# Process isolation for real time IoT devices — an overview

Fabian Kovacs

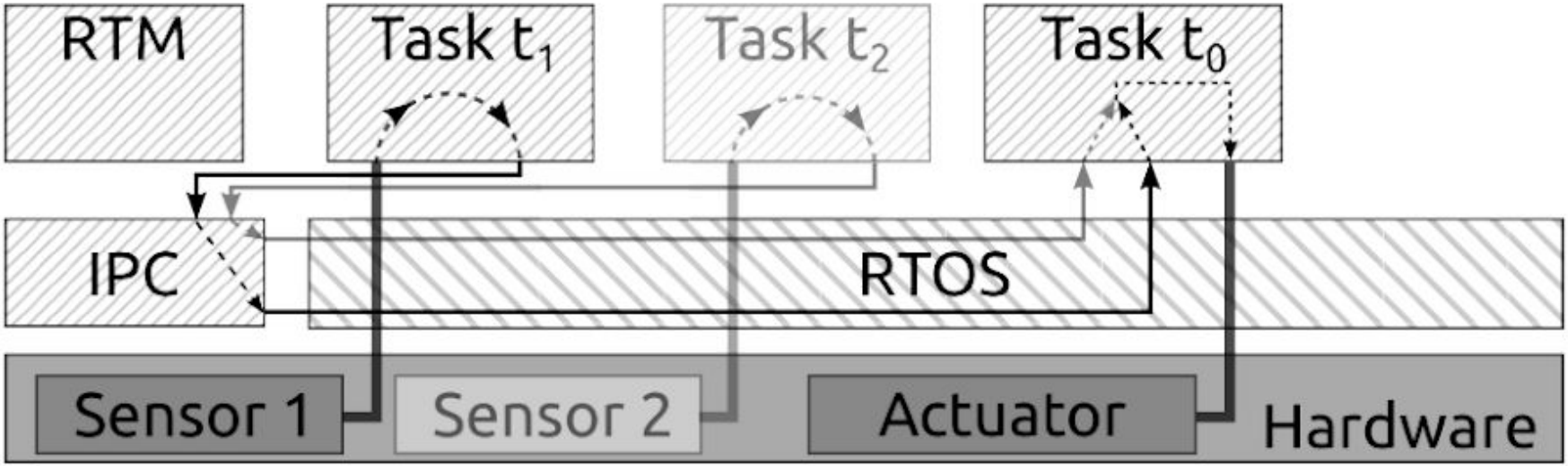
1. introduce and motivate your topic (1-2 slides)
2. mention some related work / references you looked at, but will *not* cover in your survey  (1 slide). The goal here is to help position and define your focus.
3. provide a tentative structure/skeleton for your report (1-2 slides)
4. provide a sample list of references you plan to cover in your survey and 1-2 references that you will study in more depth (1-2 slides).
    a. Aim to briefly present orally (e.g. 1-2 sentence per item) the main references in your list
5. - break down your tentative schedule towards final report submission (1 slide)
6. - list open questions you may have (1 slide)

# Motivation

# Process Isolation —  with realtime requirements

1. Multiple Interested Parties on same device of potentially conflicting interest and origin
   a. Eg.: Device with multiple sensors by different vendors, where data needs to be combined
2. Minimize the potential harm of a misbehaving actor
   a. Protect memory domains from untrusted access
   b. Secure communication between processes
   c. Processes may not starve resources
   d. Minimize reliance on Trusted Computing Base

# Examplary Architecture from TyTAN



# Real time IoT devices

1. Processes are working with deadlines
    a. Hard tasks must meet their deadlines — Soft tasks try minimize average response time
    b. Missing hard deadlines can be fatal
2. Realtime implies low computational overhead and guaranteed execution
3. Solutions for process isolation exist in commodity hardware (virtualization / paging / rings)
    a. Expensive hardware and high overhead
4. Real time requirements dictate low computational overhead
    a. And fault tolerance in case of uncooperative software
5. IoT devices need to be cheap and have modest power consumption
6. Bespoke Hardware solutions are feasible at IoT scale

# Related Work — What's not covered.

- Brief discussion of Attestation/Measuring
- Brief intro to Realtime
- Memory encryption will not be discussed
- Side Channel attacks will not be discussed

# Structure

1. Motivation for Process Isolation & Protected Module Architectures
    a. Define / Differentiate Process Isolation
    b. Define / Differentiate Real Time
    c. Threat Model(s)
    d. Challenges  from real time requirements
    e. Secure Interprocess Communication
2. Overview / Structure
3. Process Isolation
    a. Limitations for IoT devices / Real Time applications
    b. Secure IPC (A little bit on Attestation)
    c. Software Solutions — Overview and Challenges
    d. **Hardware (assisted) Solutions** — Overview and Challenges

# Structure — cont’d

1. Real time solutions
    a. Challenges from Multiple Actors
    b. Scheduling and Interrupt handling at real time
    c. Process Isolation with real time guarantees

# References


- Maene et al. (2018). **Hardware-Based Trusted Computing Architectures for Isolation and Attestation**. IEEE Transactions on Computers, 67(3), 361–374. https://doi.org/10.1109/TC.2017.2647955
  - Very recent overview of relevant technologies.
- Noorman et al. (2017). **Sancus 2.0**. ACM Transactions on Privacy and Security, 20(3), 1–33. https://doi.org/10.1145/3079763
  - Recent and seemingly popular implementation of OS/Hardware solution
- Brasser et al. (2015). **TyTAN**. 1–6. https://doi.org/10.1145/2744769.2744922
  - Implementation and Discussion of details for Process Isolation mechanisms
- Van Bulck et al. (2016). **Towards availability and real-time guarantees for protected module architectures**. 146–151. https://doi.org/10.1145/2892664.2892693
  - Side-paper for Sancus and in-depth discussion for PMA which seems promising

# Schedule

- Weeks 1-3: Writeup and discussion of Process Isolation Technologies
  - Selection of candidates for deeper study
- Week 2-3: Comparison of Technologies and evaluation for Realtime applications
- Week 3-4: Deep dive into selected candidates

# Open Questions

1. Are there feasible software side solutions?
2. Are hardware solutions feasible wrt cost and power consumption?
3. Have these solutions been implemented and evaluated in the wild? (Most seem to be emulated so far)