UCLouvain

FOSDEM 2019

Hardening the Operating System against transient faults : Dealing with external interrupts

M. P. Tokponnon, M. Lobelle, E. Ezin, P. Van Roy

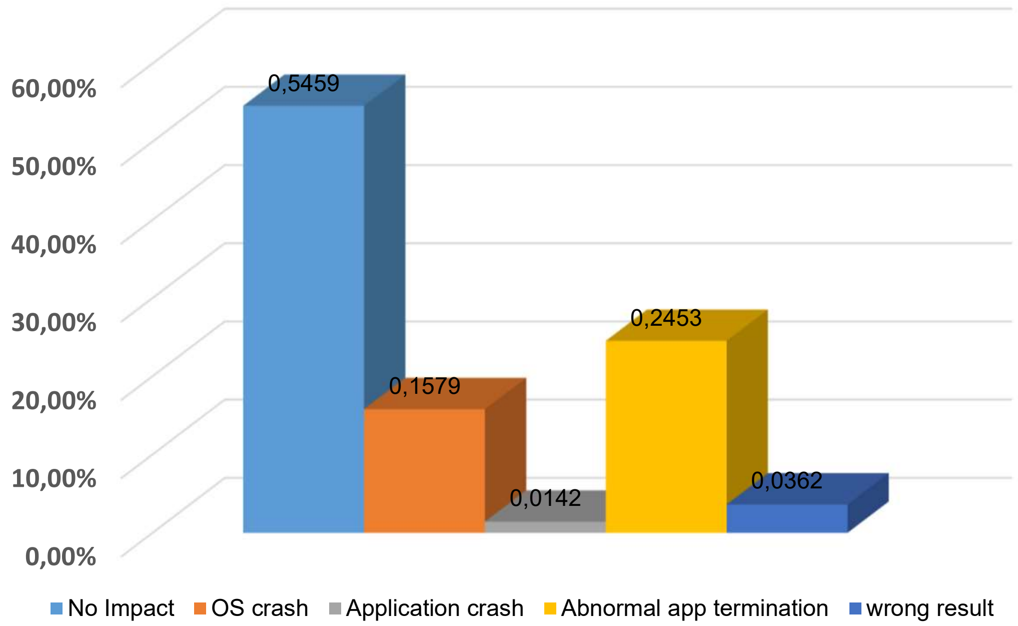
Outline

1. **Context**
2. **Objective & general approach**
3. **Interrupt handling general approach**
4. **Result with Genode Demo scenario**
4. **Conclusion**

Context

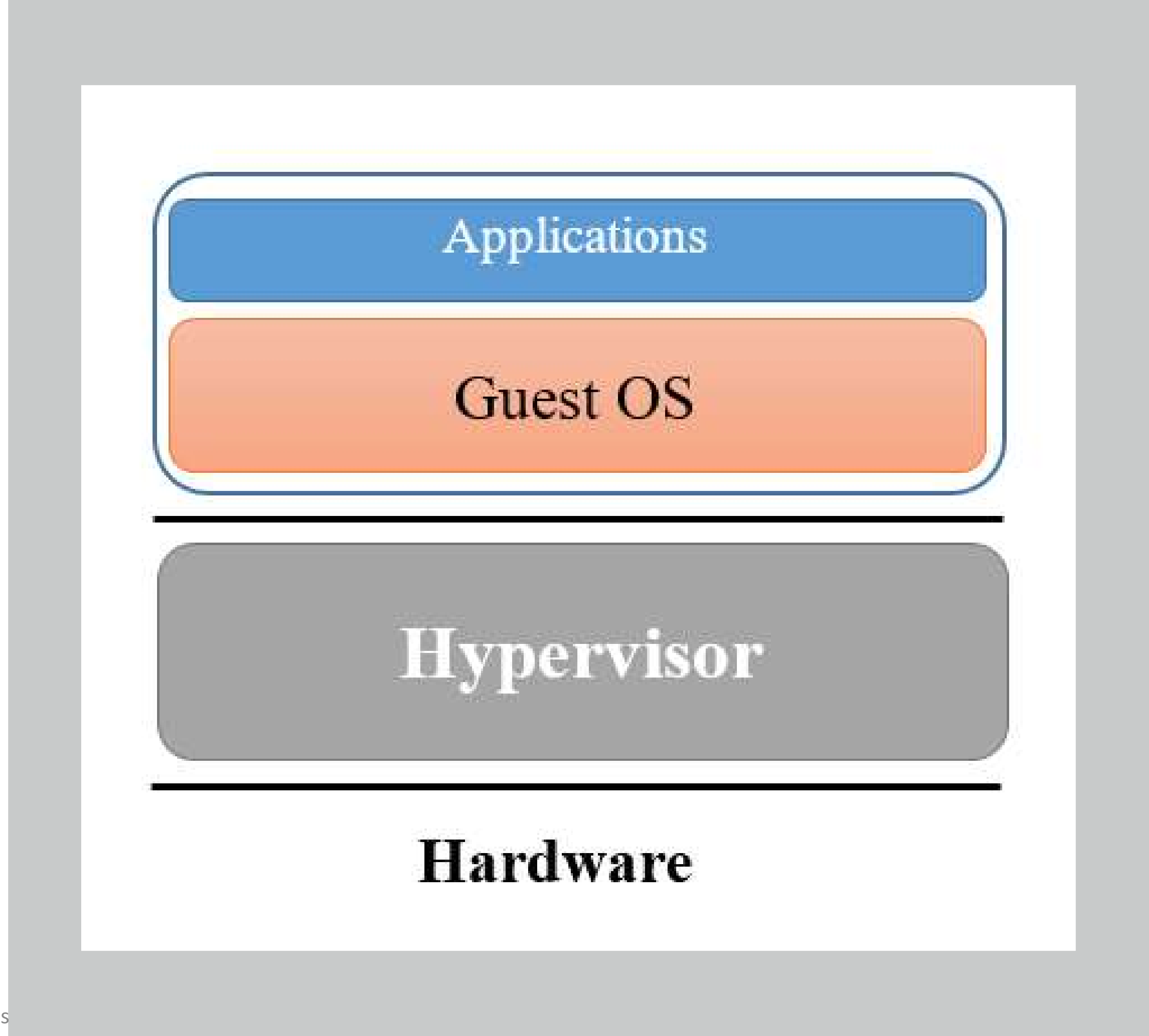
- **Transient faults**
  - **change of state in logic circuit.**
  - **caused by an ionizing particle striking a sensitive circuit**
  - **don’t damage the system**
  - **May lead to crash, hang or erratic behavior at software layer**

Transient errors impact on software (*)



Objective

- Hypervisor-based hardening (Using Nova + Genode + VirtualBox) to protect Operating system



Methodology

- **Blended Hardening of processors:**
  - **Memory is protected in hardware (ECC):**
  - **Hardware detection: e.g. by machine check exceptions and other traps (Illegal instructions, memory fault)**
  - **Detection of silent faults by double execution with comparison (DWC) of short processing element (200µs) executed atomically.**
- **How to handle asynchronous event (interrupt) in redundancy context?**

Processing Element Definition

Processing element is

- **a sequence of process CPU assembly instructions**
- **delimited by:**
  - **Maximum number of instruction (via Performance Monitoring Interrupt – PMI)**
  - **System call**
  - **CPU Exceptions (Page fault, GP fault, TSS, NM, …)**
  - **Input / Output Instructions**
  - **Process switch**
  - **Later, VM Exit**

Interrupt handling

2 classes of interrupts

- **Performance Monitoring Interrupt :**
  - **used to stop PE when a specific number of instructions is executed by CPU**
  - **handled immediately**
- **External Interrupts:**
  - **cannot be part of Processing Element**
  - **handling is delayed, queued (for differed servicing) until PE is finished:**
    - **Enqueue triggered interrupts**
    - **Execute EOI()**
    - **After committing the current PE, dequeue recorded interrupts (First In First Out)**

If the interrupt require immediate servicing and is proved not influencing PE idempotency
service it
Else Dead case

Result with Genode Demo scenario on Qemu (1/2)

- **During booting (Busy time):**
  - **665 Gcycle (≈4mn, no Idle loop)**
  - **99% of timer interrupts services are delayed**
  - **100% of other interrupts (Device originated Interrupts as GSI) are delayed**

Interrupts	CPU Cycles (Kilocycle)	Duration (µs)
Timer	47 K	18
Keyboard	103 K	41
Other GSI	128 K	51

Result with Genode Demo scenario on Qemu (2/2)

- **After boot completed:**
  - **323 Gcycle (≈2mn9s, 5s of Idle loop) : 4% of time in idle loop**
  - **21% of timer interrupts services are delayed**
  - **None of other interrupts are delayed**

Conclusion

- **When double executing process instructions, external interrupts servicing is delayed to preserve idempotency among the two runs**
- **We investigate this interrupt delaying impact on process execution in Genode Demonstration operating system**
- **We found that while performance penalty is quite large during CPU bound operations (like when booting), it is negligible for common workload**
- **We will run some common benchmarks to access more accurately this performance impact**

UCLouvain

Mens Agitat Molem
UNIVERSITE D'ABOMEY-CALAVI

FOSDEM 2019

Hardening the Operating System against transient faults : Dealing with external interrupts

M. P. Tokponnon, M. Lobelle, E. Ezin, P. Van Roy