
Subject: [PATCH] ecryptfs: handles AOP_TRUNCATED_PAGE better

Posted by [Dmitriy Monakhov](#) on Fri, 23 Feb 2007 15:44:42 GMT

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

- In fact we don't have to fail if AOP_TRUNCATED_PAGE was returned from prepare_write or commit_write. It is better to retry attempt where it is possible.
- Rearrange ecryptfs_get_lower_page() error handling logic, make it more clean.

Signed-off-by: Dmitriy Monakhov <dmonakhov@openvz.org>

fs/ecryptfs/mmap.c | 28 ++++++-----

1 files changed, 16 insertions(+), 12 deletions(-)

diff --git a/fs/ecryptfs/mmap.c b/fs/ecryptfs/mmap.c

index 842497d..b8fc1ef 100644

--- a/fs/ecryptfs/mmap.c

+++ b/fs/ecryptfs/mmap.c

```
@@ -447,6 +447,7 @@ static int ecryptfs_write_inode_size_to_header(struct file *lower_file,
    const struct address_space_operations *lower_a_ops;
    u64 file_size;
```

+retry:

```
    header_page = grab_cache_page(lower_inode->i_mapping, 0);
    if (!header_page) {
        ecryptfs_printk(KERN_ERR, "grab_cache_page for "
@@ -457,9 +458,10 @@ static int ecryptfs_write_inode_size_to_header(struct file *lower_file,
        lower_a_ops = lower_inode->i_mapping->a_ops;
        rc = lower_a_ops->prepare_write(lower_file, header_page, 0, 8);
        if (rc) {
- if (rc == AOP_TRUNCATED_PAGE)
+ if (rc == AOP_TRUNCATED_PAGE) {
            ecryptfs_release_lower_page(header_page, 0);
- else
+ } else
+ goto retry;
        ecryptfs_release_lower_page(header_page, 1);
        goto out;
    }
@@ -474,9 +476,10 @@ static int ecryptfs_write_inode_size_to_header(struct file *lower_file,
    if (rc < 0)
        ecryptfs_printk(KERN_ERR, "Error committing header page "
            "write\n");
- if (rc == AOP_TRUNCATED_PAGE)
+ if (rc == AOP_TRUNCATED_PAGE) {
        ecryptfs_release_lower_page(header_page, 0);
- else
```

```

+ goto retry;
+ } else
    ecryptfs_release_lower_page(header_page, 1);
    lower_inode->i_mtime = lower_inode->i_ctime = CURRENT_TIME;
    mark_inode_dirty_sync(inode);
@@ -565,6 +568,7 @@ int ecryptfs_get_lower_page(struct page **lower_page, struct inode
*lower_inode,
{
    int rc = 0;

+retry:
    *lower_page = grab_cache_page(lower_inode->i_mapping, lower_page_index);
    if (!(*lower_page)) {
        rc = -EINVAL;
@@ -578,18 +582,18 @@ int ecryptfs_get_lower_page(struct page **lower_page, struct inode
*lower_inode,
        byte_offset,
        region_bytes);
    if (rc) {
- ecryptfs_printk(KERN_ERR, "prepare_write for "
+ if (rc == AOP_TRUNCATED_PAGE) {
+ ecryptfs_release_lower_page(*lower_page, 0);
+ goto retry;
+ } else {
+ ecryptfs_printk(KERN_ERR, "prepare_write for "
    "lower_page_index = [0x%.16x] failed; rc = "
    "[%d]\n", lower_page_index, rc);
- }
-out:
- if (rc && (*lower_page)) {
- if (rc == AOP_TRUNCATED_PAGE)
- ecryptfs_release_lower_page(*lower_page, 0);
- else
    ecryptfs_release_lower_page(*lower_page, 1);
- (*lower_page) = NULL;
+ (*lower_page) = NULL;
+ }
+ }
+out:
    return rc;
}

--
1.4.4.4

```
