
Subject: [PATCH] pci_get_device call from interrupt in reboot fixups

Posted by [den](#) on Fri, 03 Aug 2007 10:39:24 GMT

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

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.

Signed-off-by: Denis V. Lunev <den@openvz.org>

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.

*/

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);
```
