
Subject: Re: [PATCH v5] Fix rmmod/read/write races in /proc entries

Posted by [Andrew Morton](#) on Fri, 16 Mar 2007 11:50:30 GMT

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

On Fri, 16 Mar 2007 12:16:13 +0300 Alexey Dobriyan <adobriyan@sw.ru> wrote:

> On Thu, Mar 15, 2007 at 05:53:04PM -0800, Andrew Morton wrote:

> > My, what a lot of code you have here. I note that nobody can be assed even
> > reviewing it. Now why is that?

>

> I hope, AI could find some time again.

>

> > On Sun, 11 Mar 2007 20:04:56 +0300 Alexey Dobriyan <adobriyan@sw.ru> wrote:

> > > Fix following races:

> > > =====

> > > 1. Write via ->write_proc sleeps in copy_from_user(). Module disappears
> > > meanwhile. Or, more generically, system call done on /proc file, method
> > > supplied by module is called, module disappears meanwhile.

> > >

> > > pde = create_proc_entry()

> > > if (!pde)

> > > return -ENOMEM;

> > > pde->write_proc = ...

> > > open

> > > write

> > > copy_from_user

> > > pde = create_proc_entry();

> > > if (!pde) {

> > > remove_proc_entry();

> > > return -ENOMEM;

> > > /* module unloaded */

> > > }

> >

> > We usually fix that race by pinning the module: make whoever registered the
> > proc entries also register their THIS_MODULE, do a try_module_get() on it
> > before we start to play with data structures which the module owns.

> >

> > Can we do that here?

>

> We can, but it will be unreliable:

>

> Typical proc entry creation sequence is

>

> pde = create_proc_entry(...);

> if (pde)

> pde->owner = THIS_MODULE;

>

> Right after create_proc_entry() ->owner is NULL, so try_module_get()

> won't do anything, but `proc_delete_inode()` could put module which was
> never gotten.
>
> This should be fixable by always setting `->owner` before proc entry is
> glued to proc entries tree. Something like this:
>
> `#define create_proc_entry(...) __create_proc_entry(..., THIS_MODULE)`

Yes, I was thinking of something like that.

> However, I think it's not enough: `delete_module(2)` first waits for
> `refcount` becoming zero, only then calls module's exit function which
> starts removing proc entries. In between, proc entries are accessible
> and fully-functional, so `try_module_get()` can again get module and
> `module_put(pde->owner)` can happen AFTER module disappears.
> What will it put?
>
> And how can you fix that? The only way I know is to REMOVE `->owner`
> completely, once we agree on this `pde_users/pde_unload_lock` stuff.

I think the `rmmod` code will take care of that.

Once `delete_module()` has called `try_stop_module()`, no following
`try_module_get()` will succeed. And see that `wait_for_zero_refcount()` call
in there which waits for any present users of the module to go away.
