Status and Direction of Kernel Development

Andrew Morton
<akpm@linux-foundation.org>
<akpm@google.com>
Linux Foundation
Japan Linux Symposium 2008
July 2008

Overview

- Process
  - The linux-next tree
  - Embedded development
  - X86
  - Realtime
- Technology
  - Filesystems
  - Memory management
  - Hardware virtualisation (Xen/vmware/KVM)
  - OS virtualisation (containers)
  - RAS
  - AIO
  - Tracing

The “linux-next” tree

- A GIT tree operated by Stephen Rothwell
  - Merges together 104 subsystem development trees
  - This represents 85% of the changes in Linux. The remaining 15% is in the -mm tree.
- I will soon feed most of the -mm tree into linux-next so we get close to 100%

Why “linux-next” exists

- To reduce the amount of breakage which occurs during the 2.6.x-rc1 merge window.
  - because it gets additional testing
- To provide an integration tree in which the various subsystem changes can be tested together
- To help persuade subsystem developers to not make conflicting changes
  - I was seeing a lot of this happening. The prominence of linux-next makes people more careful about what they merge.
- To take the "subsystem tree integration" function out of my hands
  - I wasn't doing it very well and it was getting harder and harder

“linux-next” status

- It is going well - perhaps better than I had expected
- It still isn't receiving as much testing as I would like.
- There is some testing by "testers" but few if any "developers" appear to be testing linux-next.
- This is bad because individual developers are now testing their changes in the 2.6.x environment, but that code is destined to be integrated into 2.6.x+1.
- Usually, this doesn't matter
- But one of my secret plans with linux-next was to get the developers testing each others' new work. Because developers make the best testers and bug reporters.
- This has not succeeded.

“linux-next” status (continued)

- People are still raising patches against mainline too often.
- linux-next is the candidate 2.6.x+1 tree, so people should be preparing and testing 2.6.x+1 patches against linux-next.
- But instead people continue to work against 2.6.x, which is the "wrong" tree
- Often the reason for this is to avoid testing other people's new work.

Embedded

- Still an important application of the kernel
- Still under-represented in the core kernel development effort
- But a lot of embedded-related work happens at the architecture support level, and some in drivers and filesystems.
- The CELF Embedded Linux conference in April seems to have been quite successful. It got quite a lot of public attention.

x86

- x86 maintainership has transferred from Andi Kleen over to Ingo Molnar and Thomas Gleixner
- The rate of change has gone up a lot
  - Quite a bit of this has been once-off cleanups and things should ramp down into basic maintenance mode
- I am concerned that there is a lot of platform-level breakage in x86, and our fix rate is low
  - Often affects older machines
  - It is not obvious whether the problems lie in x86, PCI or ACPI
  - Often it is ACPI. Often it is due to hardware or BIOS errors which we need to implement workarounds for
  - These bugs are hard to fix, and impact few people, so it is hard to find people to work on them

realtime

- Important for embedded applications and some financial server-style workloads
- Work from the "-rt" tree continues to be merged into mainline
- I have no estimate of when this will be completed
- But I see no particular blockage in getting it all merged - it is just a matter of doing the work

filesystems

- I think we have problems with filesystems
  - ext3 is old and is getting older. Its feature set is minimal (compression? encryption? checksumming? multi-device?) and performance is often quite poor
  - ext4 addresses a small number of ext3 deficiencies but many will not be addressed and some of the performance problems have quite fundamental causes
  - and ext4 progress is quite slow
- XFS often has better performance
  - but its complexity and narrow support base make XFS hard for vendors to support and enhance

Filesystems (continued)

- I am hoping that btrfs will save us. But as far as I know it is not getting as much external development support as it warrants
  - Merging btrfs into mainline might help here
- I am concerned that SSD technology will catch us unprepared
  - We don't really have a filesystem which is explicitly designed to exploit SSD
  - a large increase in the availability of SSD hardware might expose a Linux shortcoming which we will need to hurriedly fill

Memory Management

- Large changes continue to flow into core MM
  - Mainly large-system support - NUMA and other complex physical memory layouts
  - Some of which is also being used by embedded (SuperH)
- Memory hotplug, per-container resource control, etc.
- Ongoing work with hugetlbfs
- Lockless pagecache is in -mm, should be in 2.6.27
- Major changes to page reclaim (vmscan) are also in -mm, not ready for 2.6.27.

Hardware virtualisation

- As far as I am concerned, VMI and Xen support are fully merged up and are in maintenance mode.
- There is still a high rate of change in KVM support
- KVM support for is64 is in progress

OS virtualisation/resource management

- Work is ongoing
- Network namespace support recently merged, other namespace support ongoing.
- Memory resource controller merged, other resource controllers ongoing
- Generally everything seems to be going OK and the merge plan which we originally decided upon worked out well.
- Ubuntu are now enabling cgroups (to access the fair scheduler?). It is unclear what other distributors' adoption plans are

OS virtualisation/resource management (cont'd)

- It is unclear (to me) which features are still outstanding
- The "namespace virtualisation" and "resource control" aspects of this feature are quite separated
  - Different developers have different interests and work on each part in isolation.
  - Which is good, but it makes overall progress more unclear.

RAS (reliability and serviceability)

- Very little activity here
- Some enhancements to taskstats
- Nothing is happening on driver hardening, kernel messages or anything else

Tracing

- Dynamic tracepoints (via kprobes) have been in place for a long time
- Work is proceeding with static tracepoints. In conjunction with the systemtap developers
- I expect that we will be able to merge the LTTng functionality within a year
- But the whole systemtap situation is apparently not a good one
  - The implementation is hard to use, not as good as Sun's dtrace
  - The systemtap developers are said to be focussing on enterprise distros, not mainline

Tracing

- I expect tracing/systemtap/dtrace to be a hot topic at kernel summit 2008
- Hopefully the end result will be that some more mainline kernel-focussed developers work on improving the kernel's support for systemtap