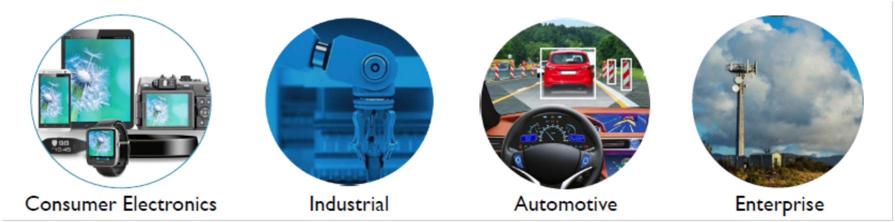
The evolution of Virtualization in the Arm Architecture

Julien Grall <julien.grall@arm.com>

Xen Developer Summit 2018

Use cases



Type of hypervisors

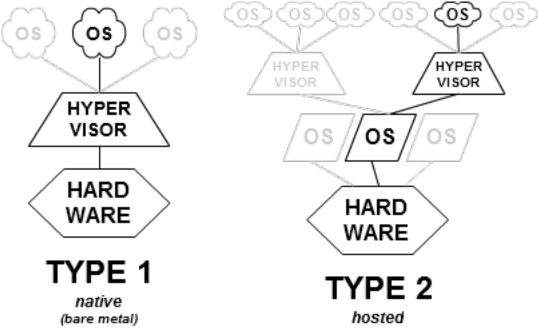
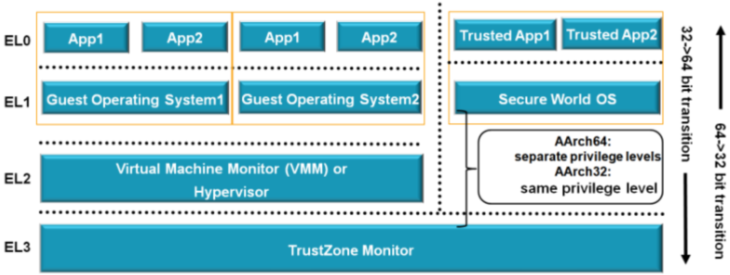


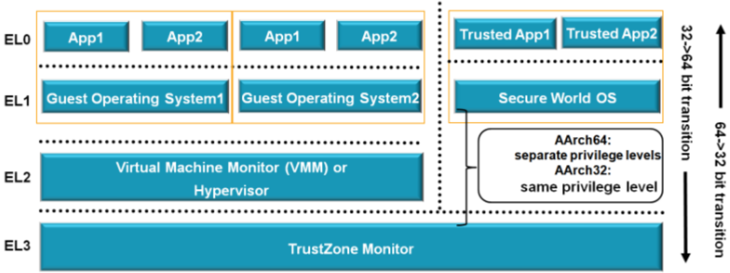
Figure: From wikipedia

ARMv8-A Privilege Model



- Support both AArch32 and AArch64 execution modes
- 32-64bit inter-working limited to exception boundaries
- AArch64 always has a higher privilege than AArch32
- AArch64 state is a superset of AArch32 state

ARM virtualization



- Introduced with the latest version of ARMv7 architecture
- New hypervisor execution state
- Non-Secure world, higher privilege than EL1

Virtualization in a nutshell

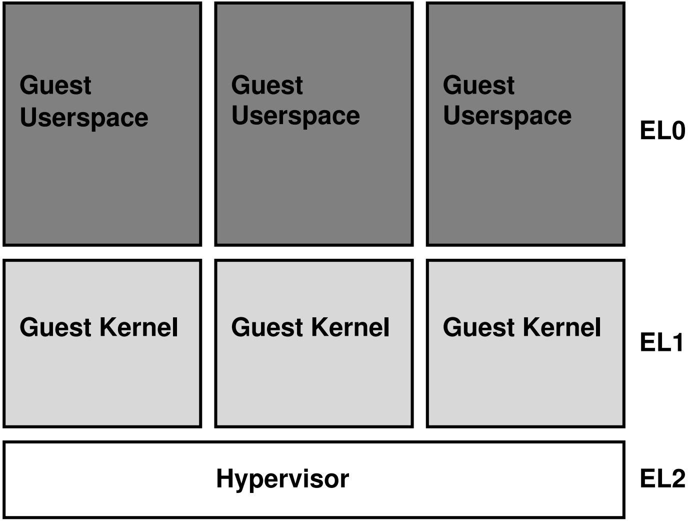
- Second stage of memory translation
  - Adds an extra level of indirection between guests and physical memory
  - TLBs are tagged by Virtual Machine ID (VMID)
- Ability to trap access of most system registers
  - The hypervisor decides what it wants to trap
- Can handle IRQs, FIQs and asynchronous aborts
  - The guest doesn’t see physical interrupts firing, for example
- Guests can call into EL2 mode (HVC instruction)
  - Allows para-virtualizated services
- Standard architecture peripherals are virtualization-aware
  - GIC and timer have specific features to help virtualization

EL2: Not EL1++ (ARMv8.0-A)

- EL2 is not a superset of NS-EL1
  - Orthogonal mode to EL1
  - Allows multiplexing of NS-EL1 guests on the hardware
- Own translation regime
  - Separate Stage-1 translation, no Stage-2 translation
- It would be difficult to run Linux in EL2
  - Requires too many changes to be practical
- EL2 could be used as a ”world switch”
  - Between guests (baremetal hypervisor/Type I)
  - Between host and guest (hosted hypervisor/Type II)

    This makes the host a form of specialized guest.

Hypervisor architecture - Type I

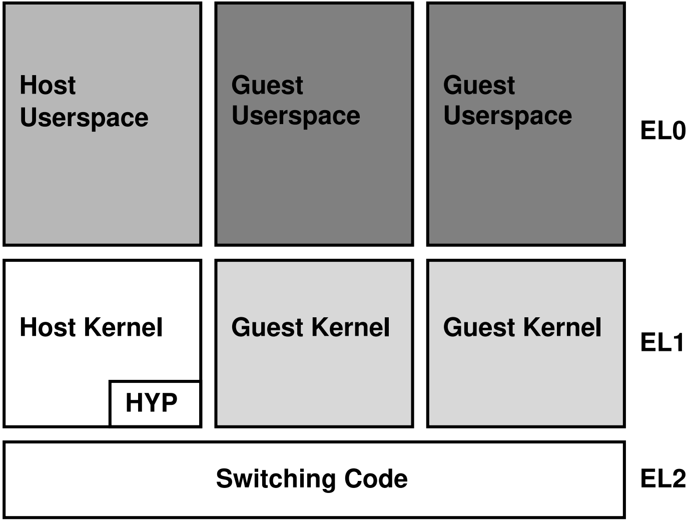


Guest Userspace | Guest Userspace | Guest Userspace — EL0

Guest Kernel | Guest Kernel | Guest Kernel — EL1

Hypervisor — EL2

Hypervisor architecture - Type II



Host Userspace

Guest Userspace

Guest Userspace

EL0

Host Kernel

HYP

Guest Kernel

Guest Kernel

EL1

Switching Code

EL2

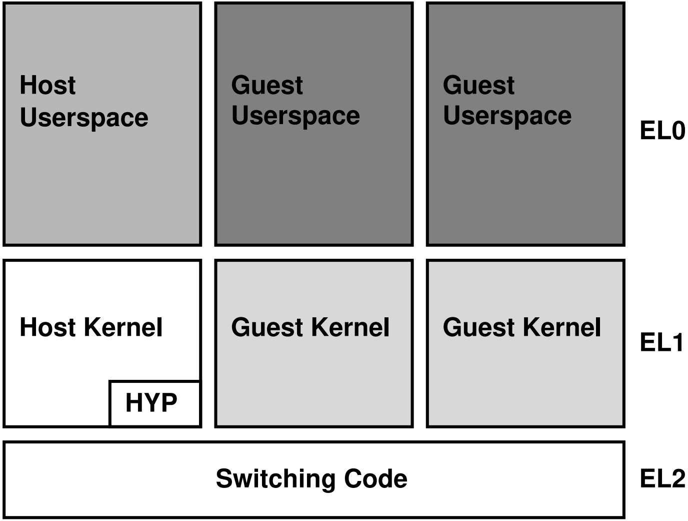
EL2 enhancement (ARMv8.1-A)

The Virtualization Host Extension (VHE) expands the capability of EL2:

- Designed to improve the support of the Type-2 hypervisors
- Allows the host OS to be run at EL2
- The host OS requires minimal changes to run at EL2
- User-space still runs at EL0
- Host has no software running at EL1
- AArch64 specific

EL2 becomes a strict superset of EL1

Hosted hypervisor architecture on a platform without VHE



Host Userspace

Guest Userspace

Guest Userspace

EL0

Host Kernel

HYP

Guest Kernel

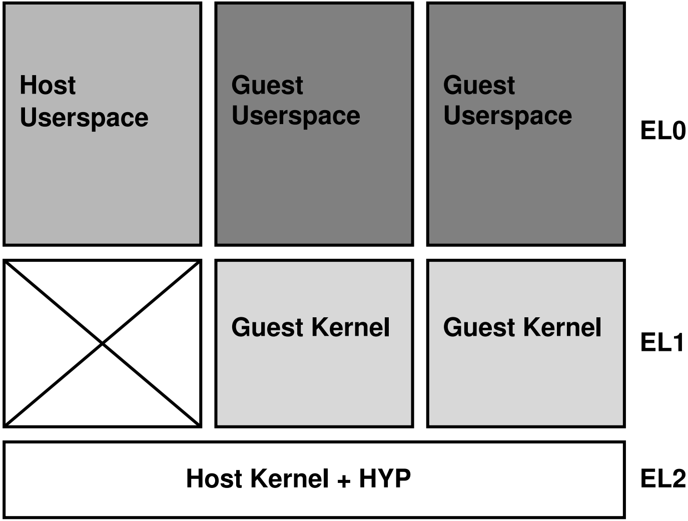
Guest Kernel

EL1

Switching Code

EL2

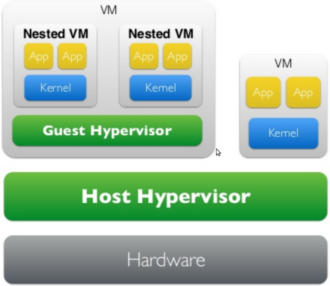
Hosted hypervisor architecture on a platform with VHE

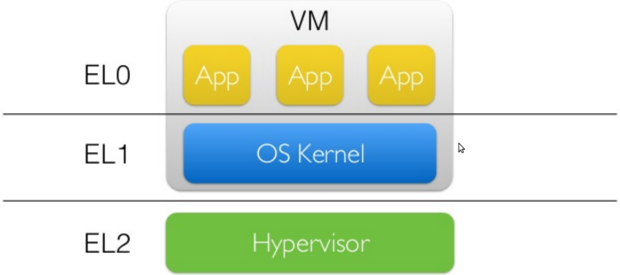


Nested Virtualization (ARMv8.3-A)

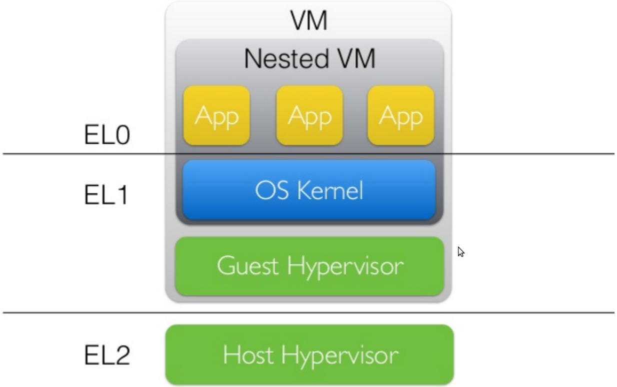
The Nested Virtualization extension allows a hypervisor in a VM.

- Unmodified guest hypervisor running in NS EL1
- Implementation of a host hypervisor required
  - Running at EL2
- AArch64 specific

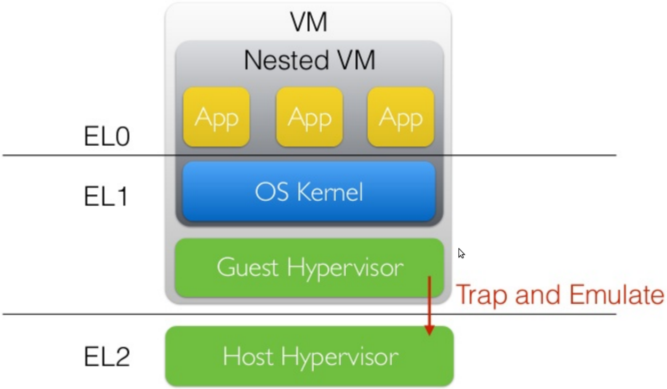



VM
EL0
App
App
App
EL1
OS Kernel
EL2
Hypervisor

Nested Virtualization on Arm



Nested Virtualization (ARMv8.3-A)



Nested Virtualization (Armv8.4.A)

ARMv8.4 extends Nested Virtualization to:

- Reduce the number of traps
- Improve performance of nested hypervisor

Memory Partitioning And Monitoring (ARMv8.3-A)

The Memory Partitioning And Monitoring (MPAM) allows:

- to limit the memory system performance impact of a VM
  - provide more bandwidth to some processes

A hypervisor can monitor and control how VM uses:

- memory of a system
- communicate with other system components

Summary

- Robust set of virtualization features
  - Not just about CPU virtualization
  - Covers the whole systems architecture
- An architecture in motion:
  - ARMv8.1-A: `https://goo.gl/Ox4thV`
  - ARMv8.2-A: `https://goo.gl/0Ns37U`
  - ARMv8.3-A: `https://goo.gl/CJv1n0`
  - ARMv8-4-A: `https://goo.gl/tYZmhR`

Questions?

arm