Android Kernel
How the Sausage is Made

Todd Kjos
September 21, 2018
An update on the Android problem

Android has been a great boon to the kernel community, having brought a great deal of growth in both the user and the development communities. But Android has also been a problem in that devices running it ship with kernels containing large amounts (often millions of lines) of out-of-tree code. That fragments the development community and makes it impossible to run mainline kernels on this hardware. The problematic side of Android was discussed at the 2017 Maintainer Summit; the picture that resulted is surprisingly optimistic.

Greg Kroah-Hartman started by saying that he has been working for some time with the system-on-chip (SoC) vendors to try to resolve this problem, which he blames primarily on Qualcomm for having decided not to work upstream. Qualcomm has since concluded that this decision was a mistake and is determined to fix it, but the process of doing so will take years. The other SoC vendors are also committed to closing the gap between the kernels they provide and the mainline but, again, getting there will take a while.

Google's new rules requiring the use of long-term support kernels with Android and keeping up with updates should also help. If vendors do not follow those rules, he said, he will eventually stop maintaining the LTS releases. For now, though, he is running an experiment where he will support the 4.4.x kernels for a period of six years. Vendors are coming around to using those updates, he said, but there is a new problem in the form of carriers who are proving unwilling to ship those updates. He is trying to get carriers to put one out every six months for now.

Rom Lemarchand, Google's Android kernel manager, said that newer devices are shipping with 4.4 kernels now. The SoC market cycle is such that these chips will always run a two-year-old kernel. The two-year support lifetime for LTS kernels thus didn't work well for SoC vendors; just about the time that they ship something, the support goes away. Hopefully the six-year support period will work better. Updates are still a problem, though; vendors still are working under the mentality that they only need to take patches that have CVE numbers attached to them, which is not the case. Kroah-Hartman added that they weren't even taking all of the patches with CVE numbers. Kees Cook said that none of the vendors have decent testing for their kernels and don't want to merge any changes at all. They don't, he said, want to admit that they are bringing in LTS patches.

Along the lines of testing, there was some discussion of the Linux Test Project (LTP). This project has tended to be viewed dismissively by...
"The SoC market cycle is such that these chips will always run a two-year-old kernel."
## Long-Term Supported Kernels

<table>
<thead>
<tr>
<th>Version</th>
<th>Maintainer</th>
<th>Released</th>
<th>Projected EOL</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.19</td>
<td>Greg Kroah-Hartman</td>
<td>TBD</td>
<td>TBD</td>
</tr>
<tr>
<td>4.14</td>
<td>Greg Kroah-Hartman</td>
<td>2017-11-12</td>
<td>Jan, 2020</td>
</tr>
<tr>
<td>4.9</td>
<td>Greg Kroah-Hartman</td>
<td>2016-12-11</td>
<td>Jan, 2023</td>
</tr>
<tr>
<td>4.4</td>
<td>Greg Kroah-Hartman</td>
<td>2016-01-10</td>
<td>Feb, 2022</td>
</tr>
<tr>
<td>3.16</td>
<td>Ben Hutchings</td>
<td>2014-08-03</td>
<td>Apr, 2020</td>
</tr>
</tbody>
</table>

[https://www.kernel.org/category/releases.html](https://www.kernel.org/category/releases.html)
Android Kernel Versions

- Nougat: 3.18
- 4.4 (Jan 2016)
- Marshmallow: 3.10
- 3.1
- 3.4
- Oreo
- Gingerbread
- Ice Cream Sandwich
- Jelly Bean
- KitKat
- Lollipop

Pie: 4.9 (Dec 2016)

8/31/2018 https://developer.android.com/about/dashboards/
# Kernel Versions on Nexus 5 (Hammerhead)

<table>
<thead>
<tr>
<th>Date</th>
<th>Release</th>
<th>Kernel Version</th>
<th>Time since LTS release (months)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oct 2013</td>
<td>KitKat</td>
<td>3.4.0</td>
<td>17</td>
</tr>
<tr>
<td>Nov 2013</td>
<td>KitKat MR1</td>
<td>3.4.0</td>
<td>18</td>
</tr>
<tr>
<td>Mar 2014</td>
<td>KitKat MR2</td>
<td>3.4.0</td>
<td>22</td>
</tr>
<tr>
<td>Oct 2014</td>
<td>Lollipop</td>
<td>3.4.0</td>
<td>29</td>
</tr>
<tr>
<td>Mar 2015</td>
<td>Lollipop MR1</td>
<td>3.4.0</td>
<td>34</td>
</tr>
<tr>
<td>Oct 2015</td>
<td>Marshmallow</td>
<td>3.4.0</td>
<td>41</td>
</tr>
<tr>
<td>Mar 2016</td>
<td>Marshmallow MR1</td>
<td>3.4.0</td>
<td>46</td>
</tr>
<tr>
<td>Aug 2016</td>
<td>Marshmallow MR2</td>
<td>3.4.0</td>
<td>51</td>
</tr>
</tbody>
</table>
Security Patches

Kernel components

The most severe vulnerability in this section could enable a local malicious application to execute arbitrary code within the context of a privileged process.

<table>
<thead>
<tr>
<th>CVE</th>
<th>References</th>
<th>Type</th>
<th>Severity</th>
<th>Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVE-2018-5703</td>
<td>A-73543437</td>
<td>EoP</td>
<td>High</td>
<td>IPV6 stack</td>
</tr>
<tr>
<td></td>
<td>Upstream kernel</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CVE-2018-9422</td>
<td>A-74250718</td>
<td>EoP</td>
<td>High</td>
<td>futex</td>
</tr>
<tr>
<td></td>
<td>Upstream kernel</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CVE-2018-9417</td>
<td>A-74447444*</td>
<td>EoP</td>
<td>High</td>
<td>USB driver</td>
</tr>
<tr>
<td></td>
<td>Upstream kernel*</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CVE-2018-6927</td>
<td>A-76106267</td>
<td>EoP</td>
<td>High</td>
<td>futex</td>
</tr>
<tr>
<td></td>
<td>Upstream kernel</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Nexus 5 Lollipop Memory Leak

Caused by Nexus 5 kernel tree missing a bugfix from upstream 3.4 LTS tree
Kernel Branch Hierarchy

- Stable Long Term Support
- Android Common
- SoC Kernel
- Device Kernel

<table>
<thead>
<tr>
<th>Device Release</th>
<th>Device Update</th>
<th>Device Update</th>
<th>Device Update</th>
<th>Device Update</th>
</tr>
</thead>
</table>

Clone
Merge
Cherry-Pick
Device Update
"But Android has also been a problem in that devices running it ship with kernels containing large amounts (often millions of lines) of out-of-tree code. That fragments the development community and makes it impossible to run mainline kernels on this hardware."
Adding SoC Support to Linux Mainline

- Currently no way to test cutting edge mobile HW with the latest Linux kernels
- Generally hasn't been done due to opportunity costs
  - Relatively short product lifetimes
  - Once SoC is product ready, team moves on to next generation
- But upstream support can help improve code quality and reduce porting cost for next generation
- Starting to see more upstreaming of SoC support
  - eg. Qualcomm's SDM845
Kernel Branch Hierarchy

Stable Long Term Support

Android Common

SoC Kernel

Device Kernel

Device Release | Device Update | Device Update | Device Update | Device Update
### Experimental 6-year LTS (4.4 and now 4.9)

<table>
<thead>
<tr>
<th>Device Type</th>
<th>Device Release</th>
<th>Device Update</th>
<th>Device Update</th>
<th>Device Update</th>
<th>Device Update</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stable Long Term Support</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Android Common</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SoC Kernel</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Device Kernel</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Clone
- Merge
- Cherry-Pick
## Kernel Versions on Pixel 2

<table>
<thead>
<tr>
<th>Date</th>
<th>Release</th>
<th>Kernel Version</th>
<th>Time since LTS release (months)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oct 2017</td>
<td>Oreo</td>
<td>4.4.56</td>
<td>7</td>
</tr>
<tr>
<td>Dec 2017</td>
<td>Oreo MR1</td>
<td>4.4.88</td>
<td>3</td>
</tr>
<tr>
<td>Aug 2018</td>
<td>Pie</td>
<td>4.4.116</td>
<td>6</td>
</tr>
</tbody>
</table>
"Vendors are coming around to using those [LTS] updates, he said, but there is a new problem in the form of carriers who are proving unwilling to ship those updates"
Goal: Security Updates Contain full LTS Updates

- Linux community performs works hard to ensure updates don't break userspace ABI.
- Adding Linux Test Project (LTP) tests to explicitly verify all syscalls relied upon by Android
- Adding Vendor Test Suite (VTS) tests to verify android-specific functionality like sysfs nodes

Some carriers are getting close to accepting LTS merges w/o full certification.
Project Treble

AOSP Ext

"HALs"

AOSP Ext
With Project Treble, the Board/OEM specific kernel components (mostly device drivers) were separated from the kernel as kernel modules.
Future: Generic Kernel Image (GKI)

We want to eventually have a generic arm64 kernel with all SoC-specific components also separated from the kernel. These could be updated separately.

- GKI Kernel is here
- Kernel Modules for SoC are here
- Kernel Modules for Board are here

Diagram:
- boot partitions
- vendor partition
- odm partition
In-Kernel ABI is "Nonsense"

ABI stability limited to

- Same ARCH (eg. arm64)
- Same Config (eg. wahoo_defconfig)
- Same Toolchain

And some cases like Meltdown/Spectre will break ABI. With tools to catch potential ABI issues in LTS updates, they can be managed:

- Fix in LTS prior to release
- Fix in Android Common
- If can't be fixed, must update all kernel components together

The Linux Kernel Driver Interface

(all of your questions answered and then some)

Greg Kroah-Hartman <greg@kroah.com>

This is being written to try to explain why Linux does not have a binary kernel interface, nor does it have a stable kernel interface.

...
ABI Consistency Tests

More than 30K interfaces to test, frequent issues detected
For example: 4.4.94 --> 4.4.95:

Leaf changes summary: 1 artifact changed (22 filtered out)
Added/removed functions summary: 0 Removed, 0 Added functions
Added/removed variables summary: 0 Removed, 0 Added variable

'struct key' changed:
  type size changed from 1472 to 1536 bits
  1 data member insertion:
    'short int key::state', at offset 960 (in bits) at key.h:165:1
  there are data member changes:
    'unsigned long int key::flags' offset changed from 960 to 1024 (in bits) (by +64 bits)

3410 impacted interfaces:
  ata_port_operations ahci_ops
  ata_port_operations ahci_platform_ops
  ata_port_operations ahci_pmp_retry_srst_ops
device_attribute* ahci_sdev_attrs[3]
device_attribute* ahci_shadow_attrs[10]
...
Reduce the ABI by Namespacing

There are over 30,000 exported kernel interfaces that comprise the (unstable) in-kernel ABI

- Most Android hardware drivers and vendor code are out-of-tree (not in Linux mainline), so they are not verified by upstream developers
- Recently submitted upstream patchset introducing kernel Namespaces has been well received
  - [https://lkml.org/lkml/2018/7/16/566](https://lkml.org/lkml/2018/7/16/566)
  - Reduces ABI risks by declaring which kernel exports are actually used by kernel modules
  - Allows enforcement of future requirement that all out-of-tree drivers use the correct exported kernel interfaces (even when linked to the kernel)
Android Kernel To-Do List

● Eliminate Technical Debt in Android Common Kernel
  ○ Insist on "upstream first" for incoming patches (with few exceptions)
  ○ Drop non-essential patches
  ○ Re-implement in Android userspace
  ○ Send patches upstream

● Work with Vendors and Carriers to Regularly Merge LTS to Device Kernels

● Improve Tests to Ensure safe ABI for Kernel Modules

● Progress Toward GKI
  ○ Start with SoC-GKI : Single Kernel for SoC Family
  ○ Work with Vendors to minimize core kernel changes
THANK YOU