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

Alan Bennett, Linaro, Technologies Division

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.
In Linaro Technologies, we ‘put it all together’

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

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*
Security Engineering
Key Best Practice Sources

  - Outstanding and comprehensive book bringing all the right back to the top
  - Chapter 25 “Managing the development of Secure Systems”

  - 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

- Spoofing
- Tampering
- Repudiation
- Information Disclosure
- Denial of Service
- Elevation 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

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
LEADING COLLABORATION IN THE ARM ECOSYSTEM

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
Hopeful about AGL
Contributed code and systems

hur-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

Linaro LAVA - [http://validation.linaro.org](http://validation.linaro.org)

KernelCI - [http://kernelci.org](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

<table>
<thead>
<tr>
<th>Bluetooth LE</th>
<th>6LoWPAN</th>
<th>Gateway</th>
<th>Device Management</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zephyr Apps (HTTP/S) (MQTT) (LWM2M)</td>
<td>IPv6 over BLE TCP/UDP</td>
<td>Tiny Proxy IPv6 - IPv4 MQTT BLE Device Pairing Service</td>
<td>Cloud Web Dashboards Enterprise Services</td>
</tr>
<tr>
<td>IoT Endpoints</td>
<td>6LoWPAN</td>
<td>Gateway</td>
<td>Device Management</td>
</tr>
</tbody>
</table>

... 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
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

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

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

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

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

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

LEADING COLLABORATION IN THE ARM ECOSYSTEM
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
Following a model for secure/updatable systems

LEADING COLLABORATION IN THE ARM ECOSYSTEM

Hardware

Key storage, Secure Elements

trusted execution environment

bootloader

ROM bootloader

minimal os

kernel

container runtime

app

App

App

App

container orchestration

On-device Container Orchestration

On-device Container Orchestration

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

Linaro Technologies
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-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](http://docs.linarotechnologies.org/fota-demo/index.html)

Software Repositories
  ● [https://github.com/Linaro-technologies/](https://github.com/Linaro-technologies/)

Contact:
  alan.bennett@linaro.org

Next Release: June/July 2017
Arigatou gozaimasu