YGREKY

Elephants in our Embedded Security Room

Marta Rybczynska

YGREKY

How did embedded development look like 20 years ago?

YGREKY

What has changed?

YGREKY

What has changed?
-> Many things
-> Like: open source has won!

Who am I?

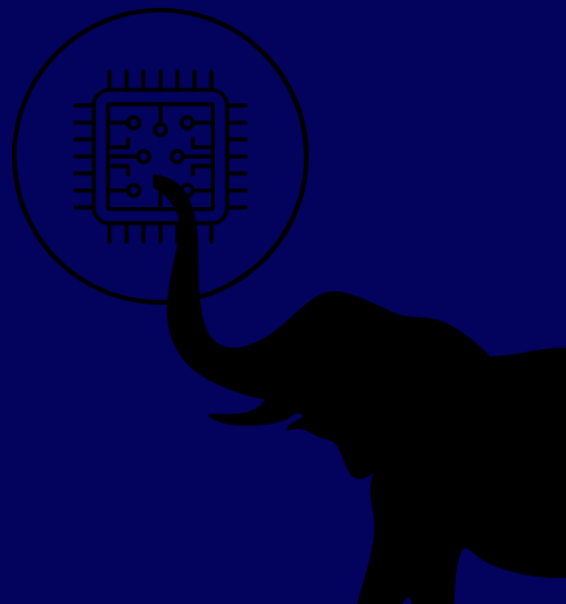
YGREKY

- PhD in Telecommunications
  - On anonymity systems
- 20+ years in open source
  - Contributions to the Linux kernel, Yocto Project, various other projects
  - Security team member of Eclipse Foundation and the Yocto Project
- Strong security focus
  - Security processes
  - Tooling (Yocto Project's cve-check)
  - Those days also: subject around the CRA implementation
- Founder of Ygreky, an open source security company

ALL OPINIONS HERE ARE MY OWN

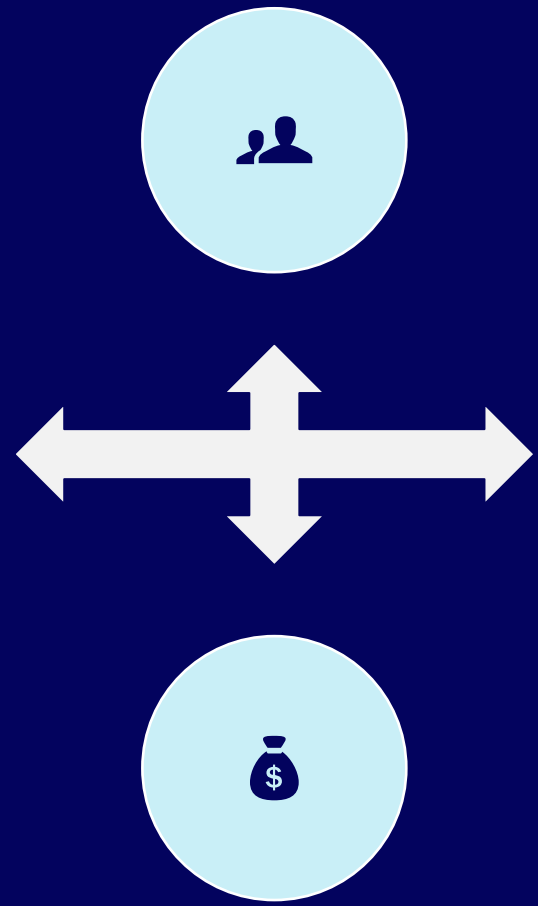
YGREKY

Where are the elephants?



YGREKY

Open source



Device count & scope

Maintenance cycle

Cost reduction & small teams

Open source in embedded
What does it mean

YGREKY

- How it was?
  - Proprietary RTOSes (Real-time Operating Systems)
  - Important licence fees or write your own RTOS
  - Various APIs - libraries needed adjustments

Open source in embedded
What does it mean

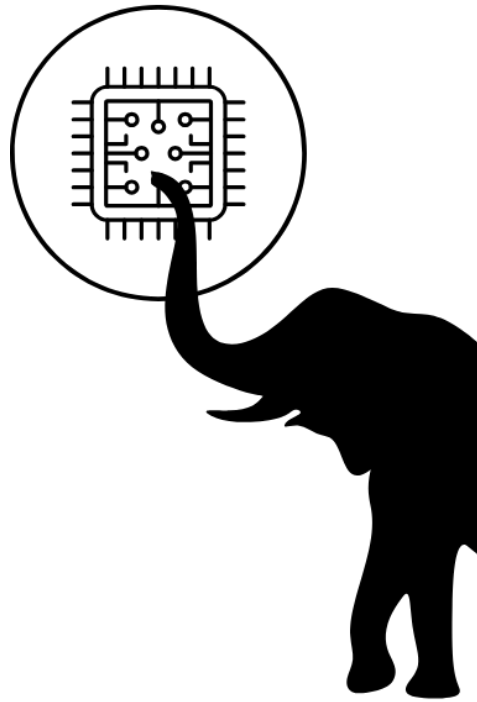
YGREKY

- What it became?
  - Open source operating systems (Linux and RTOSes)
  - More usage of standard APIs - easier to port libraries and software stacks
  - Increased software stack complexity
  - **Small teams can create complex products**
  - Close to zero perceived cost of software

Open source in embedded
What does it mean

YGREKY

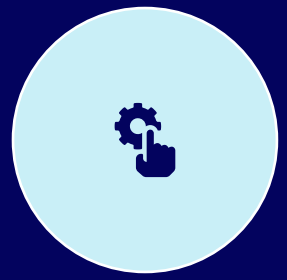
- Structural issues
  - OSS understood as “zero cost”-> changing slowly
  - “If the licence is fine, we can include it” -> maintenance rarely a choice factor
  - Lacking OSS culture -> upstreaming can be slow (even for patches)
  - Frequently “proof-of-concept culture” -> “maker” background

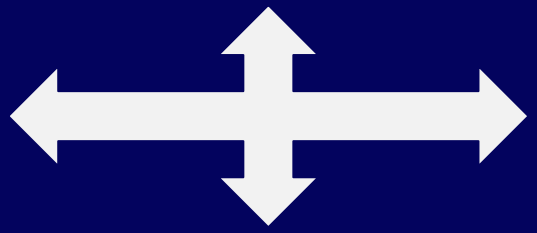


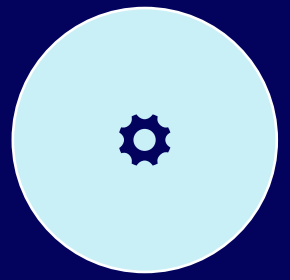
YGREKY

Open source

Device count & scope







Maintenance cycle (lack of)

Cost reduction & small teams

Maintenance cycle
What does it mean

- How it was?
  - Closed devices (fixed functionality)
  - Rare or non-existent updates

Maintenance cycle
What does it mean

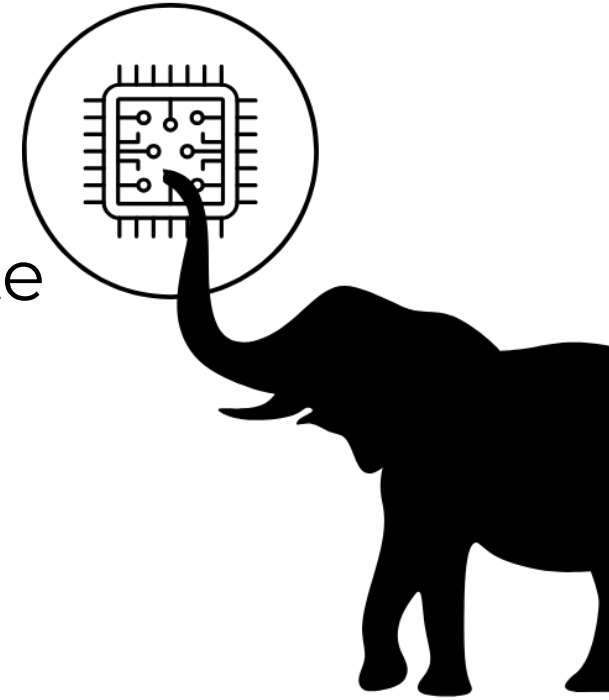
YGREKY

- What it became?
  - **Complex, multi-purpose devices**
  - Presence of updates varies
  - Multiple competing update stacks and home-made integrations
  - Security updates based on CVEs (Common Vulnerabilities and Exposures)
  - Vendor-specific device management solutions
  - (Recent) raise of remote processing and management apps

Maintenance cycle
What does it mean

YGREKY

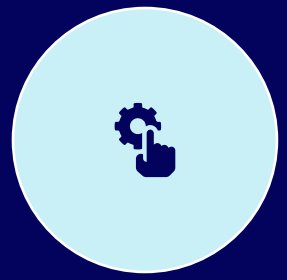
- Structural issues
  - Dependency graph not totally managed (or even not known)
    - Statically linked libraries, copied code…
  - Missing important issues if they don’t have a CVE
  - Board/chip/vendor proprietary stacks
    - Heavily patched software, difficult to update
  - Maintenance frequently not in the planned lifecycle

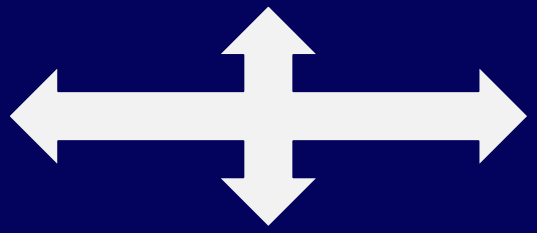


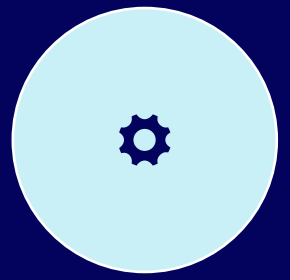
YGREKY

Open source

Device count & scope







Maintenance cycle (lack of)

Cost reduction & small teams

Cost pressure
What does it mean

- How it was?
  - In-house expertise: custom hardware designs, but with a small number of components
  - Licence fees on RTOSes and libraries

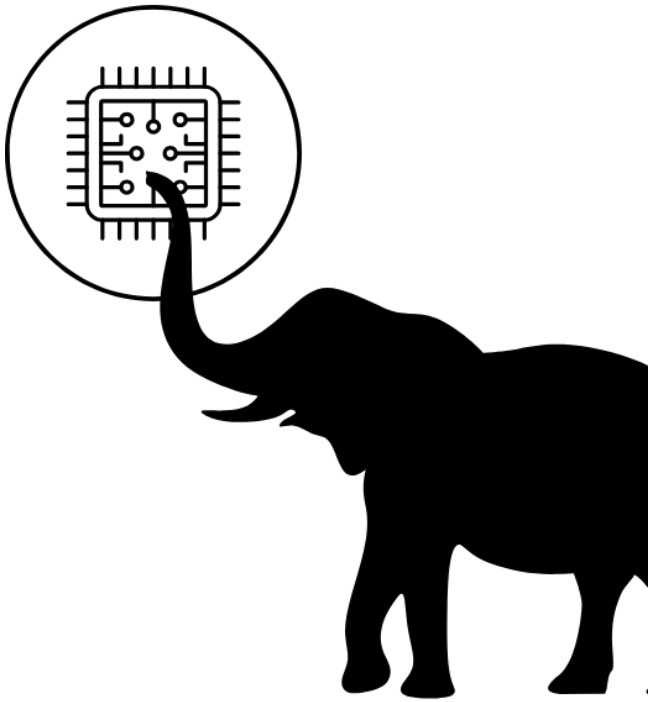
Cost pressure
What does it mean

- What it became?
  - More complex designs, frequently with multiple boards from 3rd parties OR ready-to-use boards
  - Heavy use of open source: **perceived ~0 cost of software**
  - Global market, competition between vendors
  - Maintenance as a business model

Cost pressure
What does it mean

YGREKY

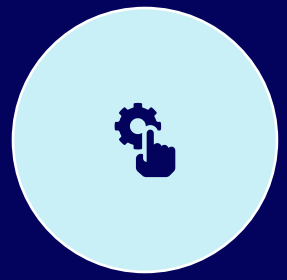
- Structural issues
  - The tendency to fulfill (only) the basic requirements
  - Small teams developing devices more complex than ever
  - Testing coverage not optimal (additional cost)
  - Security as an option, added feature
  - Pression to deliver as soon as possible

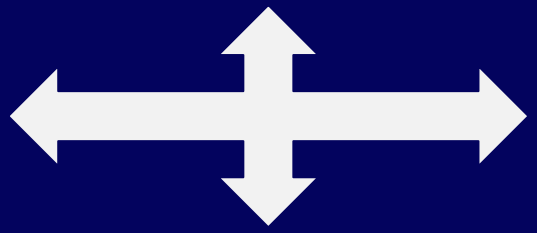


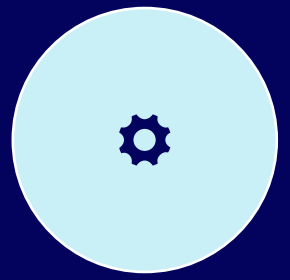
YGREKY

Open source

Device count & scope







Maintenance cycle (lack of)

Cost reduction & small teams

Popularity and scope
What does it mean

- How it was?
  - One-purpose devices
  - No network connection
  - Manual configuration

Popularity and scope
What does it mean

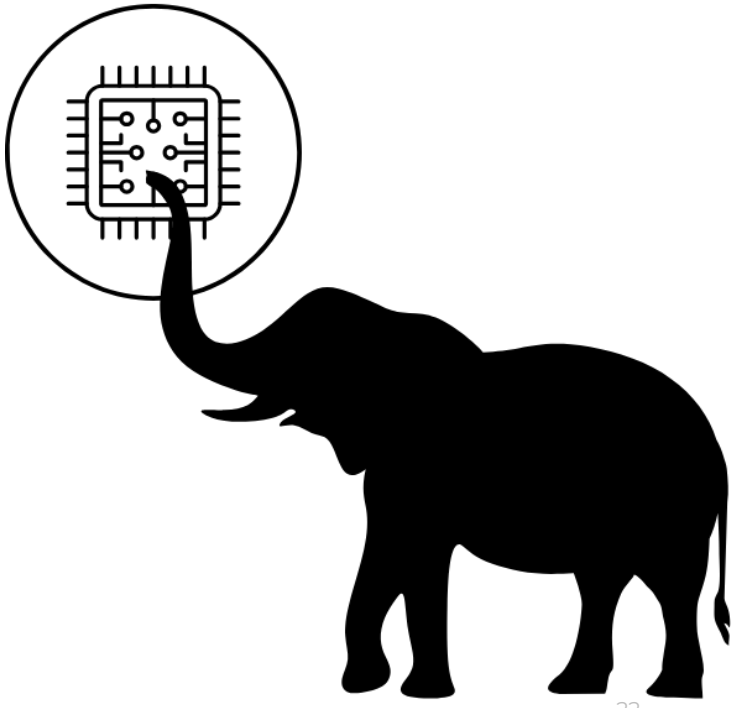
YGREKY

- What it became?
  - Multi-purpose (at least potentially), powerful devices
  - Management possible at scale (remote processing, control application using vendor’s server)
  - **Business-critical, sometimes life-critical**

Popularity and scope
What does it mean

YGREKY

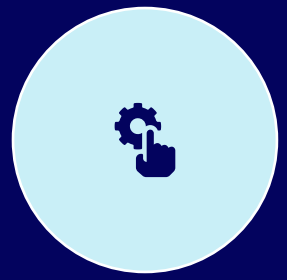
- Structural issues
  - Devices became business critical -> development mindset didn’t follow
  - Attacks on embedded devices are profitable
  - Management web interfaces -> often outsourced, unmaintained

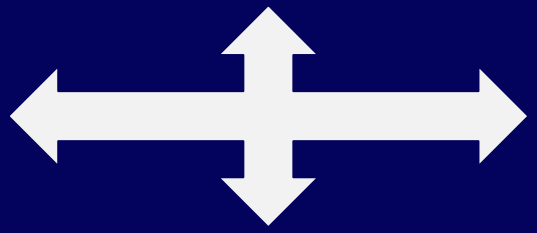


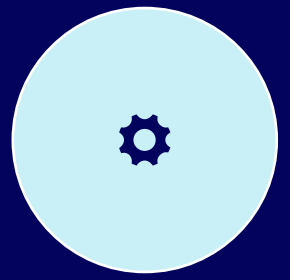
YGREKY

Open source

Device count & scope







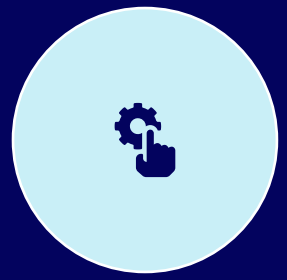
Maintenance cycle (lack of)

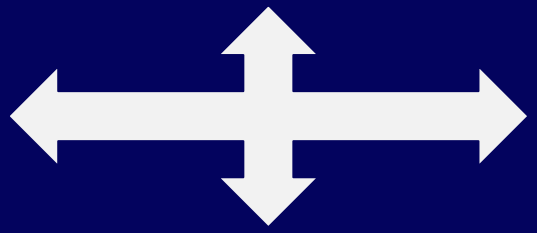
Cost reduction & small teams

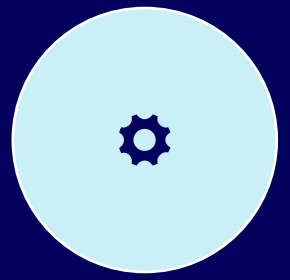
YGREKY

Open source

Device count & scope







Maintenance cycle (lack of)

CRA (and other regulations) cause

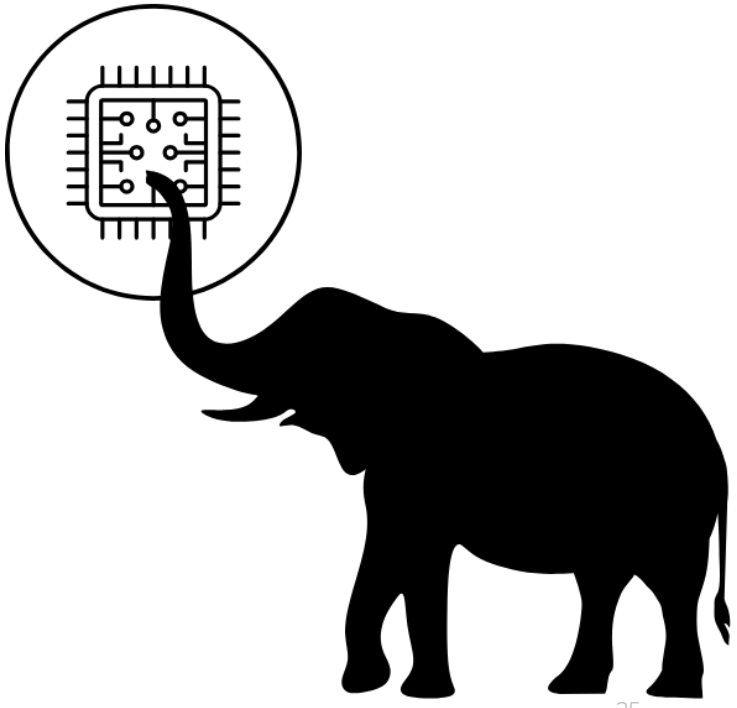
CRA (and other regulations) impact

Cost reduction & small teams

So, what next?
Predictions for the next two years

YGREKY

- Big companies will start moving first
  - Risk of fines (and fines) are higher for them
- Paperwork vs real work
  - “Let’s do the minimum possible”
- There will be panic in 2027

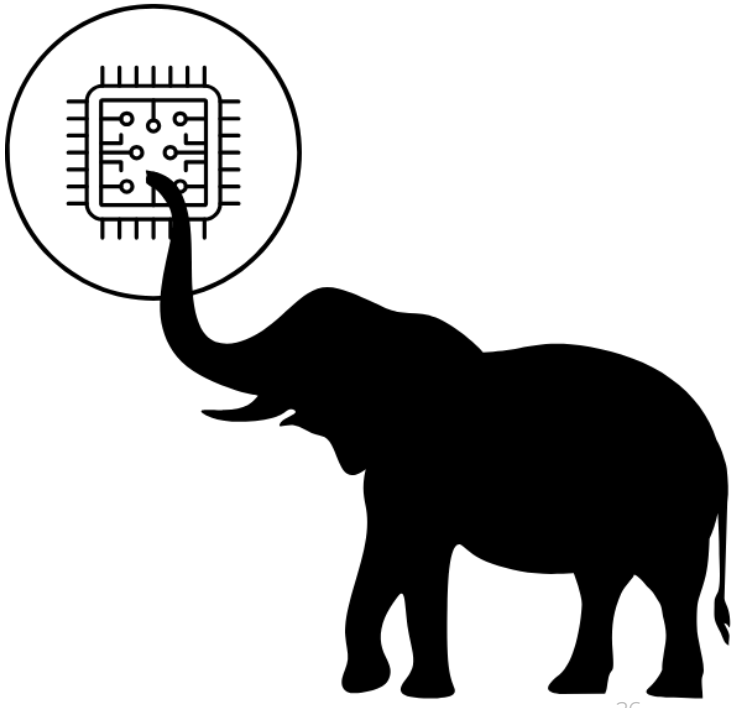


If you want to act now

To avoid problems in 2027… and do the right thing

YGREKY

- Software updates are a must
  - You can fix much by an update
- Review your dependencies
  - If the licence is good, it doesn’t mean it’s a good idea to include
  - Would you maintain it for 10 year?
- Move to the “update by default” model
  - Instead of waiting for a CVE
  - Requires either stable APIs, or good test coverage
- Learn and apply
  - Learn about security best practices in embedded
  - Input them to product plans (including cost)



Questions?

Contact: Marta Rybczynska
marta.rybczynska@ygreky.com
LinkedIn:
https://www.linkedin.com/in/mrybczynska/