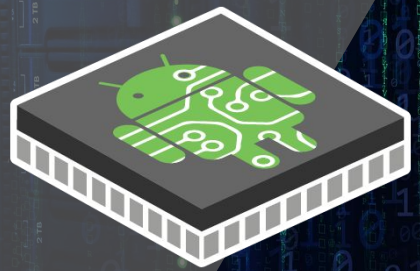
Android Open Source Project in Docker Containers in a Nutshell

Khasim Syed Mohammed
khasim.mohammed@linaro.org



Linaro connect
Bangkok 2019

AOSP on ARM servers

Just like any other distribution Android Open Source Project (AOSP) can be run in & as single or multiple instances on ARM Servers : Two ways to do that

- AOSP on Virtual machines
- AOSP on Docker Containers

Scope : Discuss AOSP on Docker Containers.

Where do we begin

Third : “Host machine” to/from “thin client”

- Means: We need a UI renderer that streams the AOSP UI to thin clients and pass on user inputs to Host.

Second : “Fit AOSP in container” and allow access to “host machine”

- Means: We need a software interface block that interfaces peripherals on host machine to AOSP in container and vice versa.

First : The software should runs on the device - “to make it run inside a container” - on the same device.

- Means: We should port AOSP on the server platform

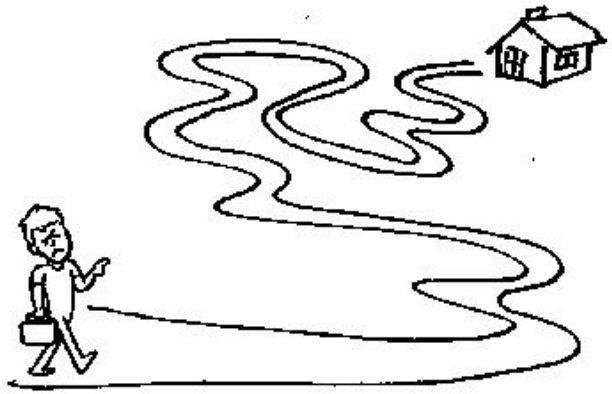
First: Port AOSP to Server Platform

- AOSP lunch doesn't support the device. Hence add it.
  - If you have experience in AOSP bringup on new hardware this step is easy.
  - Just few folders and makefile entries, can follow any AOSP supported hardware like Hikey.

- Kernel provided for server platform should have Android AOSP patches
  - AFAIK, if the kernel is current mainline then the required patches are all in there.
    - @mywork on 4.1 and 4.9 I had to back port few patches on SELinux, Alarm, etc.
  - The kernel configs should include SELinux, fb, Android Configs, etc.
  - Support for display, USB HID drivers
    - AOSP doesn't boot without display. @mywork this wasn't easy as I was first one to tryout display.

- Server setup is not same as embedded hardware platform
  - No u-boot, No fastboot, @mywork - Installing and Configuring Grub was new to me.

YES - far from what we want to achieve … !

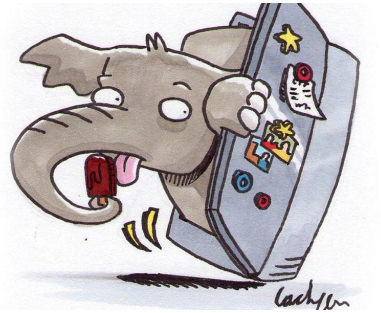


- Docker will need standard Linux distro on server but we ran AOSP on the server ?
- Is AOSP we built in First step docker ready ?
- Will the kernel support both Android and Linux distro ?

By porting AOSP, we have ensured that kernel booting on Server is fit for AOSP.

Assuming you will get distro with docker running on server with the same kernel, will move to second topic.

Second : “AOSP in container”



AOSP in container needs host communication interface.

AOSP needs unrestricted access to peripherals on host machine

AOSP isn’t designed for “many” core, multi instance architectures

AOSP key components aren’t namespace aware

Leverage Open Source Project

Fix or Implement ?

Two interesting Open Source Projects

https://github.com/anbox/anbox

- Uses Linux namespaces (user, pid, uts, net, mount, ipc)
- Android inside the container has no direct access to any hardware
- All hardware access is going through the anbox daemon on the host.
- Reusing what Android implemented within the QEMU-based emulator for OpenGL ES
- Uses different pipes to communicate with the host system

https://github.com/clondroid

- Two different approaches :
  a. Virtualization of smartphones
  b. Virtual smartphone in cloud
- Option a : AOSP is run on Host, LXC and namespace awareness is introduced into AOSP
  a. Peripherals, Services and SurfaceFlinger is shared, each instance has a separate window.
- Option b : With debian rootfs on Host using Android Kernel (similar to Anbox project)

Many Gaps in offering

GRAPHICS

- Not integrated or tested with GPU for graphics acceleration on ARM64 platforms.
- Limits the number of applications that can be run on this platform due to missing EGL extensions.
- Performance impact with soft rendering
- With no soft GPU rendering support in AOSP, the implementation is confined to run on Android N & below version only.

Fix the graphics acceleration.

- Use the hooks provided for OpenGL and integrate graphics acceleration for better performance.

SECURITY

- Platforms are vulnerable to security issues as the containers running AOSP are given high level of permission, same as HOST.

Fix the security issues by using

- Set the required capability to run Android in container using cap-add and --cap drop options.
- Allow Host access to the syscalls required by AOSP using Seccomp filters.

PERIPHERAL ACCESS

- Mouse events aren't recognized as Touch.
- Sensor data needs to be simulated and presented to AOSP for apps to detect rotation, etc.
- Support for Audio, Camera ??

Apps - challenges

Quite a few issues should be solved to make the containerized AOSP fit for running standard apps from playstore.

- Access to PlayStore itself
  - Try with Gapps for quick prototyping.
  - Port Micro G (re-implementation of Google's proprietary Android userspace apps and libraries)
  - Both would deviate from getting Google certifications.

- Browser and Browser plungins need to be fixed as apps trigger web browser to get user credentials for first time.

- Apps have custom libraries that tries to access host software, peripherals - the apps crash when the libraries don't find relevant APIs or permissions to access.

Third : “Host machine” to/from “thin client”

- User can log into the server and get access to display.

- UI can be streamed over network to client device.

The current implementation from open source projects isn’t sufficient for commercializing the concept.

More work is required on this topic and it mainly depends on user experience offered and the target market being addressed.

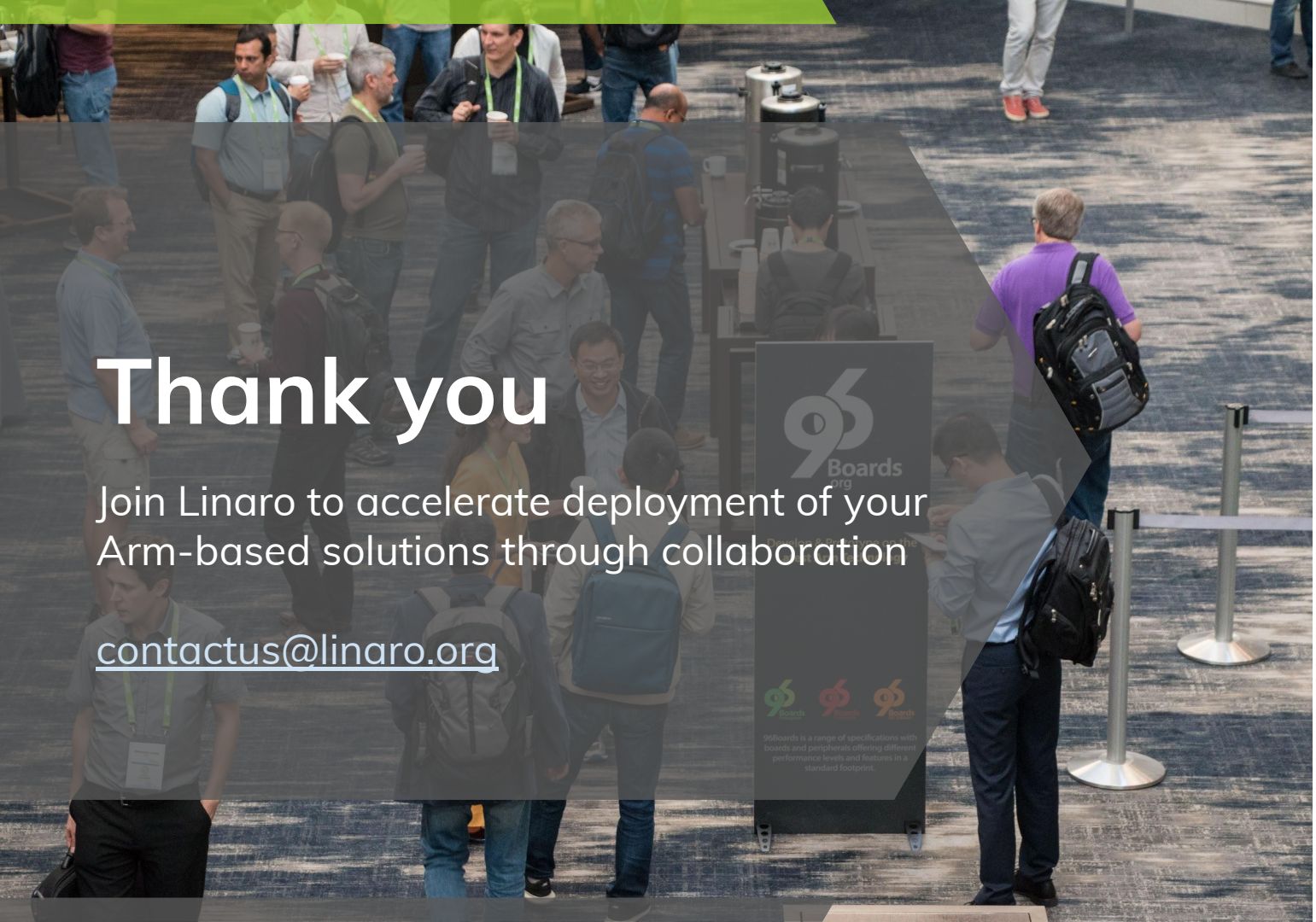
Conclusion - what can be done ?

- Migrate to latest kernel and latest AOSP
- Complete CTS and VTS validation to ensure apps run on the given platform.
- Browsers and Plugin for Browser
- Graphics acceleration and finalize on rendering protocol and framework for better performance
- Enable access to peripherals like camera, location, sensors.
- Performance improvements and productization

Will Stadia change the requirements ?

Linaro
Developer Services

Questions ... ?



Thank you

Join Linaro to accelerate deployment of your Arm-based solutions through collaboration

contactus@linaro.org

Linaro connect
Bangkok 2019