End-to-End Open Source IoT with RIOT

www.riot-os.org
Emmanuel Baccelli (*Cédric Adjih)
Inria
on behalf of the RIOT Community

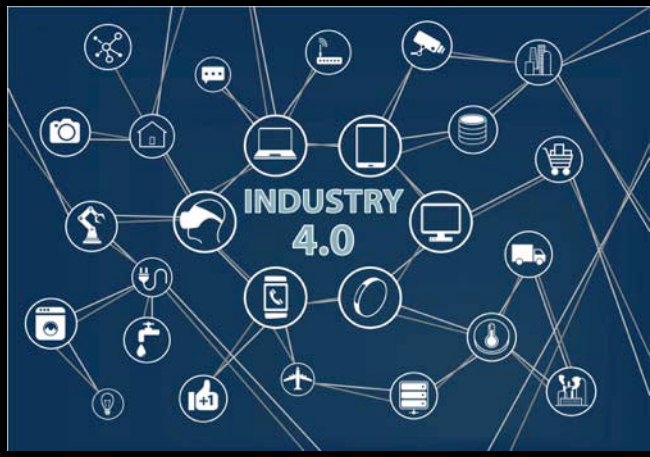
Agenda

- Which IoT are we talking about?
- Why an OS for Low-End IoT devices?
- How?
- What is RIOT?
  - Solving IoT technical challenge 1: constrained devices
  - Solving IoT technical challenge 2: interoperability
  - Solving IoT technical challenge 3: trust

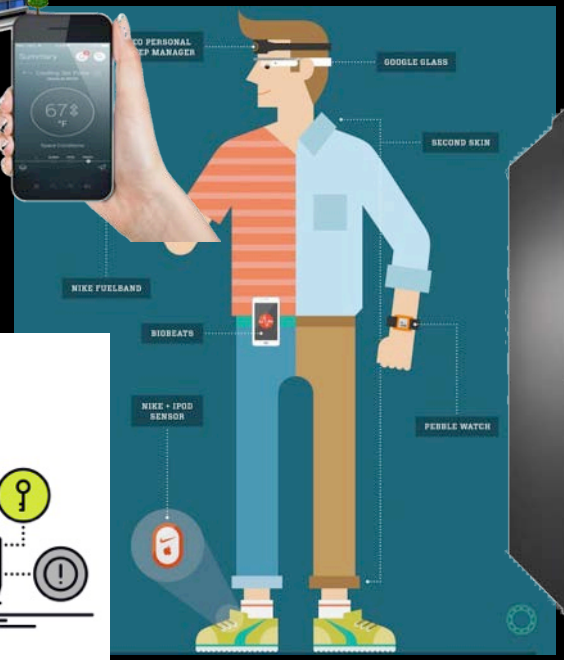
Many Expected IoT Applications

Building Automation

Industrial Automation

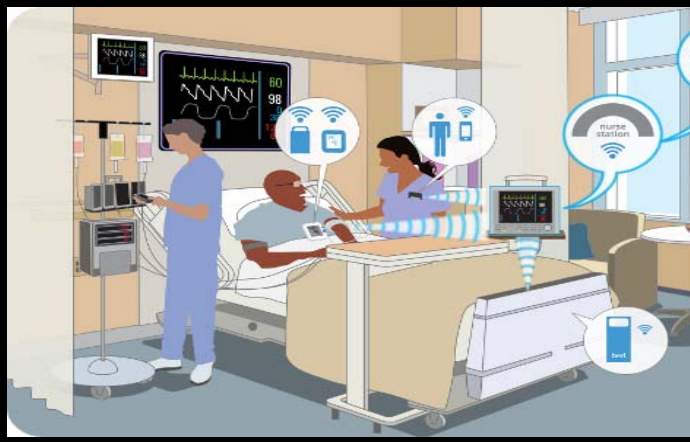


Connected Vehicles



Wearable technology

Smart homes



eHealth

IoT = Future Internet's extremity

Benefits: extremely big business…

But also costs: extreme challenges w.r.t.

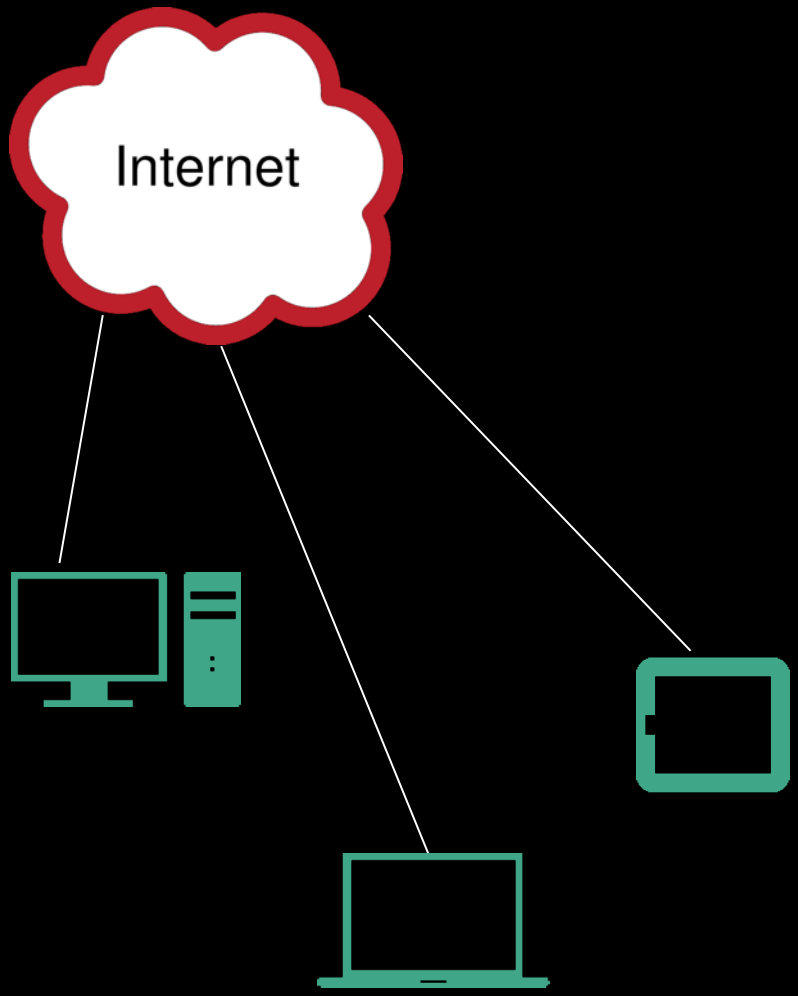
- Interoperability
- Performance w. constrained hardware
- Trust, privacy & network security

Definition of IoT?

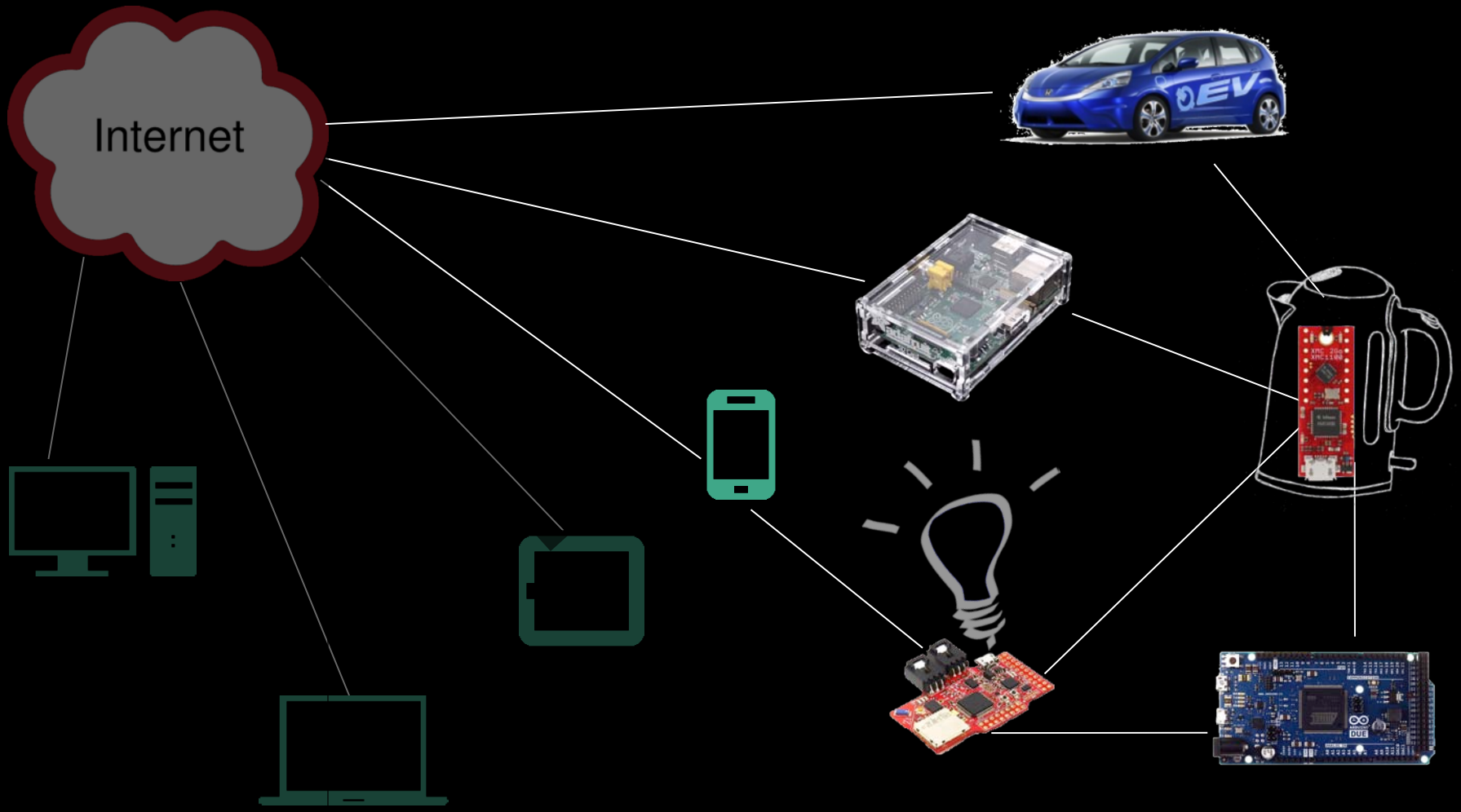
- IoT ≈ *internetworking of physical objects—devices, vehicles, buildings and other items embedded with electronics—which enables such objects to collect & exchange data.*

- IoT is also known as:
  - M2M (Machine to Machine)
  - Physical Web (Google)
  - Physical Computing (Massimo Banzi, Arduino)
  - Internet of Everything (Cisco)
  - World Size Web (Bruce Schneier)
  - …

Zoom: IoT Hardware



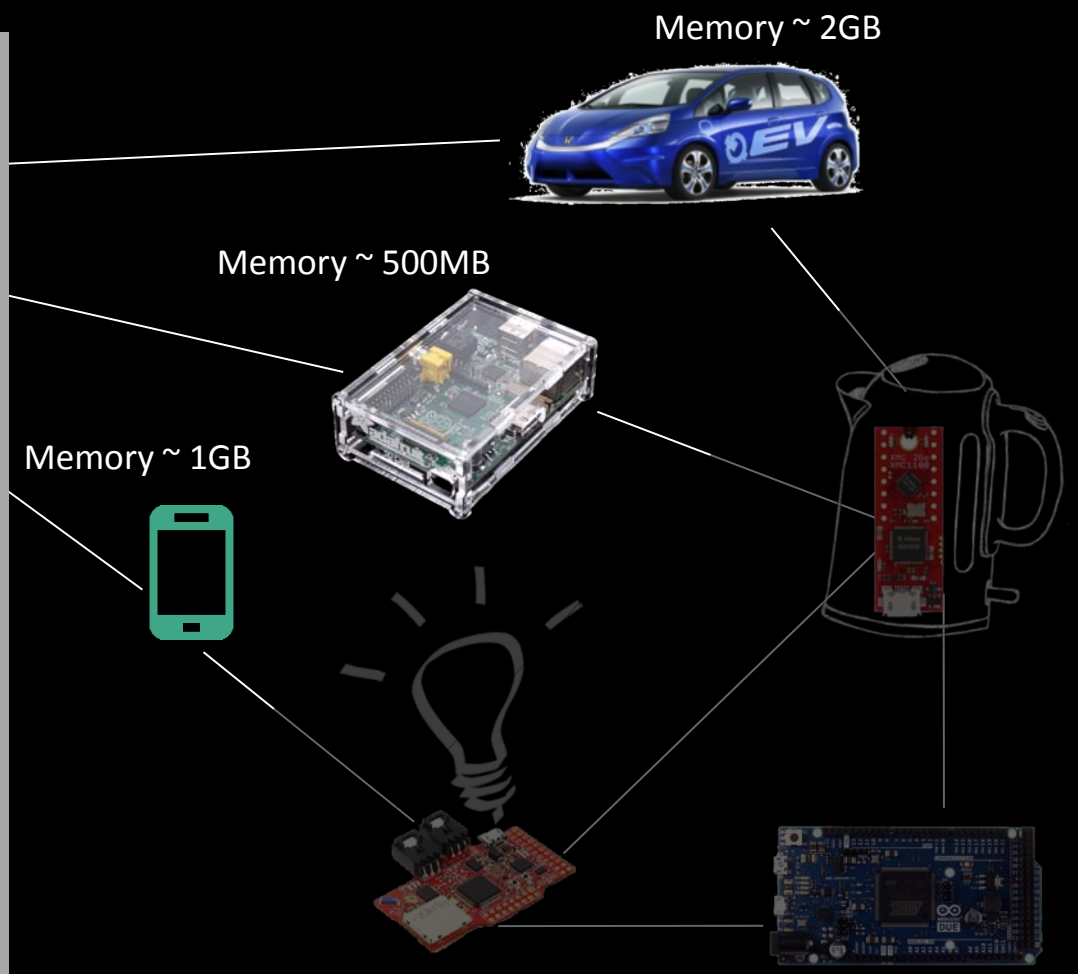
Zoom: IoT Hardware



High-end IoT Hardware

- single-board computers such as
  - ✓ RasberryPi, connected cars, smartphones…
- hardware resources similar to average Internet devices
  - ✓ memory, computation power, network throughput…

→ **Can run usual TCP/IP protocols**
→ **Can run usual OS such as Linux**

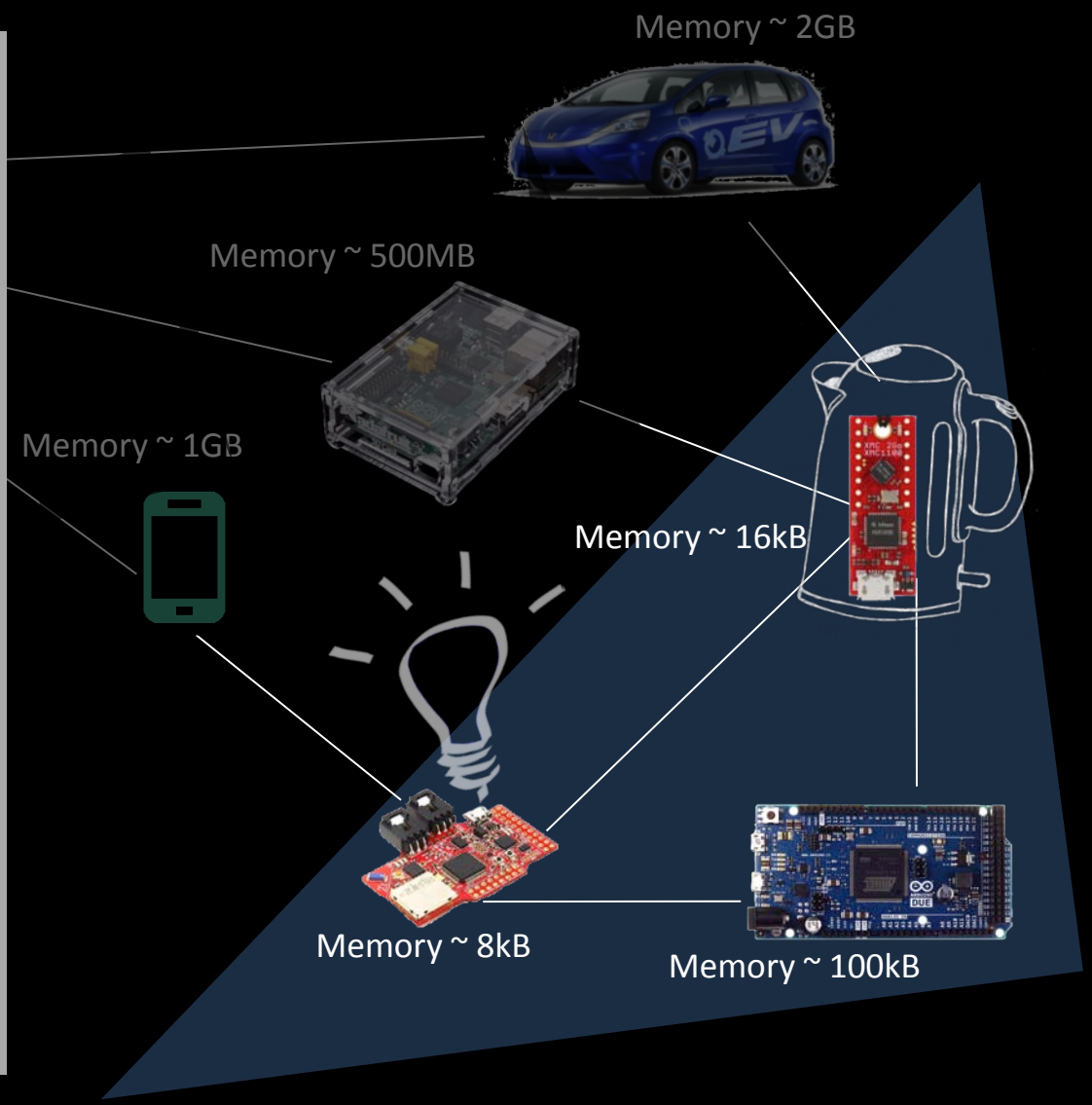


Low-end IoT Hardware

- Smaller & cheaper
  smart objects
- Low-power
  microcontrollers & radios

ENERGY
  Milliwatt instead of Watt
CPU
  Megahertz instead of Gigahertz
Memory
  Kilobytes instead of Gigabytes

→ **KEY CHALLENGE**
for communication protocols
for software platforms



Agenda

- Which IoT are we talking about?
- Why an OS for Low-End IoT devices?
- How?
- What is RIOT?
  - Solving IoT technical challenge 1: constrained devices
  - Solving IoT technical challenge 2: interoperability
  - Solving IoT technical challenge 3: trust

Why a software platform for Low-end IoT devices?

- Linux, Android… bare-metal?

Memory ~ 32kB

Memory ~ 8kB

- But as IoT software evolves…
  - more complex pieces, e.g. an IP network stack
  - evolution of application logic

- … non-portable IoT software slows innovation

  90% of IoT soft. should be hardware-independent

  → this is achievable with a good software platform (but not if you develop bare-metal)

How to achieve a good software platform?

- Experience (e.g. with Linux) points towards:
  - **open source**
  - **free core**
  - **driven by a grassroots community**

Indirect business models

Geopolitical neutrality

- But technically, departure from Linux is needed

Software platform on low-end IoT devices?

- The good news:
  - no need for advanced GUI (simple shell is enough!)
  - no need for high throughput performance (kbit/s)
  - no need to support dozens of concurrent applications
- The bad news:
  - kBytes of memory!
  - typically no MMU!
  - extreme energy efficiency must be built-in!

Software Platforms for Low-End IoT Devices

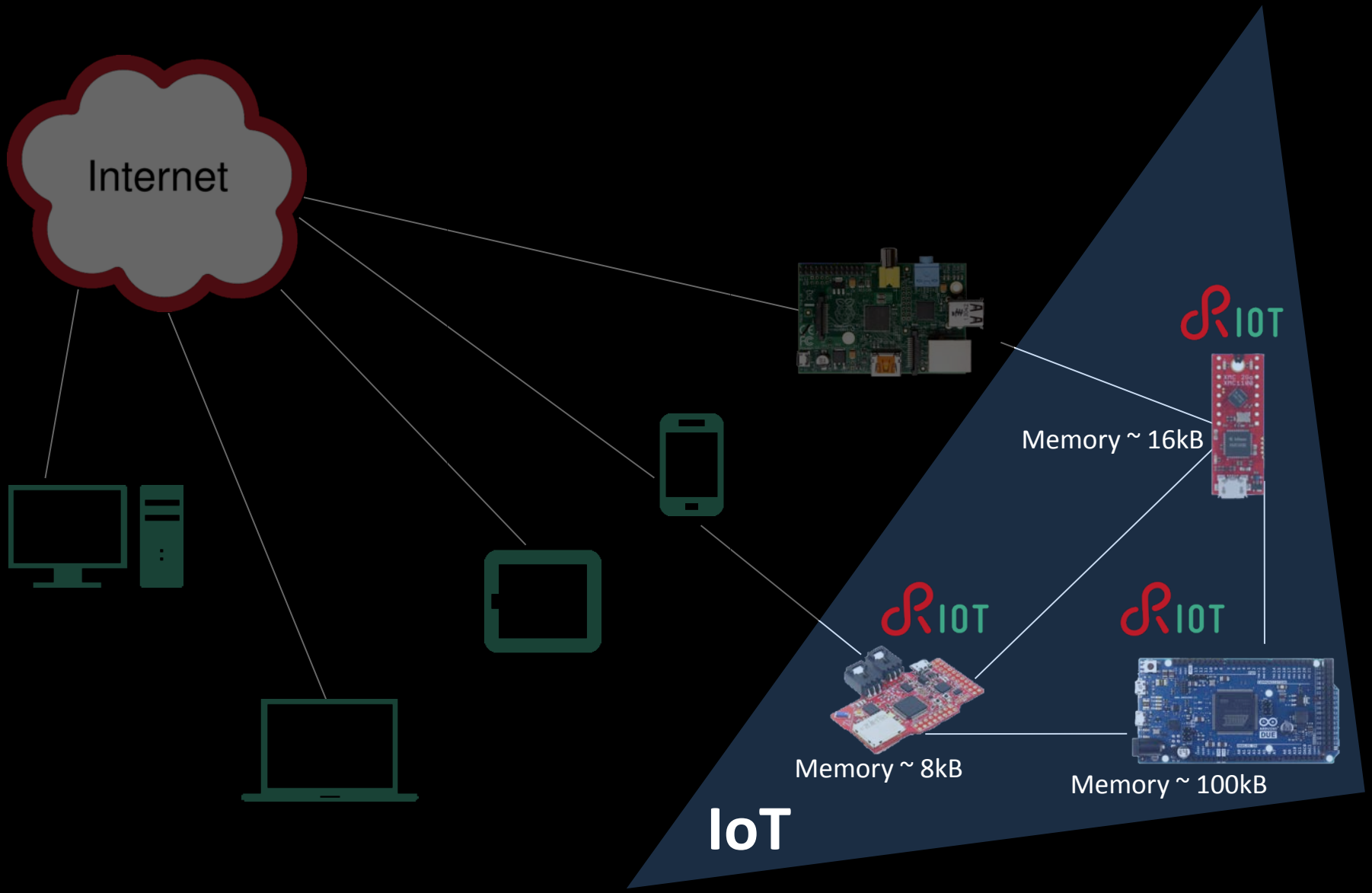
- Contiki
- RIOT
- TinyOS
- mbedOS (ARM)
- Zephyr (Intel)
- LiteOS (Huawei)
- ...
- … and closed-source alternatives

Reference:
O. Hahm et al. "*Operating Systems for Low-End Devices in the Internet of Things: A survey*," IEEE Internet of Things Journal, 2016.

Agenda

- Which IoT are we talking about?
- Why an OS for Low-End IoT devices?
- How?
- What is RIOT?
  - Solving IoT technical challenge 1: constrained devices
  - Solving IoT technical challenge 2: interoperability
  - Solving IoT technical challenge 3: trust

RIOT : an OS that fits IoT devices



RIOT : an OS that fits IoT devices

- RIOT is the combination of:

  - ❑ needed memory & energy efficiency to fit IoT devices

  - ❑ functionalities of a full-fledged operating system

    - Advanced, consistent APIs across 32-bit, 16-bit, 8-bit hardware
    - Full-featured, extensible network stacks

Agenda

- Which IoT are we talking about?
- Why an OS for Low-End IoT devices?
- How?
- What is RIOT?
  - Solving IoT technical challenge 1: constrained devices
  - Solving IoT technical challenge 2: interoperability
  - Solving IoT technical challenge 3: trust

IoT Challenge 1: Constrained Devices

ENERGY

- Milliwatt instead of Watt

CPU

- Megahertz instead of Gigahertz

Memory

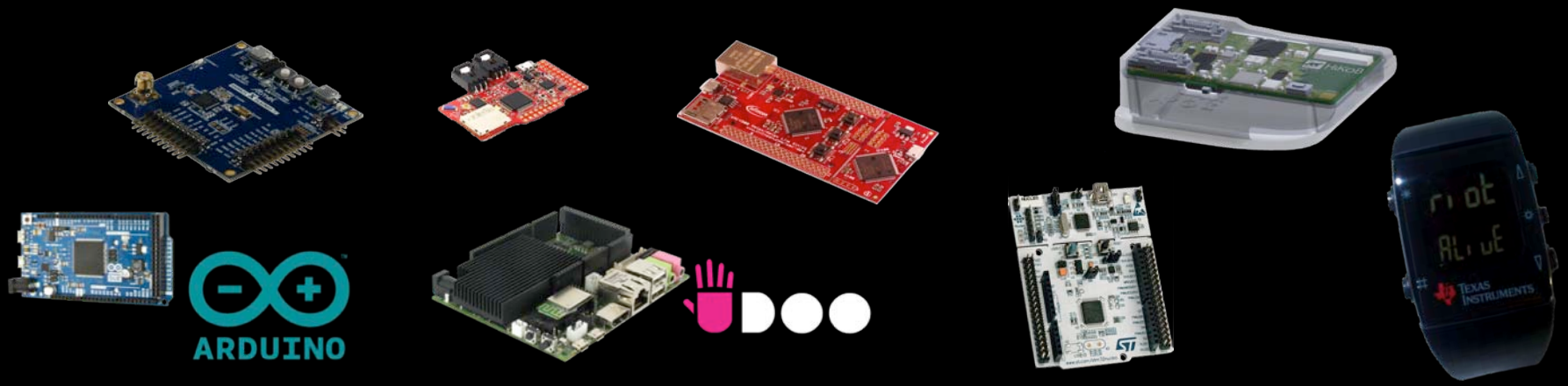
- Kilobytes instead of Gigabytes

How RIOT solves Challenge 1

- Micro-kernel architecture (contrary to Linux)
  → minimal requirements around 1kB RAM
- Tickless scheduler → energy efficiency
- Deterministic O(1) scheduler → real-time
- Low latency interrupt handler → reactivity

How RIOT solves Challenge 1

- Consistent, powerful API on 8-bit, 16-bit, 32-bit
  → preemptive multithreading, IPC…

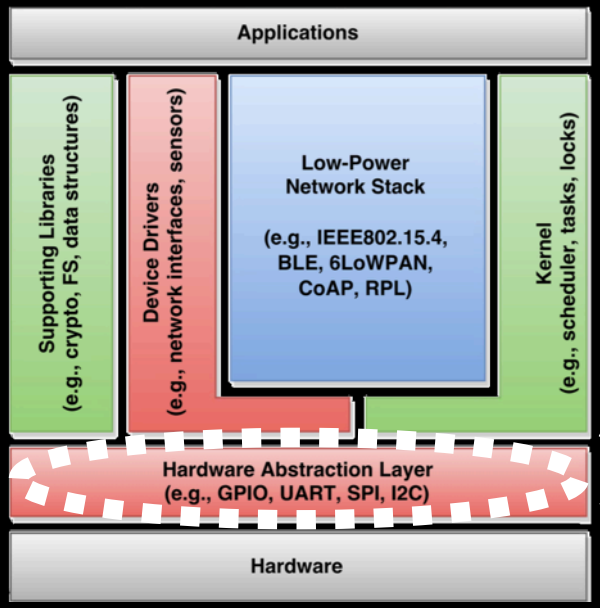


- Modular structure, adaptive to diverse hardware
  → support for 50+ different IoT boards/devices and counting

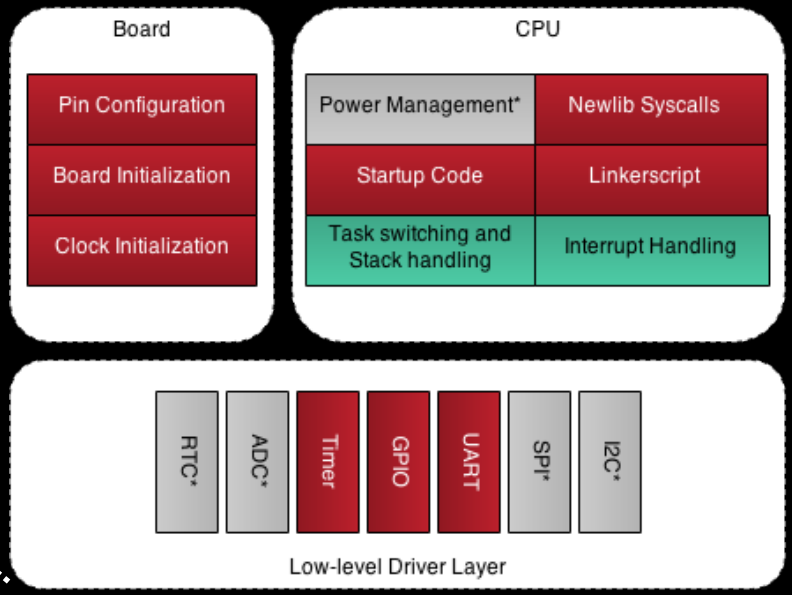
MCUs: ARM Cortex M, TI MSP430, AVR… Radios: AT86RF233…

How RIOT solves Challenge 1

- Efficient HAL: minimized hardware-dependent code



Zoom on hardware-dependent code



Red: must have

Green: must have but shared by all ports with same architecture

Grey: optional for initial porting

Task Switching, Stack Handling, Interrupt Handling: done for ARM Cortex M3, M4 and M0 is on the way

(GPIO, UART, SPI, Timers: done for STM, Atmel, NXP…)

How RIOT solves Challenge 1

RAM/ROM usage on a Cortex-M IoT device

Configuration	Hardware Specific: Platform	Hardware Specific: Drivers	Kernel	Net	Σ
ROM					
minimal	1,754	0	854	0	2,816
WSN default	4,684	6,183	2,233	4,105	37,002
gnrc_minimal	2,732	4106	2,140	12,298	27,524
gnrc	3,675	4138	2,700	30,985	74,752
RAM					
minimal	656	0	2,022	0	2,880
WSN default	681	0	2,022	2,066	6,344
gnrc_minimal	676	0	2,022	2,990	7,016
gnrc	676	0	2,022	15,815	20,828

With a simple application over a IPv6/6LoWPAN stack in RIOT, 95% of the code is hardware-independent and/or reusable.

Well-known tools are usable!

- Compliance with common system standards
  - ✓ POSIX sockets, pthreads
  - ✓ standard C, C++ application coding

→ Much shorter development life-cycles

- ✓ Run & debug as native process in Linux
- ✓ Use of well known debug tools enabled

GDB
The GNU Project Debugger

Valgrind

WIRESHARK

Third-party code is usable!

Package	Overall Diff Size	Relative Diff Size
libcoap	639 lines	6.3 %
libfixmath	34 lines	0.2 %
lwip	767 lines	1.3 %
micro-ecc	14 lines	0.8 %
relic	24 lines	<0.1 %

Packages (similar to BSD ports) for third-party open source code

- ✓ Small porting effort needed (negligible % LoC)
- ✓ Use code not initially developed for RIOT
- ✓ Use code not even initially developed for IoT!

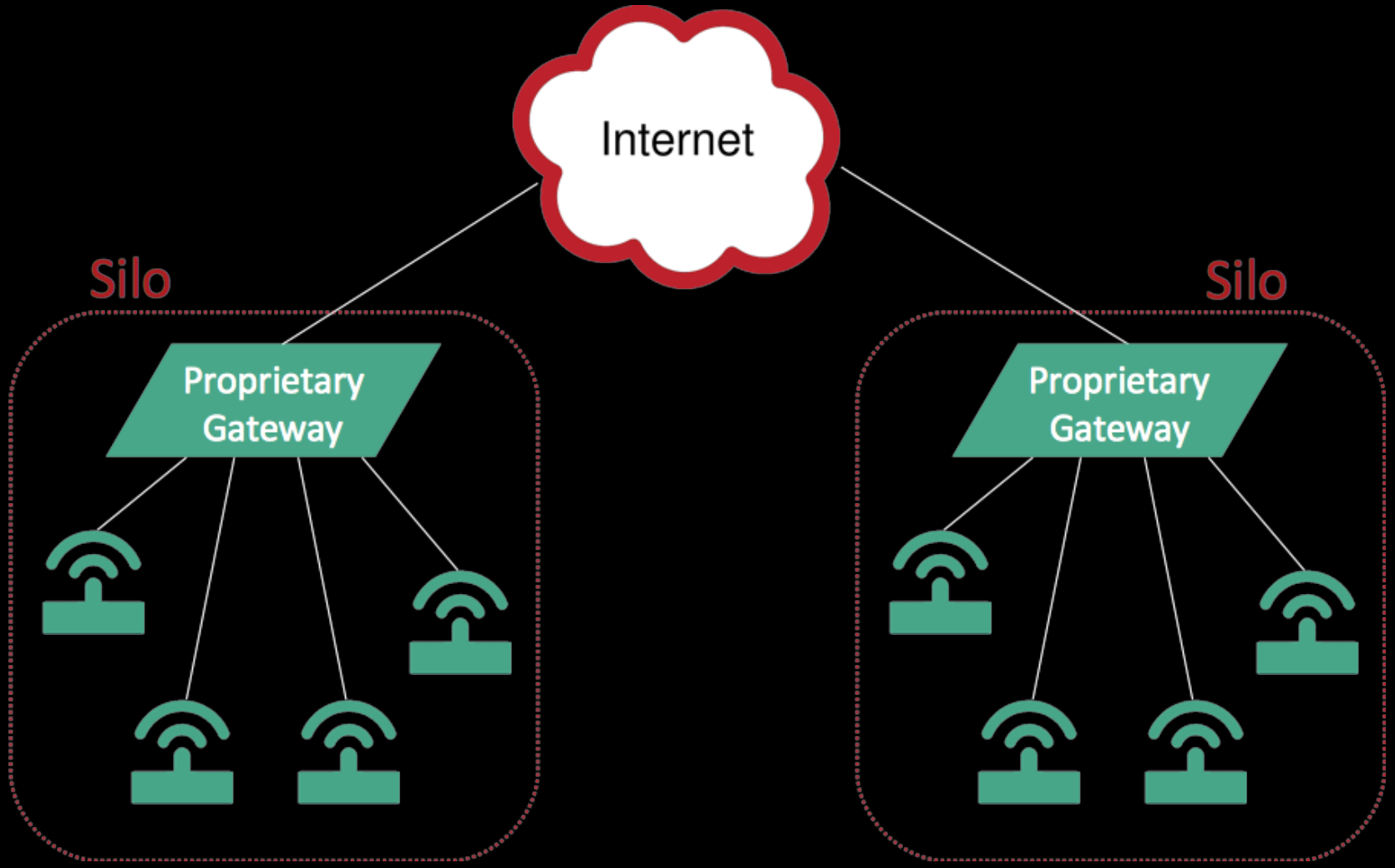
Agenda

- Which IoT are we talking about?
- Why an OS for Low-End IoT devices?
- How?
- What is RIOT?
  - Solving IoT technical challenge 1: constrained devices
  - Solving IoT technical challenge 2: interoperability
  - Solving IoT technical challenge 3: trust

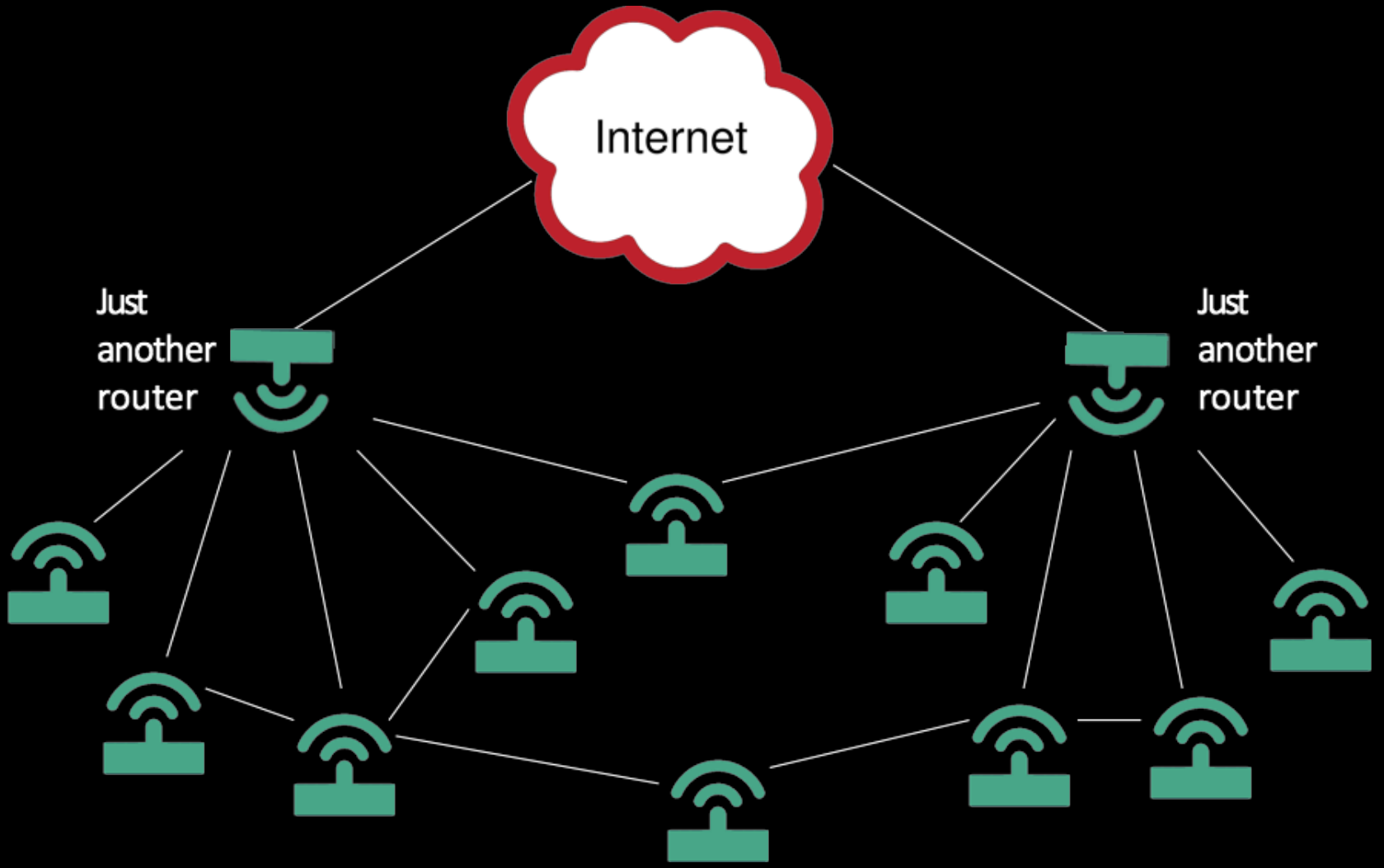
IoT Challenge 2: Interoperability

- System-level interoperability
  - Hardware-independent IoT software
  - Usability of third-party, well-known tools
- Network level interoperability
  - End-to-end connectivity per default
  - Device-to-device connectivity

IoT Interoperability Challenge: The IoT today looks mostly like this



IoT Interoperability Challenge: The IoT we want looks more like that



The IoT we want is… the Internet!

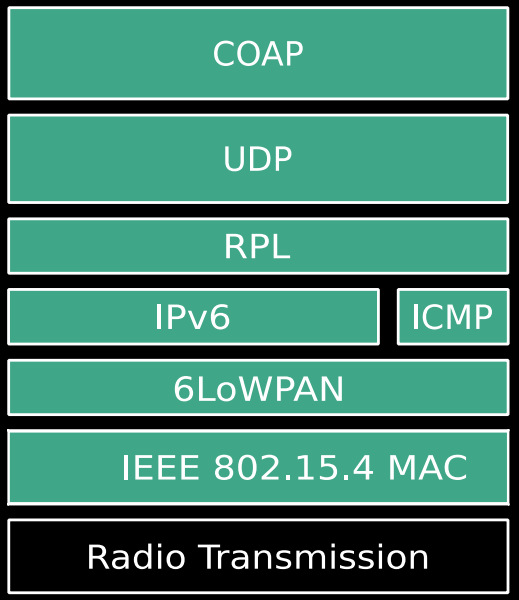
Standard IoT protocols? On the way!
Work in progress at IETF, IEEE, W3C, OMA…

New specs for link layer technologies
- Low-power radios, PLC, BACnet
- IEEE 802.15.4, Z-Wave, BLE, LoRa (and IEEE 802.11)
- More to come…

New specs for network layer protocols
- Fitting IoT requirements and interoperable with IP
- 6TiSCH, 6LoWPAN, RPL, OLSRv2, AODVv2
- More to come…

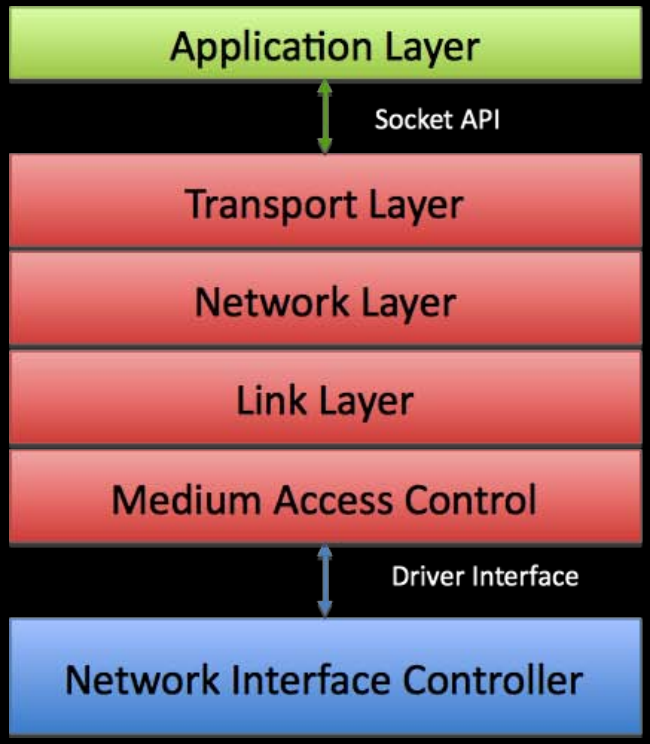
New specs for application layer protocols
- Fitting IoT requirements and interoperable with web
- CoAP, LwM2M, CBOR
- More to come…

New network paradigms
- Content-centric networking for IoT
- More to come…

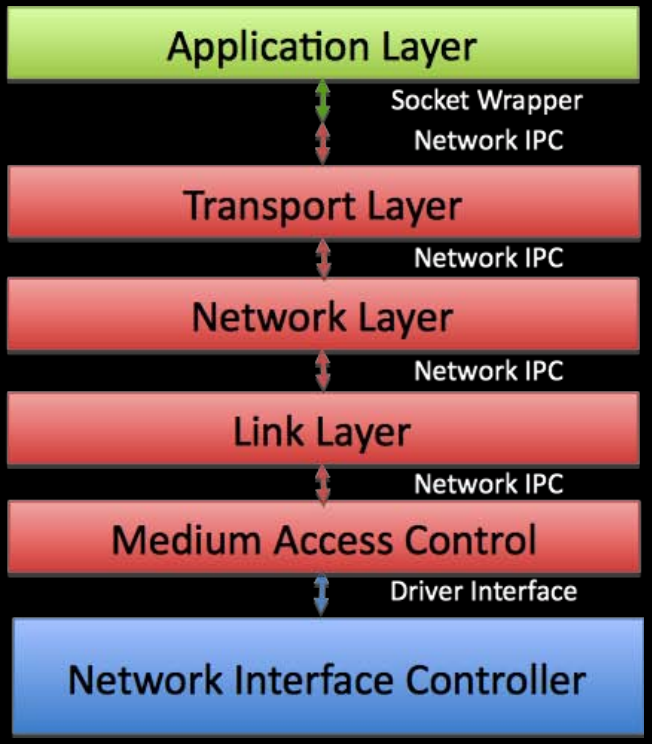


How RIOT solves Challenge 2

→ Network stack ultra-flexibility and modularity

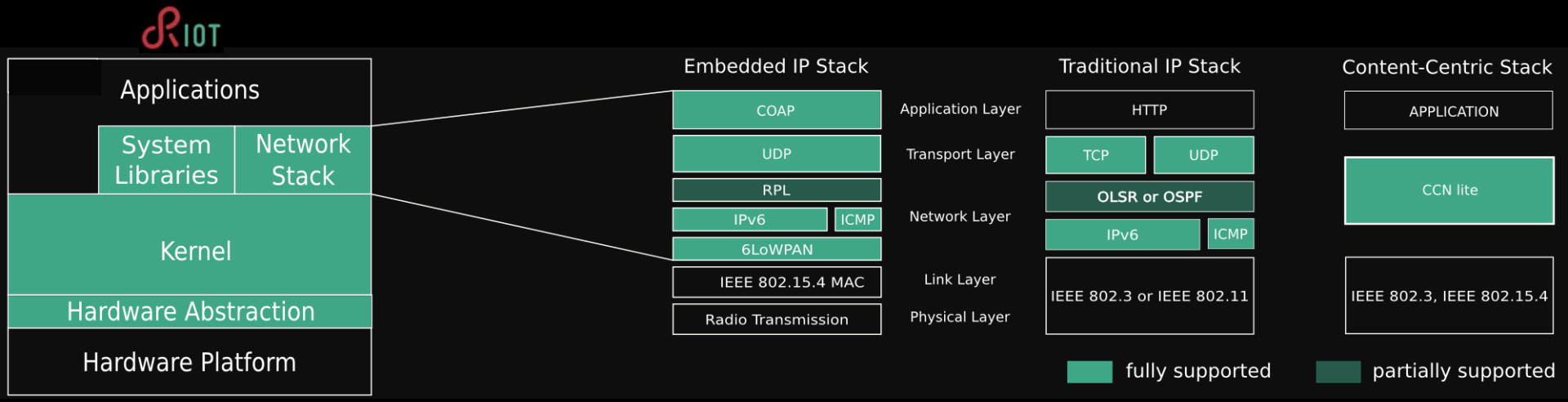


Traditional stack



GNRC stack

How RIOT solves Challenge 2

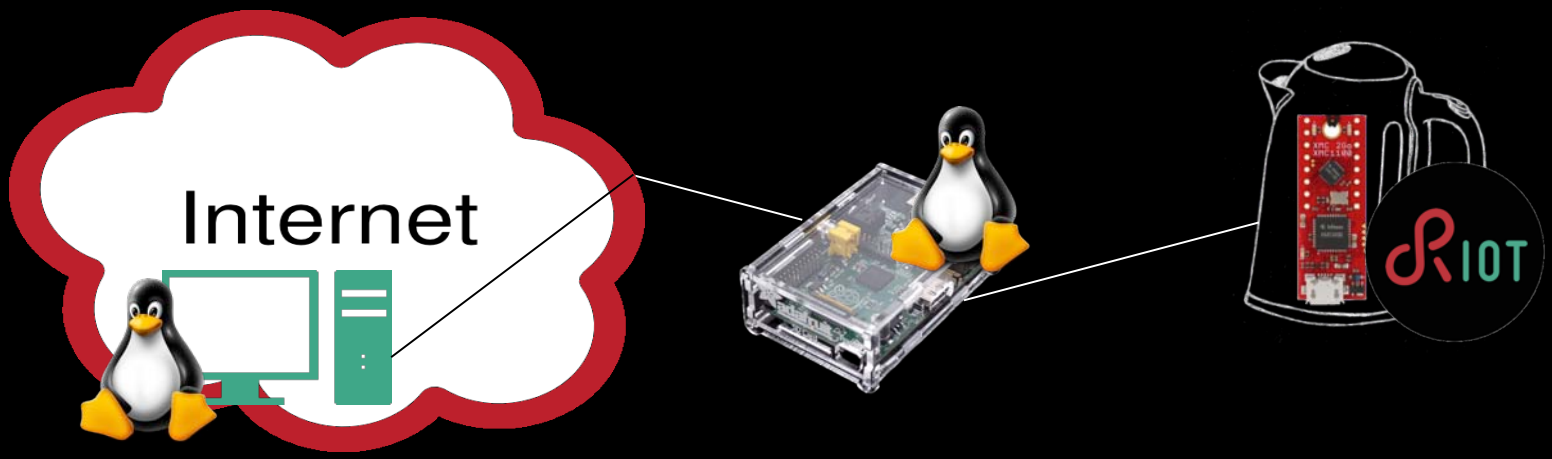


- ✓ 6LoWPAN stack, supporting IoT wireless tech.
- ✓ Standard IPv6 stack
- ✓ Packages for third-party modules/stacks:
  - OpenWSN, CCN-lite, Emb6, lwIP, tinyDTLS…

Agenda

- Which IoT are we talking about?
- Why an OS for Low-End IoT devices?
- How?
- What is RIOT?
    - Solving IoT technical challenge 1: constrained devices
    - Solving IoT technical challenge 2: interoperability
    - Solving IoT technical challenge 3: trust

Some level of trust with IoT?



Combining RIOT & Linux, IoT is possible with

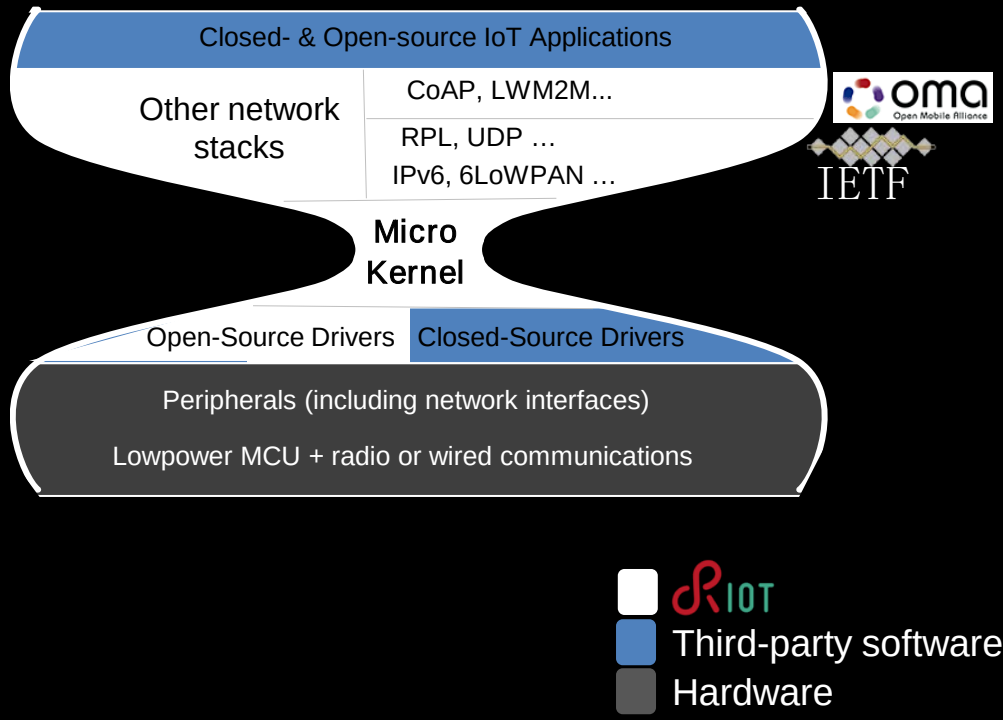
- ✓End-to-end open source
- ✓End-to-end secure & open communication standards
- ✓From anywhere in the Internet all to the way to (low-end) IoT devices

RIOT in a nutshell

Free, open-source plaftorm for portable IoT software

RIOT offers a platform functionally equivalent to Linux, based on:

- open-source,
- open-access protocol specs,
- community-driven dev.



Closing words

- IoT software platforms: a very active field
  - Current situation comparable to mobile phones transition before/after Android & iOS
- Big progress in IoT connectivity & interoperability
- The toughest challenges remain
  - Security (avoid IoT botnets armageddon?)
  - Privacy (reconciled with IoT and big data?)

Hands-on with RIOT

- ✓ Setup: get RIOT code

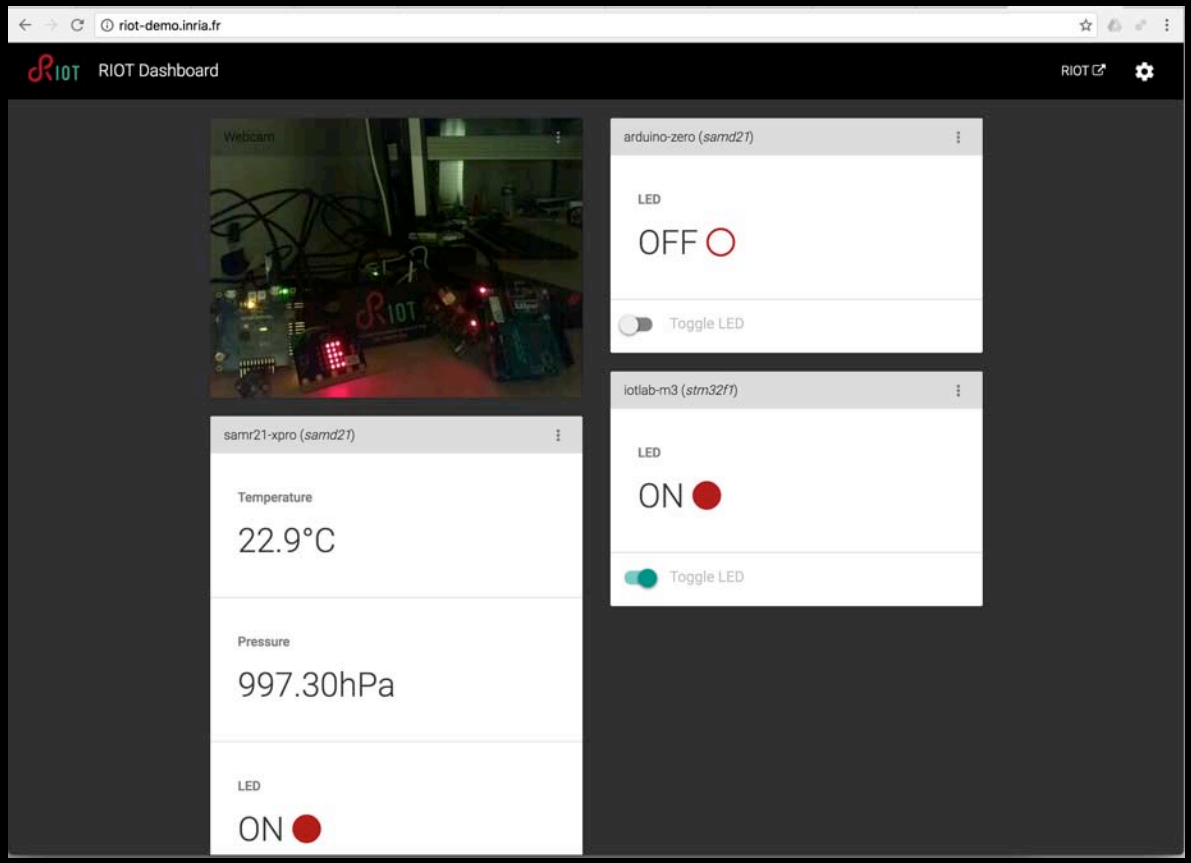
  `git clone https://github.com/RIOT-OS/RIOT.git`

  Caveat: You may have to install `git` and toolchain (things like `gcc` ...)

  See https://github.com/RIOT-OS/RIOT/wiki/Introduction

RIOT in Action

- http://riot-demo.inria.fr/
- RIOT
- Diverse hardware
- Web Server
- COAP/IPv6 6LoWPAN

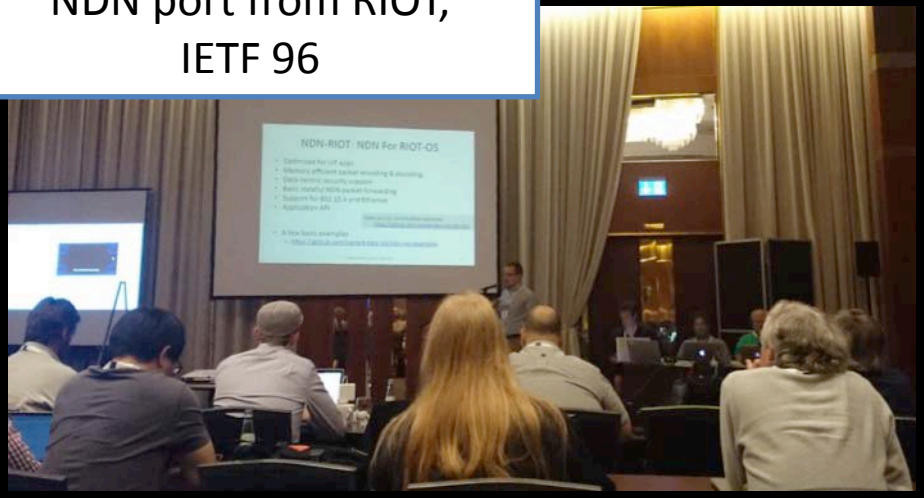


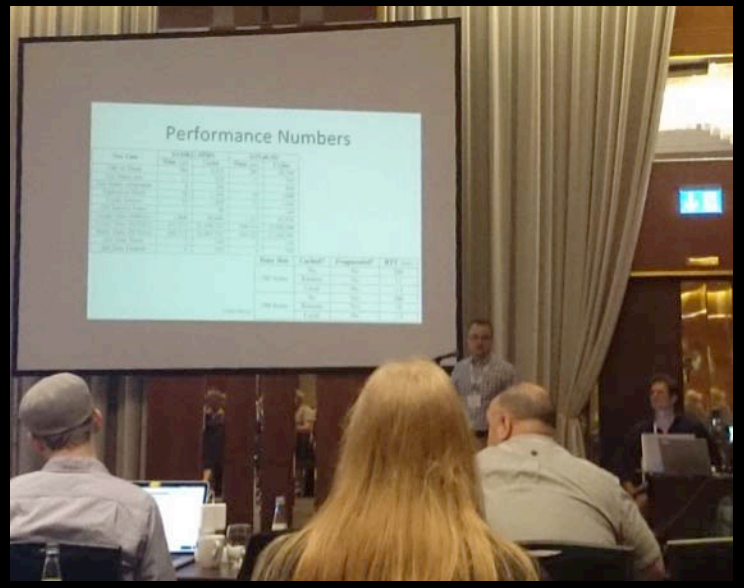
RIOT at IETF

RIOT Hackaton @IETF

6lobac report with RIOT, IETF 96

NDN port from RIOT, IETF 96





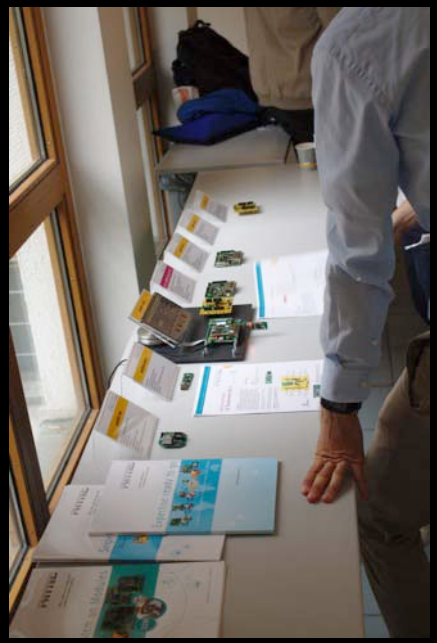
RIOT Summit

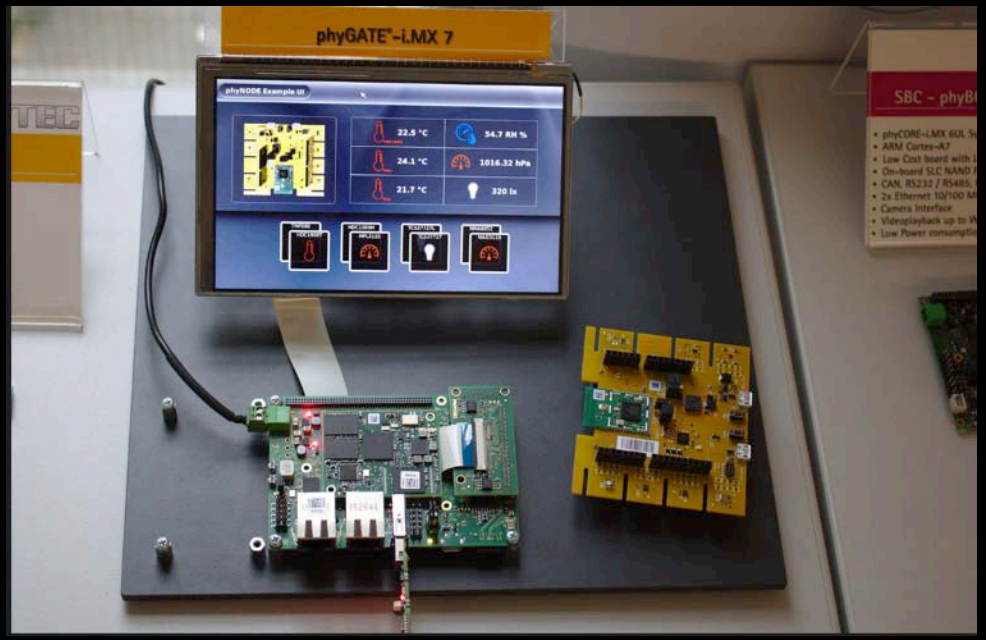
http://summit.riot-os.org/

- Ralph Droms (Cisco)
  - RIOT ICN
- Carsten Borman (TZI)
  - IETF Protocols for IoT
- Alexander Pelov (ackl.io)
  - LPWAN
- Nordic / Samsung
- Zolertia/Phytec/ImgTec/…
- […]

http://tinyurl.com/riot-summit-photo

http://tinyurl.com/riot-summit-photo

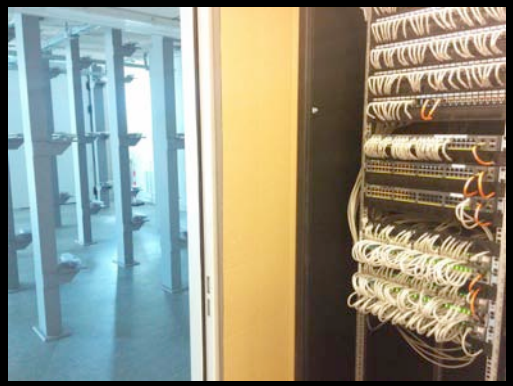


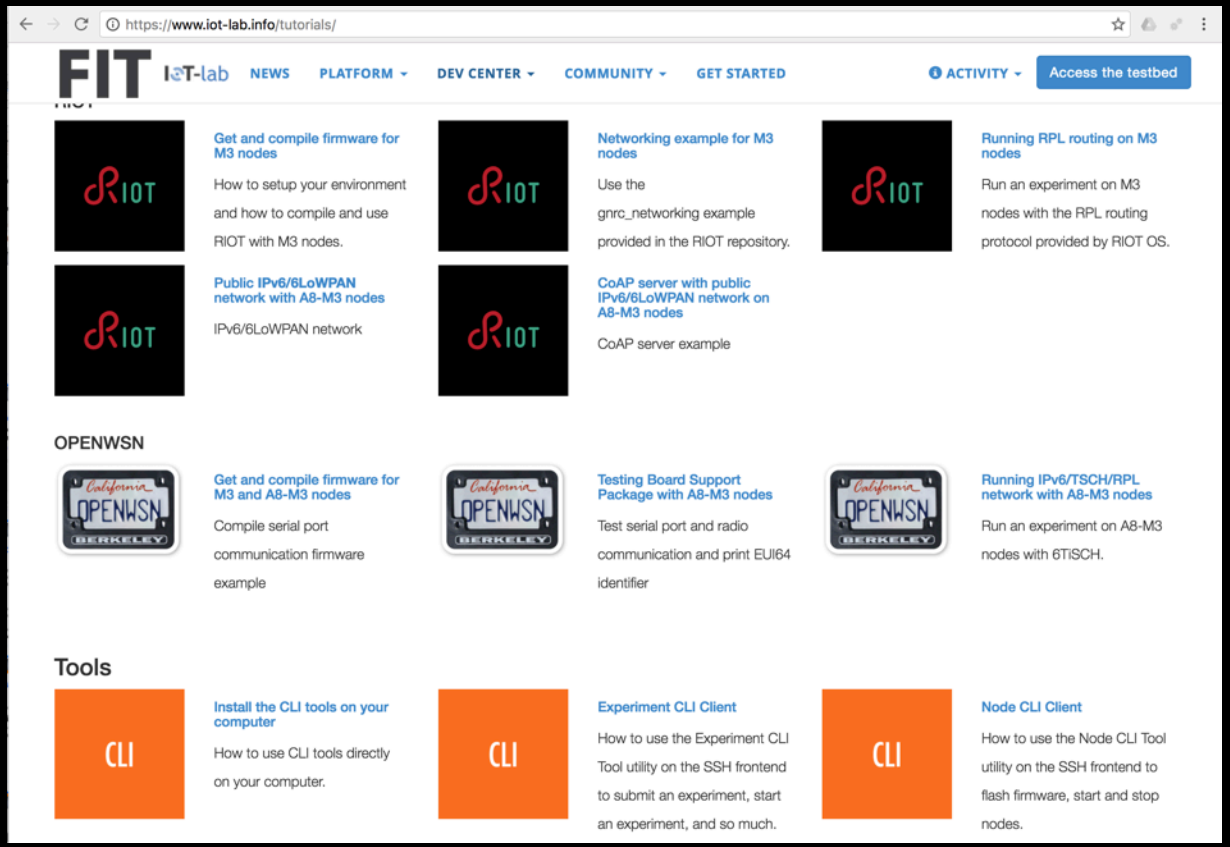


http://summit.riot-os.org/

RIOT on FIT IoT-LAB
https://www.iot-lab.info/

FIT IoT-lab





Thanks for your interest!

News: https://twitter.com/RIOT_OS

For cooperation questions: riot@riot-os.org

For developer questions: devel@riot-os.org

Support & discussions on IRC: irc.freenode.org #riot-os

RIOT Roots & Evolution

- 2008 – 2012
  Ancestors of RIOT kernel developed in research projects (FireKernel, uKleos).

- 2013 – 2016
  Branding of RIOT started, source code moved to Github, major development of the network stack & the OS as such.

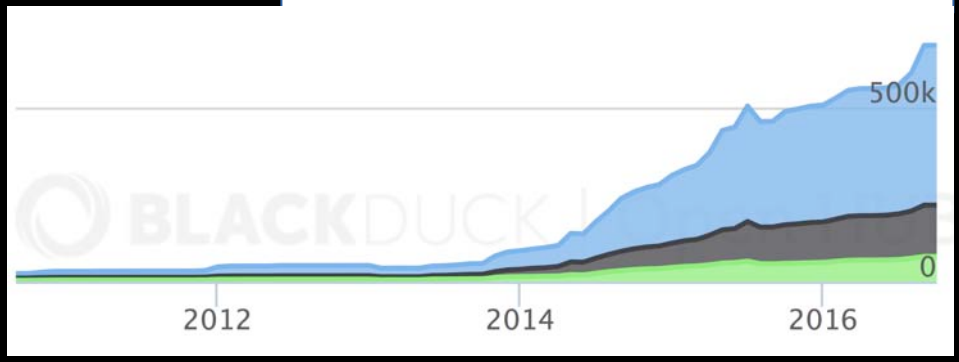
- Speed-evolution
  of the code-base.
  110+ contributors worldwide (makers, academic industrial)

Founding institutions

Freie Universität Berlin

Inria
INVENTORS FOR THE DIGITAL WORLD

Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences



Some supporters/users

… and dozens of independent developers around the world!

Hands-on with RIOT

- ✓ Setup: get RIOT code

  `git clone https://github.com/RIOT-OS/RIOT.git`

  Caveat: You may have to install `git` and toolchain (things like `gcc` ...)

  See https://github.com/RIOT-OS/RIOT/wiki/Introduction

- ✓ Hands-on: compile, flash, run RIOT

  Demo: RIOT shell on SAMR21 board (32bit ARM M0)

- ✓ Hands-on: communication with RIOT (IPv6, 6LoWPAN)

  Demo: PING with 2 Atmel SAMR21 boards