

[PATCH][RFC] ext3: Handle ext[34]_journal_stop() failure

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2007-02/msg10102.html>

- From: Dmitriy Monakhov <dmonakhov@xxxxx>
- Date: Wed, 28 Feb 2007 19:46:15 +0300

Where are many places where xxxx_journal_stop() return code wasn't checked. Off cause xxxx_journal_stop() failed very rarely (and usually with fatal consequences), but this does'n meen it should not be checked.

For example most retry loops looks like follows:

```
ext3_journal_stop(handle);
if (err == -ENOSPC && ext3_should_retry_alloc(dir->i_sb, &retries))
goto retry;
```

It is realy insane do "retry" if ext3_journal_stop() has failed, because this may increase damage in case of real problem.

Signed-off-by: Monakhov Dmitriy <dmonakhov@xxxxxxxxxxx>

```
----
fs/ext3/acl.c | 12 ++++++-----
fs/ext3/inode.c | 22 ++++++-----
fs/ext3/ioctl.c | 12 ++++++-----
fs/ext3/namei.c | 48 ++++++-----
fs/ext3/xattr.c | 4 ++--
fs/ext4/acl.c | 12 ++++++-----
fs/ext4/inode.c | 22 ++++++-----
fs/ext4/ioctl.c | 12 ++++++-----
fs/ext4/namei.c | 48 ++++++-----
fs/ext4/xattr.c | 4 ++--
10 files changed, 134 insertions(+), 62 deletions(-)
```

```
diff --git a/fs/ext3/acl.c b/fs/ext3/acl.c
index 1e5038d..bbf4d8a 100644
--- a/fs/ext3/acl.c
+++ b/fs/ext3/acl.c
@@ -375,7 +375,7 @@ int
ext3_acl_chmod(struct inode *inode)
{
struct posix_acl *acl, *clone;
- int error;
+ int error, error2;

if (S_ISLNK(inode->i_mode))
return -EOPNOTSUPP;
```

[PATCH][RFC] ext3: Handle ext[34]_journal_stop() failure

```
@@ -402,7 +402,9 @@ ext3_acl_chmod(struct inode *inode)
goto out;
}
error = ext3_set_acl(handle, inode, ACL_TYPE_ACCESS, clone);
- ext3_journal_stop(handle);
+ error2 = ext3_journal_stop(handle);
+ if (error2 && (!error || error == -ENOSPC))
+ error = error2;
if (error == -ENOSPC &&
ext3_should_retry_alloc(inode->i_sb, &retries))
goto retry;
@@ -485,7 +487,7 @@ ext3_xattr_set_acl(struct inode *inode, int type, const void *value,
{
handle_t *handle;
struct posix_acl *acl;
- int error, retries = 0;
+ int error, error2, retries = 0;

if (!test_opt(inode->i_sb, POSIX_ACL))
return -EOPNOTSUPP;
@@ -509,7 +511,9 @@ retry:
if (IS_ERR(handle))
return PTR_ERR(handle);
error = ext3_set_acl(handle, inode, type, acl);
- ext3_journal_stop(handle);
+ error2 = ext3_journal_stop(handle);
+ if (error2 && (!error || error == -ENOSPC))
+ error = error2;
if (error == -ENOSPC && ext3_should_retry_alloc(inode->i_sb, &retries))
goto retry;

diff --git a/fs/ext3/inode.c b/fs/ext3/inode.c
index 816f0c5..68d1c71 100644
--- a/fs/ext3/inode.c
+++ b/fs/ext3/inode.c
@@ -962,7 +962,9 @@ static int ext3_get_block(struct inode *inode, sector_t iblock,
* Huge direct-io writes can hold off commits for long
* periods of time. Let this commit run.
*/
- ext3_journal_stop(handle);
+ ret = ext3_journal_stop(handle);
+ if (ret)
+ goto get_block;
handle = ext3_journal_start(inode, DIO_CREDITS);
if (IS_ERR(handle))
ret = PTR_ERR(handle);
@@ -2998,7 +3000,13 @@ int ext3_setattr(struct dentry *dentry, struct iattr *attr)
if (attr->ia_valid & ATTR_GID)
inode->i_gid = attr->ia_gid;
error = ext3_mark_inode_dirty(handle, inode);
- ext3_journal_stop(handle);
```

[PATCH][RFC] ext3: Handle ext[34]_journal_stop() failure

```
+ if (error) {
+ ext3_journal_stop(handle);
+ goto err_out;
+ }
+ error = ext3_journal_stop(handle);
+ if (error)
+ goto err_out;
+ }

if (S_ISREG(inode->i_mode) &&
@@ -3016,7 +3024,9 @@ int ext3_setattr(struct dentry *dentry, struct iattr *attr)
rc = ext3_mark_inode_dirty(handle, inode);
if (!error)
error = rc;
- ext3_journal_stop(handle);
+ rc = ext3_journal_stop(handle);
+ if (!error)
+ error = rc;
+ }

rc = inode_setattr(inode, attr);
@@ -3231,7 +3241,7 @@ int ext3_change_inode_journal_flag(struct inode *inode, int val)
{
journal_t *journal;
handle_t *handle;
- int err;
+ int err, err2;

/*
* We have to be very careful here: changing a data block's
@@ -3274,7 +3284,9 @@ int ext3_change_inode_journal_flag(struct inode *inode, int val)

err = ext3_mark_inode_dirty(handle, inode);
handle->h_sync = 1;
- ext3_journal_stop(handle);
+ err2 = ext3_journal_stop(handle);
+ if (!err)
+ err = err2;
ext3_std_error(inode->i_sb, err);

return err;
diff --git a/fs/ext3/ioctl.c b/fs/ext3/ioctl.c
index 9b8090d..c90999f 100644
--- a/fs/ext3/ioctl.c
+++ b/fs/ext3/ioctl.c
@@ -32,7 +32,7 @@ int ext3_ioctl (struct inode * inode, struct file * filp, unsigned int cmd,
return put_user(flags, (int __user *) arg);
case EXT3_IOC_SETFLAGS: {
handle_t *handle = NULL;
- int err;
+ int err, err2;
```

[PATCH][RFC] ext3: Handle ext[34]_journal_stop() failure

```
struct ext3_iloc iloc;
unsigned int oldflags;
unsigned int jflag;
@@ -100,7 +100,9 @@ int ext3_ioctl (struct inode * inode, struct file * filp, unsigned int cmd,

err = ext3_mark_iloc_dirty(handle, inode, &iloc);
flags_err:
- ext3_journal_stop(handle);
+ err2 = ext3_journal_stop(handle);
+ if (!err)
+ err = err2;
if (err) {
mutex_unlock(&inode->i_mutex);
return err;
@@ -119,7 +121,7 @@ flags_err:
handle_t *handle;
struct ext3_iloc iloc;
__u32 generation;
- int err;
+ int err, err2;

if ((current->fsuid != inode->i_uid) && !capable(CAP_FOWNER))
return -EPERM;
@@ -137,7 +139,9 @@ flags_err:
inode->i_generation = generation;
err = ext3_mark_iloc_dirty(handle, inode, &iloc);
}
- ext3_journal_stop(handle);
+ err2 = ext3_journal_stop(handle);
+ if (!err)
+ err = err2;
return err;
}
#ifdef CONFIG_JBD_DEBUG
diff --git a/fs/ext3/namei.c b/fs/ext3/namei.c
index 49159f1..1b41704 100644
--- a/fs/ext3/namei.c
+++ b/fs/ext3/namei.c
@@ -1645,7 +1645,7 @@ static int ext3_create (struct inode * dir, struct dentry * dentry, int mode,
{
handle_t *handle;
struct inode * inode;
- int err, retries = 0;
+ int err, err2, retries = 0;

retry:
handle = ext3_journal_start(dir, EXT3_DATA_TRANS_BLOCKS(dir->i_sb) +
@@ -1665,7 +1665,9 @@ retry:
ext3_set_aops(inode);
err = ext3_add_nondir(handle, dentry, inode);
}
```

[PATCH][RFC] ext3: Handle ext[34]_journal_stop() failure

```
- ext3_journal_stop(handle);
+ err2 = ext3_journal_stop(handle);
+ if (err2 && (!err || err == -ENOSPC))
+ err = err2;
if (err == -ENOSPC && ext3_should_retry_alloc(dir->i_sb, &retries))
goto retry;
return err;
@@ -1676,7 +1678,7 @@ static int ext3_mknod (struct inode * dir, struct dentry *dentry,
{
handle_t *handle;
struct inode *inode;
- int err, retries = 0;
+ int err, err2, retries = 0;

if (!new_valid_dev(rdev))
return -EINVAL;
@@ -1700,7 +1702,9 @@ retry:
#endif
err = ext3_add_nondir(handle, dentry, inode);
}
- ext3_journal_stop(handle);
+ err2 = ext3_journal_stop(handle);
+ if (err2 && (!err || err == -ENOSPC))
+ err = err2;
if (err == -ENOSPC && ext3_should_retry_alloc(dir->i_sb, &retries))
goto retry;
return err;
@@ -1712,7 +1716,7 @@ static int ext3_mkdir(struct inode * dir, struct dentry * dentry, int mode)
struct inode * inode;
struct buffer_head * dir_block;
struct ext3_dir_entry_2 * de;
- int err, retries = 0;
+ int err, err2, retries = 0;

if (dir->i_nlink >= EXT3_LINK_MAX)
return -EMLINK;
@@ -1774,7 +1778,9 @@ retry:
ext3_mark_inode_dirty(handle, dir);
d_instantiate(dentry, inode);
out_stop:
- ext3_journal_stop(handle);
+ err2 = ext3_journal_stop(handle);
+ if (err2 && (!err || err == -ENOSPC))
+ err = err2;
if (err == -ENOSPC && ext3_should_retry_alloc(dir->i_sb, &retries))
goto retry;
return err;
@@ -2000,7 +2006,7 @@ out_brelse:

static int ext3_rmdir (struct inode * dir, struct dentry *dentry)
{
```

[PATCH][RFC] ext3: Handle ext[34]_journal_stop() failure

```
- int retval;
+ int retval, err;
struct inode * inode;
struct buffer_head * bh;
struct ext3_dir_entry_2 * de;
@@ -2052,14 +2058,16 @@ static int ext3_rmdir (struct inode * dir, struct dentry *dentry)
ext3_mark_inode_dirty(handle, dir);

end_rmdir:
- ext3_journal_stop(handle);
+ err = ext3_journal_stop(handle);
+ if (err && (!retval || retval == -ENOSPC))
+ retval = err;
brelse (bh);
return retval;
}

static int ext3_unlink(struct inode * dir, struct dentry *dentry)
{
- int retval;
+ int retval, err;
struct inode * inode;
struct buffer_head * bh;
struct ext3_dir_entry_2 * de;
@@ -2106,7 +2114,9 @@ static int ext3_unlink(struct inode * dir, struct dentry *dentry)
retval = 0;

end_unlink:
- ext3_journal_stop(handle);
+ err = ext3_journal_stop(handle);
+ if (!retval)
+ retval = err;
brelse (bh);
return retval;
}
@@ -2116,7 +2126,7 @@ static int ext3_symlink (struct inode * dir,
{
handle_t *handle;
struct inode * inode;
- int l, err, retries = 0;
+ int l, err, err2, retries = 0;

l = strlen(symname)+1;
if (l > dir->i_sb->s_blocksize)
@@ -2161,7 +2171,9 @@ retry:
EXT3_I(inode)->i_disksize = inode->i_size;
err = ext3_add_nondir(handle, dentry, inode);
out_stop:
- ext3_journal_stop(handle);
+ err2 = ext3_journal_stop(handle);
+ if (err2 && (!err || err == -ENOSPC))
```

[PATCH][RFC] ext3: Handle ext[34]_journal_stop() failure

```
+ err = err2;
if (err == -ENOSPC && ext3_should_retry_alloc(dir->i_sb, &retries))
goto retry;
return err;
@@ -2172,7 +2184,7 @@ static int ext3_link (struct dentry * old_dentry,
{
handle_t *handle;
struct inode *inode = old_dentry->d_inode;
- int err, retries = 0;
+ int err, err2, retries = 0;

if (inode->i_nlink >= EXT3_LINK_MAX)
return -EMLINK;
@@ -2197,7 +2209,9 @@ retry:
atomic_inc(&inode->i_count);

err = ext3_add_nondir(handle, dentry, inode);
- ext3_journal_stop(handle);
+ err2 = ext3_journal_stop(handle);
+ if (err2 && (!err || err == -ENOSPC))
+ err = err2;
if (err == -ENOSPC && ext3_should_retry_alloc(dir->i_sb, &retries))
goto retry;
return err;
@@ -2218,7 +2232,7 @@ static int ext3_rename (struct inode * old_dir, struct dentry *old_dentry,
struct inode * old_inode, * new_inode;
struct buffer_head * old_bh, * new_bh, * dir_bh;
struct ext3_dir_entry_2 * old_de, * new_de;
- int retval;
+ int retval, err;

old_bh = new_bh = dir_bh = NULL;

@@ -2358,7 +2372,9 @@ end_rename:
brelse (dir_bh);
brelse (old_bh);
brelse (new_bh);
- ext3_journal_stop(handle);
+ err = ext3_journal_stop(handle);
+ if (!retval)
+ retval = err;
return retval;
}

diff --git a/fs/ext3/xattr.c b/fs/ext3/xattr.c
index 99857a4..18b74df 100644
--- a/fs/ext3/xattr.c
+++ b/fs/ext3/xattr.c
@@ -1047,11 +1047,11 @@ retry:
error = ext3_xattr_set_handle(handle, inode, name_index, name,
value, value_len, flags);
```

[PATCH][RFC] ext3: Handle ext[34]_journal_stop() failure

[PATCH][RFC] ext3: Handle ext[34]_journal_stop() failure

```
error2 = ext3_journal_stop(handle);
+ if (error2 && (!error || error == -ENOSPC))
+ error = error2;
if (error == -ENOSPC &&
ext3_should_retry_alloc(inode->i_sb, &retries))
goto retry;
- if (error == 0)
- error = error2;
}

return error;
diff --git a/fs/ext4/acl.c b/fs/ext4/acl.c
index 9e88254..10e02c1 100644
--- a/fs/ext4/acl.c
+++ b/fs/ext4/acl.c
@@ -375,7 +375,7 @@ int
ext4_acl_chmod(struct inode *inode)
{
struct posix_acl *acl, *clone;
- int error;
+ int error, error2;

if (S_ISLNK(inode->i_mode))
return -EOPNOTSUPP;
@@ -402,7 +402,9 @@ ext4_acl_chmod(struct inode *inode)
goto out;
}
error = ext4_set_acl(handle, inode, ACL_TYPE_ACCESS, clone);
- ext4_journal_stop(handle);
+ error2 = ext4_journal_stop(handle);
+ if (error2 && (!error || error == -ENOSPC))
+ error = error2;
if (error == -ENOSPC &&
ext4_should_retry_alloc(inode->i_sb, &retries))
goto retry;
@@ -485,7 +487,7 @@ ext4_xattr_set_acl(struct inode *inode, int type, const void *value,
{
handle_t *handle;
struct posix_acl *acl;
- int error, retries = 0;
+ int error, error2, retries = 0;

if (!test_opt(inode->i_sb, POSIX_ACL))
return -EOPNOTSUPP;
@@ -509,7 +511,9 @@ retry:
if (IS_ERR(handle))
return PTR_ERR(handle);
error = ext4_set_acl(handle, inode, type, acl);
- ext4_journal_stop(handle);
+ error2 = ext4_journal_stop(handle);
+ if (error2 && (!error || error == -ENOSPC))
```

[PATCH][RFC] ext3: Handle ext[34]_journal_stop() failure

```
+ error = error2;
if (error == -ENOSPC && ext4_should_retry_alloc(inode->i_sb, &retries))
goto retry;

diff --git a/fs/ext4/inode.c b/fs/ext4/inode.c
index 02735b1..937d625 100644
--- a/fs/ext4/inode.c
+++ b/fs/ext4/inode.c
@@ -961,7 +961,9 @@ static int ext4_get_block(struct inode *inode, sector_t iblock,
* Huge direct-io writes can hold off commits for long
* periods of time. Let this commit run.
*/
- ext4_journal_stop(handle);
+ ret = ext4_journal_stop(handle);
+ if (ret)
+ goto get_block;
handle = ext4_journal_start(inode, DIO_CREDITS);
if (IS_ERR(handle))
ret = PTR_ERR(handle);
@@ -3009,7 +3011,13 @@ int ext4_setattr(struct dentry *dentry, struct iattr *attr)
if (attr->ia_valid & ATTR_GID)
inode->i_gid = attr->ia_gid;
error = ext4_mark_inode_dirty(handle, inode);
- ext4_journal_stop(handle);
+ if (error) {
+ ext4_journal_stop(handle);
+ goto err_out;
+ }
+ error = ext4_journal_stop(handle);
+ if (error)
+ goto err_out;
}

if (S_ISREG(inode->i_mode) &&
@@ -3027,7 +3035,9 @@ int ext4_setattr(struct dentry *dentry, struct iattr *attr)
rc = ext4_mark_inode_dirty(handle, inode);
if (!error)
error = rc;
- ext4_journal_stop(handle);
+ rc = ext4_journal_stop(handle);
+ if (!error)
+ error = rc;
}

rc = inode_setattr(inode, attr);
@@ -3245,7 +3255,7 @@ int ext4_change_inode_journal_flag(struct inode *inode, int val)
{
journal_t *journal;
handle_t *handle;
- int err;
+ int err, err2;
```

[PATCH][RFC] ext3: Handle ext[34]_journal_stop() failure

```
/*
 * We have to be very careful here: changing a data block's
 @@ -3288,7 +3298,9 @@ int ext4_change_inode_journal_flag(struct inode *inode, int val)

err = ext4_mark_inode_dirty(handle, inode);
handle->h_sync = 1;
- ext4_journal_stop(handle);
+ err2 = ext4_journal_stop(handle);
+ if (!err)
+ err = err2;
ext4_std_error(inode->i_sb, err);

return err;
diff --git a/fs/ext4/ioctl.c b/fs/ext4/ioctl.c
index 500567d..9a62c9a 100644
--- a/fs/ext4/ioctl.c
+++ b/fs/ext4/ioctl.c
@@ -32,7 +32,7 @@ int ext4_ioctl (struct inode * inode, struct file * filp, unsigned int cmd,
return put_user(flags, (int __user *) arg);
case EXT4_IOC_SETFLAGS: {
handle_t *handle = NULL;
- int err;
+ int err, err2;
struct ext4_iloc iloc;
unsigned int oldflags;
unsigned int jflag;
@@ -100,7 +100,9 @@ int ext4_ioctl (struct inode * inode, struct file * filp, unsigned int cmd,

err = ext4_mark_iloc_dirty(handle, inode, &iloc);
flags_err:
- ext4_journal_stop(handle);
+ err2 = ext4_journal_stop(handle);
+ if (!err)
+ err = err2;
if (err) {
mutex_unlock(&inode->i_mutex);
return err;
@@ -119,7 +121,7 @@ flags_err:
handle_t *handle;
struct ext4_iloc iloc;
__u32 generation;
- int err;
+ int err, err2;

if ((current->fsuid != inode->i_uid) && !capable(CAP_FOWNER))
return -EPERM;
@@ -137,7 +139,9 @@ flags_err:
inode->i_generation = generation;
err = ext4_mark_iloc_dirty(handle, inode, &iloc);
}

```

[PATCH][RFC] ext3: Handle ext[34]_journal_stop() failure

```
- ext4_journal_stop(handle);
+ err2 = ext4_journal_stop(handle);
+ if (!err)
+ err = err2;
return err;
}
#ifdef CONFIG_JBD_DEBUG
diff --git a/fs/ext4/namei.c b/fs/ext4/namei.c
index e7e1d79..8c12a14 100644
--- a/fs/ext4/namei.c
+++ b/fs/ext4/namei.c
@@ -1643,7 +1643,7 @@ static int ext4_create (struct inode * dir, struct dentry * dentry, int mode,
{
handle_t *handle;
struct inode * inode;
- int err, retries = 0;
+ int err, err2, retries = 0;

retry:
handle = ext4_journal_start(dir, EXT4_DATA_TRANS_BLOCKS(dir->i_sb) +
@@ -1663,7 +1663,9 @@ retry:
ext4_set_aops(inode);
err = ext4_add_nondir(handle, dentry, inode);
}
- ext4_journal_stop(handle);
+ err2 = ext4_journal_stop(handle);
+ if (err2 && (!err || err == -ENOSPC))
+ err = err2;
if (err == -ENOSPC && ext4_should_retry_alloc(dir->i_sb, &retries))
goto retry;
return err;
@@ -1674,7 +1676,7 @@ static int ext4_mknod (struct inode * dir, struct dentry *dentry,
{
handle_t *handle;
struct inode *inode;
- int err, retries = 0;
+ int err, err2, retries = 0;

if (!new_valid_dev(rdev))
return -EINVAL;
@@ -1698,7 +1700,9 @@ retry:
#endif
err = ext4_add_nondir(handle, dentry, inode);
}
- ext4_journal_stop(handle);
+ err2 = ext4_journal_stop(handle);
+ if (err2 && (!err || err == -ENOSPC))
+ err = err2;
if (err == -ENOSPC && ext4_should_retry_alloc(dir->i_sb, &retries))
goto retry;
return err;
```

[PATCH][RFC] ext3: Handle ext[34]_journal_stop() failure

```
@@ -1710,7 +1714,7 @@ static int ext4_mkdir(struct inode * dir, struct dentry * dentry, int mode)
struct inode * inode;
struct buffer_head * dir_block;
struct ext4_dir_entry_2 * de;
- int err, retries = 0;
+ int err, err2, retries = 0;

if (dir->i_nlink >= EXT4_LINK_MAX)
return -EMLINK;
@@ -1772,7 +1776,9 @@ retry:
ext4_mark_inode_dirty(handle, dir);
d_instantiate(dentry, inode);
out_stop:
- ext4_journal_stop(handle);
+ err2 = ext4_journal_stop(handle);
+ if (err2 && (!err || err == -ENOSPC))
+ err = err2;
if (err == -ENOSPC && ext4_should_retry_alloc(dir->i_sb, &retries))
goto retry;
return err;
@@ -1998,7 +2004,7 @@ out_brelse:

static int ext4_rmdir (struct inode * dir, struct dentry *dentry)
{
- int retval;
+ int retval, err;
struct inode * inode;
struct buffer_head * bh;
struct ext4_dir_entry_2 * de;
@@ -2050,14 +2056,16 @@ static int ext4_rmdir (struct inode * dir, struct dentry *dentry)
ext4_mark_inode_dirty(handle, dir);

end_rmdir:
- ext4_journal_stop(handle);
+ err = ext4_journal_stop(handle);
+ if (err && (!retval || retval == -ENOSPC))
+ retval = err;
brelse (bh);
return retval;
}

static int ext4_unlink(struct inode * dir, struct dentry *dentry)
{
- int retval;
+ int retval, err;
struct inode * inode;
struct buffer_head * bh;
struct ext4_dir_entry_2 * de;
@@ -2104,7 +2112,9 @@ static int ext4_unlink(struct inode * dir, struct dentry *dentry)
retval = 0;
```

[PATCH][RFC] ext3: Handle ext[34]_journal_stop() failure

```
end_unlink:
- ext4_journal_stop(handle);
+ err = ext4_journal_stop(handle);
+ if (!retval)
+   retval = err;
brelse (bh);
return retval;
}
@@ -2114,7 +2124,7 @@ static int ext4_symlink (struct inode * dir,
{
handle_t *handle;
struct inode * inode;
- int l, err, retries = 0;
+ int l, err, err2, retries = 0;

l = strlen(symname)+1;
if (l > dir->i_sb->s_blocksize)
@@ -2159,7 +2169,9 @@ retry:
EXT4_I(inode)->i_disksize = inode->i_size;
err = ext4_add_nondir(handle, dentry, inode);
out_stop:
- ext4_journal_stop(handle);
+ err2 = ext4_journal_stop(handle);
+ if (err2 && (!err || err == -ENOSPC))
+   err = err2;
if (err == -ENOSPC && ext4_should_retry_alloc(dir->i_sb, &retries))
goto retry;
return err;
@@ -2170,7 +2182,7 @@ static int ext4_link (struct dentry * old_dentry,
{
handle_t *handle;
struct inode *inode = old_dentry->d_inode;
- int err, retries = 0;
+ int err, err2, retries = 0;

if (inode->i_nlink >= EXT4_LINK_MAX)
return -EMLINK;
@@ -2195,7 +2207,9 @@ retry:
atomic_inc(&inode->i_count);

err = ext4_add_nondir(handle, dentry, inode);
- ext4_journal_stop(handle);
+ err2 = ext4_journal_stop(handle);
+ if (err2 && (!err || err == -ENOSPC))
+   err = err2;
if (err == -ENOSPC && ext4_should_retry_alloc(dir->i_sb, &retries))
goto retry;
return err;
@@ -2216,7 +2230,7 @@ static int ext4_rename (struct inode * old_dir, struct dentry *old_dentry,
struct inode * old_inode, * new_inode;
struct buffer_head * old_bh, * new_bh, * dir_bh;
```

[PATCH][RFC] ext3: Handle ext[34]_journal_stop() failure

```
struct ext4_dir_entry_2 * old_de, * new_de;
- int retval;
+ int retval, err;
```

```
old_bh = new_bh = dir_bh = NULL;
```

```
@@ -2356,7 +2370,9 @@ end_rename:
br else (dir_bh);
br else (old_bh);
br else (new_bh);
- ext4_journal_stop(handle);
+ err = ext4_journal_stop(handle);
+ if (!retval)
+ retval = err;
return retval;
}
```

```
diff --git a/fs/ext4/xattr.c b/fs/ext4/xattr.c
```

```
index dc969c3..47d9ce5 100644
```

```
--- a/fs/ext4/xattr.c
```

```
+++ b/fs/ext4/xattr.c
```

```
@@ -1047,11 +1047,11 @@ retry:
```

```
error = ext4_xattr_set_handle(handle, inode, name_index, name,
value, value_len, flags);
```

```
error2 = ext4_journal_stop(handle);
```

```
+ if (error2 && (!error || error == -ENOSPC))
```

```
+ error = error2;
```

```
if (error == -ENOSPC &&
```

```
ext4_should_retry_alloc(inode->i_sb, &retries))
```

```
goto retry;
```

```
- if (error == 0)
```

```
- error = error2;
```

```
}
```

```
return error;
```

```
---
```

```
1.5.0.1
```

```
-
```

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to majordomo@xxxxxxxxxxxxxxxxxxx

More majordomo info at <http://vger.kernel.org/majordomo-info.html>

Please read the FAQ at <http://www.tux.org/lkml/>