LMG Lightning Talks - Agenda

- LCR Status and Plans - Yongqin Liu
- Moving Towards a unified AOSP and AOSP TV Build - Bero
- Upgrading the kernel on production devices - Bero
- Lsk topic branches - Amit Pundir
- AOSP on Dragonboards - Amit Pundir
- Drm-hwcomposer - John Stultz
- Status update on L2TP in AOSP - Sam Protsenko
LCR Status and Plans

Yongqin Liu@LMG
LCR Builds Current Status

- Support for HiKey and X15
- Updated with latest stable tag, android-8.1.0_r18 now
- Integrated with OP-TEE 2.6.0 for HiKey.
- Tested with latest CTS package, Android 8.1 R3 now
- Tested with VTS from oreo-mr1-vts-release branch
- Other tests include
  - Boot Time measurement for first boot and second boot
  - Memory measurement for first boot and second boot
  - Monkey test to run more than 1 hour
  - OP-TEE xtest
  - 3rd party benchmarks like Antutu6.0, BenchmarkPi, QuadrandPro, Vellamo3, etc
- ~70+ hours test for hikey, ~30+ hours test for X15
- ~50+ patches in total for bugfix/optimization/feature
Future Plan

- P builds released timely, test with P-Preview builds
- New kernel version, Clang compiled kernel/bootloader
- OP-TEE Support for X15 boards and Treble builds
- More packages and more duration for Monkey test
- 100% pass rate with known failures for CTS test
- 100% pass rate with known failures for VTS test
Need More Effort Input

- LCR builds are based on the latest AOSP stable tag which is presumed to be fully tested by google already.
- Plus linaro effort(tests, fixes, optimizations), to quickly provide stable, higher quality reference builds for members and developers.
- Use cases with LCR builds
  - The fix for hwsync timeout problem works for both HiKey and X15
  - With X15 build, some HiKey problem got resolved quickly and vice versa
  - OP-TEE integration reference build
  - As base or comparison target for new feature, kernel version upgrade development
- With help from more members, common problems could be resolved more quickly.
LCR Management System

● Target for Android developer
  ○ find bugs, provide as many as possible information for analysis
  ○ try new features easily with multiple platforms
  ○ feature/result comparison between platforms

● Finished functions
  ○ Generate the test report automatically for each build
  ○ List failed test jobs and resubmit them(one or multiple) with one click
  ○ Show difference between the build and base build to know if there is any regression
  ○ Links with information prepared for bug report
  ○ Comments could be added for each test to understand the issue

● TODOs
  ○ Trend chart
  ○ Flexible comparison between builds
  ○ Enable PreMerge feature
  ○ Fast response for result display
## Result Display Screenshots

### Basic Weekly

<table>
<thead>
<tr>
<th>Index</th>
<th>Job Name</th>
<th>Test Name</th>
<th>Base(21)</th>
<th>Current Build</th>
<th>Bugs</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>basic</td>
<td>memintofirst</td>
<td>40/40/0</td>
<td>40/40/0</td>
<td>0/0/0</td>
<td>100.0%</td>
</tr>
<tr>
<td>2</td>
<td>basic</td>
<td>meminsecond</td>
<td>40/40/0</td>
<td>40/40/0</td>
<td>0/0/0</td>
<td>100.0%</td>
</tr>
<tr>
<td>3</td>
<td>basic</td>
<td>busybox</td>
<td>10/0/0</td>
<td>10/0/10</td>
<td>0/0/0</td>
<td>100.0%</td>
</tr>
<tr>
<td>4</td>
<td>basic</td>
<td>ping</td>
<td>1/0/1</td>
<td>1/0/1</td>
<td>0/0/0</td>
<td>100.0%</td>
</tr>
<tr>
<td>5</td>
<td>basic</td>
<td>lino-android-kernel-tests</td>
<td>54/0</td>
<td>54/0</td>
<td>54/0</td>
<td>100.0%</td>
</tr>
<tr>
<td>6</td>
<td>basic</td>
<td>tijbench</td>
<td>24/0/0</td>
<td>24/0/24</td>
<td>0/0/24</td>
<td>100.0%</td>
</tr>
<tr>
<td>7</td>
<td>weekly</td>
<td>media-codecs</td>
<td>10/2/12</td>
<td>0/0/0</td>
<td>0/0/0</td>
<td>0.93340</td>
</tr>
<tr>
<td>8</td>
<td>weekly</td>
<td>piglit glis2</td>
<td>3/0/3</td>
<td>0/0/0</td>
<td>0/0/0</td>
<td>0.93340</td>
</tr>
</tbody>
</table>

### CTS

The following table shows the latest build status and comments for each test case.

<table>
<thead>
<tr>
<th>Index</th>
<th>Plan</th>
<th>Module</th>
<th>Test Cases</th>
<th>Base(21)</th>
<th>Current Build</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a.b.c</td>
<td>d.e.f</td>
<td>g.h.i</td>
<td>3/0/0</td>
<td>3/0/0/0</td>
<td>pass</td>
</tr>
<tr>
<td>2</td>
<td>a.b.c</td>
<td>d.e.f</td>
<td>g.h.i</td>
<td>3/0/0</td>
<td>3/0/0/0</td>
<td>pass</td>
</tr>
<tr>
<td>3</td>
<td>a.b.c</td>
<td>d.e.f</td>
<td>g.h.i</td>
<td>3/0/0</td>
<td>3/0/0/0</td>
<td>pass</td>
</tr>
<tr>
<td>4</td>
<td>a.b.c</td>
<td>d.e.f</td>
<td>g.h.i</td>
<td>3/0/0</td>
<td>3/0/0/0</td>
<td>pass</td>
</tr>
<tr>
<td>5</td>
<td>a.b.c</td>
<td>d.e.f</td>
<td>g.h.i</td>
<td>3/0/0</td>
<td>3/0/0/0</td>
<td>pass</td>
</tr>
</tbody>
</table>

Moving towards a unified AOSP and AOSP TV build

Bernhard “Bero” Rosenkränzer, LMG
AOSP TV

- We’ve all seen AOSP… But what is AOSP TV?

We may have seen its commercial variant, Android TV
AOSP TV

- AOSP TV is essentially AOSP with a few (harmless on mobile) extra patches and a different launcher for TV/STBs
- Unfortunately, the launcher (Leanback) is not Open Source and not even distributed as a standalone APK or similar...
- … but it IS part of the Nexus Player blob package, so we can pick it up there and put it on relevant boards (e.g. Poplar) for testing
- There is an open API for content delivery, making our DVB project possible
- It would be really nice to have an Open Source replacement for the Leanback launcher -- if any member has an alternative for Leanback, please consider opening it!
Enabling a board for AOSP TV

- Essentially comes down to copying the Nexus Player blobs and adding in the board .mk file:

```makefile
ifeq ($(TARGET_BUILD_FOR_TV), true)
    PRODUCT_CHARACTERISTICS := tv
    $(call inherit-product,
        device/google/atv/products/atv_base.mk)
endif
```

- Also useful: Enable multimedia related drivers in the kernel config -- DVB input, remote control input, ...
Why do this?

- Obviously: additional uses for boards
- Not so obvious: Android TV releases usually lag behind mobile AOSP releases by months -- most real world devices are still on 7.0, Nexus Player (at the time of writing the slides) is still on 8.0, stock AOSP is at 8.1.0_r16. Would be nice to work towards moving AOSP and AOSP TV closer together upstream… And maybe have the launcher opened at the same time
Upgrading the kernel on production devices

Bernhard “Bero” Rosenkränzer, LMG
Problem: Kernels on phones tend to be OLD...

Despite LTS efforts, most phones out there are running really old kernels - sometimes with a few patches from later releases backported - sometimes not even that.

4.4.13 was released on June 8th, 2016 - still used on a phone that got updates in January 2018.

4.4.78 was released July 21st 2017 - better, but still missing out on numerous fixes made since then on a phone getting updates.
Proof of concept: Updating an Xperia XZ1

To see if there’s anything preventing moving to newer releases, we’ve taken an Xperia XZ1 (selected because it has an open bootloader, did a good job at releasing kernel sources, and we weren’t involved in its creation, not leaving us with an “unfair advantage”).

- Did what we think device makers should be doing: Checked out the device’s kernel git, added remotes, and started merging newer releases with “git merge”.
- With an engineering effort of about 1 day (+night), 1 developer, got to 4.4.111 (current LTS within the 4.4 series at the time) - without a lot of effort, updated to 4.4.120 (current as of writing the slides).
- No regressions detected - phone works, LTE works, WiFi works, Bluetooth works, …

A similar community project exists for some Pixel and Nexus devices -- e.g.
https://forum.xda-developers.com/pixel-2/development/walleye-flash-kernel-1-00-t3702055/post75786529#post75786529 These kernels, too, are working as expected.
Out of tree patches

- **Problem:** Most phone kernels still include thousands of out-of-tree patches.
- This wasn’t a very big deal while rebasing within the 4.4 series (several merge conflicts, usually easy to resolve), it will be more problematic when trying to move to a newer series. It would be nice to move an ancient phone to 4.14 (or even 4.16).
Isk-android topic branches

- Isk-v3.18-android is deprecated
- Isk-v4.4-android is actively maintained
- linaro-android-llct is discontinued in favor of aosp/experimental/android-mainline-tracking tree
  - Do attend Android common kernel and out of tree Android patchset status session on Tuesday to learn more.
AOSP on Qcom Dragonboards

● AOSP on db410c
  ○ Linux-v4.14 boots straight out of the box on db410c with WiFi/BT/Audio(?)/Venus(hw accelerated v4l2 codecs) support, which makes db410c an attractive AOSP development board.
  ○ Boots to UI with freedreno and near upstream drm/mesa support
  ○ Limited RAM no longer a major blocker
    ■ Build and run Android with GO profiles for sub 1GB RAM devices
    ■ Switch to 32-bit only userspace build instead of multilib 64_32 build.

● AOSP on db820c
  ○ Getting Started.
  ○ Will co-share AOSP device config with db410c
    ■ And likely most of userspace HAL configurations as well
  ○ Working with QC LT, Feature set status to be shared soon.

● 10 years extend life support and availability through 2025
drm_hwcomposer (HWC2)

- Enables hardware compositing of various layers
- Uses standard kernel interfaces
  - DRM/KMS
  - atomic mode setting
  - dmabuf fences
- Does not require fully open graphics stack!
- Working on:
  - db410c (freedreno)
  - qemu (virgil)
  - hikey/hikey960 (mali)
  - and more!

- Also related work ongoing w/ mesa, gbm_gralloc, libdrm & kernel

- Want to drive wider adoption!
Status update on L2TP in AOSP

Sam Protsenko, Texas Instruments
Agenda

● Context (quick refresh)
● Updates (work delta from BUD17):
  ○ Migrating from Android-N to Android-O
  ○ Productizing the patches
● Current status
Context
Using L2TP from upstream kernel in AOSP

- There are two L2TP implementations in Android kernel at the moment:

<table>
<thead>
<tr>
<th>Upstream kernel</th>
<th>Android kernel</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>net/l2tp/l2tp_ppp.c</code> (PX_PROTO OL2TP)</td>
<td><code>drivers/net/ppp/pppola.c</code> (PX_PROTO OLAC)</td>
</tr>
</tbody>
</table>

- We can drop PX_PROTO OLAC in favor of PX_PROTO OL2TP
What should be changed?

● **external/pppd:**
  ○ Add pppol2tp plugin (from upstream ppp project)
  ○ Make the plugin be loadable by pppd

● **external/mtpd:**
  ○ Use `PX_PROTO_O2TP` instead of `PX_PROTO_OLAC`
  ○ Fallback to `PX_PROTO_OLAC`
  ○ Endianness concerns

● **kernel:**
  ○ Remove `drivers/net/ppp/pppolac.c`
Updates
“Network is unreachable” issue

- Configure eth0 interface and routing table
- “ping 192.168.0.1” doesn’t work
- “ping -I eth0 192.168.0.1” works, but it’s not a solution
- “ip rule show”: main routing table is missing or has low priority (23000)

Fix:

```bash
# ip rule add from all lookup main pref 99
```
“Received signal 15” mtpd issue

- There is a new patch in Android-O kernel:

**ANDROID**: make PF_KEY SHA256 use RFC-compliant truncation

net/xfrm/xfrm_algo.c:

```c
.uinfo = {
    .auth = {
        .icv_truncbits = 96,
        +   .icv_truncbits = 128,
        .icv_fullbits = 256,
    }
```
“Received signal 15” mtpd issue (cont’d)

- It was done for security reasons (RFC4868 compliance)
- Mainline kernel keeps old 96-bit truncation for compatibility
- **Fix:** There are two solutions:
  - Change racoon to another IPSec server (strongswan, libreswan)
  - Revert this patch
- Check IPSec auth algo and truncation:

```
# ip xfrm state
src 192.168.0.100 dst 192.168.0.1
auth-trunc hmac(sha256) ... 128
```
pppol2tp.so is 32-bit instead of 64-bit

- When doing transition to Android-O, I changed next lines:

  In `external/ppp/pppd/Android.mk`:

  ```
  -LOCAL_MODULE_PATH := $(TARGET_OUT_SHARED_LIBRARIES)/pppd/$(VER)
  -LOCAL_UNSTRIPPED_PATH := $(TARGET_OUT_SHARED_LIBRARIES_UNSTRIPPED)/pppd/$(VER)
  +LOCAL_MODULE_RELATIVE_PATH := pppd/$(VER)
  ```

- Which produced two versions of lib:

  1. /system/lib/pppd/2.4.7/pppol2tp.so
  2. /system/lib64/pppd/2.4.7/pppol2tp.so
pppol2tp.so is 32-bit instead of 64-bit (cont’d)

- pppd error:

  E pppd : dlopen failed: "/system/lib/pppd/2.4.7/pppol2tp.so" is 32-bit instead of 64-bit

- Fix:

  +#if defined(__LP64__)
  +#define LIB_DIR                "lib64"
  +#else
  +#define LIB_DIR                "lib"
  +#endif

  -#define _PATH_PLUGIN   DESTDIR "/lib/pppd/" VERSION
  +#define _PATH_PLUGIN   DESTDIR "/" LIB_DIR "/pppd/" VERSION
pppol2tp.so wasn’t built on fresh AOSP build

- There was a rule to build pppol2tp.so
- But it wasn’t triggered anywhere
- **Fix:** pppol2tp must be a dependency of pppd

```
LOCAL_MODULE:= pppd

+LOCAL_REQUIRED_MODULES := pppol2tp

include $(BUILD_EXECUTABLE)
```

- **LOCAL_SHARED_LIBRARIES** won’t work (we don’t want linking)
- **Adding it to PRODUCT_PACKAGES** neglects the dependency path; no go
pppol2tp.so can’t be loaded

● Plugin resides in /system/lib[64]/pppd/2.4.7/
● Error occurs:

E pppd : dlopen failed: library "/system/lib64/pppd/2.4.7/pppol2tp.so" needed or dlopened by "/system/bin/pppd" is not accessible for the namespace "(default)"

● **Fix:** Install the plugin to /system/lib[64]/

   In pppd/pathnames.h:

   -#define _PATH_PLUGIN   DESTDIR "/" LIB_DIR "/pppd/" VERSION
   +#define _PATH_PLUGIN   DESTDIR "/" LIB_DIR
Current Status
Patches

- https://android-review.googlesource.com/#/c/platform/external/ppp/+/330806
- https://android-review.googlesource.com/#/c/platform/external/ppp/+/330807
- https://android-review.googlesource.com/#/c/platform/external/ppp/+/330808
- https://android-review.googlesource.com/#/c/platform/external/ppp/+/330809
- https://android-review.googlesource.com/#/c/platform/external/ppp/+/566042
- https://android-review.googlesource.com/#/c/platform/external/ppp/+/566043
- https://android-review.googlesource.com/#/c/platform/external/mtpd/+/330856
- https://android-review.googlesource.com/#/c/platform/external/mtpd/+/330857
How to use

- Make sure `CONFIG_PPPOL2TP` is enabled in kernel
- After applying patches, the same L2TP + IPsec procedure works as before (both GUI and console ways should work fine)
- If upstream kernel L2TP is not enabled, mtpd will fallback to AOSP one:
  ```
  if (check_ol2tp()) {
      create_pppox_ol2tp(...);
      start_pppd_ol2tp(...);
  } else {
      start_pppd(create_pppox_olac());
  }
  ```
- Check `logcat` output for details
Patches status

- Testing in Google: passed
- Security review in Google: passed
- Patches status: **not merged**
- Why?
What can we do about it?

- Maintenance costs
- Security reasons
- More features (e.g. L2TPv3)
- More review from related engineers
- What else?
Security concerns

● Fixed L2TP issues in upstream kernel:
  https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=linux+kernel+l2tp
● CVE-2008-2750
● CVE-2010-2495
● CVE-2010-4160
● CVE-2012-6543
● CVE-2013-3230
● CVE-2013-7264
● CVE-2014-4943
● CVE-2016-10200
● ...
● How many security issues are in Android kernel L2TP?
Questions?

More details: https://wiki.linaro.org/LMG/Kernel/PPP
Appendix
pppol2tp.so can’t be loaded: full story

- Plugin resides in /system/lib[64]/pppd/2.4.7/
- It can’t be loaded with `dlopen()` from `pppd` binary
- Reason: that path is not permitted path for default linker namespace
- Next errors occur:

```
E linker : library "/system/lib64/pppd/2.4.7/pppol2tp.so" needed or dlopened by "/system/bin/pppd" is not accessible for the namespace:
```

```
E pppd : dlopen failed: library "/system/lib64/pppd/2.4.7/pppol2tp.so" needed or dlopened by "/system/bin/pppd" is not accessible for the namespace ")(default)"
```
pppol2tp.so can’t be loaded: full story (cont’d)

● Bionic loader configuration: /etc/ld.config.txt
● There are three possible variants of ld.config.txt (see system/core/rootdir/etc):
  1. ld.config.legacy.txt: for non-Treble-ized devices
     (PRODUCT_TREBLE_LINKER_NAMESPACES is false)
  2. ld.config.txt: for Treble-ized devices
     (PRODUCT_TREBLE_LINKER_NAMESPACES is true) but without strict linker namespace restriction (BOARD_VNDK_RUNTIME_DISABLE is true)
  3. ld.config.txt.in: for Treble-ized devices with strict linker namespace restriction
● Only 3rd option causes the problem: the default namespace is configured as 'isolated' and system/${LIB}/pppd/ is not in the permitted paths of the default namespace
pppol2tp.so can’t be loaded: full story (cont’d)

- Possible fixes:
  a. Add /system/${LIB}/pppd to namespace.default.permitted.paths
  b. Move your lib under /system/lib64 (or under other paths in permitted.paths)

- Fix: option (b): Install the plugin to /system/lib[64]/

  ln pppd/pathnames.h:

  `-#define _PATH_PLUGIN  DESTDIR "/" LIB_DIR "/pppd/" VERSION
  `+#define _PATH_PLUGIN  DESTDIR "/" LIB_DIR
Configure DNS

- **On host**: Access for public net for the board:
  
  ```
  $ sudo iptables -A POSTROUTING -t nat -o wlan0
  -s 192.168.0.0/24 -j MASQUERADE
  ```

- **On board**: Configure DNS:
  
  ```
  # ndc network interface add 100 eth0
  # ndc resolver setnetdns 100 localdomain 8.8.8.8 8.8.4.4
  # ndc network default set 100
  ```

- **Test**:
  
  ```
  # ping google.com
  ```
Thank You
Questions/Comments:
amit.pundir@linaro.org, yongqin.liu@linaro.org, bero@linaro.org,
semen.protsenko@linaro.org, john.stultz@linaro.org

#HKG18
HKG18 keynotes and videos on: connect.linaro.org
For further information: www.linaro.org