# ARM Virtualization: Performance and Architectural Implications

Christoffer Dall, Shih-Wei Li, Jin Tack Lim,
Jason Nieh, and Georgios Koloventzos

COLUMBIA UNIVERSITY
IN THE CITY OF NEW YORK

# ARM®

ARM Servers

ARM Network Equipment

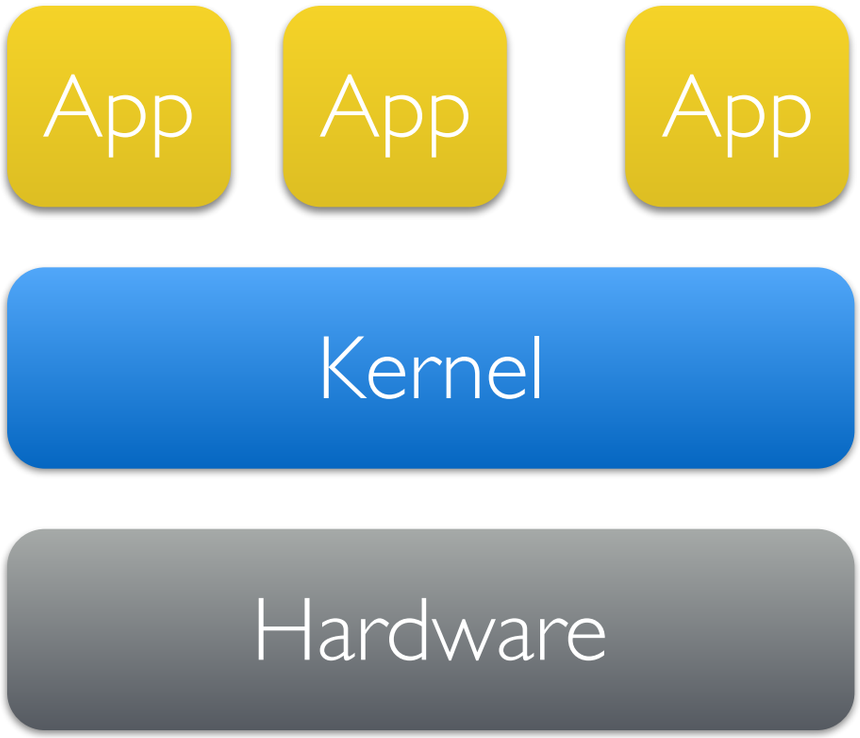
KVM

Xen

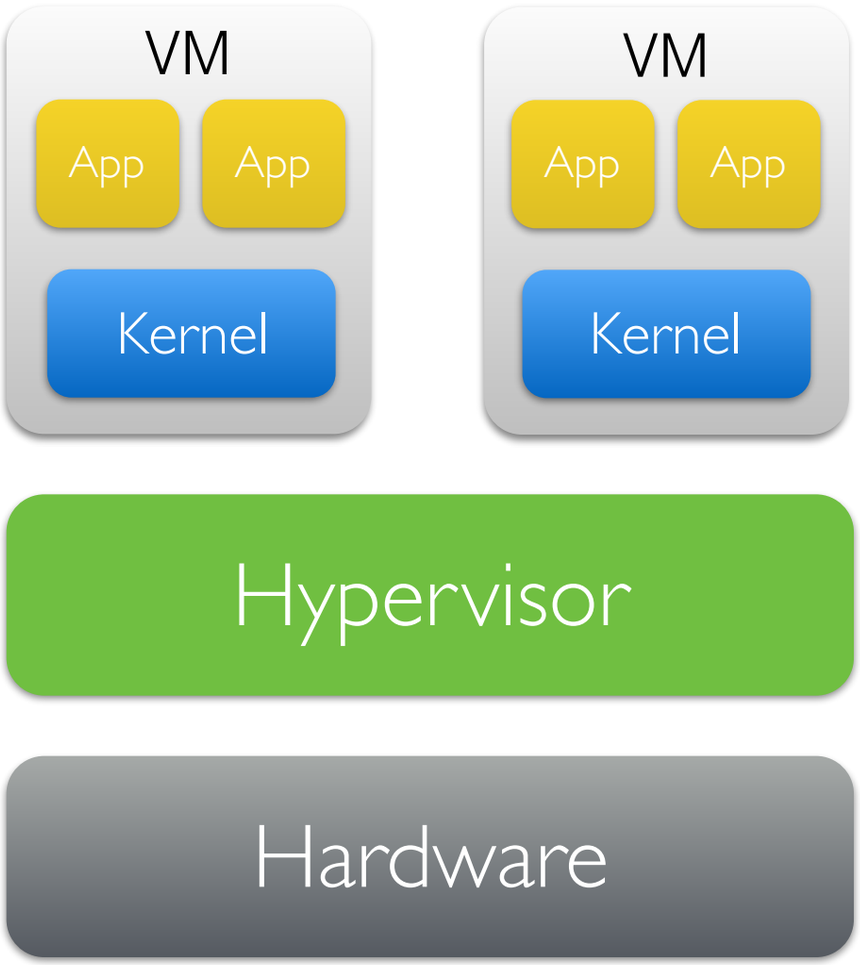
Virtualization

# Virtualization

## Native



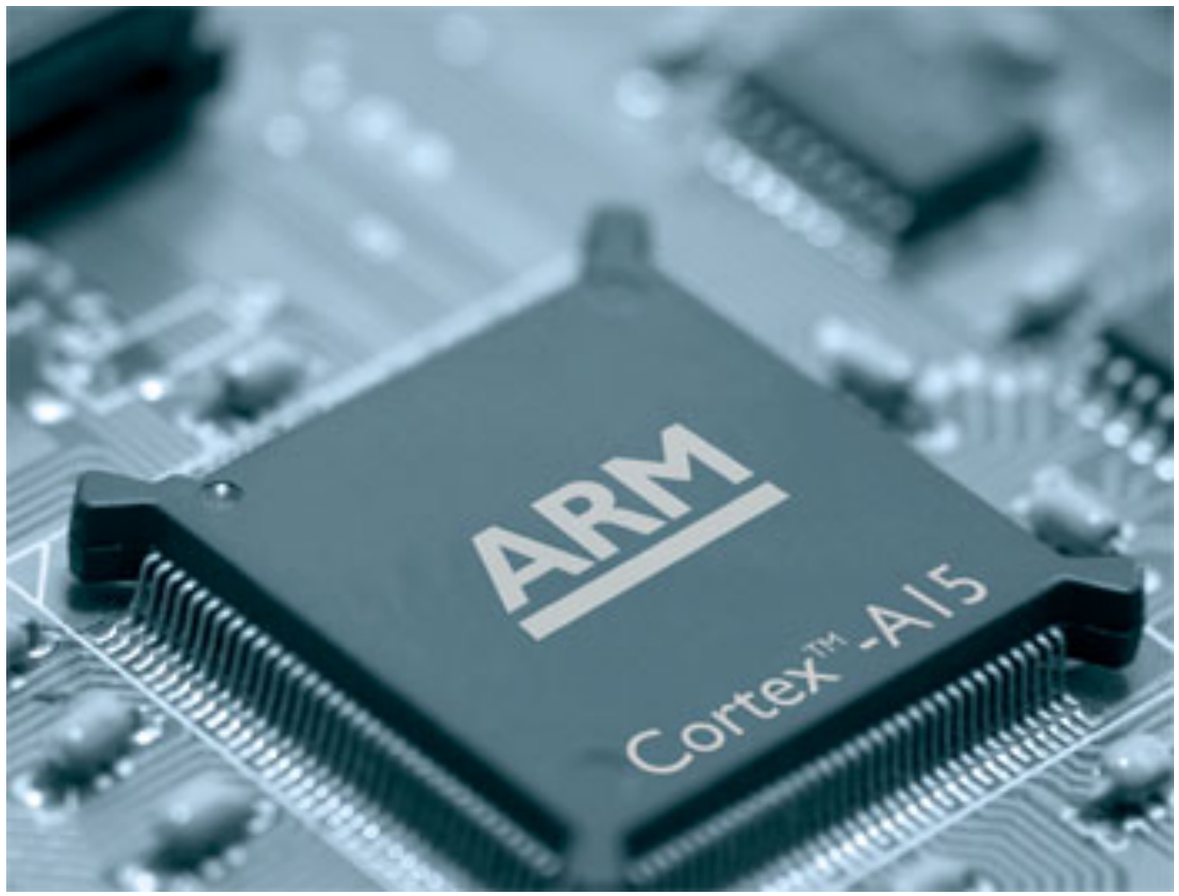

## Virtual Machines



# ARM Hardware Virtualization Support

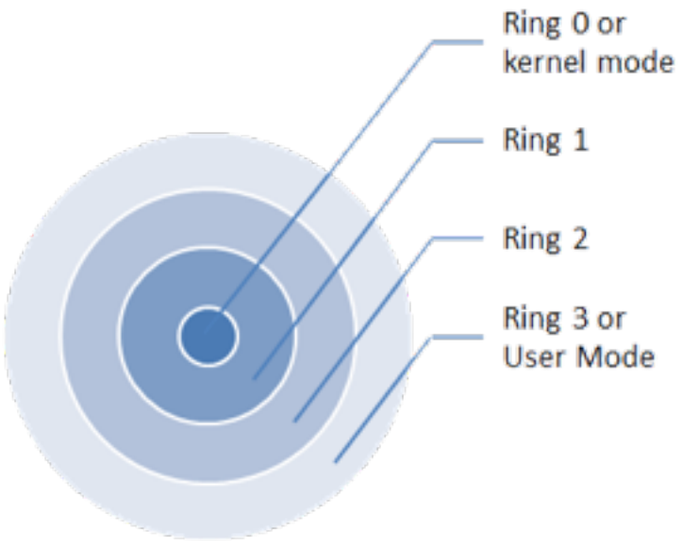
ARM®

## Virtualization Extensions

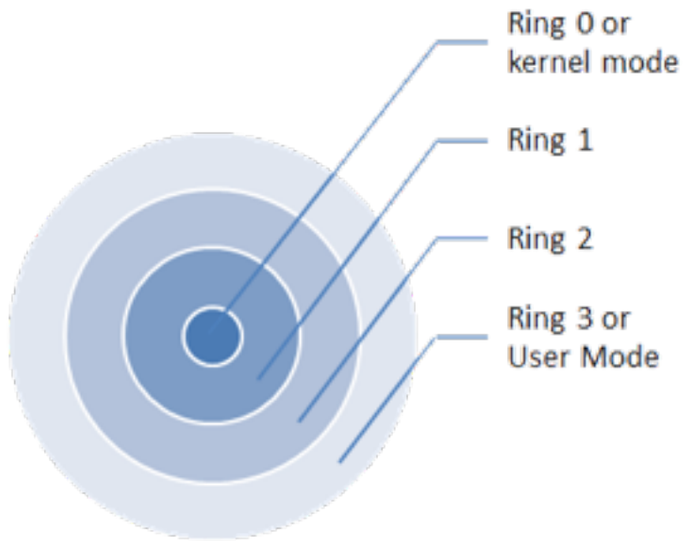


# x86

## Root (Hypervisor)



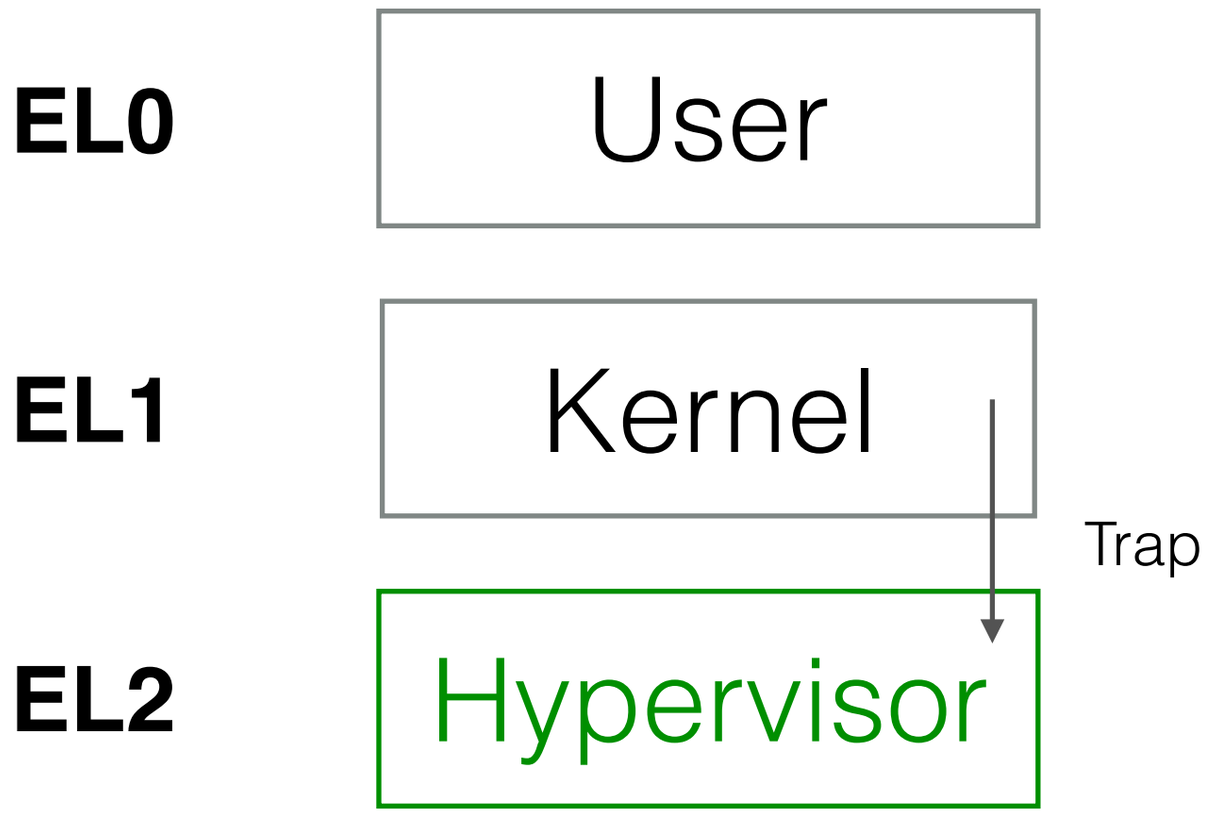

## Non-Root (VM)




VM Exit

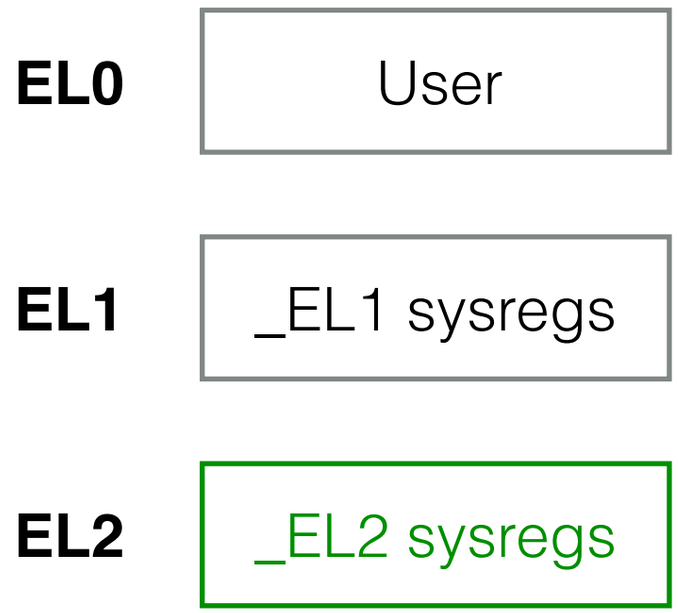
Save/Restore state to VMCS

# ARM Virtualization Extensions



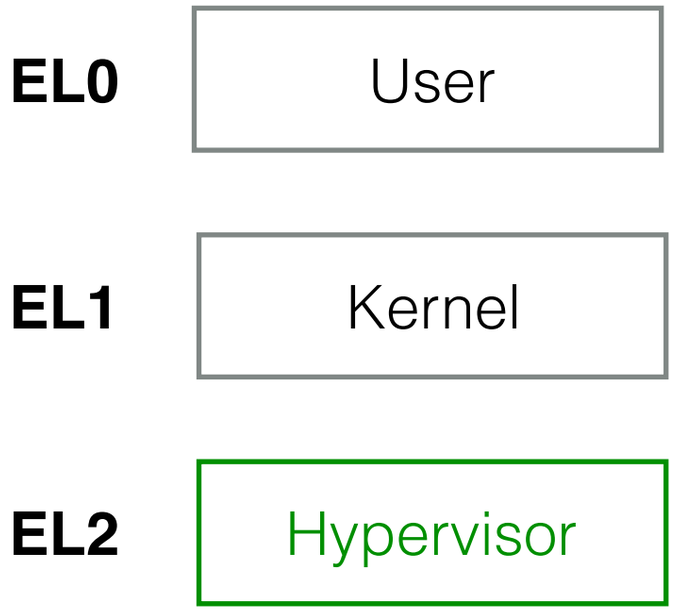
# EL2

- Controlled by EL2 system registers
- Limited to support hypervisors, not OS kernels



# ARM Virtualization Extensions Design Choices

1. Clear hierarchy from user to kernel to hypervisor
2. Reduced complexity



# ARM Virtualization Performance ?

# Measurement Study

- Micro-benchmarks: low-level hypervisor operations
- Macro-benchmarks: application workloads

# Hardware Setup

## ARM Hardware

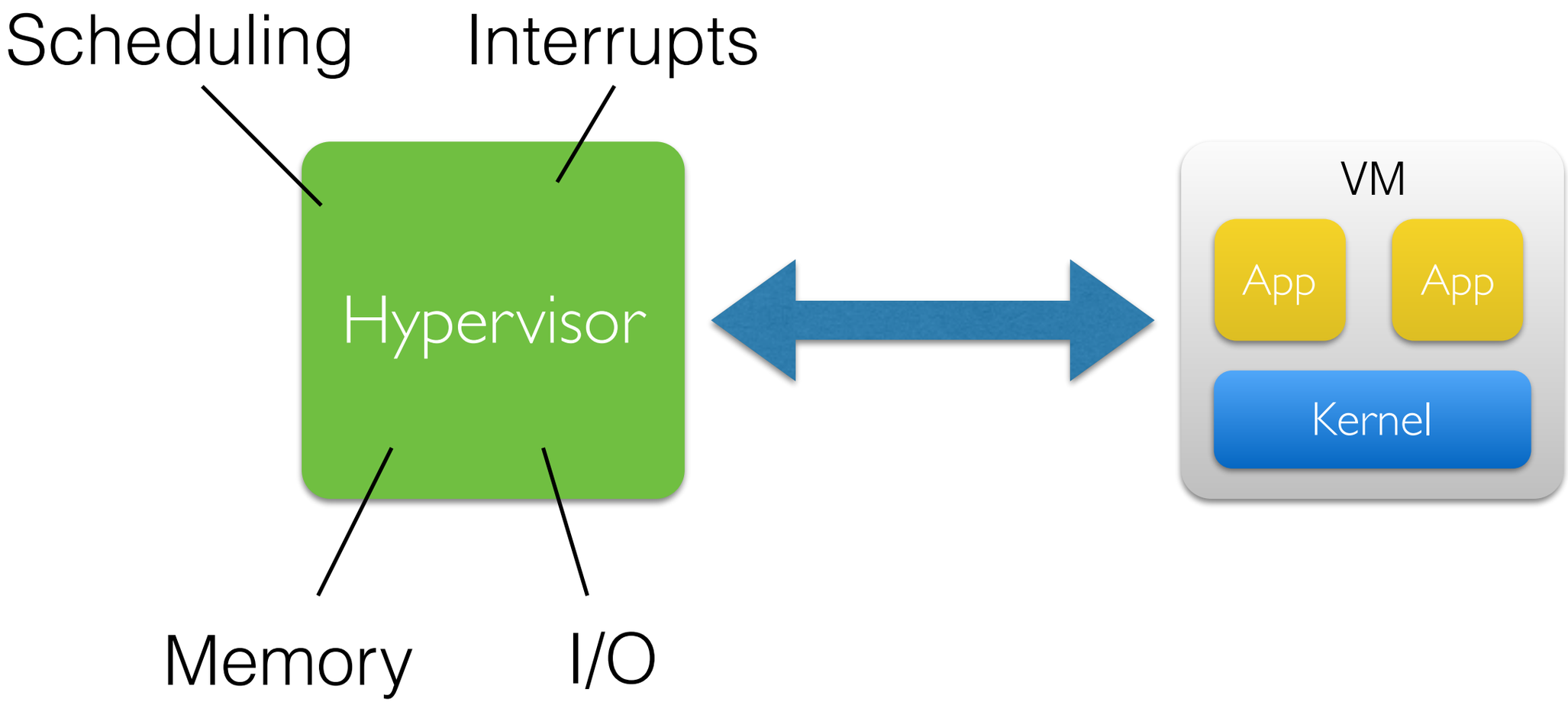
- HP Moonshot m400
- 64-bit ARMv8-A
- 2.4 GHz APM Atlas CPU
- 8-way SMP
- 64 GB RAM (capped at 16 GB)
- 10 GB Ethernet

## x86 Hardware

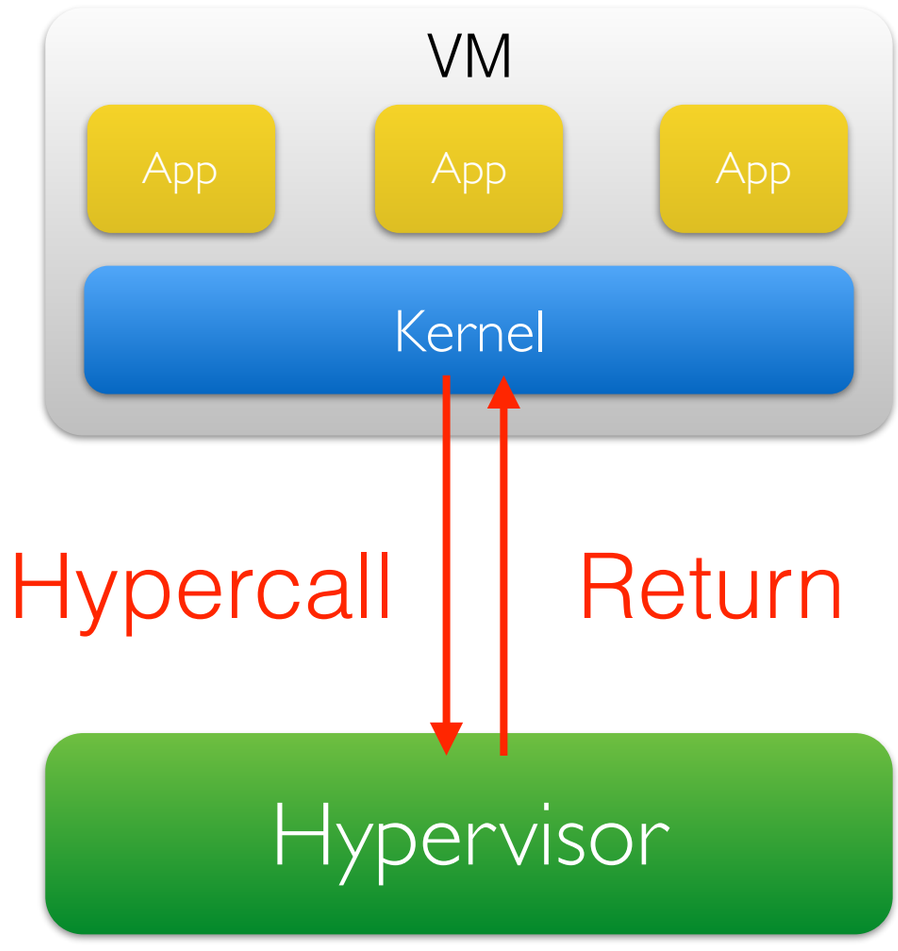
- Dell PowerEdge r320
- 64-bit x86_x64
- 2.1 GHz Intel Xeon ES-2450
- 8-way SMP
- 16 GB RAM
- 10 GB Ethernet

# Software Setup


# VM-to-Hypervisor Transitions



# No-Op Hypercall



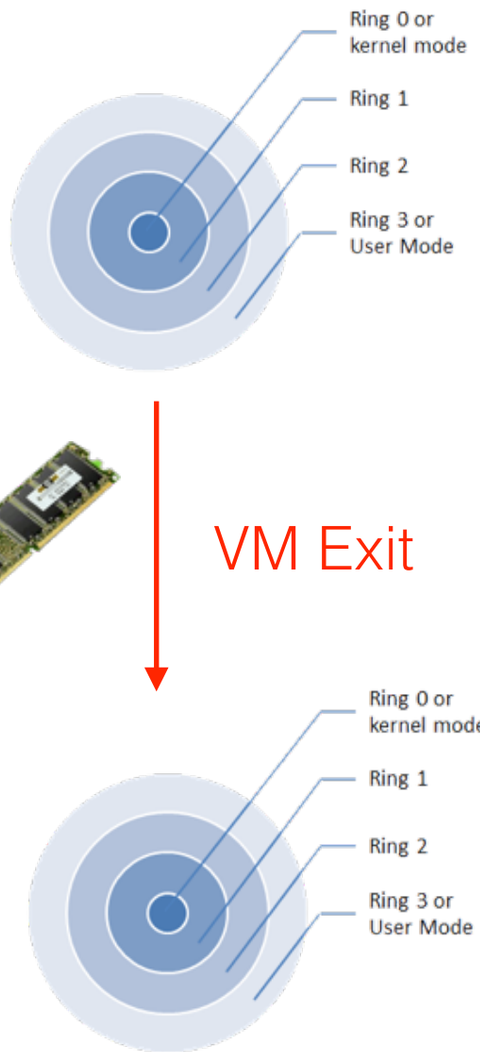
# Micro Results

| CPU Clock Cycles | ARM | | x86 | |
|---|---|---|---|---|
| | KVM | Xen | KVM | Xen |
| Hypercall | 6,500 | 376 | 1,300 | 1,228 |

1: ARM can be either much faster or slower than x86

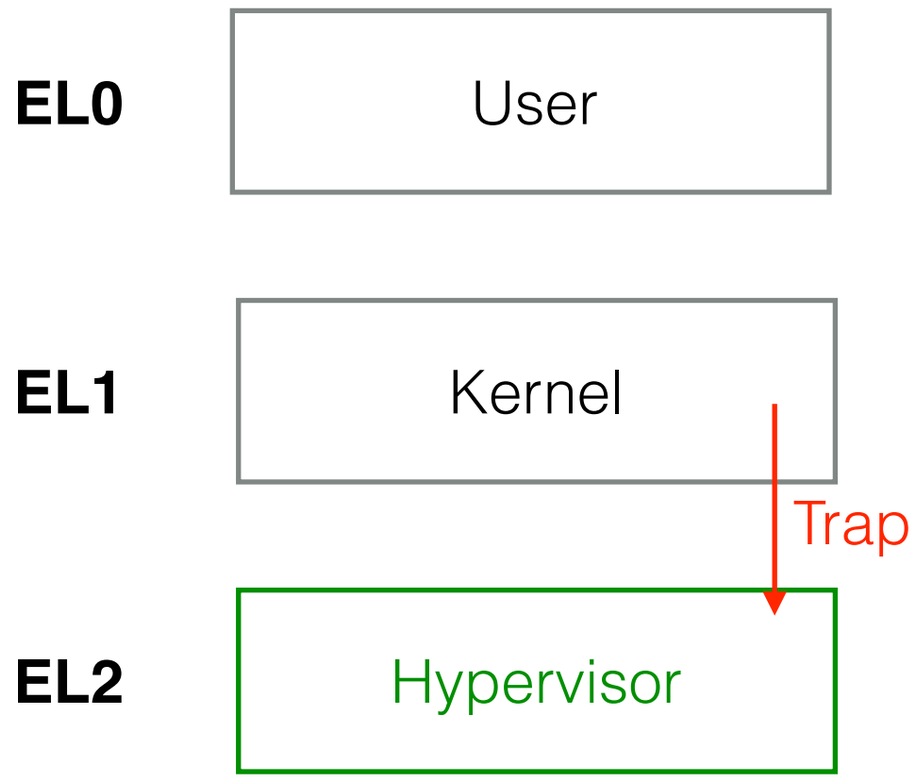
# x86

Non-Root (VM)




Root (Hypervisor)

# ARM



# Micro Results

| CPU Clock Cycles | ARM | | x86 | |
|---|---|---|---|---|
| | KVM | Xen | KVM | Xen |
| Hypercall | 6,500 | 376 | 1,300 | 1,228 |

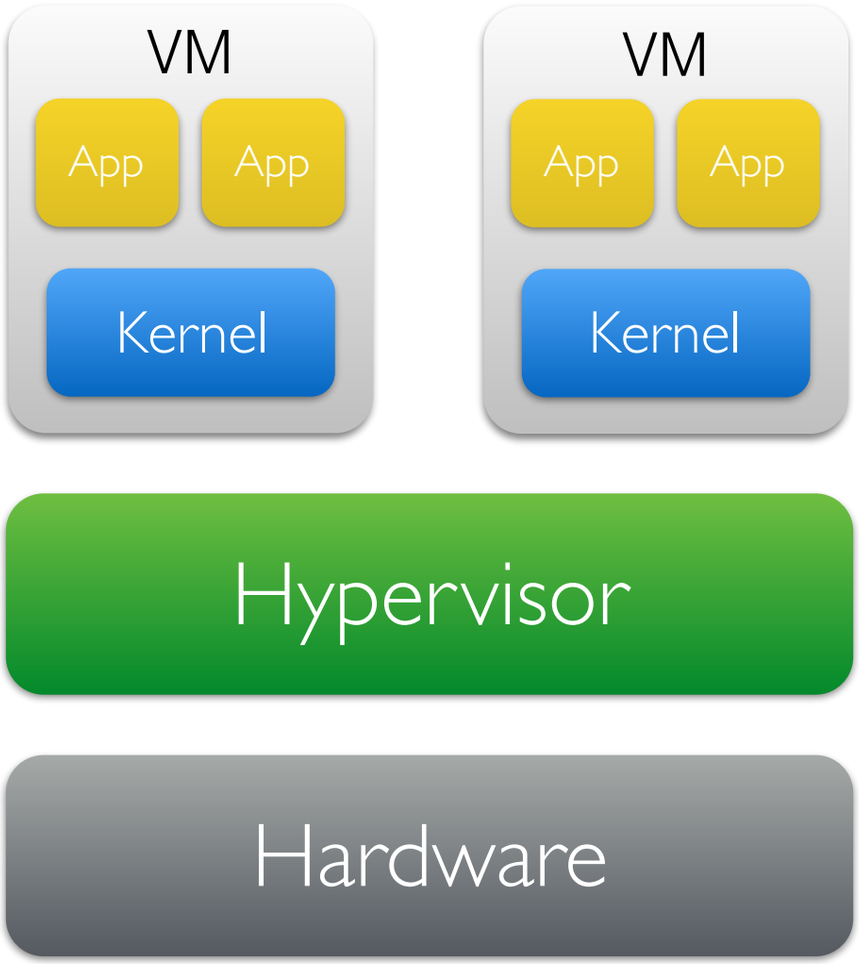
1: ARM can be either much faster or slower than x86

-> x86 VM Exit more complicated than ARM Trap

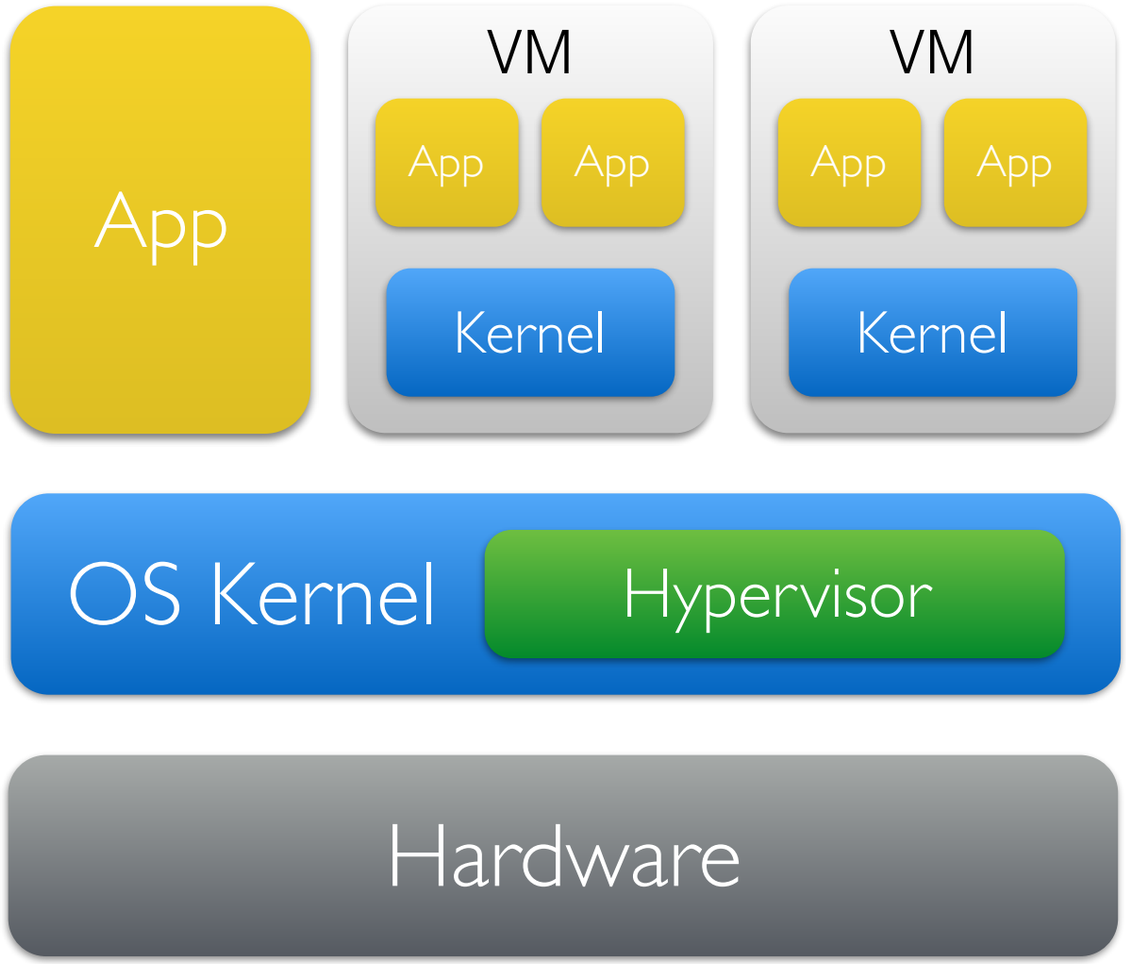
2: KVM is much slower than Xen on ARM

# Hypervisor Design

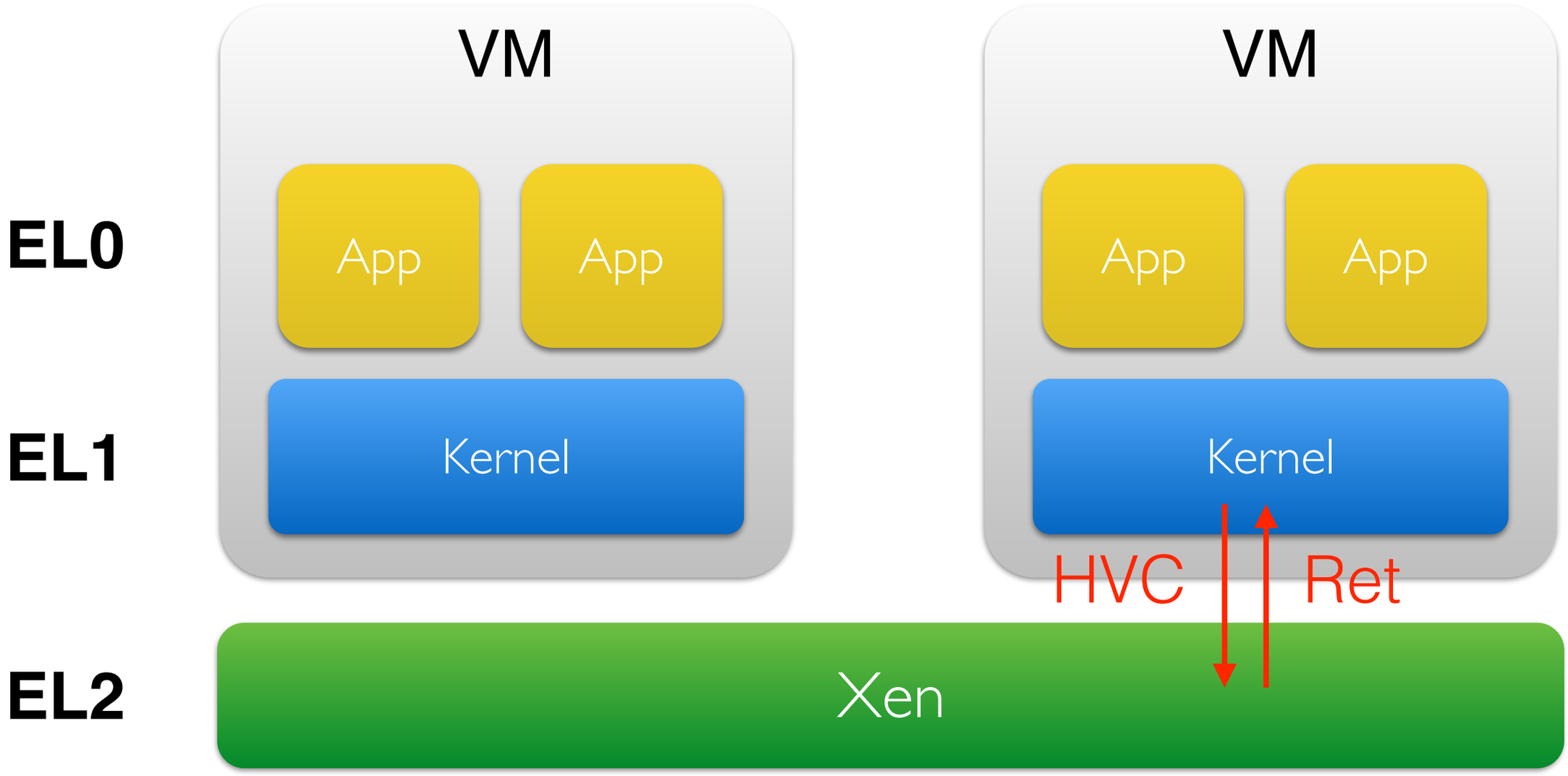
## Type 1 (Bare-Metal)



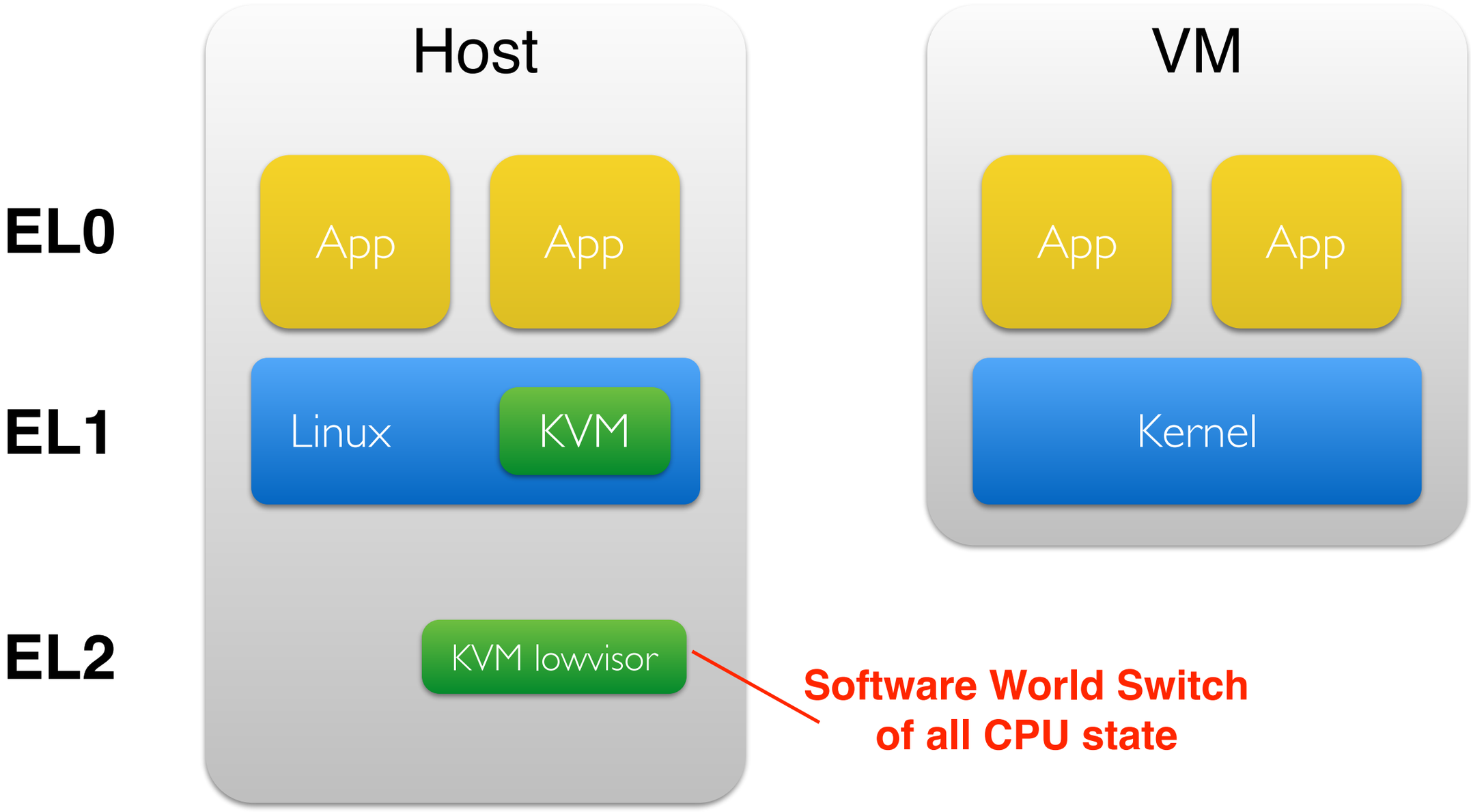

## Type 2 (Hosted)



# Xen ARM: Bare-Metal

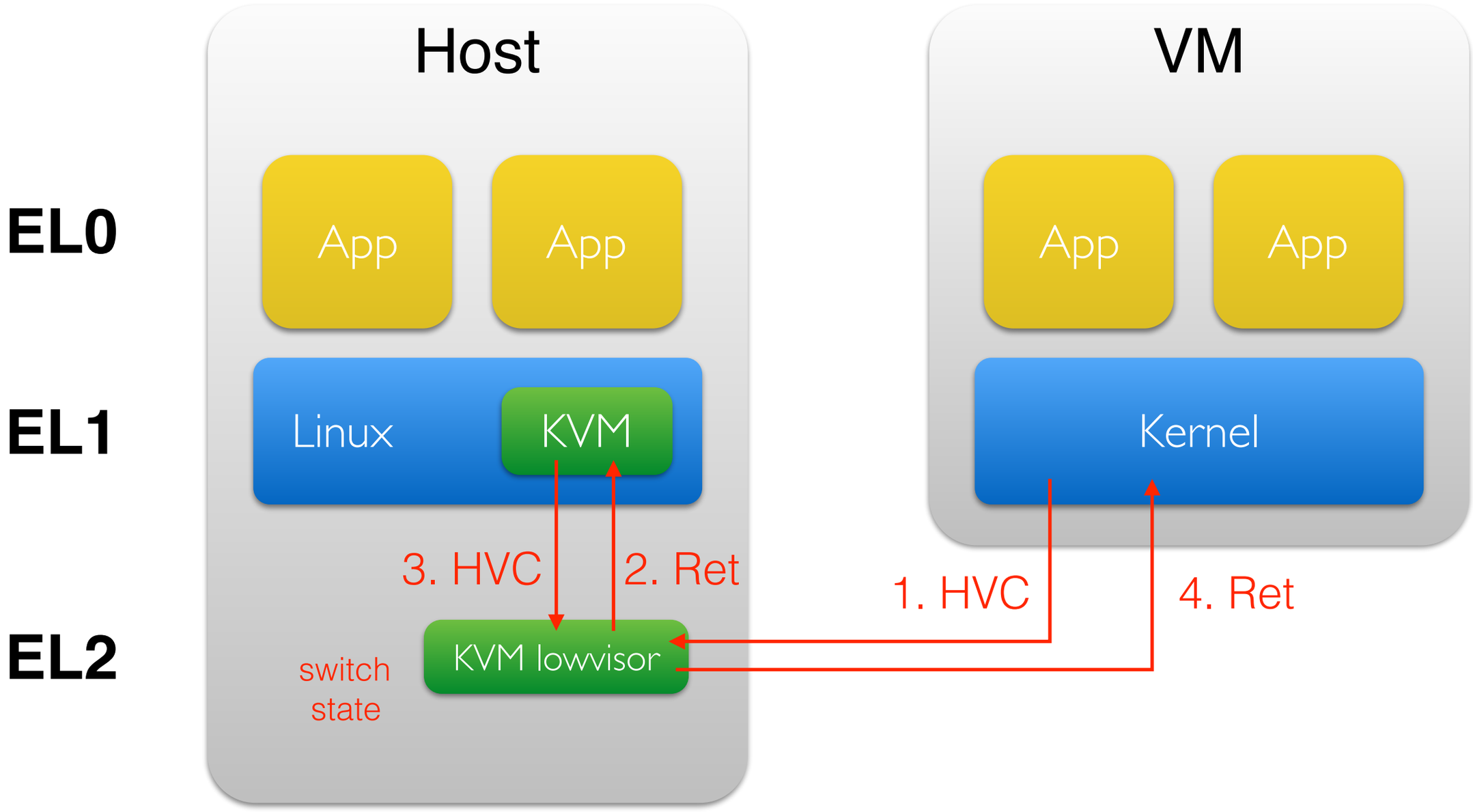


# KVM/ARM: Hosted




*ASPLOS 2014: KVM/ARM: The Design and Implementation of the Linux ARM Hypervisor

# KVM/ARM: Hosted




*ASPLOS 2014: KVM/ARM: The Design and Implementation of the Linux ARM Hypervisor

# Micro Results

| CPU Clock Cycles | ARM | | x86 | |
|---|---|---|---|---|
| | KVM | Xen | KVM | Xen |
| Hypercall | 6,500 | 376 | 1,300 | 1,228 |

1: ARM can be either much faster or slower than x86

-> x86 VM Exit more complicated than ARM Trap

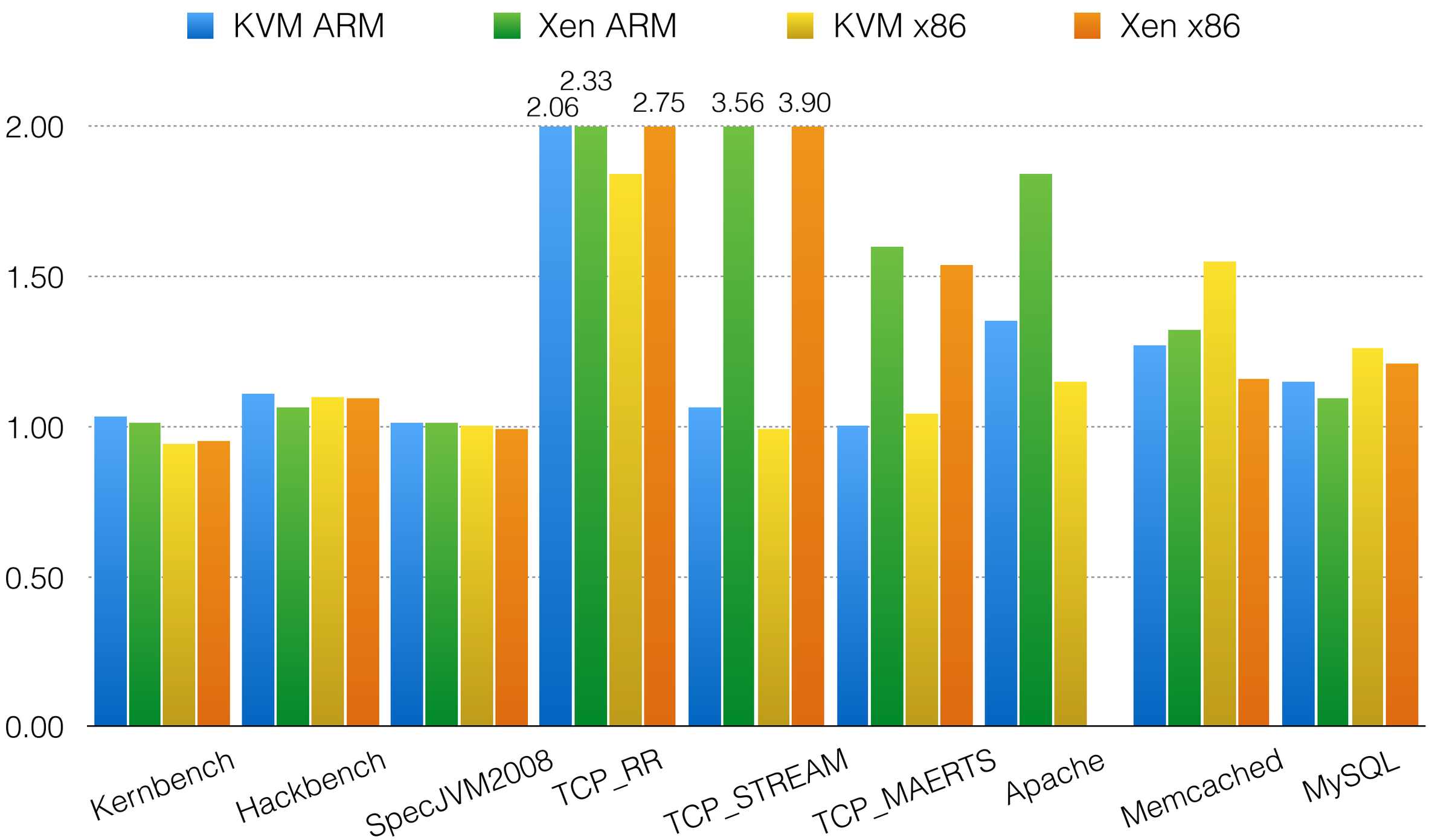
2: KVM is much slower than Xen on ARM

-> ARM architecture not designed for Type 2

# Application Workloads

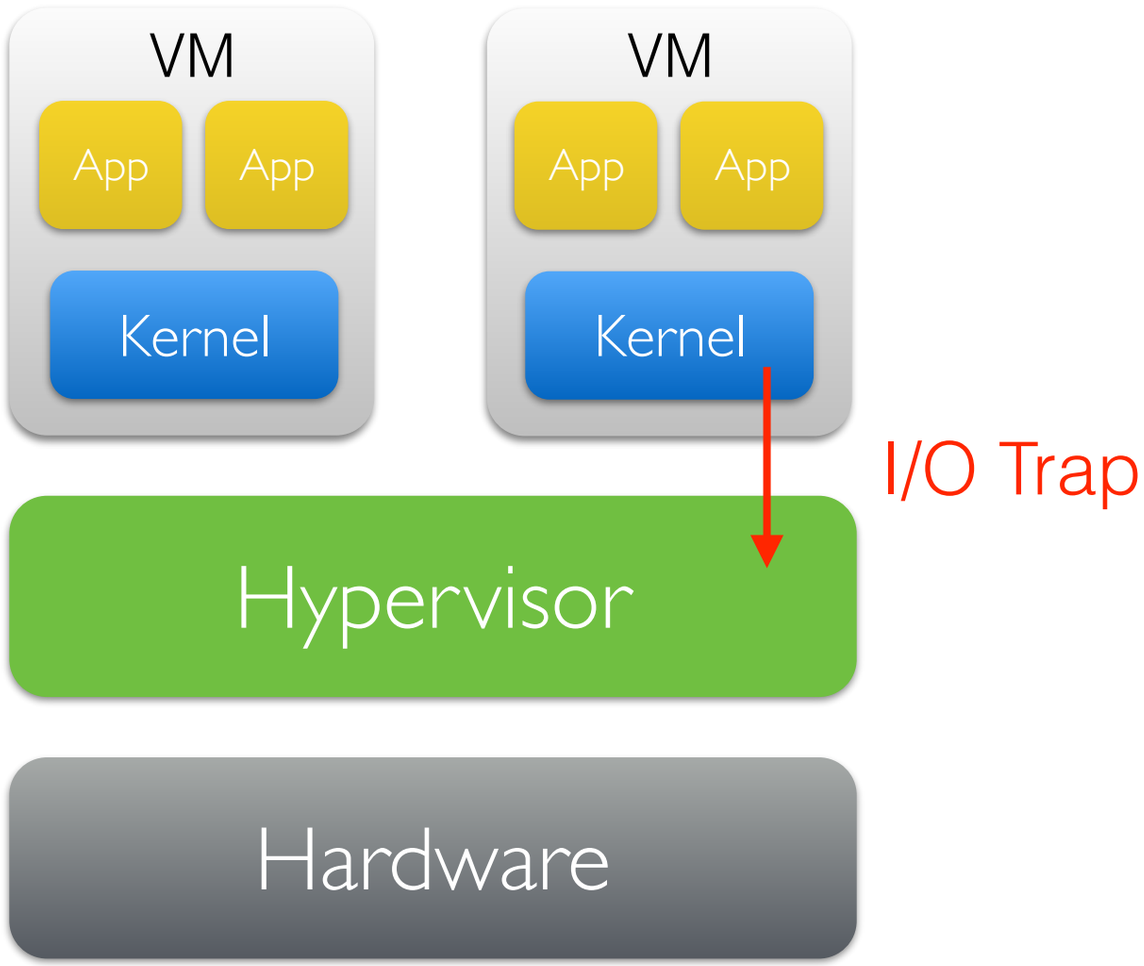
| Application | Description |
|---|---|
| Kernbench | Kernel compile |
| Hackbench | Scheduler stress |
| SPECjvm2008 | Java workload |
| Netperf | Network performance |
| Apache | Web server stress |
| Memcached | Key-Value store |
| MySQL | Database workload |

# Application Performance

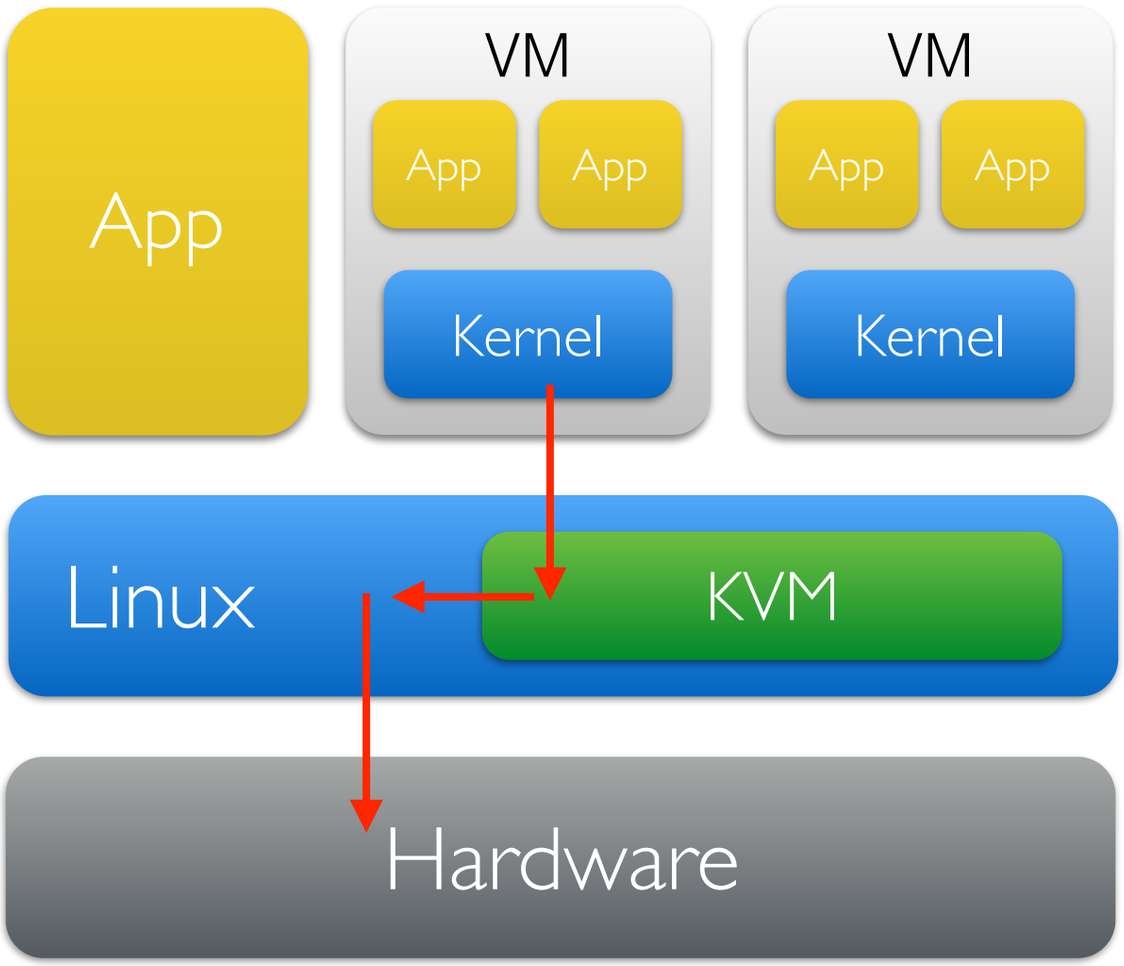



Normalized overhead (lower is better)

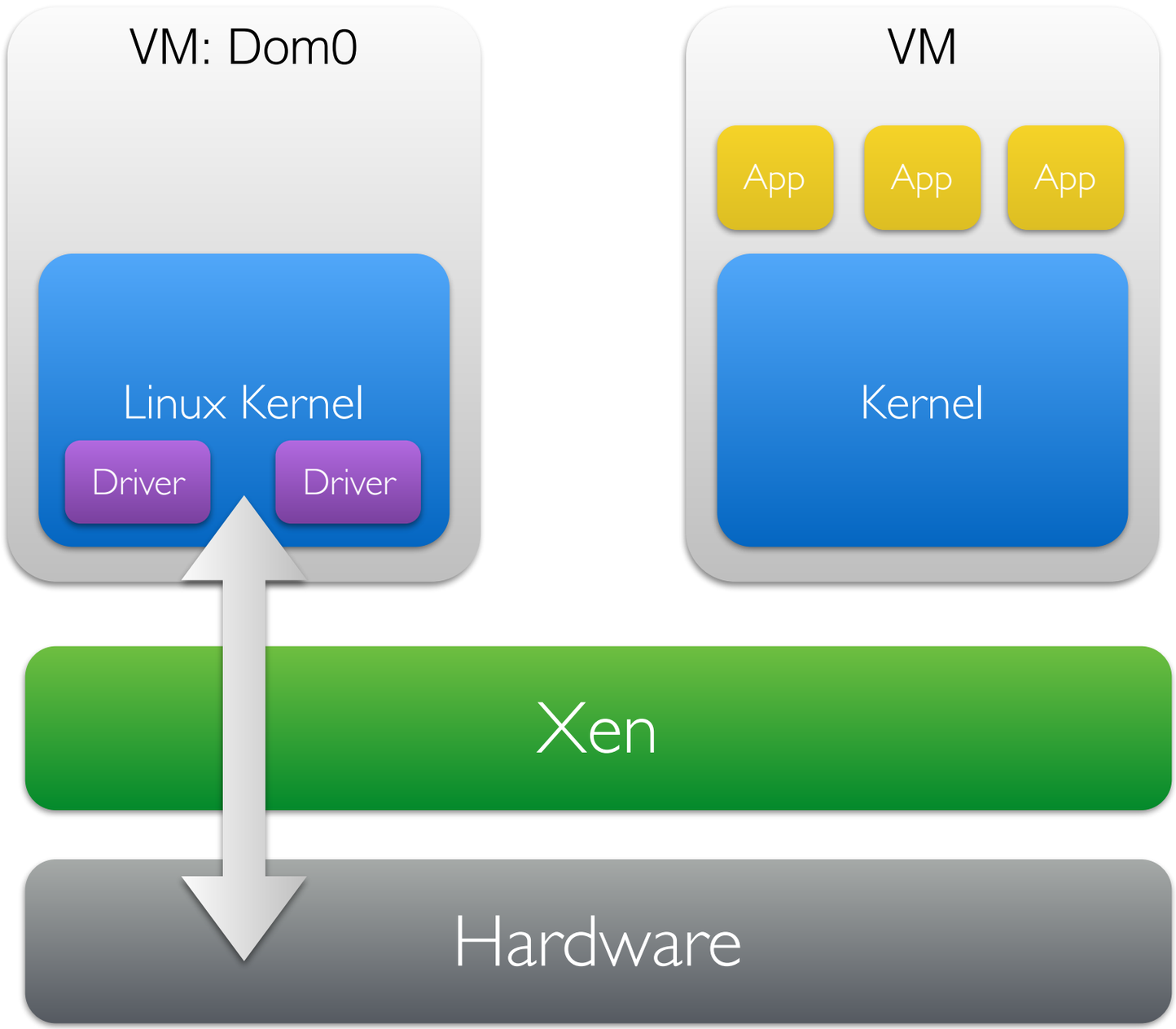
# Virtualized I/O



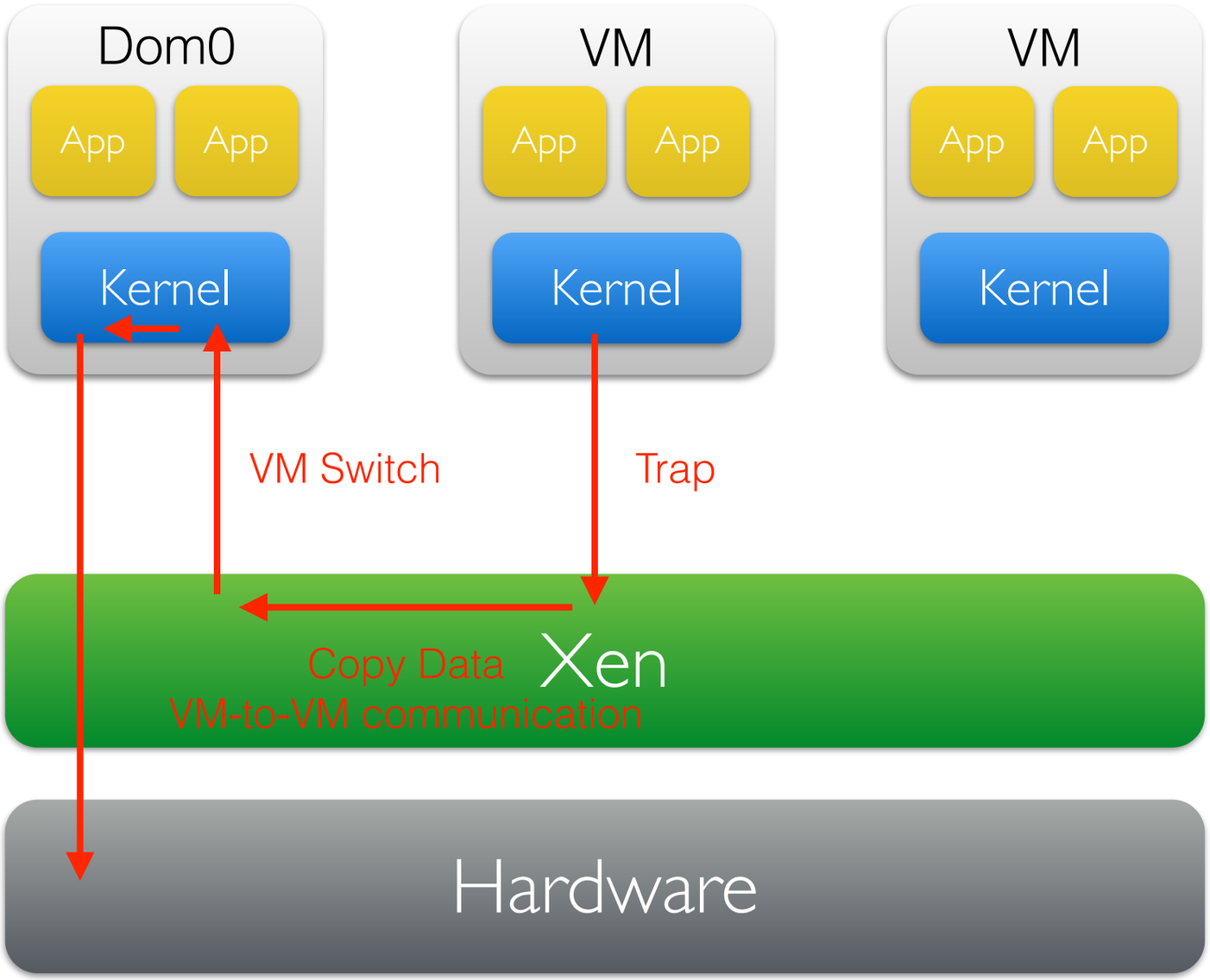
# KVM I/O Model



# Xen Device Drivers



# Xen I/O Model


Dom0
App
App
Kernel
VM
App
App
Kernel
VM
App
App
Kernel
VM Switch
Trap
Copy Data
VM-to-VM communication
Xen
Hardware

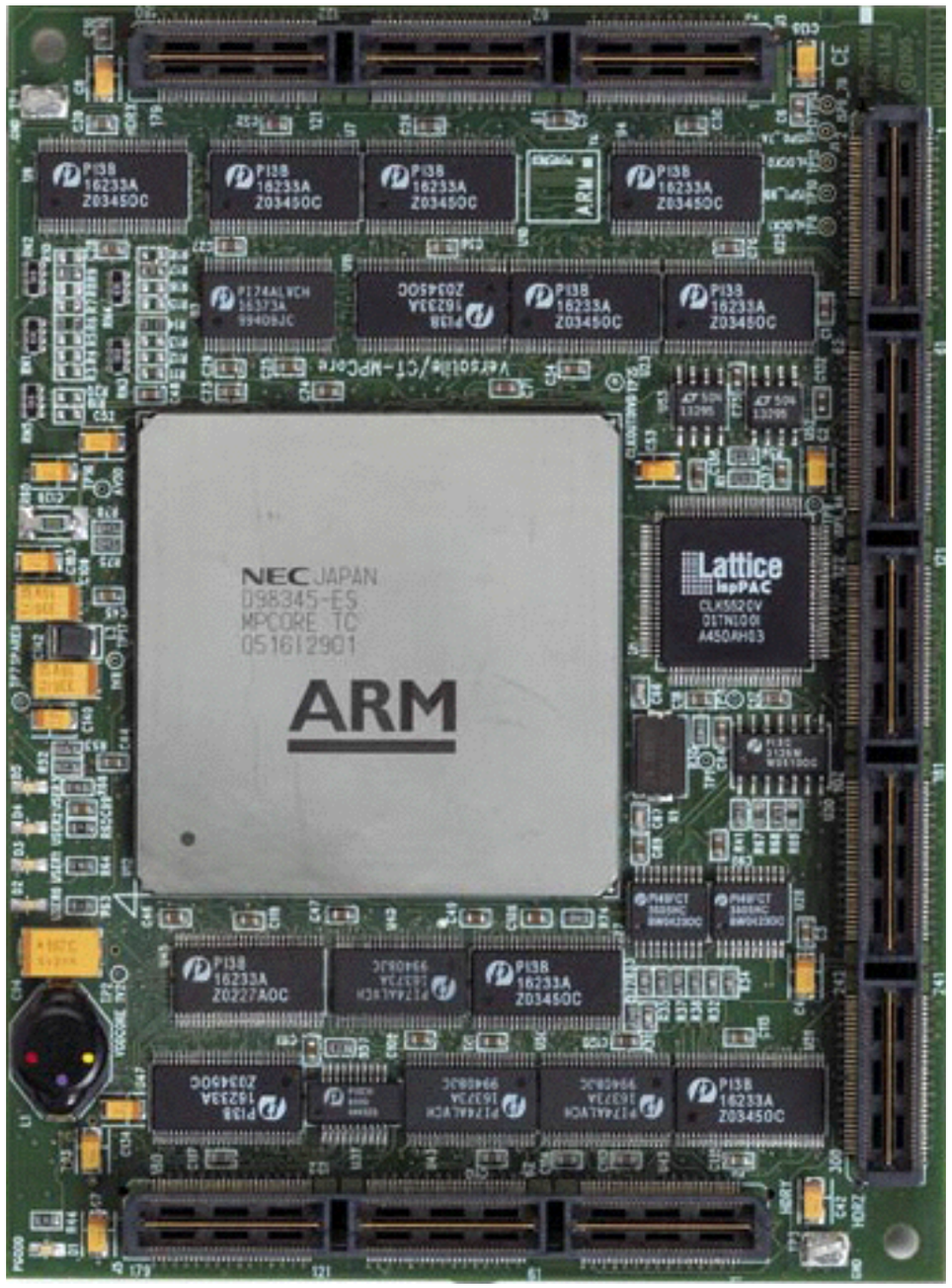
# KVM ARM I/O

- Trap is slow
- But all you do is a trap

# Xen ARM I/O

- Trap is fast
- But you do much more…

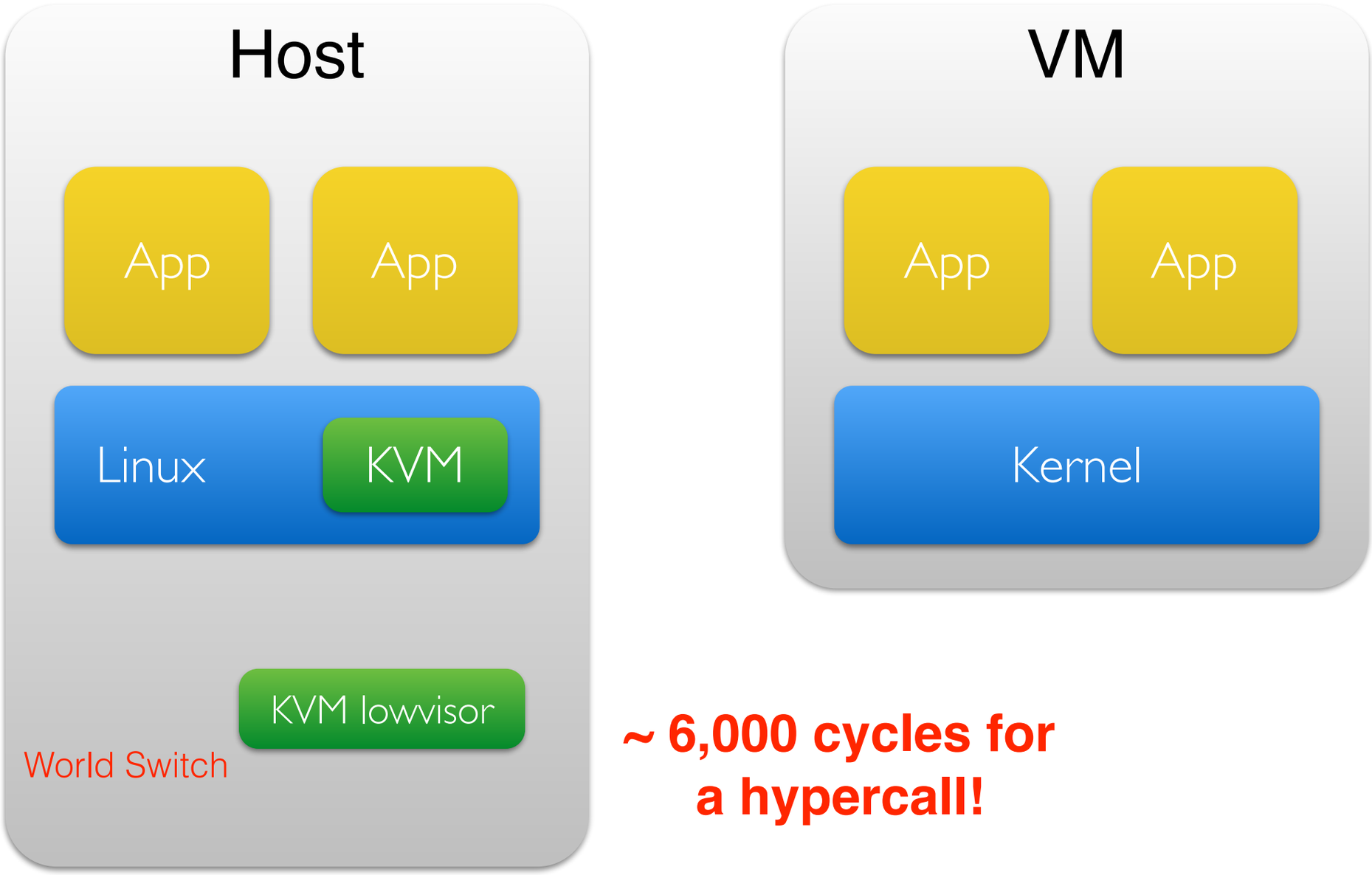
# Architecture Improvements



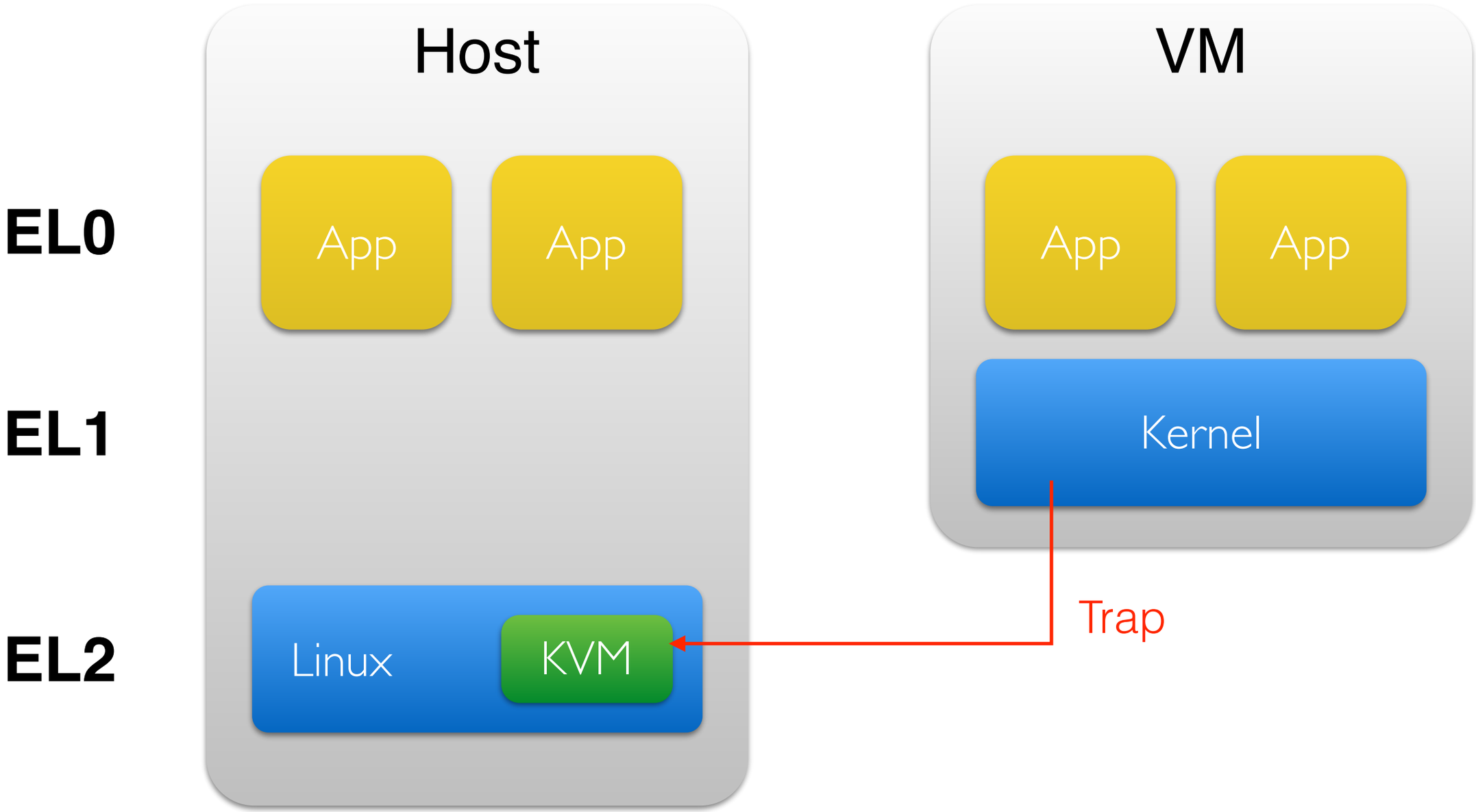
# VHE

Virtualization Host Extension

# KVM


EL0
EL1
EL2
Host
App
App
Linux
KVM
KVM lowvisor
World Switch
VM
App
App
Kernel
~ 6,000 cycles for
a hypercall!

# KVM + VHE



# VHE

1. Expand EL2 to support all EL1 features

2. EL1 register accesses to go to EL2

# VHE

- Available in ARMv8.1

- No (public) hardware yet

# Conclusions

- Micro operations: ARM can be faster than x86
- Not achievable for Type 2 hypervisors
- Type 1 is dominated by other I/O costs
- ARM overhead is comparable to x86
- ARMv8.1 adds VHE for hosted hypervisors
- The software matters!