FOSDEM 2023

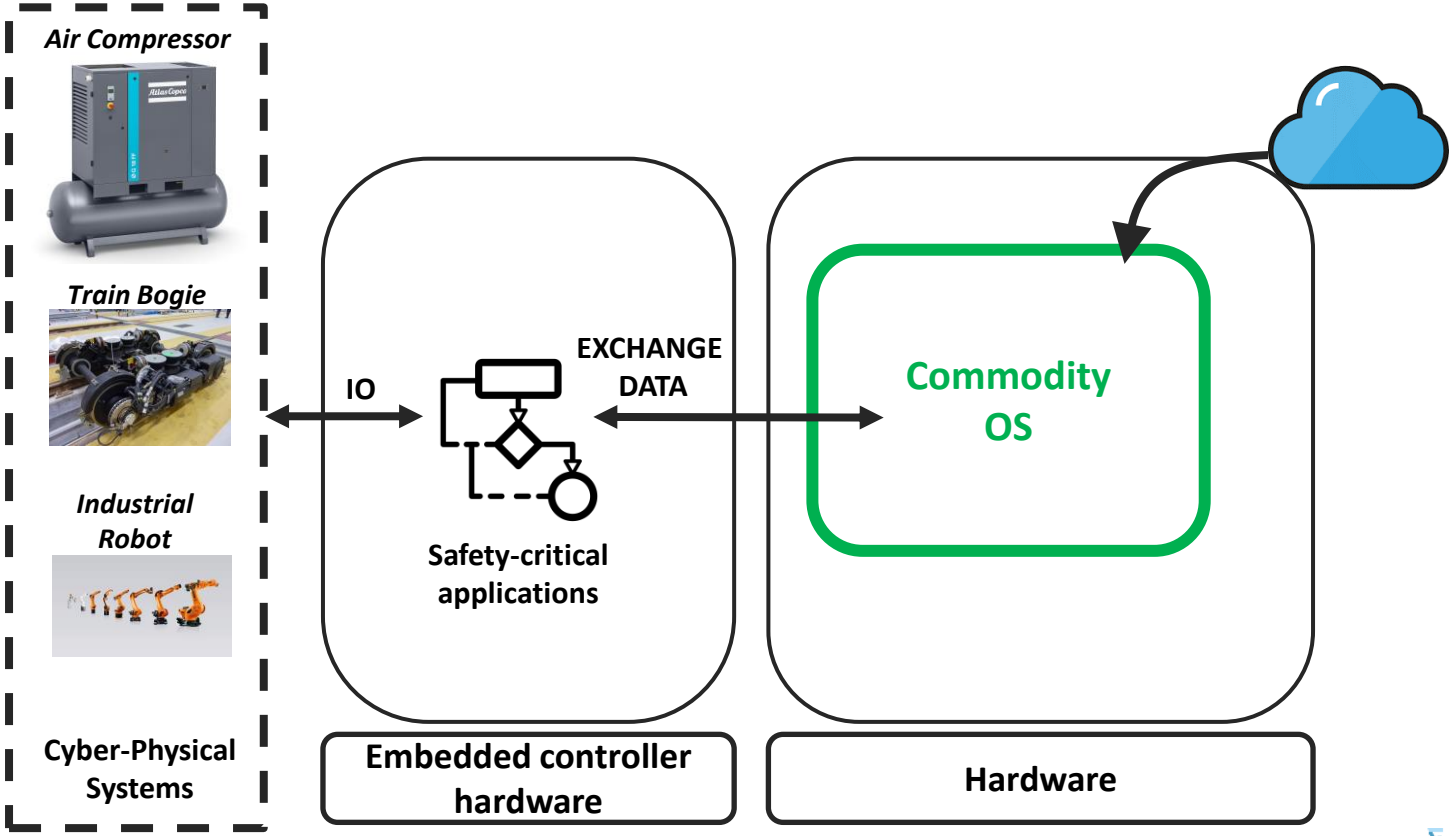
Keeping safety-critical programs alive when Linux isn’t able to

Tom Van Eyck – PhD student @ DistriNet

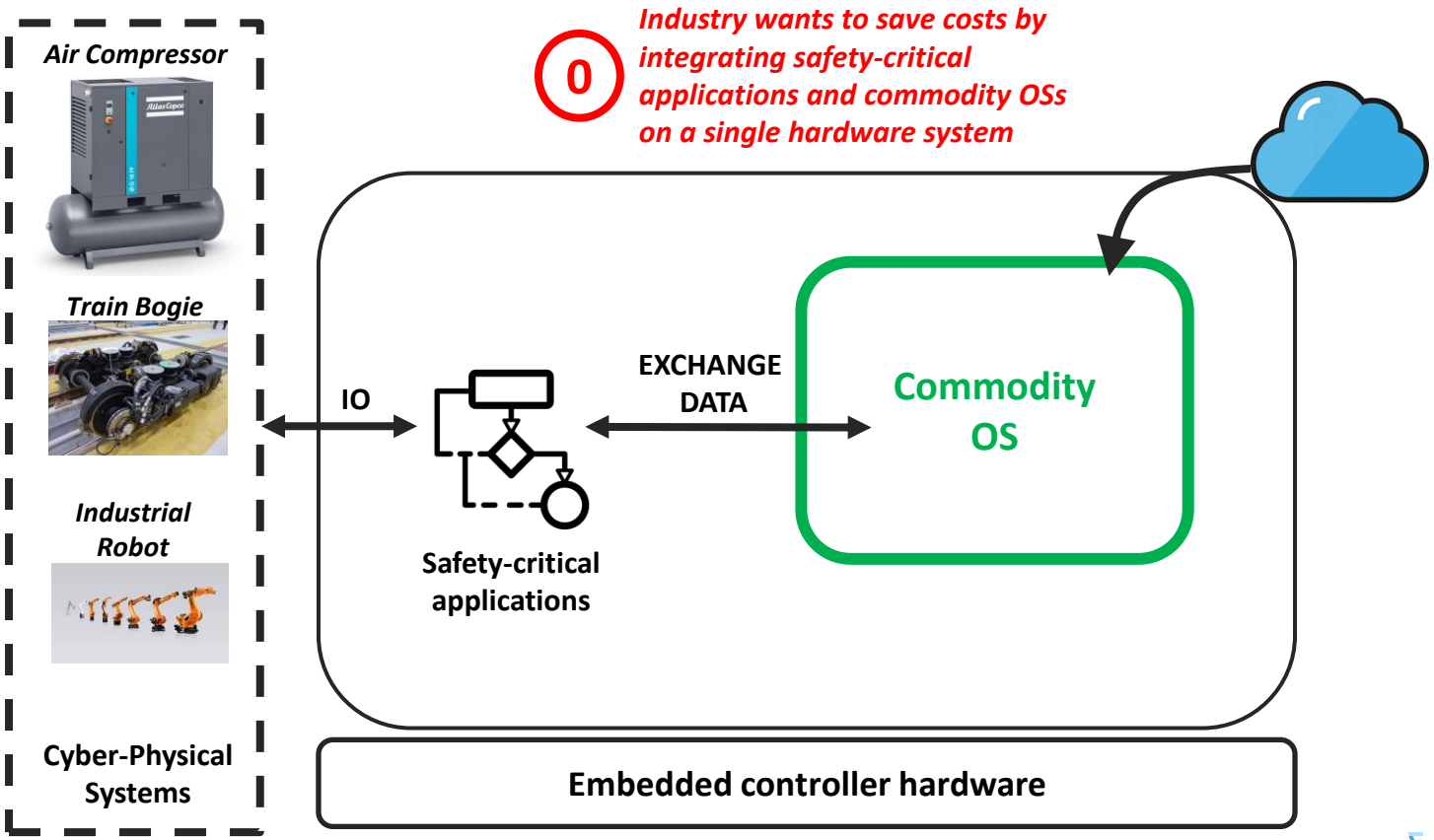
Co-authors: Hamdi Trimech, Majid Salehi,
Thanh-Liem Ta, Sam Michiels,
Danny Hughes, Hassaan Janjua

KU LEUVEN DistriNet

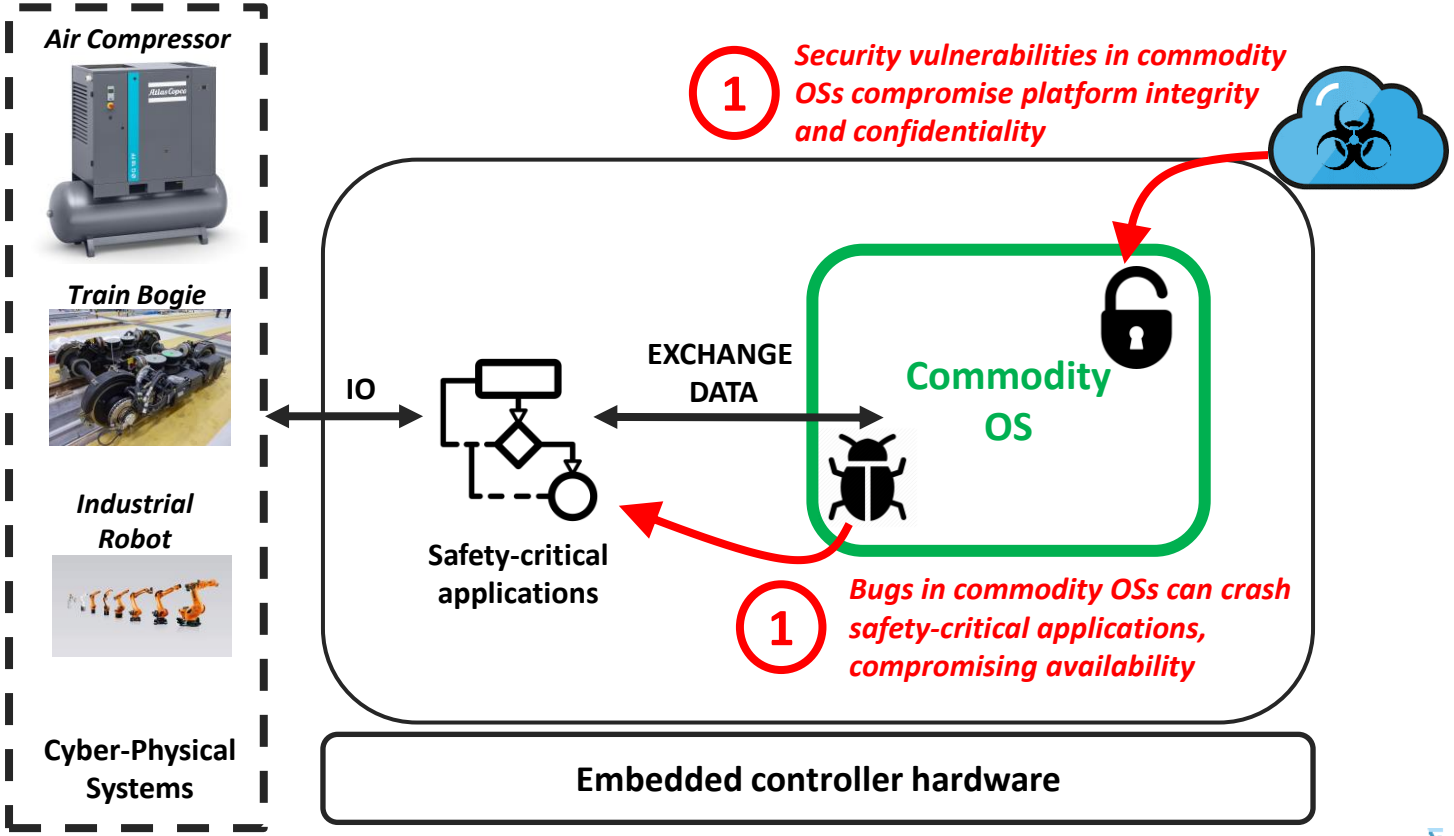
Cyber-Physical Systems



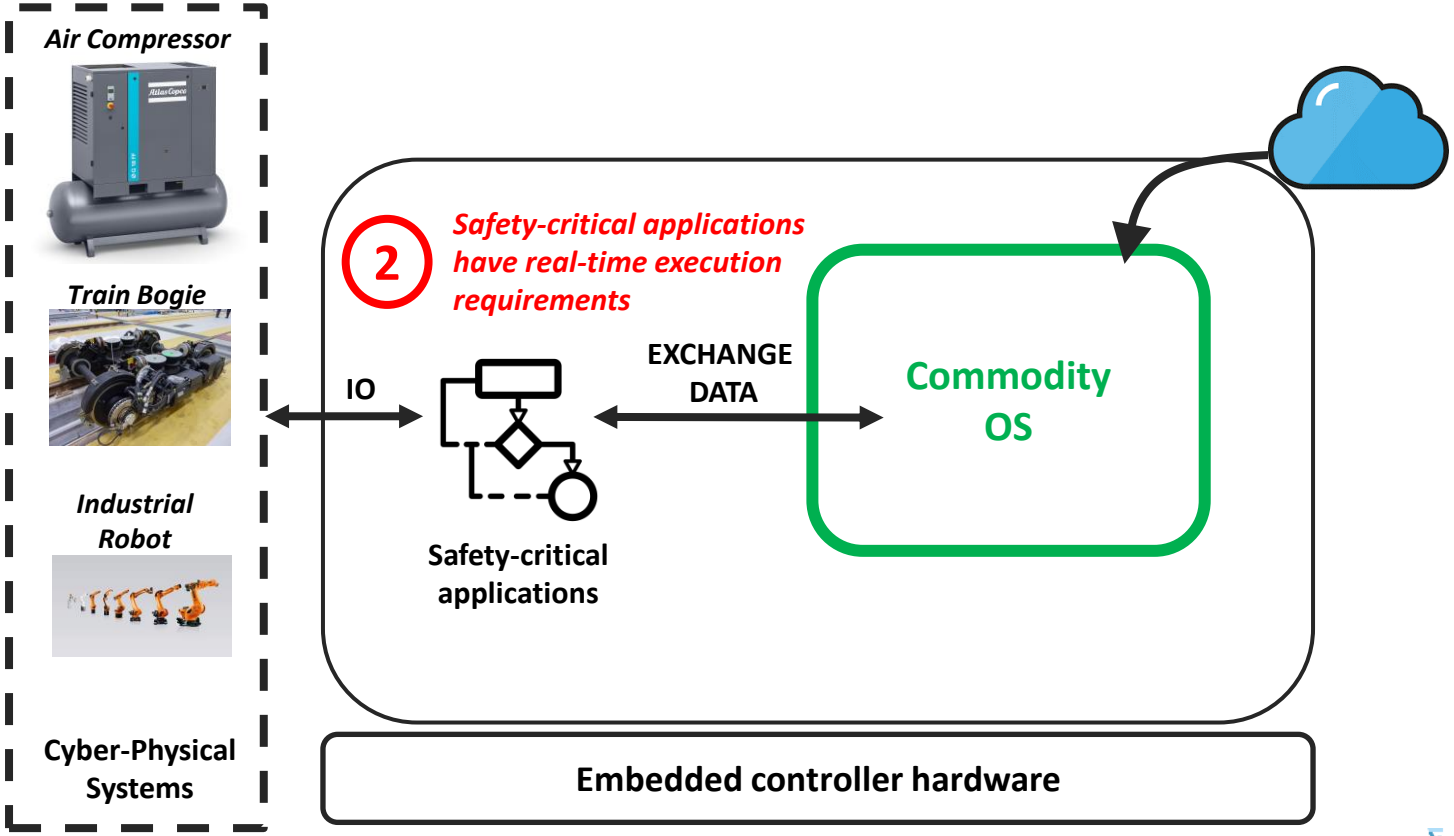
The problem


Air Compressor
Train Bogie
Industrial Robot
Cyber-Physical Systems
0
Industry wants to save costs by integrating safety-critical applications and commodity OSs on a single hardware system
IO
EXCHANGE DATA
Commodity OS
Safety-critical applications
Embedded controller hardware

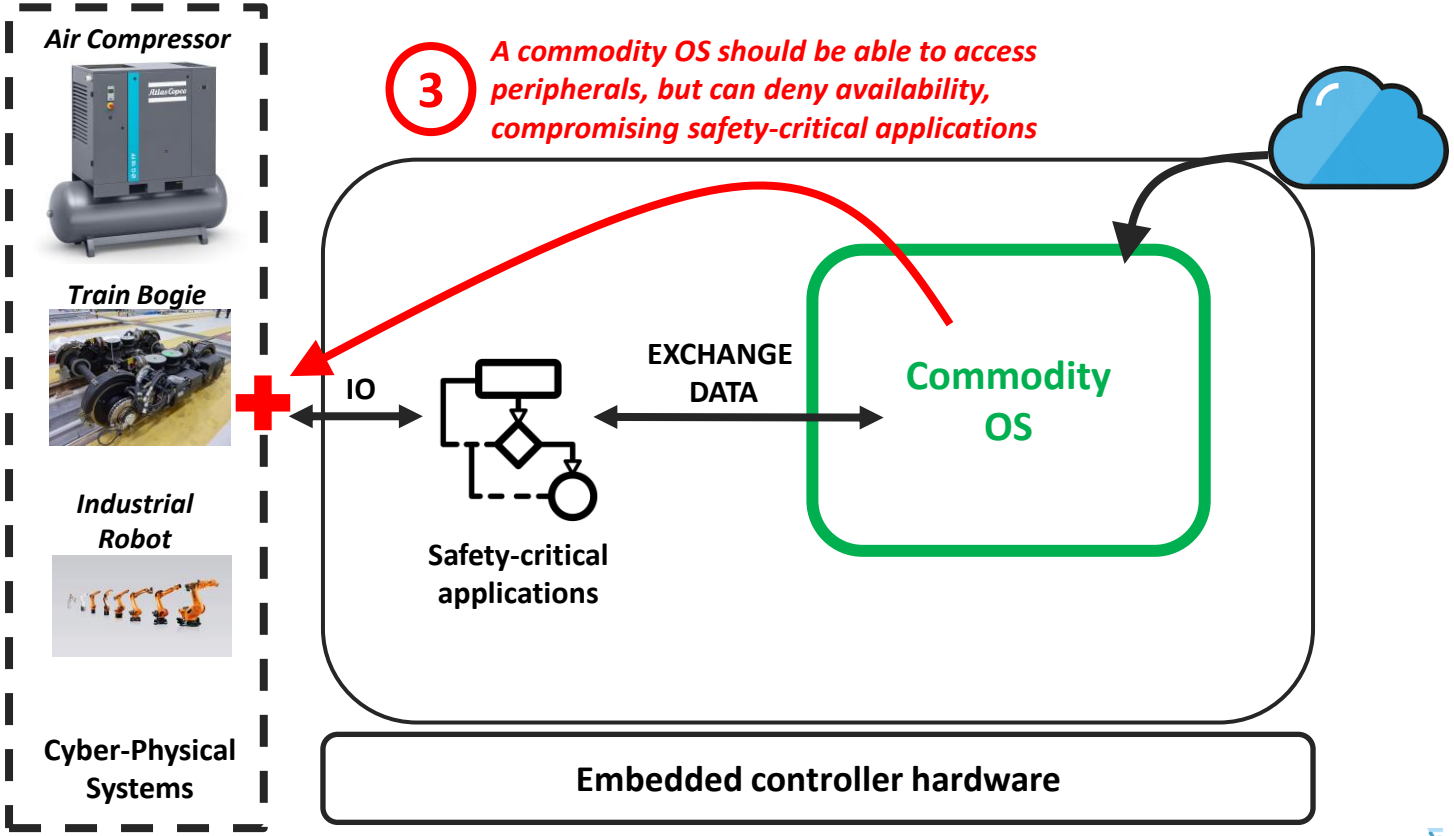
The problem – Isolation



The problem – Real-time execution



The problem – Peripheral sharing



The problem - Requirements

Can we ensure availability for safety-critical applications while running a commodity OS on the same system with little developer impact?

1. Isolation of critical applications
2. Real-time execution of critical applications
3. Transparent sharing of peripherals

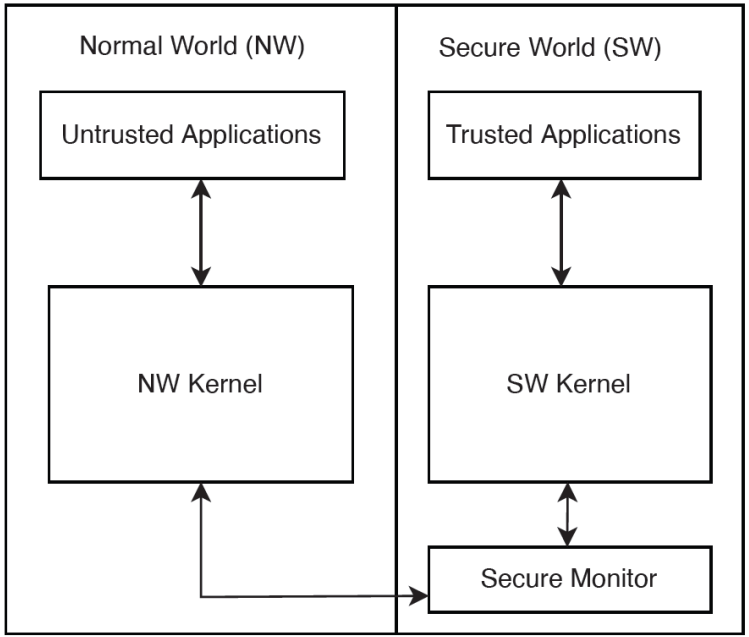
Threat model

- Strong remote adversary with root privileges
- Denial of Service for safety-critical applications
- Assumptions
  - **Trusted**
    Hardware
    Critical applications
    Peripherals
  - **Not trusted**
    Everything else

Arm TrustZone

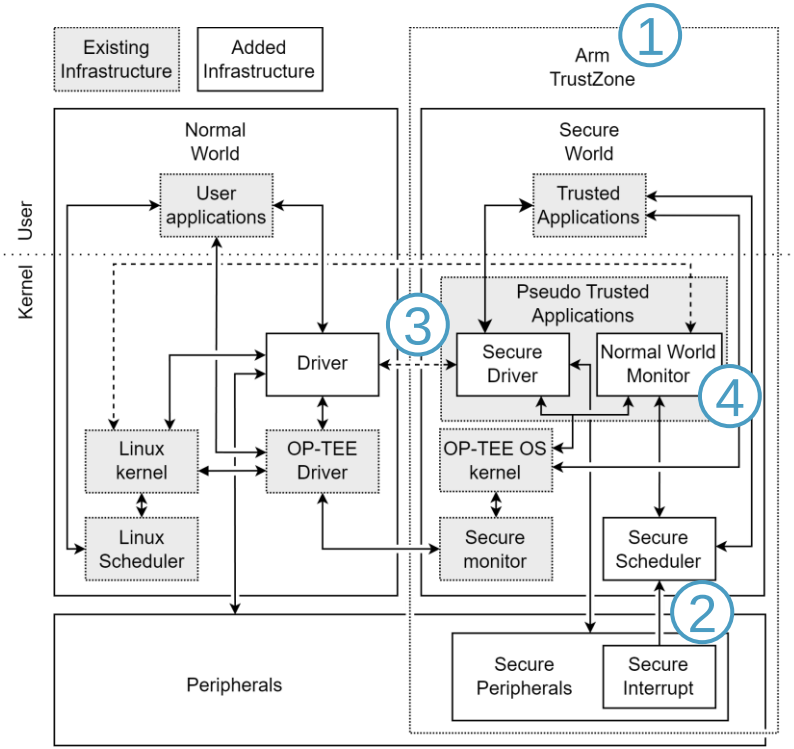
Requirement 1 – Isolation of critical applications

- Hardware based isolation
  - Two security states & two address spaces
  - Confidentiality & Integrity
- Integrated in high and low-end
- OP-TEE
  - Open source TEE implementation
  - Works with Linux



Architecture

1. *Requirement 1:*
   Isolation of critical applications
2. *Requirement 2:*
   Real-time execution of critical applications
3. *Requirement 3:*
   Transparent sharing of peripherals
   (with minimal developer overhead)
4. *Use case:*
   Monitoring of Linux kernel

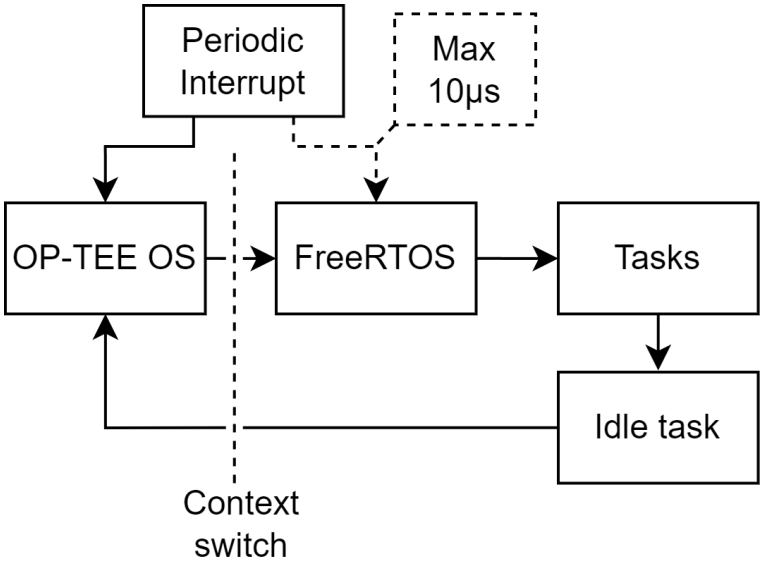


KU LEUVEN DistriNet

Real-Time Secure Scheduler

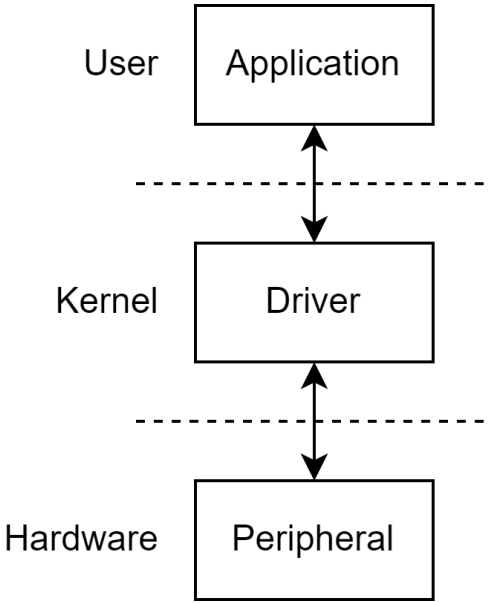
Requirement 2 – Real-time execution of critical applications

- Periodic Interrupt
  - Absolute priority
  - Protected from Normal World
- FreeRTOS
  - Well-known & relatively small
  - Task prioritization & preemption



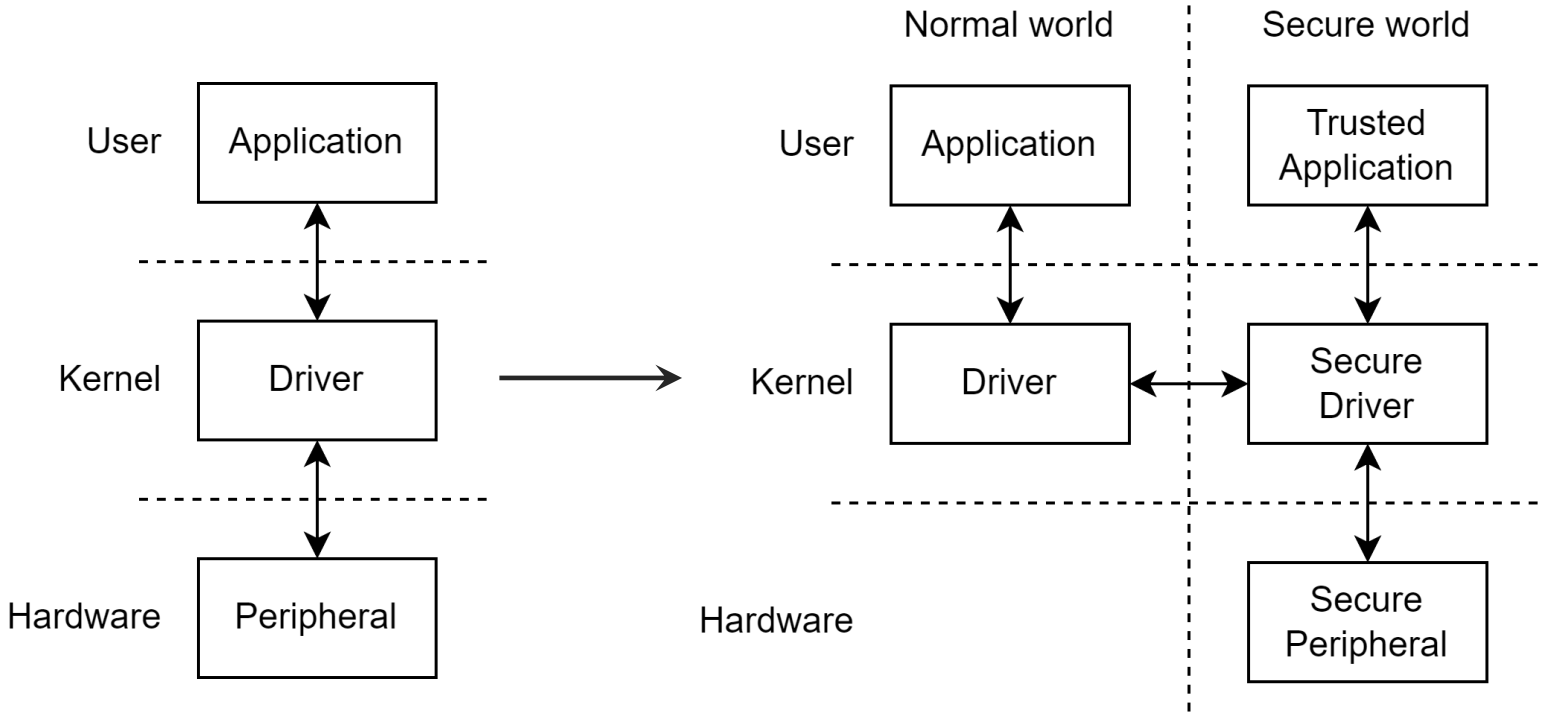
Driver splitting

Requirement 3 – Transparent sharing of peripherals



Driver splitting

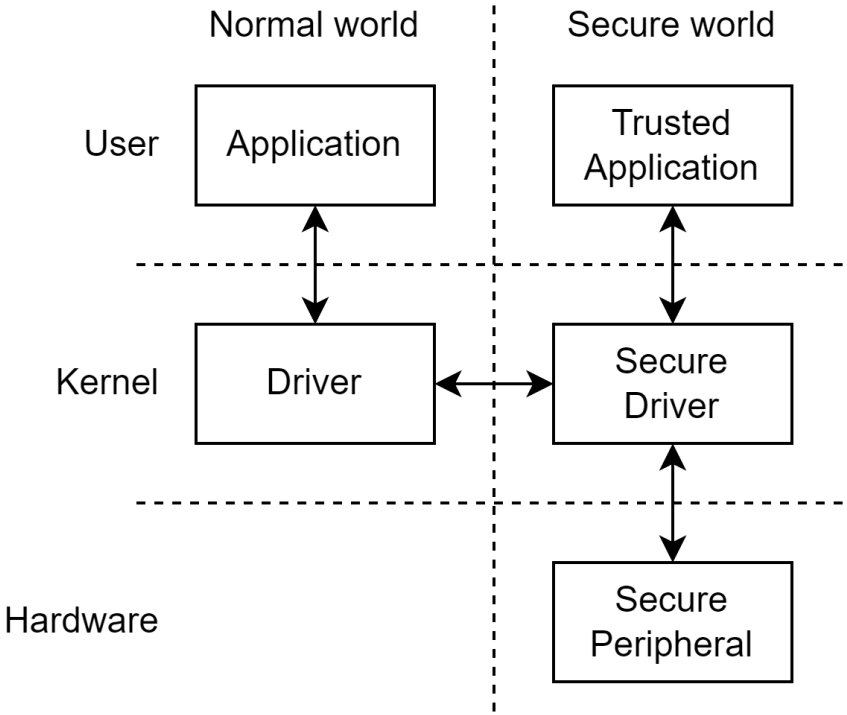
Requirement 3 – Transparent sharing of peripherals



Driver splitting

Requirement 3 – Transparent sharing of peripherals

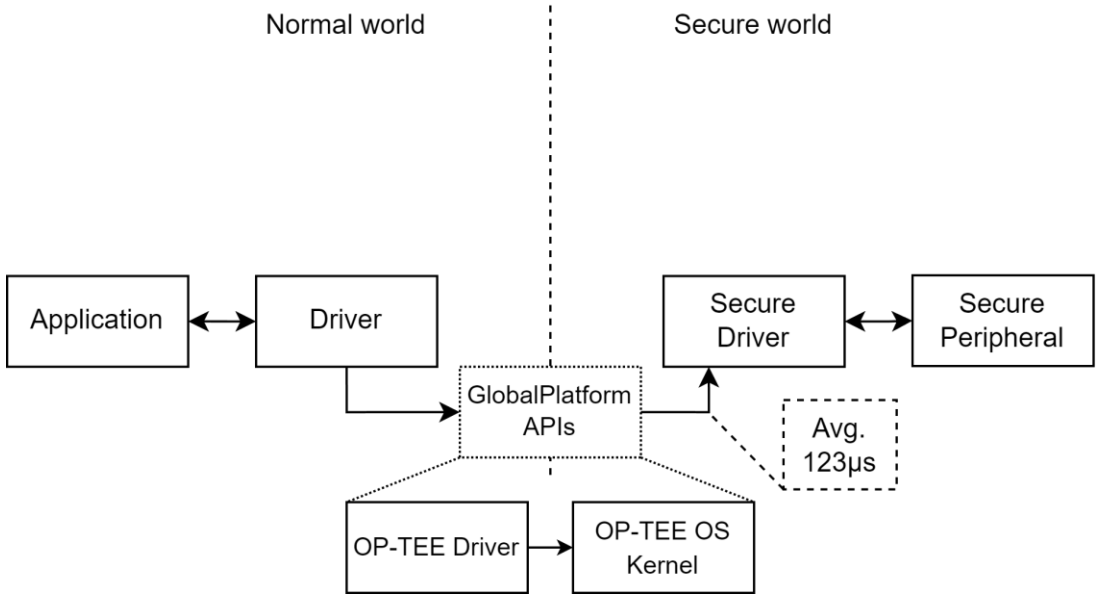
- Secure Driver
  - Security policies
  - Sharing logic
- Only driver developers need to care



Peripheral access for Normal World

Requirement 3 – Transparent sharing of peripherals

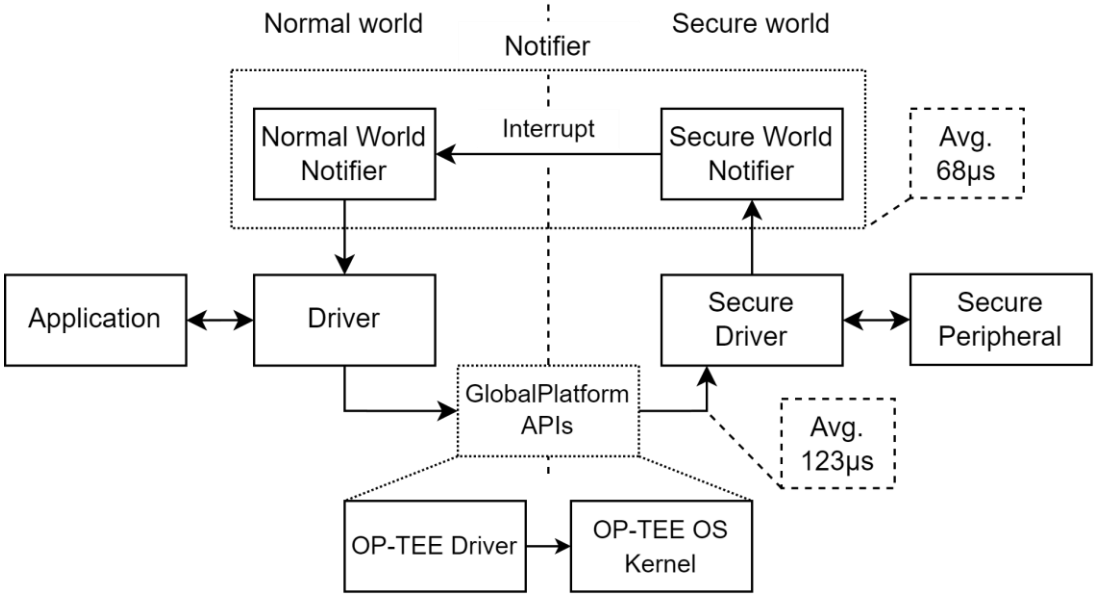
- Read & Write access
  - GlobalPlatfom APIs



Peripheral access for Normal World

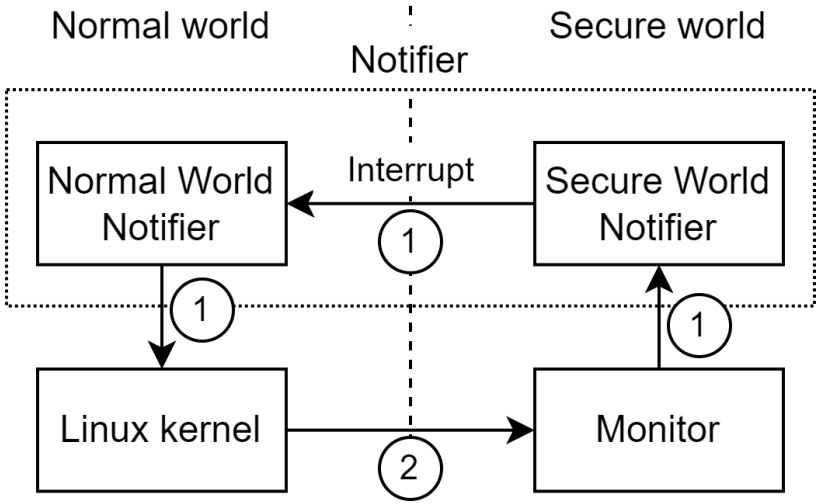
Requirement 3 – Transparent sharing of peripherals

- Read & Write access
  - GlobalPlatfom APIs
- Peripheral Interrupt
  - Notification system
  - Interrupt in Linux kernel
  - Publish-Subscribe



Use case: Monitoring Linux

- Check state of Linux kernel
- Challenge – Response
- Remedial action
  - Kernel dump
  - Kernel reboot



Demo: Rebooting Linux

1. Store kernel image at boot
2. Disable all cores
3. Write image to kernel memory
4. Jump to kernel start address

Conclusion

Can we ensure availability for safety-critical applications while running a commodity OS on the same system with little developer impact?

YES!

Conclusion

Can we ensure availability for safety-critical applications while running a commodity OS on the same system with little developer impact?

1. Leveraging TrustZone isolation
2. Secure scheduling with FreeRTOS
3. Transparent peripheral sharing

Try it yourself!

- Tutorial: https://distrinet-tacos.github.io/documentation
  - Boundary Devices BD-SL-i.MX6
- OP-TEE documentation: https://optee.readthedocs.io/en/latest/index.html
- Contact me: tom.vaneyck@kuleuven.be

FOSDEM 2023

Keeping safety-critical programs alive when Linux isn’t able to

Tom Van Eyck – tom.vaneyck@kuleuven.be

Co-authors: Hamdi Trimech, Majid Salehi, Thanh-Liem Ta, Sam Michiels, Danny Hughes, Hassaan Janjua

With the support of VLAIO.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 101020416.

KU LEUVEN DistriNet

VLAIO