It’s time to collaboratively build an “open source” platform for secure over-the-air updates

Alan Bennett, Linaro, Technologies Division

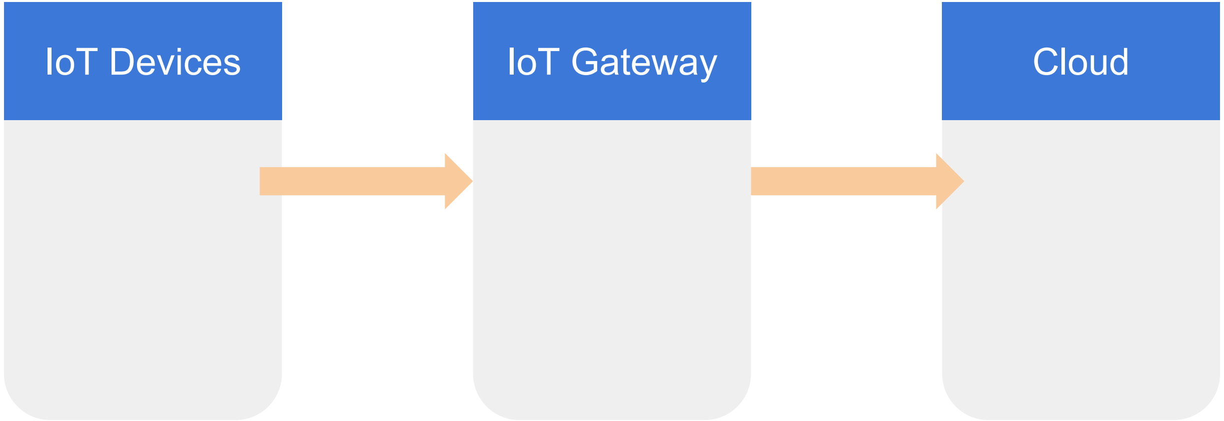
LEADING COLLABORATION IN THE ARM ECOSYSTEM

Linaro’s mission is to lead collaboration in the ARM ecosystem by bringing together industry and the open source community to work on key projects, deliver great tools, reduce industry wide fragmentation and redundant effort, and provide common software foundations for all. The mission is not exclusive to ARM – Linaro can work on other architectures and technologies where the work benefits Linaro members and the ARM ecosystem.

How this got started

In Linaro Technologies, we 'put it all together'

End-to-end market segment references 'Product Quality'
with upstream / 'near-tip software



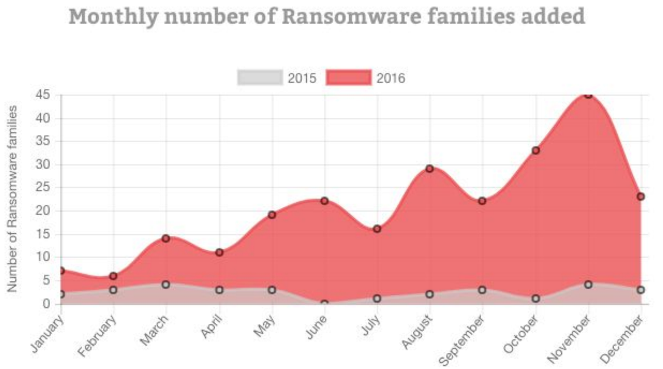
IoT Devices

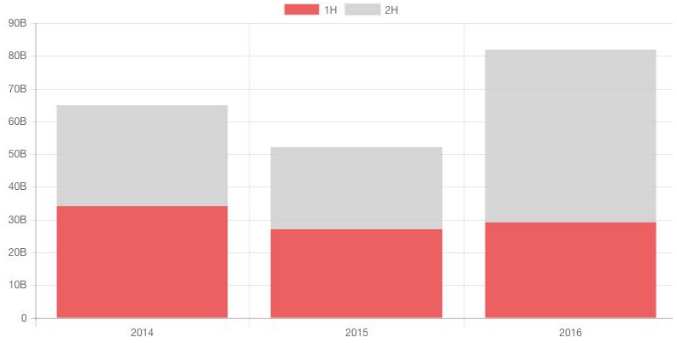
IoT Gateway

Cloud

The Problem

- Connected products are under siege
- Ransomware spiked 752% in 2016
  - and … RaaS is a thing
- IoT and massive DDoS attacks
  - Mirai botnet ~ 100,000 compromised systems
- Account data breaches hit new records
  - 1.5B from only Elex, Bon Secours, Disney, Epic Games, Yahoo!, Washington Dept of Fish and Wildlife, Weebly, Foursquare, FriendFinder, Michigan state, Yahoo, Android
- Landscape is evolving faster than the products
  - Ransom.Wannacry
    - Often times attacked products are EoL, but still used
      - Windows XP, pirated copies attacking us
    - Patches exist and just aren’t applied
- It’s not ok…



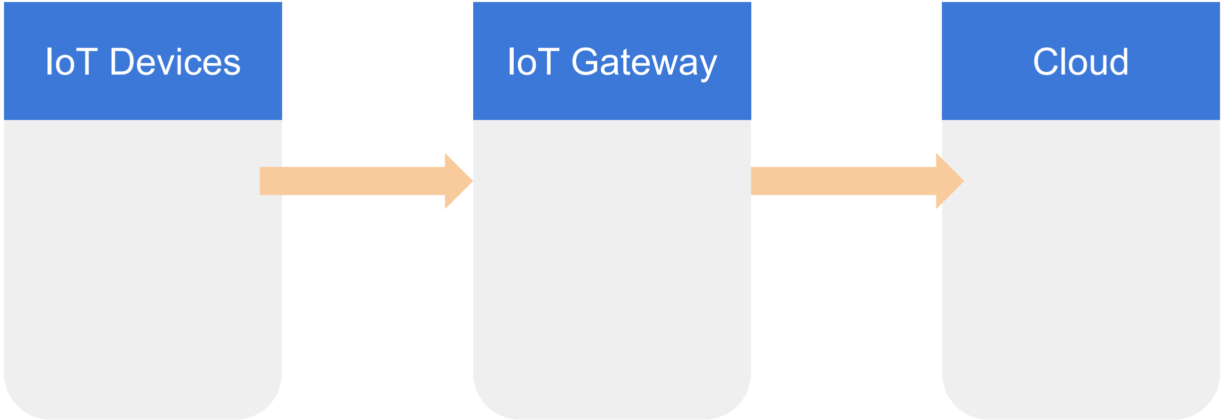


https://www.trendmicro.com/vinfo/us/security/research-and-analysis/threat-reports/roundup

How this got started

In Linaro Technologies, we ‘put it all together’

End-to-end market segment references ‘Product Quality’
with upstream / ‘near-tip software



IoT Devices

IoT Gateway

Cloud

Security Engineering

Key Best Practice Sources

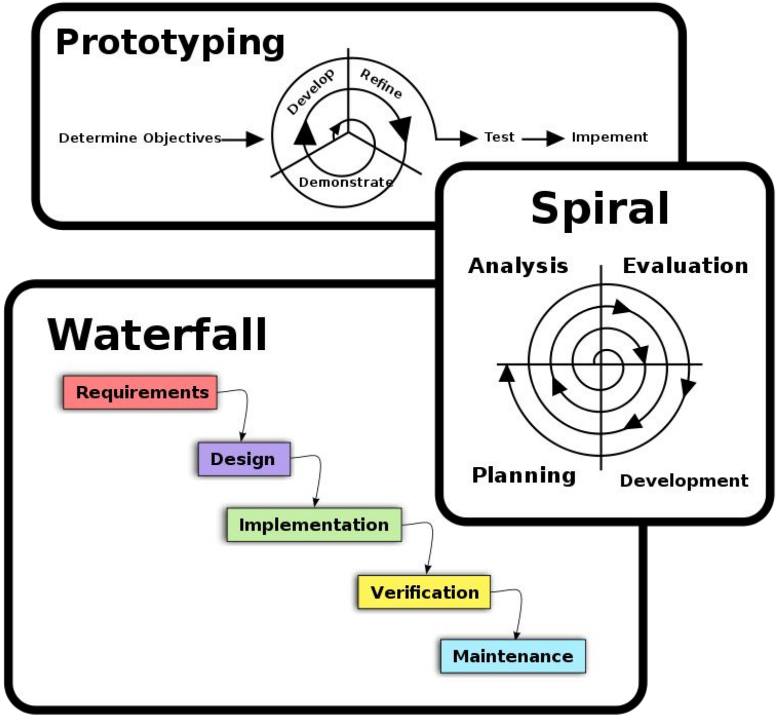
- Security Engineering: A Guide to Building Dependable Distributed Systems - Ross Anderson
  - Outstanding and comprehensive book bringing all the right back to the top
  - Chapter 25 “Managing the development of Secure Systems”

- http://safecode.org/ - Software Assurance Forum for Excellence in Code
  - Non-profit organization dedicated to increasing trust in information and communication tech
  - Managing Security Risks Inherent in the Use of Third-party Components White Paper
  - SAFECode Tactical Threat Modeling White Paper
  -

Security Engineering from the ground up

Security involves more than the product

- Organizational Issues
  - Re-structure, re-organizations, mergers, acquisitions
- Personnel Issues
  - Motivation, stability
  - Organizational structures and uncertainty
- Intrinsic complexity of Software
  - Waterfall Model
    - “Order out of Chaos”
    - Easy clarification of system goals, architecture and interfaces; definite milestones
    - BUT, what if you don’t know the requirements in detail, in advance of development
  - Iterative Model
    - Designers help the customer decide what they want
    - Current Generation is the last build that ‘worked’
    - Evolutionary design and development



https://en.wikibooks.org/wiki/Introduction_to_Software_Engineering/Process/Methodology

Learn from safety critical systems

- All it takes is one exploitable flaw in a connected system
- Methodologies to help manage risk
  - Identify hazards and assess risks
  - Decide on strategy to cope with them
    - Avoidance, constraint, redundancy
  - Traceability down to HW and SW components
  - Minimize attack surfaces
  - Operator procedures
- Identify Failures that could cause accidents
  - Fault tree and Threat tree analysis
- Ultimately mitigate or remove identified hazards
- Find people or build this expertise in your teams

CRITICAL

Threat Modeling

- Applied as soon as an architecture has been established "built-in", not "bolted on"
- Threat models need to be updated
  - Changes to communication, data processing, adding new components, new security controls; Authentication/Authorization; logging, monitoring, alerting; Cryptography
- Activities in threat modeling
  - System Description; i.e. data flow diagrams (DFD)
  - Use cases, misuse cases and abuse cases
  - Identify threats relevant to this system
- Results
  - More product requirements; specifically security requirements that evolve over time after release

Analysis using STRIDE or OWASP top 10 lists

Consider STRIDE for all components

- **S**poofing
- **T**ampering
- **R**epudiation
- **I**nformation Disclosure
- **D**enial of Service
- **E**levation of Privilege

The OWASP Top 10

- A1 - Injection
- A2 - Broken Authentication and Session Management
- A3 - Cross-site Scripting (XSS)
- A4 - Broken Access Control
- A5 - Security Misconfiguration
- A6 - Sensitive Data Exposure
- A7 - Insufficient Attack Protection
- A8 - Cross-Site Request Forgery (CSRF)
- A9 - Using Components with Known Vulnerabilities
- A10 - Underprotected APIs

https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project
https://msdn.microsoft.com/en-us/library/ee823878(v=cs.20).aspx

Is OTA ‘ready’ for open collaboration

From the outside it fits

Problems

- Building secure systems is HARD
- Security threats continuously evolve *‘The street finds its own uses for things’*
- Companies get bored; ship, sustain for a bit, then forget
- Security Expertise is expensive to find or build
- Existing solutions may not fit your use case or needs

Enter collaboration

- Built with experts from around the world
- Across segment groups and companies
- Device management and on-target software, tools & processes
- Successful collaborative projects evolve with their environments
- Open source - community helps to identify & fix flaws

Landscape is congested

Android 5.0 and later

- Block-based OTAs
- Single binary patches
- A/B system updates (seamless updates)
  - Reboot and rollback if OTA fails

ChromeOS

- Delta-compressed over the wire
- A/B partition supporting roll-back
- System sw and user data separation
- Can support Verified boot

Delta updates

- Binary diff’s
- OSTree

Many methodologies, but most are vendor or market segment specific

IoTivity
CoreOS
resin.io
ubuntu
snappy
LinuxKIT
RANCHEROS
androidthings
MENDER.io
Deploy Software Updates for Linux Devices
RED HAT ENTERPRISE LINUX ATOMIC HOST
oma
Open Mobile Alliance
For a Connected World
Ats
ATS GARAGE
uptane

Hopeful about AGL
Contributed code and systems

AUTOMOTIVE GRADE LINUX

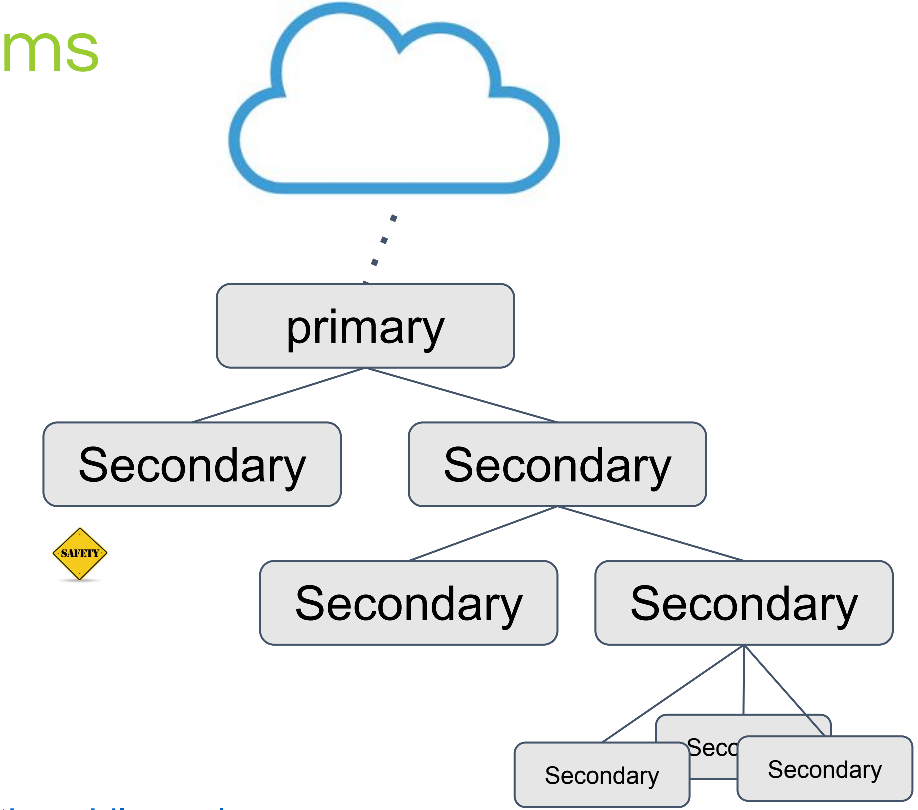
Konsulko Group

Advanced Telematic SYSTEMS

IoTivity

libOSTree

uptane



https://als2017.sched.com/event/AQ5H/iotivity-for-automotive-phil-coval-samsung
https://als2017.sched.com/event/AFC9/secure-ota-updates-for-vehicles-with-uptane-arthur-taylor-ats-advanced-telematic-systems-gmbh

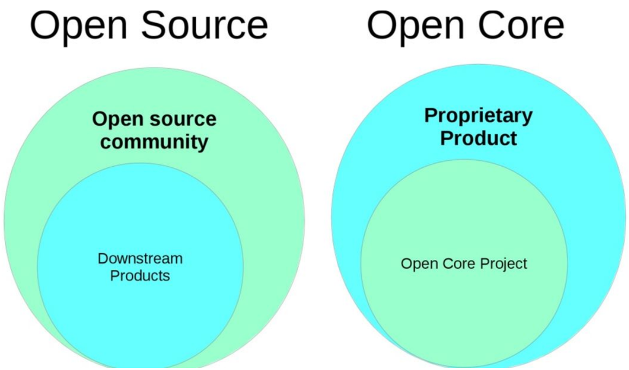
Resist open core OTA projects

Chapter 2: The battle of “open core” software

Open core

is a business model for the monetization of commercially produced open source software. Coined by Andrew Lampitt in 2008, the open core model primarily involves offering a "core" or feature-limited version of a software product as free and open-source software, while offering "commercial" versions or add-ons as proprietary software.

https://en.wikipedia.org/wiki/Open_core

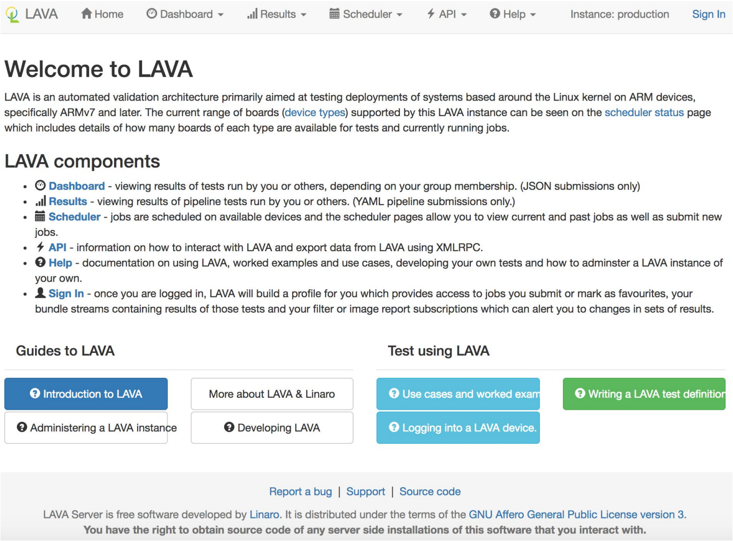


‘easier’ to create business cases around “open core” vs. fully open

It is sometimes difficult to justify “for the greater good” open source

True open projects can be valuable

- In our experience
  - Often times the goal is not as a product
  - Created as a side project to support a larger goal



Welcome to LAVA

LAVA is an automated validation architecture primarily aimed at testing deployments of systems based around the Linux kernel on ARM devices, specifically ARMv7 and later. The current range of boards (device types) supported by this LAVA instance can be seen on the scheduler status page which includes details of how many boards of each type are available for tests and currently running jobs.

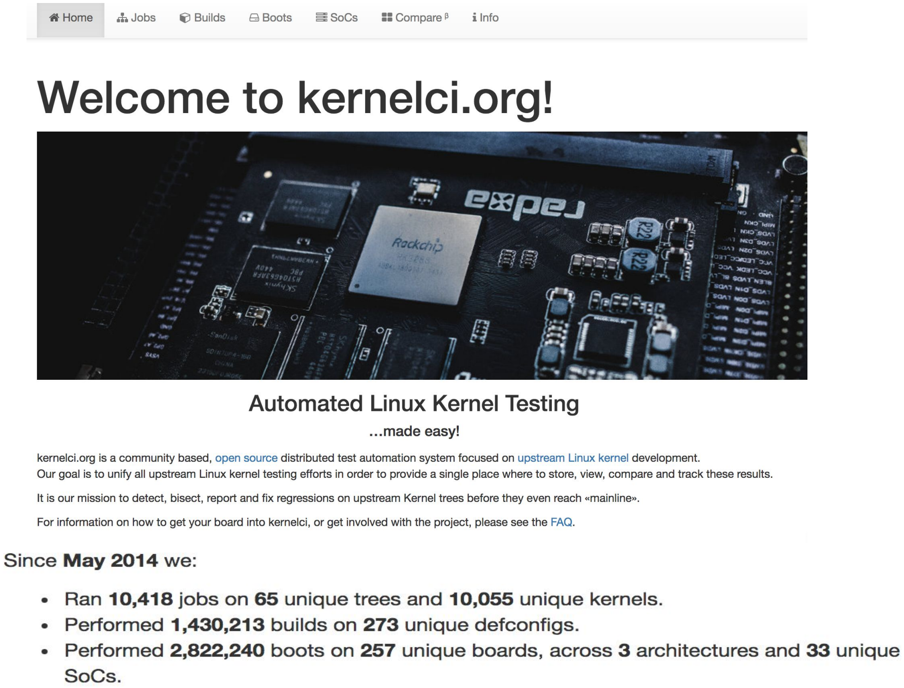
LAVA components

- **Dashboard** - viewing results of tests run by you or others, depending on your group membership. (JSON submissions only)
- **Results** - viewing results of pipeline tests run by you or others. (YAML pipeline submissions only.)
- **Scheduler** - jobs are scheduled on available devices and the scheduler pages allow you to view current and past jobs as well as submit new jobs.
- **API** - information on how to interact with LAVA and export data from LAVA using XMLRPC.
- **Help** - documentation on using LAVA, worked examples and use cases, developing your own tests and how to adminster a LAVA instance of your own.
- **Sign In** - once you are logged in, LAVA will build a profile for you which provides access to jobs you submit or mark as favourites, your bundle streams containing results of those tests and your filter or image report subscriptions which can alert you to changes in sets of results.

Guides to LAVA
Introduction to LAVA
More about LAVA & Linaro
Administering a LAVA instance
Developing LAVA
Test using LAVA
Use cases and worked exam
Writing a LAVA test definitior
Logging into a LAVA device.
Report a bug | Support | Source code

LAVA Server is free software developed by Linaro. It is distributed under the terms of the GNU Affero General Public License version 3.
You have the right to obtain source code of any server side installations of this software that you interact with.

Linaro LAVA - http://validation.linaro.org



Welcome to kernelci.org!

Automated Linux Kernel Testing

...made easy!

kernelci.org is a community based, open source distributed test automation system focused on upstream Linux kernel development.
Our goal is to unify all upstream Linux kernel testing efforts in order to provide a single place where to store, view, compare and track these results.

It is our mission to detect, bisect, report and fix regressions on upstream Kernel trees before they even reach «mainline».

For information on how to get your board into kernelci, or get involved with the project, please see the FAQ.

Since **May 2014** we:

- Ran **10,418** jobs on **65** unique trees and **10,055** unique kernels.
- Performed **1,430,213** builds on **273** unique defconfigs.
- Performed **2,822,240** boots on **257** unique boards, across **3** architectures and **33** unique SoCs.

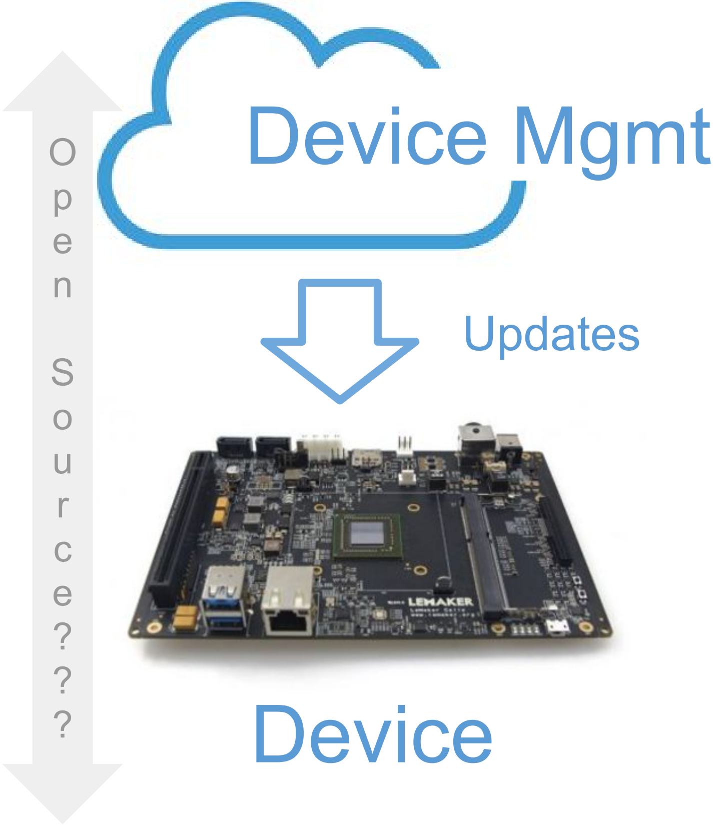
KernelCI - http://kernelci.org

Is true collaboration possible?

- Is open-core open enough?

Ideally, an Open OTA project

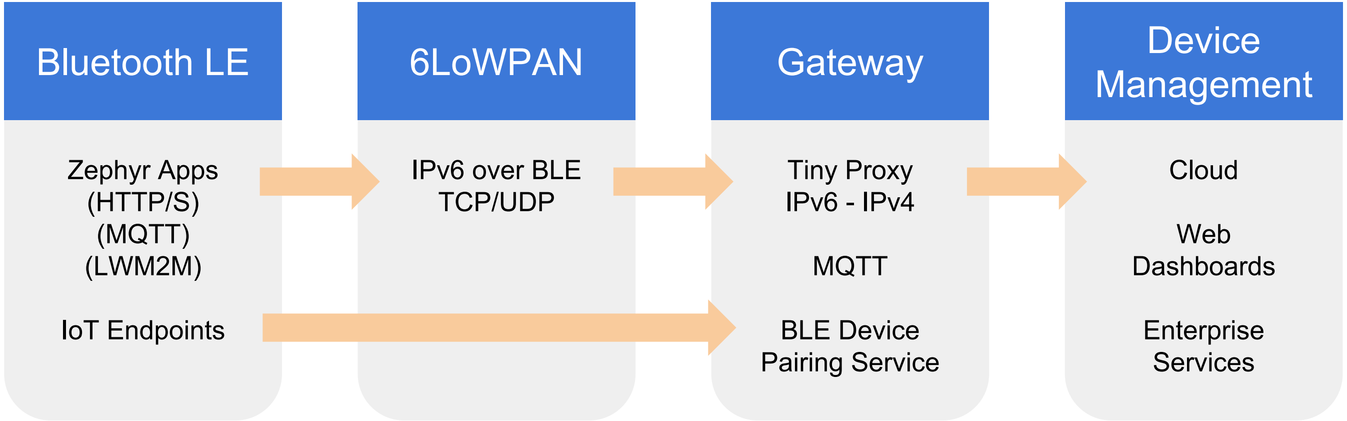
- No vendor lock-in (hopefully lots of choices)
- A community
  - Security experts
  - System builders
  - Cloud providers
- A variety of segments and safety levels
  - Critical / Automotive
  - Infotainment, Consumer, Industrial
- A starting point for system designers



What are we doing in Linaro Technologies?

Linaro IoT End-to-End Demonstration System

- Microcontroller focus
- Zephyr ™ Project (open source collaborative RTOS)
  - Developed with security in mind, delivered on resource constrained devices
  - Neutrally governed, Established and proven development model, Permissively licensed
  - Connectivity protocols optimized for resource constrained devices
- FOTA + Sensor Data flow + End-to-End Integration with PaaS providers



Bluetooth LE

Zephyr Apps
(HTTP/S)
(MQTT)
(LWM2M)

IoT Endpoints

6LoWPAN

IPv6 over BLE
TCP/UDP

Gateway

Tiny Proxy
IPv6 - IPv4

MQTT

BLE Device
Pairing Service

Device Management

Cloud

Web
Dashboards

Enterprise
Services

ARMmbed

LESHAN

hawkBit

amazon web services™

IBM Bluemix™

... Others

Linaro IoT End-to-End Demonstration System **Future**

- Work tightly with Linaro and Zephyr ™ communities
  - Work to meet Zephyr’s secure development guidelines
  - Encryption, key management
  - Bootloader and full FOTA capabilities, Recovery, Rollback
  - A:B with Power-safe updates, Binary deltas
  - Secure boot
  - Generalize the FOTA framework within Zephyr for hosting 3rd party “end-user” applications
- Effectively work to bring a general update solution to microcontrollers

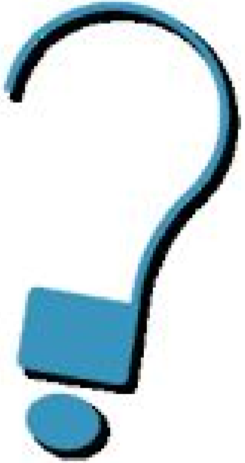
Linaro

LITE
IoT/EMBEDDED

Zephyr™ Project

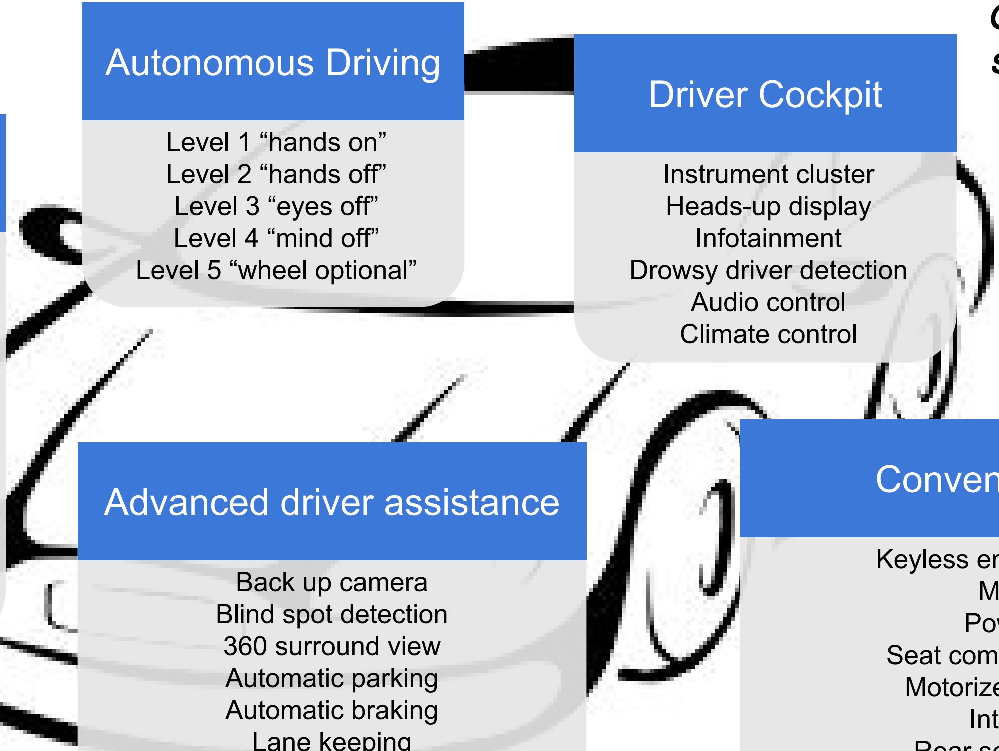
What about more capable systems (> MCU)?

- More complex SoC designs?
  - Not as memory constrained
  - Substantial processing power
  - General-purpose Embedded OS running Linux Kernel
  - Secure boot support in bootloaders; UEFI, uboot, uboot/UEFI
  - Embedded Linux solutions are a well established and fragmented market



Like needed in Automotive

it's just getting started*
Complexity will require
security & updatability



Vehicle Systems

Engine control
Throttle control
Transmission control
Adaptive suspension
Active Steering
Anti-lock braking
Battery management
Passenger airbags
Tire pressure monitoring
Immobilizer and alarms
Telematics
Communication gateway

Autonomous Driving

Level 1 “hands on”
Level 2 “hands off”
Level 3 “eyes off”
Level 4 “mind off”
Level 5 “wheel optional”

Driver Cockpit

Instrument cluster
Heads-up display
Infotainment
Drowsy driver detection
Audio control
Climate control

Advanced driver assistance

Back up camera
Blind spot detection
360 surround view
Automatic parking
Automatic braking
Lane keeping
Pedestrian and sign recognition

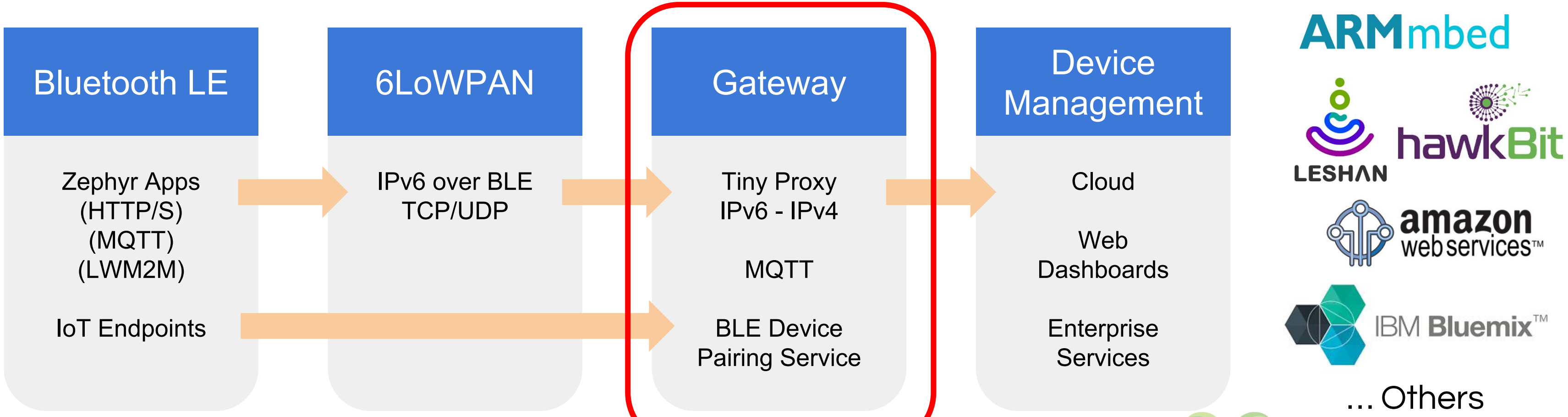
Convenience features

Keyless entry and remote start
Mirror control
Power windows
Seat comfort and adjustment
Motorized trunks lift gates
Interior lighting
Rear seat entertainment
Wipers

So we have created a simple base os

Need a stable hardware platform for our IoT gateway

- Test a variety of ARM 32 and 64-bit platforms
- Wanted to make sure any design was freely available
- Had reasonable upstream Linux kernel support



Bluetooth LE

Zephyr Apps
(HTTP/S)
(MQTT)
(LWM2M)

IoT Endpoints

6LoWPAN

IPv6 over BLE
TCP/UDP

Gateway

Tiny Proxy
IPv6 - IPv4

MQTT

BLE Device
Pairing Service

Device Management

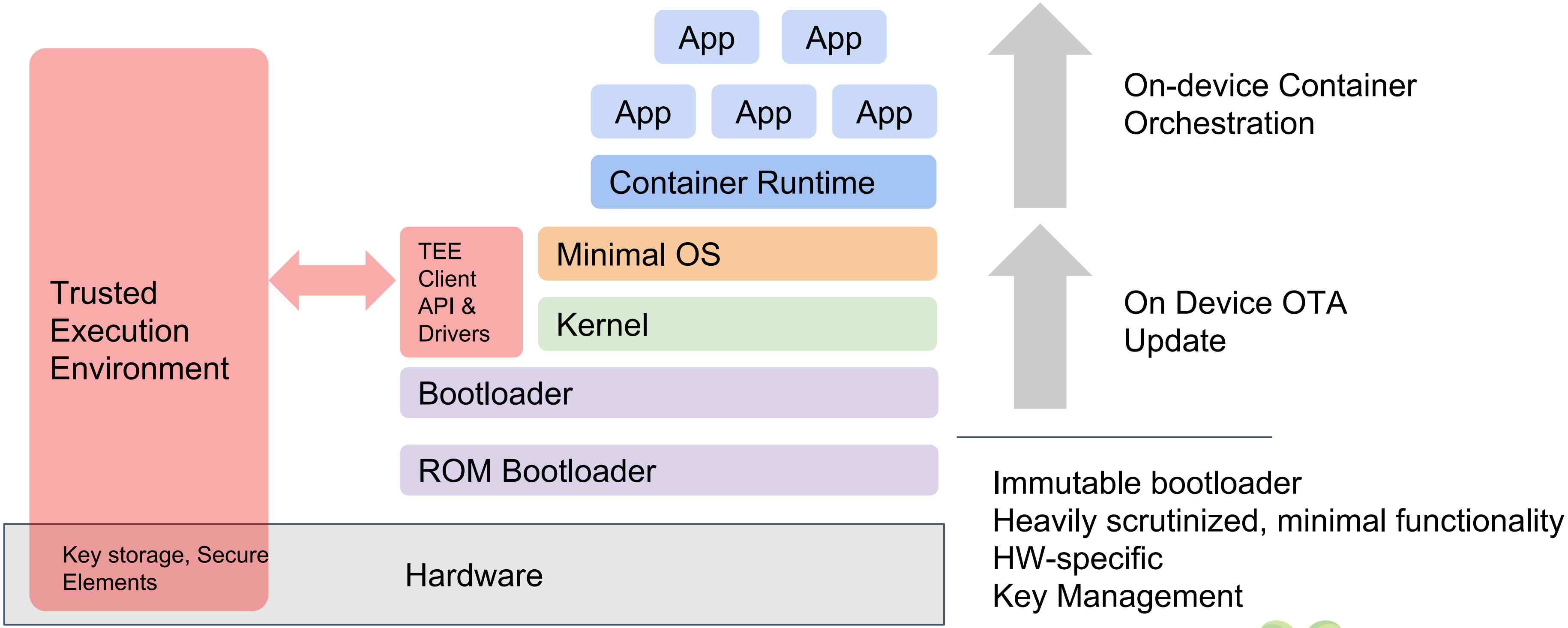
Cloud

Web
Dashboards

Enterprise
Services

... Others

Following a model for secure/updatable systems



App

App

App

App

App

Container Runtime

Minimal OS

Kernel

TEE Client API & Drivers

Bootloader

ROM Bootloader

Trusted Execution Environment

Key storage, Secure Elements

Hardware

On-device Container Orchestration

On Device OTA Update

Immutable bootloader
Heavily scrutinized, minimal functionality
HW-specific
Key Management

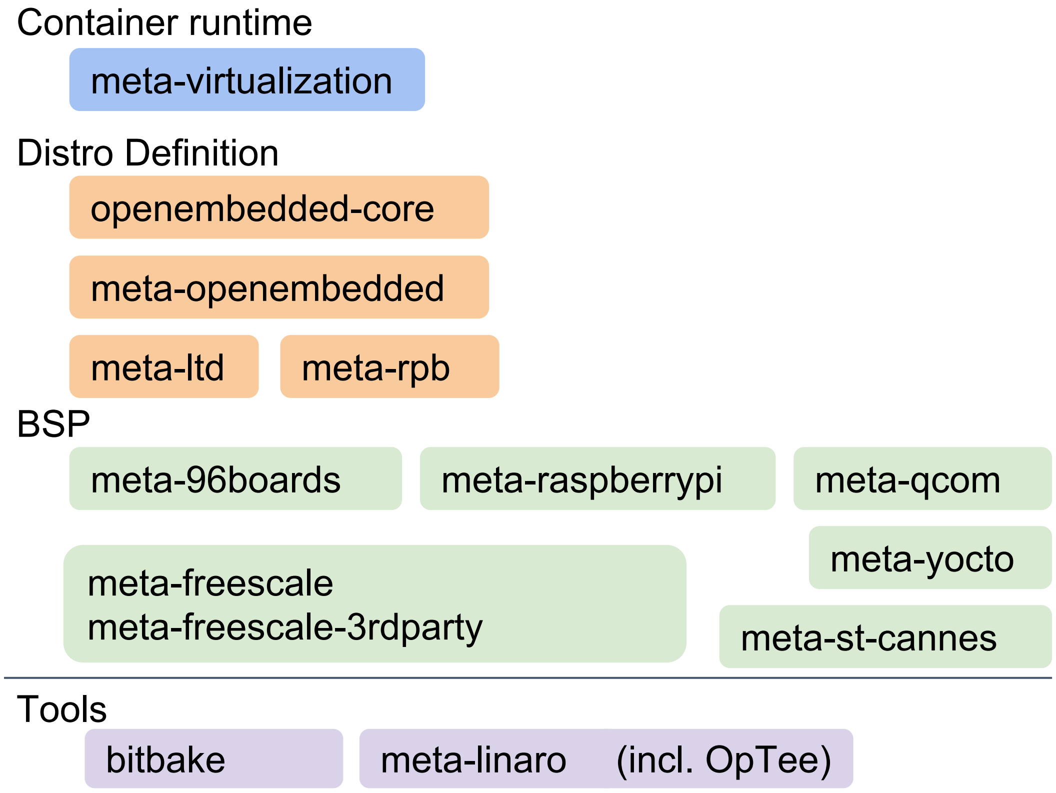
Where are we starting

Hardware

- Working with 96Boards.org to develop secure reference HW

Simple OS base

- Open Embedded (moving)
- a minimal set of Layers
- Unified Kernel / BSP supporting community boards
  - 96Boards (410c, Hikey), Raspberry Pi 3, QC 410c/820c, Beaglebone Black Wireless, i.mx6/7/8
- Virtualization / Docker runtime



Container runtime

meta-virtualization

Distro Definition

openembedded-core

meta-openembedded

meta-ltd

meta-rpb

BSP

meta-96boards

meta-raspberrypi

meta-qcom

meta-yocto

meta-freescale
meta-freescale-3rdparty

meta-st-cannes

Tools

bitbake

meta-linaro (incl. OpTee)

Closing

- Connected devices ‘must’ be updated over their ‘actual’ lifetime
- Companies building connected products often don’t have the security, connected experience to build connected products fast and secure
- Open core is not necessarily open source; join / fund open groups / companies
- Leveraging community and open source, companies can build products, benefiting from others ‘build on the shoulders of giants’
- In the end, companies need to understand systems evolve over time

More Information

End-to-End IoT System / March 2017 release

Documentation (Feb/March 2017)

- http://docs.linarotechnologies.org/fota-demo/index.html

Software Repositories

- https://github.com/Linaro-technologies/

Contact:

alan.bennett@linaro.org

Next Release: June/July 2017

Arigatou gozaimasu