A Separation Logic for Communicating Virtual Machines

@2nd Iris Workshop

Zongyuan Liu

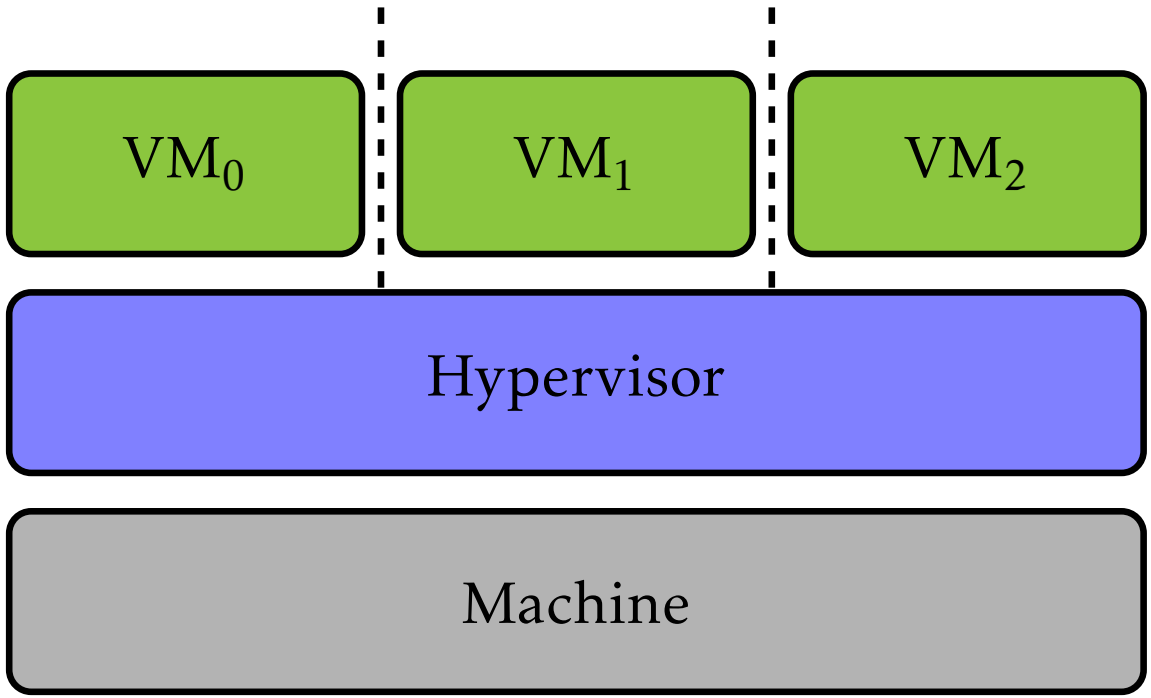
Joint work with Sergei Stepanenko, Aslan Askarov, Jean Pichon-Pharabod, Amin Timany, Lars Birkedal

Aarhus University

May 2, 2022

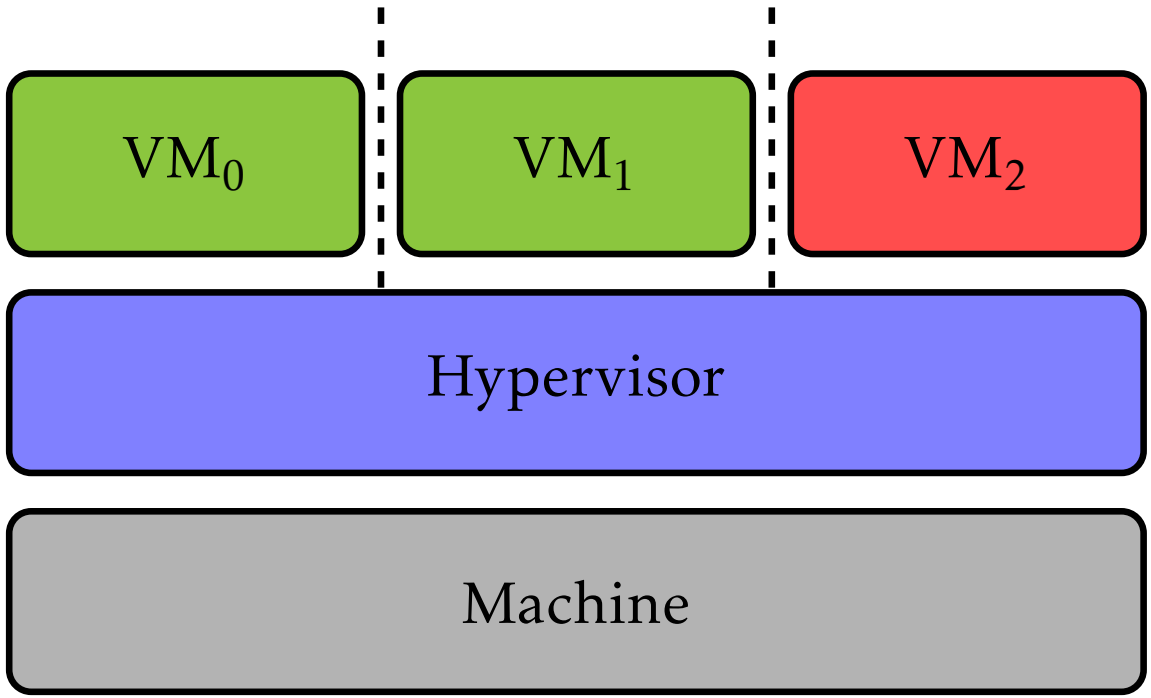
Hypervisor and Virtual Machines

- Allows one host machine to run multiple guest VMs
- Ensures VMs run as if on bare metal,
  with their own CPUs, registers, memory etc.
- Provides *isolation* between VMs
- Allows controlled communication



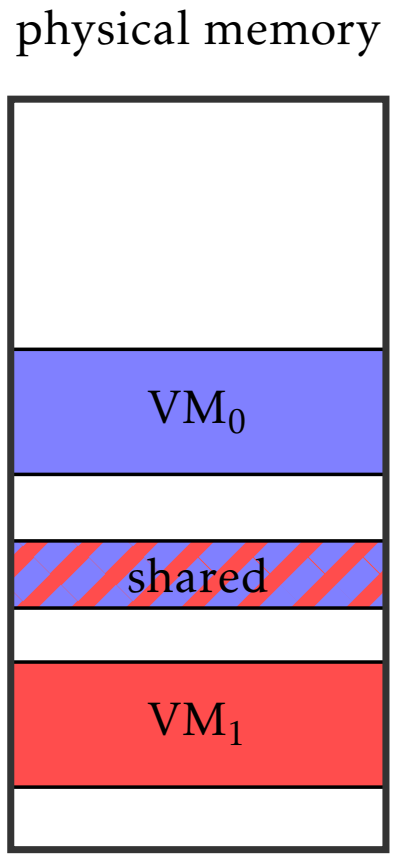
Hypervisor and Virtual Machines

- Allows one host machine to run multiple guest VMs
- Ensures VMs run as if on bare metal,
  with their own CPUs, registers, memory etc.
- Provides *isolation* between VMs
- Allows controlled communication



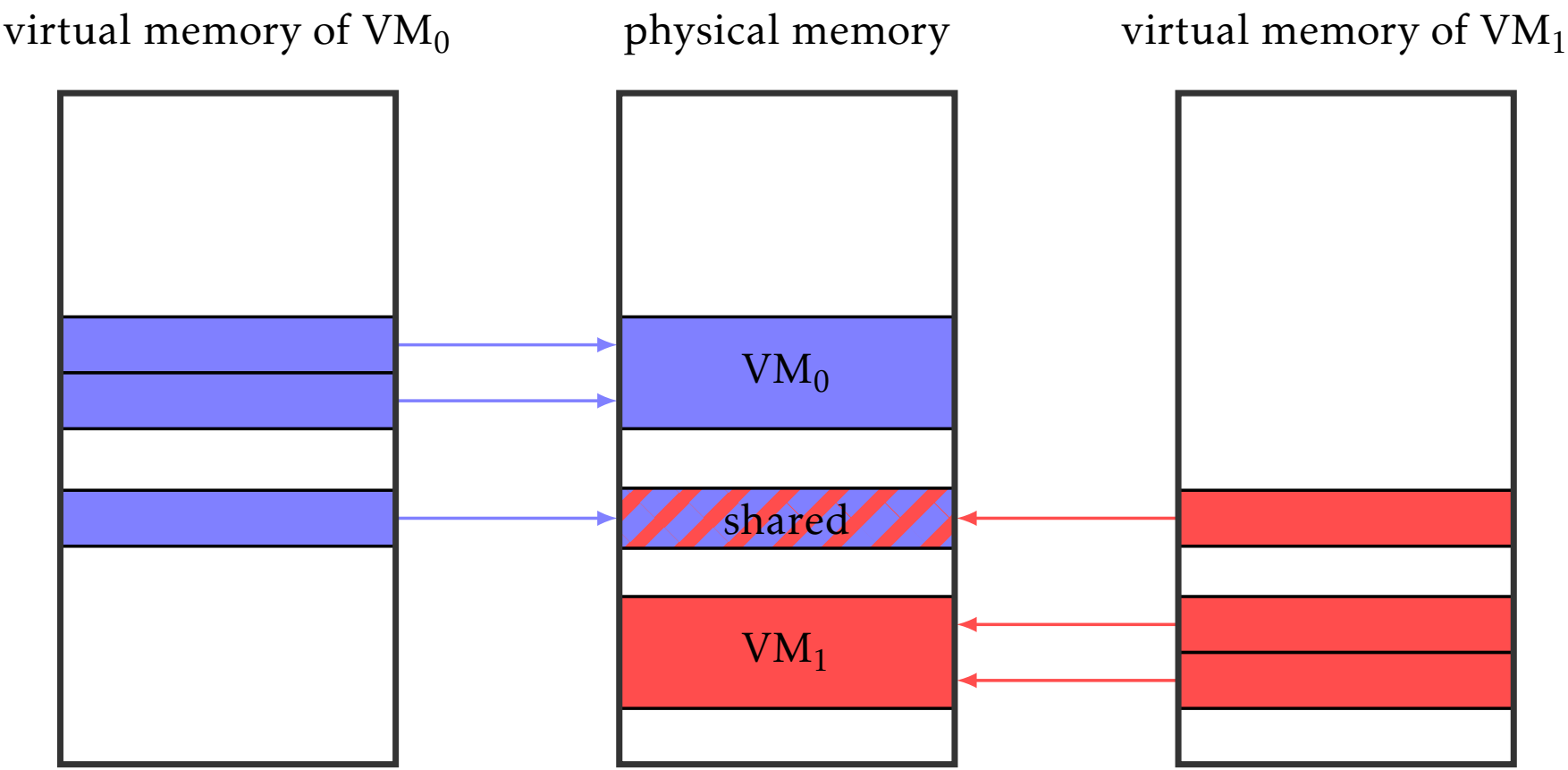
Memory Management in Hypervisors

- Controlling memory access of VMs is crucial for isolation



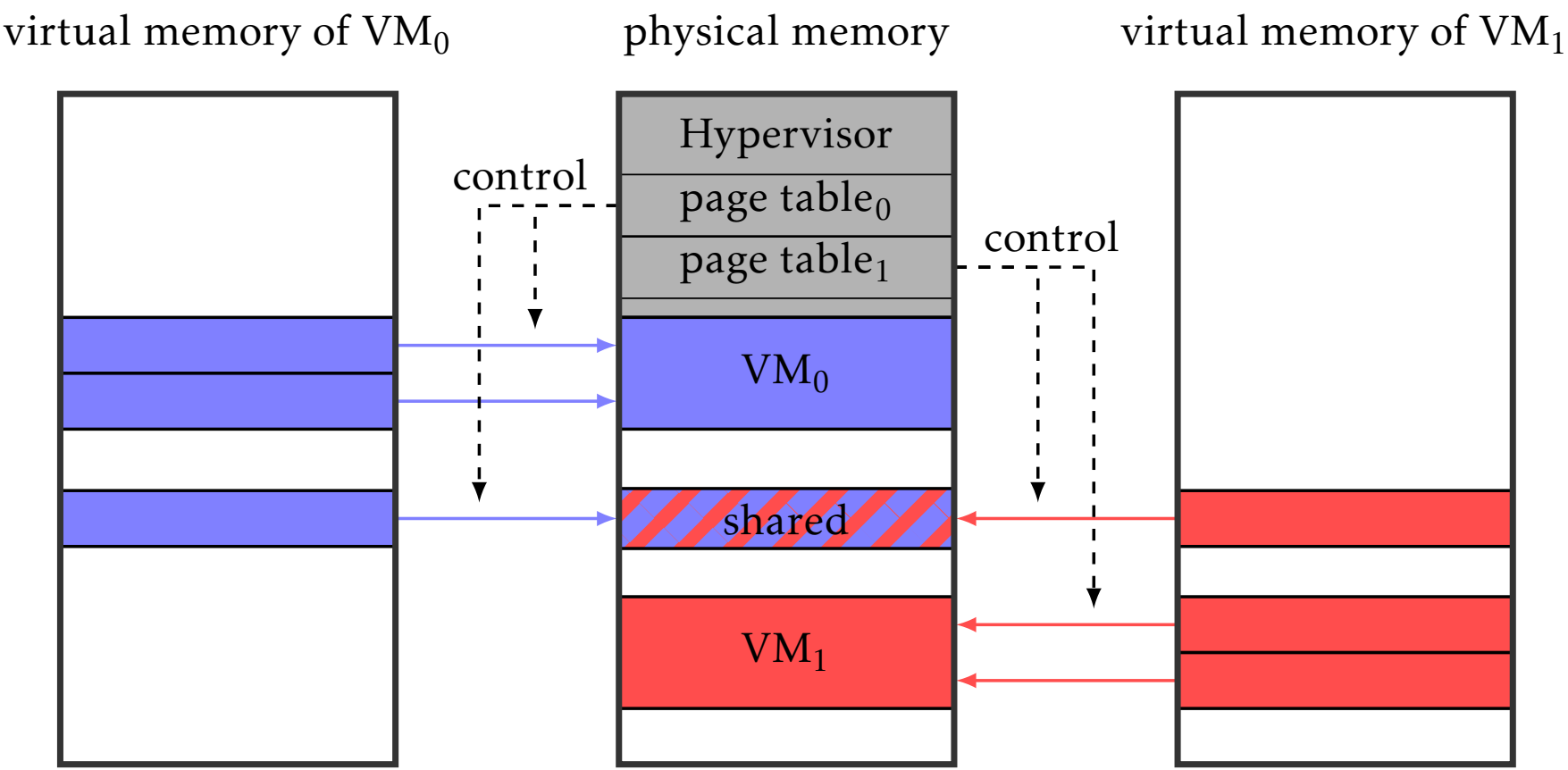
Memory Management in Hypervisors

- Controlling memory access of VMs is crucial for isolation
- Access control is implemented by address translation



Memory Management in Hypervisors

- Controlling memory access of VMs is crucial for isolation
- Access control is implemented by address translation
- Page tables of VMs are managed by the hypervisor



Verifying Communicating VMs

Separation logic nicely captures domain concepts:

Hypervisor	**Separation Logic**
Communicating VMs	Cooperative threads
Permissions: access, share, ...	Ownership
Sharing memory pages	Transferring ownership
Memory isolation	Separation
Containment of unknown code	Logical relation

Contributions

- An **operational semantics** for the combination of a machine and a hypervisor
  - With hypercalls (hvc) based on Google's Hafnium hypervisor: run, yield, share, reclaim, etc.

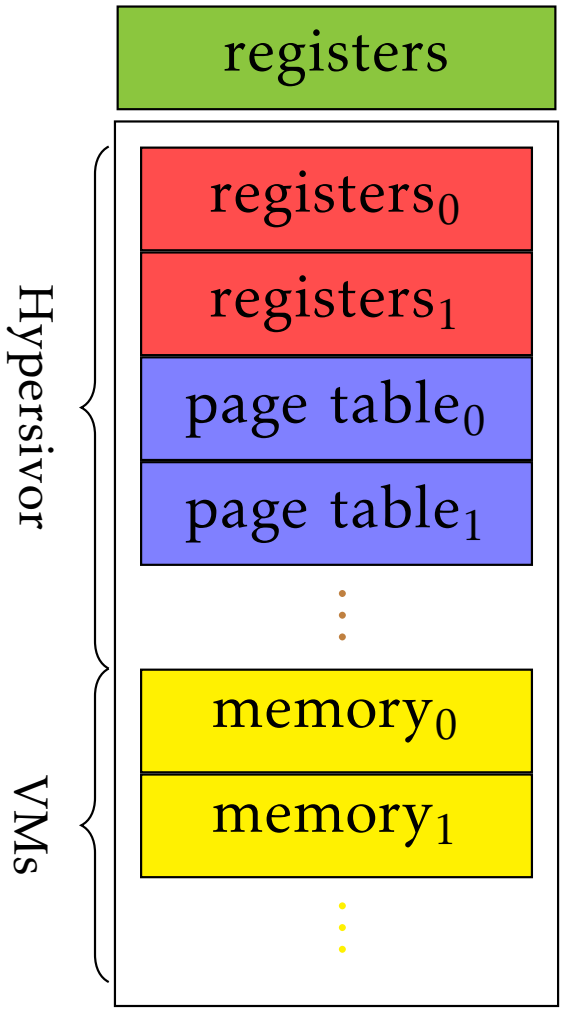
- A **program logic** for modular reasoning about VMs with communication

- A **logical relation** for robust safety property[1]
  - Allows us to verify VMs interacting with arbitrary untrusted VMs

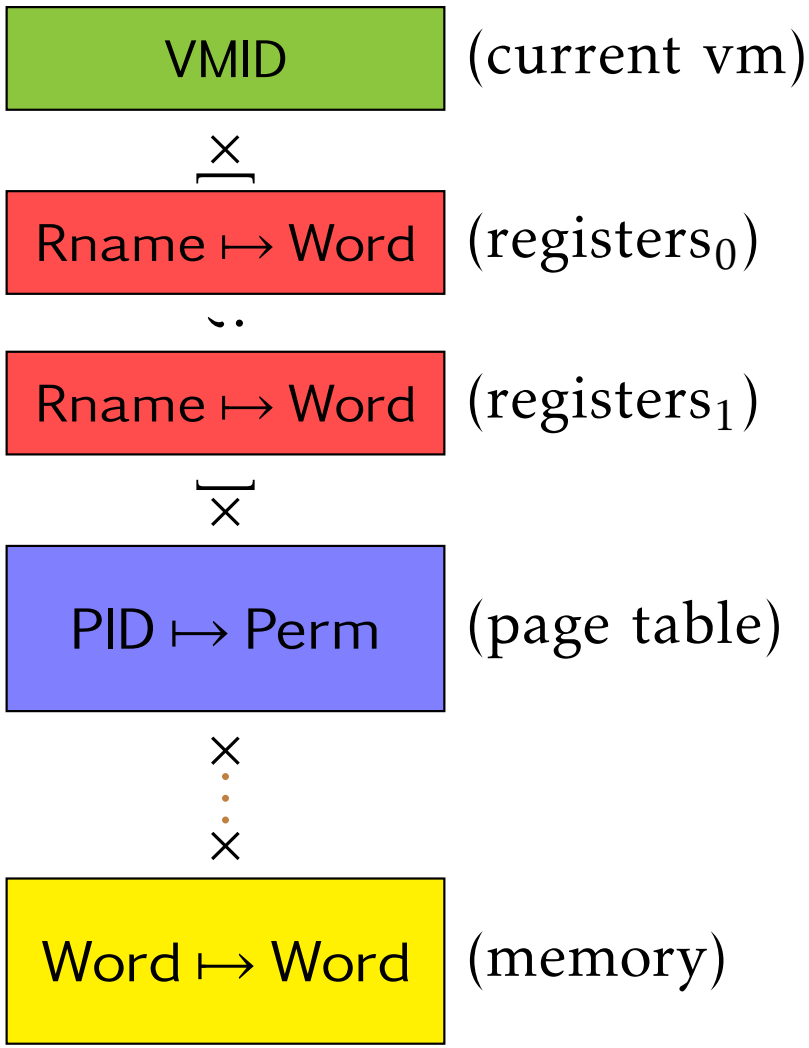
[1] [Swasey et al. 2017]

Abstraction of Real Machines

Real machines



Our abstraction



Program Logic for Reasoning about VMs

Resources for machine state

- Registers are indexed by VMID: $\mathsf{PC}\ @0 \overset{\text{reg}}{\longmapsto} a$
- Points-to for page table access: $\mathsf{Pgt}\ @0 \overset{\text{acc}}{\longmapsto} \{p\}$
- Regular points-to for memory cells: $a \overset{\text{mem}}{\longmapsto} w$

A variant of weakestpre:

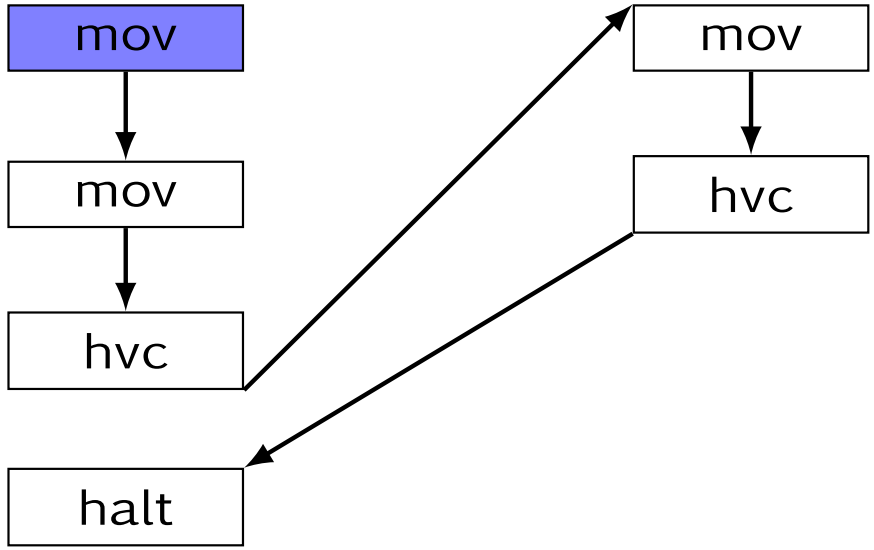
$$\mathsf{WP}\ m\ @i\ \{\Phi\}$$

$m \in \mathsf{Mode} \triangleq \texttt{ExecI} \mid \texttt{Halt} \mid \texttt{PageFault} \mid \ldots$[2] $\qquad i \in \mathsf{VMID}$

Support for partial evaluation:

$$\mathsf{WP}\ m\ @i\ \{\Phi\} \dashv\vdash \mathsf{SSWP}\ m\ @i\ \{\mathtt{m}', \mathsf{WP}\ m'\ @i\ \{\Phi\}\}$$

[2] from Cerise



VM_0 (primary)

VM_1 (secondary)

regs

$\text{PC}\ @0 \overset{\text{reg}}{\longmapsto} a_{pc0} + 0$

$* \text{R0}\ @0 \overset{\text{reg}}{\longmapsto} -$

$* \text{R1}\ @0 \overset{\text{reg}}{\longmapsto} -$

program

$* a_{pc0} + 0 \overset{\text{mem}}{\longmapsto} \text{mov R0 Run}$

$* a_{pc0} + 1 \overset{\text{mem}}{\longmapsto} \text{mov R1 1}$

$* a_{pc0} + 2 \overset{\text{mem}}{\longmapsto} \text{hvc}$

$* a_{pc0} + 3 \overset{\text{mem}}{\longmapsto} \text{halt}$

pgt

$* \text{Pgt}\ @0 \overset{\text{acc}}{\longmapsto} \{\text{pid}(a_{pc0})\}$

$\text{WP ExecI}\ @0\ \{m, \ldots\}$

$\text{PC}\ @1 \overset{\text{reg}}{\longmapsto} a_{pc1} + 0$

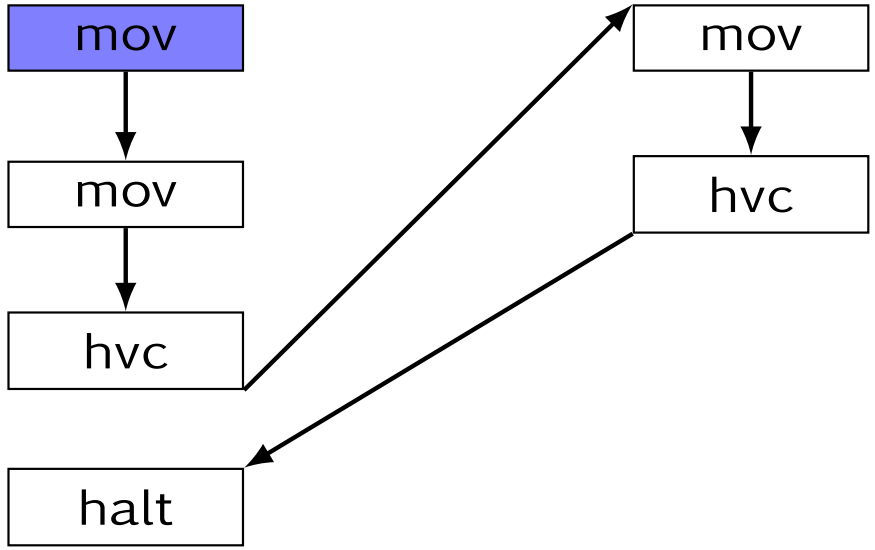
$* \text{R0}\ @1 \overset{\text{reg}}{\longmapsto} -$

$* a_{pc1} + 0 \overset{\text{mem}}{\longmapsto} \text{mov R0 Yield}$

$* a_{pc1} + 1 \overset{\text{mem}}{\longmapsto} \text{hvc}$

$* \text{Pgt}\ @1 \overset{\text{acc}}{\longmapsto} \{\text{pid}(a_{pc1})\}$

$\text{WP ExecI}\ @1\ \{m, \ldots\}$



PC @0 $\overset{\text{reg}}{\longmapsto} a_{pc0} + 0$

$*$ R0 @0 $\overset{\text{reg}}{\longmapsto} -$

$*$ R1 @0 $\overset{\text{reg}}{\longmapsto} -$

$*$ $a_{pc0} + 0 \overset{\text{mem}}{\longmapsto}$ mov R0 Run

$*$ $a_{pc0} + 1 \overset{\text{mem}}{\longmapsto}$ mov R1 1

$*$ $a_{pc0} + 2 \overset{\text{mem}}{\longmapsto}$ hvc

$*$ $a_{pc0} + 3 \overset{\text{mem}}{\longmapsto}$ halt

$*$ Pgt @0 $\overset{\text{acc}}{\longmapsto} \{\text{pid}(a_{pc0})\}$

PC @1 $\overset{\text{reg}}{\longmapsto} a_{pc1} + 0$

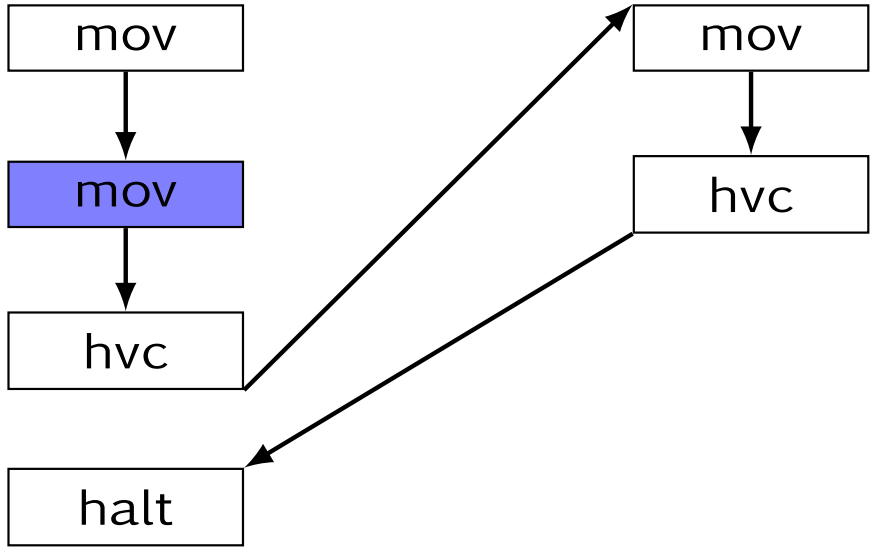
$*$ R0 @1 $\overset{\text{reg}}{\longmapsto} -$

$*$ $a_{pc1} + 0 \overset{\text{mem}}{\longmapsto}$ mov R0 Yield

$*$ $a_{pc1} + 1 \overset{\text{mem}}{\longmapsto}$ hvc

$*$ Pgt @1 $\overset{\text{acc}}{\longmapsto} \{\text{pid}(a_{pc1})\}$

SSWP ExecI @0 $\{m, \text{WP } m \text{ @0 } \{\ldots\}\}$ WP ExecI @1 $\{m, \ldots\}$



VM_0 (primary)

VM_1 (secondary)

$\mathrm{PC}\ @0 \overset{\mathrm{reg}}{\longmapsto} a_{pc0} + 1$

$* \ \mathrm{R0}\ @0 \overset{\mathrm{reg}}{\longmapsto} \mathit{Run}$

$* \ \mathrm{R1}\ @0 \overset{\mathrm{reg}}{\longmapsto} -$

$* \ a_{pc0} + 0 \overset{\mathrm{mem}}{\longmapsto} \mathrm{mov\ R0\ Run}$

$* \ a_{pc0} + 1 \overset{\mathrm{mem}}{\longmapsto} \mathrm{mov\ R1\ 1}$

$* \ a_{pc0} + 2 \overset{\mathrm{mem}}{\longmapsto} \mathrm{hvc}$

$* \ a_{pc0} + 3 \overset{\mathrm{mem}}{\longmapsto} \mathrm{halt}$

$* \ \mathrm{Pgt}\ @0 \overset{\mathrm{acc}}{\longmapsto} \{\mathrm{pid}(a_{pc0})\}$

$\mathrm{WP\ ExecI}\ @0\ \{m, \ldots\}$

$\mathrm{PC}\ @1 \overset{\mathrm{reg}}{\longmapsto} a_{pc1} + 0$

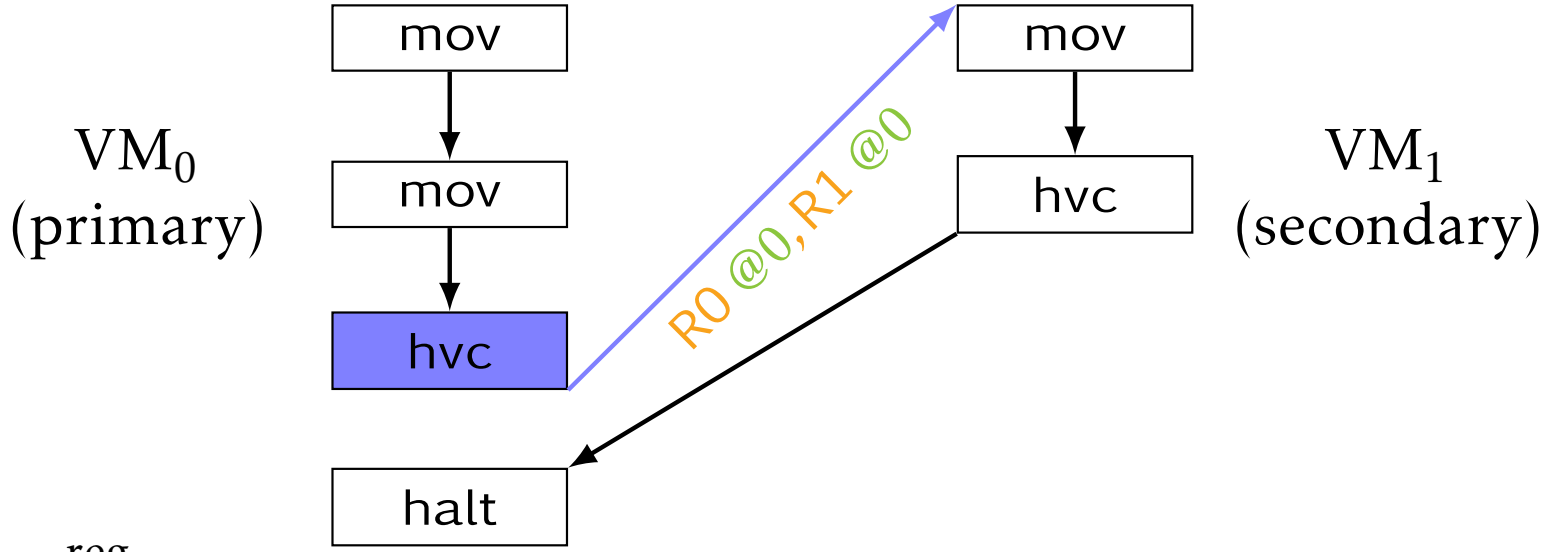
$* \ \mathrm{R0}\ @1 \overset{\mathrm{reg}}{\longmapsto} -$

$* \ a_{pc1} + 0 \overset{\mathrm{mem}}{\longmapsto} \mathrm{mov\ R0\ Yield}$

$* \ a_{pc1} + 1 \overset{\mathrm{mem}}{\longmapsto} \mathrm{hvc}$

$* \ \mathrm{Pgt}\ @1 \overset{\mathrm{acc}}{\longmapsto} \{\mathrm{pid}(a_{pc1})\}$

$\mathrm{WP\ ExecI}\ @1\ \{m, \ldots\}$



$\text{PC } @0 \overset{\text{reg}}{\longmapsto} a_{pc0} + 2$

$* \text{ R0 } @0 \overset{\text{reg}}{\longmapsto} \mathit{Run}$

$* \text{ R1 } @0 \overset{\text{reg}}{\longmapsto} 1$

$* \; a_{pc0} + 0 \overset{\text{mem}}{\longmapsto} \text{mov R0 Run}$

$* \; a_{pc0} + 1 \overset{\text{mem}}{\longmapsto} \text{mov R1 1}$

$* \; a_{pc0} + 2 \overset{\text{mem}}{\longmapsto} \text{hvc}$

$* \; a_{pc0} + 3 \overset{\text{mem}}{\longmapsto} \text{halt}$

$* \text{ Pgt } @0 \overset{\text{acc}}{\longmapsto} \{\text{pid}(a_{pc0})\}$

WP ExecI @0 $\{m, \ldots\}$

$\text{PC } @1 \overset{\text{reg}}{\longmapsto} a_{pc1} + 0$

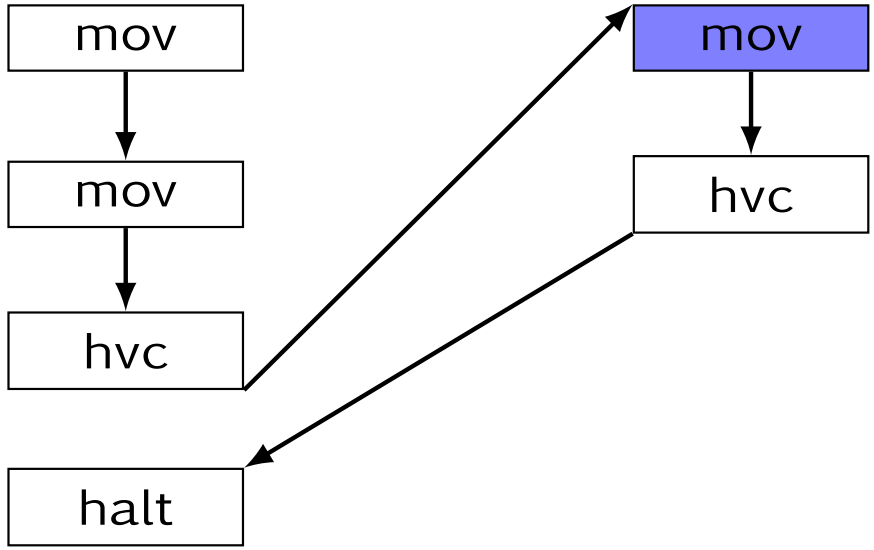
$* \text{ R0 } @1 \overset{\text{reg}}{\longmapsto} -$

$* \; a_{pc1} + 0 \overset{\text{mem}}{\longmapsto} \text{mov R0 Yield}$

$* \; a_{pc1} + 1 \overset{\text{mem}}{\longmapsto} \text{hvc}$

$* \text{ Pgt } @1 \overset{\text{acc}}{\longmapsto} \{\text{pid}(a_{pc1})\}$

WP ExecI @1 $\{m, \ldots\}$



VM_0 (primary)

VM_1 (secondary)

PC @0 $\overset{\text{reg}}{\longmapsto} a_{pc0} + 3$

$* a_{pc0} + 0 \overset{\text{mem}}{\longmapsto}$ mov R0 Run

$* a_{pc0} + 1 \overset{\text{mem}}{\longmapsto}$ mov R1 1

$* a_{pc0} + 2 \overset{\text{mem}}{\longmapsto}$ hvc

$* a_{pc0} + 3 \overset{\text{mem}}{\longmapsto}$ halt

$*$ Pgt @0 $\overset{\text{acc}}{\longmapsto} \{\text{pid}(a_{pc0})\}$

WP ExecI @0 $\{m, \ldots\}$

PC @1 $\overset{\text{reg}}{\longmapsto} a_{pc1} + 0$

$*$ R0 @1 $\overset{\text{reg}}{\longmapsto} -$

$*$ R0 @0 $\overset{\text{reg}}{\longmapsto} \mathit{Run}$

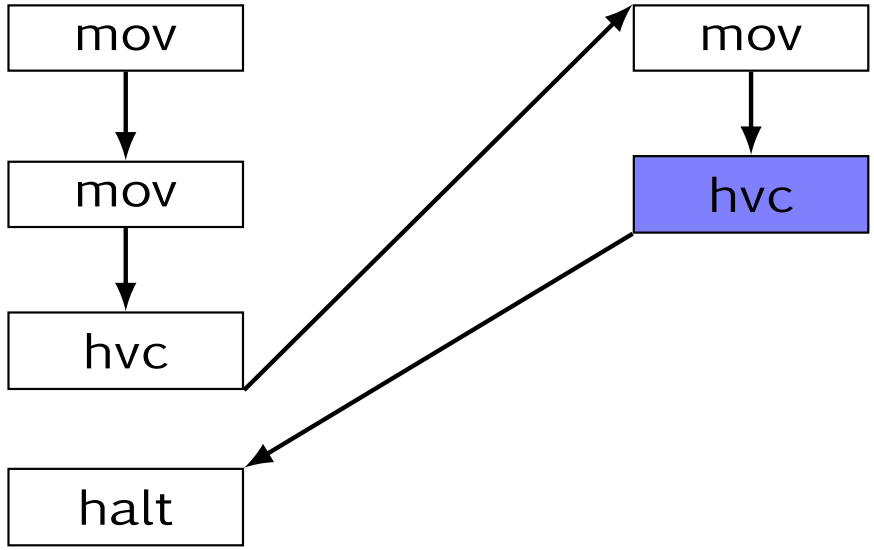
$*$ R1 @0 $\overset{\text{reg}}{\longmapsto} 1$

$* a_{pc1} + 0 \overset{\text{mem}}{\longmapsto}$ mov R0 Yield

$* a_{pc1} + 1 \overset{\text{mem}}{\longmapsto}$ hvc

$*$ Pgt @1 $\overset{\text{acc}}{\longmapsto} \{\text{pid}(a_{pc1})\}$

WP ExecI @1 $\{m, \ldots\}$



VM_0 (primary)

VM_1 (secondary)

PC @0 $\overset{\text{reg}}{\longmapsto} a_{pc0} + 3$

$* a_{pc0} + 0 \overset{\text{mem}}{\longmapsto}$ mov R0 Run

$* a_{pc0} + 1 \overset{\text{mem}}{\longmapsto}$ mov R1 1

$* a_{pc0} + 2 \overset{\text{mem}}{\longmapsto}$ hvc

$* a_{pc0} + 3 \overset{\text{mem}}{\longmapsto}$ halt

$* \text{Pgt @0} \overset{\text{acc}}{\longmapsto} \{\text{pid}(a_{pc0})\}$

WP ExecI @0 $\{m, \ldots\}$

PC @1 $\overset{\text{reg}}{\longmapsto} a_{pc1} + 1$

$* \text{R0 @1} \overset{\text{reg}}{\longmapsto} \mathit{Yield}$

$* \text{R0 @0} \overset{\text{reg}}{\longmapsto} \mathit{Run}$

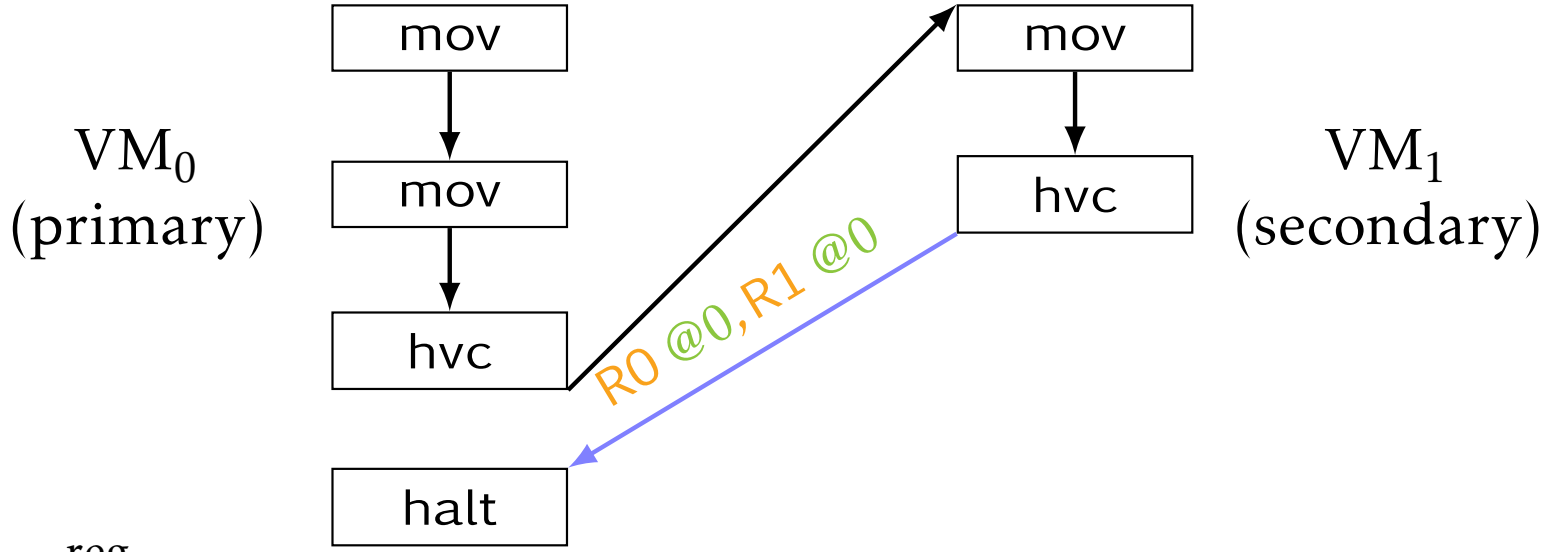
$* \text{R1 @0} \overset{\text{reg}}{\longmapsto} 1$

$* a_{pc1} + 0 \overset{\text{mem}}{\longmapsto}$ mov R0 Yield

$* a_{pc1} + 1 \overset{\text{mem}}{\longmapsto}$ hvc

$* \text{Pgt @1} \overset{\text{acc}}{\longmapsto} \{\text{pid}(a_{pc1})\}$

WP ExecI @1 $\{m, \ldots\}$



$\text{PC } @0 \overset{\text{reg}}{\longmapsto} a_{pc0} + 3$

$* a_{pc0} + 0 \overset{\text{mem}}{\longmapsto} \text{mov R0 Run}$

$* a_{pc0} + 1 \overset{\text{mem}}{\longmapsto} \text{mov R1 1}$

$* a_{pc0} + 2 \overset{\text{mem}}{\longmapsto} \text{hvc}$

$* a_{pc0} + 3 \overset{\text{mem}}{\longmapsto} \text{halt}$

$* \text{Pgt } @0 \overset{\text{acc}}{\longmapsto} \{\text{pid}(a_{pc0})\}$

WP ExecI @0 $\{m, \ldots\}$

$\text{PC } @1 \overset{\text{reg}}{\longmapsto} a_{pc1} + 2$

$* \text{R0 } @1 \overset{\text{reg}}{\longmapsto} Yield$

$* \text{R0 } @0 \overset{\text{reg}}{\longmapsto} Yield$

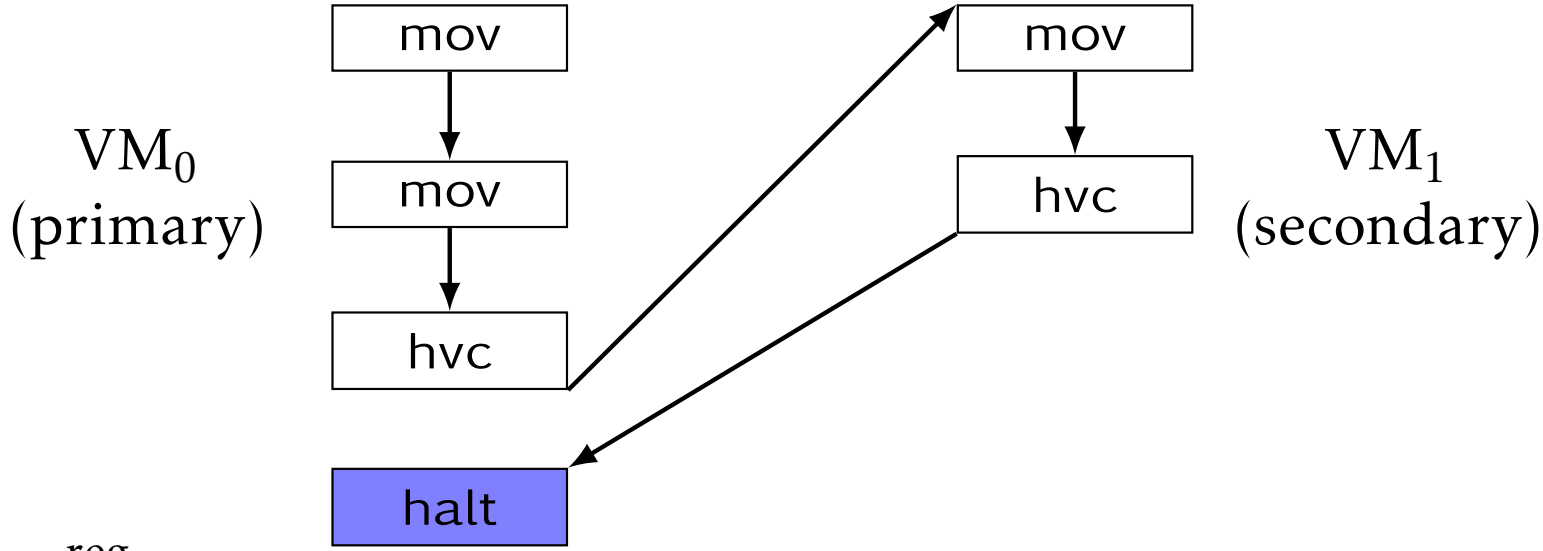
$* \text{R1 } @0 \overset{\text{reg}}{\longmapsto} 1$

$* a_{pc1} + 0 \overset{\text{mem}}{\longmapsto} \text{mov R0 Yield}$

$* a_{pc1} + 1 \overset{\text{mem}}{\longmapsto} \text{hvc}$

$* \text{Pgt } @1 \overset{\text{acc}}{\longmapsto} \{\text{pid}(a_{pc1})\}$

WP ExecI @1 $\{m, \ldots\}$



$\text{PC}\ @0 \overset{\text{reg}}{\longmapsto} a_{pc0} + 3$

$* \ \text{R0}\ @0 \overset{\text{reg}}{\longmapsto} Yield$

$* \ \text{R1}\ @0 \overset{\text{reg}}{\longmapsto} 1$

$* \ a_{pc0} + 0 \overset{\text{mem}}{\longmapsto} \text{mov R0 Run}$

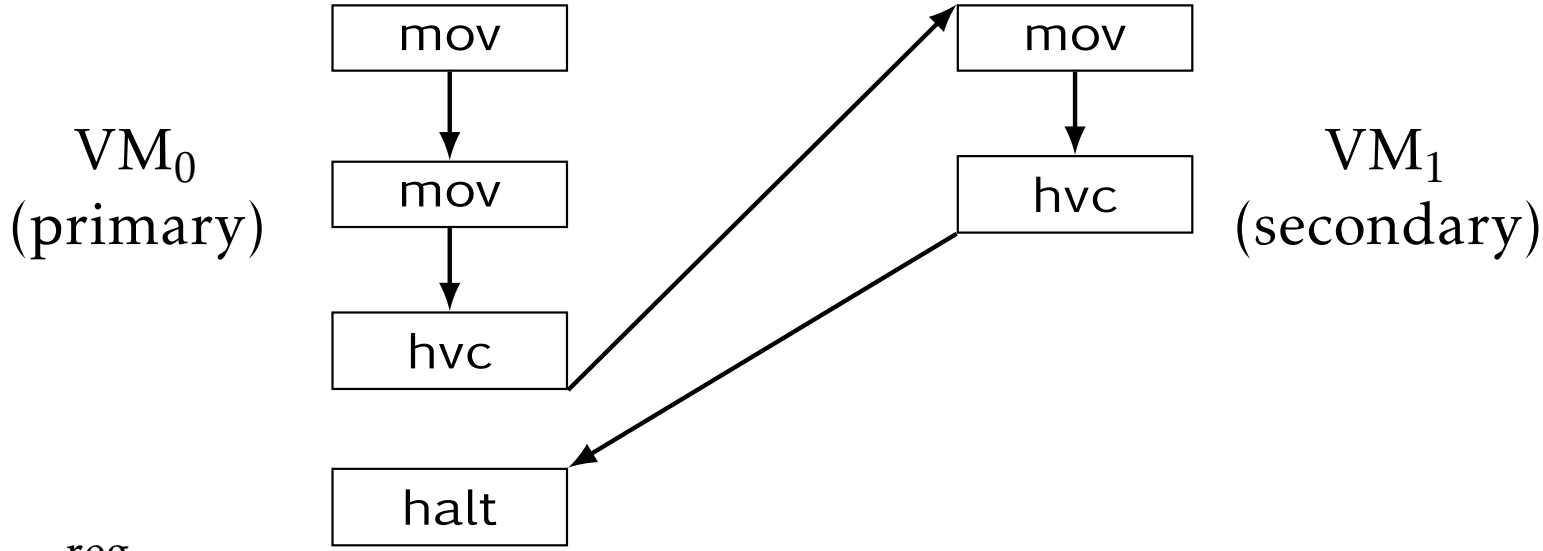
$* \ a_{pc0} + 1 \overset{\text{mem}}{\longmapsto} \text{mov R1 1}$

$* \ a_{pc0} + 2 \overset{\text{mem}}{\longmapsto} \text{hvc}$

$* \ a_{pc0} + 3 \overset{\text{mem}}{\longmapsto} \text{halt}$

$* \ \text{Pgt}\ @0 \overset{\text{acc}}{\longmapsto} \{\text{pid}(a_{pc0})\}$

WP ExecI @0 $\{m,\ldots\}$



$$\text{PC } @0 \overset{\text{reg}}{\longmapsto} a_{pc0} + 4$$

$$* \text{ R0 } @0 \overset{\text{reg}}{\longmapsto} Yield$$

$$* \text{ R1 } @0 \overset{\text{reg}}{\longmapsto} 1$$

$$* \; a_{pc0} + 0 \overset{\text{mem}}{\longmapsto} \text{mov R0 Run}$$

$$* \; a_{pc0} + 1 \overset{\text{mem}}{\longmapsto} \text{mov R1 1}$$

$$* \; a_{pc0} + 2 \overset{\text{mem}}{\longmapsto} \text{hvc}$$

$$* \; a_{pc0} + 3 \overset{\text{mem}}{\longmapsto} \text{halt}$$

$$* \text{ Pgt } @0 \overset{\text{acc}}{\longmapsto} \{\text{pid}(a_{pc0})\}$$

WP `Halt` @0 $\{m, \ldots\}$

Resumption Conditions - Session-type Like Protocols

Allow us to specify what is needed to resume the execution of a VM, and transfer resources accordingly

$$\mathsf{ResumCond}\ @1\ 1/2 \left\{ \begin{array}{l} \mathsf{R0}\ @0 \overset{\mathrm{reg}}{\longmapsto} \mathit{Run} \\ * \mathsf{R1}\ @0 \overset{\mathrm{reg}}{\longmapsto} 1 \\ \\ * \mathsf{ResumCond}\ @0\ 1/2 \left\{ \begin{array}{l} \mathsf{R0}\ @0 \overset{\mathrm{reg}}{\longmapsto} \mathit{Yield} \\ * \mathsf{R1}\ @0 \overset{\mathrm{reg}}{\longmapsto} 1 \end{array} \right\} \end{array} \right\}$$

$$* \mathsf{ResumCond}\ @0\ 1 \left\{ \begin{array}{l} \mathsf{R0}\ @0 \overset{\mathrm{reg}}{\longmapsto} \mathit{Yield} \\ * \mathsf{R1}\ @0 \overset{\mathrm{reg}}{\longmapsto} 1 \end{array} \right\}$$

Resumption Conditions - Session-type Like Protocols

$$\text{ResumCond } @1 \; 1/2 \left\{ \begin{array}{l} \text{R0 } @0 \overset{\text{reg}}{\longmapsto} Run \\ * \text{ R1 } @0 \overset{\text{reg}}{\longmapsto} 1 \\ * \text{ ResumCond } @0 \; 1/2 \left\{ \begin{array}{l} \text{R0 } @0 \overset{\text{reg}}{\longmapsto} Yield \\ * \text{ R1 } @0 \overset{\text{reg}}{\longmapsto} 1 \end{array} \right\} \end{array} \right\}$$

$$* \text{ ResumCond } @0 \; 1 \left\{ \begin{array}{l} \text{R0 } @0 \overset{\text{reg}}{\longmapsto} Yield \\ * \text{ R1 } @0 \overset{\text{reg}}{\longmapsto} 1 \end{array} \right\}$$

$$* \text{ R0 } @0 \overset{\text{reg}}{\longmapsto} Run$$

$$* \text{ R1 } @0 \overset{\text{reg}}{\longmapsto} 1$$

$$* \; a_{pc0} + 0 \overset{\text{mem}}{\longmapsto} \text{mov R0 Run}$$

$$* \; a_{pc0} + 1 \overset{\text{mem}}{\longmapsto} \text{mov R1 1}$$

$$* \; a_{pc0} + 2 \overset{\text{mem}}{\longmapsto} \text{hvc}$$

$$* \; a_{pc0} + 3 \overset{\text{mem}}{\longmapsto} \text{halt}$$

$$\text{WP ExecI } @0 \; \{m, \ldots\}$$

Resumption Conditions - Session-type Like Protocols

$$\text{ResumCond } @0 \ 1/2 \ \left\{ \begin{array}{l} \text{R0 } @0 \overset{\text{reg}}{\longmapsto} Yield \\ * \ \text{R1 } @0 \overset{\text{reg}}{\longmapsto} 1 \end{array} \right\}$$

$$\frac{\begin{array}{l} * \ a_{pc0} + 0 \overset{\text{mem}}{\longmapsto} \text{mov R0 Run} \\ * \ a_{pc0} + 1 \overset{\text{mem}}{\longmapsto} \text{mov R1 1} \\ * \ a_{pc0} + 2 \overset{\text{mem}}{\longmapsto} \text{hvc} \\ * \ a_{pc0} + 3 \overset{\text{mem}}{\longmapsto} \text{halt} \end{array}}{\text{ResumCondHolds } @0 \twoheadrightarrow \text{WP ExecI } @0 \ \{m, \ldots\}}$$

$$\frac{\text{ResumCondHolds } @i * \text{ResumCond } @i \ 1/2 \ \{P\}}{\text{ResumCond } @i \ 1 \ \{P\} * \triangleright P}$$

Resumption Conditions - Session-type Like Protocols

$$\mathsf{ResumCond}\ @0\ 1\ \left\{\begin{array}{l}\mathsf{R0}\ @0 \overset{\mathrm{reg}}{\longmapsto} Yield \\ * \mathsf{R1}\ @0 \overset{\mathrm{reg}}{\longmapsto} 1\end{array}\right\}$$

$$* \mathsf{R0}\ @0 \overset{\mathrm{reg}}{\longmapsto} Yield$$

$$* \mathsf{R1}\ @0 \overset{\mathrm{reg}}{\longmapsto} 1$$

$$* a_{pc0} + 0 \overset{\mathrm{mem}}{\longmapsto} \mathsf{mov\ R0\ Run}$$

$$* a_{pc0} + 1 \overset{\mathrm{mem}}{\longmapsto} \mathsf{mov\ R1\ 1}$$

$$* a_{pc0} + 2 \overset{\mathrm{mem}}{\longmapsto} \mathsf{hvc}$$

$$* a_{pc0} + 3 \overset{\mathrm{mem}}{\longmapsto} \mathsf{halt}$$

$$\mathsf{WP\ ExecI}\ @0\ \{m, \ldots\}$$

$$\frac{\mathsf{ResumCondHolds}\ @i * \mathsf{ResumCond}\ @i\ ½\ \{P\}}{\mathsf{ResumCond}\ @i\ 1\ \{P\} * \triangleright P}$$

Resumption Conditions - Session-type Like Protocols

$$\mathsf{ResumCond}\ @1\ 1/2\ \left\{\begin{array}{l} \mathsf{R0}\ @0 \overset{\mathrm{reg}}{\longmapsto} Run \\ * \mathsf{R1}\ @0 \overset{\mathrm{reg}}{\longmapsto} 1 \\ \\ * \mathsf{ResumCond}\ @0\ 1/2\ \left\{\begin{array}{l} \mathsf{R0}\ @0 \overset{\mathrm{reg}}{\longmapsto} Yield \\ * \mathsf{R1}\ @0 \overset{\mathrm{reg}}{\longmapsto} 1 \end{array}\right\} \end{array}\right\}$$

$$\begin{array}{l} * \mathsf{R0}\ @1 \overset{\mathrm{reg}}{\longmapsto} - \\ * a_{pc0} + 0 \overset{\mathrm{mem}}{\longmapsto} \mathsf{mov\ R0\ Yield} \\ * a_{pc0} + 1 \overset{\mathrm{mem}}{\longmapsto} \mathsf{hvc} \end{array}$$

$$\frac{\mathsf{ResumCondHolds}\ @i * \mathsf{ResumCond}\ @i\ 1/2\ \{P\}}{\mathsf{ResumCond}\ @i\ 1\ \{P\} * \triangleright P}$$

$$\mathsf{ResumCondHolds}\ @1 \twoheadrightarrow \mathsf{WP\ ExecI}\ @1\ \{m, \ldots\}$$

Resumption Conditions - Session-type Like Protocols

$$\text{ResumCond } @1\ 1 \left\{ \begin{array}{l} \text{R0 } @0 \overset{\text{reg}}{\longmapsto} Run \\ * \text{ R1 } @0 \overset{\text{reg}}{\longmapsto} 1 \\ \\ * \text{ ResumCond } @0\ 1/2 \left\{ \begin{array}{l} \text{R0 } @0 \overset{\text{reg}}{\longmapsto} Yield \\ * \text{ R1 } @0 \overset{\text{reg}}{\longmapsto} 1 \end{array} \right\} \end{array} \right\}$$

$$* \triangleright \text{ResumCond } @0\ 1/2 \left\{ \begin{array}{l} \text{R0 } @0 \overset{\text{reg}}{\longmapsto} Yield \\ * \text{ R1 } @0 \overset{\text{reg}}{\longmapsto} 1 \end{array} \right\}$$

$$* \text{ R0 } @0 \overset{\text{reg}}{\longmapsto} Run$$

$$* \text{ R1 } @0 \overset{\text{reg}}{\longmapsto} 1$$

$$* \text{ R0 } @1 \overset{\text{reg}}{\longmapsto} -$$

$$* \ a_{pc0} + 0 \overset{\text{mem}}{\longmapsto} \text{mov R0 Yield}$$

$$* \ a_{pc0} + 1 \overset{\text{mem}}{\longmapsto} \text{hvc}$$

$$\frac{\text{ResumCondHolds } @i * \text{ResumCond } @i\ 1/2\ \{P\}}{\text{ResumCond } @i\ 1\ \{P\} * \triangleright P}$$

$$\text{WP ExecI } @1\ \{m, \ldots\}$$

Resumption Conditions - Session-type Like Protocols

$$\text{ResumCond } @1\ 1 \left\{ \begin{array}{l} \text{R0 } @0 \overset{\text{reg}}{\longmapsto} Run \\ * \text{ R1 } @0 \overset{\text{reg}}{\longmapsto} 1 \\ * \text{ ResumCond } @0\ 1/2 \left\{ \begin{array}{l} \text{R0 } @0 \overset{\text{reg}}{\longmapsto} Yield \\ * \text{ R1 } @0 \overset{\text{reg}}{\longmapsto} 1 \end{array} \right\} \end{array} \right\}$$

$$* \text{ ResumCond } @0\ 1/2 \left\{ \begin{array}{l} \text{R0 } @0 \overset{\text{reg}}{\longmapsto} Yield \\ * \text{ R1 } @0 \overset{\text{reg}}{\longmapsto} 1 \end{array} \right\}$$

$$* \text{ R0 } @0 \overset{\text{reg}}{\longmapsto} Run$$

$$* \text{ R1 } @0 \overset{\text{reg}}{\longmapsto} 1$$

$$* \text{ R0 } @1 \overset{\text{reg}}{\longmapsto} Yield$$

$$* \ a_{pc0} + 0 \overset{\text{mem}}{\longmapsto} \text{mov R0 Yield}$$

$$* \ a_{pc0} + 1 \overset{\text{mem}}{\longmapsto} \text{hvc}$$

$$\text{WP ExecI } @1\ \{m, \ldots\}$$

Resumption Conditions - Session-type Like Protocols

$$\text{ResumCond } @1\ 1 \left\{ \begin{array}{l} \text{R0 } @0 \overset{\text{reg}}{\longmapsto} Run \\ * \text{ R1 } @0 \overset{\text{reg}}{\longmapsto} 1 \\ \\ * \text{ ResumCond } @0\ ½ \left\{ \begin{array}{l} \text{R0 } @0 \overset{\text{reg}}{\longmapsto} Yield \\ * \text{ R1 } @0 \overset{\text{reg}}{\longmapsto} 1 \end{array} \right\} \end{array} \right\}$$

$$* \text{ ResumCond } @0\ ½ \left\{ \begin{array}{l} \text{R0 } @0 \overset{\text{reg}}{\longmapsto} Yield \\ * \text{ R1 } @0 \overset{\text{reg}}{\longmapsto} 1 \end{array} \right\}$$

$$* \text{ R0 } @0 \overset{\text{reg}}{\longmapsto} Yield$$

$$* \text{ R1 } @0 \overset{\text{reg}}{\longmapsto} 1$$

$$* \text{ R0 } @1 \overset{\text{reg}}{\longmapsto} Yield$$

$$* \ a_{pc0} + 0 \overset{\text{mem}}{\longmapsto} \text{mov R0 Yield}$$

$$* \ a_{pc0} + 1 \overset{\text{mem}}{\longmapsto} \text{hvc}$$

$$\text{WP ExecI } @1\ \{m, \ldots\}$$

Resumption Conditions - Session-type Like Protocols

$$\mathsf{ResumCond}\ @1\ 1\ \left\{\begin{array}{l} \mathsf{R0}\ @0 \overset{\mathrm{reg}}{\longmapsto} Run \\ * \mathsf{R1}\ @0 \overset{\mathrm{reg}}{\longmapsto} 1 \\ \\ * \mathsf{ResumCond}\ @0\ 1/2\ \left\{\begin{array}{l} \mathsf{R0}\ @0 \overset{\mathrm{reg}}{\longmapsto} Yield \\ * \mathsf{R1}\ @0 \overset{\mathrm{reg}}{\longmapsto} 1 \end{array}\right\} \end{array}\right\}$$

$$* \mathsf{R0}\ @1 \overset{\mathrm{reg}}{\longmapsto} Yield$$
$$* a_{pc0} + 0 \overset{\mathrm{mem}}{\longmapsto} \mathsf{mov\ R0\ Yield}$$
$$* a_{pc0} + 1 \overset{\mathrm{mem}}{\longmapsto} \mathsf{hvc}$$

$$\frac{\mathsf{ResumCondHolds}\ @i * \mathsf{ResumCond}\ @i\ 1/2\ \{P\}}{\mathsf{ResumCond}\ @i\ 1\ \{P\} * \triangleright P}$$

$$\mathsf{ResumCondHolds}\ @1 \twoheadrightarrow \mathsf{WP\ ExecI}\ @1\ \{m, \ldots\}$$

What If We Don't Know Code of VM_1?

Robust safety: arbitrary unknown code won't break known VMs: it can only change memory it has (or can get) access to

$$\frac{? * \mathsf{ResumCond}\ @1\ 1/2\ \{?\}}{\mathsf{ResumCondHolds}\ @1 \twoheadrightarrow \mathsf{WP}\ \mathtt{ExecI}\ @1\ \{m, \ldots\}}$$

What If We Don't Know Code of VM_1?

Robust safety: arbitrary unknown code won't break known VMs: it can only change memory it has (or can get) access to

$$\frac{\mathsf{Owned}(pgt) * \mathsf{ResumCond}\ @1\ ½\ \{\mathsf{Protocol}(pgt)\}}{\mathsf{ResumCondHolds}\ @1 \mathrel{-\!\!*} \mathsf{WP}\ \mathtt{ExecI}\ @1\ \{m,\ldots\}}$$

We define a logical relation that enforces **robust safety**

- No assumptions on contents of memory, only on the page tables
- Challenge is to consider all possible interactions, incl. in-flight memory sharing, etc.

What If We Don't Know Code of VM_1?

Robust safety: arbitrary unknown code won't break known VMs: it can only change memory it has (or can get) access to

$$\frac{\textsc{FundamentalTheorem}\quad \mathsf{Owned}(pgt) * \mathsf{ResumCond}\ @i\ 1/2\ \{\mathsf{Protocol}(pgt)\}}{\mathsf{ResumCondHolds}\ @i \twoheadrightarrow \mathsf{WP}\ \mathtt{ExecI}\ @i\ \{m, \top\}}$$

We define a logical relation that enforces **robust safety**

- No assumptions on contents of memory, only on the page tables
- Challenge is to consider all possible interactions, incl. in-flight memory sharing, etc.

Conclusion

Contributions

- An **operational semantics** for the combination of a machine and a hypervisor
- A **program logic** for modular reasoning about VMs with communication
  - Verified the run & yield example with resumption conditions (saved ~200 LOC comparing to invariants)
- A **logical relation** for robust safety property
  - The run & yield example with an unknown VM (~80 more LOC)

Next Steps

- Adding interrupts/exceptions - crucial for scheduling
- Considering multi-core CPUs - concurrency
- Non-interference