Ubuntu system architecture

Presentation by
Jesse Sung
jesse.sung@canonical.com
www.canonical.com
Dec. 2011
Ubuntu system architecture

- Early init
- Early userspace
- Real userspace
Ubuntu system architecture

Early init - kernel

- Low level initializations
- initcalls
- Driver model
Ubuntu system architecture

Early init – Low level initializations

- arch-specific init

```
init/main.c:
start_kernel() {
    ...   
    setup_arch();
    ...
}
```
Ubuntu system architecture

Early init – Low level initializations

• init internal data structures

```
init/main.c:
start_kernel() {
    ...
    tick_init();
    ...
    pidhash_init();
    ...
}
```
Ubuntu system architecture

Early init – Low level initializations

• `mm_init`

```c
init/main.c:
start_kernel() {
...
    build_all_zonlists();
    page_alloc_init();
...
    mm_init();
...
}
```
Early init – Low level initializations

• sched_init

```
init/main.c:
start_kernel() {
  ...
  sched_init();
  ...
}
```
Ubuntu system architecture

Early init – Low level initializations

• init_IRQ

```c
init/main.c: start_kernel() {
    ... 
    init_IRQ();
    ... 
}```
Early init – Low level initializations

• Timers

```c
init/main.c:
start_kernel() {
    ...
    init_timers();
    hrtimers_init();
    ...
}
```
Early init – Low level initializations

• a lot of works done in this stage
• Please refer to init/main.c
Early init – initcalls

• Functions to be called when kernel starts

#include/linux/init.h:
early_initcall(fn)
pure_initcall(fn)
core_initcall(fn)
...
device_initcall(fn)
device_initcall_sync(fn)
late_initcall(fn)
late_initcall_sync(fn)
Ubuntu system architecture

Early init – initcalls

• Pointers of initcalls are stored in an array

```c
__initcall_start
  early_p1
  early_p2
  ...
__early_initcall_end
  init_p1
  init_p1
  ...
__initcall_end

early_init_1() { ... }
early_init_2() { ... }
init_1() { ... }
init_2() { ... }
```
Early init – initcalls

• Arrange function pointers with macro

includes/linux/init.h:
#define __define_initcall(level,fn,id) \
  static initcall_t __initcall_##fn##id __used \
  __attribute__((__section__(".initcall" level ".init"))) = fn
Early init – initcalls

- `module_init()` is included in initcalls for build-in modules

```c
#include/linux/init.h:
define __initcall(fn) device_initcall(fn)
...
define module_init(x) __initcall(x)
```
Ubuntu system architecture

Early init – initcalls

- early_initcalls would be called before smp start

init/main.c:

do_pre_smp_initcalls() {
  initcall_t *fn;

  for (fn = __initcall_start; fn < __early_initcall_end; fn++)
    do_one_initcall(*fn);
}

Early init – initcalls

- Other initcalls would be called after smp start

```c
Init/main.c:
kernel_init() {
    ...
    smp_init();
    sched_init_smp();
    do_basic_setup();
    ...
}

do_basic_setup() {
    ...
do_initcalls();
}
```
Ubuntu system architecture

Early init – initcalls

- initcalls called order
  - early_initcall
  - pure_initcall
  - core_initcall
  - core_initcall_sync
  - postcore_initcall
  - postcore_initcall_sync
  - arch_initcall
  - arch_initcall_sync
Early init – initcalls

- initcalls called order (cont.)
  - subsys_initcall
  - subsys_initcall_sync
  - fs_initcall
  - fs_initcall_sync
  - rootfs_initcall
  - device_initcall – module_init
  - device_initcall_sync
  - late_initcall / late_initcall_sync
Early init

- Initialization code are stored in a separate memory section
- Freed after kernel starts
Ubuntu system architecture

Driver model

- bus
- device
- driver
Driver model - bus

- A bus is a channel between the processor and one or more devices
- PCI, USB, ...
Driver model – bus

• Ability to enumerate devices on the bus
Driver model – bus

• Descriptor of each device
  • Vendor ID
  • Device ID
  • Device class
  • ...

Ubuntu system architecture
Driver model – bus

- platform bus: a virtual bus
- integrated peripherals on SOCs
- “legacy” devices
  - i8042
  - pcspkr
  - serial8250
  - ...

Ubuntu system architecture
Driver model – device

- Every device is represented by an instance of `struct device`

```c
#include <linux/device.h>

struct device {
    ...
    struct bus_type *bus;
    struct device_driver *driver;
    ...
};
```
Driver model – driver

• The device driver-model tracks all of the drivers known to the system

```c
#include/linux/device.h:

struct device_driver {
    ...
    struct bus_type *bus;
    ...
    int (*probe)(struct device *dev);
    ...
};
```
Driver model – driver vs device

- Match is a bus-specific task
- `bus_driver` should provide a match function

```c
#include <linux/device.h>

struct bus_type {
    ...
    int (*match)(struct device *dev, struct device_driver *drv);
    ...
};
```
Driver model – example: pci_driver

- Register with supported device table

```c
#include <linux/mod_devicetable.h>

struct pci_device_id {
    __u32 vendor, device;
    __u32 subvendor, subdevice;
    __u32 class, class_mask;
    ...
};
```
Driver model – example: pci_driver

- `probe()` will be called when matched

```
includes/linux/pci.h:

struct pci_driver {
    ...
    const struct pci_device_id *id_table;
    int (*probe)(struct pci_dev *dev, const struct pci_device_id *id);
    ...
};
```
Early userspace

• We do not want a huge kernel including everything to support booting from various hardwares / setup
Ubuntu system architecture

Early userspace - initrd

• Divide system startup into two phase
• initial RAM disk (initrd)
Early userspace - initrd

- Kernel mounts initrd as rootfs
Ubuntu system architecture

Early userspace - initrd

- Execute `/init`
Ubuntu system architecture

Early userspace – init in initrd

- Load essential kernel modules
Early userspace – init in initrd

• Parse some boot / rootfs related boot parameters
Early userspace – init in initrd

- Convert UUID and LABEL into device name
Early userspace – init in initrd

• Start plymouth
  • Hide scary boot messages from user
Ubuntu system architecture

Early userspace – init in initrd

• Start udev
Early userspace – init in initrd

- Handle crypted rootfs / nfsroot / netboot
Ubuntu system architecture

Early userspace – init in initrd

• Mount rootfs
Ubuntu system architecture

Early userspace – init in initrd

• End udev
Early userspace – init in initrd

- Chroot to rootfs and transfer to init in rootfs
Early userspace – build an initrd

• Created with mkinitfs, based on things in /usr/share/initramfs-tools
Ubuntu system architecture

Early userspace – build an initrd

• Settings: /etc/initramfs/tools/initramfs.conf
• MODULES=[ most | netboot | dep | list ]
Ubuntu system architecture

Real userspace

• upstart is started by initrd
Real userspace

- `upstart` starts/stops services according to runlevel
Thank you

Jesse Sung
jesse.sung@canonical.com
www.canonical.com