arm DEV SUMMIT

FOUNDRIES.IO

#ArmDevSummit

(Ultra) LTS for Industrial and Infrastructure IoT and Edge devices

Platforms vs. the kernel

George Grey, CEO

Foundries.io

What are we here to talk about?

What is the problem with LTS?

… it’s not just the kernel any more

Is LTS the only maintenance solution?

Ultra long-term maintenance for IoT and Edge products

It can happen to anyone …

Exploiting a Zero Day Vulnerability

Day One …

You wake up

Your flagship Linux-based smart building elevator range “Pro2”, first shipped 3 years ago, with half a million units deployed, has been hacked

The press want to talk to you

Your security team have been up all night and tells you that they think the problem is a recently reported Zero Day Linux kernel CVE called “Megazilla”

A patch for Megazilla was posted to the mainline upstream kernel a month before the CVE was announced

What happens next?

Are you prepared?

Long Term Support

What does it mean?

It depends on the product

- Phones, Consumer Devices — 3-5 years
- Servers — 3-10 years
- Cars — 15-20 years
- Smart city infrastructure — 20+ years

How do you update your connected product software?

Use case dependent

- You can’t

- Truck roll - Service engineer

- Product recall - Return to service center

- Over the Air Update

What kernel are you using?

Most likely, in the Pro2 smart elevator something more than 3 years old ….

It probably doesn’t matter …

You almost certainly have a bigger problem …

LTS is not the same as Long Term Support

Open Source (and other) Software Projects offer ”LTS” versions

Long Term (Product) Support is a GOOD thing

- Meets customer needs
- Avoids or delays product obsolescence
- Customers and the environment benefit

Alone, LTS Kernels/Distros/Projects do not work

- Stability depends on the whole software stack
- Cost of LTS increases over time
- Use of LTS may ruin your (brand’s) day in the future (or in this case today)

Let’s take a deeper look

LTS Release Offerings

Linux kernel	6 years LTS for annual Linux releases 4.14, 4.19, 5.4
CIP SLTS	10 years support for 4.19 kernel on certain SoCs
Yocto	2 years LTS for Dunfell
Ubuntu	5 years LTS for bi-annual releases 18.04, 20.04, 22.04 ESM contracts for further 5 years
SoC BSPs	2-5 years support typical for a vendor LTS BSP kernel based on latest LTS
U-Boot	No LTS strategy – quarterly rolling releases
UEFI (Tianocore)	No LTS strategy – quarterly rolling releases

SoC Vendor Challenge

LTS Offerings

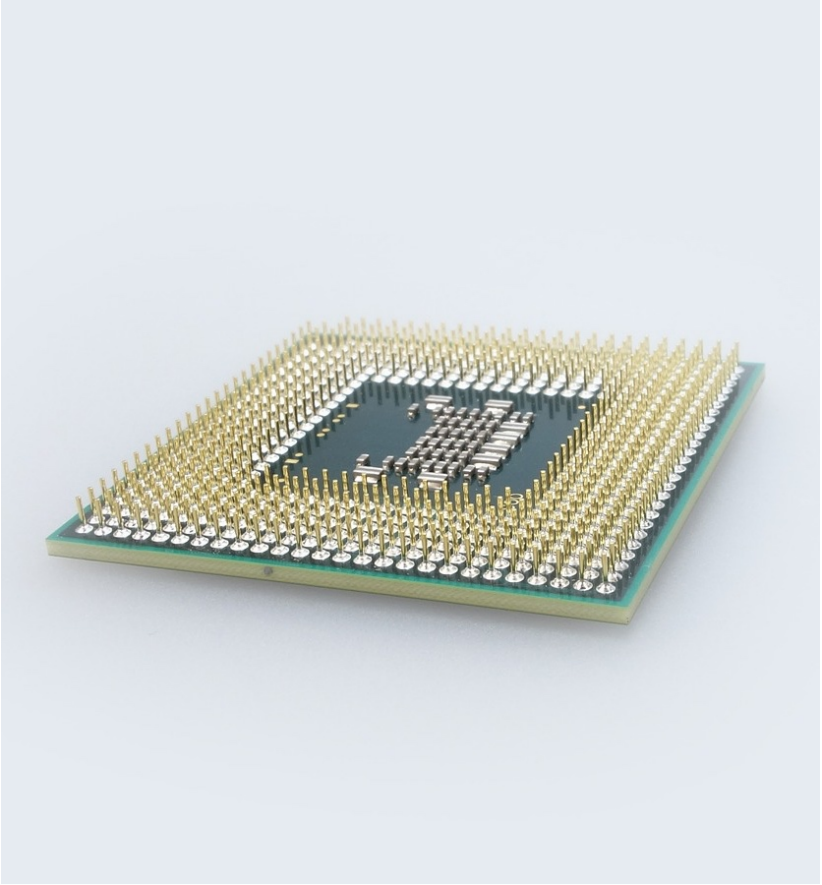
LTS kernels (6 years support for each kernel)
Annual releases, plus SLTS releases (CIP maintains its own kernel tree)

Vendors offer multiple SoCs
5 families with different Arm cores and custom peripheral IP
(GPU, TPU, wireless, storage, network, I/O) is not unusual

Such an SoC vendor supporting only the LTS kernels will need to be maintaining 30 software BSP builds at any one time

Complexity increases as many vendors also maintain large amounts of proprietary code (GPU, media codecs, security IP, AI/ML IP) out of tree

Inevitably SoC Vendor BSPs lag the LTS kernels, often by many months - if they support a particular LTS kernel at all



Real World Device Development

LTS Offerings

Hardware and software choices both need to be considered early in the design process

- For the Arm ecosystem there is variable support for SoCs upstream. If the functionality you need is supported upstream, then you can leverage the kernel LTS support directly
- If you need the vendor BSP, then you also depend on the vendor and their own software, LTS strategy and schedule
- If you add your own kernel changes, then you need to build your own long-term maintenance strategy to maintain those too

What happens after LTS ends?

In house or Professional Services maintenance

Costs rise each year as complexity of maintaining and backporting to older kernels gets harder and harder

There is little re-use of code because the final product kernel is “unique”

Supply is less than demand, so these services become increasingly expensive, and suitable contractors harder to find

As a result many OEMs find themselves unable to maintain products, increasing the risk of business and/or brand damage from malicious attacks

Back to our Zero Day problem …

A day later

A day later

The security response team has been looking at the Pro2 device code

Good news

"Our kernel uses 4.14 LTS with 6 years support till 2022!"

"we think we can OTA update the Pro2 product fleet in the field"

Bad news

"But there isn't a patch yet for 4.14 LTS for Megazilla - the upstream patch doesn't work on 4.14 LTS and the maintainers are still working on it"

"We also need an update to the SoC BSP, but our vendor has the same problem"

"We haven't ever tried to update Pro2 since original testing – it's never had a problem before"

"We've received a ransomware demand for $50M ….."

What went wrong?

Isn’t this what LTS is for?
Yes, but ….

LTS maintenance is hard

The latest software is the most secure

"Megazilla" was fixed upstream before announcement

Next, the upstream maintainers need to backport those patches to all the previous LTS kernels (6 years) – this can take many months

After that, the SoC vendor likely needs to update their own BSP kernel – this can also take months

After that, the OEM needs to re-integrate the updated kernel, and may need to make changes in any custom changes they have made

How do we maintain products for 20+ years

OTA changes everything

OTA and Security

Another threat vector

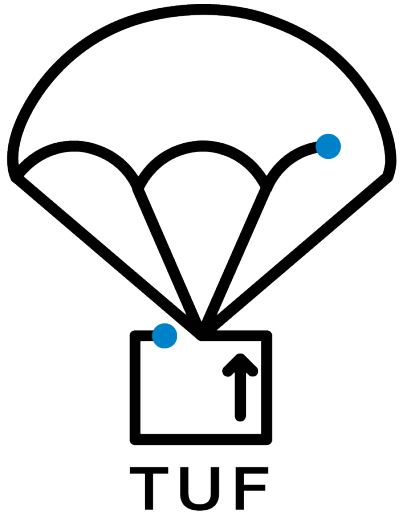
OTA updates will scale from tens to millions of devices

But they must be secure, or they just introduce new exploit opportunities

Is your OTA update system secure?

TUF and Uptane

Securing OTA updates



Uptane

The Update Framework (TUF)

- Cloud Native Computing Foundation project
- Threat-modeled specification for a secure software update system
- Used by companies including Microsoft, Google, Amazon and Docker
- Separation of trust – can address compromised keys by revocation and rotation

Uptane

- Update security framework based on TUF, for the automotive industry
- Originally funded by US Department of Homeland Security
- Capable of withstanding nation-state level attacks
- Handles multi-ECU automotive applications

Secure OTA changes everything

Even if you never have to use it

Development devices can be remote

Tests can be carried out on any device

Production devices can become test devices

New software can be continuously integrated and tested on your production hardware

Deploy OTA updates based on your product, use case, internal process and market need

Until now – Build everything

Maintain over product lifetime

Use in-house engineering or Professional Services

Leverage LTS/S-LTS OSS projects where appropriate

Build and maintain your own unique platform end to end

Live with the cost of ownership and the technical debt

Scarce and expensive expertise is necessary to maintain your own software platform, rather than focusing on your IP and value add

Cost increases exponentially over time as old software is no longer supported and back porting new fixes gets harder and harder

Zero-day exploits remain a business and brand risk

The new recipe

The most secure and most tested software is the latest software

Product Lifetime Maintenance

20+ year examples

Smart City Buildings or Infrastructure

Smart appliances/white goods

Electric cars – expect longer lifetimes than using ICE

Factory automation – Industry 4.0

Platforms for Embedded Products

Check list

It’s not just the kernel
Review your supply chain
– how many vendors are you using?

- ❑ Boot firmware
- ❑ TEE or HSM for trusted applications, key storage, crypto
- ❑ Kernel & Drivers
- ❑ User space/distribution
- ❑ Containers/Application Platform
- ❑ Services & Applications
- ❑ Device deployment & management system
- ❑ OTA update system(s) for all of the above
- ❑ Maintenance – one party or many?

Is the entire stack secure from end to end?

The goal is a maintained platform for everything except your own IP

Strategy for achieving 20+ year support

The latest software is the most secure software

Hardware may last 20+ years, software generally does not

Focus on end to end security and an OTA solution for ALL the device software

Leverage maintained open source software wherever possible

Work with vendor(s) who provide tooling to support your customization at any level

Release and update according to your market and product needs, not your vendors’

Strategy for achieving 20+ year support

The latest software is the most secure software

Maintain a build for your product with the latest software

This is your (ultra) long term maintenance

Even if you never deploy it, this is your insurance policy against a zero-day attack

This is the most important slide in this presentation

Revisiting our Zero Day problem …

The same day

Someone was prepared …

“We have discovered that a new projects team did a PoC on a continuous maintenance solution last month, and they used Pro2!”

“They have been running the latest Linux kernel and have been testing the whole software stack on production Pro2 devices”

“We’re updating the fleet and rotating new security keys as we speak – the hacker can no longer access the Pro2 devices”

Back to the future …

Device Software Development

Where next?

The Future is in the Clouds

This discussion has been about relatively complex devices, typically running Linux

Enterprise Cloud Software is coming to Linux-based embedded, IoT and Edge computing for orchestration of services and applications, and for software development

What about microcontrollers and RTOS?

- Increasing complexity
- Extensive OS fragmentation
- Vertical supplier “silos”

Common architecture and APIs such as Arm PSA for secure updates for both RTOS and Linux are emerging for all connected devices

Thank You!

Stay up to date on the latest from Arm by checking out these developer resources: