

# [PATCH 2.6.10-rc1 0/4] driver-model: manual device attach

*Source:* <http://linux.derkeiler.com/Mailing-Lists/Kernel/2004-11/1130.html>

---

*From:* Tejun Heo ([tj\\_at\\_home-tj.org](mailto:tj_at_home-tj.org))

*Date:* 11/04/04

Date: Thu, 4 Nov 2004 16:43:30 +0900

To: [rusty@rustcorp.com.au](mailto:rusty@rustcorp.com.au), [mochel@osdl.org](mailto:mochel@osdl.org), [greg@kroah.com](mailto:greg@kroah.com)

Hello, again. :-)

These are the manual device attach patches I was talking about in the previous posting. These patches need devparam patches to be applied first. It's composed of two parts.

## 1. sysctl node dev.autoattach

dev.autoattach is read/write integer sysctl node which controls driver-model's behavior regarding device - driver association.

- 0: autoattach disabled. devices are not associated with drivers automatically. i.e. insmod'ing e100.ko won't cause it to attach to the actual e100 devices.
- 1: autoattach enabled. The default value. This is the same as the current driver model behavior. Driver model automatically associates devices to drivers.
- 2: rescan command. If this value is written, bus\_rescan\_devices() is invoked for all the registered bus types; thus attaching all devices to available drivers. After rescan is complete, the autoattach value is set to 1.

## 2. per-device attach and detach sysfs node.

Two files named attach and detach are created under each device's sysfs directory. Reading attach node shows the name of applicable drivers. Writing a driver name attaches the device to the driver. Also, per-device parameters can be specified when writing to an attach node. Writing anything to the write-only detach node detaches the driver from the currently associated driver.

===== So, for example, on my machine which has two e100's...

```
# pwd
/sys/bus/pci/devices/0000:00:04.0
```

```
# sysctl dev.autoattach
dev.autoattach = 1
# modprobe e100
e100: Intel(R) PRO/100 Network Driver, 3.2.3-k2-NAPI
e100: Copyright(c) 1999-2004 Intel Corporation
e100: eth0: e100_probe: addr 0xfeafe000, irq 20, MAC addr 00:E0:81:01:7F:F3
e100: eth1: e100_probe: addr 0xfeafd000, irq 21, MAC addr 00:E0:81:01:7F:F4
# modprobe eeepro100
eeepro100.c:v1.09j-t 9/29/99 Donald Becker http://www.scyld.com/network...
eeepro100.c: $Revision: 1.36 $ 2000/11/17 Modified by Andrey V. Savochkin...
# ls -l driver
lrwxrwxrwx 1 root root 0 Nov 4 16:26 driver -> ../../bus/pci/drivers/e100
# rmmod e100
# rmmod eeepro100
# sysctl -w dev.autoattach=0
dev.autoattach = 0
# modprobe e100
e100: Intel(R) PRO/100 Network Driver, 3.2.3-k2-NAPI
e100: Copyright(c) 1999-2004 Intel Corporation
# modprobe eeepro100
eeepro100.c:v1.09j-t 9/29/99 Donald Becker http://www.scyld.com/network...
eeepro100.c: $Revision: 1.36 $ 2000/11/17 Modified by Andrey V. Savochkin...
# ls -l driver
ls: driver: No such file or directory
# cat attach
e100
eeepro100
# echo eeepro100 > attach
eth0: OEM i82557/i82558 10/100 Ethernet, 00:E0:81:01:7F:F3, IRQ 20.
Receiver lock-up bug exists -- enabling work-around.
Board assembly 123456-120, Physical connectors present: RJ45
...
# ls -l driver
lrwxrwxrwx 1 root root 0 Nov 4 16:27 driver -> ../../bus/pci/drivers/eeepro100
# sysctl -w dev.autoattach=2
e100: eth1: e100_probe: addr 0xfeafd000, irq 21, MAC addr 00:E0:81:01:7F:F4
dev.autoattach = 2
```

=====  
And, drivers can accept per-device parameters like the following.

```
# pwd
/sys/bus/dp/devices/dp_dev1
# ls -l
total 0
-rw-r--r-- 1 root root 4096 Nov 4 16:34 attach
--w----- 1 root root 4096 Nov 4 16:34 detach
-rw-r--r-- 1 root root 4096 Nov 4 16:34 detach_state
# cat attach
babo
# echo babo n=15 opts=0,9,8 dp_class.integer=12 > attach
dp_bus: 0 1 2 1,2,3,0,0,0 cnt=3
```

```
dp_drv: 15 0,9,8,0 cnt=3
dp_class: 12 120 "dp_class"
# ls -l driver
lrwxrwxrwx 1 root root 0 Nov 4 16:35 driver -> ../../bus/dp/drivers/babo
# ls parameters/
a ar b byte c charp integer n opts
```

=====

We'll need to expand user-space hotplug facility to make full use of manualattach feature. I think with a bit more information exported to userland and some standardized parameters (i.e. `name' parameter for all class devices), we'll be able to give high-granularity control over driver-model to users.

What do you guys think? I think this can be quite useful with right user-space tools.

Thanks.

--

tejun

-

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to majordomo@vger.kernel.org

More majordomo info at <http://vger.kernel.org/majordomo-info.html>

Please read the FAQ at <http://www.tux.org/lkml/>