Subject: [PATCH] Fix race between proc_get_inode() and remove_proc_entry()
Posted by adobriyan on Wed, 07 Mar 2007 08:54:00 GMT
View Forum Message <> Reply to Message

```
proc_lookup    remove_proc_entry
===========    =================

lock_kernel();
spin_lock(&proc_subdir_lock);
[find PDE with refcount 0]
spin_unlock(&proc_subdir_lock);
     spin_lock(&proc_subdir_lock);
     [find PDE with refcount 0]
     [check refcount and free PDE]
     spin_unlock(&proc_subdir_lock);
proc_get_inode:
 de_get(de); /* boom */

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

 fs/proc/generic.c       |    2 ++
 fs/proc/inode.c         |   12 ++++++--------
 include/linux/proc_fs.h |    3 +++
 3 files changed, 9 insertions(+), 8 deletions(-)

--- a/fs/proc/generic.c
+++ b/fs/proc/generic.c
@@ -398,6 +398,7 @@ struct dentry *proc_lookup(struct inode
    if (!memcmp(dentry->d_name.name, de->name, de->namelen)) {
     unsigned int ino = de->low_ino;

+    de_get(de);
     spin_unlock(&proc_subdir_lock);
     error = -EINVAL;
     inode = proc_get_inode(dir->i_sb, ino, de);
@@ -414,6 +415,7 @@ struct dentry *proc_lookup(struct inode
   d_add(dentry, inode);
   return NULL;
  }
+ de_put(de);
  return ERR_PTR(error);
 }

--- a/fs/proc/inode.c
+++ b/fs/proc/inode.c
@@ -21,7 +21,7 @@ #include <asm/uaccess.h>
```

```
 #include "internal.h"

-static inline struct proc_dir_entry * de_get(struct proc_dir_entry *de)
+struct proc_dir_entry * de_get(struct proc_dir_entry *de)
 {
  if (de)
   atomic_inc(&de->count);
@@ -31,7 +31,7 @@ static inline struct proc_dir_entry * de
 /*
  * Decrements the use count and checks for deferred deletion.
  */
-static void de_put(struct proc_dir_entry *de)
+void de_put(struct proc_dir_entry *de)
 {
  if (de) {
   lock_kernel();
@@ -147,11 +147,6 @@ struct inode *proc_get_inode(struct supe
 {
  struct inode * inode;

- /*
-  * Increment the use count so the dir entry can't disappear.
-  */
- de_get(de);
-
  WARN_ON(de && de->deleted);

  if (de != NULL && !try_module_get(de->owner))
@@ -185,7 +180,6 @@ out_ino:
  if (de != NULL)
   module_put(de->owner);
 out_mod:
- de_put(de);
  return NULL;
 }

@@ -200,6 +194,7 @@ int proc_fill_super(struct super_block *
  s->s_op = &proc_sops;
  s->s_time_gran = 1;

+ de_get(&proc_root);
  root_inode = proc_get_inode(s, PROC_ROOT_INO, &proc_root);
  if (!root_inode)
   goto out_no_root;
@@ -213,6 +208,7 @@ int proc_fill_super(struct super_block *
 out_no_root:
  printk("proc_read_super: get root inode failed\n");
  iput(root_inode);
```

```
+ de_put(&proc_root);
  return -ENOMEM;
 }
 MODULE_LICENSE("GPL");
--- a/include/linux/proc_fs.h
+++ b/include/linux/proc_fs.h
@@ -105,6 +105,9 @@ unsigned long task_vsize(struct mm_struc
 int task_statm(struct mm_struct *, int *, int *, int *, int *);
 char *task_mem(struct mm_struct *, char *);

+struct proc_dir_entry * de_get(struct proc_dir_entry *de);
+void de_put(struct proc_dir_entry *de);
+
 extern struct proc_dir_entry *create_proc_entry(const char *name, mode_t mode,
       struct proc_dir_entry *parent);
 extern void remove_proc_entry(const char *name, struct proc_dir_entry *parent);
```