Subject: [PATCH 03/14] sysfs: Move all of inode initialization into sysfs_init_inode
Posted by ebiederm on Tue, 31 Jul 2007 10:22:56 GMT
View Forum Message <> Reply to Message

Besides being a good general cleanup the ultimate effect
of this patch is that sysfs_get_inode now gives you a
usable inode that doesn't need any further initialization
to be useful.

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>

---
 fs/sysfs/dir.c | 37 ----------------------------------
 fs/sysfs/inode.c | 48 ++---
 fs/sysfs/mount.c | 5 -----
 3 files changed, 45 insertions(+), 45 deletions(-)

diff --git a/fs/sysfs/dir.c b/fs/sysfs/dir.c
index b27a38c..834ed74 100644
--- a/fs/sysfs/dir.c
+++ b/fs/sysfs/dir.c
@@ -741,24 +741,12 @@ int sysfs_create_dir(struct kobject * kobj)
  return error;
 }

-static int sysfs_count_nlink(struct sysfs_dirent *sd)
-{
- struct sysfs_dirent *child;
- int nr = 0;
-
- for (child = sd->s_children; child; child = child->s_sibling)
- if (sysfs_type(child) == SYSFS_DIR)
-   nr++;
- return nr + 2;
-}
-
 static struct dentry * sysfs_lookup(struct inode *dir, struct dentry *dentry,
     struct nameidata *nd)
 {
  struct sysfs_dirent * parent_sd = dentry->d_parent->d_fsdata;
  struct dentry *result = NULL;
  struct sysfs_dirent * sd;
- struct bin_attribute *bin_attr;
  struct inode *inode;
  int found = 0;

@@ -782,31 +770,6 @@ static struct dentry * sysfs_lookup(struct inode *dir, struct dentry
*dentry,

```
   goto out;
  }

- if (inode->i_state & I_NEW) {
-  /* initialize inode according to type */
-  switch (sysfs_type(sd)) {
-  case SYSFS_DIR:
-   inode->i_op = &sysfs_dir_inode_operations;
-   inode->i_fop = &sysfs_dir_operations;
-   inode->i_nlink = sysfs_count_nlink(sd);
-   break;
-  case SYSFS_KOBJ_ATTR:
-   inode->i_size = PAGE_SIZE;
-   inode->i_fop = &sysfs_file_operations;
-   break;
-  case SYSFS_KOBJ_BIN_ATTR:
-   bin_attr = sd->s_elem.bin_attr.bin_attr;
-   inode->i_size = bin_attr->size;
-   inode->i_fop = &bin_fops;
-   break;
-  case SYSFS_KOBJ_LINK:
-   inode->i_op = &sysfs_symlink_inode_operations;
-   break;
-  default:
-   BUG();
-  }
- }
-
   sysfs_instantiate(dentry, inode);
   sysfs_attach_dentry(sd, dentry);

diff --git a/fs/sysfs/inode.c b/fs/sysfs/inode.c
index 9671164..e832a5d 100644
--- a/fs/sysfs/inode.c
+++ b/fs/sysfs/inode.c
@@ -123,8 +123,22 @@ static inline void set_inode_attr(struct inode * inode, struct iattr * iattr)
   */
 static struct lock_class_key sysfs_inode_imutex_key;

+static int sysfs_count_nlink(struct sysfs_dirent *sd)
+{
+ struct sysfs_dirent *child;
+ int nr = 0;
+
+ for (child = sd->s_children; child; child = child->s_sibling)
+  if (sysfs_type(child) == SYSFS_DIR)
+   nr++;
+
```

```
+ return nr + 2;
+}
+
 static void sysfs_init_inode(struct sysfs_dirent *sd, struct inode *inode)
 {
+ struct bin_attribute *bin_attr;
+
  inode->i_blocks = 0;
  inode->i_mapping->a_ops = &sysfs_aops;
  inode->i_mapping->backing_dev_info = &sysfs_backing_dev_info;
@@ -140,6 +154,37 @@ static void sysfs_init_inode(struct sysfs_dirent *sd, struct inode *inode)
   set_inode_attr(inode, sd->s_iattr);
  } else
   set_default_inode_attr(inode, sd->s_mode);
+
+
+ /* initialize inode according to type */
+ switch (sysfs_type(sd)) {
+ case SYSFS_ROOT:
+  inode->i_op = &sysfs_dir_inode_operations;
+  inode->i_fop = &sysfs_dir_operations;
+  inc_nlink(inode); /* directory, account for "." */
+  break;
+ case SYSFS_DIR:
+  inode->i_op = &sysfs_dir_inode_operations;
+  inode->i_fop = &sysfs_dir_operations;
+  inode->i_nlink = sysfs_count_nlink(sd);
+  break;
+ case SYSFS_KOBJ_ATTR:
+  inode->i_size = PAGE_SIZE;
+  inode->i_fop = &sysfs_file_operations;
+  break;
+ case SYSFS_KOBJ_BIN_ATTR:
+  bin_attr = sd->s_elem.bin_attr.bin_attr;
+  inode->i_size = bin_attr->size;
+  inode->i_fop = &bin_fops;
+  break;
+ case SYSFS_KOBJ_LINK:
+  inode->i_op = &sysfs_symlink_inode_operations;
+  break;
+ default:
+  BUG();
+ }
+
+ unlock_new_inode(inode);
 }

 /**
```

```
@@ -181,9 +226,6 @@ void sysfs_instantiate(struct dentry *dentry, struct inode *inode)
 {
  BUG_ON(!dentry || dentry->d_inode);

- if (inode->i_state & I_NEW)
-  unlock_new_inode(inode);
-
  d_instantiate(dentry, inode);
 }

diff --git a/fs/sysfs/mount.c b/fs/sysfs/mount.c
index 0c016e1..919eaaf 100644
--- a/fs/sysfs/mount.c
+++ b/fs/sysfs/mount.c
@@ -50,11 +50,6 @@ static int sysfs_fill_super(struct super_block *sb, void *data, int silent)
   return -ENOMEM;
  }

- inode->i_op = &sysfs_dir_inode_operations;
- inode->i_fop = &sysfs_dir_operations;
- inc_nlink(inode); /* directory, account for "." */
- unlock_new_inode(inode);
-
  /* instantiate and link root dentry */
  root = d_alloc_root(inode);
  if (!root) {
--
1.5.1.1.181.g2de0
```