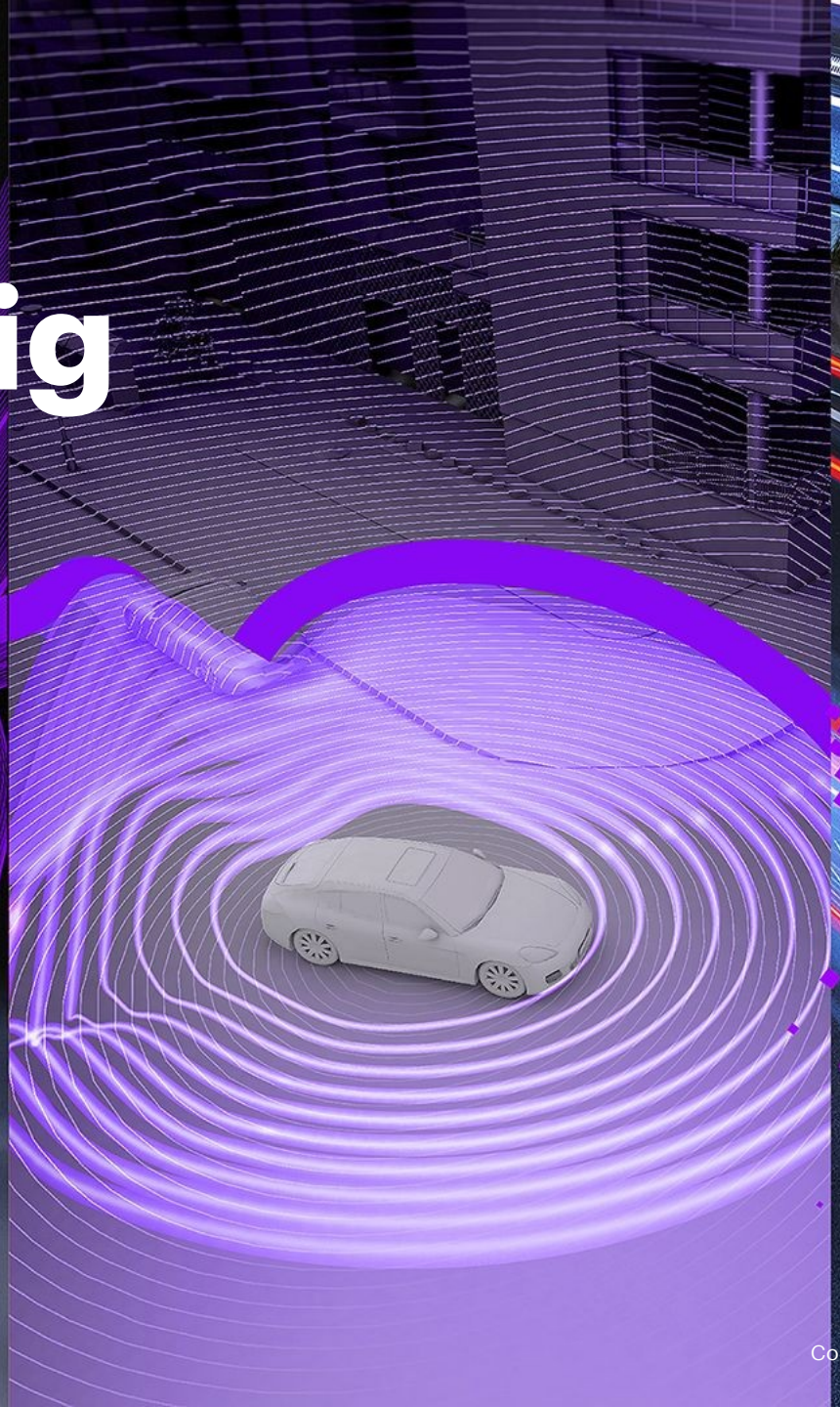
accenture

Eclipse Heimlig

A firmware for Hardware Security Modules (HSMs) written in Rust

Norbert Fabritius

29/03/2023



Security Challenges in Modern Cars

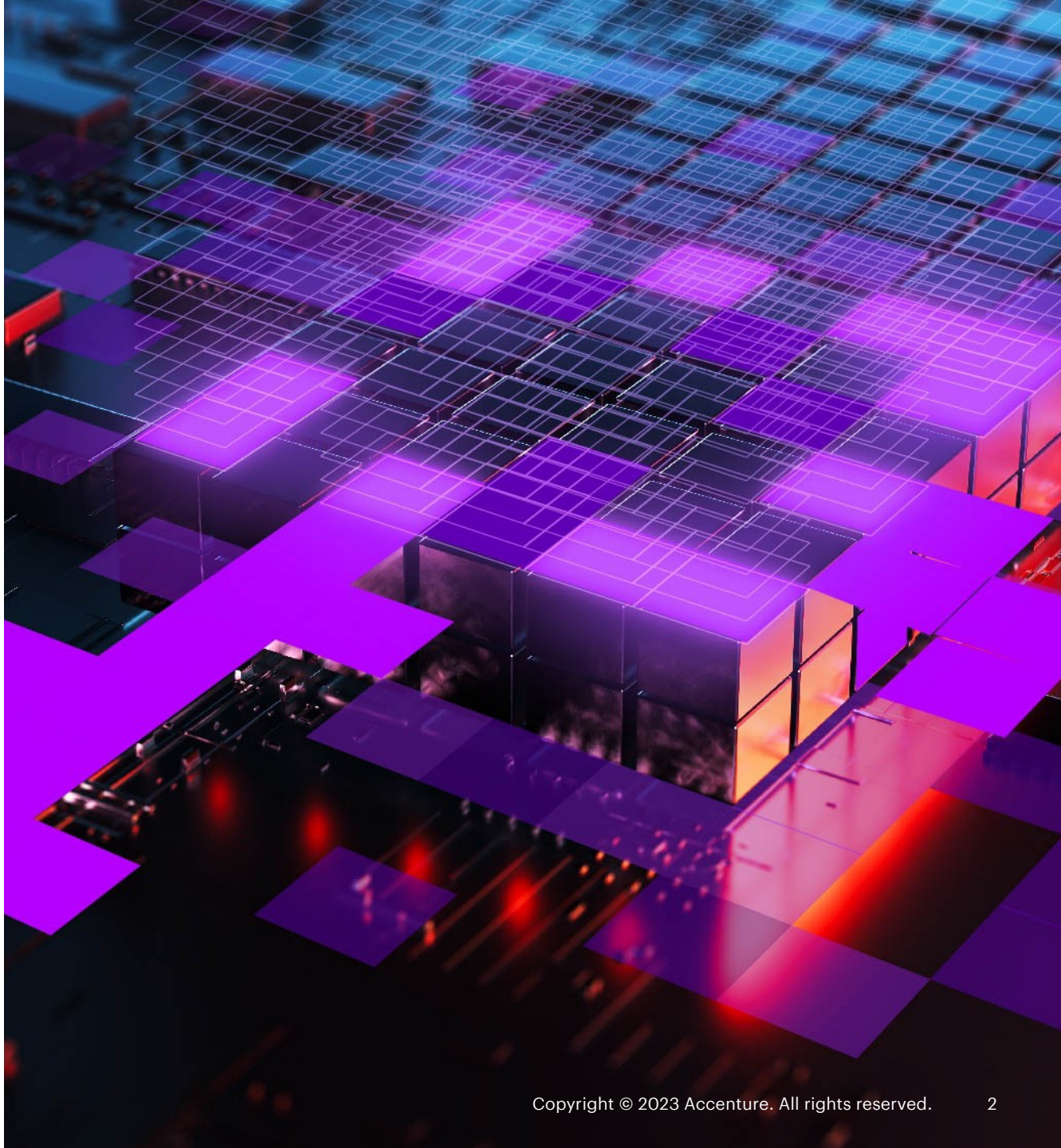
"A single modern luxury vehicle now can integrate as many as 150 ECUs."

eeNews Europe, 2019

Today's cars are large internet-connected computer networks on wheels.

To keep these networks secure, different cryptographic services is required:

1) Secure **transmission** of data
2) Secure **storage** of data
3) Secure **generation** of key material



What is a Hardware Security Module (HSM)?

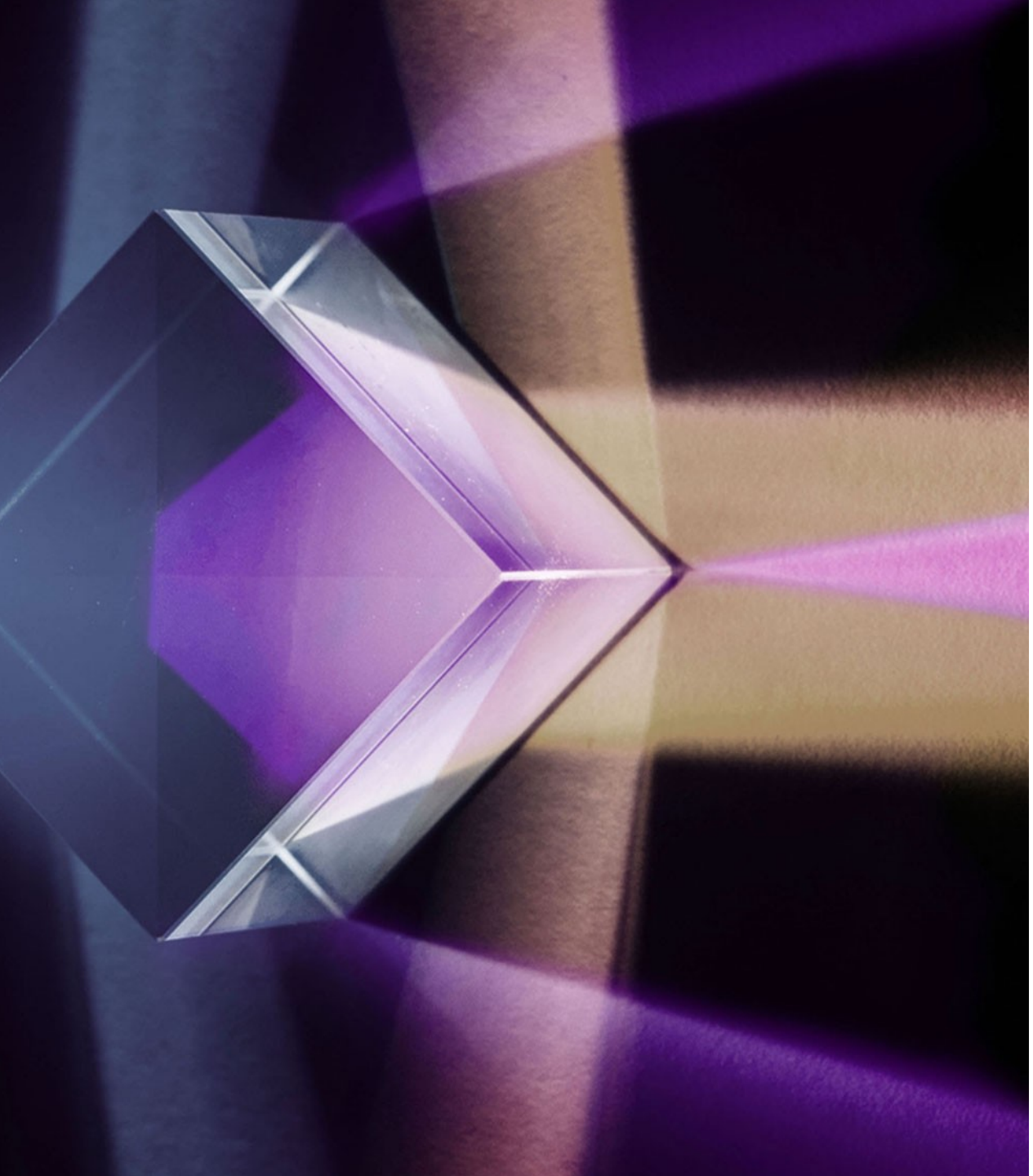
An HSM is a **dedicated hardware** component that provides cryptographic services to other components:

- Secure generation and storage of cryptographic keys.
- Hardware-accelerated and -protected cryptographic operations: Encryption, decryption, signing and verification of data
- An HSM **never reveals** private key material
  - Even if all other systems are compromised.
  - It might even **destroy keys** if a physical attack is detected.

The HSM **hardware** only provides low-level cyptographic operations.

The HSM **firmware** uses these to build a ready-to-use component.

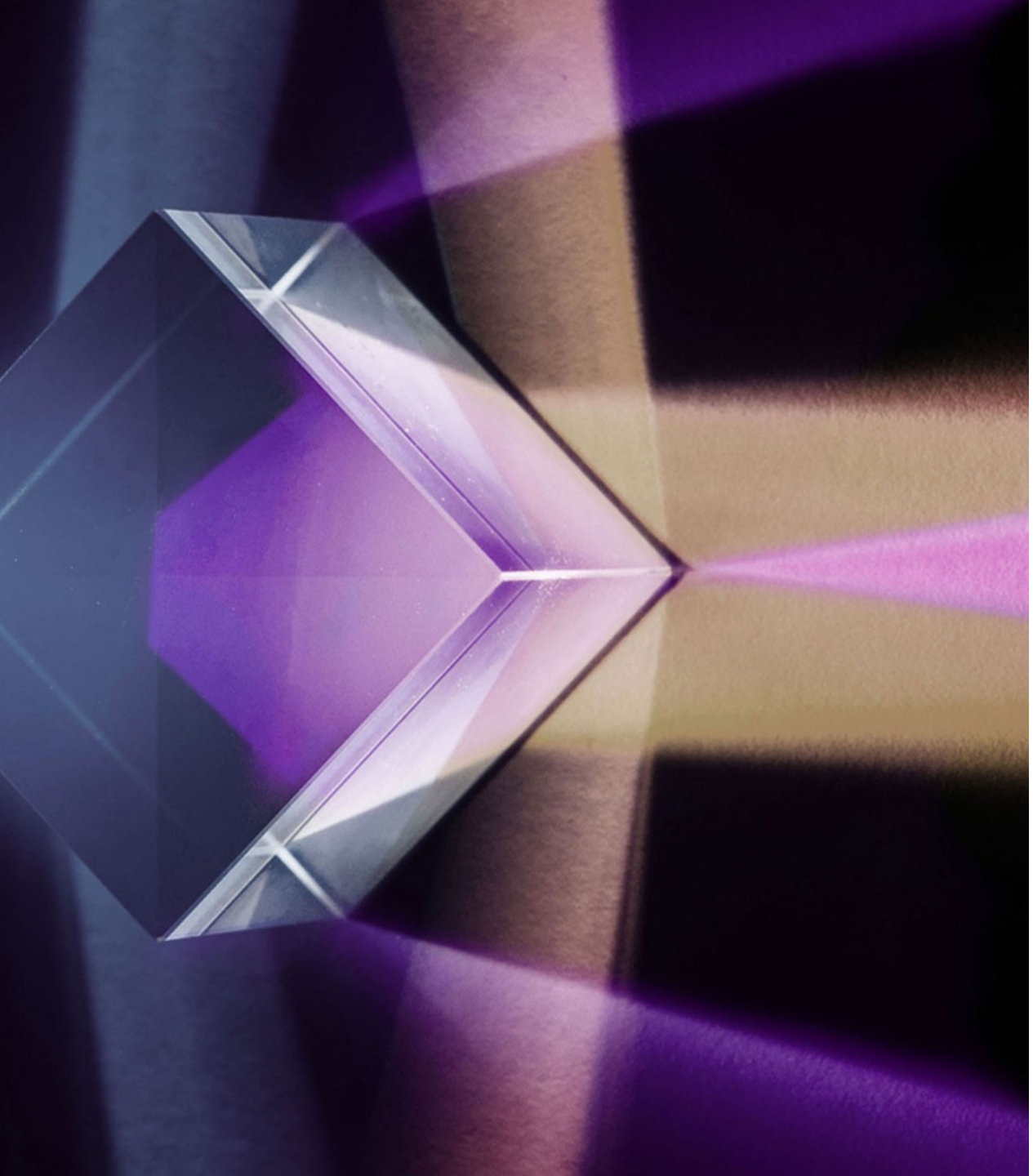
A traditional **operating system** (e.g. FreeRTOS, Embedded Linux) is not required.



Why another HSM firmware?

- HSMs are not a new idea
- Most are written in C or C++
  - Insecure **memory handling**:
    Buffer overflows,
    access to invalid memory
  - Little protection from **data races**:
    Multiple contexts accessing the same memory at the same time
  - Error-prone **complexity**

⇒ Never ending source of **vulnerabilities**



Why another HSM firmware?

"It will demonstrate several attack paths [...]

full control of the HSM. [...]

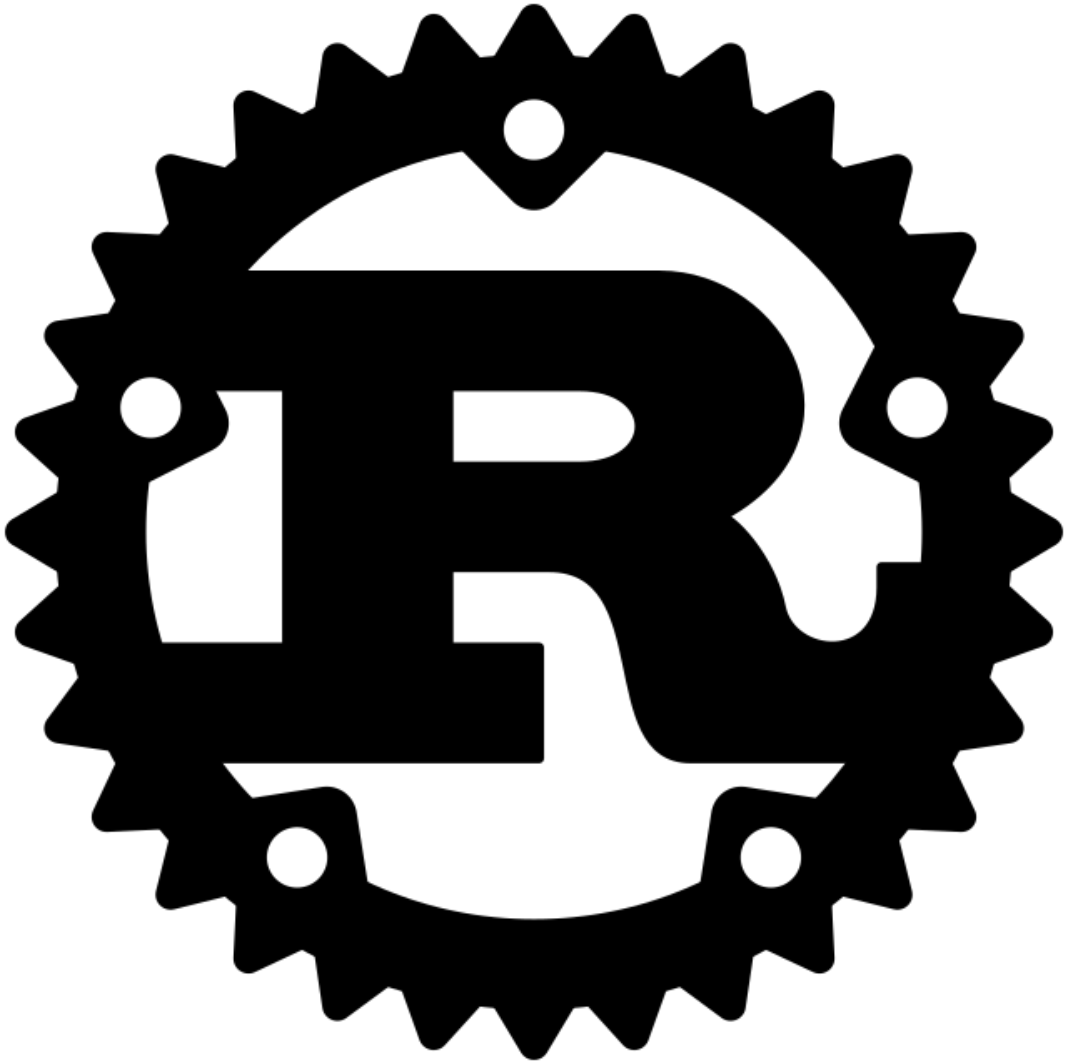
retrieving all HSM secrets remotely [...]

persistent backdoor that survives a firmware update."

What is Rust?

Rust is an emerging programming language.

1) **Reliable**
   Strong memory safety and
   concurrency guarantees
   → Ideal for **security** components
2) **Efficient**
   Compile to native binaries
   Little overhead (”Pay for what you use”)
   → Required for **embedded** components
3) **Productive**
   Provide modern tools and ready-to-use libraries



What is Eclipse Heimlig?

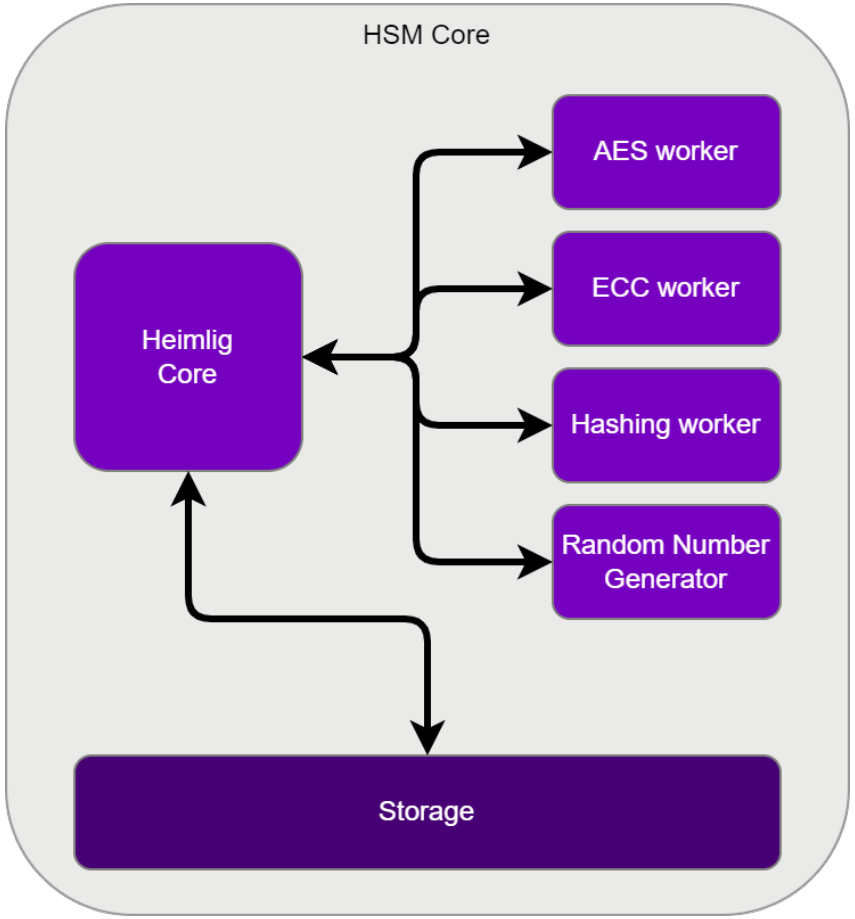
Eclipse Heimlig is an **HSM firmware** written in **Rust**.

Generic Core

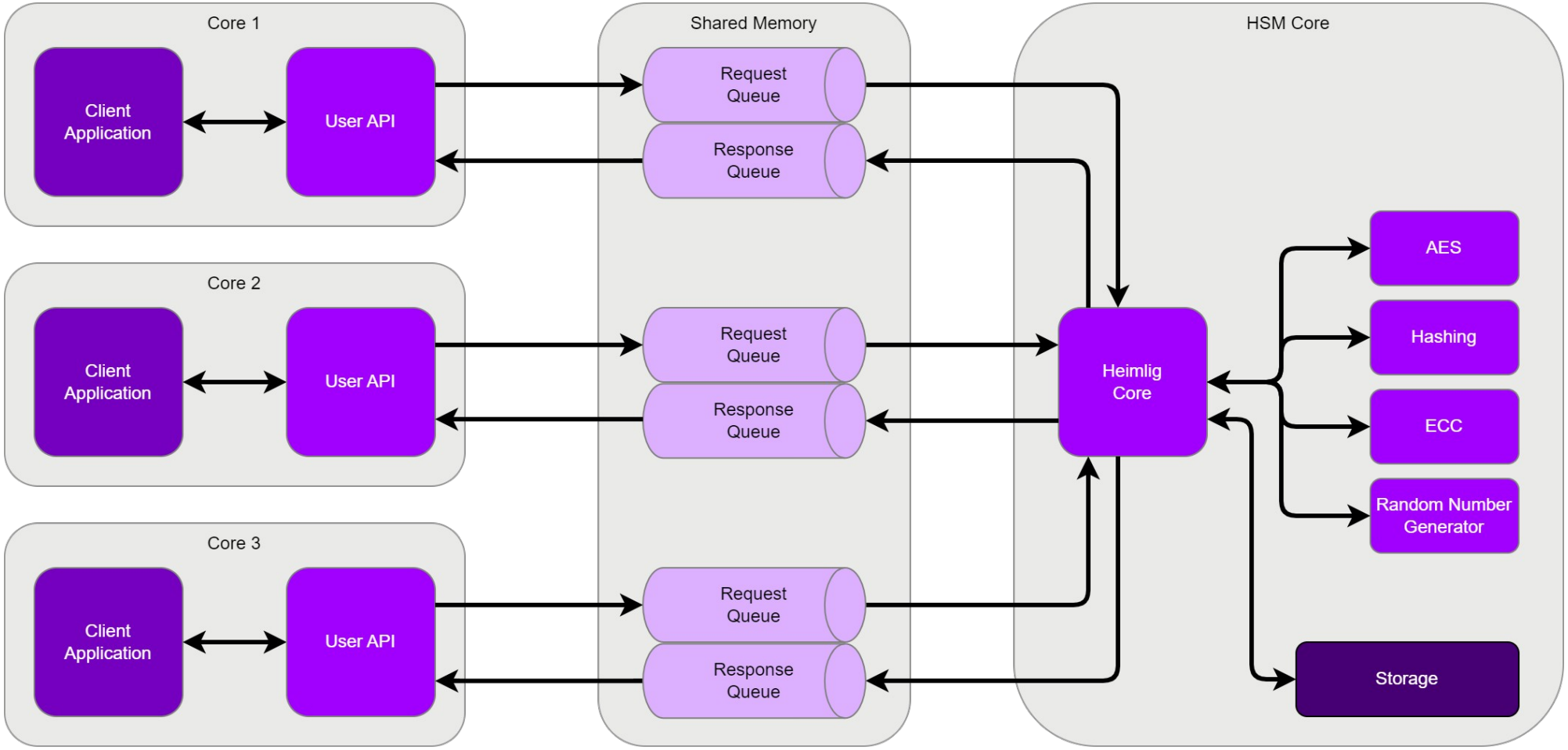
- Accepts and schedules incoming crypto. requests
- Manages keys and forwards requests to workers
- Routes responses back to clients

Workers

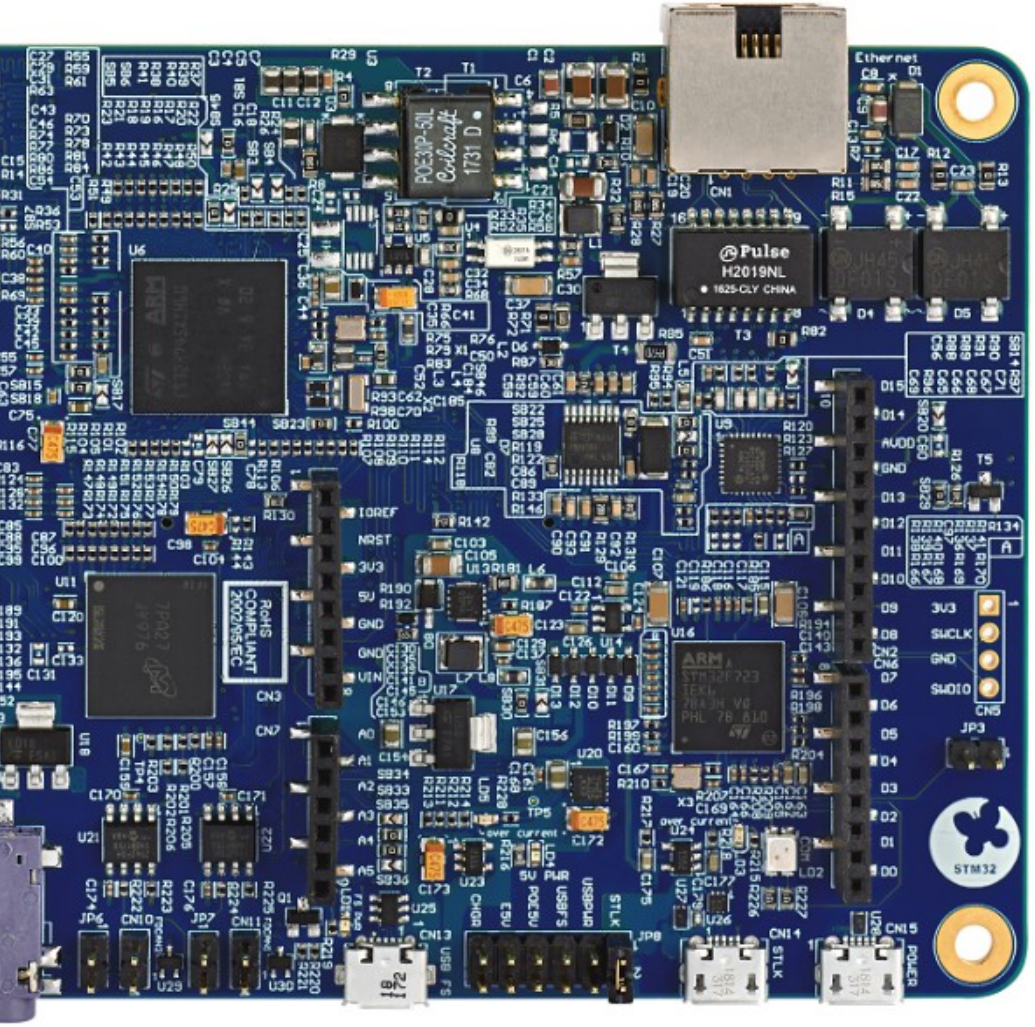
- Modules that perform cryptographic operations
- Portable Software workers for common cryptographic algorithms are provided
- Custom workers that leverage hardware-specific functions can be added by the integrator
- Workers can be **“mixed and matched”**



Architecture



Current Status



STM32H745I-DISCO

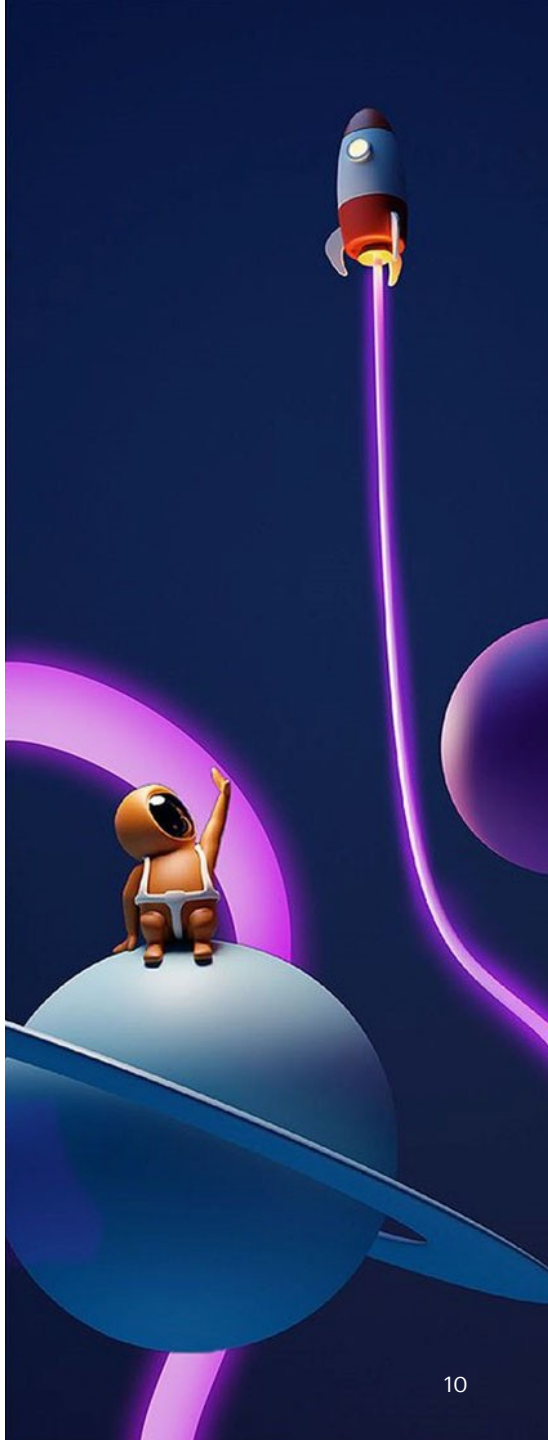
- Heimlig is in the prototyping stage
- Runs on STMicroelectronics discovery board
  - ARM Cortex-M7 processor
  - Uses hardware random number generator
- Runs on Linux for development
- Implements common cryptographic algorithms as software workers:
  - Encryption:
    AES-CBC/-GCM/-CCM, Chacha20Poly1305
  - Hashing:
    SHA-2, SHA-3, BLAKE3
  - Signing, verifying and key exchange:
    ECDSA, ECDH
  - Random number generation:
    ChaCha20 seeded by hardware

Next Steps

Publish code on Eclipse repository.
Contributions, reviews and ideas are **very welcome** ❤️

Research and implement:

- Efficient and safe shared memory interface
- Parallelize worker operations
  - Using Rust's native async mechanism requires a hardware-specific executor
  - Simpler approaches possible
- Secure software update and JTAG unlocking in the field



Thank You

Norbert Fabritius

norbert.fabritius@esrlabs.com