Android and Linux Kernel

Herding billions of penguins, one version at a time

Sandeep Patil
13 November, 2018
kernel-team@android.com
Agenda

Life of Android device kernels
“The Android Problem(s)”
... and their solutions.

Android Kernel Development Process
Project Treble & Kernel
Questions
Life of Android device kernels

- **Device Kernel**
- **SoC Kernel**
- **Android Common**
- **Stable Long Term Support**
The Android Problem(s)

- Shipping **older** kernels
- Manage **multiple** kernel versions
- **Slower** (non-existent) kernel updates
- **No CI** for kernels!
- **Can’t run/test** mainline kernels with Android
- **Millions** of lines of out-of-tree code.
Older, Multiple kernels

- Oreo: 3.18, 4.4 & 4.9
- Pie: 4.4.107+, 4.9.84+, 4.14.42+
  - ~2 year delay.
- Android MUST continue to work on following kernel versions.
  - 3.18, 4.4, 4.9, 4.14, 4.19
Adding CI

- **LKFT** tests of LTS, rc, android common...
- **kernelci** testing of android common kernels
- LTP improvements: syscall coverage, fixing breakage
- Pre-submit testing on Android kernels using **Cuttlefish**
- Testing from SoC vendors.
Non-existent kernel updates

- Kernel upgrades a huge issue with carriers and vendors alike.
- Oreo: Minimum kernel version defined and required.
- Pie: Minimum kernel version with LTS defined and required.
- Continue moving the needle.
- Include LTS releases instead of Patches in security bulletin
  - “A bug is a bug is a bug”.
No testing targets

- None of the Android devices run mainline kernels.
- Problem for both Android & kernel developers.
- Large amounts of out-of-tree code.
  - Android common
  - Hardware support
Millions of lines of out-of-tree code

- Android common kernel
  - A lot has merged upstream
  - Android v4.19 kernel has **only** about ~30 patches (yay!).
    - 83 files changed, 6474 insertions(+), 173 deletions(-)
    - Numerous changes are dropped in v4.19 as a result of deprecation, user space alternatives and / or upstreaming.
  - Work yet to be upstreamed:
    - Binder priority inheritance, EAS, SDCardFS etc.
Upstreaming out-of-tree HW-specific code

\_\_\_\_(ツ)_/\_

(.. but more on this later)
Project Treble & Kernel

- **Vendor Interface (VINTF)**
  - Collection of versioned HAL interfaces
  - **Linux Kernel**
- **Generic System Image (GSI)**

Frameworks and Builds:
- **Framework Build**
  - Part of the Android build that is hardware-agnostic
- **VINTF Implementation**
  - Part of the Android build that is aware of the hardware and implements the corresponding Vendor Interface (VINTF)
- **Vendor Implementation of HAL Interface**
- **Android Framework**
- **HAL Interface**
- **Linux Kernel**
- **Hardware Components**
- **Applications**
Platform vs Vendor Split

Vendor Implementation of HAL Interface
G is for Generic

Vendor Implementation of HAL Interface

Linux Kernel Modules for SoCs / Peripherals
How can we get there ...

- Kernel symbol namespaces
- Single compiler for Android
  - Both userspace and kernel.
- In-kernel ABI monitoring
Android kernel development Process updates

- Upstream first!

- Proactively report vulnerabilities and work w/ Upstream.

- Mainline, -next, -stable testing on ARM hardware.
  - Same done with Cuttlefish on emulated hardware.
More updates in Android MC

- Userspace low-memory killer
- Userdata checkpoints
- De-staging: Ashmem, Ion.
- DRM/KMS
- Updates on Android’s use of Device Tree.
- LVM, Android and resizable partitions.
Questions?