
Subject: [PATCH 7/28] [PREP 7/14] Prepare proc_flush_task to flush entries from multiple proc trees

Posted by [Pavel Emelianov](#) on Fri, 15 Jun 2007 16:05:32 GMT

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

Since a task will appear in more than one proc tree we need to shrink many trees. For this case we pass the struct pid to proc_flush_task() and shrink the mounts of all the namespaces this pid belongs to.

The NULL passed to it means that only global mount is to be flushed.

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

```
fs/proc/base.c      | 25 ++++++-----
include/linux/proc_fs.h | 6 +++--
kernel/exit.c        | 2 +-
3 files changed, 27 insertions(+), 6 deletions(-)
```

--- ./fs/proc/base.c.procflushtask 2007-06-15 15:02:29.000000000 +0400

+++ ./fs/proc/base.c 2007-06-15 15:04:18.000000000 +0400

@@ -2184,7 +2184,7 @@ static const struct inode_operations pro

* that no dcache entries will exist at process exit time it

* just makes it very unlikely that any will persist.

*/

-void proc_flush_task(struct task_struct *task)

+static void proc_flush_task_mnt(struct task_struct *task, struct vfsmount *mnt)

```
{
    struct dentry *dentry, *leader, *dir;
    char buf[PROC_NUMBUF];
```

@@ -2192,7 +2192,7 @@ void proc_flush_task(struct task_struct

```
    name.name = buf;
```

```
    name.len = snprintf(buf, sizeof(buf), "%d", task->pid);
```

- dentry = d_hash_and_lookup(proc_mnt->mnt_root, &name);

+ dentry = d_hash_and_lookup(mnt->mnt_root, &name);

```
    if (dentry) {
```

```
        shrink_dcache_parent(dentry);
```

```
        d_drop(dentry);
```

@@ -2204,7 +2204,7 @@ void proc_flush_task(struct task_struct

```
    name.name = buf;
```

```
    name.len = snprintf(buf, sizeof(buf), "%d", task->tgid);
```

- leader = d_hash_and_lookup(proc_mnt->mnt_root, &name);

+ leader = d_hash_and_lookup(mnt->mnt_root, &name);

```
    if (!leader)
```

```
        goto out;
```

@@ -2230,6 +2230,25 @@ out:

```
    return;  
}
```

```
+/*
```

```
+ * when flushing dentries from proc one need to flush them from global  
+ * proc (proc_mnt) and from all the namespaces' procs this task was seen  
+ * in. this call is supposed to make all this job.
```

```
+ */
```

```
+#ifndef CONFIG_PID_NS
```

```
+static inline void proc_flush_task_ns(struct task_struct *tsk, struct pid *pid)
```

```
+{
```

```
+}
```

```
+#else
```

```
+#endif
```

```
+
```

```
+void proc_flush_task(struct task_struct *task, struct pid *pid)
```

```
+{
```

```
+ proc_flush_task_mnt(task, proc_mnt);
```

```
+ if (pid != NULL)
```

```
+ proc_flush_task_ns(task, pid);
```

```
+}
```

```
+
```

```
static struct dentry *proc_pid_instantiate(struct inode *dir,
```

```
    struct dentry * dentry,
```

```
    struct task_struct *task, const void *ptr)
```

```
--- ./include/linux/proc_fs.h.procflushtask 2007-06-15 15:00:32.000000000 +0400
```

```
+++ ./include/linux/proc_fs.h 2007-06-15 15:03:10.000000000 +0400
```

```
@@ -111,7 +111,7 @@ extern void proc_misc_init(void);
```

```
struct mm_struct;
```

```
-void proc_flush_task(struct task_struct *task);
```

```
+void proc_flush_task(struct task_struct *task, struct pid *pid);
```

```
struct dentry *proc_pid_lookup(struct inode *dir, struct dentry * dentry, struct nameidata *);
```

```
int proc_pid_readdir(struct file * filp, void * dirent, filldir_t filldir);
```

```
unsigned long task_vsize(struct mm_struct *);
```

```
@@ -223,7 +223,9 @@ static inline void proc_net_remove(const
```

```
#define proc_net_create(name, mode, info) ({ (void)(mode), NULL; })
```

```
static inline void proc_net_remove(const char *name) {}
```

```
-static inline void proc_flush_task(struct task_struct *task) { }
```

```
+static inline void proc_flush_task(struct task_struct *task, struct pid *pid)
```

```
+{
```

```
+}
```

```
static inline struct proc_dir_entry *create_proc_entry(const char *name,
```

```
mode_t mode, struct proc_dir_entry *parent) { return NULL; }  
--- ./kernel/exit.c.procflushtask 2007-06-15 15:02:29.000000000 +0400  
+++ ./kernel/exit.c 2007-06-15 15:03:10.000000000 +0400  
@@ -185,7 +185,7 @@ repeat:  
}  
  
write_unlock_irq(&tasklist_lock);  
- proc_flush_task(p);  
+ proc_flush_task(p, NULL);  
release_thread(p);  
call_rcu(&p->rcu, delayed_put_task_struct);
```

Containers mailing list
Containers@lists.linux-foundation.org
<https://lists.linux-foundation.org/mailman/listinfo/containers>
