
Subject: [PATCH 16/17] Pid-NS(V3) Remove proc_mnts use for killing inodes
Posted by [Sukadev Bhattiprolu](#) on Sat, 16 Jun 2007 23:05:32 GMT
[View Forum Message](#) <> [Reply to Message](#)

From: sukadev@linux.vnet.ibm.com
Subject: [PATCH 16/17] Pid-NS(V3) Remove proc_mnts use for killing inodes

We use proc_mnt as a shortcut to find a superblock on which to go killing /proc inodes. This will break if we ever have more than one /proc superblock. So, use the superblock list to go find each of the /proc sb's and kill inodes on each superblock.

This does introduce an extra lock grab from what was there before, but the list should be only 1 long 99% of the time, and we don't exactly remove proc entries in hot paths. Note that this *isn't* the path that we use to get rid of the actual /proc pid entries. Those are a different beast.

Changelog:

2.6.22-rc4-mm2-pidns1:
- [Sukadev Bhattiprolu] Initialize proc_fs_type.fs_supers.

Signed-off-by: Dave Hansen <haveblue@us.ibm.com>
Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>

fs/proc/generic.c | 28 ++++++-----
fs/proc/root.c | 1 +
2 files changed, 24 insertions(+), 5 deletions(-)

Index: lx26-22-rc4-mm2/fs/proc/generic.c

```
=====
--- lx26-22-rc4-mm2.orig/fs/proc/generic.c 2007-06-15 17:21:28.000000000 -0700
+++ lx26-22-rc4-mm2/fs/proc/generic.c 2007-06-15 18:16:57.000000000 -0700
@@ -555,13 +555,10 @@ static int proc_register(struct proc_dir
    return 0;
 }

-/*
- * Kill an inode that got unregistered..
- */
-static void proc_kill_inodes(struct proc_dir_entry *de)
+static void proc_kill_inodes_sb(struct proc_dir_entry *de,
+ struct super_block *sb)
 {
    struct list_head *p;
```

- struct super_block *sb = proc_mnt->mnt_sb;

```
/*
 * Actually it's a partial revoke().
@@ -585,6 +582,27 @@ static void proc_kill_inodes(struct proc
    file_list_unlock();
}
```

```
+/*
+ * Kill an inode that got unregistered..
+ */
+static void proc_kill_inodes(struct proc_dir_entry *de)
+{
+ struct list_head *l;
+ struct file_system_type *procfs;
+
+ procfs = get_fs_type("proc");
+ if (!procfs)
+ return;
+
+ spin_lock(&sb_lock);
+ list_for_each(l, &procfs->fs_supers) {
+ struct super_block *sb;
+ sb = list_entry(l, struct super_block, s_instances);
+ proc_kill_inodes_sb(de, sb);
+ }
+ spin_unlock(&sb_lock);
+}
+
+static struct proc_dir_entry *proc_create(struct proc_dir_entry **parent,
+    const char *name,
+    mode_t mode,
```

Index: lx26-22-rc4-mm2/fs/proc/root.c

=====

--- lx26-22-rc4-mm2.orig/fs/proc/root.c 2007-06-15 17:21:28.000000000 -0700

+++ lx26-22-rc4-mm2/fs/proc/root.c 2007-06-15 18:16:57.000000000 -0700

```
@@ -44,6 +44,7 @@ static struct file_system_type proc_fs_t
    .name = "proc",
    .get_sb = proc_get_sb,
    .kill_sb = kill_anon_super,
+ .fs_supers = LIST_HEAD_INIT(proc_fs_type.fs_supers),
};
```

```
void __init proc_root_init(void)
```

--

Containers mailing list

