn *fc = get_fuse_conn(olddir);
> - struct fuse_req *req = fuse_get_req(fc);
> + struct fuse_req *req = fuse_get_req_nopages(fc);
>
>   if (IS_ERR(req))
>    return PTR_ERR(req);
> @@ -776,7 +776,7 @@ static int fuse_link(struct dentry *entry, struct inode *newdir,
```

```
>   struct fuse_link_in inarg;
>   struct inode *inode = entry->d_inode;
>   struct fuse_conn *fc = get_fuse_conn(inode);
> - struct fuse_req *req = fuse_get_req(fc);
> + struct fuse_req *req = fuse_get_req_nopages(fc);
>   if (IS_ERR(req))
>    return PTR_ERR(req);
>
> @@ -848,7 +848,7 @@ static int fuse_do_getattr(struct inode *inode, struct kstat *stat,
>   struct fuse_req *req;
>   u64 attr_version;
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req_nopages(fc);
>   if (IS_ERR(req))
>    return PTR_ERR(req);
>
> @@ -1029,7 +1029,7 @@ static int fuse_access(struct inode *inode, int mask)
>   if (fc->no_access)
>    return 0;
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req_nopages(fc);
>   if (IS_ERR(req))
>    return PTR_ERR(req);
>
> @@ -1167,7 +1167,7 @@ static int fuse_readdir(struct file *file, void *dstbuf, filldir_t filldir)
>   if (is_bad_inode(inode))
>    return -EIO;
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req(fc, 1);
>   if (IS_ERR(req))
>    return PTR_ERR(req);
>
> @@ -1197,7 +1197,7 @@ static char *read_link(struct dentry *dentry)
>  {
>   struct inode *inode = dentry->d_inode;
>   struct fuse_conn *fc = get_fuse_conn(inode);
> - struct fuse_req *req = fuse_get_req(fc);
> + struct fuse_req *req = fuse_get_req_nopages(fc);
>   char *link;
>
>   if (IS_ERR(req))
> @@ -1410,7 +1410,7 @@ static int fuse_do_setattr(struct dentry *entry, struct iattr *attr,
>   if (attr->ia_valid & ATTR_SIZE)
>    is_truncate = true;
>
```

> - req = fuse_get_req(fc);
> + req = fuse_get_req_nopages(fc);
>   if (IS_ERR(req))
>   return PTR_ERR(req);
>
> @@ -1518,7 +1518,7 @@ static int fuse_setxattr(struct dentry *entry, const char *name,
>   if (fc->no_setxattr)
>   return -EOPNOTSUPP;
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req_nopages(fc);
>   if (IS_ERR(req))
>   return PTR_ERR(req);
>
> @@ -1557,7 +1557,7 @@ static ssize_t fuse_getxattr(struct dentry *entry, const char *name,
>   if (fc->no_getxattr)
>   return -EOPNOTSUPP;
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req_nopages(fc);
>   if (IS_ERR(req))
>   return PTR_ERR(req);
>
> @@ -1609,7 +1609,7 @@ static ssize_t fuse_listxattr(struct dentry *entry, char *list, size_t size)
>   if (fc->no_listxattr)
>   return -EOPNOTSUPP;
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req_nopages(fc);
>   if (IS_ERR(req))
>   return PTR_ERR(req);
>
> @@ -1654,7 +1654,7 @@ static int fuse_removexattr(struct dentry *entry, const char *name)
>   if (fc->no_removexattr)
>   return -EOPNOTSUPP;
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req_nopages(fc);
>   if (IS_ERR(req))
>   return PTR_ERR(req);
>
> diff --git a/fs/fuse/file.c b/fs/fuse/file.c
> index 7423ea4..214e13e 100644
> --- a/fs/fuse/file.c
> +++ b/fs/fuse/file.c
> @@ -25,7 +25,7 @@ static int fuse_send_open(struct fuse_conn *fc, u64 nodeid, struct file
*file,
>   struct fuse_req *req;

> int err;
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req_nopages(fc);
> if (IS_ERR(req))
> return PTR_ERR(req);
>
> @@ -368,7 +368,7 @@ static int fuse_flush(struct file *file, fl_owner_t id)
> if (fc->no_flush)
> return 0;
>
> - req = fuse_get_req_nofail(fc, file);
> + req = fuse_get_req_nofail_nopages(fc, file);
> memset(&inarg, 0, sizeof(inarg));
> inarg.fh = ff->fh;
> inarg.lock_owner = fuse_lock_owner_id(fc, id);
> @@ -436,7 +436,7 @@ int fuse_fsync_common(struct file *file, loff_t start, loff_t end,
>
> fuse_sync_writes(inode);
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req_nopages(fc);
> if (IS_ERR(req)) {
> err = PTR_ERR(req);
> goto out;
> @@ -544,7 +544,7 @@ static int fuse_readpage(struct file *file, struct page *page)
> */
> fuse_wait_on_page_writeback(inode, page->index);
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req(fc, 1);
> err = PTR_ERR(req);
> if (IS_ERR(req))
> goto out;
> @@ -657,7 +657,7 @@ static int fuse_readpages_fill(void *_data, struct page *page)
> (req->num_pages + 1) * PAGE_CACHE_SIZE > fc->max_read ||
> req->pages[req->num_pages - 1]->index + 1 != page->index)) {
> fuse_send_readpages(req, data->file);
> - data->req = req = fuse_get_req(fc);
> + data->req = req = fuse_get_req(fc, FUSE_MAX_PAGES_PER_REQ);
> if (IS_ERR(req)) {
> unlock_page(page);
> return PTR_ERR(req);
> @@ -683,7 +683,7 @@ static int fuse_readpages(struct file *file, struct address_space
*mapping,
>
> data.file = file;
> data.inode = inode;

```
> - data.req = fuse_get_req(fc);
> + data.req = fuse_get_req(fc, FUSE_MAX_PAGES_PER_REQ);
>   err = PTR_ERR(data.req);
>   if (IS_ERR(data.req))
>    goto out;
> @@ -890,7 +890,7 @@ static ssize_t fuse_perform_write(struct file *file,
>    struct fuse_req *req;
>    ssize_t count;
>
> -  req = fuse_get_req(fc);
> +  req = fuse_get_req(fc, FUSE_MAX_PAGES_PER_REQ);
>    if (IS_ERR(req)) {
>     err = PTR_ERR(req);
>     break;
> @@ -1072,7 +1072,7 @@ ssize_t fuse_direct_io(struct file *file, const char __user *buf,
>   ssize_t res = 0;
>   struct fuse_req *req;
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req(fc, FUSE_MAX_PAGES_PER_REQ);
>   if (IS_ERR(req))
>    return PTR_ERR(req);
>
> @@ -1108,7 +1108,7 @@ ssize_t fuse_direct_io(struct file *file, const char __user *buf,
>     break;
>    if (count) {
>     fuse_put_request(fc, req);
> -   req = fuse_get_req(fc);
> +   req = fuse_get_req(fc, FUSE_MAX_PAGES_PER_REQ);
>     if (IS_ERR(req))
>      break;
>    }
> @@ -1470,7 +1470,7 @@ static int fuse_getlk(struct file *file, struct file_lock *fl)
>   struct fuse_lk_out outarg;
>   int err;
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req_nopages(fc);
>   if (IS_ERR(req))
>    return PTR_ERR(req);
>
> @@ -1505,7 +1505,7 @@ static int fuse_setlk(struct file *file, struct file_lock *fl, int flock)
>   if (fl->fl_flags & FL_CLOSE)
>    return 0;
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req_nopages(fc);
>   if (IS_ERR(req))
```

```
>    return PTR_ERR(req);
>
> @@ -1574,7 +1574,7 @@ static sector_t fuse_bmap(struct address_space *mapping, sector_t
block)
>   if (!inode->i_sb->s_bdev || fc->no_bmap)
>    return 0;
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req_nopages(fc);
>   if (IS_ERR(req))
>    return 0;
>
> @@ -1872,7 +1872,7 @@ long fuse_do_ioctl(struct file *file, unsigned int cmd, unsigned long
arg,
>    num_pages++;
>   }
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req(fc, FUSE_MAX_PAGES_PER_REQ);
>   if (IS_ERR(req)) {
>    err = PTR_ERR(req);
>    req = NULL;
> @@ -2075,7 +2075,7 @@ unsigned fuse_file_poll(struct file *file, poll_table *wait)
>    fuse_register_polled_file(fc, ff);
>   }
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req_nopages(fc);
>   if (IS_ERR(req))
>    return POLLERR;
>
> @@ -2193,7 +2193,7 @@ long fuse_file_fallocate(struct file *file, int mode, loff_t offset,
>   if (fc->no_fallocate)
>    return -EOPNOTSUPP;
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req_nopages(fc);
>   if (IS_ERR(req))
>    return PTR_ERR(req);
>
> diff --git a/fs/fuse/fuse_i.h b/fs/fuse/fuse_i.h
> index e686bf1..fd69d62 100644
> --- a/fs/fuse/fuse_i.h
> +++ b/fs/fuse/fuse_i.h
> @@ -677,14 +677,25 @@ struct fuse_req *fuse_request_alloc_nofs(int npages);
>  void fuse_request_free(struct fuse_req *req);
>
>  /**
```

> - * Get a request, may fail with -ENOMEM
> + * Get a request, may fail with -ENOMEM,
> + * caller should specify # elements in req->pages[] explicitly
> */
> -struct fuse_req *fuse_get_req(struct fuse_conn *fc);
> +struct fuse_req *fuse_get_req(struct fuse_conn *fc, int npages);

Ditto.

> +
> +/**
> + * Get a request, may fail with -ENOMEM,
> + * useful for callers who doesn't use req->pages[]
> + */
> +static inline struct fuse_req *fuse_get_req_nopages(struct fuse_conn *fc)
> +{
> + return fuse_get_req(fc, 0);
> +}
>
> /**
> * Gets a requests for a file operation, always succeeds
> */
> -struct fuse_req *fuse_get_req_nofail(struct fuse_conn *fc, struct file *file);
> +struct fuse_req *fuse_get_req_nofail_nopages(struct fuse_conn *fc,
> + struct file *file);
>
> /**
> * Decrement reference count of a request. If count goes to zero free
> diff --git a/fs/fuse/inode.c b/fs/fuse/inode.c
> index 3f399ba..efb6144 100644
> --- a/fs/fuse/inode.c
> +++ b/fs/fuse/inode.c
> @@ -411,7 +411,7 @@ static int fuse_statfs(struct dentry *dentry, struct kstatfs *buf)
> return 0;
> }
>
> - req = fuse_get_req(fc);
> + req = fuse_get_req_nopages(fc);
> if (IS_ERR(req))
> return PTR_ERR(req);
>