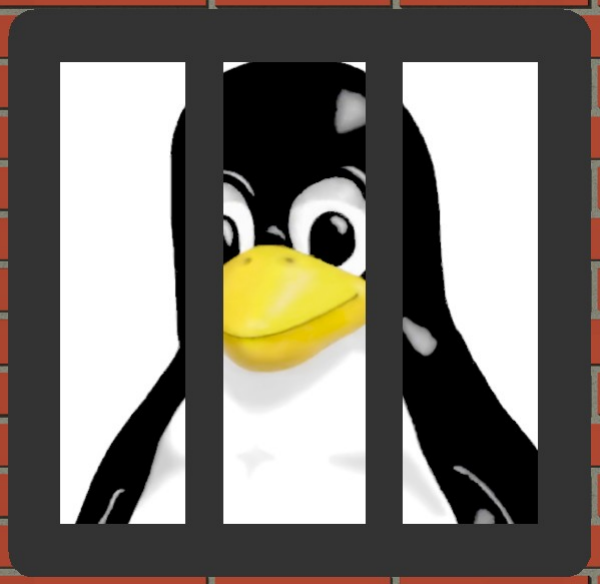
SIEMENS



Siemens Corporate Technology | August 2015

Hard Partitioning for Linux: The Jailhouse Hypervisor

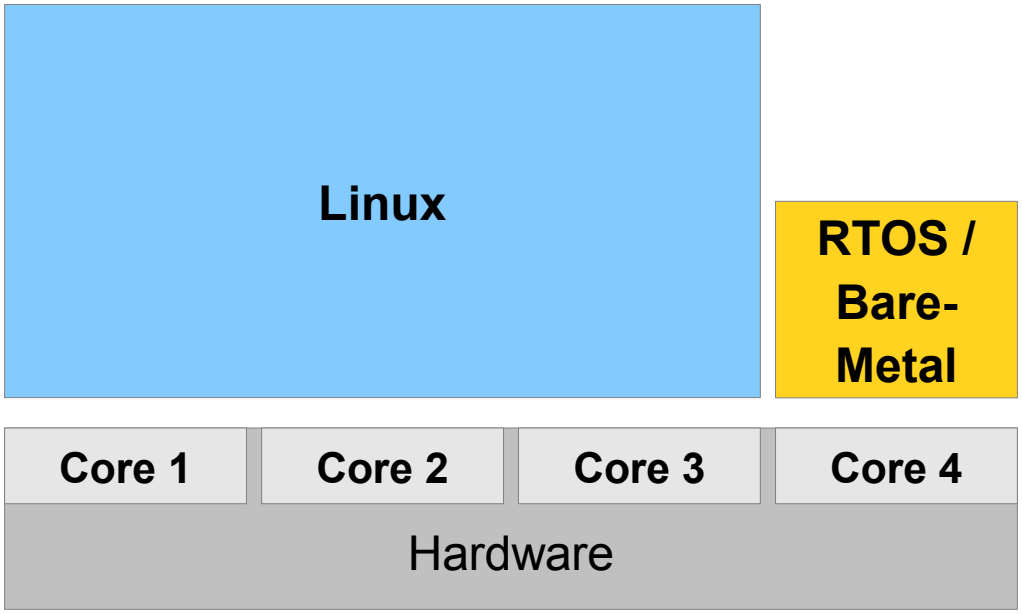
Hard Partitioning for Linux: The Jailhouse Hypervisor

Agenda

Motivation
Jailhouse introduction & philosophy
Current status (with demo)
Configuration
Running multiple Linux instances (with demo)
Summary

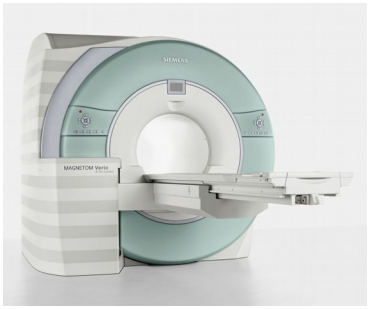
SIEMENS

Asymmetric Multi-Processing (AMP) & Linux



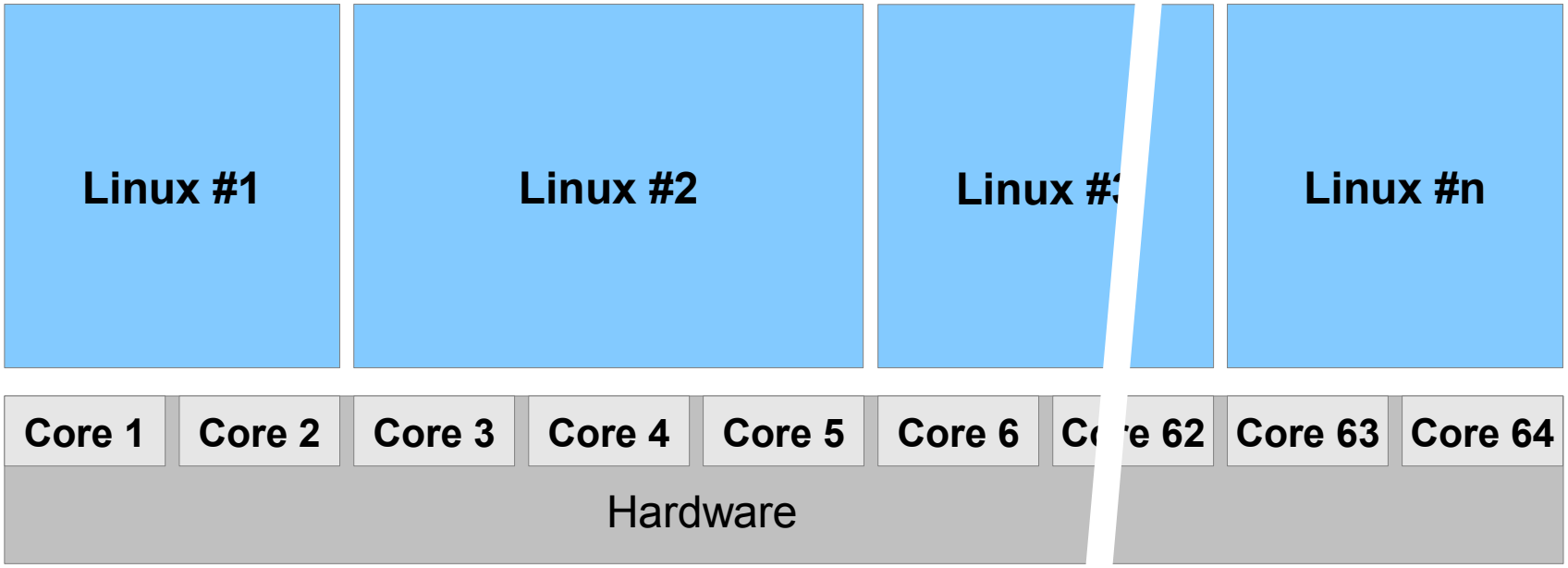
AMP Drivers

- **Low latency & high throughput**
- **Hard real-time**
- **Preexisting software**
- **Mixed criticality**



SIEMENS

AMP for Linux?



Further AMP Drivers

- Consolidation of (Linux) services
- Enabling of open platforms
- Low latency & high throughput
- Avoid SMP scalability bottlenecks
- Strict quality of service needs

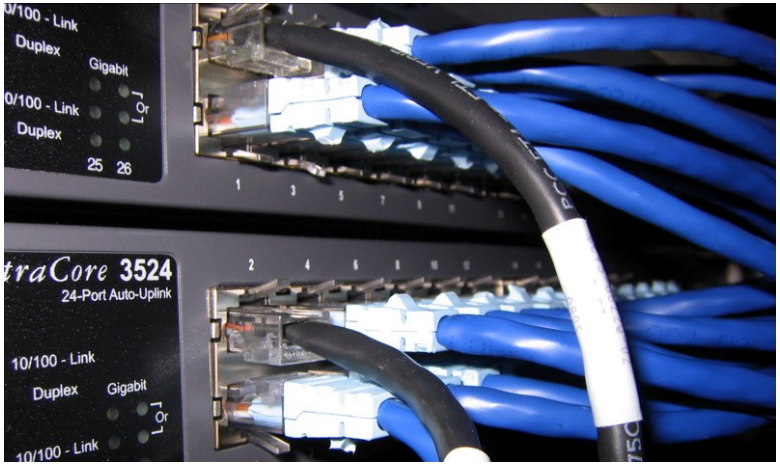


Image: Ben Stanfield, licensed under CC BY-SA 2.0

Hard Partitioning for Linux: The Jailhouse Hypervisor

Agenda

- Motivation
- **Jailhouse introduction & philosophy**
- Current status (with demo)
- Configuration
- Running multiple Linux instances (with demo)
- Summary

SIEMENS

What is Jailhouse?

A tool to run

… real-time and/or safety tasks

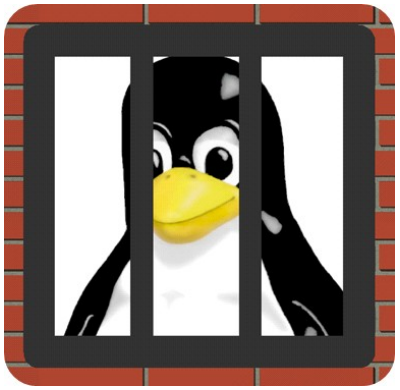
… on multicore platforms (AMP)

… aside Linux

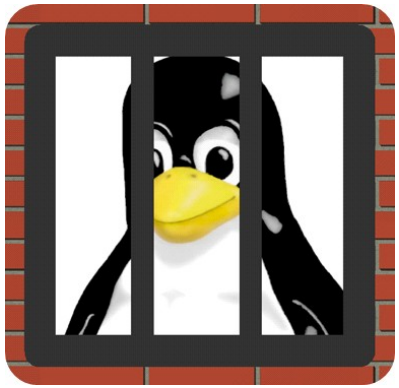
It provides

- **strong & clean isolation**
- **bare-metal-like performance & latencies**
- **no reason to modify Linux** (well, almost)

… and it's open source (GPLv2)



What is it not?



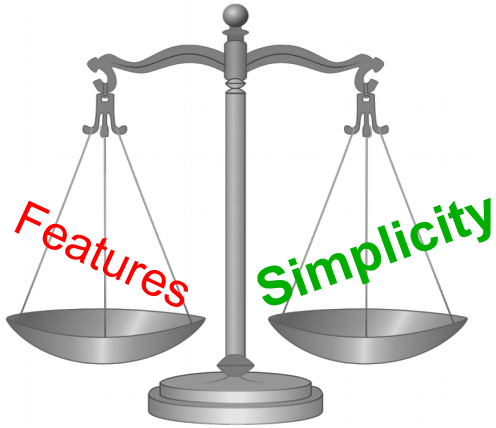
!=

...

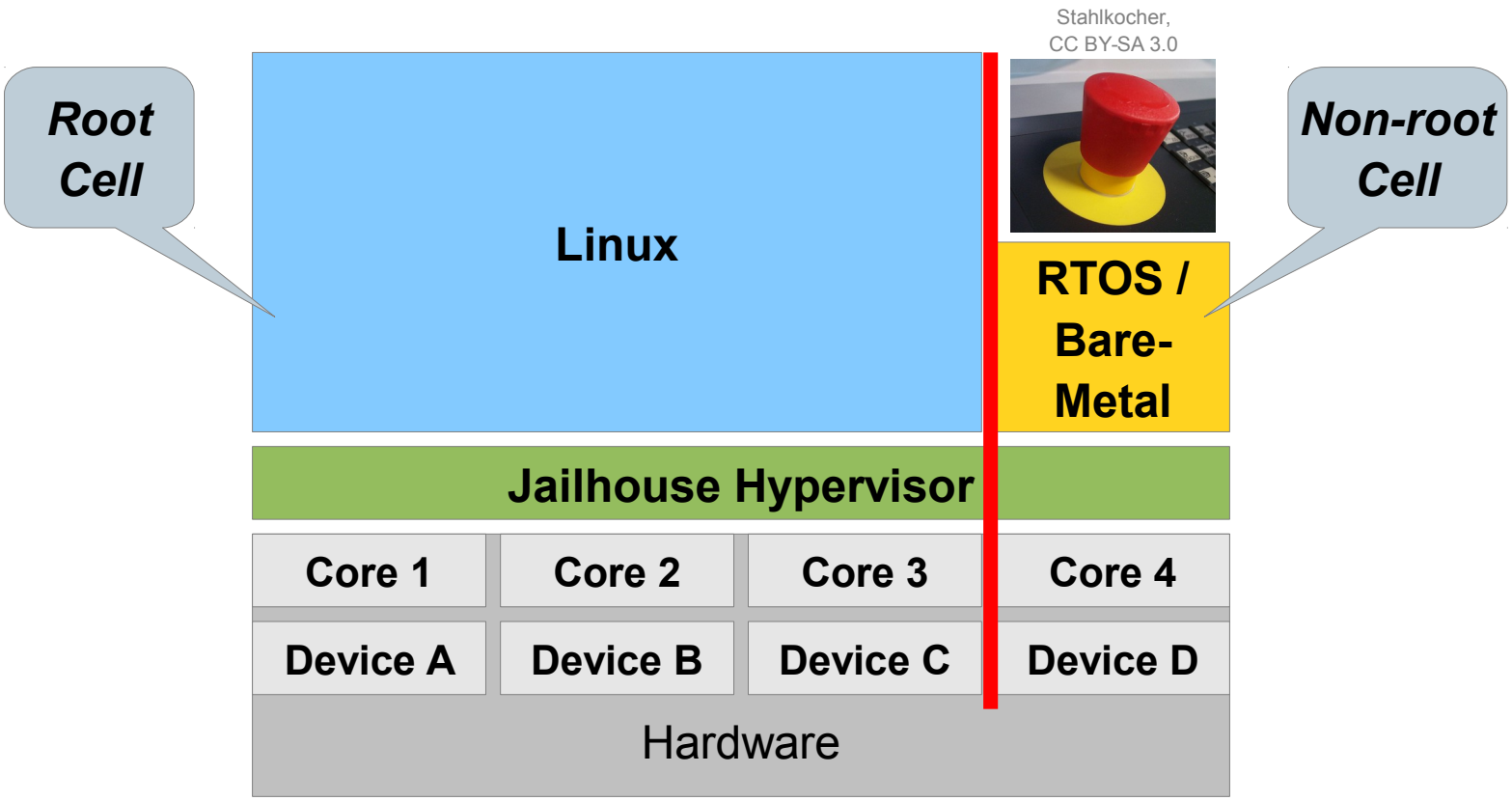
SIEMENS

What makes Jailhouse different?

- **Use virtualization for isolation – *ok, nothing new***
- **Prefer simplicity over features**
  - Resource access control
    instead of resource virtualization
  - 1:1 resource assignment
    instead of scheduling
  - Partition booted system
    instead of booting Linux
  - Do not hide existence of Jailhouse
- **Offload work to Linux**
  - System boot
  - Jailhouse and partition (“cell”) loading & starting
  - Control and monitoring



AMP with Jailhouse



SIEMENS

Hard Partitioning for Linux: The Jailhouse Hypervisor

Agenda

- Motivation
- Jailhouse introduction & philosophy
- **Current status (with demo)**
- Configuration
- Running multiple Linux instances (with demo)
- Summary

SIEMENS

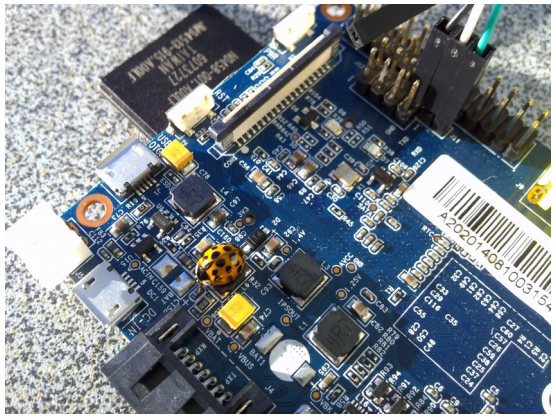
Jailhouse Status – x86

- **Initial focus on x86, first Intel, then AMD**
  - Requirement: VT-x / VT-d, AMD-V
  - AMD IOMMU support to be published soon
- **It's small!**
  - Currently ~8.5K lines of code (for Intel)
- **Direct interrupt delivery**
  - Zero VM exits, minimal latencies feasible
  - Max. timer IRQ latency (Xeon D-1540): **<2.5 µs**
- **Cache Allocation Technology**
  - Intel feature for partitioning L3 cache
  - Code ready to be merged, measurements pending

SIEMENS

Jailhouse Status – ARM

- **ARMv7**
  - Runs in FastModel, on Banana-Pi, NVIDIA Jetson TK1
  - SMMU on to-do list
- **It's small too!**
  - Currently ~6.5k lines of code (for TK1)

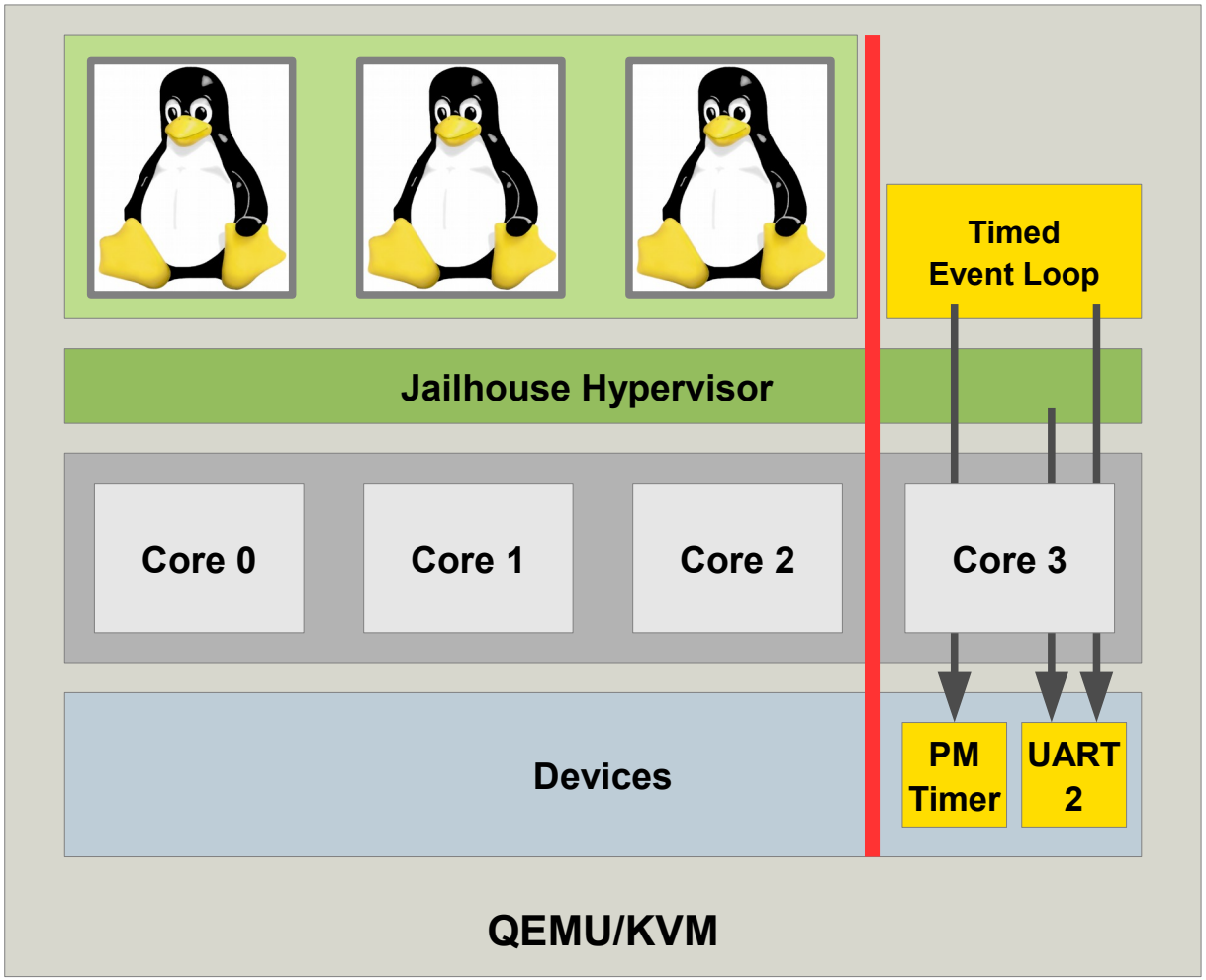


- **ARMv8**
  - First patches just posted by Huawei
  - Not yet working, but progressing quickly
  - Target: ARMv8 Foundation Model

SIEMENS

Live Demonstration, Part I

Running Jailhouse in a virtual machine?!



Hard Partitioning for Linux: The Jailhouse Hypervisor

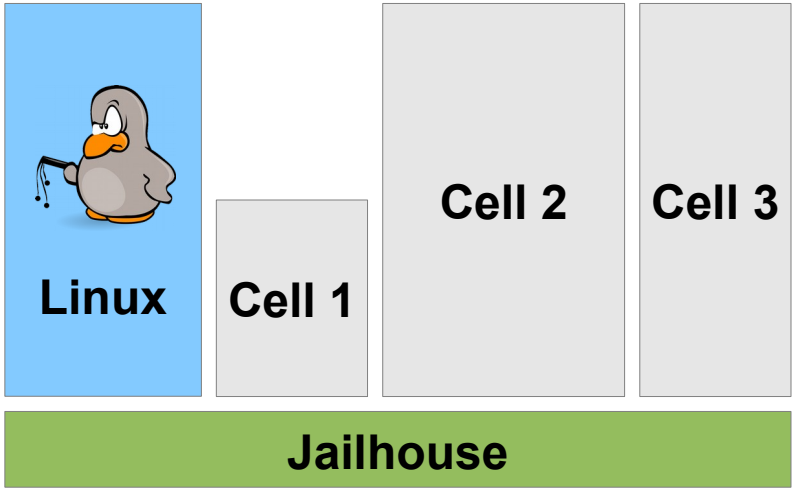
Agenda

- Motivation
- Jailhouse introduction & philosophy
- Current status (with demo)
- **Configuration**
- Running multiple Linux instances (with demo)
- Summary

SIEMENS

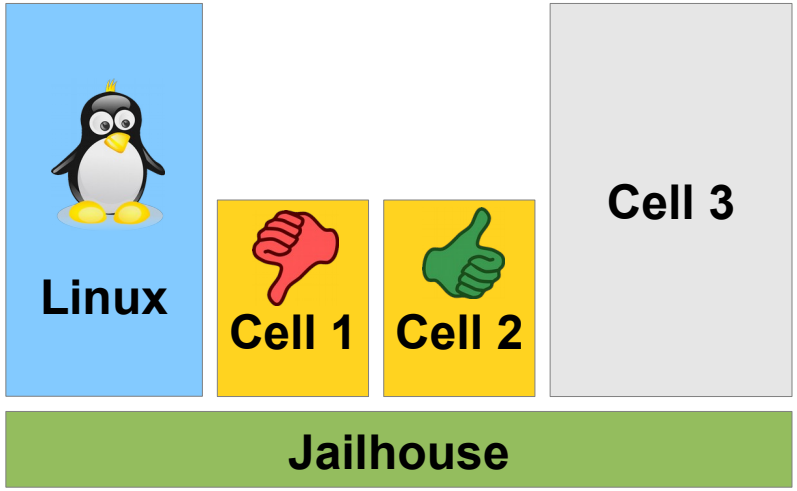
Two Management Models

Open Model



- Linux (root cell) is in control
- Cells not involved in management decisions
- Sufficient if root cell is trusted

Safety Model

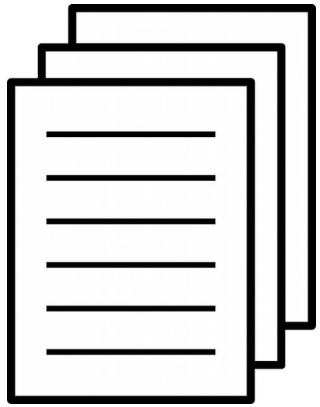


- Linux controls, but...
- Certain cells are configured to vote over management decisions
- Building block for safe operation

SIEMENS

Configuration

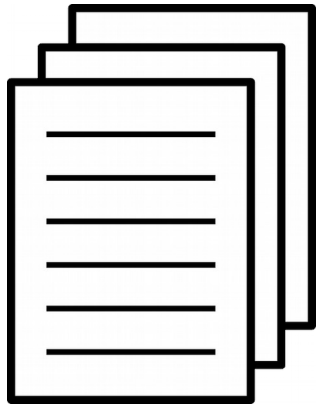
- **Requirements on raw format**
  - Easy processing by hypervisor
  - Fine-grained control
- **System configuration => for starting Jailhouse**
  - Hypervisor reservation
  - Relevant platform resources (ACPI / device tree on diet)
  - Root cell description
- **Cell description => for creating cells**
  - Available resources (CPUs, memory, PCI devices, IRQ lines, cache, …)
- **Creation process**
  - `jailhouse config create my-system.c`
  - Manual review / post-processing
  - Compile: `my-system.c` → `my-system.cell`
- **Cell config: derive from system config**



SIEMENS

Future of Jailhouse Configuration

- **In a nutshell**
  - Precise & flexible
  - ...but not yet convenient
- **Improvements under discussion**
  - Cell config generation
  - Better consistency checks
  - Format revisiting: rebase over device tree?
  - ARM will drive this, but x86 may join
- **Based on improved low-level automation**
  - libvirt support



Hard Partitioning for Linux: The Jailhouse Hypervisor

Agenda

- Motivation
- Jailhouse introduction & philosophy
- Current status (with demo)
- Configuration
- **Running multiple Linux instances (with demo)**
- Summary

SIEMENS

Linux As Non-Root Cell?

"Can you also run Linux partitions?"

"Would be too much patching of Linux."

"Why not using Xen PV interfaces? Linux already supports it."

"I try to get a linux kernel run as inmate on Jailhouse [on ARM]."

OK, let's think about this again...

SIEMENS

Looking at x86...

Regular PC

- All devices available
  (APIC, IOAPIC, PIT, PIC, UARTs, CMOS, RTC, PCI bridges...)
- Resource discovery via ACPI
- Sophisticated handover from BIOS / UEFI

Jailhouse non-root cell

- Only subset of devices
  - APIC (with certain addressing restrictions) → IPI, MSI, timer
  - IOAPIC pins (if assigned)
  - UART (if assigned)
  - PM Timer (as reference clock)
  - TSC (as fast-past clock)
  - Selected PCI devices, no bridges
- Basic resource discovery via communication region
  - PM timer address
  - Number of logical CPUs

A Simpler Solution Than Expected

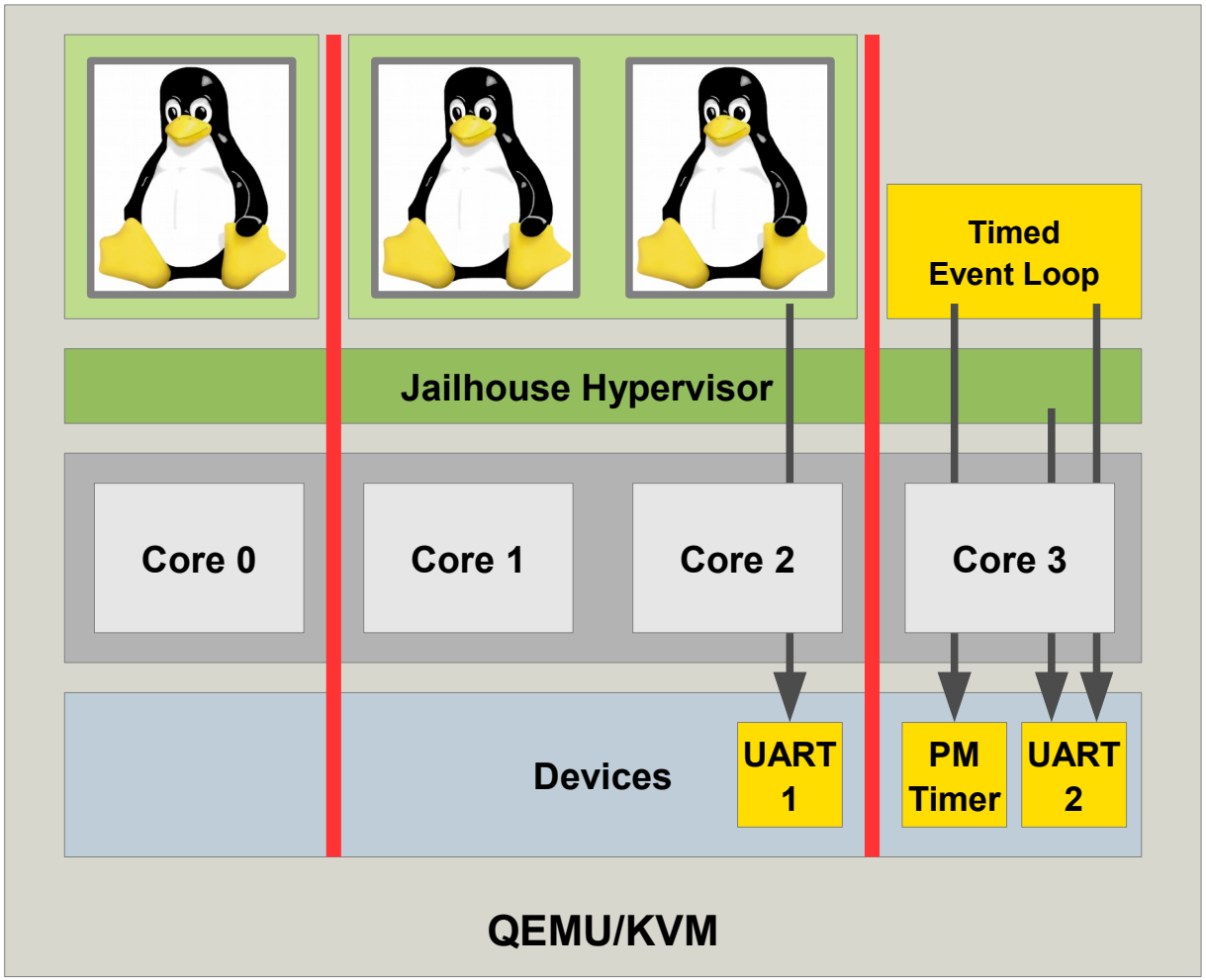
So, we *do* need to patch a lot?

No!

```
arch/x86/Kconfig                          |  10 +
arch/x86/include/asm/hypervisor.h         |   1 +
arch/x86/include/asm/jailhouse_para.h     |  27 ++
arch/x86/kernel/Makefile                  |   2 +
arch/x86/kernel/apic/apic_flat_64.c       |  12 +-
arch/x86/kernel/cpu/hypervisor.c          |   3 +
arch/x86/kernel/jailhouse.c               | 229 ++++++++++++++
arch/x86/kernel/smpboot.c                 |   7 +-
8 files changed, 286 insertions(+), 5 deletions(-)
```

Live Demonstration, Part II

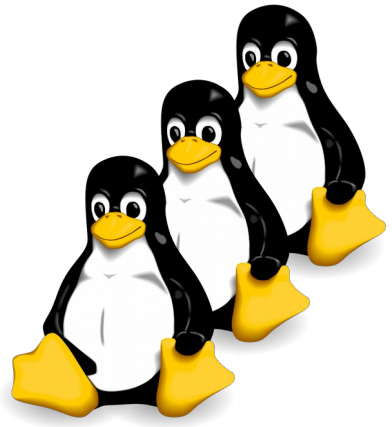
Jailhouse booting Linux



SIEMENS

Status of Non-Root Linux Cells

- **x86**
  - Assignment of MSI/MSI-X PCI devices working, no legacy INTx
  - SMP working
  - Inter-cell shared memory working
- **ARM**
  - Partly easier due to device tree description
  - Pitfall: shared resources (e.g. clock gate control on Banana Pi)
  - No publicly available reference setup so far
- **Common limitation: no virtual consoles yet**
  - Could be built on top of ivshmem
  - Or we add virtio compatibility to Jailhouse



SIEMENS

Inter-Cell Communication

- **ivshmem**
  - Shared r/w RAM region of two cells
  - Signaling (MSIs)
  - No messaging layer on top yet
- **Why not... virtio, rpmsg, you-name-it...?**
  - Must not share everything
  - Minimize copying
  - Minimize hypervisor effort
  - Avoid dynamic page remappings
- **No perfect solution, need to find the least evil one**
  - See also http://thread.gmane.org/gmane.linux.jailhouse/3479
  - BOF on virtio-based VM-to-VM communication at KVM Forum

Hard Partitioning for Linux: The Jailhouse Hypervisor

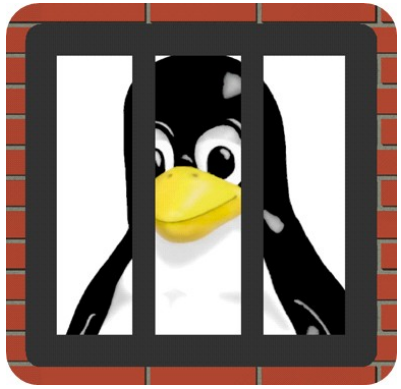
Agenda

- Motivation
- Jailhouse introduction & philosophy
- Current status (with demo)
- Configuration
- Running multiple Linux instances (with demo)
- **Summary**

SIEMENS

Summary

- **Jailhouse provides clean AMP for Linux**
  - Full CPU isolation
  - Minimal I/O latency
  - Reduced to the minimum (goal: <10k LOC/arch)
- **Jailhouse can run multiple Linux cells**
  - Small Linux patch set for x86
  - Feasible with customizations on ARM
- **Jailhouse is a community project**
  - GPLv2, public development for 2 years
  - Significant contributions enabled / are enabling AMD64, ARMv7, and now ARMv8
  - You are invited to join!



SIEMENS

Any Questions?

Thank you!

https://github.com/siemens/jailhouse

Jan Kiszka <jan.kiszka@siemens.com>