Subject: [patch 21/22] sys_mknodat(): elevate write count for vfs_mknod/create()
Posted by Cedric Le Goater on Thu, 07 Jun 2007 15:25:48 GMT
View Forum Message <> Reply to Message

From: Dave Hansen <hansendc@us.ibm.com>

This takes care of all of the direct callers of vfs_mknod().
Since a few of these cases also handle normal file creation
as well, this also covers some calls to vfs_create().

Signed-off-by: Dave Hansen <hansendc@us.ibm.com>

---
 fs/namei.c        |   12 ++++++++++
 fs/nfsd/vfs.c     |    8 ++++++--
 net/unix/af_unix.c |    4 ++++
 3 files changed, 22 insertions(+), 2 deletions(-)

Index: 2.6.22-rc4-mm2-robindmount/fs/namei.c
===================================================================
--- 2.6.22-rc4-mm2-robindmount.orig/fs/namei.c
+++ 2.6.22-rc4-mm2-robindmount/fs/namei.c
@@ -1949,14 +1949,26 @@ asmlinkage long sys_mknodat(int dfd, con
  if (!IS_ERR(dentry)) {
   switch (mode & S_IFMT) {
   case 0: case S_IFREG:
+   error = mnt_want_write(nd.mnt);
+   if (error)
+    break;
    error = vfs_create(nd.dentry->d_inode,dentry,mode,&nd);
+   mnt_drop_write(nd.mnt);
    break;
   case S_IFCHR: case S_IFBLK:
+   error = mnt_want_write(nd.mnt);
+   if (error)
+    break;
    error = vfs_mknod(nd.dentry->d_inode,dentry,mode,
     new_decode_dev(dev));
+   mnt_drop_write(nd.mnt);
    break;
   case S_IFIFO: case S_IFSOCK:
+   error = mnt_want_write(nd.mnt);
+   if (error)
+    break;
    error = vfs_mknod(nd.dentry->d_inode,dentry,mode,0);
+   mnt_drop_write(nd.mnt);
    break;
   case S_IFDIR:

```
    error = -EPERM;
Index: 2.6.22-rc4-mm2-robindmount/fs/nfsd/vfs.c
===================================================================
--- 2.6.22-rc4-mm2-robindmount.orig/fs/nfsd/vfs.c
+++ 2.6.22-rc4-mm2-robindmount/fs/nfsd/vfs.c
@@ -1192,7 +1192,11 @@ nfsd_create(struct svc_rqst *rqstp, stru
  case S_IFBLK:
  case S_IFIFO:
  case S_IFSOCK:
+  host_err = mnt_want_write(fhp->fh_export->ex_mnt);
+  if (host_err)
+   break;
   host_err = vfs_mknod(dirp, dchild, iap->ia_mode, rdev);
+  mnt_drop_write(fhp->fh_export->ex_mnt);
   break;
  default:
         printk("nfsd: bad file type %o in nfsd_create\n", type);
@@ -1811,7 +1815,7 @@ nfsd_permission(struct svc_export *exp,
   inode->i_mode,
   IS_IMMUTABLE(inode)? " immut" : "",
   IS_APPEND(inode)? " append" : "",
-  __mnt_is_readonly(exp->mnt)? " ro" : "");
+  __mnt_is_readonly(exp->ex_mnt)? " ro" : "");
  dprintk("      owner %d/%d user %d/%d\n",
   inode->i_uid, inode->i_gid, current->fsuid, current->fsgid);
 #endif
@@ -1822,7 +1826,7 @@ nfsd_permission(struct svc_export *exp,
   */
  if (!(acc & MAY_LOCAL_ACCESS))
   if (acc & (MAY_WRITE | MAY_SATTR | MAY_TRUNC)) {
-   if (EX_RDONLY(exp) || __mnt_is_readonly(exp->mnt))
+   if (EX_RDONLY(exp) || __mnt_is_readonly(exp->ex_mnt))
     return nfserr_rofs;
    if (/* (acc & MAY_WRITE) && */ IS_IMMUTABLE(inode))
     return nfserr_perm;
Index: 2.6.22-rc4-mm2-robindmount/net/unix/af_unix.c
===================================================================
--- 2.6.22-rc4-mm2-robindmount.orig/net/unix/af_unix.c
+++ 2.6.22-rc4-mm2-robindmount/net/unix/af_unix.c
@@ -815,7 +815,11 @@ static int unix_bind(struct socket *sock
    */
   mode = S_IFSOCK |
         (SOCK_INODE(sock)->i_mode & ~current->fs->umask);
+  err = mnt_want_write(nd.mnt);
+  if (err)
+   goto out_mknod_dput;
   err = vfs_mknod(nd.dentry->d_inode, dentry, mode, 0);
+  mnt_drop_write(nd.mnt);
```

```
   if (err)
    goto out_mknod_dput;
   mutex_unlock(&nd.dentry->d_inode->i_mutex);
```

--