

---

Subject: Re: [PATCH 17/17] Pid-NS(V3) Introduce proc\_mnt for pid\_ns  
Posted by [Sukadev Bhattiprolu](#) on Wed, 20 Jun 2007 01:19:11 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Pavel Emelianov [xemul@openvz.org] wrote:

| sukadev@us.ibm.com wrote:

| > From: sukadev@linux.vnet.ibm.com

| > Subject: [PATCH 17/17] Pid-NS(V3) Introduce proc\_mnt for pid\_ns

| >

| > The following patch completes the removal of the global proc\_mnt.

| > It fetches the mnt on which to do dentry invalidations from the

| > pid\_namespace in which the task appears.

| >

| > For now, there is only one pid namespace in mainline so this is

| > straightforward. In the -lxc tree we'll have to do something

| > more complex. The proc\_flush\_task() code takes a task, and

| > needs to be able to find the corresponding proc superblocks on

| > which that task's /proc/<pid> directories could appear. We

| > can tell in which pid namespaces a task appears, so I put a

| > pointer from the pid namespace to the corresponding proc\_mnt.

| >

| > /proc currently has some special code to make sure that the root

| > directory gets set up correctly. It proc\_mnt variable in order

| > to find its way to the root inode.

| >

| > Changelog:

| >

| > 2.6.22-rc4-mm2-pidns1:

| >

| > - Call proc\_fill\_super once per pid namespace

| > - Call proc\_flush\_task() before detaching a task's 'struct pid'.

| > - Get a reference to pid namespace when mounting/remounting /proc.

| > Put this reference when unmounting.

| >

| > Signed-off-by: Dave Hansen <haveblue@us.ibm.com>

| > Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>

| > ---

| >

| > fs/proc/base.c | 30 ++++++

| > fs/proc/inode.c | 12 +++++

| > fs/proc/root.c | 65 ++++++

| > include/linux/pid\_namespace.h | 1

| > include/linux/proc\_fs.h | 1

| > kernel/exit.c | 2 -

| > 6 files changed, 87 insertions(+), 24 deletions(-)

| >

| [snip]

|

```
| > @@ -55,12 +90,6 @@ void __init proc_root_init(void)
| > err = register_filesystem(&proc_fs_type);
| > if (err)
| > return;
| > - proc_mnt = kern_mount(&proc_fs_type);
| > - err = PTR_ERR(proc_mnt);
| > - if (IS_ERR(proc_mnt)) {
| > - unregister_filesystem(&proc_fs_type);
| > - return;
| > - }
```

| Wow! Is this safe? Everyone expects proc\_mnt to be not NULL.

Can you elaborate ? Between the previous patch (i.e 16/17) and this we made sure no one is using the global proc\_mnt. Everyone should be using the pid\_ns->proc\_mnt instead.

```
| > proc_misc_init();
| > proc_net = proc_mkdir("net", NULL);
| > proc_net_stat = proc_mkdir("net/stat", NULL);
|
| [snip]
|
| > Index: lx26-22-rc4-mm2/kernel/exit.c
| > =====
| > --- lx26-22-rc4-mm2.orig/kernel/exit.c 2007-06-16 04:15:23.000000000 -0700
| > +++ lx26-22-rc4-mm2/kernel/exit.c 2007-06-16 04:15:23.000000000 -0700
| > @@ -157,6 +157,7 @@ void release_task(struct task_struct * p
| > struct task_struct *leader;
| > int zap_leader;
| > repeat:
| > + proc_flush_task(p);
```

| Such a flush won't actually remove \*any\* dentries from the tree and thus may be omitted.

Again, can you elaborate ?

The previous version had proc\_flush\_task() much later in release\_task() and that did not free dentries bc task\_pid() was NULL. Moving proc\_flush\_task() up, should fix that and it did fix the leak we were seeing before.

```
| > atomic_dec(&p->user->processes);
| > write_lock_irq(&tasklist_lock);
| > ptrace_unlink(p);
| > @@ -185,7 +186,6 @@ repeat:
```

```
|> }  
|>  
|> write_unlock_irq(&tasklist_lock);  
|> - proc_flush_task(p);  
|> 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>

---