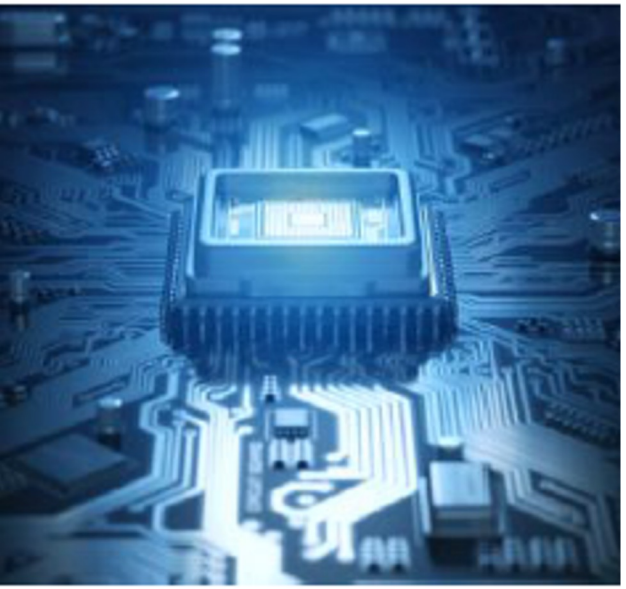


REWIRE TEE Keystone Extensions

CROSSCON and (Secure) Friends

Samira Briongos
NEC Laboratories Europe

27/06/26

REWIRE Overview

REWiring the ComposItional Security VeRification and AssurancE of Systems of Systems Lifecycle

- **Continuous security assessment** and management of IoT devices throughout the entire lifecycle
- **Security-by-design** through formally verified open-source software and open standard hardware designs for attack surface minimisation
- **Runtime verification** of IoT trustworthiness through cryptographically verifiable security proofs and efficient **attestation**
- Auditable security **patch management** and **misbehaviour detection**
- Continuous **authentication and authorization** for the secure communication and identity management in IoT ecosystems

Why RISC-V?

- Open Source ISA with many softcore implementations available
  - Details of the HW implementation available
  - Avoid Security by Obscurity
  - Customization
- Growing community
  - Both for Hardware and Software
- Potential contributions to reduce security threats of open-source hardware and software
- Openness and freedom

REWIRE Use cases

Smart Cities

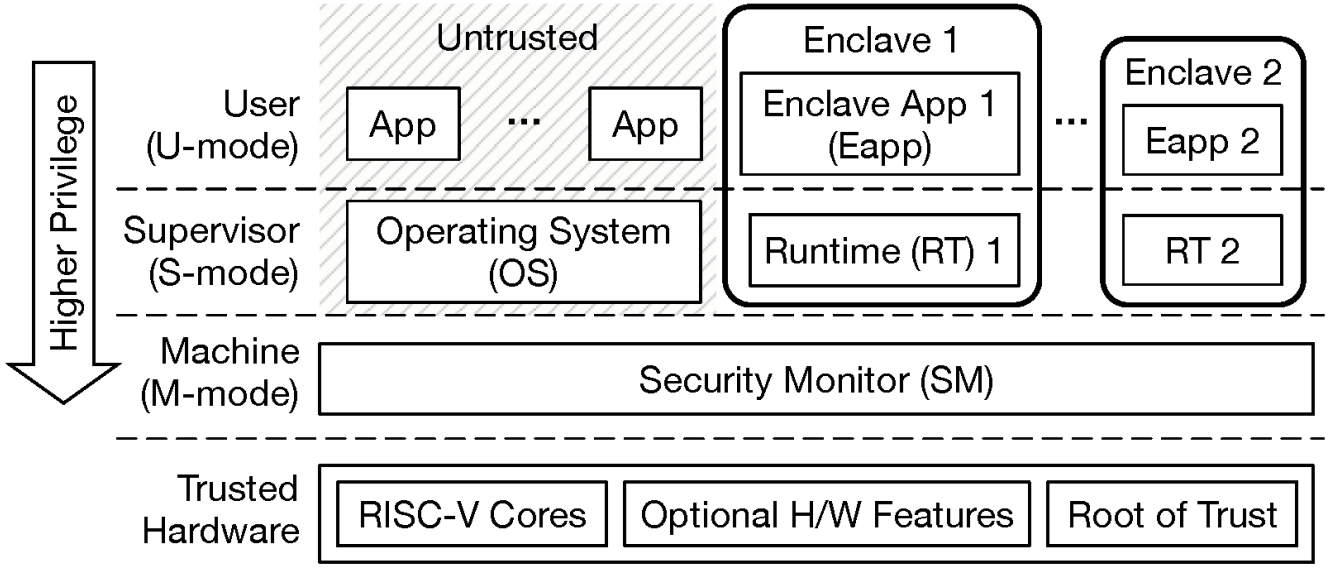
Automotive

Smart Satellites

REWIRE Outcomes

- Enhanced Configuration Integrity Verification (CIV) **attestation**
- Enclave **update** and migration
- **“Crypto Agility Layer”**
  - Cryptographic primitives, Key Management System and Key Hierarchies.
- **Formal verification** of cryptographic protocols
- **Zero-Touch Onboarding** mechanisms
- **Binary Instrumentation** for instantiating monitoring hooks capable of tracing the binary control-flow - for RISC-V binaries
- AI based **misbehavior detection**
- …..

REWIRE's TEE Architecture



Source: Dayeol Lee, David Kohlbrenner, Shweta Shinde, Krste Asanović, and Dawn Song. 2020. Keystone: an open framework for architecting trusted execution environments. In Proceedings of the Fifteenth European Conference on Computer Systems (EuroSys '20). Association for Computing Machinery, New York, NY, USA, Article 38, 1–16. https://doi.org/10.1145/3342195.3387532

- RISC-V defines three different privileges:
  - M-Mode
  - S-Mode
  - U-Mode
- M-mode can control access to physical memory from lower privileges (U-/S-modes). -> *isolation*
- **Keystone** is an open source implementation of the SM that is software based and just leverages the ISA and standard HW features to build TEEs.

Extending Keystone

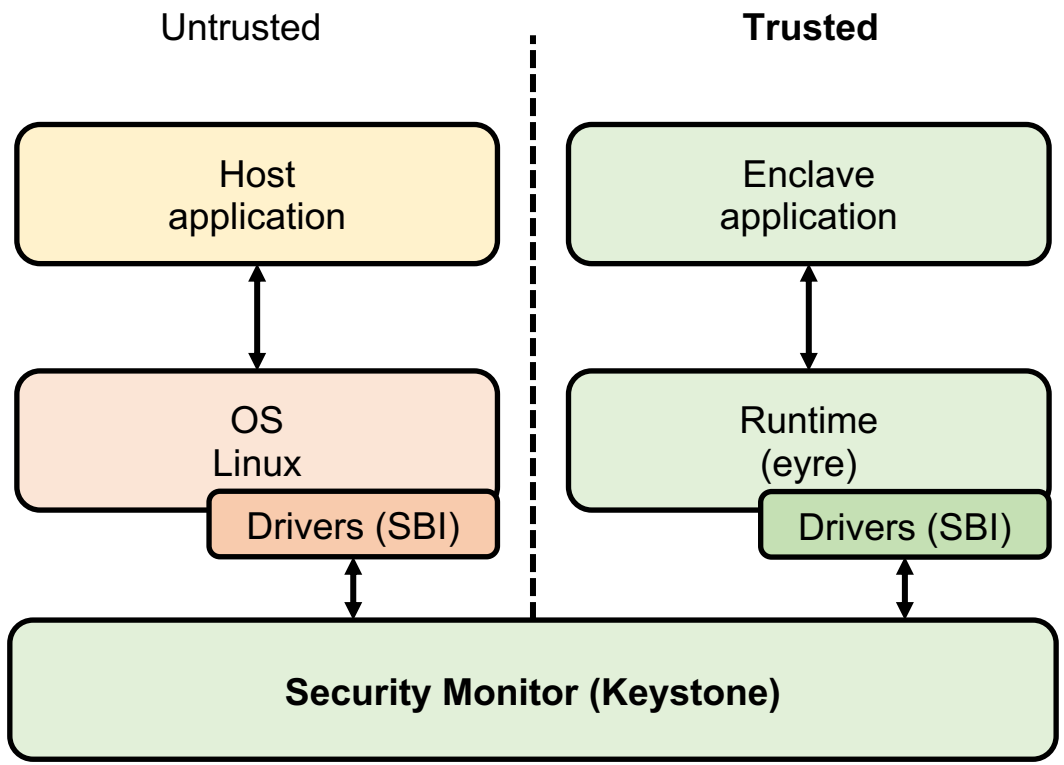
Extending the SBI

[Context]: The SBI (Supervisor Binary Interface) is an interface between the Supervisor Execution Environment (SEE) and the supervisor. The supervisor can then make requests to the M-Mode software via *ecalls* and access to hardware resources managed by the M-Mode. The SBI reduces duplicate platform code across OSes and makes it possible to have common drivers for them.

REWIRE Extensions:

- New interfaces to communicate **from the trusted world with the Security Monitor (SM)**
- New interfaces to communicate **from the untrusted world with the enclaves via the SM**
- Procedure to expose functionality from the SM to the untrusted and trusted apps.

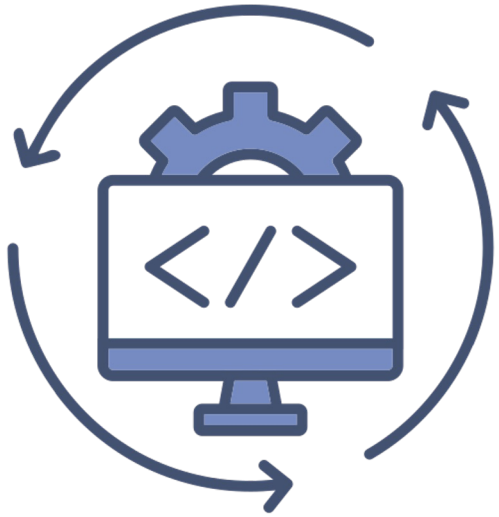
Understanding the communication between host and enclave



- Enclave and host applications communicate with the Runtime and OS via syscalls
- Runtime and OS communicate with the SM via SBI calls

Support for Secure SW/FW Updates

The secure Software Development lifecycle



1. Analysis
2. Design
3. Implementation
4. Test
5. Deployment
6. Maintenance

The secure life cycle management of a device requires a secure update mechanism, which allows the update of enclaves and state transfer without breaking any security guarantee.

Challenges of a Secure SW Update

- Ensure only authenticated and authorized updates are installed.
- Prevent rollback to previous (insecure) versions of the software.
  - ✓ While offering a fall-back mechanism in case of failure
- Provide state synchronization between the terminated and updated enclaves.
- Support for different update policies and triggers
  - ✓ E.g. Failed attestation
- Provide **strong** evidence that the update has been installed correctly.

The SW Update Package

- Modified regular Keystone software package (*.ke) to include metadada
- Update package generation process:

Developer

Update package:
- SW (binaries)
- SW ID
- SW version number
- Sig_{sk}(SW, ID, version)

publish update package →

Blockchain

Step 1: Update Trigger

- A new threat is identified through the **REWIRE risk assessment process**, e.g. after
  - failed attestation, or
  - design-time formal verification process.
- A new feature is introduced to the software.
- A new version of the enclave is available.

Step 2: Update Verification

Blockchain

SW Update Service

Host application

Security Monitor

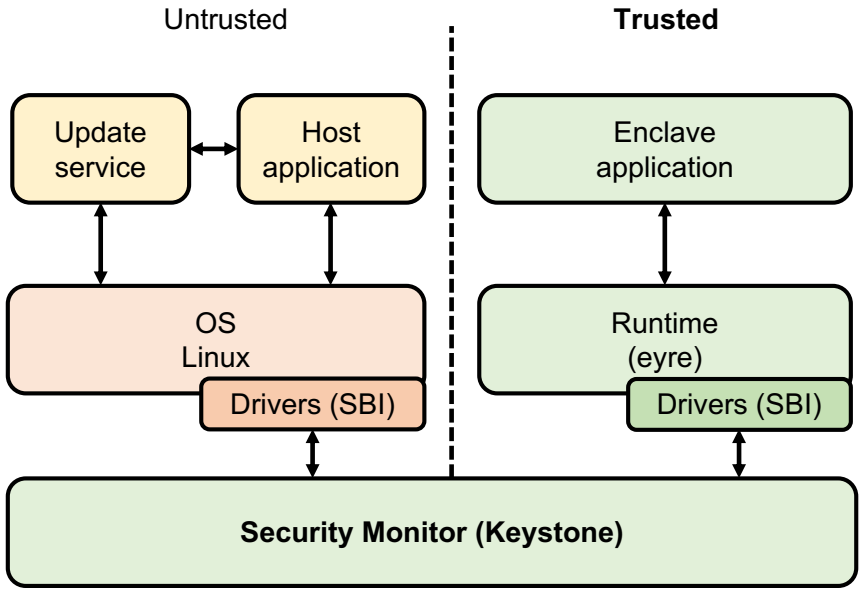
fetch update package

update package

start

Init (SW)

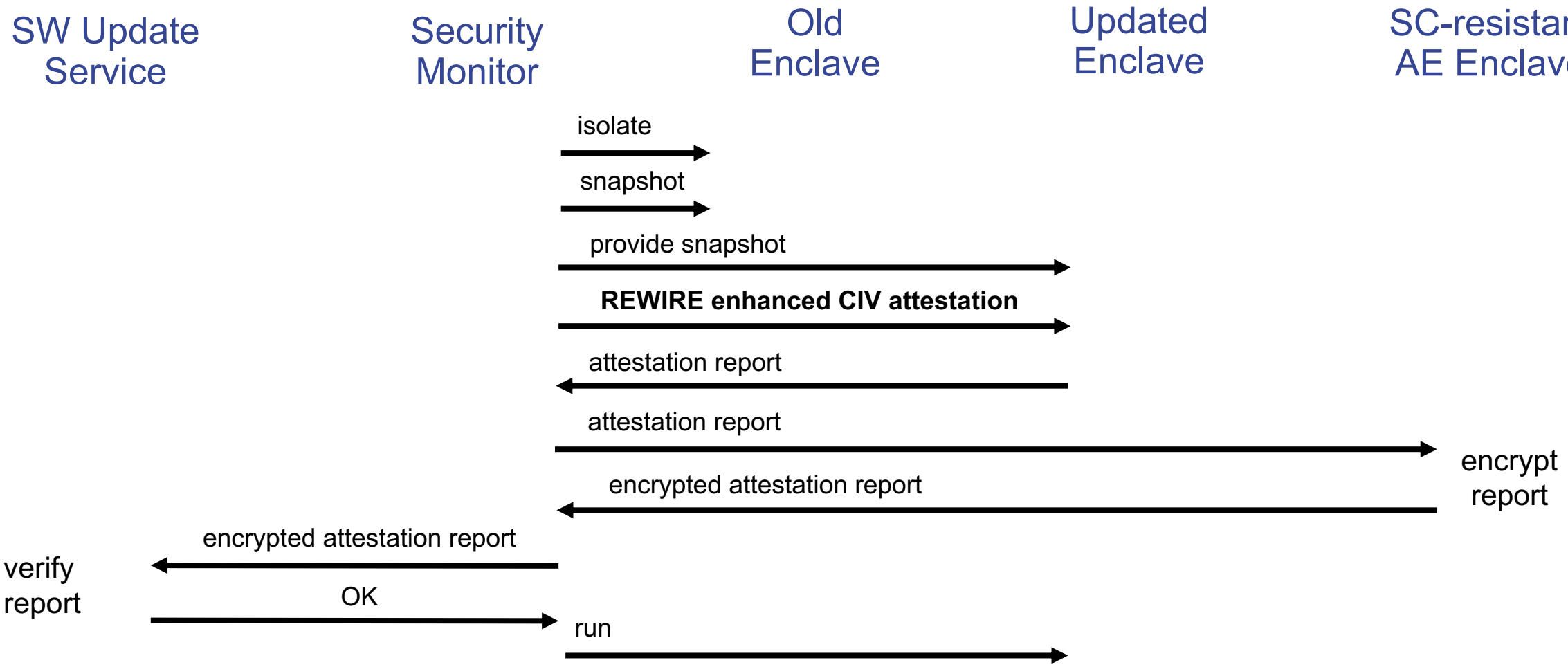
run_update



Verify:

- signature
- SW ID
- Version number

Step 3: Update Installation



Conclusions

- The REWIRE project:
  - Continuous security assessment and management of IoT devices throughout the entire lifecycle
  - Security-by-design, runtime verification, attestation, patch management, misbehavior detection, continuous authentication and authorization, …
- Extensions to Keystone (TEE architecting framework)
  - Understanding the project to enhance it
  - Secure Software update as an example
- Contributions to the Risc-V community.

REWIRE Grant Agreement No. 101070627

Thank you!

If you need further information, please contact me:

Samira Briongos

Samira.Briongos@neclab.eu

Funded by the European Union

Funded by the European Union under grant agreement no. 101070627. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or CINEA. Neither the European Union nor the granting authority can be held responsible for them.