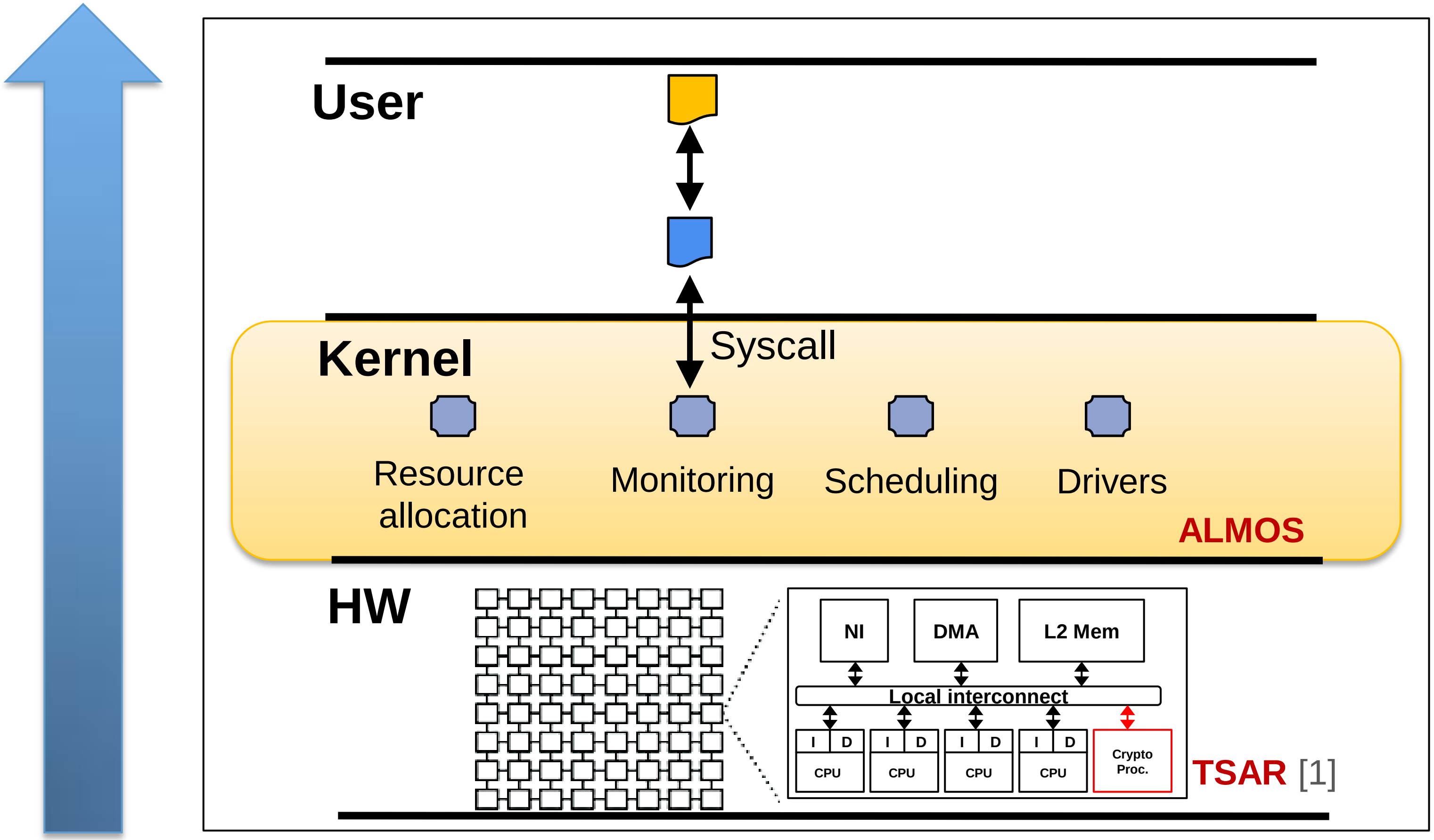
ALMOS many-core operating system extension with secure-enable mechanisms for dynamic creation of secure zones

Maria Méndez Real, Vincent Migliore, Vianney Lapotre, Guy Gogniat

Université de Bretagne-Sud, Lab-STICC

PDP 2016

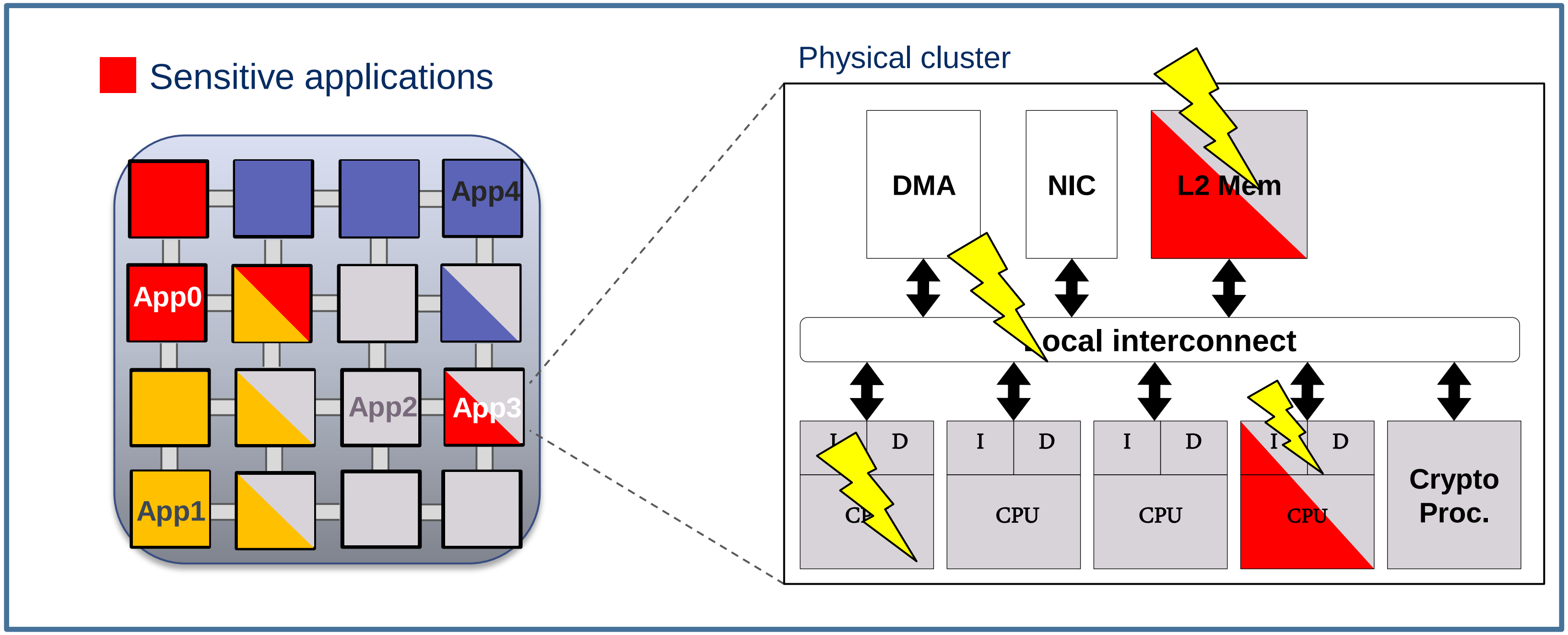
Chain of trust from HW to SW



Building a chain of trust from HW to SW

[1] C. Lopez, et al., “Trusted computing using enhanced many-core architectures with cryptoprocessors,” in Proc. of IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC), 2014.

Thread model



⚠ Sharing resources ➡ Potential attacks

SW attacks

- Confidentiality and integrity attacks (C&I)
- Denial of Services (DoS)
- Leakage of information (Cache side Channel attacks (SCA))[2][3]

[2] J. Demme and S. Sethumadhavan, "Side-channel vulnerability metrics: Svf vs. csv," in Proc. of 11th Annual Workshop on Duplicating, Deconstructing and Debunking (WDDD), 2014.

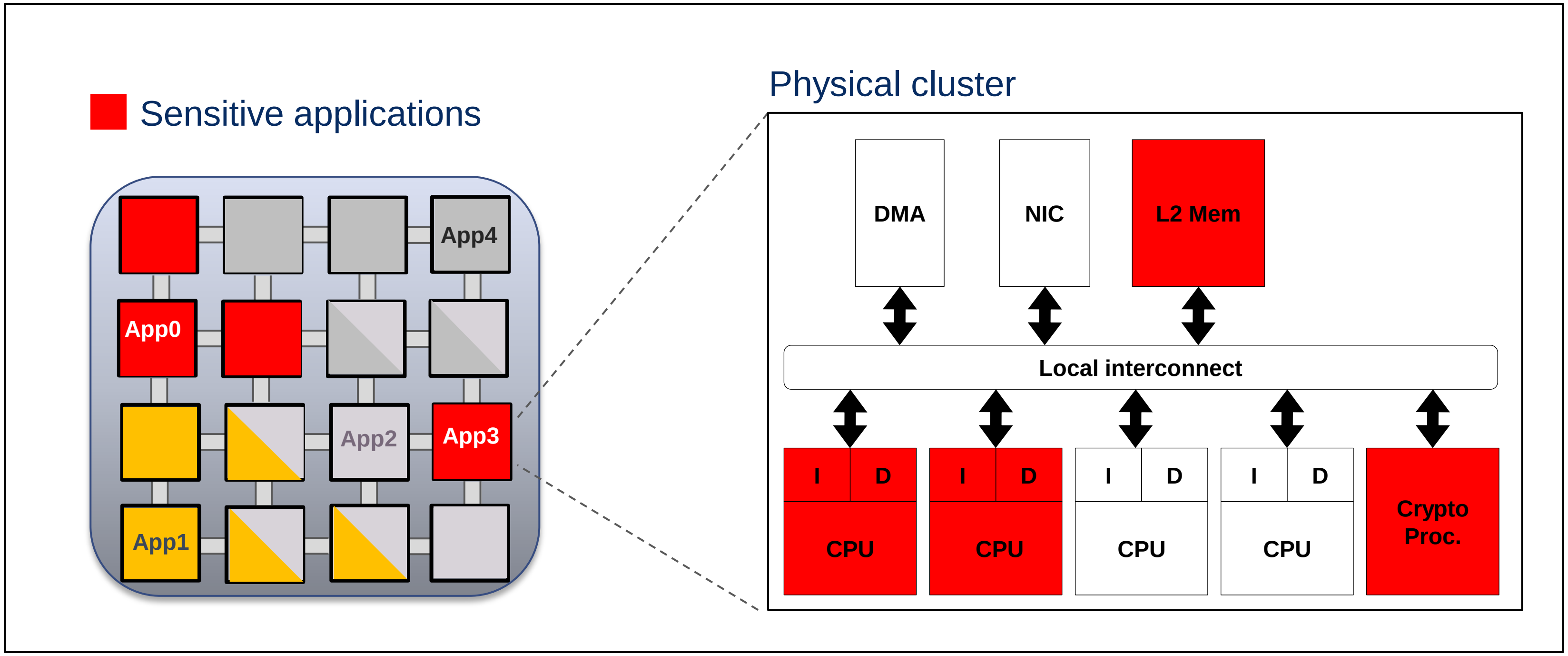
[3] Y. Wang and G. Suh, "Efficient timing channel protection for on chip networks," in Proc. of the 2012 IEEE/ACM Sixth International Symposium on Networks-on-Chip (NOCS), 2012, pp. 142–151.

State of the art

Countermeasure	C&I	Cache SCA	Communication SCA	DoS
Bi partitioning the processor [4]	✓	✗	✗	✗
Logical isolation (MMU, MPU, NoC MMU [5][6])	✓	✗	✗	✗
Monitoring mechanisms [7]	✗	✗	✗	✓
NoC protection [8]	✗	✗	✓	✗

[4] www.arm.com/products/processors/technologies/trustzone/
[5] R. Masti, et al., “Isolated execution in many-core architectures,” in Proc. of Network and Distributed System Security Simposium (NDSS), 2014.
[6] G. Kornaros, et al., “Hardware Support for Cost-Effective System-level Protection in Multi-Core SoCs”, in Proc. of Digital System esign (DSD), 2015.
[7]L. Fiorin, et al., “A security monitoring service for nocs”, in Proc. of Hardware/Software codesign and system synthesis (CODES+ISSS), 2008.
[8]J. Sepulveda, et al., “Hierarchical noc-based security for mp-soc dynamic protection”, Proc. of Circuits and Systems (LASCAS), 2012.

Physical isolation for sensitive applications

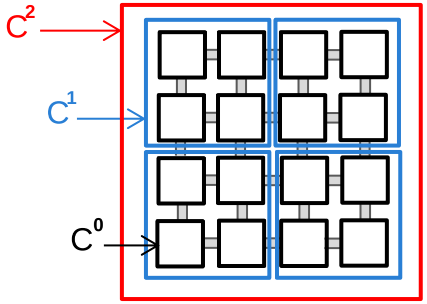


1) How can this be achieved?
2) How can the performance overhead be evaluated?
3) How can this overhead be reduced?

Extension of ALMOS OS

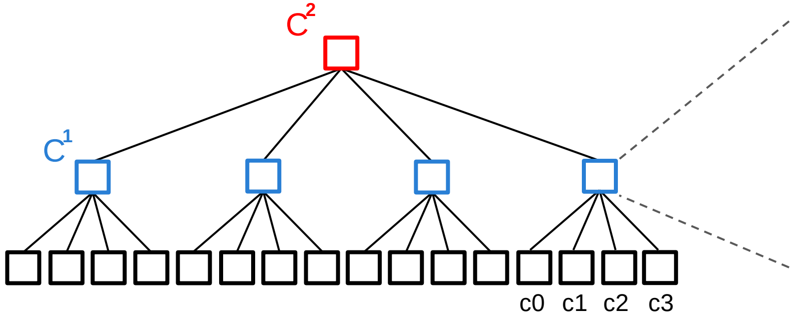
- **Scheduling**
- **Monitoring**
- **Application mapping**
- **Task (thread/fork) mapping**
- **Memory allocation (level 2 cache)**

Distributed Quaternary Decision Tree (DQDT)



M : Physical pages number
T : Threads number (Runnable)
U : Processor utilization

Tcy : Crypto tasks number
Ucy : Crypto-processor utilization
S : Secure zone ID

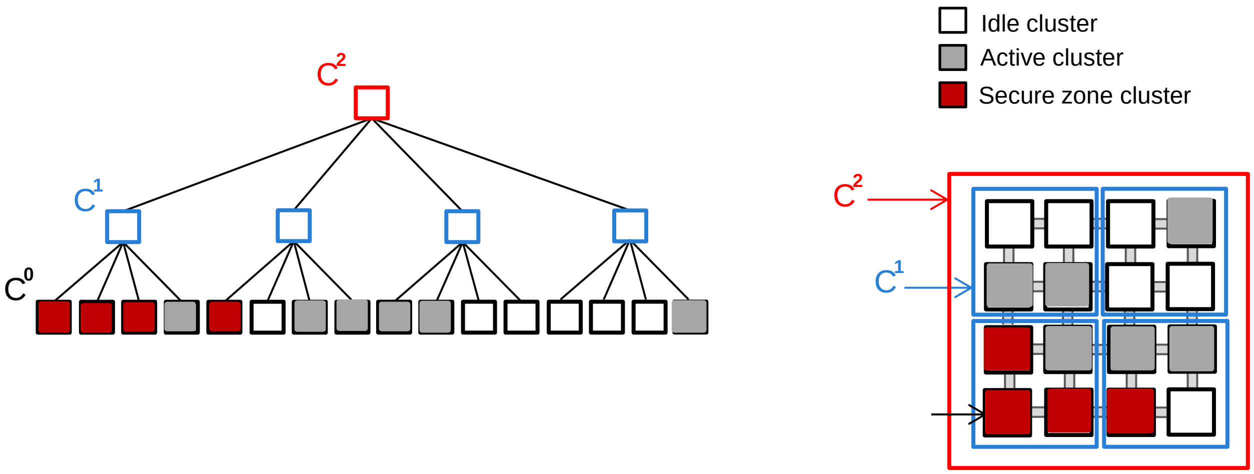


c0	c1	c2	c3	total
M	M	M	M	M
T	T	T	T	T
U	U	U	U	U
Tcy	Tcy	Tcy	Tcy	Tcy
Ucy	Ucy	Ucy	Ucy	Ucy
				S

Extension of ALMOS OS

Maximum parallelism of an application → Searching for idle contiguous physical clusters → Creation of a secure zone

- **Scheduling**
- **Monitoring**
- **Application mapping**
- **Task (thread/fork) mapping**
- **Memory allocation (level 2 cache)**



Extension of ALMOS OS

Scheduling

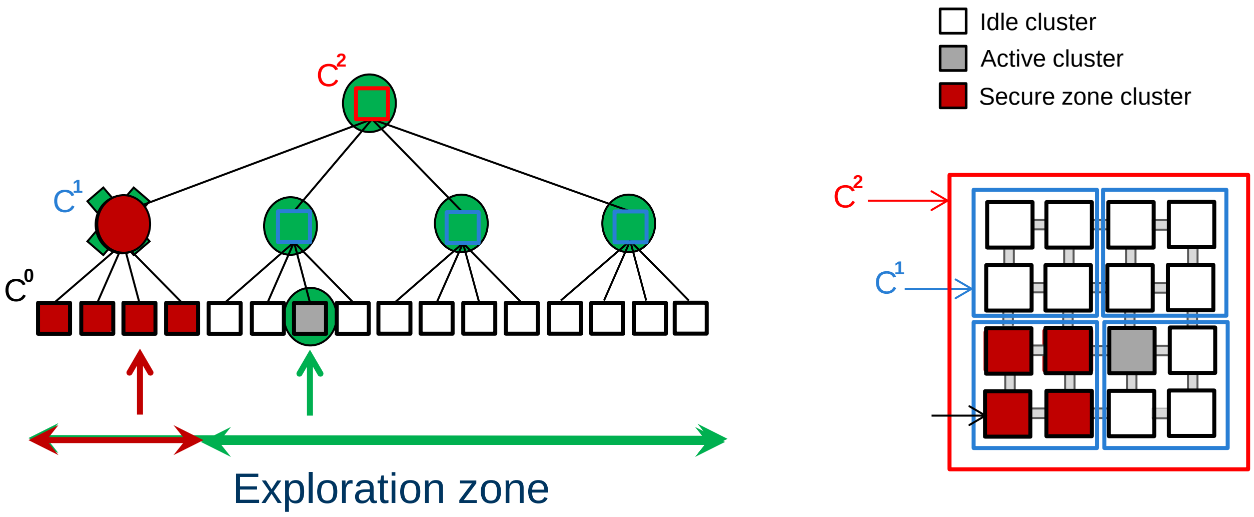
Monitoring

Application mapping

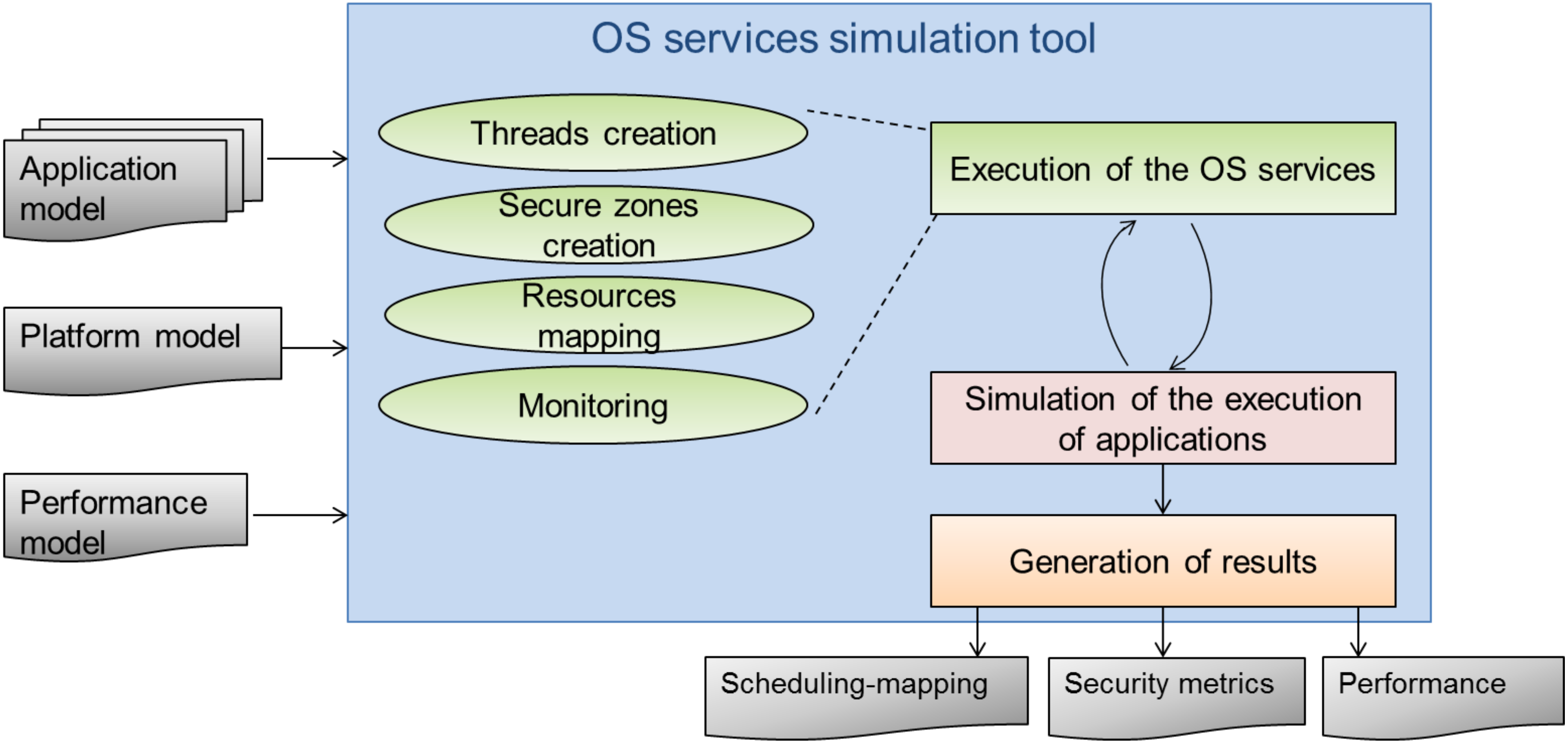
Task (thread/fork) mapping

Memory allocation (level 2 cache)

After physical isolation mechanisms



Evaluation of ALMOS OS extension



OS services simulation tool

Application model

Platform model

Performance model

Threads creation

Secure zones creation

Resources mapping

Monitoring

Execution of the OS services

Simulation of the execution of applications

Generation of results

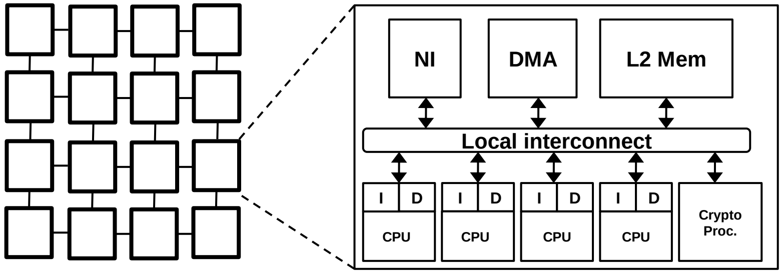
Scheduling-mapping

Security metrics

Performance

Experimental set up

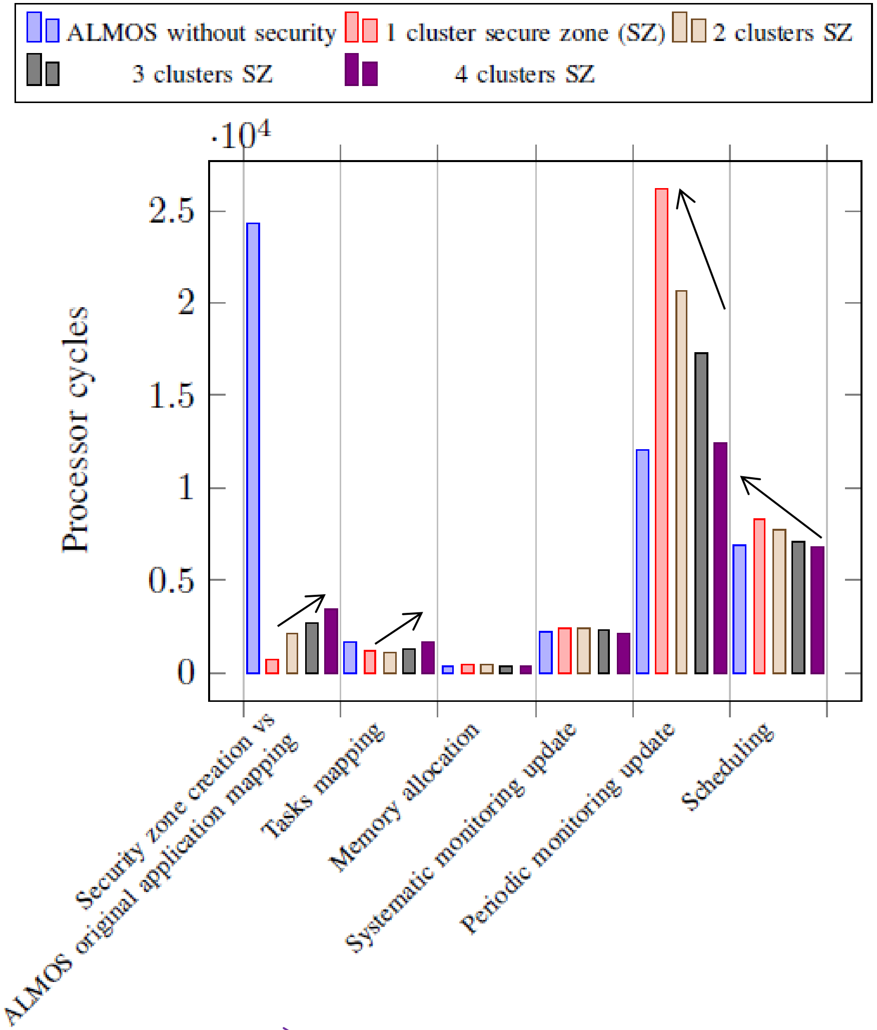
- ALMOS – TSAR system configuration
  - Access time to a local memory bank
  - Access time to a distant memory bank per hop
  - Computation power of processors
  - …
- 4x4 cluster architecture (4*4 clusters *4 processors = 64 processors)



- Synthetic application task graphs with high parallelization degree

Time spent on the OS services

- According to the size of the secure zone

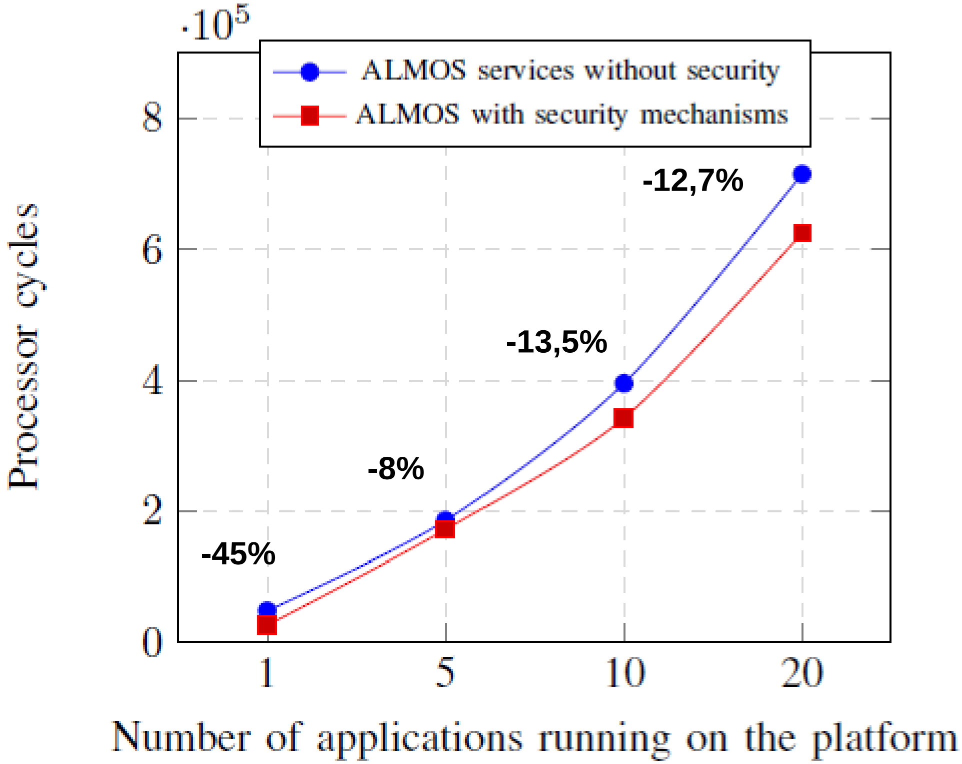


Isolation mechanisms dependent

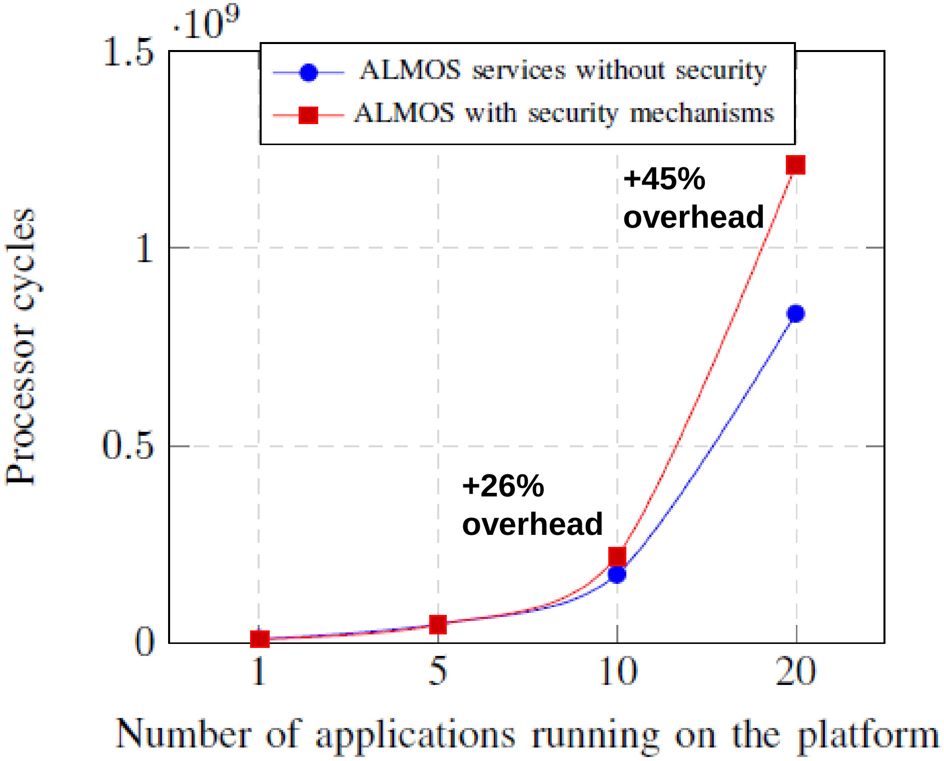
Total execution time dependent

Time spent on the OS services vs total execution time

- **Time spent on OS services** according to the workload on the platform when one single application is physically isolated (4 clusters secure zone)



- **Total execution time of non isolated applications** when one single application is physically isolated (4 clusters secure zone)



Discussion and future work

Conclusion

- Physical isolation
- Reduction of the time spent on ALMOS services
- Performance overhead receivable when workload < 27%

Discussion

- Focus on the OS services
- ALMOS-TSAR oriented study
- Synthetic applications' task graphs

Work in progress

- Study on generic multicore/many-core architectures through Open Virtual Platforms (OVP) and SystemC environment
- Communication between applications
- Mechanisms seeking to reduce the induced performance overhead

Thank you for your attention!

Maria Méndez Real, Vincent Migliore, Vianney Lapotre, Guy Gogniat