
Drivers: their structure, next steps and what to do if..

Jiri Slaby

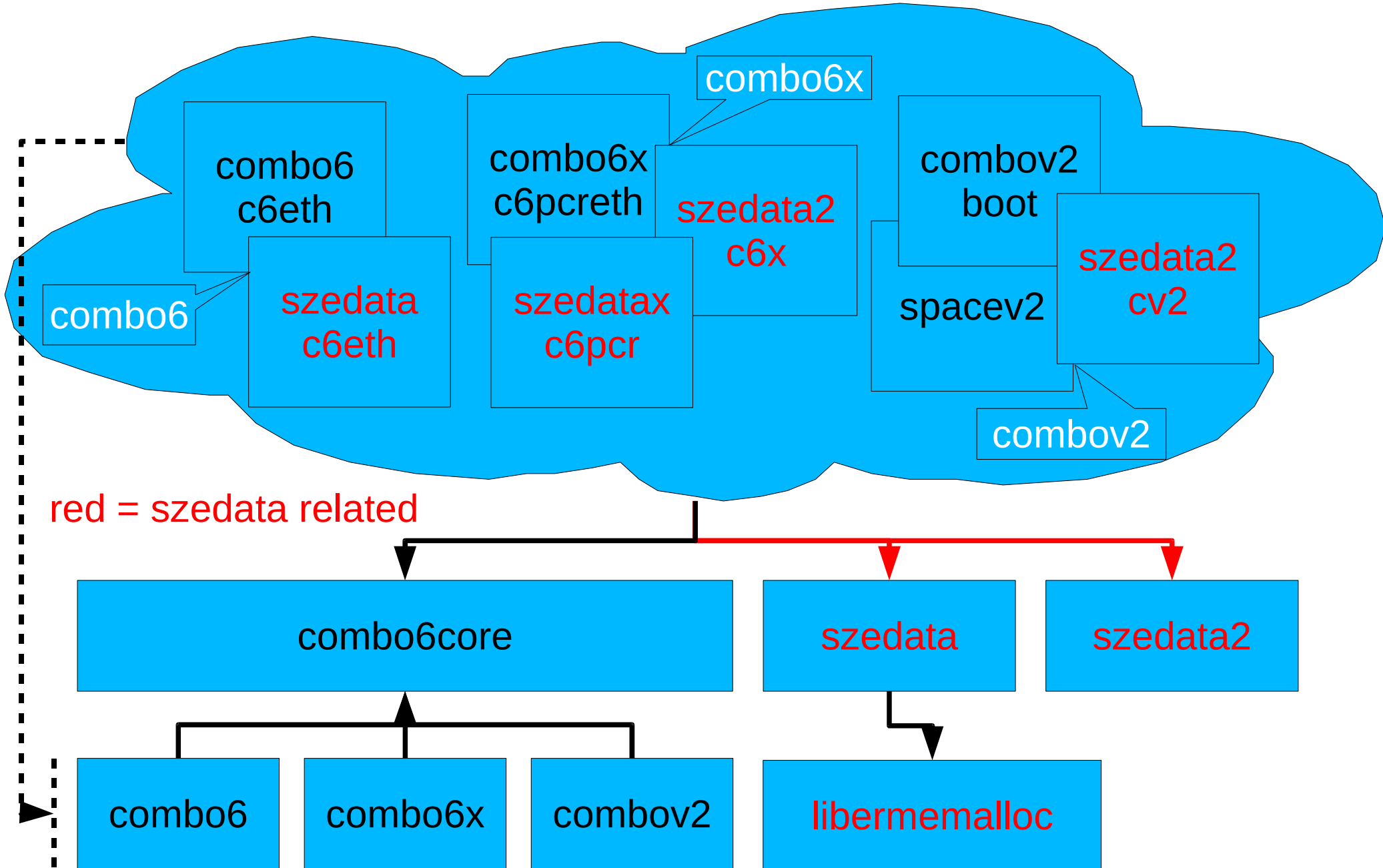
slaby@liberouter.org



Structure

- Drivers are split into 3 layers
 - Cores
 - combo6 – virt. devices maintenance, /dev+/proc support, ...
 - szedata 1 & 2 – memory and size structures allocations
 - Card specific – device initialization incl. booting, IRQs and ISRs
 - Design specific – communication with design
- Dependencies (of functions from modules)
 - Both specific modules depend on combo6core
 - Design modules depend on the card ones
 - Szedata design modules depend on szedata

Structure, cont.



Next steps

- Upstream merge
 - combo6x + szedata 1 + flowmon
- Rebuild the build system
 - It's unusable
 - Many enterprise kernels are unsupported, since the modules don't build on them
- Conversion to a bus? (no workers available)
 - It may be a good bachelor thesis
- Petr Kastovsky: modified sze2 HW and transfers via Linux net stack
- Andrej Hank: multithreaded szedata2

Drivers internals

- ELF objects with `.ko` file suffix
 - Obj divided into sections: `.text`, `.data`, ..., `.modinfo`
- `.modinfo` contains NULL-terminated strings
 - Author(s) info
 - License info
 - Module and built-against kernel versions
 - Module description
 - Known module parameters (and their types)
 - Dependencies
 - Aliases (stay tuned, see further)
 - ...

Extracting .modinfo

“It's an ELF, so what prevents me to perform:”

```
$ objdump -s -j .modinfo combo6x.ko
combo6x.ko:      file format elf32-i386
```

Contents of section .modinfo:

```
0000 61757468 6f723d43 45534e45 543b204a  author=CESNET; J
0010 61726f73 6c617620 4b797365 6c61203c  aroslav Kysela <
0020 70657265 78407065 7265782e 637a3e00  perex@perex.cz>.
0030 00000000 00000000 00000000 00000000  .....
0040 64657363 72697074 696f6e3d 436f6d62  description=Comb
0050 6f362077 69746820 50434920 49502063  o6 with PCI IP c
0060 6f726520 4c696e75 78204472 69766572  ore Linux Driver
0070 006c6963 656e7365 3d47504c 00706172  .license=GPL.par
0080 6d747970 653d656e 61626c65 3a617272  mtype=enable:arr
0090 6179206f 6620696e 74000000 00000000  ay of int.....
00a0 7061726d 3d656e61 626c653a 456e6162  parm=enable:Enab
00b0 6c652043 6f6d626f 36782063 6172642e  le Combo6x card.
00c0 00000000 00000000 00000000 00000000  .....
00d0 00000000 00000000 00000000 00000000  .....
00e0 7061726d 74797065 3d696e76 616c6964  parmtime=invalid
00f0 6d656d73 697a653a 61727261 79206f66  memsize:array of
0100 20696e74 00000000 00000000 00000000  int.....
```

...

Extracting .modinfo, cont.

Answer is: “Nothing, if you want to be cool! But ordinary people use modinfo tool.”

```
$ modinfo combo6x
filename:          /lib/modules/2.6.19.1/kernel/.../combo6x.ko
author:           CESNET; Jaroslav Kysela <perex@perex.cz>
description:      Combo6 with PCI IP core Linux Driver
license:         GPL
parmtype:         enable:array of int
parm:             enable:Enable Combo6x card.
parmtype:         invalidmemsiz:array of int
parm:             invalidmemsiz:Invalid PCI memory area size -
                  fix to 64MB.
```

[other params were here]

```
vermagic:         2.6.19.1 SMP mod_unload 686 REGPARAM
depends:          combo6core
alias:           pci:v000018ECd0000C045sv*sd*bc*sc*i*
alias:           pci:v000018ECd0000C058sv*sd*bc*sc*i*
alias:           pci:v000018ECd00006D05sv*sd*bc*sc*i*
```

Parsing modinfo output

- Parm*
 - parmtype: enable:array of int
 - parm: enable:Enable Combo6x card.
 - Format: parameter_name:type/description
 - modprobe combo6x enable=0,1,1,1
- Alias
 - * in an alias means anything, a wildcard
 - pci:v000018ECd0000C045sv*sd*bc*sc*i*
 - Format: **vVENDORdDEVICEsvSUBVENDORsdSUBDEVICE **
bcBASECLASSscSUBCLASSiINTERFACE
 - it can handle a 18ec:c045 (CESNET:combo6x) PCI device
 - Another example with subids (szedata2-cv2):
pci:v000018ECd00006D05sv000018ECsd00000200bc*sc*i*
 - See /lib/modules/<version>/modules.alias

Drivers' problems

1. I've loaded all the modules, but the `/dev/combosix*` doesn't exist
2. I've switched the firmware and it tells me: file descriptor in a bad state
3. A firmware loaded, missing `/dev/szedata*`
4. A bug in drivers, irremovable module
5. I have an amazing new FW with changed PCI IDs, missing support in drivers, slaby not at jabber

First, let me introduce *sysfs* et al. which is a cure (or gives a clue at worst) for most of these problems.

“...have you tried turning it off and on again?”

- Sysfs provides valuable info about a system
 - Much info regarding devices, useful for udev
 - Modules info (used parameters, ELF headers, ...):
`/sys/module/<name>`
 - User interface for buses: `/sys/bus`
 - Subsystems implemented as buses: PCI, USB, I2C, ACPI, HID, PNP, ISA, SCSI, ... (e.g. `/sys/bus/pci`)
 - Each bus dir contains *drivers* and *devices* subdir
 - Drivers: Combo6X, ComboV2 BOOT, ...
 - Devices: 0000:09:01.0, 0000:0d:01.0, ...
- For us, the most important information is in `/sys/bus/pci/drivers` and `/sys/bus/pci/devices`

Other helpers

- `lspci`
 - shows if the device is really there
- `dmesg`
 - may reveal a cause of driver/device loading problems
- `csid -s`
 - if the both above fail to tell us something, this one usually helps to find out the reason (such as wrong or none IDs in the ID component etc.)

Problem 1

I've loaded all the modules, but the `/dev/combosix*` doesn't exist

- Check the device is there and visible: `lspci`
 - Remember the device IDs (e.g. `18ec:c032-18ec:0200`)
- Check modules, that the device has expected IDs
 - Recall `modinfo` and `modules.alias`
- Check `dmesg` (VM exhaustion is often a reason)
- `csid -s` should show nothing aswell
- If everything fails, it's a bug

Problem 2

I've switched the firmware and it tells me: file descriptor in a bad state

- Your firmware is broken, the ID component is inaccessible or returns 0 or ~0
- Check `csid -s`, until you see a
Firmware: OK
line, you're likely to blame, unless there is a bug, indeed

Problem 3

A firmware loaded, missing /dev/szedata*

- We have been hitting this in the past, due to boot module sitting on the device, which is no longer a problem. I mention this for completeness.
- In the case you hit it again, check correspondence of IDs and modules, like in the problem no. 1
- Check dmesg

Problem 4

A bug in drivers, irremovable module

- If you still want to unbind the device from the module for a whatever reason (e.g. unbind from the v2 boot in the past), there is a bind/unbind file in the sysfs for these purposes
- Checkout lspci, remember the device placement (e.g. 0000:08:00.0)

```
# cd /sys/bus/pci/devices/0000:08:00.0/driver  
[driver is a link to the current driver in ../../drivers]  
# /bin/echo -n '0000:08:00.0' > unbind  
# cd /sys/bus/pci/drivers/ComboV2\ szedata2  
# /bin/echo -n '0000:08:00.0' > bind
```

- Unbind should remove the device link named 0000:08:00.0 in the driver's directory, bind should install it back
- Always use /bin/echo, some shell's echo doesn't complain about errors!

Problem 5

I have an amazing new FW with changed PCI IDs, missing support in drivers, slaby not at jabber

- As a **temporary solution** (till the next reboot or a module removal), you might use a `new_id` PCI facility in the sysfs
- Checkout `lspci`, remember the device placement (e.g. `0000:08:00.0`) and IDs (`18ec:c032-18ec:beef`)
- Find out a driver you want to add the IDs to (e.g. `ComboV2` `szedata2`)

```
# cd /sys/bus/pci/drivers/ComboV2\ szedata2
# /bin/echo -n '18ec c032 18ec beef' > new_id
# /bin/echo -n '0000:08:00.0' > bind
```

- It should appear in the driver's directory as a link named `0000:08:00.0` to the `../../devices/`

Conclusion

- We showed
 - the driver structure and dependencies
 - what areas we will spend time on
 - how to extract information from a module
 - what may be /sys good for
 - few solutions for problems we had met

Thank you for your attention!