RISC-V Hypervisor extension semantics in Sail

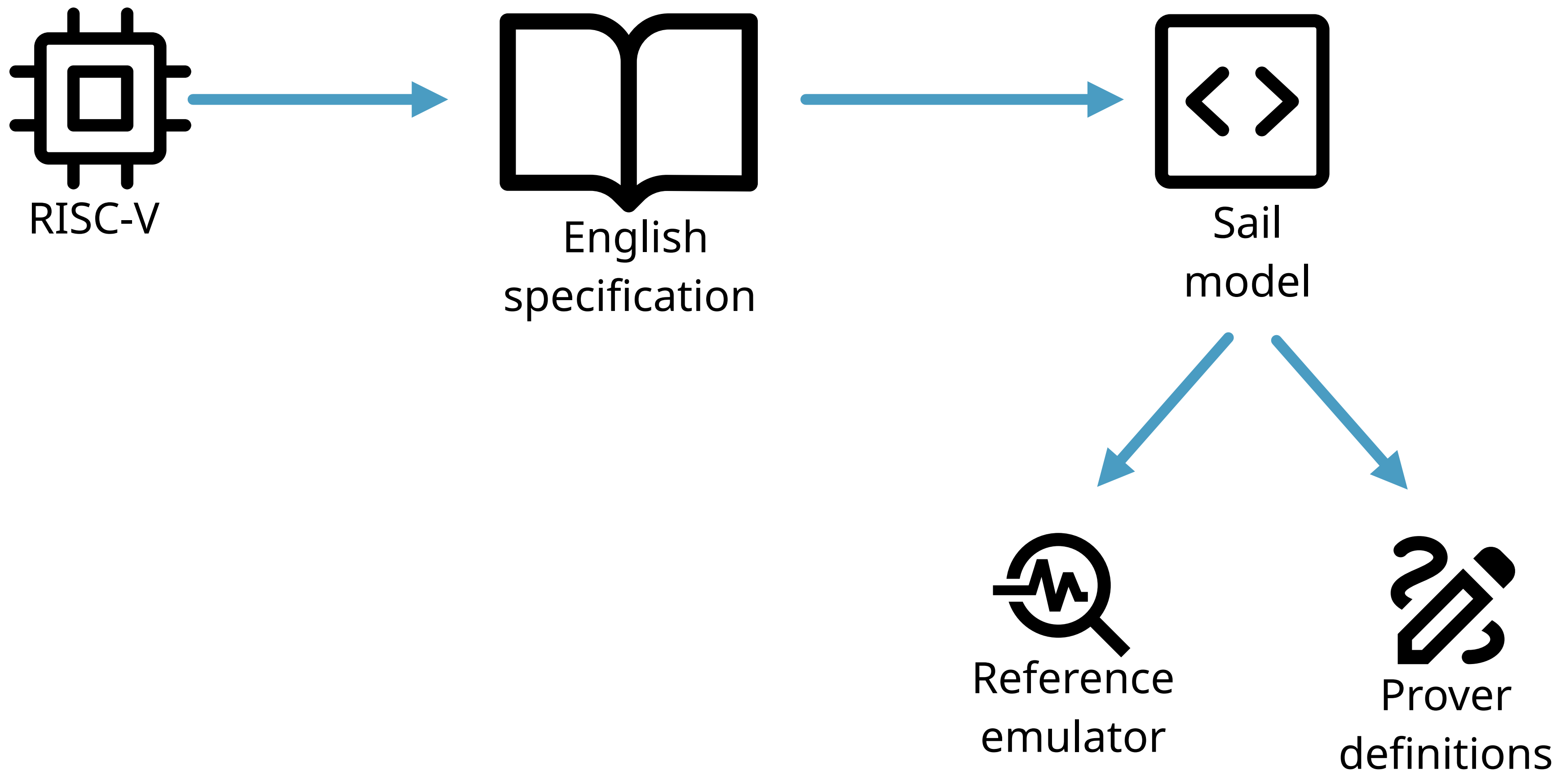
Lowie Deferme

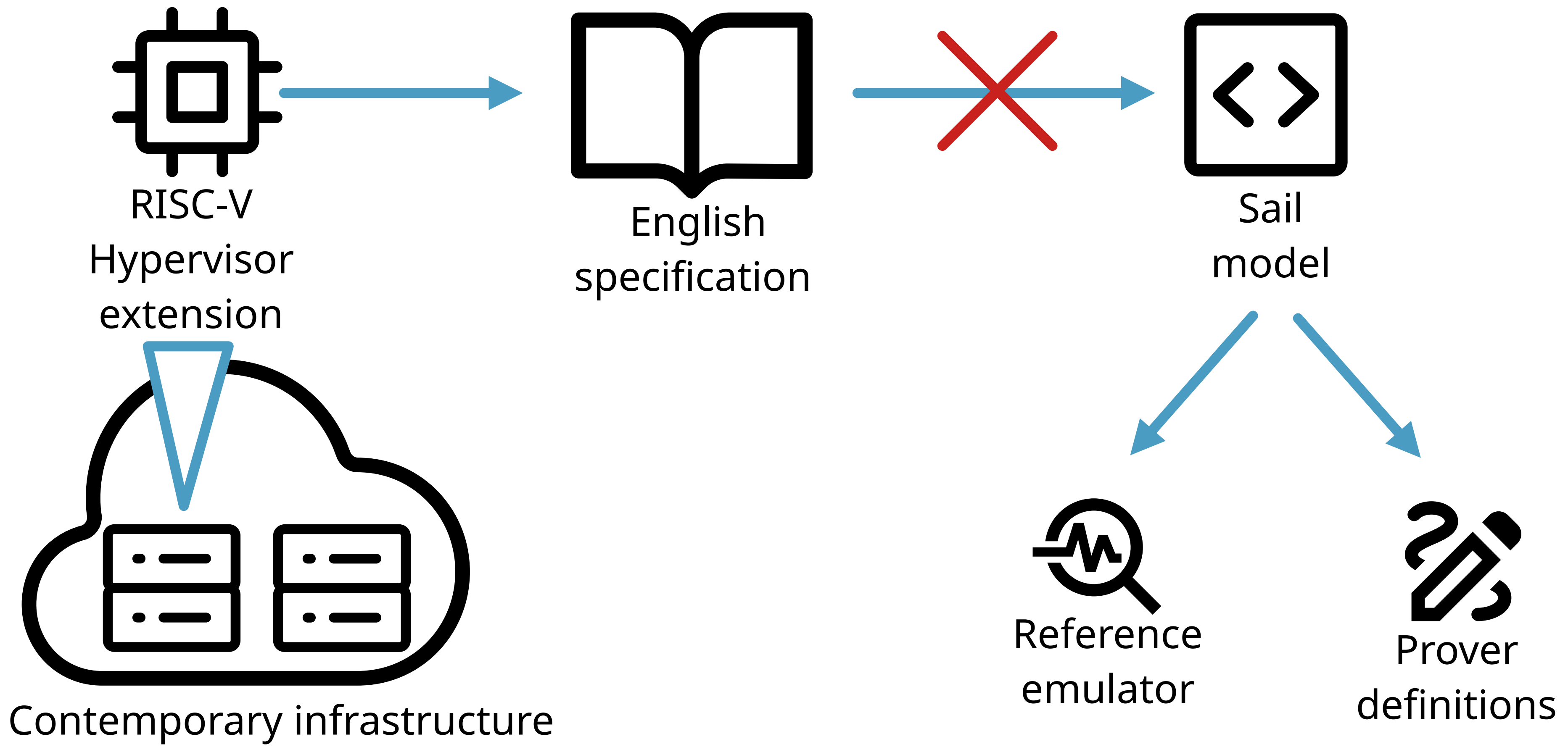
Supervisor: Dominique Devriese

DistriNet, KU Leuven, 3001 Leuven, Belgium

Funded by the European Union

European Research Council
Established by the European Commission

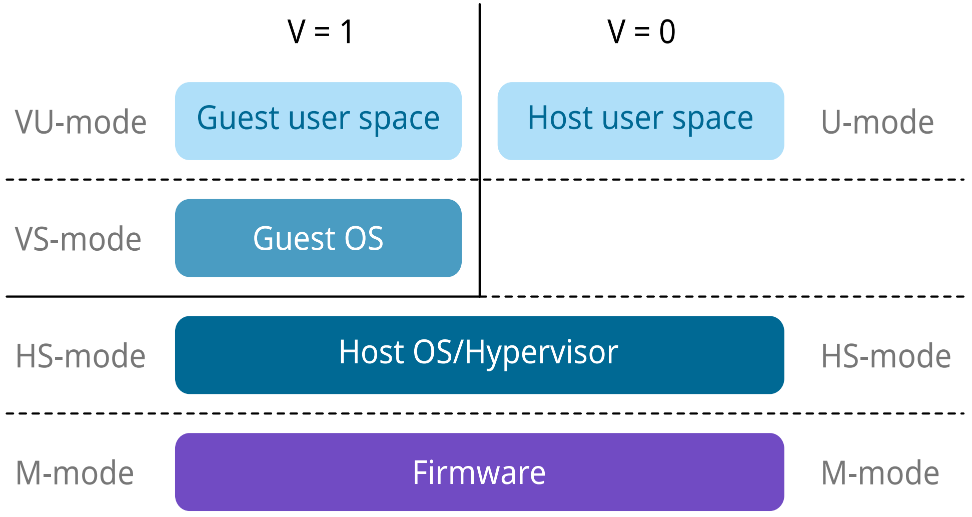

RISC-V
English specification
Sail model
Reference emulator
Prover definitions


RISC-V
Hypervisor
extension
English
specification
Sail
model
Reference
emulator
Prover
definitions
Contemporary infrastructure

RISC-V Hypervisor extension

Virtualization mode (V)

- Virtualization off (V=0)
  - Traditional privilege modes
  - S-mode → HS-mode
- Virtualization on (V=1)
  - Virtual supervisor (VS-mode)
  - Virtual user (VU-mode)



RISC-V Privilege levels [1]

[1] Sá, Bruno et al. (2021). A First Look at RISC-V Virtualization from an Embedded Systems Perspective.

Control & Status Register banking

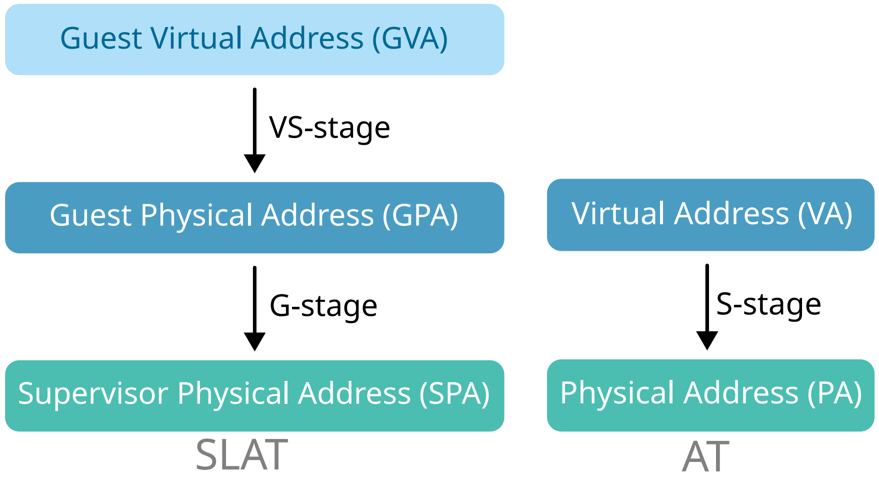
- Difference between HS- and VS-mode software?
- Banked Control & Status Registers (CSR)
- Substituted when V=1
  - VS-level software is unaware

```
# a0 holds user mode entry address
li a0, <u_entry_addr>

# Set sstatus.SPP to User (0b0)
li t0, 0x800
csrc sstatus, t0
# Set user mode entry point
csrw sepc, a0
# Return to user mode
sret
```

Two stage address translation

- Second Level Address Translation (SLAT)
  - V=1 → Enabled
  - V=0 → Dedicated instructions
- Address translation schemes
  - Different schemes for VS- and G-stage
  - Corresponding schemes have same page table entry format


Guest Virtual Address (GVA)
VS-stage
Guest Physical Address (GPA)
G-stage
Supervisor Physical Address (SPA)
SLAT
Virtual Address (VA)
S-stage
Physical Address (PA)
AT

Sail Implementation

Implementation

- Functionality (obviously)
- As modular as possible
- Minimize code churn
- Maximize readability (it’s a spec. after all)

Control and status registers

1) CSR definition

```
bitfield Vsstatus : xlenbits = {
SD : xlen - 1,
MXR : 19,
SUM : 18,
XS : 16 .. 15,
FS : 14 .. 13,
VS : 10 .. 9,
SPP : 8,
UBE : 6,
SPIE : 5,
SIE : 1
}
register vsstatus : Vsstatus
```

2) Handles for “Zicsr” instructions

```
function clause ext_read_CSR(0x200) = {
  Some(vsstatus.bits())
}

function clause ext_write_CSR(0x200, value) = {
  vsstatus = legalize_vsstatus(vsstatus, value);
  Some(vsstatus.bits())
}
```

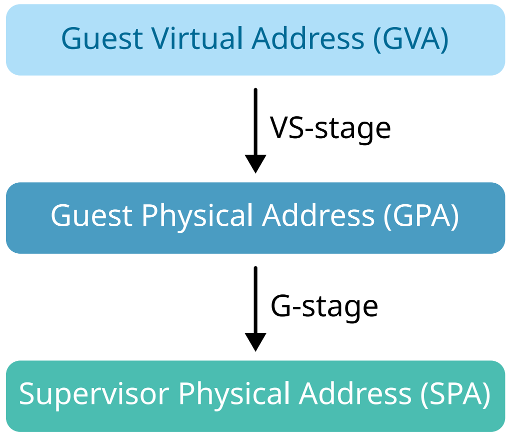
3) CSR specific logic…

- Substitute `sstatus` when V=1
- Automatically update upon trap entry/exit

DistriNet

Two-stage address translation

- Based on unified virtual memory code
  - Original had code duplication for AT schemes
  - Significantly simplifies two-stage AT implementation
  - Recently merged with the model (PR #408)
- Two-stage AT implementation
  - Parameters for new AT schemes
  - Top-level AT function delegates to stage-specific one based on privilege and virtualization mode
  - Page walk function performs G-stage AT for VS-stage page table accesses
  -


Guest Virtual Address (GVA)
VS-stage
Guest Physical Address (GPA)
G-stage
Supervisor Physical Address (SPA)

DistriNet

Evaluation

Evaluation

How to verify “golden reference” model?

- Unit tests based on prose specification
- Real-world software on extracted emulator

DistriNet

Unit tests

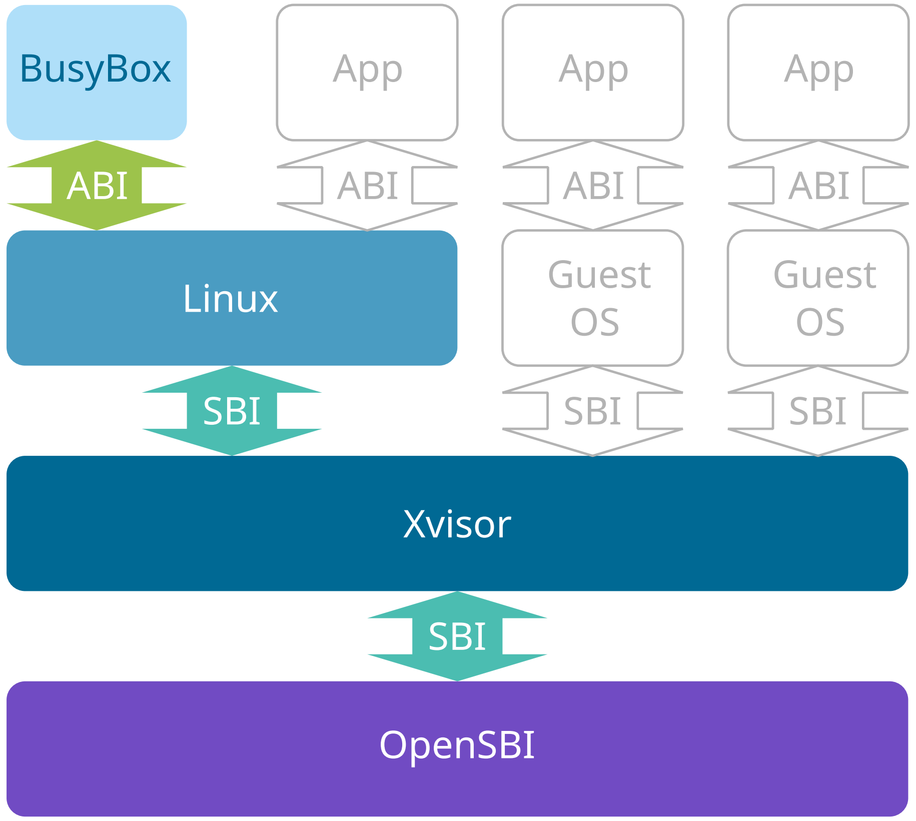
- Existing logic
  - Tests bundled with model
  - Checking for regressions
- Hypervisor extension logic
  - Developed simultaneous with model
  - Limited amount of third-party tests exist

Unit tests

- Bundled tests
  - All succeed on modified model
- Developed tests
  - Checked against Spike emulator
  - All 81 tests succeed
- Third-party tests
  - 133 out of 136 succeed
  - Two fail due to non-hypervisor extension related aspects
  - Last one is suspected error in third-party suite

Real-world software

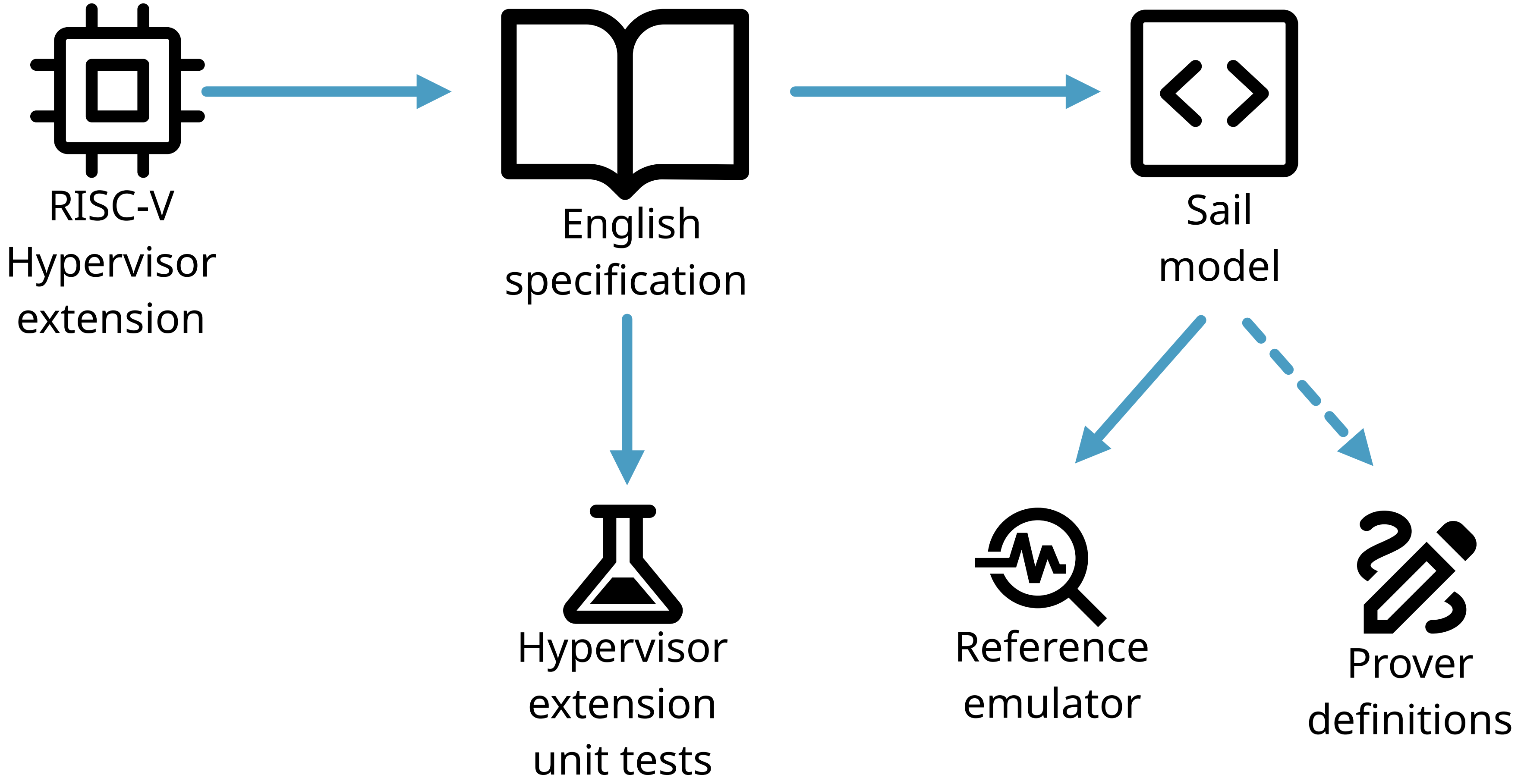
- Xvisor
  - Type I hypervisor
  - Targets SBI impl.
- Linux as guest OS
- BusyBox
  - Common UNIX utilities



DistriNet

Real-world software

- Boots on extracted emulator
  - Provides confidence in hypervisor extension model
  - Very slow (circa 13h)
- Build requires various tools, code bases, ...
  - Docker image is available:

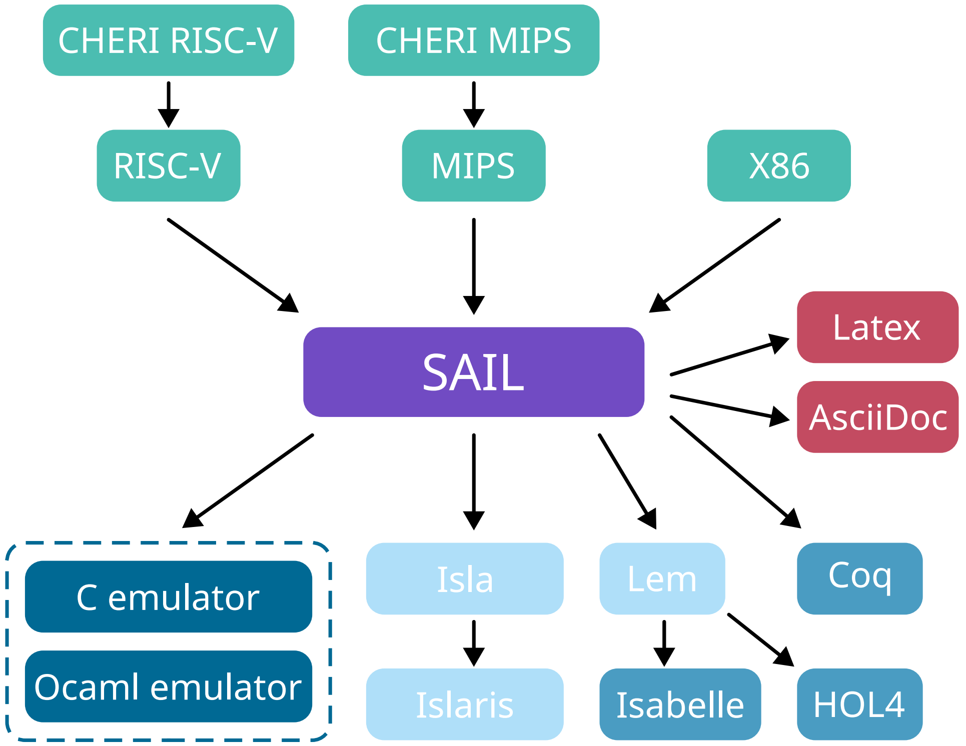

RISC-V Hypervisor extension
English specification
Sail model
Hypervisor extension unit tests
Reference emulator
Prover definitions

Questions?

This research was partially funded by the Research Fund KU Leuven, by the Flemish Research Programme Cybersecurity and by the European Union (ERC, UniversalContracts, 101040088). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

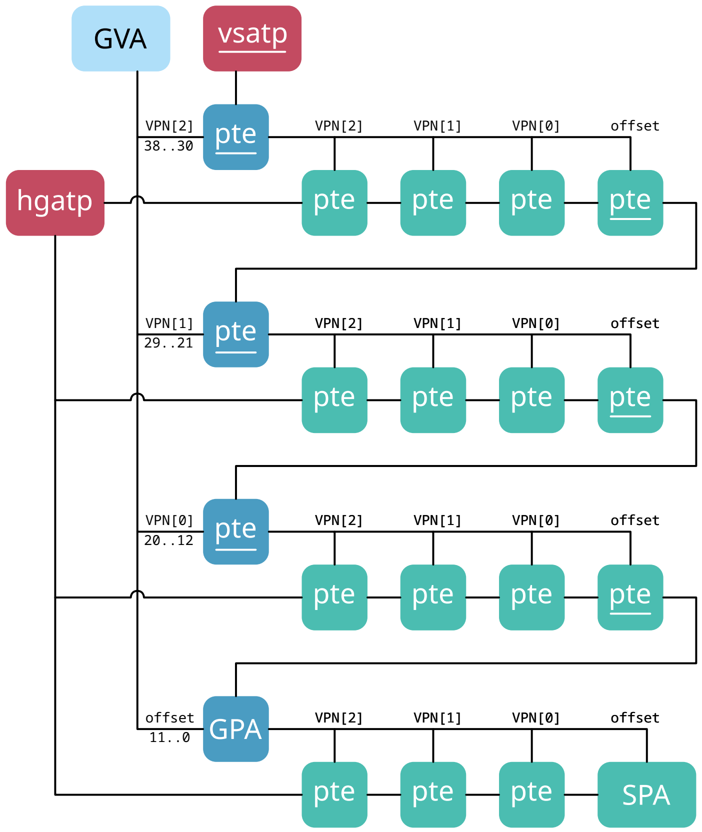
Sail ecosystem

- Functionally correct emulators
  - Tandem verification
- Definitions for theorem provers
- Symbolic execution
- Documentation snippets



Overview of the Sail ecosystem [Sail repo]

Sv39 + Sv39x4 address translation

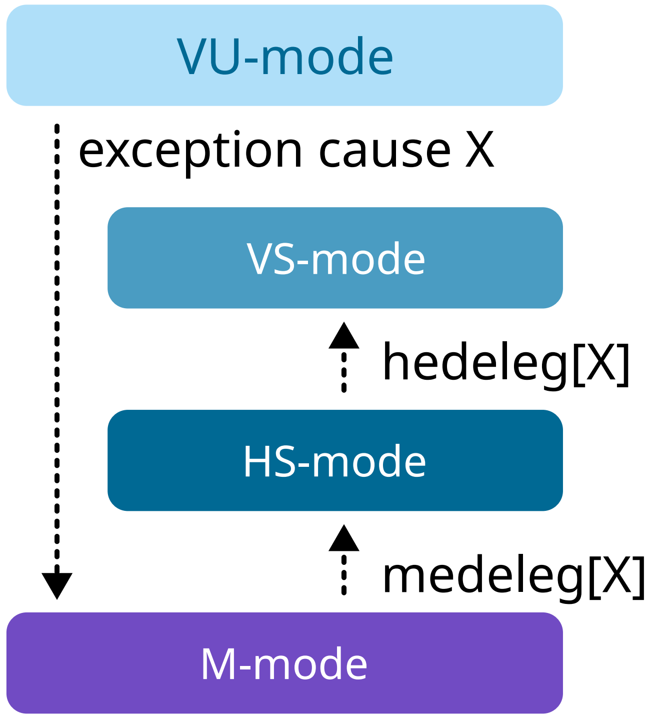


Sv39 + Sv39x4 SLAT

- Translate VS-stage PTE addresses using G-stage AT
- Several memory accesses
- Root page address
  - `vsatp` → VS-stage
  - `hgatp` → G-stage

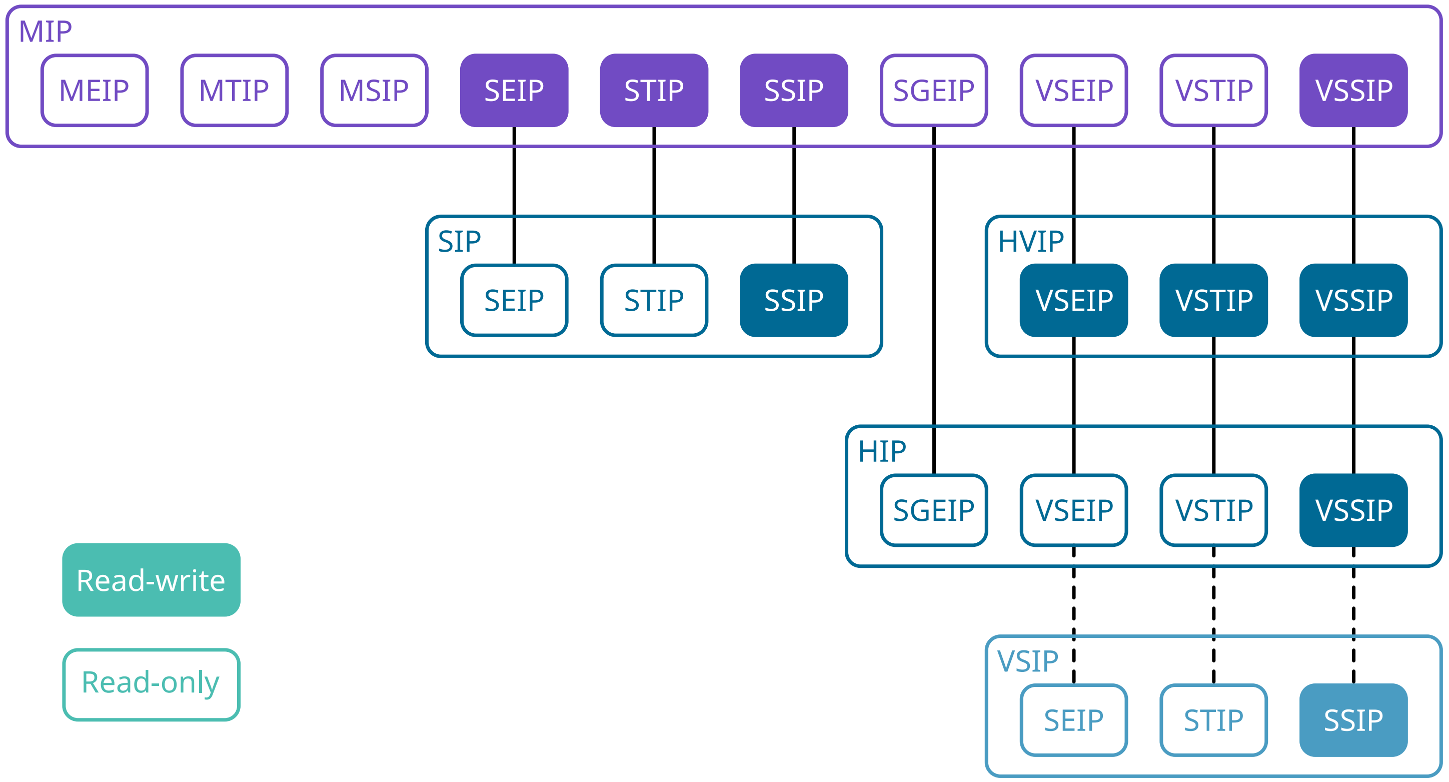
Interrupt/Exception delegation

- Traps can be delegated upto VS-mode
- Never to less privileged mode
- Exception/interrupt delegation registers



Exception delegation to VS-mode

Relation between interrupt CRSs



Read-write

Read-only

SIP, HVIP, HIP and VSIP are views of MIP in the Sail model

DistriNet