

Re: newbus flaw

Source: <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/arch/2004-05/0022.html>

From: Doug Rabson (dfr_at_nlsystems.com)

Date: 05/11/04

To: Dag-Erling Smørgrav <des@des.no>
Date: Tue, 11 May 2004 16:07:50 +0100

On Tue, 2004-05-11 at 15:39, Dag-Erling Smørgrav wrote:

- > *I've found what I believe is a serious flaw in newbus.*
- >
- > *When a driver that has a DEVICE_IDENTIFY method is loaded, the*
- > *identify method is called. If it finds supported hardware, it uses*
- > *BUS_ADD_CHILD to notify the parent bus of the presence of that*
- > *hardware. At some later point, during a bus rescan, the attach*
- > *routine is called for each device that was identified in this manner.*
- >
- > *When the driver is unloaded, the device is detached, but it remains on*
- > *the bus's list of child devices. The next time the module is loaded,*
- > *its DEVICE_IDENTIFY method is called again, and incorrectly adds a*
- > *second child device to the bus, because it does not know that one*
- > *already exists.*
- >
- > *There is no way for DEVICE_IDENTIFY to check if a matching child*
- > *already exists on the bus, or for the module's event handler to unlist*
- > *the child when unloading.*
- >
- > *The first time you load the module, you get foo0; the second time, you*
- > *get foo0 *and* foo1 referencing the same physical device; the third*
- > *time, you get foo0, foo1, and foo2, etc.*
- >
- > *I've also seen something similar happen when multiple ndis drivers are*
- > *loaded; the first one re-attaches to the hardware when the second one*
- > *is loaded.*

This is a known problem. The 'right' solution is to add a new static method to the device interface which is the opposite of IDENTIFY (e.g. UNIDENTIFY). One way to get around the problem is to do something like:

```
void foo_identify(driver_t *driver, device_t parent)
{
    device_t child;
    child = device_find_child(parent, "foo", 0);
    if (!child)
        BUS_ADD_CHILD(parent, 0, "foo", 0);
}
```

```
}
```

Alternatively, you can add a module handler:

```
static device_t foo;
```

```
void foo_identify(driver_t *driver, device_t parent)
{
    foo = BUS_ADD_CHILD(parent, 0, "foo", -1);
}
```

...

```
int foo_module_handler(struct module *mod, int what, void *arg)
{
    if (what == MOD_UNLOAD && foo)
        device_delete_child(device_get_parent(foo), foo);
}
```

...

```
DRIVER_MODULE(foo, bar, foo_driver, foo_devclass, foo_module_handler,
0);
```

freebsd-arch@freebsd.org mailing list

<http://lists.freebsd.org/mailman/listinfo/freebsd-arch>

To unsubscribe, send any mail to "freebsd-arch-unsubscribe@freebsd.org"