CAM I/O Scheduler

Warner Losh
wlosh@netflix.com

Netflix, Inc.

AsiaBSDCon 2015 — Tokyo, Japan
12 Mar 2015

http://people.freebsd.org/~imp/asiabsdcon2015/pnp-slides.pd

Outline

Background and Context

Problem

Solution

NETFLIX

History

- kld replaced kmod in 2001
- Loader support for loading
- Tedious and error prone
- devd processed NOMATCH events since 2003
- USB better, but still weak added 2011

NETFLIX

Current Loading System Incomplete

- Possible to boot minimal and load modules
- Not automated
- Tedious and error prone
- USB requires automated script
- Can't KLD load a console, even from /boot/loader

NETFLIX

Drivers Are a Mess

- USB and PC Card in good shape (same format)
- PCI totally ad-hoc
- Different Drivers match on different things

NETFLIX

Good: USB

```
/* recognized device vendors/products */
static const STRUCT_USB_HOST_ID uath_devs[] = {
#define UATH_DEV(v,p) { USB_VP(USB_VENDOR_##v, USB_PRODUCT_##v##_##p) }
        UATH_DEV(ACCTON, SMCWUSBTG2),
        UATH_DEV(ATHEROS, AR5523),
...
};
...
static int
uath_match(device_t dev)
{
        struct usb_attach_arg *uaa = device_get_ivars(dev);

        return (usbd_lookup_id_by_uaa(uath_devs, sizeof(uath_devs), uaa));
}
```

NETFLIX

Bad: PCI – All different eg fxp

```
static const struct fxp_ident fxp_ident_table[] = {
    { 0x8086, 0x1029, -1, 0, "Intel 82559 PCI/CardBus Pro/100" },
    { 0x8086, 0x1030, -1, 0, "Intel 82559 Pro/100 Ethernet" },
    { 0x8086, 0x1031, -1, 3, "Intel 82801CAM (ICH3) Pro/100 VE Ethernet" },
    { 0x8086, 0x1032, -1, 3, "Intel 82801CAM (ICH3) Pro/100 VE Ethernet" },
...
        for (ident = fxp_ident_table; ident->name != NULL; ident++) {
                if (ident->vendor == vendor && ident->device == device &&
                    (ident->revid == revid || ident->revid == -1)) {
                        return (ident);
                }
        }
...
```

NETFLIX

Bad: PCI – All different eg ixl

```
static ixl_vendor_info_t ixl_vendor_info_array[] =
{
        {I40E_INTEL_VENDOR_ID, I40E_DEV_ID_SFP_XL710, 0, 0, 0},
        {I40E_INTEL_VENDOR_ID, I40E_DEV_ID_KX_A, 0, 0, 0},
        {I40E_INTEL_VENDOR_ID, I40E_DEV_ID_KX_B, 0, 0, 0},
        {I40E_INTEL_VENDOR_ID, I40E_DEV_ID_KX_C, 0, 0, 0},
...
        ent = ixl_vendor_info_array;
        while (ent->vendor_id != 0) {
                if ((pci_vendor_id == ent->vendor_id) &&
                    (pci_device_id == ent->device_id) &&

                    ((pci_subvendor_id == ent->subvendor_id) ||
                     (ent->subvendor_id == 0)) &&

                    ((pci_subdevice_id == ent->subdevice_id) ||
                     (ent->subdevice_id == 0))) {
                       sprintf(device_name, "%s, Version - %s",
                               ixl_strings[ent->index],
                               ixl_driver_version);
                       device_set_desc_copy(dev, device_name);
...
                       return (BUS_PROBE_DEFAULT);
                }
                ent++;
        }
        return (ENXIO);
}
```

NETFLIX

New Changes

- Modify module information
- Modify drivers – a little
- Modify kldxref
- Modify /boot/loader
- Modify kernel to load things

NETFLIX

New module information

```
 * Generic macros to create pnp info hints that modules may export
 * to allow external tools to parse their intenral device tables
 * to make an informed guess about what driver(s) to load.
#define MODULE_PNP_INFO(d, b, unique, t, l, n)                          \
        static const struct mod_pnp_match_info _module_pnp_##b##_##unique = {   \
                .descr = d,                                             \
                .bus = #b,                                              \
                .table = t,                                             \
                .entry_len = l,                                         \
                .num_entry = n                                          \
        };                                                              \
        MODULE_METADATA(_md_##b##_pnpinfo_##unique, MDT_PNP_INFO,       \
            &_module_pnp_##b##_##unique, #b);
 * descr is a string that describes each entry in the table. The general
 * form is (TYPE:pnp_name[/pnp_name];)*
 * where TYPE is one of the following:
 *      U8      uint8_t element
 *      V8      like U8 and 0xff means match any
 *      G16     uint16_t element, any value >= matches
 *      L16     uint16_t element, any value <= matches
 *      M16     uint16_t element, mask of which of the following fields to use.
 *      U16     uint16_t element
 *      V16     like U16 and 0xffff means match any
 *      U32     uint32_t element
 *      V32     like U32 and 0xffffffff means match any
 *      W32     Two 16-bit values with first pnp_name in LSW and second in MSW.
 *      Z       pointer to a string to match exactly
 *      D       like Z, but is the string passed to device_set_descr()
 *      P       A pointer that should be ignored
 *      E       EISA PNP Identifier (in binary, but bus publishes string)
 * The pnp_name "#" is reserved for other fields that should be ignored.
```

NETFLIX

USB Conversion

```
diff -r d7d6e17c7623 sys/dev/usb/wlan/if_uath.c
--- a/sys/dev/usb/wlan/if_uath.c
+++ b/sys/dev/usb/wlan/if_uath.c
@@ -2905,3 +2905,4 @@ DRIVER_MODULE(uath, uhub, uath_driver, u
 MODULE_DEPEND(uath, wlan, 1, 1, 1);
 MODULE_DEPEND(uath, usb, 1, 1, 1);
 MODULE_VERSION(uath, 1);
+USB_PNP_INFO(uath_devs);
#define USB_STD_PNP_INFO \
        ``M16:mask;U16:vendor;U16:product;L16:product;G16:product;'' \
        ``U8:devclass;U8:devsubclass;U8:devprotocol;'' \
        ``U8:intclass;U8:intsubclass;U8:intprotocol;''
#define USB_PNP_INFO(table) \
        MODULE_PNP_INFO(USB_STD_PNP_INFO, usb, table, table, sizeof(table[0]), \
        sizeof(table) / sizeof(table[0]))
```

NETFLIX

PCI Conversion

```
Add PNP info for ISA and PCI to proof design.

diff -r 73cba89410f5 sys/dev/ed/if_ed_isa.c
--- a/sys/dev/ed/if_ed_isa.c
+++ b/sys/dev/ed/if_ed_isa.c
@@ -201,3 +201,5 @@ static driver_t ed_isa_driver = {
 DRIVER_MODULE(ed, isa, ed_isa_driver, ed_devclass, 0, 0);
 MODULE_DEPEND(ed, isa, 1, 1, 1);
 MODULE_DEPEND(ed, ether, 1, 1, 1);
+MODULE_PNP_INFO("E:pnpid;", isa, ed, ed_ids, sizeof(ed_ids[0]),
+    sizeof(ed_ids) / sizeof(ed_ids[0]) - 1);
diff -r 73cba89410f5 sys/dev/ed/if_ed_pci.c
--- a/sys/dev/ed/if_ed_pci.c
+++ b/sys/dev/ed/if_ed_pci.c
@@ -143,3 +143,5 @@ static driver_t ed_pci_driver = {
 DRIVER_MODULE(ed, pci, ed_pci_driver, ed_devclass, 0, 0);
 MODULE_DEPEND(ed, pci, 1, 1, 1);
 MODULE_DEPEND(ed, ether, 1, 1, 1);
+MODULE_PNP_INFO("W32:vendor/device;D:human", pci, ed, pci_ids, sizeof(pci_ids[0]),
+    sizeof(pci_ids) / sizeof(pci_ids[0]) - 1);
```

NETFLIX

Other changes

- Modify kldxref (done – loader.hints)
- Modify /boot/loader
- Modify kernel to load things

NETFLIX

Questions? Comments?

Warner Losh
wlosh@netflix.com
imp@FreeBSD.org

http://people.freebsd.org/~imp/asiabsdconf2015/pnp-slides.pdf

NETFLIX