Subject: [RFC PATCH ext3/ext4] orphan list corruption due bad inode
Posted by vaverin on Mon, 04 Jun 2007 05:19:10 GMT
View Forum Message <> Reply to Message

After ext3 orphan list check has been added into ext3_destroy_inode() (please see my previous patch) the following situation has been detected:
EXT3-fs warning (device sda6): ext3_unlink: Deleting nonexistent file (37901290), 0
Inode 00000101a15b7840: orphan list check failed!
00000773 6f665f00 74616d72 00000573 65725f00 06737270 66000000 616d726f
...
Call Trace: [<ffffffff80211ea9>] ext3_destroy_inode+0x79/0x90
[<ffffffff801a2b16>] sys_unlink+0x126/0x1a0
[<ffffffff80111479>] error_exit+0x0/0x81
[<ffffffff80110aba>] system_call+0x7e/0x83

First messages said that unlinked inode has i_nlink=0, then ext3_unlink() adds this inode into orphan list.

Second message means that this inode has not been removed from orphan list. Inode dump has showed that i_fop = &bad_file_ops and it can be set in make_bad_inode() only. Then I've found that ext3_read_inode() can call make_bad_inode() without any error/warning messages, for example in the following case:

```
...
        if (inode->i_nlink == 0) {
                if (inode->i_mode == 0 ||
                    !(EXT3_SB(inode->i_sb)->s_mount_state & EXT3_ORPHAN_FS)) {
                        /* this inode is deleted */
                        brelse (bh);
                        goto bad_inode;
...
```

Bad inode can live some time, ext3_unlink can add it to orphan list, but ext3_delete_inode() do not deleted this inode from orphan list. As result we can have orphan list corruption detected in ext3_destroy_inode().

However it is not clear for me how to fix this issue correctly.

As far as i see is_bad_inode() is called after iget() in all places excluding ext3_lookup() and ext3_get_parent(). I believe it makes sense to add bad inode check to these functions too and call iput if bad inode detected.

Signed-off-by: Vasily Averin <vvs@sw.ru>

```
diff --git a/fs/ext3/namei.c b/fs/ext3/namei.c
index 9bb046d..e3ac8c3 100644
--- a/fs/ext3/namei.c
+++ b/fs/ext3/namei.c
@@ -1019,6 +1019,11 @@ static struct dentry *ext3_lookup(struct inode * dir, struct dentry
```

```
*dentry, str

   if (!inode)
    return ERR_PTR(-EACCES);
+
+  if (is_bad_inode(inode)) {
+   iput(inode);
+   return ERR_PTR(-ENOENT);
+  }
  }
  return d_splice_alias(inode, dentry);
 }
@@ -1054,6 +1059,11 @@ struct dentry *ext3_get_parent(struct dentry *child)
  if (!inode)
   return ERR_PTR(-EACCES);

+ if (is_bad_inode(inode)) {
+  iput(inode);
+  return ERR_PTR(-ENOENT);
+ }
+
  parent = d_alloc_anon(inode);
  if (!parent) {
   iput(inode);
diff --git a/fs/ext4/namei.c b/fs/ext4/namei.c
index 4ec57be..70db358 100644
--- a/fs/ext4/namei.c
+++ b/fs/ext4/namei.c
@@ -1017,6 +1017,11 @@ static struct dentry *ext4_lookup(struct inode * dir, struct dentry
*dentry, str

   if (!inode)
    return ERR_PTR(-EACCES);
+
+  if (is_bad_inode(inode)) {
+   iput(inode);
+   return ERR_PTR(-ENOENT);
+  }
  }
  return d_splice_alias(inode, dentry);
 }
@@ -1052,6 +1057,11 @@ struct dentry *ext4_get_parent(struct dentry *child)
  if (!inode)
   return ERR_PTR(-EACCES);

+ if (is_bad_inode(inode)) {
+  iput(inode);
+  return ERR_PTR(-ENOENT);
```

```
+ }
+
  parent = d_alloc_anon(inode);
  if (!parent) {
   iput(inode);
```