Subject: Re: [PATCH] ext3: ext3_symlink should use GFP_NOFS allocations inside (ver. 3)
Posted by dev on Fri, 10 Mar 2006 09:06:10 GMT
View Forum Message <> Reply to Message

>>>Andrew,
>>>
>>>Fixed both comments from Al Viro (thanks, Al):
>>>- should have a separate helper
>>>- should pass 0 instead of GFP_KERNEL in page_symlink()
>>
>>>
>>>+ page = find_or_create_page(mapping, 0,
>>>+   mapping_gfp_mask(mapping) | gfp_mask);
>>
>>
>>
>>this does not work; GFP_NOFS has a bit *LESS* than GFP_KERNEL, not a bit
>>more. As such a | operation isn't going to be useful....
>>
>>(So I think that while Al's intention was good, the implication of it
>>isn't ;)
>
>
> Yup. page_symlink() needs to pass in mapping_gfp_mask(inode->i_mapping)
> and ext3 needs to pass in, umm,
>
> mapping_gfp_mask(inode->i_mapping) & ~__GFP_FS
>
> or
>
> GFP_NOFS|__GFP_HIGHMEM.
>
> preferably the former I guess.

This looks reasonable.
See the patch attached.

Thanks,
Kirill

```
--- ./fs/ext3/namei.c.symlnkfix 2006-03-10 10:24:05.000000000 +0300
+++ ./fs/ext3/namei.c 2006-03-10 12:06:00.000000000 +0300
@@ -2141,7 +2141,8 @@ retry:
    * We have a transaction open.  All is sweetness.  It also sets
    * i_size in generic_commit_write().
    */
```

```
-  err = page_symlink(inode, symname, l);
+  err = __page_symlink(inode, symname, l,
+    mapping_gfp_mask(inode->i_mapping) & ~__GFP_FS);
   if (err) {
    ext3_dec_count(handle, inode);
    ext3_mark_inode_dirty(handle, inode);
--- ./fs/namei.c.symlnkfix 2006-03-10 10:24:05.000000000 +0300
+++ ./fs/namei.c 2006-03-10 12:07:47.000000000 +0300
@@ -2613,13 +2613,15 @@ void page_put_link(struct dentry *dentry
  }
 }

-int page_symlink(struct inode *inode, const char *symname, int len)
+int __page_symlink(struct inode *inode, const char *symname, int len,
+  gfp_t gfp_mask)
 {
  struct address_space *mapping = inode->i_mapping;
- struct page *page = grab_cache_page(mapping, 0);
+ struct page *page;
  int err = -ENOMEM;
  char *kaddr;

+ page = find_or_create_page(mapping, 0, gfp_mask);
  if (!page)
   goto fail;
  err = mapping->a_ops->prepare_write(NULL, page, 0, len-1);
@@ -2654,6 +2656,12 @@ fail:
  return err;
 }

+int page_symlink(struct inode *inode, const char *symname, int len)
+{
+ return __page_symlink(inode, symname, len,
+   mapping_gfp_mask(inode->i_mapping));
+}
+
 struct inode_operations page_symlink_inode_operations = {
  .readlink = generic_readlink,
  .follow_link = page_follow_link_light,
@@ -2672,6 +2680,7 @@ EXPORT_SYMBOL(lookup_one_len);
 EXPORT_SYMBOL(page_follow_link_light);
 EXPORT_SYMBOL(page_put_link);
 EXPORT_SYMBOL(page_readlink);
+EXPORT_SYMBOL(__page_symlink);
 EXPORT_SYMBOL(page_symlink);
 EXPORT_SYMBOL(page_symlink_inode_operations);
 EXPORT_SYMBOL(path_lookup);
--- ./include/linux/fs.h.symlnkfix 2006-03-10 10:24:05.000000000 +0300
```

```
+++ ./include/linux/fs.h 2006-03-10 10:27:40.000000000 +0300
@@ -1669,6 +1669,8 @@ extern int vfs_follow_link(struct nameid
 extern int page_readlink(struct dentry *, char __user *, int);
 extern void *page_follow_link_light(struct dentry *, struct nameidata *);
 extern void page_put_link(struct dentry *, struct nameidata *, void *);
+extern int __page_symlink(struct inode *inode, const char *symname, int len,
+  gfp_t gfp_mask);
 extern int page_symlink(struct inode *inode, const char *symname, int len);
 extern struct inode_operations page_symlink_inode_operations;
 extern int generic_readlink(struct dentry *, char __user *, int);
```