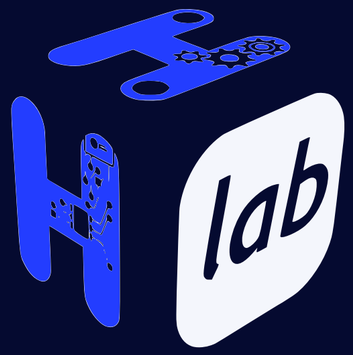

Frama-C usage in Sentry kernel

FORMAL PROOFNESS FOR MICRO-CONTROLERS

PART 1: NORTE COVERAGE OF KERNEL ENTRYPOINT

noRTE ?

- RTE == **Run Time Error**
  - OOB read or write accesses, data corruption, etc.
  - all Undefined Behaviors

- Critical in various systems (medical, cyber-physical, etc.)

- allows a first step toward effective correctness
  - once noRTE, behavioral verification can be made

Coverage

- **All the kernel entrypoint**, up-to the first user-space context switch
  - platform initialisation, kernel contexts initialization
  - coverage from reset handler

- only backend asm replaced by stubbing
  - e.g. __ioread32() returning unpredictable volatile

- STM32 kernel drivers **included**

And the hardware ?

- kernel-level hardware IPs **declared** to Frama-C as read-write areas
- Using SVD files for **automatic *predicates* generation** defining registers access
- driver's I/O accesses **assertions**, based on predicates

```
// generated predicate
predicate rcc_is_readable_register(ℤ r) = (
        r == 0x0 ||
        // [...]
        \false
);

/*@ assert rcc_is_readable_register(reg); */
```

What is verified ?

- All **Undefined behaviors** using EVA plugin
- **overflows, div by 0 & invalid memory accesses** using RTE plugin
- using an **over-approximation** of environment for completeness
  - e.g. unpredictable registers values at each access
- noRTE limited to mono-threaded mode, which is the entrypoint mode

How is it integrated ?

- Call to Frama-C in the Sentry kernel meson build system
  - Using generated compilation-database as input
- All Frama-C entrypoints associated to dedicated executable()
- Declarated as test() to allow the usage of the *meson test* subsystem
- Frama-C execution added to a dedicated CI workflow

Results

✅ Covering kernel production build, only with stubbed low level ASM

✅ Other handlers (syscalls, ticker, faults) are covered by separated Frama-C entrypoints, with the same principle

✅ Covering 95% of the theorically reachable blocks

- last 5% mostly unreachable defensive

✅ No triggered Red Alarm (all RTE fixed), only false positives

- e.g. task layout initialization

Thank you !

Special thanks to CEA-LSL team

https://github.com/camelot-os/sentry-kernel
https://frama-c.com/index.html