Subject: Re: [RFC][PATCH] Do not set /proc inode->pid for non-pid-related inodes
Posted by Dave Hansen on Tue, 20 Mar 2007 02:30:32 GMT
View Forum Message <> Reply to Message

On Mon, 2007-03-19 at 20:04 -0600, Eric W. Biederman wrote:
> Dave Hansen <hansendc@us.ibm.com> writes:
> Regardless I would like to see a little farther down on
> how we test to see if the pid namespace is alive and how we
> make these functions do nothing if it has died.

That shouldn't be too hard. We have access to the superblock pretty
much everywhere, and we now store the pid_namespace in there (with some
patches I posted earlier).

> I would also
> like to see how we perform the appropriate lookups by pid namespace.

What do you mean?

> Basically I want to see how we finish up multiple namespace support
> for /proc before we start with the micro optimizations.

Serge was tracking down some weird /proc issues and noticed that we
expect a pid_nr==1 for the pid namespace as long as it has a /proc
around. That is an assumption doesn't always hold now.

> I'm fairly certain this patch causes us to do the wrong thing when
> the pid namespace exits, and I don't see much gain except for the
> death of find_get_pid.

In the default, mainline case, it shouldn't be a problem at all. We
don't have the init pid namespace exiting.

Shouldn't the lifetime of things under a /proc mount be tied to the life
of the mount, and not to the pid_namespace it is tied to? It seems
relatively sane to me to have a /proc empty of all processes, but still
have /proc/cpuinfo even if all of its processes are gone.

> > For what I would imagine are historical reasons, we set
> > all struct proc_inode->pid fields. We use the init
> > process for all non-/proc/<pid> inodes.
> >
> > We get a handle to the init process in proc_get_sb()
> > then fetch it out in proc_pid_readdir():
> >
> > struct task_struct *reaper =
> > get_proc_task(filp->f_path.dentry->d_inode);
> >

> > The filp in that case is always the root inode on which
> > someone is doing a readdir.  This reaper variable gets
> > passed down into proc_base_instantiate() and eventually
> > set in the new inode's ->pid field.
> >
> > The problem is that I don't see anywhere that we
> > actually go and use this, outside of the /proc/<pid>
> > directories.  Just referencing the init process like
> > this is a pain for containers because our init process
> > (pid == 1) can actually go away.
>
> Which as far as can recall is part of the point.  If you have a pid
> namespace with normal semantics the child reaper pid == 1 is the last
> pid in the pid namespace to exit.  Therefore when it exists the pid
> namespace exists and with it doesn't the pid namespace does not exist.

pid_delete_dentry() looks like the remaining place that really cares.
It would be pretty easy to have it check the pid namespace.

-- Dave

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