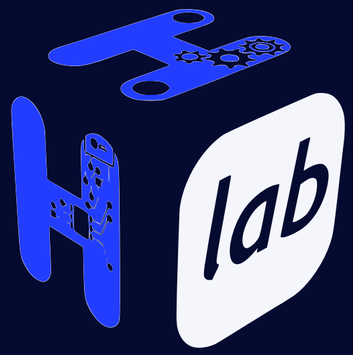

Frama-C usage in Sentry kernel

FORMAL PROOFNESS FOR MICRO-CONTROLERS

PART 3: PROVING W^X POLICY

Proving ?

- Demonstrating **behavioral properties** of a subprogram, matching a higher level specification (e.g. W^X)
- Demonstrating that an expected result or property is **always true**, whatever the call context is
- Requires two steps:
  - the sub-program does implement this property
  - this sub-program is the lonely one that can impact this property

Coverage

- **Senty kernel memory controller implementation is backed in a arch-specific leaf call graph**
  - e.g. ARM PMSA-v8 MPU driver implementation
  - upper API is kept opaque for portability purpose

- **The leaf is analyzed defining the weakest preconditions required to respect the property**
  - Whatever the callgraph is, the property must be respected

Demonstrating W^X

```
/*@
 requires \valid_read(desc);
 requires \valid(resource);
 assigns *resource;
 ensures (\result == K_STATUS_OKAY);
 */
static inline kstatus_t mpu_forge_resource(const struct mpu_region_desc *desc, layout_resource_t *resource) {
 /* W^X conformity */
 /*@
 assert desc->noexec == 0 ==>  (desc->access_perm == MPU_REGION_PERM_RO || desc->access_perm == MPU_REGION_PERM_PRIV_RO);
 assert (desc->access_perm != MPU_REGION_PERM_RO && desc->access_perm != MPU_REGION_PERM_PRIV_RO) ==> desc->noexec == 1;
 */
 resource->RBAR = ARM_MPU_RBAR_AP(
 desc->addr,
 desc->shareable ? ARM_MPU_SH_INNER : ARM_MPU_SH_NON,
 desc->access_perm,
 desc->noexec ? 1UL : 0UL
 );
[...]
```

What is verified ?

- For all callgraphs [1], the behavior is **demonstrated W^X**
- Only this implementation is able to access the MPU registers (not yet formally demonstrated)
- PMSA-v7 and PMSA-v8 (thumbv7-m & thumbv8-m) implementations are proven

[1] See Entrypoint coverage & Handler covertage H²Lab posts

How is it integrated ?

- Demonstrating a valid behavior requires to formally demonstrate the overlall call graph of a given sub-program
- While behavioral analysis is made using Frama-C WP plugin, call context coverage is made using EVA plugin
- Declarated as test() to allow the usage of the *meson test* subsystem
- Frama-C execution added to a dedicated CI workflow

Results

- ✅ Covers overall call contexts of MPU configuration
- ✅ Demonstrates that the implementation ensure a W^X property when mapping memory regions
- ✅ Demonstrates that the MPU driver is valid (MPU HW registers usage, etc.) and never fails or generates UB/RTE
- ⚠ Demonstrating that no other subprogram may impact the MPU is not yet done

Thank you !

Special thanks to CEA-LSL team

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