Semi-automatic Verification of ISA Security Guarantees in the Form of Universal Contracts

Huyghebaert, Sander; Keuchel, Steven; Devriese, Dominique; De Roover, Coen

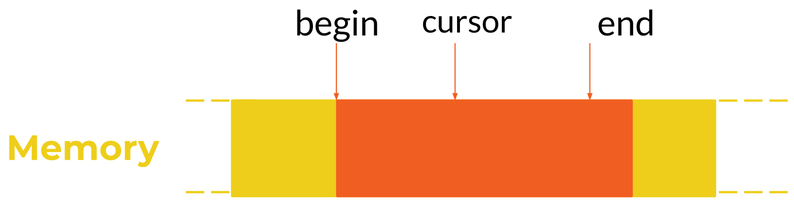
Publication date:
2022

Semi-automatic Verification of ISA Security Guarantees in the Form of Universal Contracts

Sander Huyghebaert, Steven Keuchel, Dominique Devriese, Coen De Roover

MinimalCaps

The MinimalCaps Capability Machine



Capability

- perm ∈ {O, E, R, RW}
- begin : address
- cursor : address
- end : address

Hardware Guarantees

- Capabilities are unforgeable
- Permissions are checked
- Capability manipulation is safe

Capability Safety

$$\mathcal{V}(w) \begin{cases} \mathcal{V}(z) & = \text{True } (z \text{ is an integer}) \\ \mathcal{V}(O, -, -, -) & = \text{True} \\ \mathcal{V}(E, b, e, a) & = \triangleright \square\, \varepsilon(RX, b, e, a) \\ \mathcal{V}(R, b, e, -) & = \ast_{a \in [b,e]}\ \exists\ w, a \mapsto w \ast \mathcal{V}(w) \\ \mathcal{V}(RW, b, e, -) & = \ast_{a \in [b,e]}\ \exists\ w, a \mapsto w \ast \mathcal{V}(w) \end{cases}$$

$$\varepsilon(w) = (pc \mapsto w \ast_{r \in GPR} \exists\ v, r \mapsto v) \mathrel{-\!\!\ast} \text{WP fdeCycle T}$$

Universal Contract:

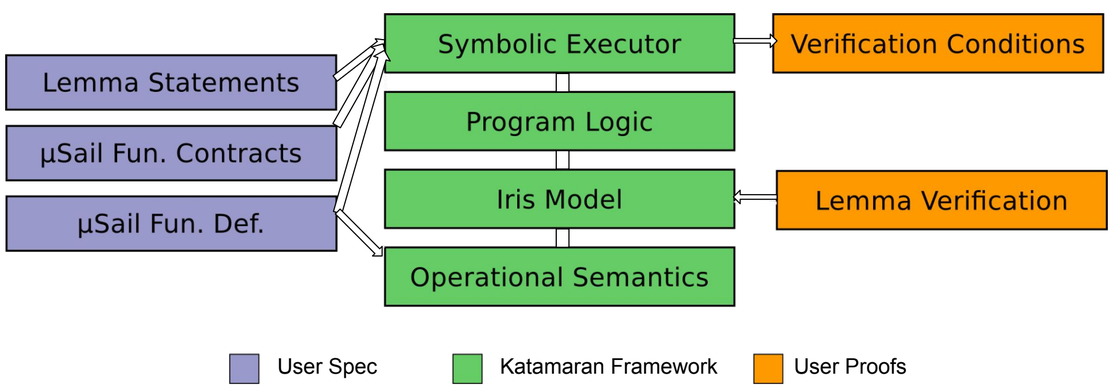
{ (∃ c . pc ↦ c ✱ $\mathcal{V}$(c) ✱ CorrectPC(c)) ✱ (∀ r ∈ GPR . ∃ w . r ↦ w ✱ $\mathcal{V}$(w))

✱ *IH* }

fdeCycle

{ WP fdeCycle T }

Katamaran



Semi-Automatic Verification

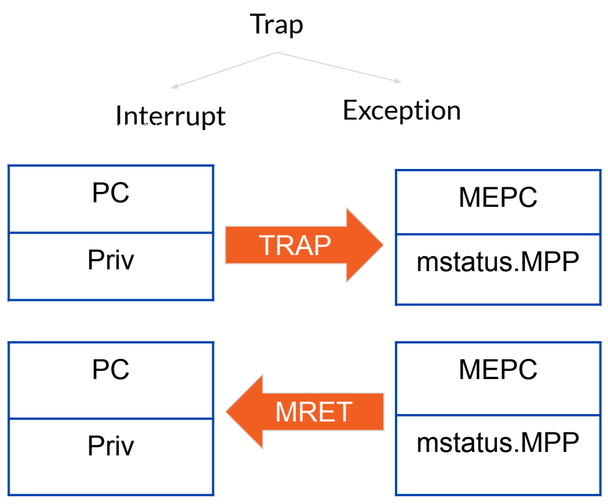
```
{ (∃ c . pc ↦ c ✱ V(c)) ✱ (∀ r ∈ GPR . ∃ w . r ↦ w ✱ V(w)) }
function exec_sd(rs : GPR, rb : GPR, immediate : int) : bool :=
  let base_cap := call read_reg_cap rb in
  let (perm, beg, end, cursor) := base_cap in
  let c := (perm, beg, end, cursor + immediate) in
  let w := call read_reg rs in
  { rb ↦ base_cap ✱ V(base_cap) ✱ rs ↦ w ✱ V(w) ...}
  use lemma  move_cursor base_cap c) ;;
  { rb ↦ base_cap ✱ V(base_cap) ✱ rs ↦ w ✱ V(w) ✱ V(c) ...}
  call write_mem c w ;;
  call update_pc ;;
  true
{ (∃ c . pc ↦ c ✱ (V(c) ∨ ε(c))
   ✱ (∀ r ∈ GPR . ∃ w . r ↦ w ✱ V(w))}
```

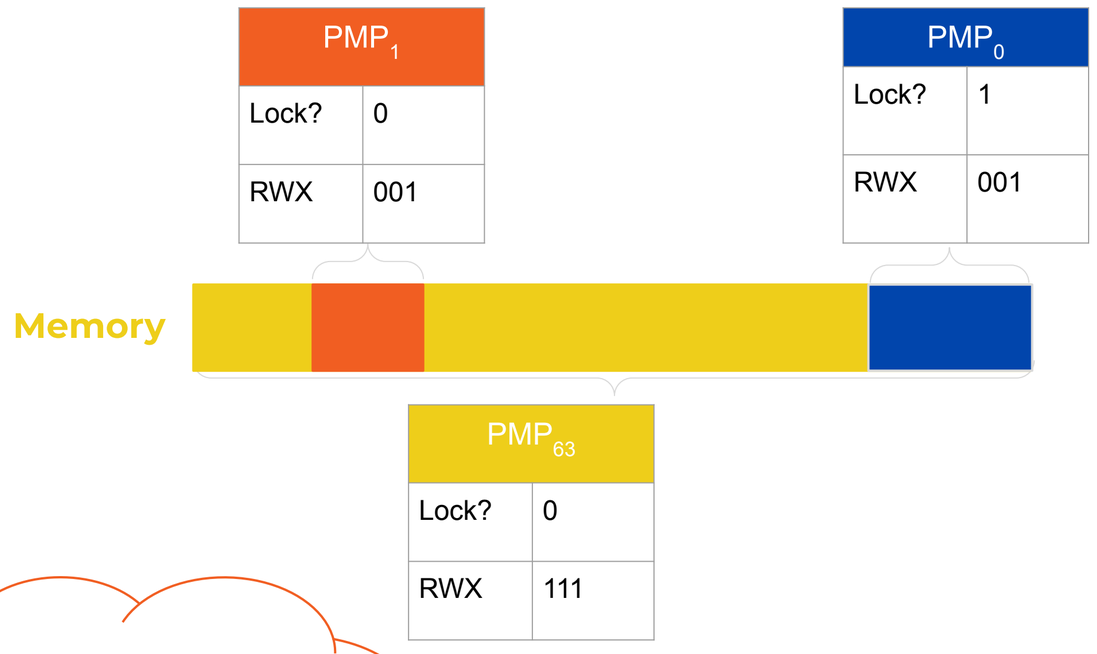
RISC-V PMP

RISC-V PMP

Physical Memory Protection

- Optional
- *Grant* permissions to S and U modes
  - By default *none*
- *Revoke* permissions from M mode
  - By default *full*
- PMP violations => trap
  - Load access fault, store access fault, …
  - Exception
- Up to 64 PMP regions
  - Statically prioritized
    Lowest number has highes





Universal Contract:

{ Start

✱ ▷ (CSRMod -✱ WP fdeCycle T)

✱ ▷ (Trap -✱ WP fdeCycle T)

✱ ▷ (Recover -✱ WP fdeCycle T) }

fdeCycle

{ WP fdeCycle T }

Start of Iteration						
mcause	cur_privilege	nextpc	pc	mtvec	mstatus	mepc
c	p	npc	i	h	mpp	mepc

Normal Execution						
mcause	cur_privilege	**nextpc**	**pc**	mtvec	mstatus	mepc
c	p	npc'	npc'	h	mpp	mepc

Trap						
mcause	cur_privilege	**nextpc**	**pc**	**mtvec**	**mstatus**	**mepc**
c'	Machine	h	h	h	p	i

Recover						
mcause	**cur_privilege**	**nextpc**	**pc**	**mtvec**	**mstatus**	mepc
c	mpp	mepc	mepc	h	User	mepc

CSR Modified						
mcause	cur_privilege	**nextpc**	**pc**	**mtvec**	**mstatus**	**mepc**
c'	Machine	npc'	npc'	h'	mpp'	mepc'

End-to-End Verification

```
init:
    la      ra, adv
    csrrw pmpaddr0, ra, t0
    lui     ra, max
    csrrw zero, pmpaddr1, ra
    lui     ra, 0x0
    csrrw zero, pmp0cfg, ra
    lui     ra, 0xf
    csrrw zero, pmp1cfg, ra
    la      ra, adv
   csrrw zero, mepc, ra
    la      ra, trap_handler
    csrrw zero, mtvec, ra
    lui     ra, 0x0
    csrrw zero, mstatus, ra
    mret

trap_handler:
    auipc ra, 0
    lw      ra, 12(ra)
    mret

data: 42
adv:  …
```

SOFTWARE LANGUAGES LAB

DistriNet KU LEUVEN