
Subject: Re: [PATCH] pci_get_device call from interrupt in reboot fixups
Posted by [Andrew Morton](#) on Mon, 06 Aug 2007 20:03:26 GMT
[View Forum Message](#) <> [Reply to Message](#)

On Fri, 3 Aug 2007 14:39:24 +0400 "Denis V. Lunev" <den@openvz.org> 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.
>
```

hm.

```
>
> diff --git a/arch/i386/kernel/reboot_fixups.c b/arch/i386/kernel/reboot_fixups.c
> index 03e1cce..873ad55 100644
> --- a/arch/i386/kernel/reboot_fixups.c
> +++ b/arch/i386/kernel/reboot_fixups.c
> @@ -37,6 +37,7 @@ struct device_fixup {
> unsigned int vendor;
> unsigned int device;
> void (*reboot_fixup)(struct pci_dev *);
> + struct pci_dev *dev;
> };
>
> static struct device_fixup fixups_table[] = {
> @@ -49,20 +50,35 @@ static struct device_fixup fixups_table[] = {
> * is a fixup, we call it and we expect to never return from it. if we
> * do return, we keep looking and then eventually fall back to the
> * standard mach_reboot on return.
> + *
> + * Unfortunately, this code can be called from an interrupt and it is
> + * impossible to get PCI device directly. So, lets prepare the list
> + * beforehand.
```

This comment should tell the reader which interrupt path that is (ie: sysrq-B).

```
> */
> void mach_reboot_fixups(void)
> {
> struct device_fixup *cur;
```

```

> - struct pci_dev *dev;
> int i;
>
> for (i=0; i < ARRAY_SIZE(fixups_table); i++) {
>   cur = &(fixups_table[i]);
> - dev = pci_get_device(cur->vendor, cur->device, NULL);
> - if (!dev)
> + if (cur->dev == NULL)
>   continue;
>
> - cur->reboot_fixup(dev);
> + cur->reboot_fixup(cur->dev);
> + }
> +}
> +
> +int mach_fixup_init(void)
> +{
> + struct device_fixup *cur;
> + int i;
> +
> + for (i=0; i < ARRAY_SIZE(fixups_table); i++) {
> +   cur = &(fixups_table[i]);
> +   cur->dev = pci_get_device(cur->vendor, cur->device, NULL);
>   }
> + return 0;
> }
>
> +module_init(mach_fixup_init);

```

I'm not sure that we want to make core PCI code capable of being called from interrupt context just for the sake of sysrq-B. It adds complexity and maintenance hassles for something which is largely a debugging feature.

otoh, the patch is fairly simple-looking and people do use sysrq-B fairly often so I guess we'll find out if we break it again.

otoh2, perhaps we can find some quicky hack on the sysrq patch to shut up the might_sleep() warnings (which I presume is the only problem which is presently being exhibited?). Something like the unpleasant `oops_in_progress`, perhaps.

Greg, any preferences?