Linaro connect
San Francisco 2017

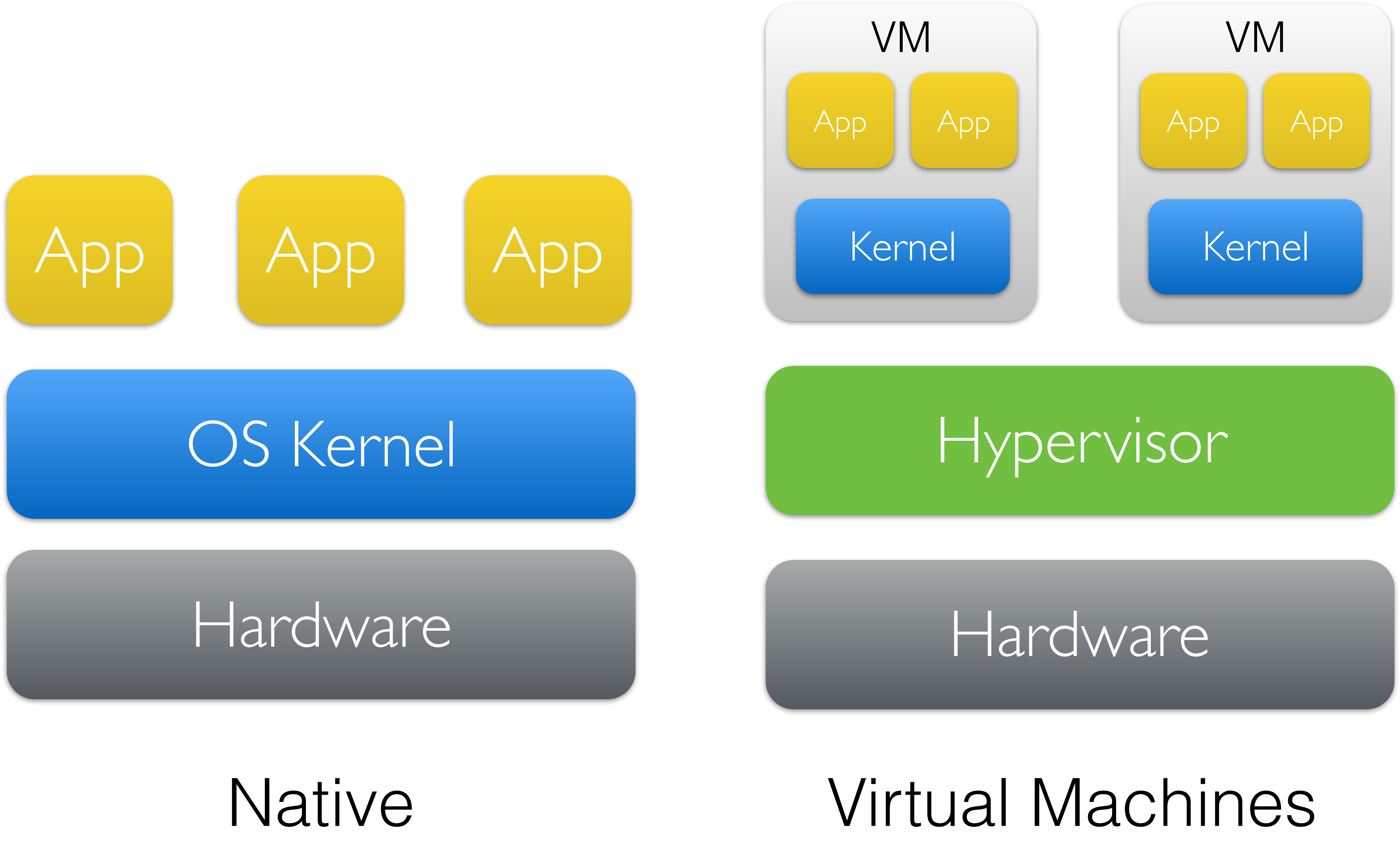
SFO17-403: Optimizing the Design and Implementation of KVM/ARM

Christoffer Dall

““***Efficient***, *isolated duplicate*
of the real machine””

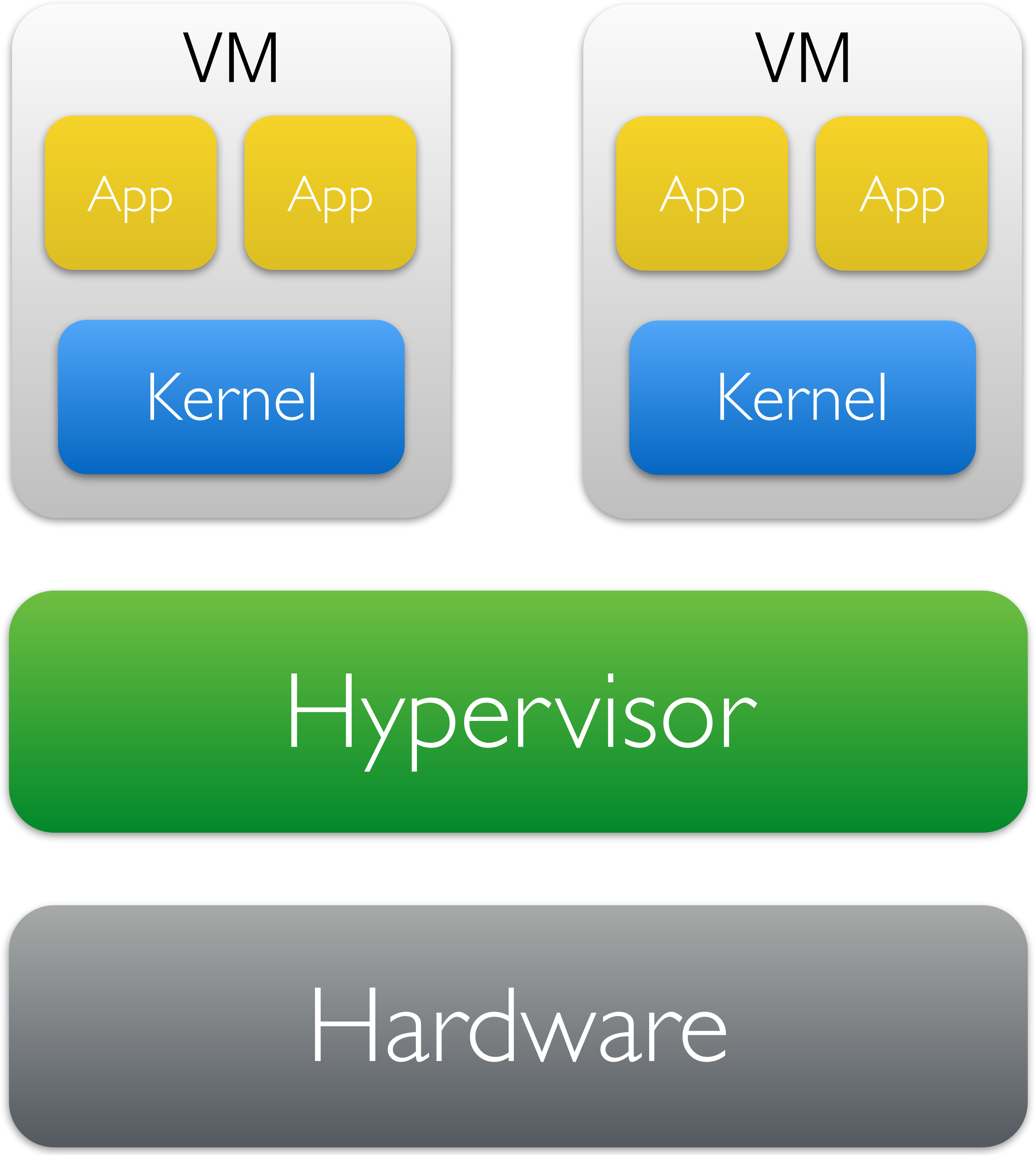
–Popek and Golberg
[Formal requirements for virtualizable third generation architectures ’74]

Virtualization

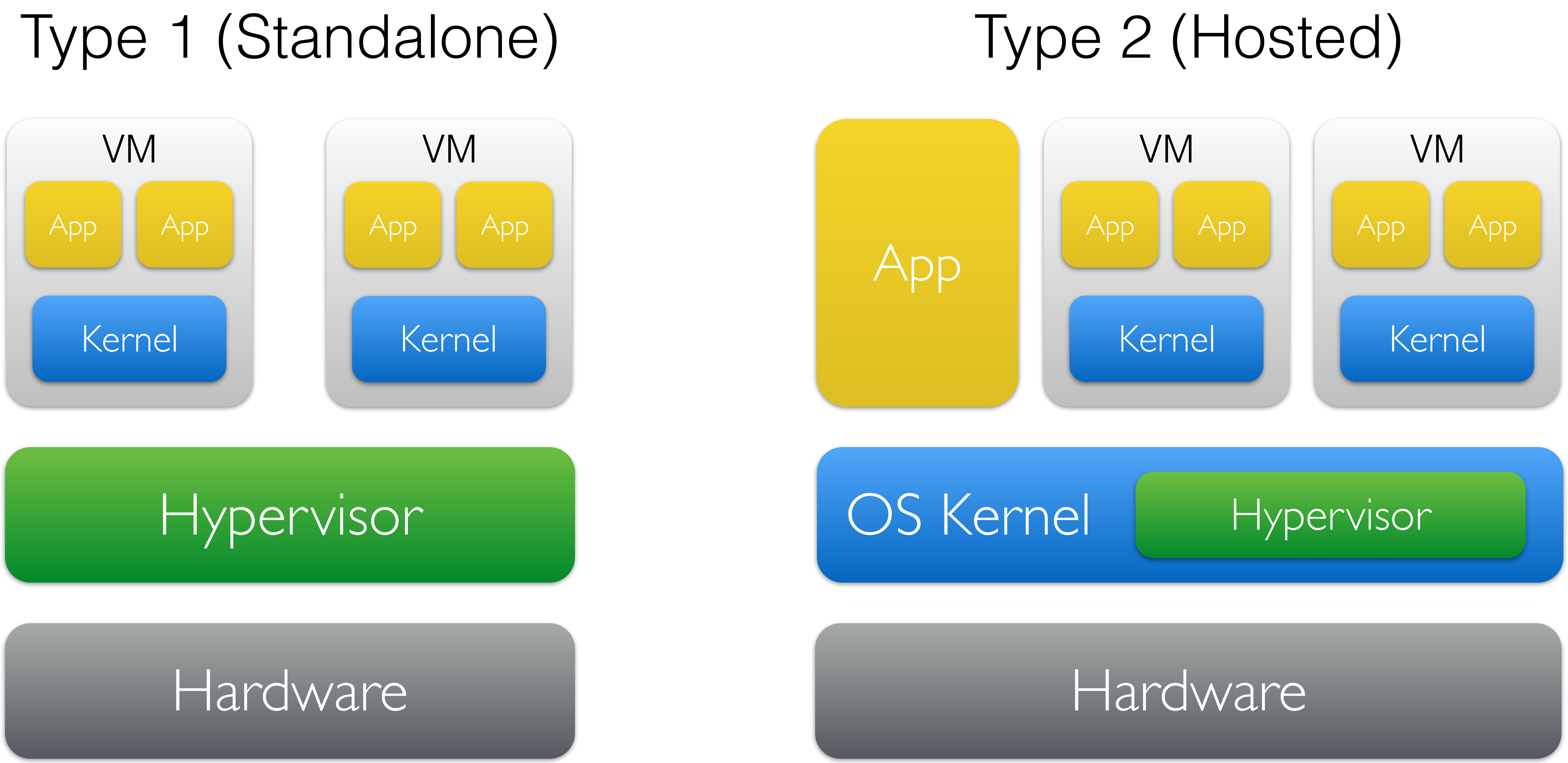


Hypervisor Design

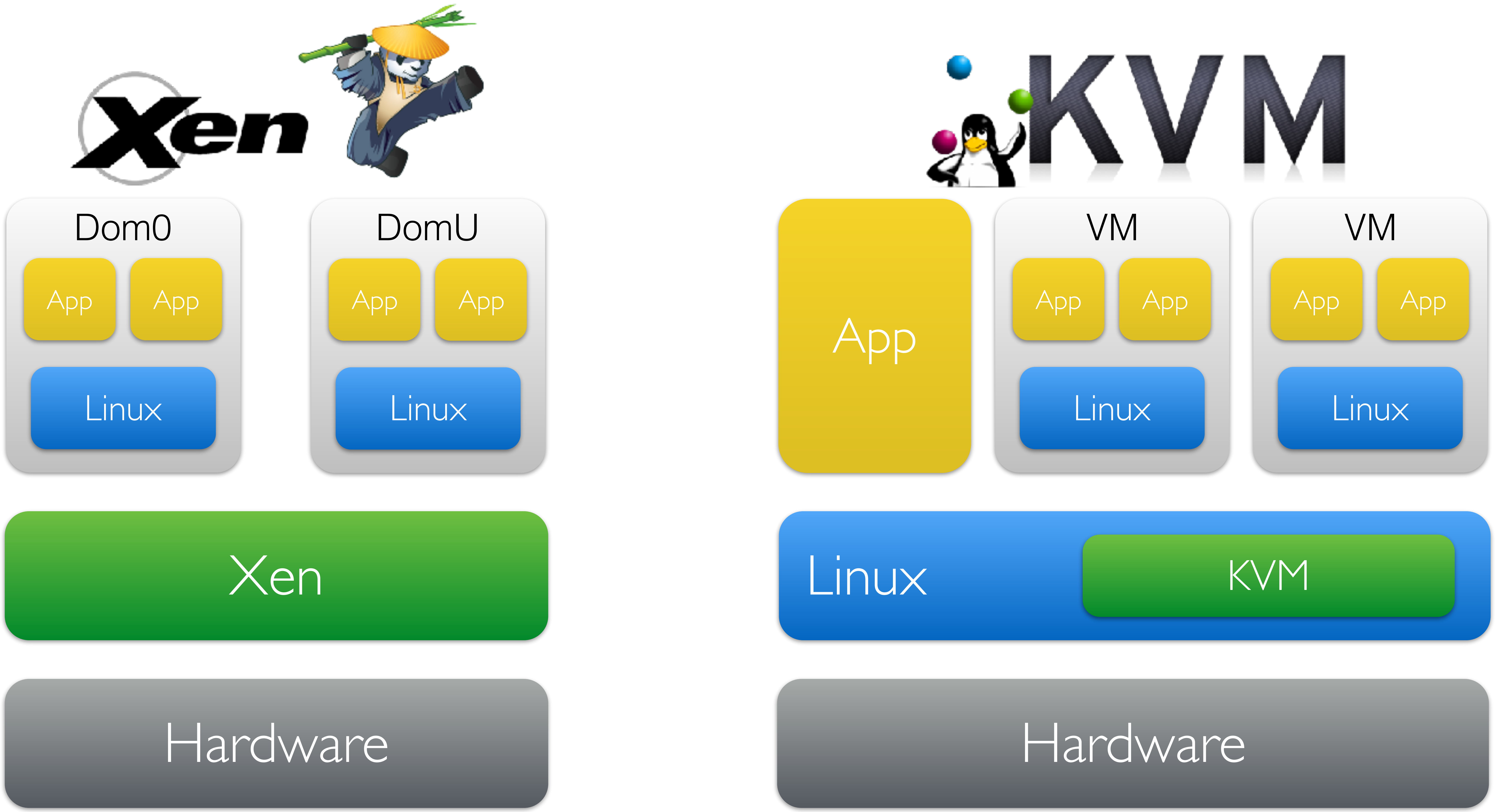
Type 1 (Standalone)



Hypervisor Design



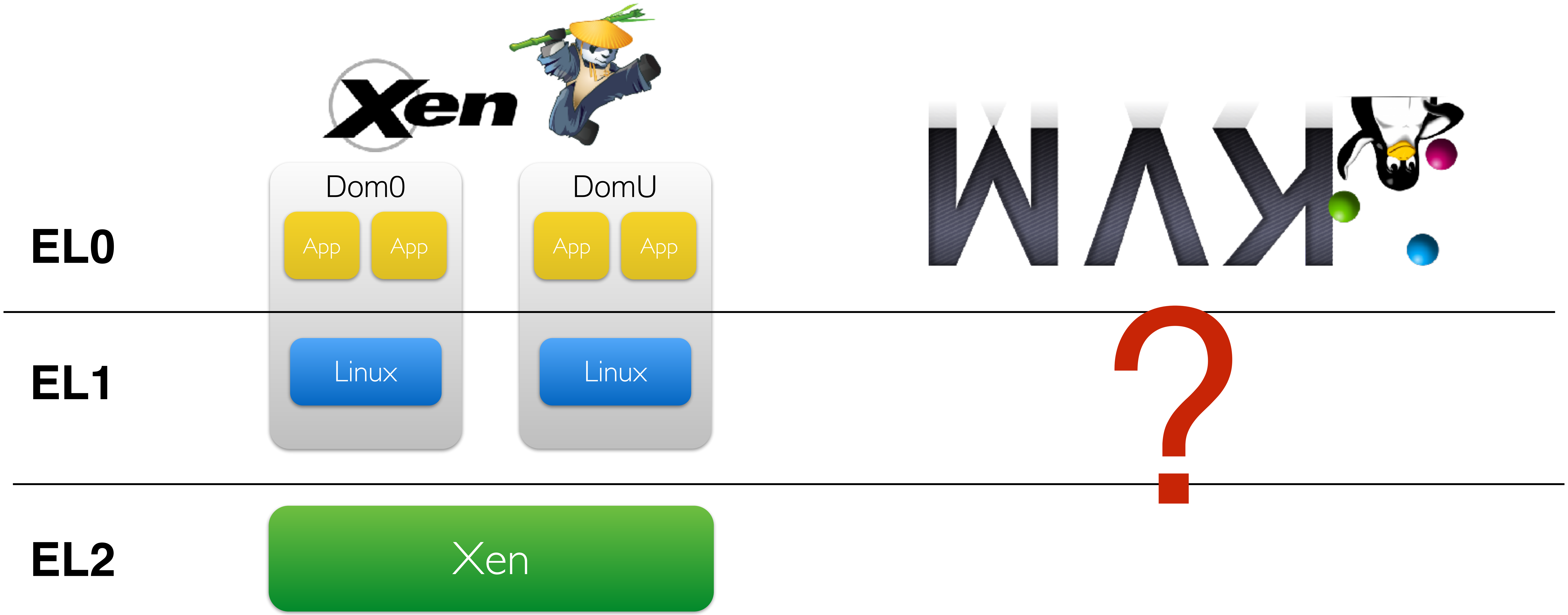
Hypervisor Design



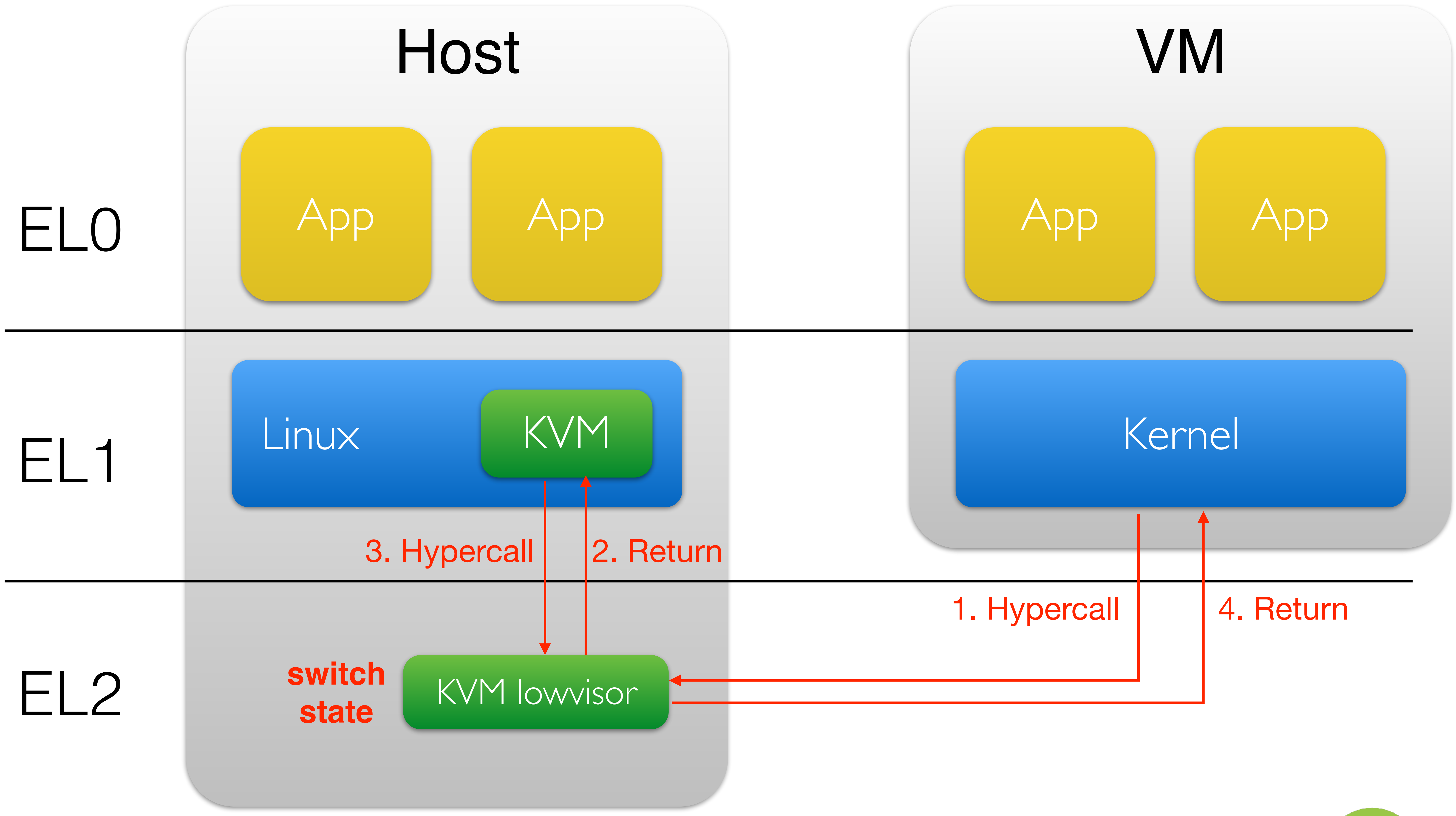
ARM Virtualization Extensions

EL0	User
EL1	Kernel
EL2	Hypervisor

ARM VE and Hypervisors

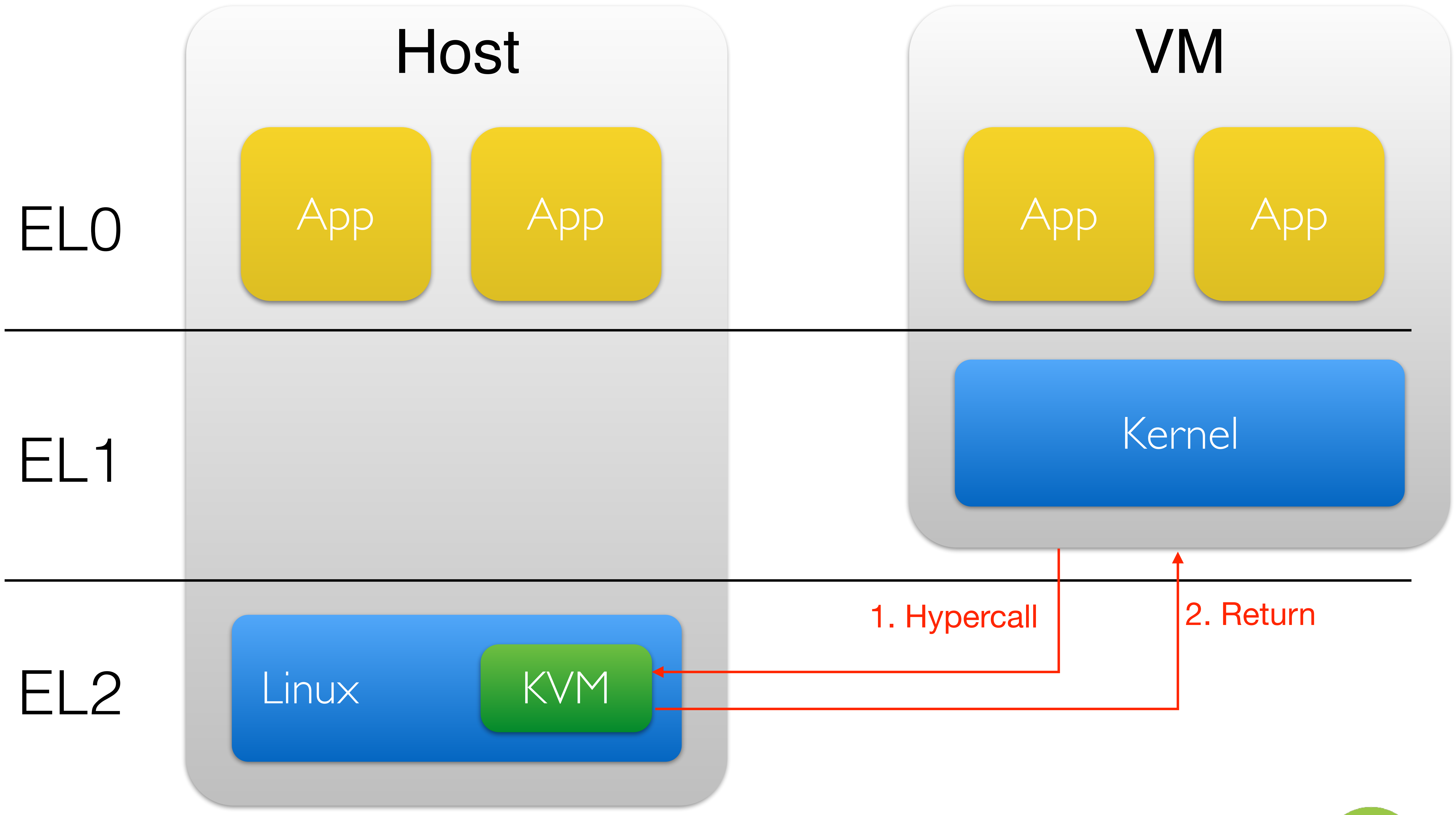


KVM/ARM



Host
VM
EL0
App
App
App
App
EL1
Linux
KVM
Kernel
3. Hypercall
2. Return
1. Hypercall
4. Return
EL2
switch state
KVM lowvisor

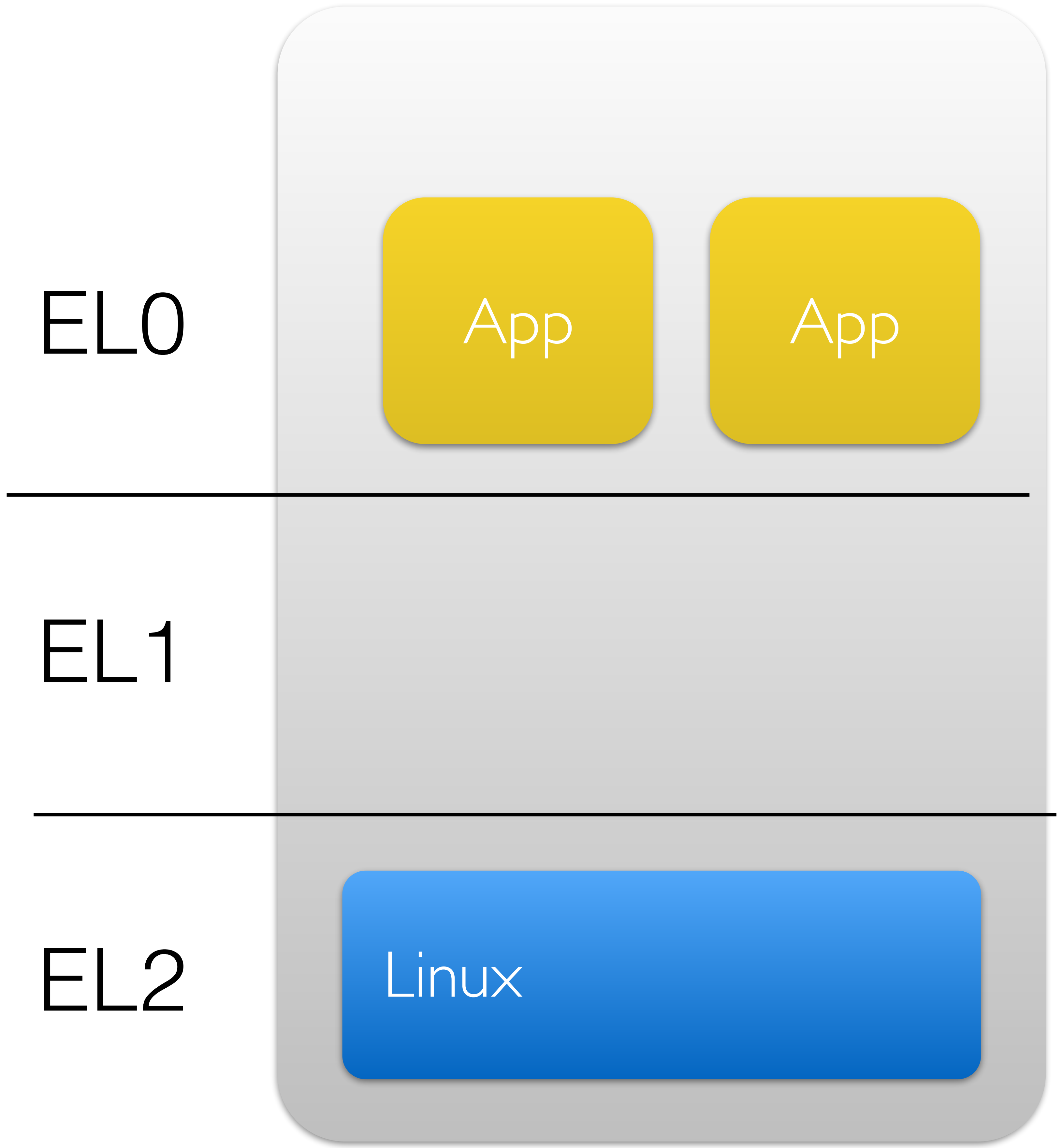
KVM/ARM



Host
VM
EL0
EL1
EL2
App
App
App
App
Kernel
Linux
KVM
1. Hypercall
2. Return

ARMv8.1 VHE

- Virtualization Host Extensions
- Supports running **unmodified** OSes in EL2 without using EL1



VHE: Backwards Compatible

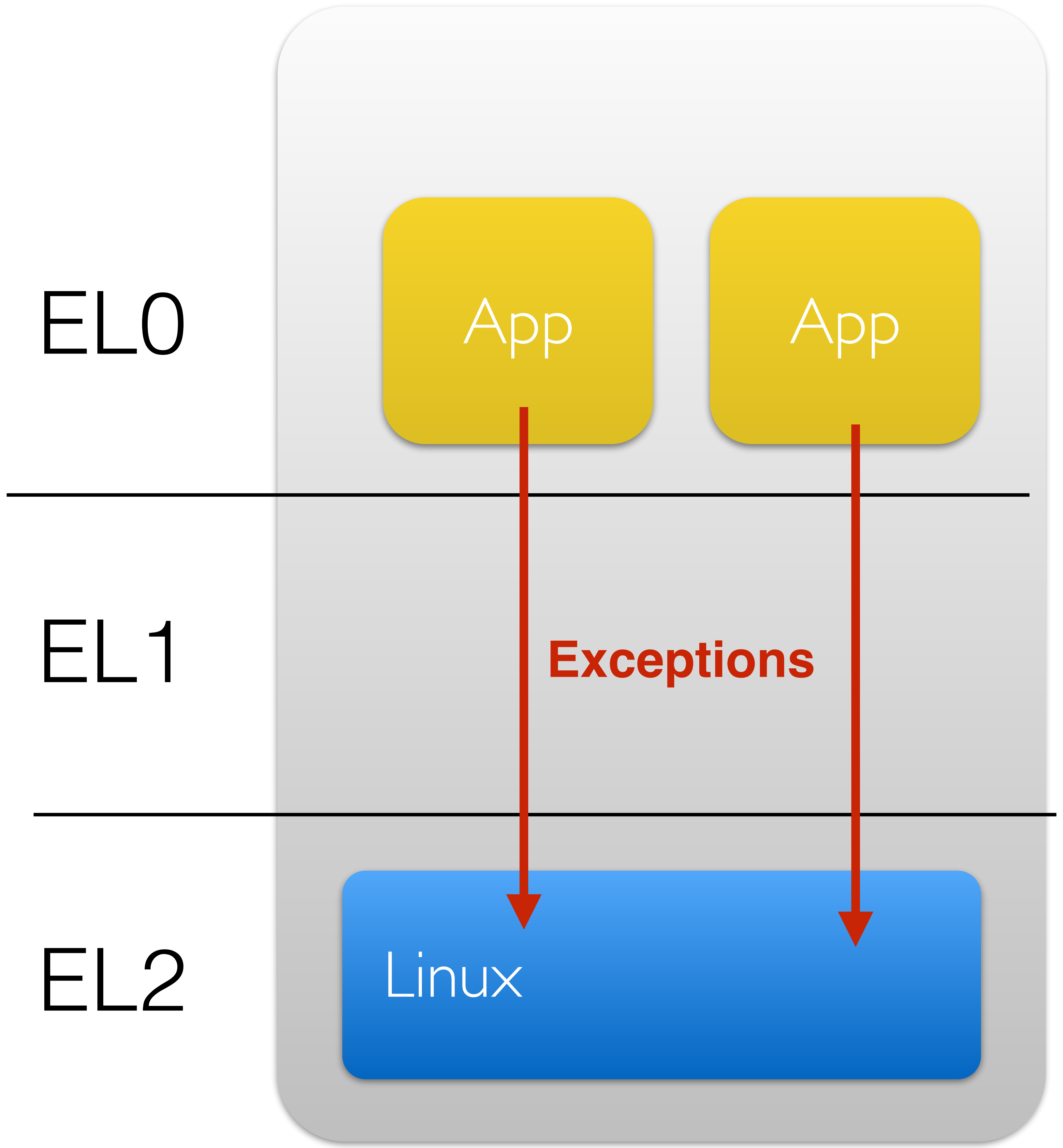
- HCR_EL2.E2H complete enables and disables VHE
- When disabled, completely backwards compatible with ARMv8.0
- Example: Xen disables VHE

VHE: Expands Functionality of EL2

- Expanded EL2 functionality
- New registers: TTBR1_EL2, CONTEXTIDR_EL2
- New virtual EL2 timer

VHE: Support Userspace in EL0

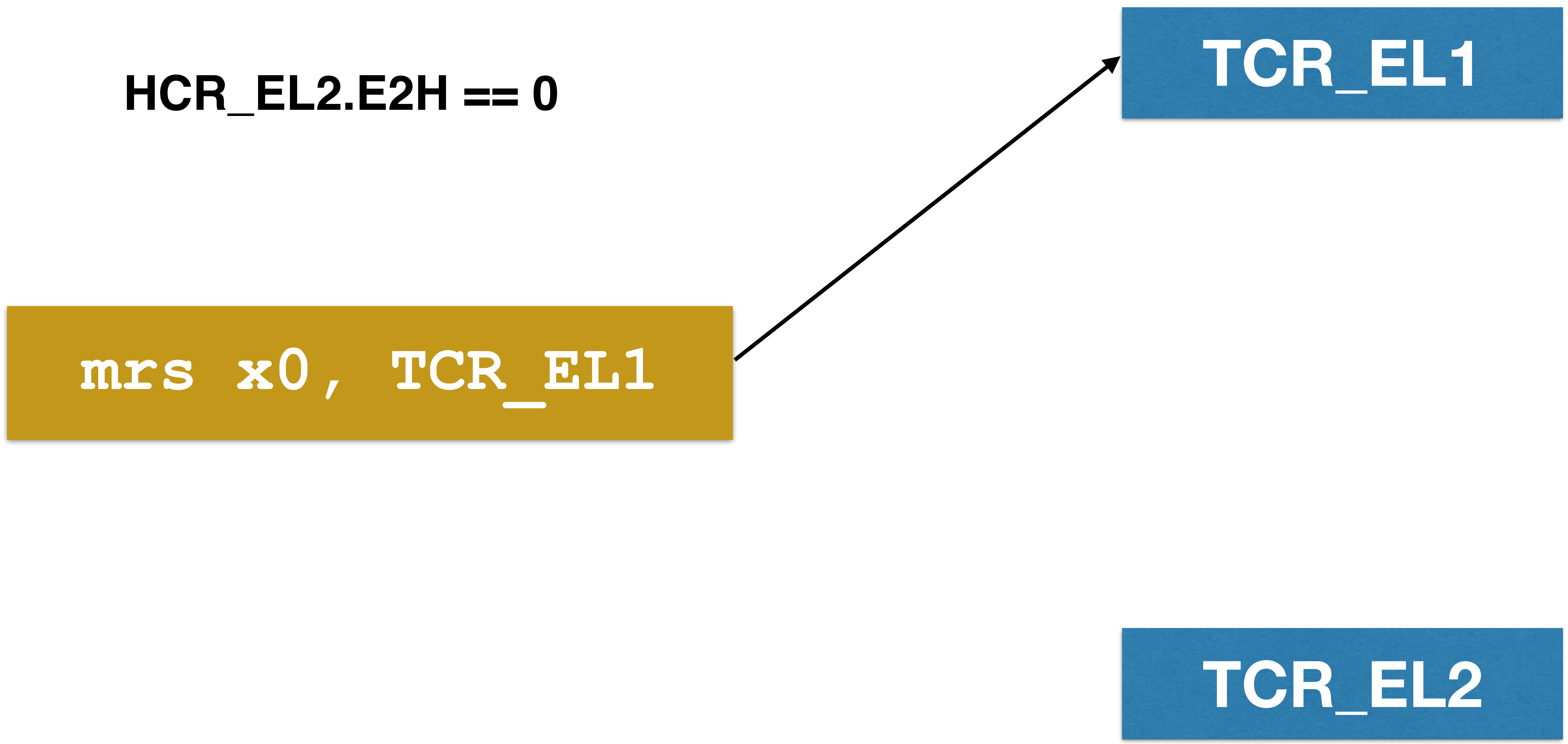
- TGE: Trap General Exceptions
- Routes all exceptions to EL2
- VHE no longer disables EL0 stage 1 MMU



VHE: EL2&0 Translation Regime

- Same page table format as EL1
- Used in EL0 with TGE bit set

VHE: System Register Redirection



VHE: System Register Redirection

HCR_EL2.E2H == 1

TCR_EL1

```
mrs x0, TCR_EL1
```

TCR_EL2

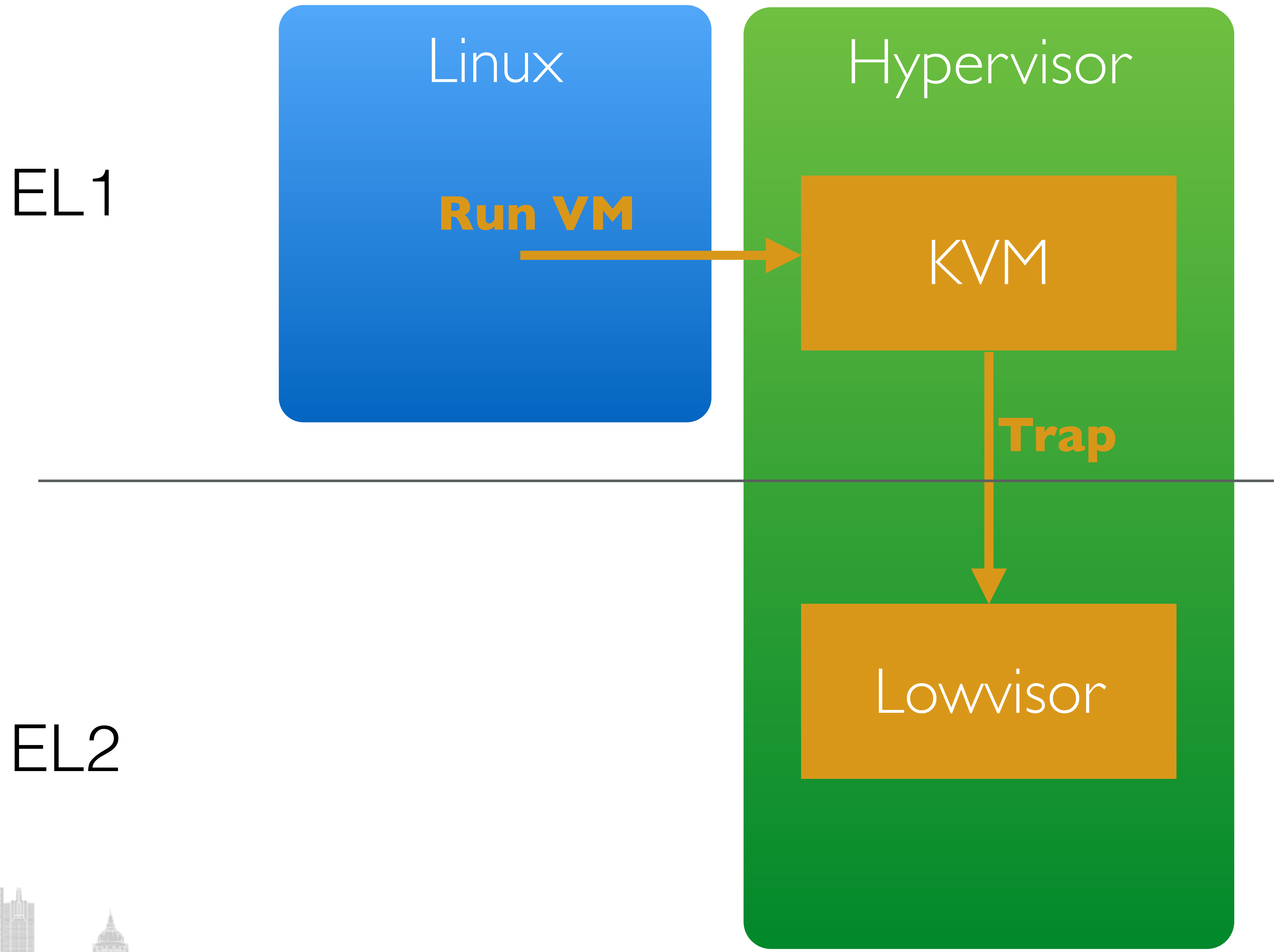
VHE Register Redirection

`mrs x0, TCR_EL12` → **TCR_EL1**

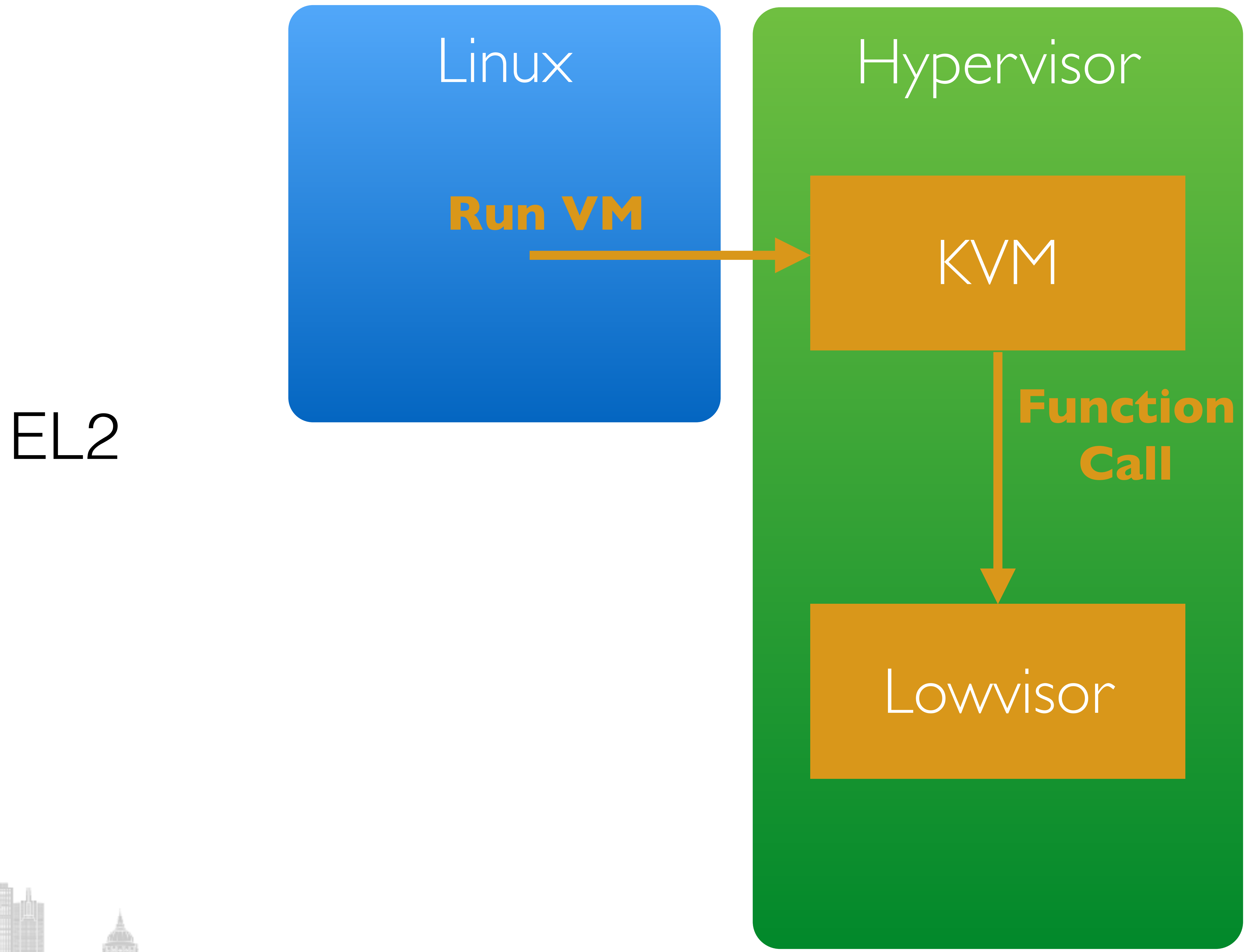
More VHE Register Redirection

- Some registers change bit position to be similar between EL1 and EL2
- Example: CNTHTCL_EL2 changes layout to match CNTKCTL_EL1 with extra bits

Legacy KVM/ARM without VHE



KVM/ARM with VHE



Experimental Setup

*Measurements obtained using Linux in EL2. See BKK16 talk.

- AMD Seattle B0
- 64-bit ARMv8-A
- 2.0 GHz AMD A1100 CPU
- 8-way SMP
- 16 GB RAM
- 10 GB Ethernet (passthrough)

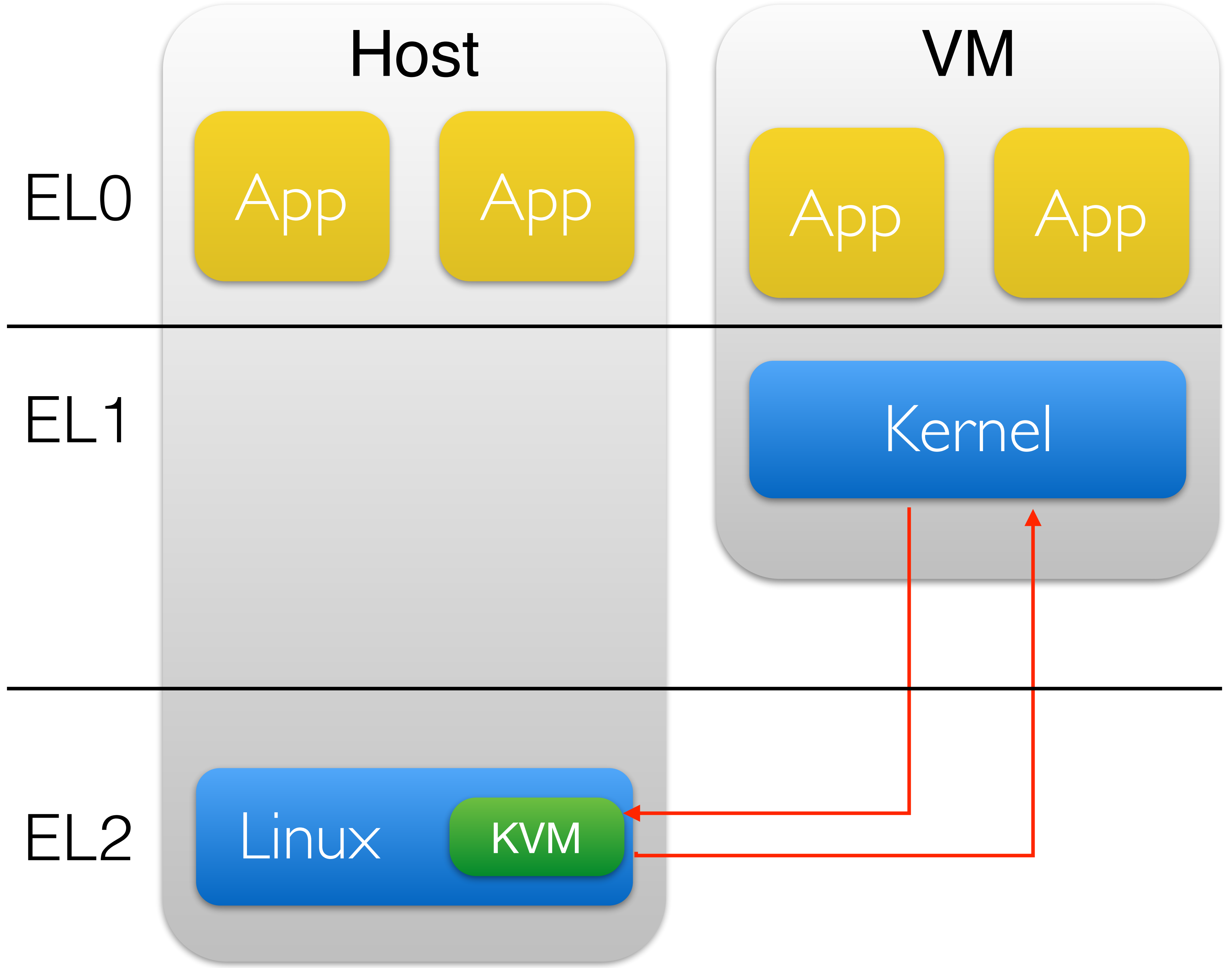
VHE Performance at First Glance

*Measurements obtained using Linux in EL2. See BKK16 talk.

CPU Clock Cycles	non-VHE	VHE*
Hypercall	3.181	3.045

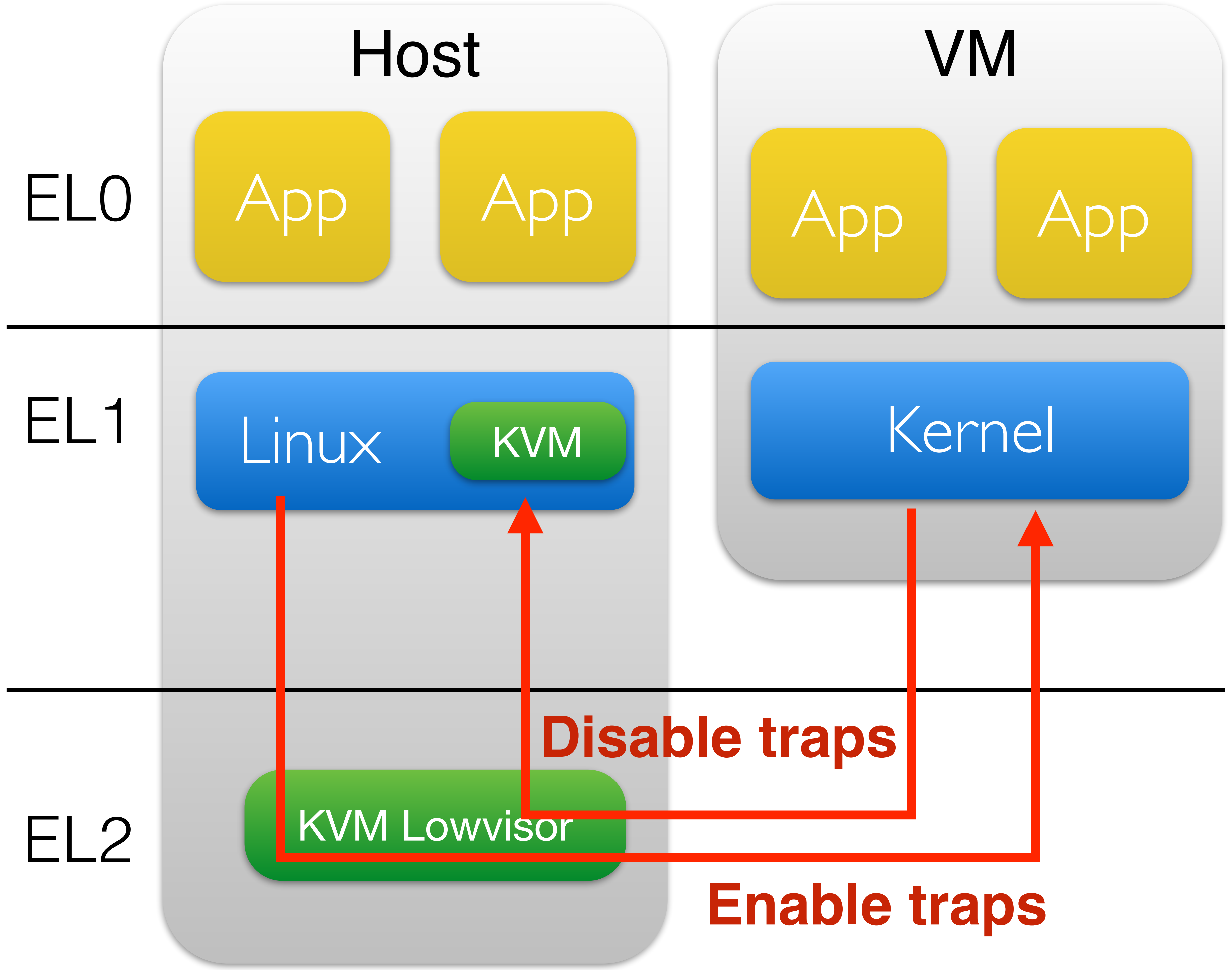
KVM/ARM Optimization #1

- Avoid saving/restoring EL1 register state



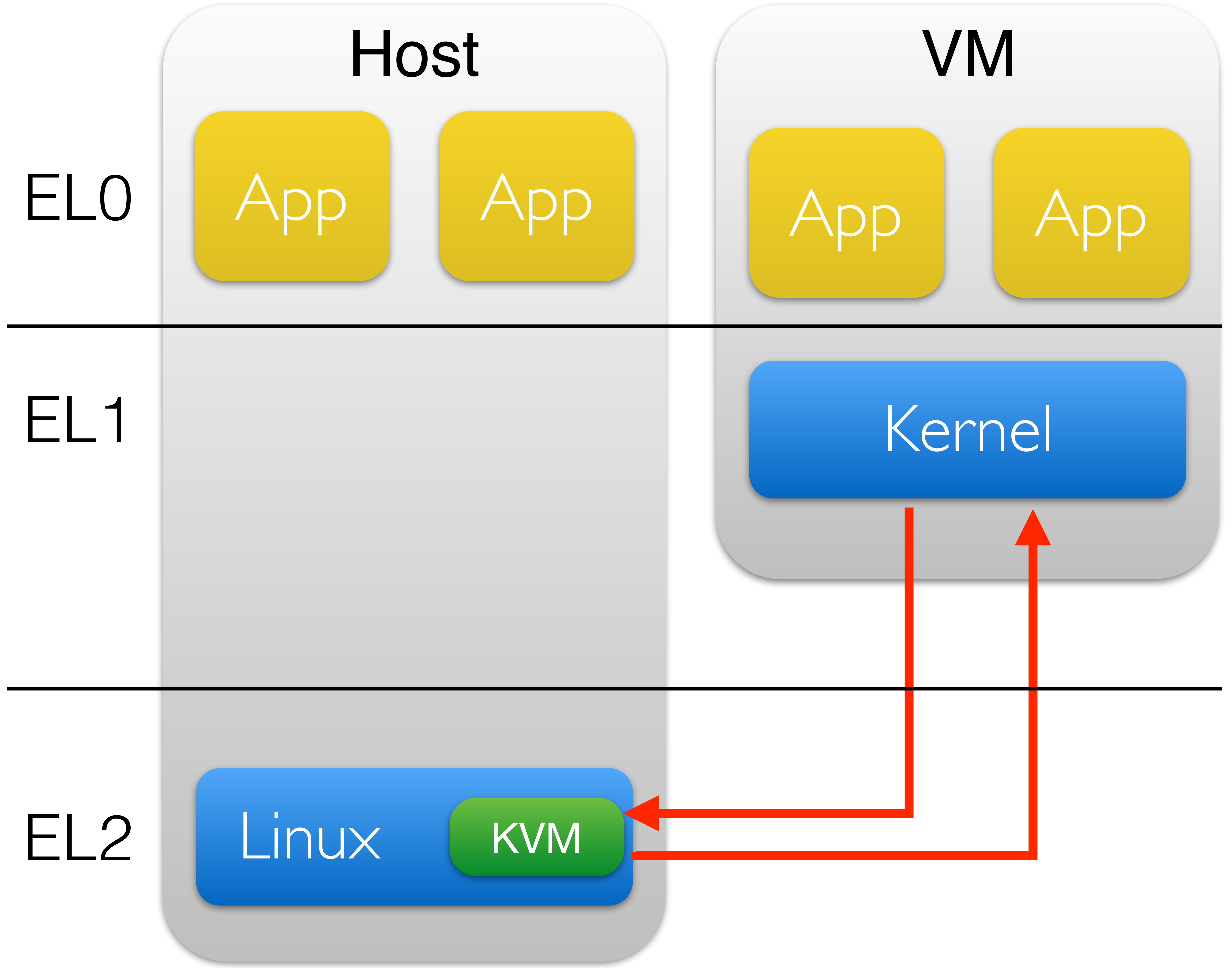
KVM/ARM Optimization #2

- Legacy KVM/ARM design enabled/disabled virtualization features on every transition
- Virtual/Physical interrupts
- Stage 2 memory translation



KVM/ARM Optimization #2

- Leave virtualization features enabled
- Host EL2 never uses stage 2 translations and always has full hardware access.

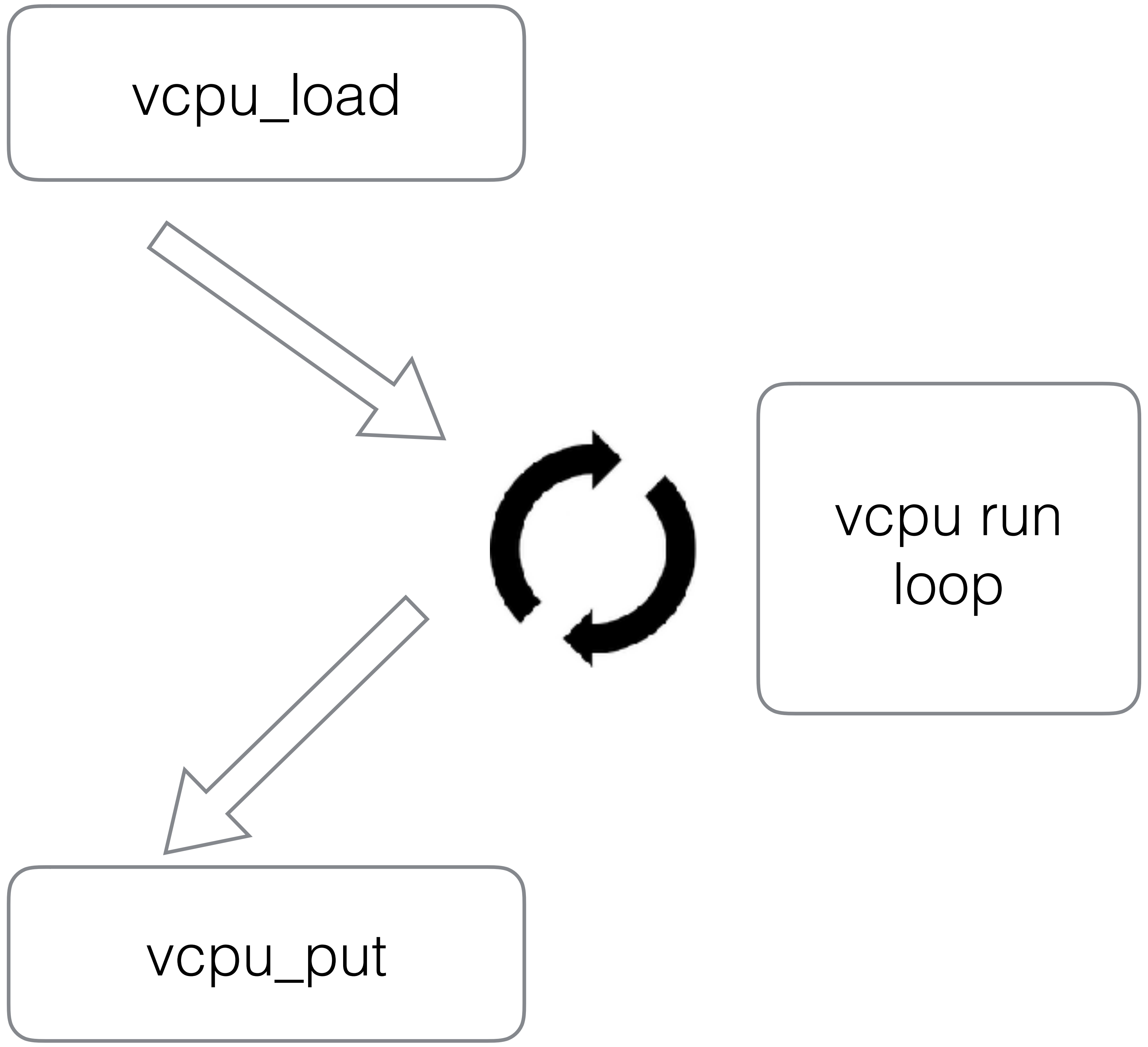


KVM/ARM Optimization #3

- Don’t context switch the timer on every exit from the VM
- Completely reworks the timer code
- 20 patches on list

KVM/ARM Optimization #4

- Reduce run loop work
- Do work in vcpu_load and vcpu_put instead
- Called when entering/exiting run-loop
- Called when preempted/scheduled
- Requires VHE



KVM/ARM Optimization #5

- Rewrite the world switch code

```
kvm_arch_vcpu_ioctl_run
{
  ...
  while (1) {
    ...
    if (has_vhe() /* static key */
      ret = kvm_vcpu_vhe_run(vcpu);
    else
      ret = kvm_call_hyp(__kvm_vcpu_run, vcpu);
    ...
  }
  ...
}
```

Microbenchmark Results

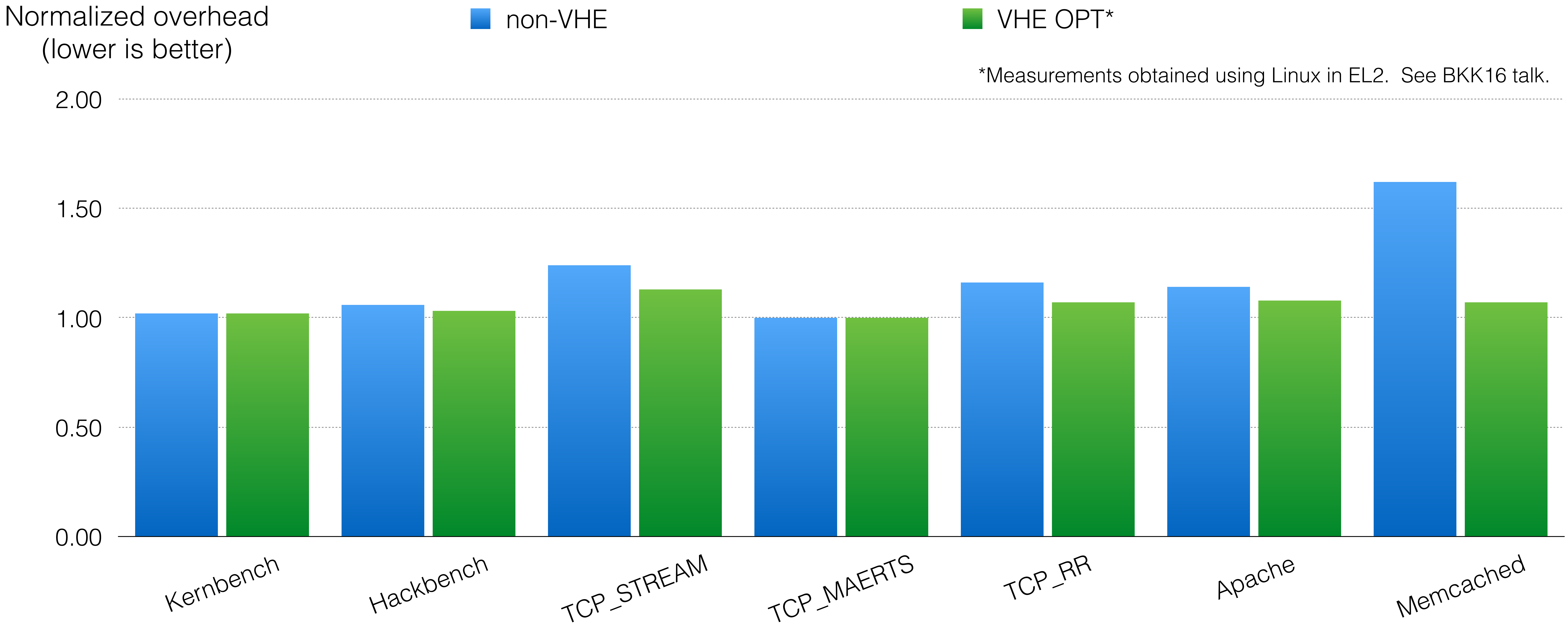
*Measurements obtained using Linux in EL2. See BKK16 talk.

CPU Clock Cycles	non-VHE	VHE OPT *	x86
Hypercall	3.181	752	1.437
I/O Kernel	3.992	1.604	2.565
I/O User	6.665	7.630	6.732
Virtual IPI	14.155	2.526	3.102

Application Workloads

Application	Description
Kernbench	Kernel compile
Hackbench	Scheduler stress
Netperf	Network performance
Apache	Web server stress
Memcached	Key-Value store

Application Workloads



Conclusions

- Optimize and redesign KVM/ARM for VHE
- Reduce hypercall overhead by more than 75%
- Better cycle counts than x86 for key hypervisor operations
- Network benchmark overhead reduced by 50%
- Key-value store workload overhead reduced by more than 80%

Upstream Status

- Timer patches on list
- Core optimization patches coming soon