
Subject: [PATCH 19/25] sys_mknodat(): elevate write count for vfs_mknod/create()

Posted by [Dave Hansen](#) on Mon, 11 Dec 2006 22:30:15 GMT

[View Forum Message](#) <> [Reply to Message](#)

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 <haveblue@us.ibm.com>

```
lxc-dave/fs/namei.c      | 12 ++++++++
lxc-dave/fs/nfsd/vfs.c   |  4 ++++
lxc-dave/net/unix/af_unix.c |  4 ++++
3 files changed, 20 insertions(+)
```

diff -puN fs/namei.c~18-24-sys-mknodat-elevate-write-count-for-vfs-mknod-create fs/namei.c

--- lxc/fs/namei.c~18-24-sys-mknodat-elevate-write-count-for-vfs-mknod-create 2006-12-11

14:22:09.000000000 -0800

+++ lxc-dave/fs/namei.c 2006-12-11 14:22:09.000000000 -0800

@@ -1899,14 +1899,26 @@ 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;
```

diff -puN fs/nfsd/vfs.c~18-24-sys-mknodat-elevate-write-count-for-vfs-mknod-create fs/nfsd/vfs.c

```

--- lxc/fs/nfsd/vfs.c~18-24-sys-mknodat-elevate-write-count-for-vfs-mknod-create 2006-12-11
14:22:09.000000000 -0800
+++ lxc-dave/fs/nfsd/vfs.c 2006-12-11 14:22:09.000000000 -0800
@@ -665,6 +665,9 @@ nfsd_open(struct svc_rqst *rqstp, struct
/* Disallow write access to files with the append-only bit set
 * or any access when mandatory locking enabled
 */
+ err = mnt_want_write(fhp->fh_export->ex_mnt);
+ if (err)
+ goto out_nfserr;
err = nfserr_perm;
if (IS_APPEND(inode) && (access & MAY_WRITE))
goto out;
@@ -1199,6 +1202,7 @@ nfsd_create(struct svc_rqst *rqstp, stru
printf("nfsd: bad file type %o in nfsd_create\n", type);
host_err = -EINVAL;
}
+ mnt_drop_write(fhp->fh_export->ex_mnt);
if (host_err < 0)
goto out_nfserr;

diff -puN net/unix/af_unix.c~18-24-sys-mknodat-elevate-write-count-for-vfs-mknod-create
net/unix/af_unix.c
--- lxc/net/unix/af_unix.c~18-24-sys-mknodat-elevate-write-count-for-vfs-mknod-create 2006-12-11
14:22:09.000000000 -0800
+++ lxc-dave/net/unix/af_unix.c 2006-12-11 14:22:09.000000000 -0800
@@ -816,7 +816,11 @@ 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);

```

Containers mailing list
Containers@lists.osdl.org
<https://lists.osdl.org/mailman/listinfo/containers>
