GKI Enforcement Tools
An update on ABI Monitoring
Matthias Männich - Linux Plumbers 2020
Stable ABIs for Android Kernels

Project Treble (Android 8)

Framework Build
Part of the Android build that is hardware-agnostic

Vendor Implementation
Part of the Android build that is aware of the hardware and implements the corresponding Vendor Interface (VINTF)

Android Next Generation

Generic Kernel Image (GKI) (arm64)
4.19.x / 5.x.y

GKI Modules
4.19.x / 5.x.y

Chip- & Board-Specific Drivers (Kernel Modules)

Stable API/ABI

Applications
Android Framework
HAL Interface
VINTF
Vendor Implementation of HAL Interface
Linux Kernel
Hardware Components
Stable ABI within Boundaries

and how Android implements that

<table>
<thead>
<tr>
<th>Branches</th>
<th>Configuration</th>
<th>Toolchain</th>
<th>Scope</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only keep ABI stable within major upstream branch</td>
<td>Single Kernel Configuration</td>
<td>Single Toolchain</td>
<td>Define what is part of the ABI</td>
</tr>
<tr>
<td>Stable per Android version</td>
<td>Suitable for all vendors</td>
<td>Hermetic Build</td>
<td>Symbols</td>
</tr>
<tr>
<td>E.g. LTS 4.19, 5.4, 5.y</td>
<td>Configuration changes allowed if they don't break ABI</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>android-4.19-stable</th>
<th>android11-5.4</th>
<th>android12-5.4</th>
<th>android12-5.yx</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generic Kernel Image (GKI) configuration</td>
<td>Clang Build (only)</td>
<td>Clang tools (nm, objcopy, ...)</td>
<td>Observable ABI</td>
</tr>
<tr>
<td>(gki_defconfig)</td>
<td>Observables ABI</td>
<td>Symbol Lists</td>
<td>Symbol Namespaces</td>
</tr>
<tr>
<td></td>
<td>Observables ABI</td>
<td>Symbol Lists</td>
<td>Symbol Namespaces</td>
</tr>
</tbody>
</table>
Symbols used by Vendor modules need to be
  ○ EXPORTed
  ○ Listed in a symbol list

The GKI binaries export only listed symbols. (trimmed)

KMI symbols kept ABI stable

Additions to the KMI possible even after the release.
Defining the Kernel Module Interface (KMI)

Adding Symbols to the KMI

1. Source code modification in vendor_mod requires new symbol. (build failure)

2. Vendor submit symbol list updates to AOSP

3. Subsequent updates of GKI
   - export c_func for modules
   - keep c_func ABI stable

---

KMI Symbol List
- func1
- func2
- var1
- var3
- a_var
- c_func

Vendor Implementation
- tool support: extract_symbols

Trim unlisted symbols
Libabigail

"Application Binary Interface Generic Analysis and Instrumentation Library"
https://sourceware.org/libabigail/

What's new?

- Support for
  - Clang-built 64bit ARM Kernels
  - Linux 4.19+
  - Modversions (CRC)
  - LTO / CFI
  - Symbol Namespaces
  - Multiple Symbol Lists

- Bugfixes
  - Type equality
  - Textual reporting
  - ...

- Maintainability fixes
  - towards ABI change ⇔ XML change
Tracking KMI breakages

Example:
- **v5 of the change** affected the KMI
- **A change** was added to the series to update the KMI definition (branch was still open for incompatible changes)
Bonus: Using BTF Type Information

CONFIG_DEBUG_INFO_BTF=y

- Contains ABI relevant type information
- Encodes deduplicated type information
- Graph data structure (with cycles)
- Quick to read and process
- Prototype implementation looks promising.
Question?

Matthias Männich  <maennich@android.com>