www.riot-os.org
Emmanuel Baccelli
Inria
on behalf of the RIOT Community
RIOT Agenda

• Why?
• How?
• What is RIOT?
  • Solving IoT technical challenge 1: constrained devices
  • Solving IoT technical challenge 2: interoperability
• Current stand, in a nutshell
• Why?

• How?

• What is RIOT?
  • Solving IoT technical challenge 1: constrained devices
  • Solving IoT technical challenge 2: interoperability

• Current stand, in a nutshell
Software Platforms Are Crucial

• Recent calls for OS and data "sovereignty"

Snowden political scandal

→ strategic & privacy-related incentives

Android generates billions of dollars

→ more obvious business incentives
The Internet of Things (IoT) is future Internet’s extremity

Advantage: IoT brings large scale business

Drawback: extreme challenge for privacy

Software platforms are even more crucial in IoT
Software Platforms for IoT?

• Great expectations for IoT, but...
  ...no standard platform yet, to program most IoT devices!

- e.g. like Arduino (but not like Raspberry Pi)
- Linux
- Android
- Moore’s law
Software Platforms for IoT?

Qualitative requirements for a software platform for IoT:

✓ long-term IoT software robustness & security
✓ trust, transparency & protection of IoT users’ privacy
✓ faster innovation by spreading IoT software dev. costs
✓ less garbage with less IoT device lock-down
RIOT Agenda

• Why?

• How?

• What is RIOT?
  • Solving IoT technical challenge 1: constrained devices
  • Solving IoT technical challenge 2: interoperability

• Current stand, in a nutshell
How can we achieve our goals?

- Experience (e.g. with Linux) shows we are likely to succeed with a platform that is:
  - open source
  - free
  - driven by a grassroots community
Community gathering contributors worldwide
  – People from industry, academia, hobbyists/makers
  – Community self-organizes, follows open processes

Large-scale distributed source code management

Geopolitically neutral
RIOT Principles

• Code of the platform is free & open source
  – core distributed with non-viral copyleft license

(favors less forks = more coherent code + community)

→ Indirect business models
  (like business with Linux)
• Why?
• How?

• What is RIOT?
  • Solving IoT technical challenge 1: constrained devices
  • Solving IoT technical challenge 2: interoperability

• Current stand, in a nutshell
RIOT: an OS that fits IoT devices

Internet

Memory ~ 16kB

Memory ~ 8kB

Memory ~ 100kB
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
    - full-featured, extensible network stacks
    - end-to-end IPv6 (e.g. from IoT device to the cloud)
    - advanced, spontaneous wireless networking
    - consistent API across 32-bit, 16-bit, 8-bit hardware
• Why?
• How?
• What is RIOT?

- Solving IoT technical challenge 1: constrained devices
- Solving IoT technical challenge 2: interoperability

• Current stand, in a nutshell
IoT Challenge 1: Constrained Devices

ENERGY
- Milli\text{watt} instead of Watt

CPU
- Mega\text{hertz} instead of Gigahertz

Memory
- Kilo\text{bytes} 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

• Same 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
How RIOT solves Challenge 1

- Efficient HAL: minimized hardware-dependent code
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
RIOT Agenda

- Why?
- How?
- What is RIOT?
  - Solving IoT technical challenge 1: constrained devices
  - Solving IoT technical challenge 2: interoperability
- Current stand, in a nutshell
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!
Internet Interoperability: Based on Open Standards

- Application: XDI, CBOR, RDF, XHTML, JSON, Telnet, CoAP, HTTP, XMPP
- Transport: TCP, UDP, TLS/SSL
- Network: OSPF, RPL, DHCP, BGP, IPv6, IPv4, OLSR, SLAAC
- Link: IEEE802.15.4, LoRa, Ethernet, BLE
Internet Interoperability: Accelerated with Open Source
Usual solutions for Interoperability: Challenged by IoT...

... because of resource constraints on IoT devices
  - Memory, CPU, energy

... because of low-power communication characteristics
  - Lossy / duty cycles
  - Super-small frames
  - Spontaneous wireless architecture

→ Adapted standard IoT protocols needed
Standard IoT protocols? On the way!
Work in progress at IETF, IEEE, W3C, OMA

New specs for radio technologies and link layers
- Low-power
- 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

- 6LoWPAN stack, supporting IoT wireless tech.
- Standard IPv6 stack
- BSD-like ports for third-party modules/stacks:
  - OpenWSN, CCN-lite, Emb6, lwIP, tinyDTLS...
How RIOT solves Challenge 2

→ Network stack ultra-flexibility and modularity

Traditional stack

RIOT stack (GNRC)
RIOT Agenda

• Why?
• How?
• What is RIOT?
  • Solving IoT technical challenge 1: constrained devices
  • Solving IoT technical challenge 2: interoperability

• Current stand, in a nutshell
RIOT in a nutshell

Free, open-source platform for portable IoT software

RIOT offers a platform functionally equivalent to Linux, based on:
- open-source,
- open-access protocol specs,
- community-driven dev.
RIOT can do more, so RIOT can do less

- Arduino scripts can run as-is on top of RIOT
- mbed applications could run on top of RIOT
- Contiki can run in a single RIOT thread
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**
Some supporters/users

... and dozens of independent developers around the world!
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
Some technical next steps for RIOT

• System
  – Generic sensor/actuator interface (SAUL)
  – Over the air (OTA) binary updates
  – Certification
  – Generic optimized LPM management
  – Distributed CI system

• Network stack
  – MAC : more link layer technologies support
  – More lightweight network security protocols
  – More application layer protocols
  – More integration or new ports of other stacks
Other IoT Software platforms?
No great fit yet.

- Some “cloud” solution? OK, but not sufficient.
- Arduino? Hardware specific, not an OS.
- mbed? Hardware- and ARM-centric, server-centric.
- Android? Big memory needs, Google-centric.
- Contiki? Fits memory, but old & exotic API,
- Zephyr? No community, Intel-product for now...

(Whatever on RasberryPi? Target is much smaller.)