Subject: [PATCH] SUNRPC: remove non-exclusive pipe creation from RPC pipefs
Posted by Stanislav Kinsbursky on Mon, 31 Oct 2011 12:09:38 GMT
View Forum Message <> Reply to Message

This patch-set was created in context of clone of git branch:
git://git.linux-nfs.org/projects/trondmy/nfs-2.6.git
and rebased on tag "v3.1".

SUNRPC pipefs non-exclusive pipe creation code looks obsolete. IOW, as I see
it, all pipes are creating with unique full path and only once.

Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>

---
 include/linux/sunrpc/rpc_pipe_fs.h |    1 -
 net/sunrpc/rpc_pipe.c              |   44 +++++----------------------------
 2 files changed, 6 insertions(+), 39 deletions(-)

```
diff --git a/include/linux/sunrpc/rpc_pipe_fs.h b/include/linux/sunrpc/rpc_pipe_fs.h
index 271e1b2..80606ab 100644
--- a/include/linux/sunrpc/rpc_pipe_fs.h
+++ b/include/linux/sunrpc/rpc_pipe_fs.h
@@ -30,7 +30,6 @@ struct rpc_inode {
  int pipelen;
  int nreaders;
  int nwriters;
- int nkern_readwriters;
  wait_queue_head_t waitq;
 #define RPC_PIPE_WAIT_FOR_OPEN 1
  int flags;
diff --git a/net/sunrpc/rpc_pipe.c b/net/sunrpc/rpc_pipe.c
index 42e4b6e..a1f23c4 100644
--- a/net/sunrpc/rpc_pipe.c
+++ b/net/sunrpc/rpc_pipe.c
@@ -558,7 +558,6 @@ static int __rpc_mkpipe(struct inode *dir, struct dentry *dentry,
  if (err)
   return err;
  rpci = RPC_I(dentry->d_inode);
- rpci->nkern_readwriters = 1;
  rpci->private = private;
  rpci->flags = flags;
  rpci->ops = ops;
@@ -591,16 +590,12 @@ static int __rpc_unlink(struct inode *dir, struct dentry *dentry)
 static int __rpc_rmpipe(struct inode *dir, struct dentry *dentry)
 {
  struct inode *inode = dentry->d_inode;
- struct rpc_inode *rpci = RPC_I(inode);
```

```
- rpci->nkern_readwriters--;
- if (rpci->nkern_readwriters != 0)
-  return 0;
  rpc_close_pipes(inode);
  return __rpc_unlink(dir, dentry);
 }

-static struct dentry *__rpc_lookup_create(struct dentry *parent,
+static struct dentry *__rpc_lookup_create_exclusive(struct dentry *parent,
        struct qstr *name)
 {
  struct dentry *dentry;
@@ -608,27 +603,13 @@ static struct dentry *__rpc_lookup_create(struct dentry *parent,
  dentry = d_lookup(parent, name);
  if (!dentry) {
   dentry = d_alloc(parent, name);
-  if (!dentry) {
-   dentry = ERR_PTR(-ENOMEM);
-   goto out_err;
-  }
+  if (!dentry)
+   return ERR_PTR(-ENOMEM);
  }
- if (!dentry->d_inode)
+ if (dentry->d_inode == NULL) {
   d_set_d_op(dentry, &rpc_dentry_operations);
-out_err:
- return dentry;
-}
-
-static struct dentry *__rpc_lookup_create_exclusive(struct dentry *parent,
-       struct qstr *name)
-{
- struct dentry *dentry;
-
- dentry = __rpc_lookup_create(parent, name);
- if (IS_ERR(dentry))
-  return dentry;
- if (dentry->d_inode == NULL)
   return dentry;
+ }
  dput(dentry);
  return ERR_PTR(-EEXIST);
 }
@@ -816,22 +797,9 @@ struct dentry *rpc_mkpipe(struct dentry *parent, const char *name,
  q.hash = full_name_hash(q.name, q.len),

  mutex_lock_nested(&dir->i_mutex, I_MUTEX_PARENT);
```

```
- dentry = __rpc_lookup_create(parent, &q);
+ dentry = __rpc_lookup_create_exclusive(parent, &q);
  if (IS_ERR(dentry))
   goto out;
- if (dentry->d_inode) {
-  struct rpc_inode *rpci = RPC_I(dentry->d_inode);
-  if (rpci->private != private ||
-    rpci->ops != ops ||
-    rpci->flags != flags) {
-   dput (dentry);
-   err = -EBUSY;
-   goto out_err;
-  }
-  rpci->nkern_readwriters++;
-  goto out;
- }
-
  err = __rpc_mkpipe(dir, dentry, umode, &rpc_pipe_fops,
       private, ops, flags);
  if (err)
```