Subject: [PATCH v2] Allow access to /proc/$PID/fd after setuid()
Posted by adobriyan on Fri, 02 Feb 2007 11:13:44 GMT
View Forum Message <> Reply to Message

crap, Pavel reminded that get_proc_task() can return NULL.
--- ---
/proc/$PID/fd has r-x------ permissions, so if process does setuid(), it
will not be able to access /proc/*/fd/. This breaks fstatat() emulation
in glibc.

```
open("foo", O_RDONLY|O_DIRECTORY)       = 4
setuid32(65534)                         = 0
stat64("/proc/self/fd/4/bar", 0xbfafb298) = -1 EACCES (Permission denied)
```

Comment from Andrew Morton.

Signed-off-by: Alexey Dobriyan <adobriyan@openvz.org>

```
 fs/proc/base.c |   23 +++++++++++++++++++++++
 1 file changed, 23 insertions(+)

--- a/fs/proc/base.c
+++ b/fs/proc/base.c
@@ -1414,10 +1414,33 @@ static struct file_operations proc_fd_op
 };

 /*
+ * /proc/pid/fd needs a special permission handler so that a process can still
+ * access /proc/self/fd after it has executed a setuid().
+ */
+static int proc_fd_permission(struct inode *inode, int mask,
+    struct nameidata *nd)
+{
+ struct task_struct *tsk;
+ int rv;
+
+ rv = generic_permission(inode, mask, NULL);
+ if (rv == 0)
+  return 0;
+ tsk = get_proc_task(inode);
+ if (tsk) {
+  if (tsk == current)
+   rv = 0;
+  put_task_struct(tsk);
+ }
+ return rv;
+}
```

```
+
+/*
  * proc directories can do almost nothing..
  */
 static struct inode_operations proc_fd_inode_operations = {
  .lookup  = proc_lookupfd,
+ .permission = proc_fd_permission,
  .setattr = proc_setattr,
 };
```