National Technical University of Athens
School of Electrical and Computer Engineering

Virtualization on Embedded Systems

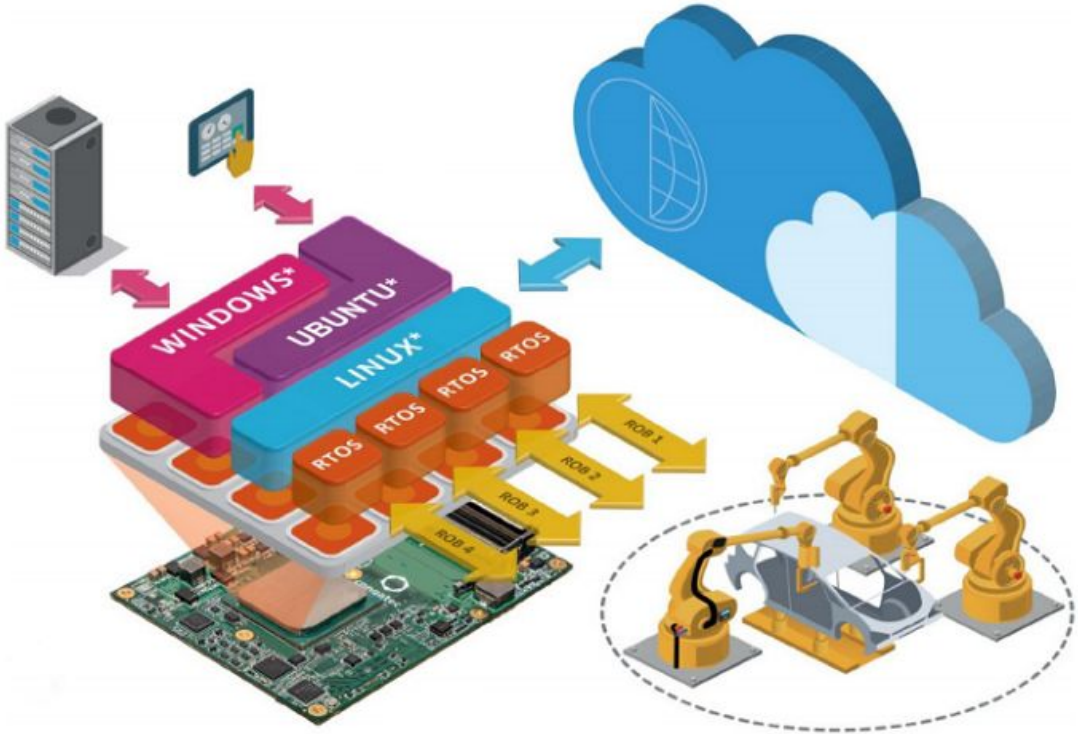
Iliakopoulou Nikoleta-Markela

Co-advised by: Dr. George Lentaris, Dimosthenis Masouros

Prof. Dimitrios Sountris

HW Connectivity

Entire Process Modeling



HaaS

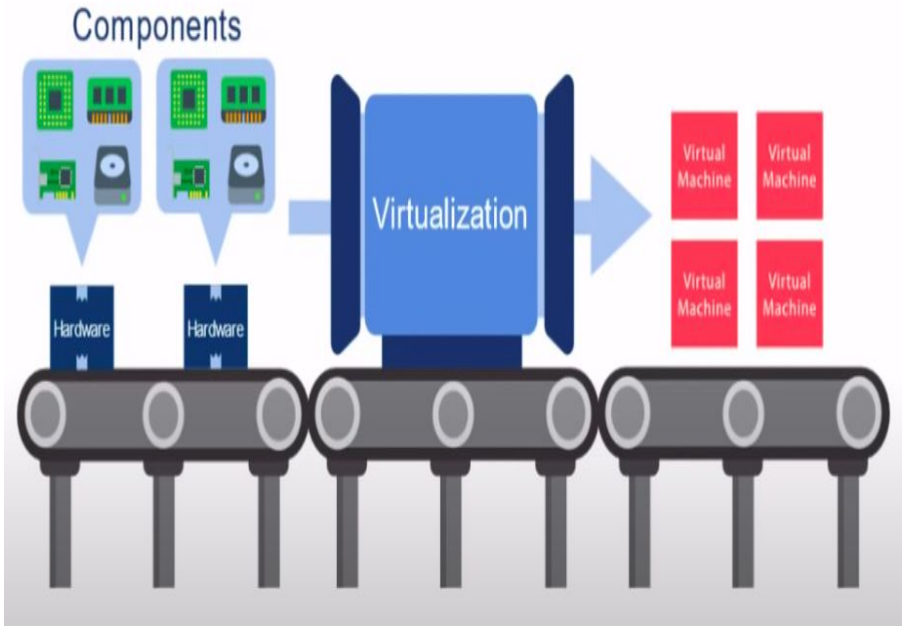
Remote Monitoring/Updating

Agenda

- *Virtualization*
- *Hypervisor vs Docker*
- *Jailhouse Project(Hypervisor)*
- *Apic-demo*

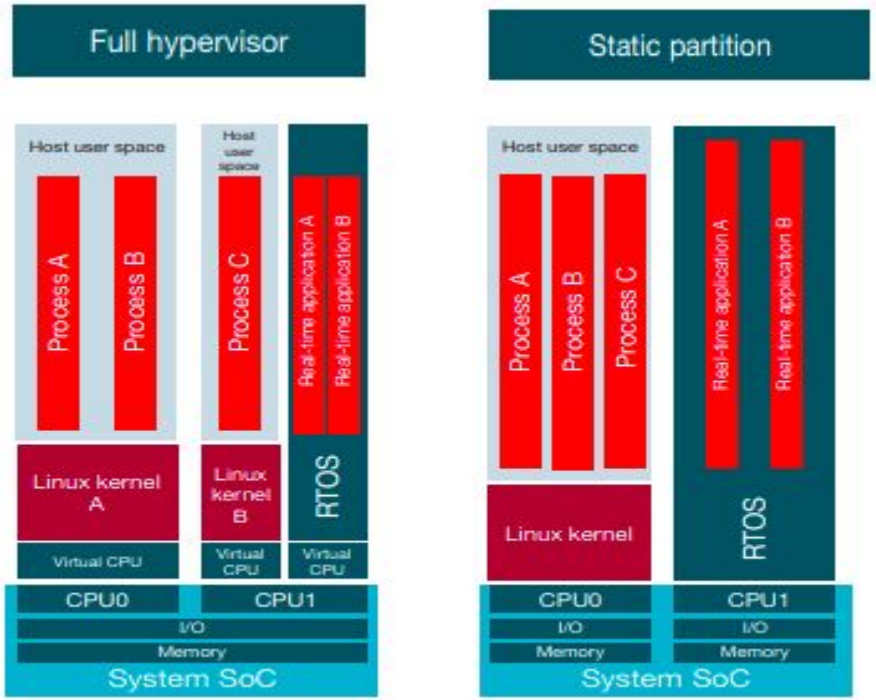
Virtualization

What is virtualization?



- *Efficiency - Energy Saving*
- *Idle Hardware* ↓
- *CPU Utilization* ↑
- *Maintenance*
- *Reusability*
- *Possible Resources Sharing*

Types of Virtualization



Full Virtualization

- *Powerful, yet impact on latency*
- *Large amount of computer resources needed*

Static Partitioning

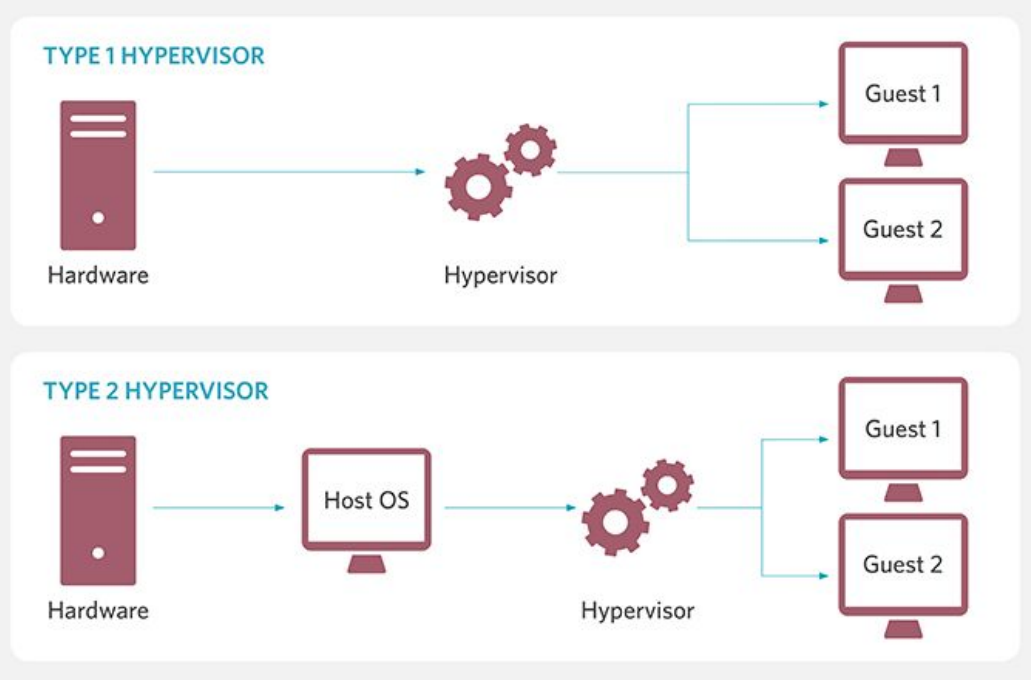
- *Same benefits of separating mixed-criticality tasks*
- *Timing of tasks is less affected*

Hypervisor vs Docker

Hypervisor

- *Software that creates and runs Virtual Machines (VMs)*
- *Easy reallocation of resources-like CPU, memory and storage*
- *Basic operating system-level components needed*
- *Resources given to VM*
- *Management of the scheduling of VM resources against physical resources*

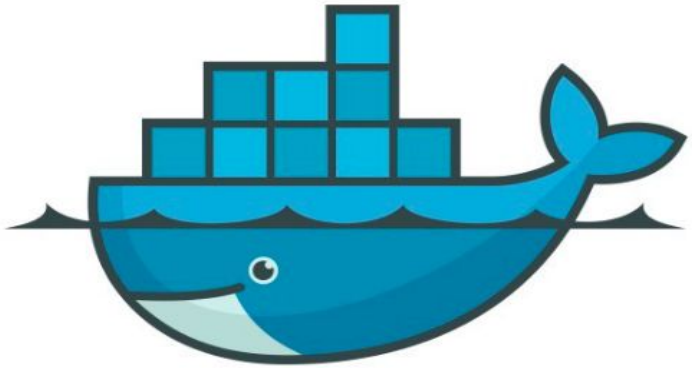
Types of Hypervisors



Bare Metal

Hosted

Docker Container



What is Docker?

- Containerization platform
- Packages the application and its dependencies inside a container

What is a container?

- Virtual runtime environment
- Runs on top of a single operating system kernel
- Emulates an OS
- Cores and memory allocated to containers by container engine
- Spatial Isolation and Security

Tabular Comparison & Conclusion

	Hypervisor	Docker
OS Support	Hypervisors are OS agnostic.	Docker supports only Linux.
Boot Time	Consumes upto 1 min to boot up.	Boots within seconds.
Security	Dual OS layers provide extra data security.	Dependent on supporting Linux kernel.
Resource Consumption	Consumes gigabytes of space.	Docker containers are lightweight.
Application Support	Can run multiple OS instances simultaneously.	Supports multiple application instances.

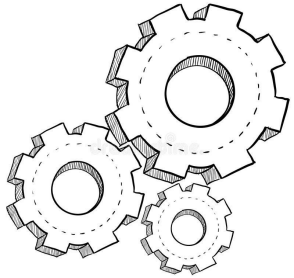
- *Major Differences though seemingly similar*
- *They both server different segments of the IT world based on the applications*
- *It is up to the organization!*

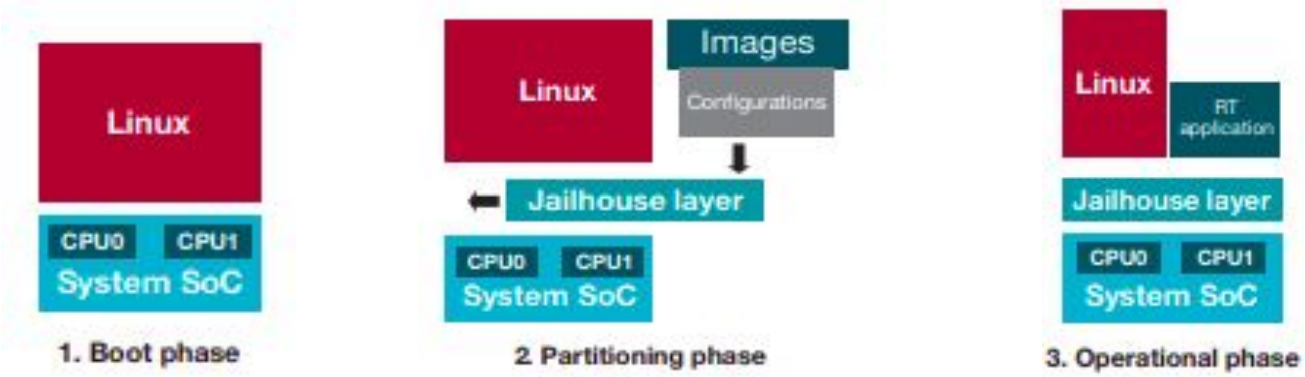
Jailhouse Project(Hypervisor)

Jailhouse Hypervisor

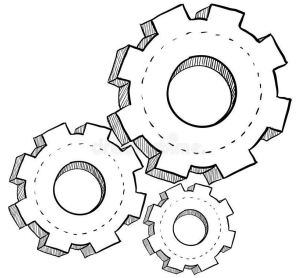
- Open-source hypervisor started by Jan Kiszka and Siemens
- Runs bare-metal
- Linux-Based
- Static Partitioning
- Optimized for simplicity rather than feature richness
- Real-Time capable
- Used for security applications
- Lightweight

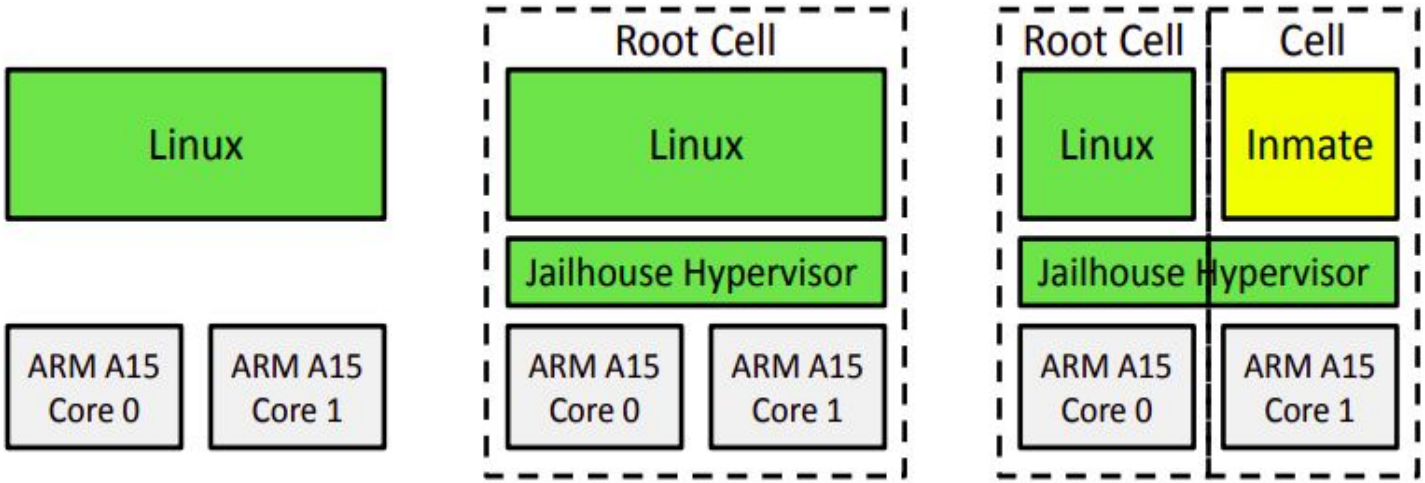
How it works (I)





How it works (II)





Apic-demo

Configuration file for apic-demo cell

```
#include <jailhouse/types.h>
#include <jailhouse/cell-config.h>

struct {
  struct jailhouse_cell_desc cell;
  __u64 cpus[1];
  struct jailhouse_memory mem_regions[2];
  struct jailhouse_cache cache_regions[1];
  struct jailhouse_pio pio_regions[3];
} __attribute__((packed)) config = {
  .cell = {
    .signature = JAILHOUSE_CELL_DESC_SIGNATURE,
    .revision = JAILHOUSE_CONFIG_REVISION,
    .name = "apic-demo",
    .flags = JAILHOUSE_CELL_VIRTUAL_CONSOLE_PERMITTED,

    .cpu_set_size = sizeof(config.cpus),
    .num_memory_regions = ARRAY_SIZE(config.mem_regions),
    .num_cache_regions = ARRAY_SIZE(config.cache_regions),
    .num_irqchips = 0,
    .num_pio_regions = ARRAY_SIZE(config.pio_regions),
    .num_pci_devices = 0,

    .console = {
      .type = JAILHOUSE_CON_TYPE_8250,
      .flags = JAILHOUSE_CON_ACCESS_PIO,
      .address = 0x3f8,
    },
  },

  .cpus = {
    0x8,
  },
```

```
  .mem_regions = {
    /* RAM */ {
      .phys_start = 0x3ef00000,
      .virt_start = 0,
      .size = 0x00100000,
      .flags = JAILHOUSE_MEM_READ | JAILHOUSE_MEM_WRITE |
        JAILHOUSE_MEM_EXECUTE | JAILHOUSE_MEM_LOADABLE,
    },
    /* communication region */ {
      .virt_start = 0x00100000,
      .size = 0x00001000,
      .flags = JAILHOUSE_MEM_READ | JAILHOUSE_MEM_WRITE |
        JAILHOUSE_MEM_COMM_REGION,
    },
  },

  .cache_regions = {
    {
      .start = 0,
      .size = 2,
      .type = JAILHOUSE_CACHE_L3,
    },
  },

  .pio_regions = {
    PIO_RANGE(0x2f8, 8), /* serial 2 */
    PIO_RANGE(0x3f8, 8), /* serial 1 */
    PIO_RANGE(0xe010, 8), /* OXPCIe952 serial */
  },
};
```

Basic Commands

```
:~# jailhouse enable /etc/jailhouse/qemu-x86.cell
:~# jailhouse cell create /etc/jailhouse/apic-demo.cell
:~# jailhouse cell load apic-demo /usr/libexec/jailhouse/demos/apic-demo.bin
:~# jailhouse cell start apic-demo
:~# jailhouse cell destroy apic-demo
:~# jailhouse disable
```

Enabling Hypervisor

```
Initializing Jailhouse hypervisor v0.12 (0-g92db71f2-dirty) on CPU 1
Code location: 0xffffffff0000050
Using x2APIC
Page pool usage after early setup: mem 47/975, remap 0/131072
Initializing processors:
 CPU 1... (APIC ID 1) OK
 CPU 0... (APIC ID 0) OK
 CPU 2... (APIC ID 2) OK
 CPU 3... (APIC ID 3) OK
Initializing unit: VT-d
DMAR unit @0xfed90000/0x1000
Reserving 24 interrupt(s) for device ff:00.0 at index 0
Initializing unit: IOAPIC
Initializing unit: Cache Allocation Technology
Initializing unit: PCI
Adding PCI device 00:01.0 to cell "QEMU-VM"
Adding PCI device 00:02.0 to cell "QEMU-VM"
Reserving 5 interrupt(s) for device 00:02.0 at index 24
Adding PCI device 00:1b.0 to cell "QEMU-VM"
Reserving 1 interrupt(s) for device 00:1b.0 at index 29
Adding PCI device 00:1f.0 to cell "QEMU-VM"
Adding PCI device 00:1f.2 to cell "QEMU-VM"
Reserving 1 interrupt(s) for device 00:1f.2 at index 30
Adding PCI device 00:1f.3 to cell "QEMU-VM"
Adding PCI device 00:1f.7 to cell "QEMU-VM"
Reserving 2 interrupt(s) for device 00:1f.7 at index 31
Adding virtual PCI device 00:0c.0 to cell "QEMU-VM"
Adding virtual PCI device 00:0d.0 to cell "QEMU-VM"
Adding virtual PCI device 00:0e.0 to cell "QEMU-VM"
Adding virtual PCI device 00:0f.0 to cell "QEMU-VM"
Page pool usage after late setup: mem 270/975, remap 65543/131072
Activating hypervisor
```

Cell creation - Demo Loading and Running

```
Created cell "apic-demo"
Page pool usage after cell creation: mem 285/975, remap 65543/131072
Cell "apic-demo" can be loaded
```

```
Started cell "apic-demo"
Calibrated TSC frequency: 2208002.000 kHz
Calibrated APIC frequency: 2208002 kHz
Timer fired, jitter:   1487 ns, min:   1487 ns, max:   1487 ns
Timer fired, jitter:   2376 ns, min:   1487 ns, max:   2376 ns
Timer fired, jitter:   1826 ns, min:   1487 ns, max:   2376 ns
Timer fired, jitter:   3049 ns, min:   1487 ns, max:   3049 ns
Timer fired, jitter:   1760 ns, min:   1487 ns, max:   3049 ns
Timer fired, jitter:   4724 ns, min:   1487 ns, max:   4724 ns
Timer fired, jitter:   2465 ns, min:   1487 ns, max:   4724 ns
Timer fired, jitter:   2398 ns, min:   1487 ns, max:   4724 ns
Timer fired, jitter:   2080 ns, min:   1487 ns, max:   4724 ns
Timer fired, jitter:   1398 ns, min:   1398 ns, max:   4724 ns
Timer fired, jitter:   2227 ns, min:   1398 ns, max:   4724 ns
Timer fired, jitter:   1617 ns, min:   1398 ns, max:   4724 ns
```

Destroy Cell and Disable Hypervisor

```
Stopped APIC demo
Cell "apic-demo" can be loaded
Closing cell "apic-demo"
Page pool usage after cell destruction: mem 272/975, remap 65543/131072
CPU 3 received SIPI, vector 9a
Shutting down hypervisor
 Releasing CPU 0
 Releasing CPU 1
 Releasing CPU 2
 Releasing CPU 3
```

Thank You!