Subject: Re: [PATCH] pci_get_device call from interrupt in reboot fixups
Posted by Andrew Morton on Tue, 07 Aug 2007 07:24:37 GMT
View Forum Message <> Reply to Message

On Mon, 6 Aug 2007 19:49:10 -0700 Greg KH <gregkh@suse.de> wrote:

> On Mon, Aug 06, 2007 at 11:16:20AM +0400, Denis V. Lunev wrote:
> > Greg KH wrote:
> > > On Fri, Aug 03, 2007 at 02:39:24PM +0400, Denis V. Lunev wrote:
> > >> The following calltrace is possible now:
> > >> handle_sysrq
> > >>   machine_emergency_restart
> > >>   mach_reboot_fixups
> > >>   pci_get_device
> > >>   pci_get_subsys
> > >>   down_read
> > >> The patch obtains PCI device during initialization to avoid bothering PCI
> > >> search engine in interrupt. Devices used in this code are not supposed to
> > >> be pluggable, so it looks safe to keep them.
> > >
> > > What devices are supposed to be affected here? Are you sure that they
> > > can't be removed later? Grabbing references here might mess with them
> > > in the future.
> > Right now the list is the following:
> > static struct device_fixup fixups_table[] = {
> > { PCI_VENDOR_ID_CYRIX, PCI_DEVICE_ID_CYRIX_5530_LEGACY,
> > cs5530a_warm_reset },
> > { PCI_VENDOR_ID_AMD, PCI_DEVICE_ID_AMD_CS5536_ISA, cs5536_warm_reset },
> > };
> >
> > Though, if the approach is not suitable, we can skip fixups if we came
> > from sysrq.
>
> I don't think we really need to do fixups when we are "crashing" like
> this. The user really isn't shutting down the kernel as it should
> normally do.
>
> Andrew, I really don't want to change the PCI core to handle this, as we
> finally fixed a lot of issues with drivers trying to walk these lists
> from interrupt context. So if you want to just hide the warning message
> as we are shutting down, that's fine with me. Or just don't do the
> fixups. But grabbing a reference to the pci device is unsafe in my
> opinion and I do not want to do that.
>

OK, good decision ;)

One approach would be for some brave soul to pick his way through

the reboot code and ensure that we are correctly and reliably setting system_state to SYSTEM_RESTART, then test that in __might_sleep().

But this does suppress somewhat-useful debugging just because of sysrq-B and I really wouldn't want to utilise the horrid system_state any more that we are presently doing. I think on balance that it would be better if we could do something more targetted, like modify emergency_restart() to test in_interrupt() and to then apologetically set some well-named global flag which will shut up __might_sleep(). Pretty foul, but I can't think of anything better.
