Programa de Pós-Graduação em Ciência da Computação – PPGCC-PUCRS

Vulnerabilities and Security in NoC-based Many-cores

FERNANDO GEHM MORAES
fernando.moraes@pucrs.br

PUCRS

Outline

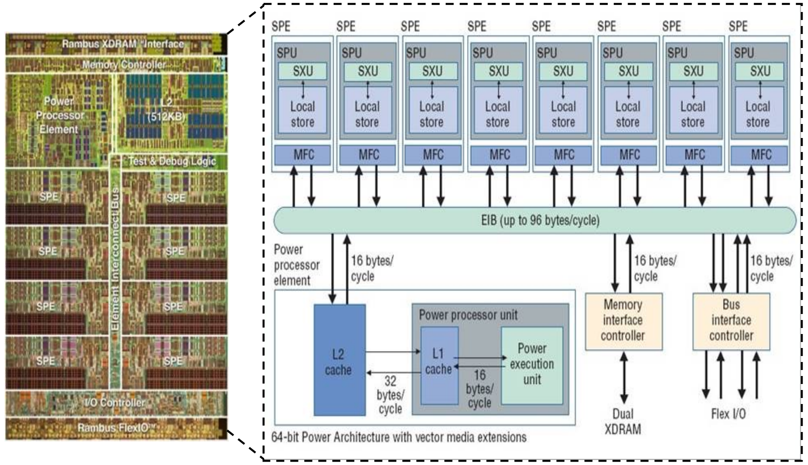
1. Introduction
2. Threat Model
3. Protecting *App* Admission
4. Protecting *App* Execution
5. Protecting IO

} Part I

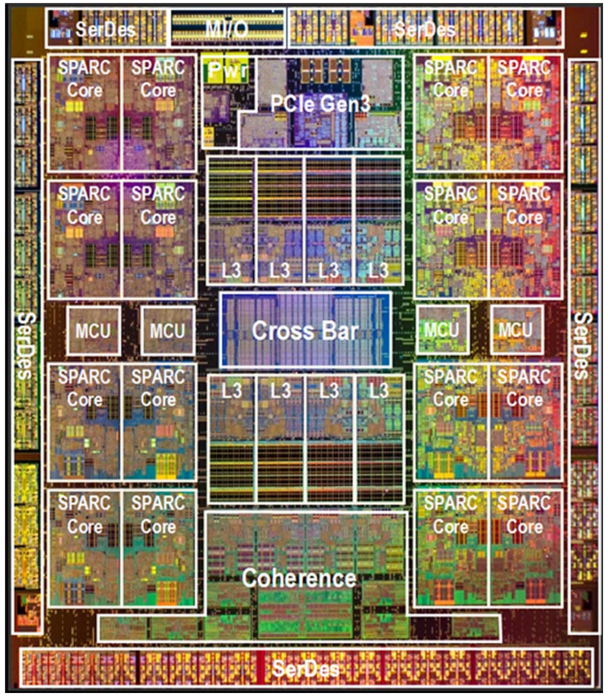
6. Security Methods Proposals } Part II

1. Introduction – many-core systems

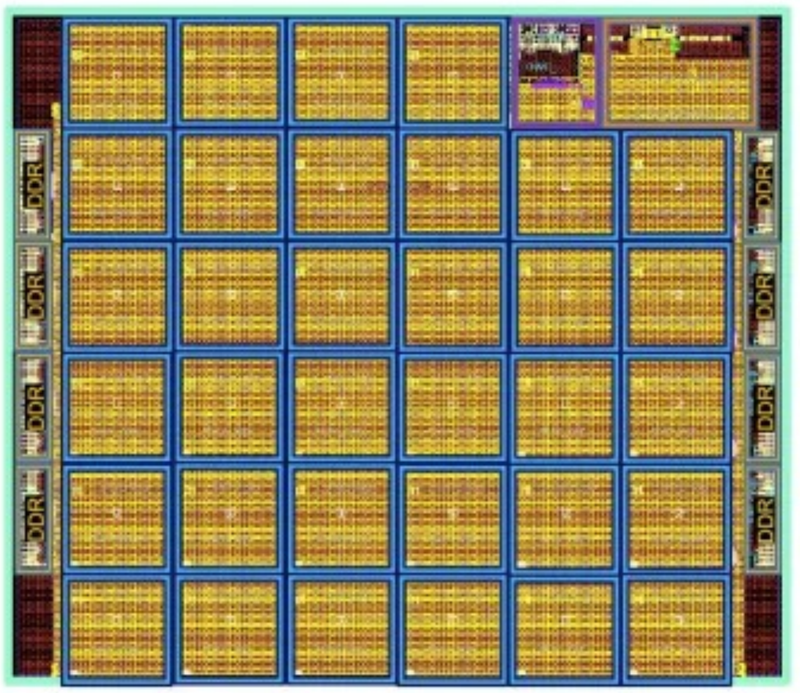
Computational systems tend towards parallel architectures with multiprocessor on chip systems – MPSoCs



Cell Processor - IBM (2006)



UltraSparc T5 (2013)



Esperanto ML Chip - 1,100 RISC-V Cores (2020)

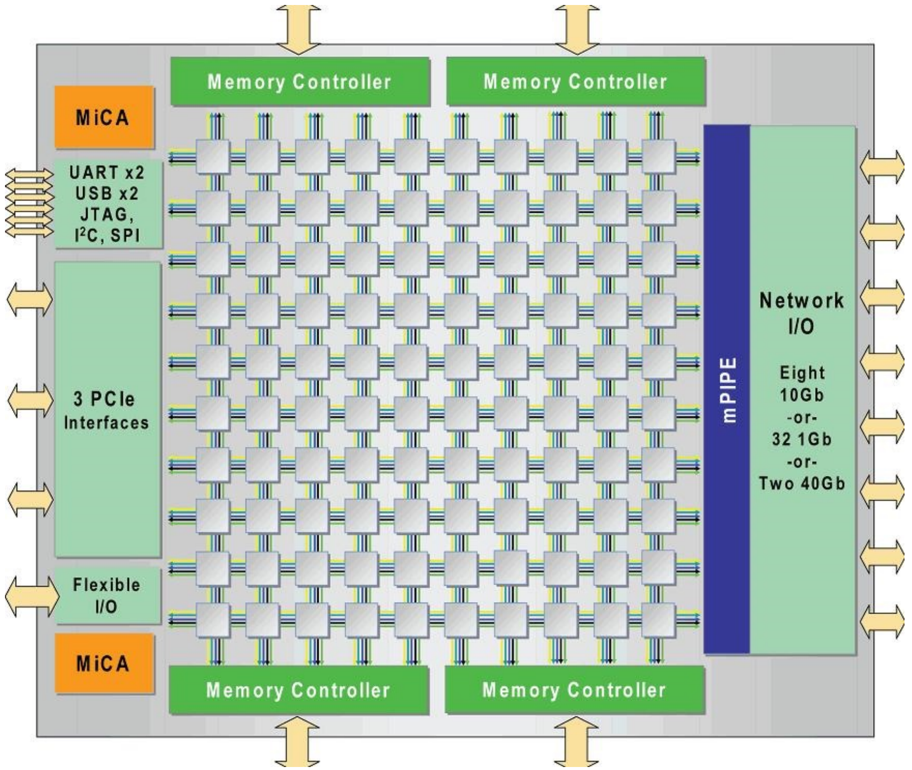
1. Introduction – many-core systems

NoC-based many-core SoCs enable

- high connectivity
- massive parallelism
- simultaneous executions of several applications

Increase and continuous adoption in electronic systems

- IoT, ML, autonomous-car systems, hardware accelerators, cell phones, …



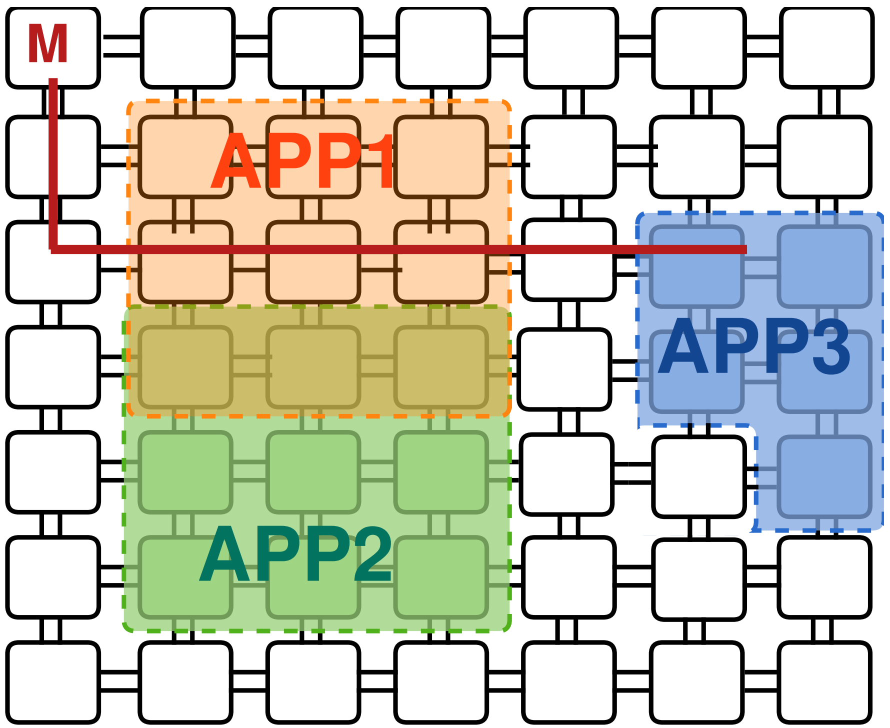
Tile GX– Tilera (100 cores) → Mellanox
→ Nvidia bought Mellanox in 2019 - 6.9 bi

1. Introduction – the security problem (hw)

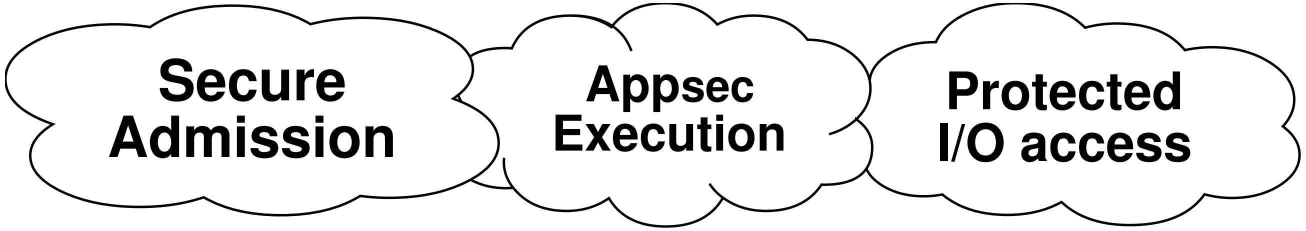
Resource sharing during the application execution

- shared computation: cores and memory
- shared communication: NoC links and routers

Access peripherals (M) without expose application data



1. Introduction – Application's Execution Phases



Application admission

- object code/data transfer from an off-chip entity to the MPSoC
- system must trust on the entity transmitting the application
- the integrity of the application/data must be verified to avoid the insertion of malicious code

Execution time

- malicious attacker may have access to sensitive <u>computation</u> or <u>communication data</u>
- computation (cores) and communication (NoC) **must be protected**

Communication with external devices

- unauthorized access to instructions and data in shared memory and peripherals can compromise the applications' execution

2. Threat Model - Security principles

[Ramachandran 2002]

PUCRS

Confidentiality

- the property of non-disclosure of information to unauthorized processes, entities or users

Availability

- the protection of resources from threats that might impact any of the system's resources availability

Integrity

- the prevention of modification or destruction of a resource by an unauthorized entity or user

Authentication

- the process of establishment and validation of a claimed identity

Authorization

- the process of determining whether a validated entity can access a secured resource based on attributes, predicates or context

Auditing

- the property of logging sufficient system activities to reconstruct events *(not applied to the MPSoC context - NA)*

Nonrepudiation

- the prevention of any participant denying his role in the interaction once it is completed *(NA)*

2. Attacks that compromise the system (1/2)

Denial-of-Service - DoS (compromises **availability**)

- disruption of the system by overloading resources
- a malicious application task generating packets with a high injection rate can produce this attack, overloading the communication infrastructure

Distributed Denial-of-Service - DDoS (compromises **availability**)

- similar to DoS, uses multiples tasks to attack and disrupt the system by overloading resources
- a malicious application running in distinct cores can coordinate an attack to a specific router overloading its communication capacity

Timing attack (side channel attack) (compromises **confidentiality**)

- explores the communication collision between the sensitive traffic and the attacker traffic
- the latency interference induced by malicious traffic can provide to the attacker some information about the timing, frequency, and volume of the secure communication

Spoofing (compromises **authorization** and **authentication**)

- a malicious application successfully falsifies its identity to obtain unauthorized privileges

2. Attacks that compromise the system (2/2)

Hijacking (compromises **authorization** and **authentication**)

- an attempt to alter the system configuration to execute a set of abnormal tasks along with normal system operation (e.g., during the load of the operating system or an application)

Man-in-the-Middle - MitM (compromises **confidentiality, authorization** and **authentication)**

- an attack where the attacker secretly relays and alters the communication between the external entity and the system
- enables the attacker to send malicious data or obtain secret information

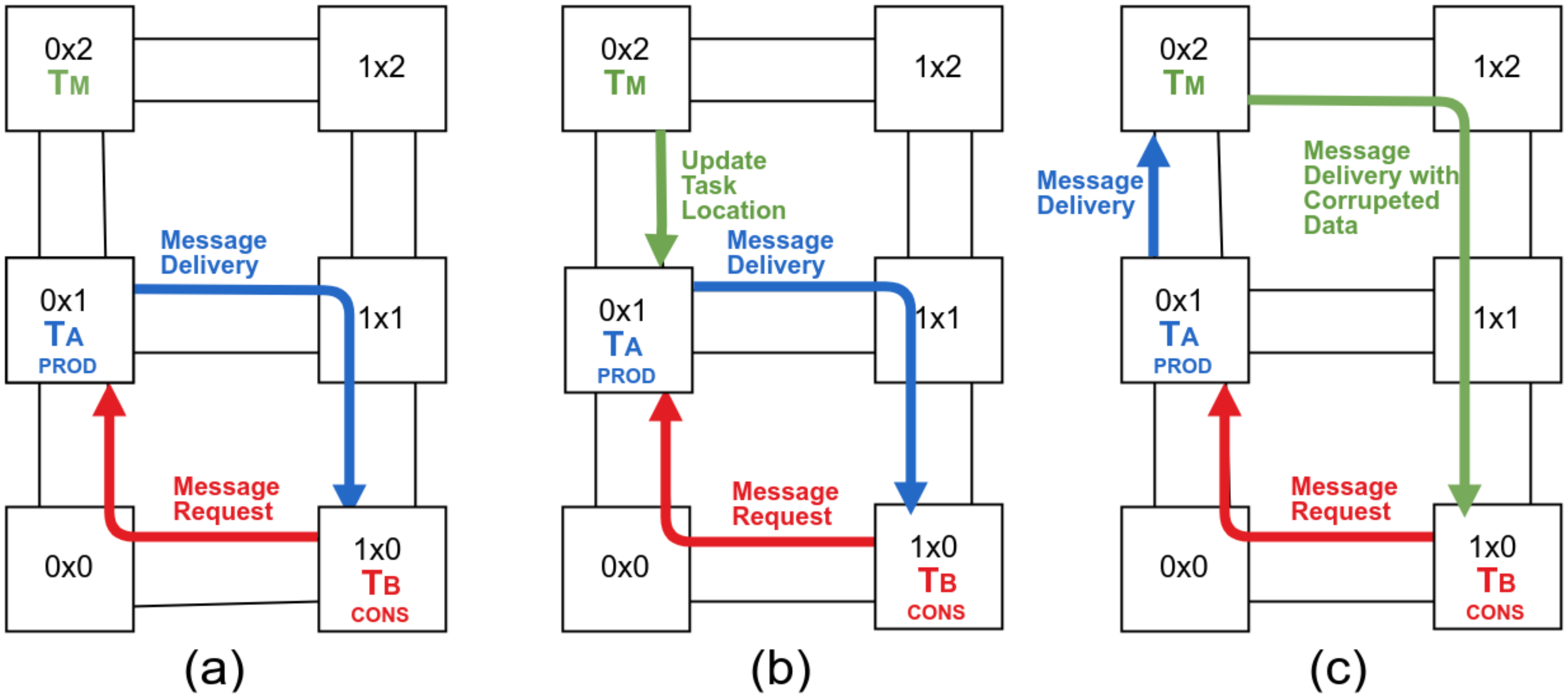
Hardware Trojans (compromises **availability**, **authorization** and **authentication**)

- a malicious modification of the system's hardware (e.g., inserted into the NoC) aiming to sniff and leak sensitive data

Trojan Horse and backdoor (compromises **availability and confidentiality**)

- the tampering of the task's source code during the admission of the application can insert malicious code

2. Man-in-the-middle attack – an example



(a) Task T_A communicates with task T_B
(b) Malicious tasks (T_M) initiates the attack
(c) T_M has access to the communication flow

3. Protecting *App* Admission

Secure Admission | Appsec Execution | Protected I/O access

The application admission corresponds to the object code transfer from an off-chip entity to the MPSoC

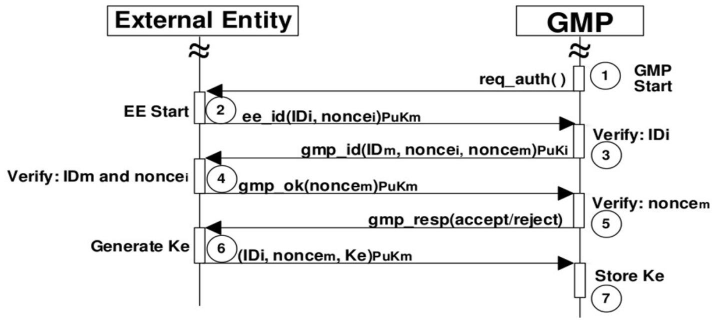
- Each actor (external entity and MPSoC) must confirm the other part's identity, and the integrity of the application must be verified to avoid the tampering of the application's object code
- Solutions to these issues exist for the Internet, computer networks, and software
- Few proposals in the many-core area

Zero Knowledge Proof protocol [Khernane 2016]

- lightweight authentication scheme for WBAN (Wireless Body Area Network) called BANZKP
- protocol confirm the identity of the sensor nodes
- after the authentication success, an encryption mechanism provides the message privacy protection

Elliptic-curve Diffie–Hellman - ECDH

- system setup, registration, and authentication
- at the end of these steps each part have a common key (K_e) used in MAC generation and verification



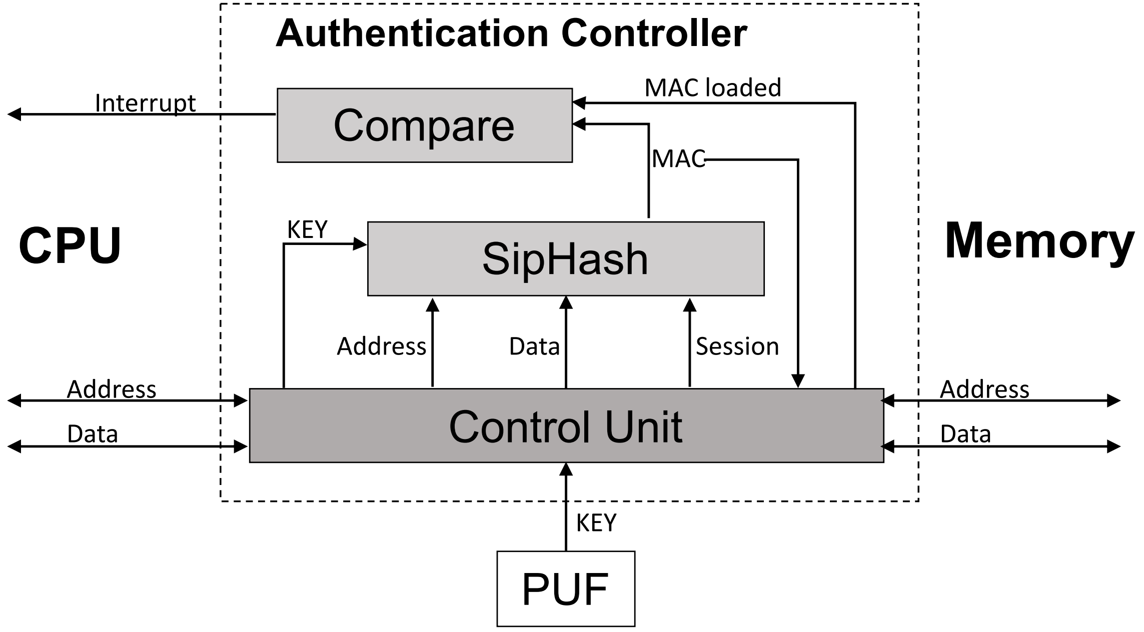
3. Protecting *App* Admission

Secure Admission | Appsec Execution | Protected I/O access

PUCRS

Integrity with PUF and MAC [Sepúlveda 2018]

- runtime mechanism based on MAC (Message Authentication Code) and PUF (Physical Unclonable Function) to provide **memory integrity and authentication**
- MAC uses SipHash algorithm
- mechanism have three stages:
  - Key generation (PUF)
  - MAC initialization and application installation
  - Operation



4. Protecting App Execution

Secure Admission | Appsec Execution | Protected I/O access

PUCRS

Protecting:

Communication	Computation	Comm. and Comp.
• firewalls • routing scheme • encryption • temporal network partition • packet validation	• logical and spatial isolation (clusters) • ARM TrustZone (ATZ)	• secure zones - partition and encryption • secure zones - spatial isolation and encryption • obfuscation

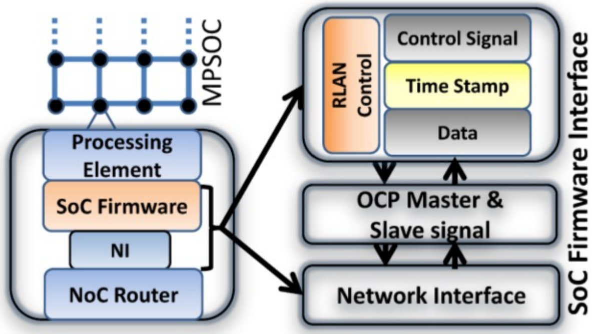
4a. Protecting Communication

Firewall

- hardware barrier placed at the communication structure ports to control the input and output of an element
- **tables** to store the recognized trusted sources and a **controller** that allows the authorized traffic and blocks unauthorized traffic

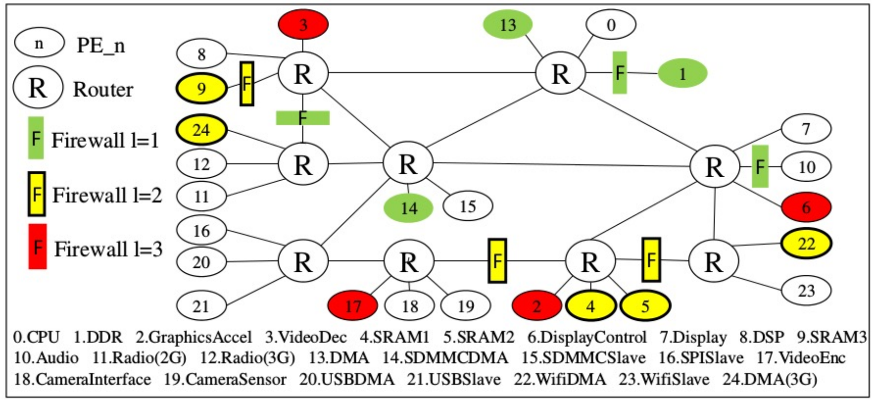
Rajesh et al. (2015)

- runtime latency auditor, called RLAN, to dynamically monitor the on-chip resources availability and properly filter the malicious traffic
- packets traversing routes have spatial (source-target pairs) and temporal similarity (latency)



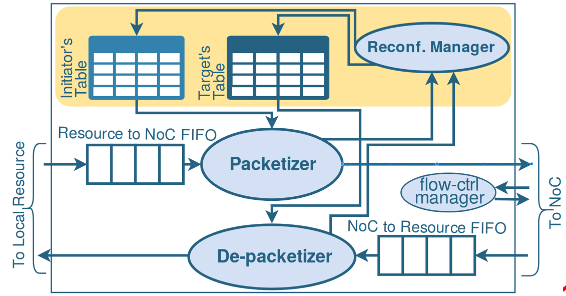
Hu et al. (2015)

- design time analysis of the traffic and the NoC architecture select the levels and position of the firewalls:
  (a) between a PE and a router
  (b) between routers



Azad et al. (2018/2019)

- Firewall placed at the NI, with two tables:
  - initiator table, checks if the source has permission to send messages
  - target table, which verifies if the message can enter the target unit



4a. Protecting Communication

Communication

- firewalls
- routing scheme
- encryption
- temporal network partition
- packet validation

PUCRS

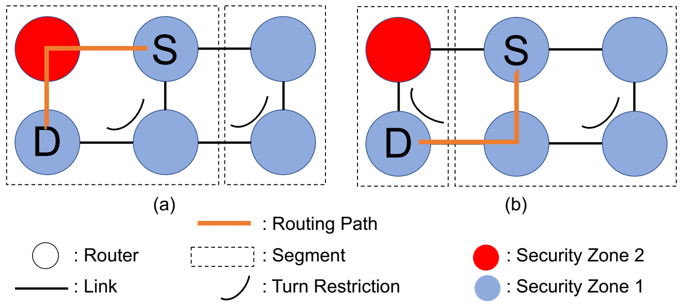
Routing Scheme

Sepúlveda et al. (2015)

- Threat model: SCA
- **adaptive routing and random arbitration**
  - random arbitration - remove the **temporal correlation** of malicious injected traffic and memory access.
  - adaptive West-First routing method, to make turns to escape from blocking conditions

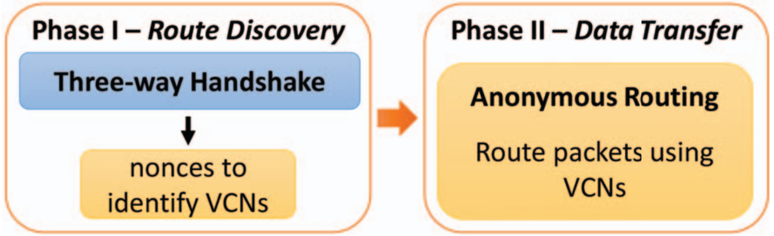
Fernandes et al. (2016)

- Create Secure Zones at design time
- Configure **routing tables** to avoid DoS and timing side channel attacks



Charles et al. (2020)

- **Anonymous routing** using virtual circuit numbers (VCN)
- Two phases method
  - **Route Discovery** - PE sends a packet to discover the route and distributes parameters among participants
  - **Data Transfer** - the path set is used to transfer messages from S to D anonymously.



Indrusiak et al. (2019)

- **route randomization**
- varying the routes taken by sensitive traffic prevents the collision with malicious traffic making the SCA information extraction harder since the timing measures are not precise

4a. Protecting Communication

Communication

- firewalls
- routing scheme
- encryption
- temporal network partition
- packet validation

PUCRS

Encryption

Ancajas et al. (2014)

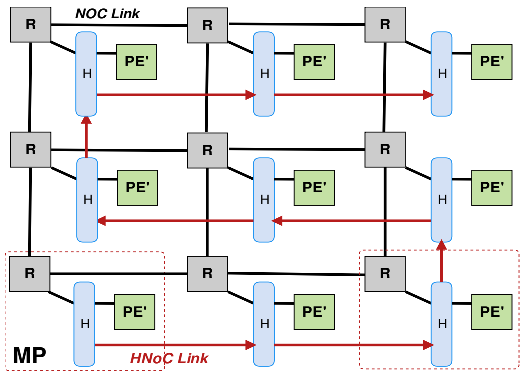
- Assumes **HT**
- Three-layer security mechanism
  - **Data Scrambling,** XOR cipher encryption (lightweight cryptography)
  - **Packet Certification,** attaches an encrypted tag at the end of the packet
  - **Node Obfuscation (NObf),** decouples the source and destination nodes using task migration

Zeferino et al. (2017)

- Use an **AES** block and a KDC (Key Distribution Center), adding authenticity and confidentiality in the message flow of the SoCIN NoC.

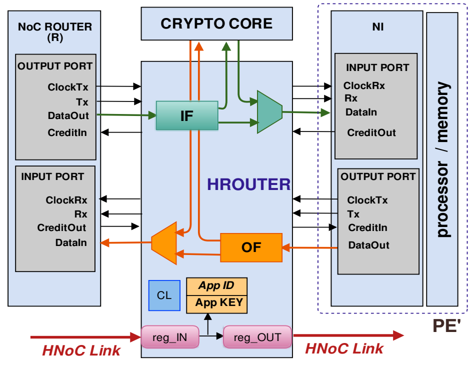
Oliveira et al. (2018) / Santanta (2021)

- Protects against DoS, MitM
  - spatial isolation of applications
  - a dedicated network to send sensitive data
  - filters to block malicious traffic (*simple firewall*)
  - AES or lightweight cryptography



(a)

Dedicated NoC (in red)



(b)

Filter with AES (in red)

Encryption drawbacks: crypto core area / latency to encrypt/decrypt

4a. Protecting Communication

Temporal network partitioning (TNP)

- explicit flow separation to avoid interference of low-priority flows in high priority flows
- mitigate DoS, timing side-channel attacks and information leakage

Wassel et al. (2014)

- design time method to create domains of non-interference between flows
- use of virtual channels, priority arbitration, called **surf scheduling**
- a packet waits in one dimension (X), after finishing the first dimension, the packet might experience another wait until it can be forwarded to the next dimension (Y).
- Drawback: increasing the number of domains also increases the number of virtual channels, increasing the router area and power consumption.

Wang et al. (2012)

- design time priority-based arbitration
- assign high-priority to low-security traffic, in such way that its behavior is not affected by high-security traffic.
- Virtual channels are statically allocated to each security domain to remove interference in buffers.

4a. Protecting Communication

Communication

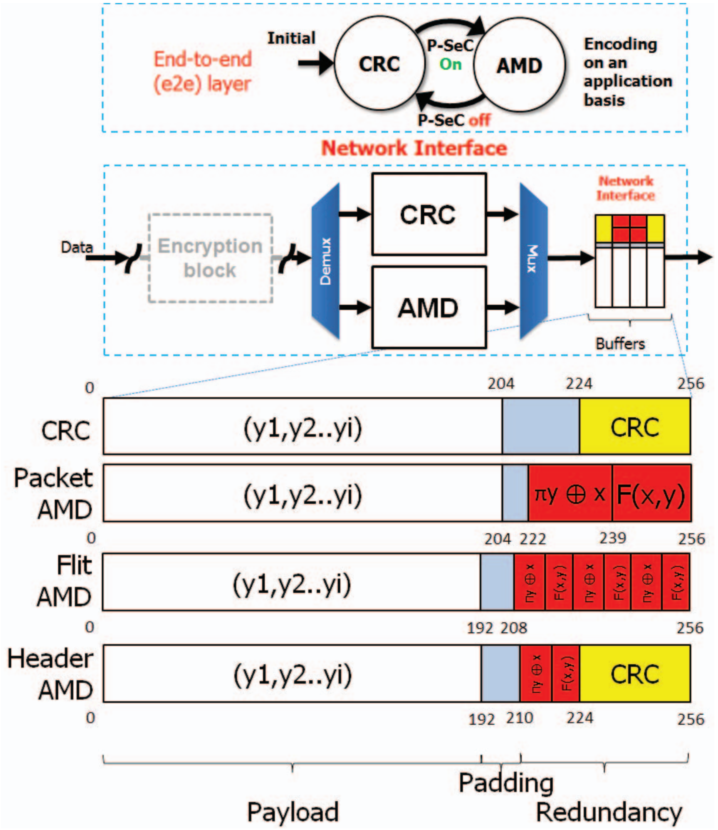
- firewalls
- routing scheme
- encryption
- temporal network partition
- packet validation

Packet Validation

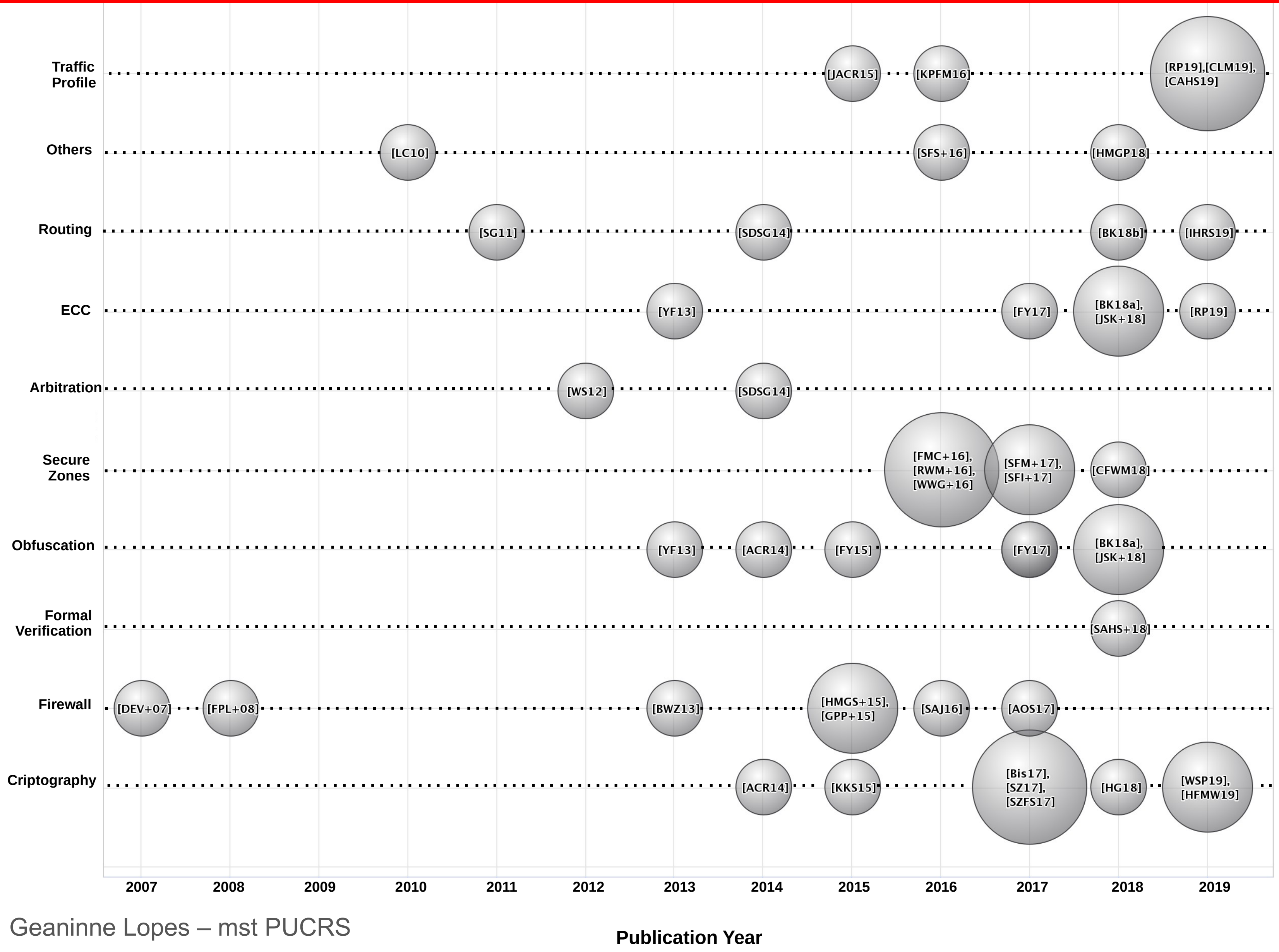
Boraten et al. (2016)

- runtime packet-security (P-Sec) method, protecting against SCA, DoS, HTs
- adopts two error detection schemes
  - cyclic redundancy check (**CRC**) codes
  - algebraic manipulation detection (**AMD**)

- Overhead: increases packet size



4a. Protecting Communication – global view



Methods:
- Firewalls (access control)
- TNP (avoid temporal sharing)
- Secure Zones (avoid flow sharing)

Prevent:
- Access control attacks
- DoS
- Timing SCA
- Hardware trojans

4b. Protecting Computation

Computation

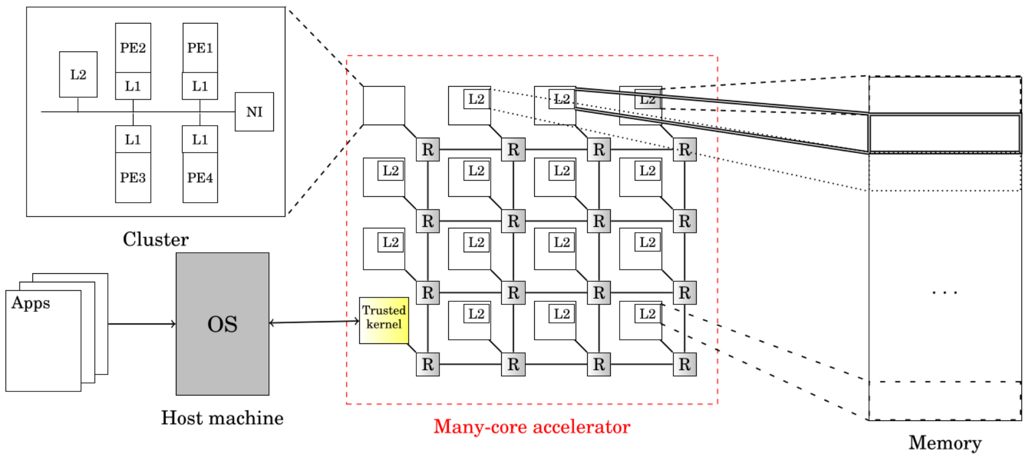
- logical and spatial isolation (clusters)
- ARM TrustZone (ATZ)

PUCRS

The computation protection include mechanisms to avoid processors' sharing between distinct applications

Real et al. (2018)

- logical and spatial isolation of sensitive applications through the **dynamic creation of secure zones** (SZ) to mitigate DoS and cache SCA attacks at runtime
- hybrid architecture, with a 2D-Mesh NoC where each router is connected to a **cluster** with 4 processors, 1 shared memory and 1 shared bus
- **only cluster resources are isolated by the SZ**
- if a task needs to communicate with a task in another cluster the message is sent through an insecure channel



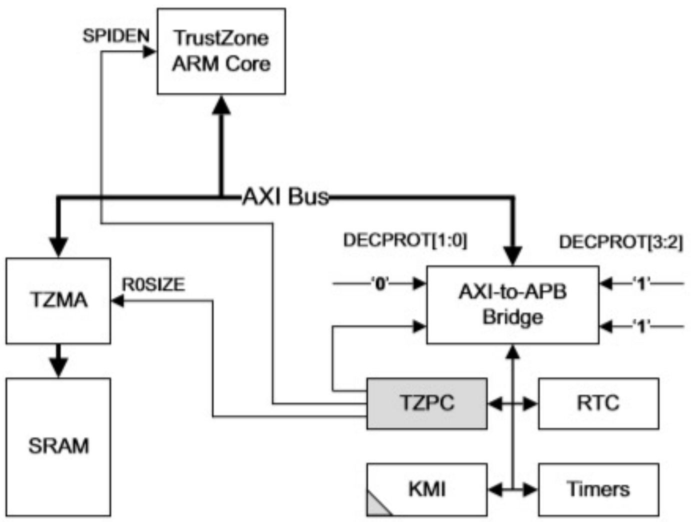
4b. Protecting Computation

Computation

- logical and spatial isolation (clusters)
- ARM TrustZone (ATZ)

ARM TrustZone – ATZ (2018)

- isolation of applications in the **same** processor (spatial isolation)
- hardware support for the creation at runtime of **Trusted Execution Environments** (TEEs)
- creates two virtual processors and two Memory Management Units, allowing to execute a secure and a non-secure application simultaneously
- **drawback**: in many-core systems applications running on different processors share resources such as the NoC/buses and memory



Methods:
- Secure Zones → isolation

Ensures:
- data integrity
- confidentiality
- access control

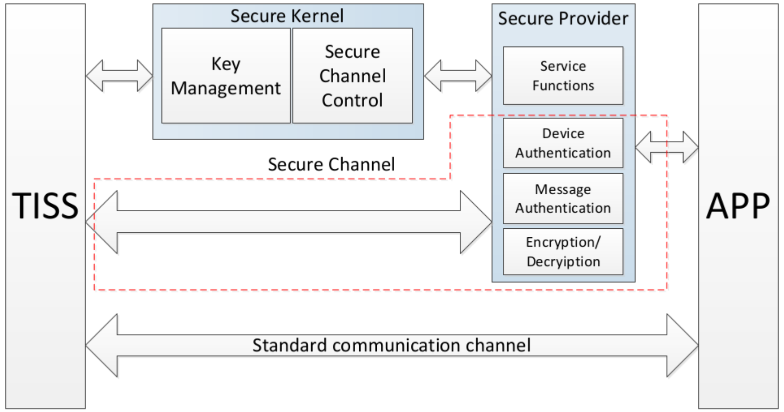
4c. Protecting Comm. and Comp.

Comm. and Comp.

- secure Zones - partition and encryption
- secure Zones - spatial isolation and encryption
- obfuscation

Isakovic et al. (2013)

- Secure zones: computation and communication protection using **spatial isolation** with **encryption** mechanisms
- architectural partitioning of the MPSoC resources at **design time**
- mechanisms: secure microkernel secure channel infrastructure that includes cryptography and firewalls



TISS – Trusted Interface Subsystem

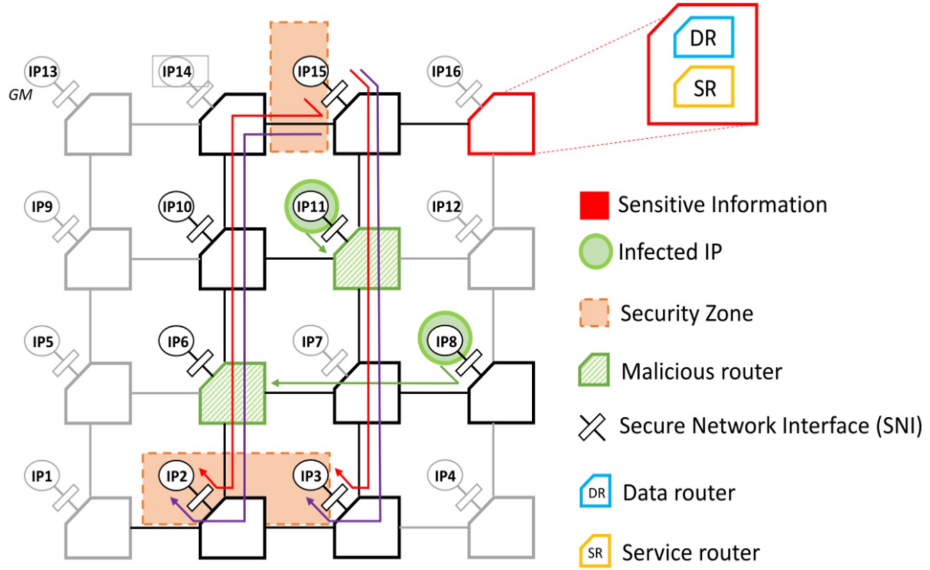
4c. Protecting Comm. and Comp.

Comm. and Comp.

- secure Zones - partition and encryption
- secure Zones - spatial isolation and encryption
- obfuscation

Sepúlveda et al. (2017)

- **Protects computation** - spatial isolation through secure zoned
  - non-continuous SZ, defined at runtime
- **Protects communication** - cryptography (DH and XOR)
- Two NoCs:
  - data NoC, used by the application data
  - service NoC, used to exchange the security control packets
- After mapping the application, a key agreement protocol is executed between the mapped PEs using the service NoC.
- The encryption/decryption is obtained XORing the message with the shared key
- Ensure data integrity, confidentiality and availability

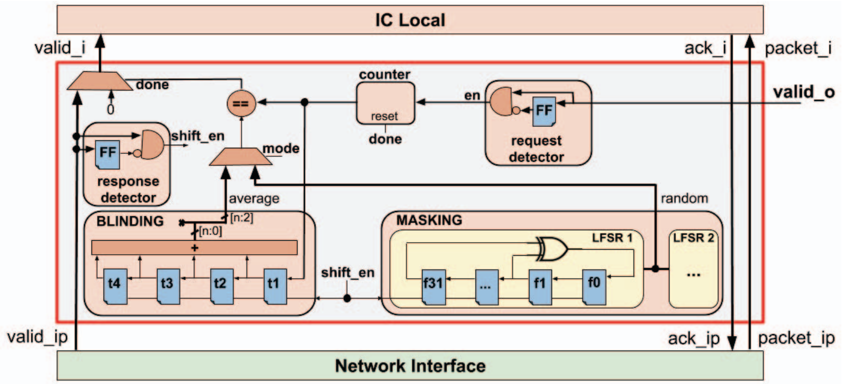


4c. Protecting Comm. and Comp.

PUCRS

Reinbrecht et al. (2020)

- **Obfuscation technique**
- 3 techniques to prevent timing attacks:
  - blinding - changes the response time to have a constant value
  - masking - insert delays on the responses, operating as a noise source
  - dual communication strategy - use packet and circuit switching simultaneously (secure flows: packet switching)
- Blinding and masking: protects computation
- Dual communication strategy: protects communication



4d. Discussion

Communication

- most works related to the security protect just the communication subsystem
- Several works adopt design time methods
  - ✓ **Pros**: enable the adoption of sophisticated and robust algorithms
  - ✓ **Cons**: design time methods are not applicable in dynamic workload scenarios.
- The most common and intuitive approach to protect communication is **encryption** - provides data confidentiality but still expose the traffic to DoS and timing SCA attacks
- **Firewalls** ensure access control to the communication system, avoiding DoS attacks and minimizing the possibility of data extraction by a malicious process
- **TNP** provide temporal and logical traffic isolation avoiding the interference on secure flows, enabling communication availability and timing SCA attacks protection

Computation

- adopts temporal, logical or spatial **isolation** as main mechanism

Communication + Computation

- **spatial isolation** and **encryption**

5. Protecting Ext. Communication

Secure Admission | Appsec Execution | Protected I/O access

Reinbrecht et al. (2017)

- **Mitigate SCA attacks to memories**
- Gossip NoC combines two strategies to protect the MPSoC against timing SCA:
  - **detection**, which includes a bandwidth monitoring and a gossip message generation in the presence of an abnormal behavior that enables the second strategy
  - **protection**, triggered when any gossip message is received and modify the routing (XY routing algorithm to the YX)

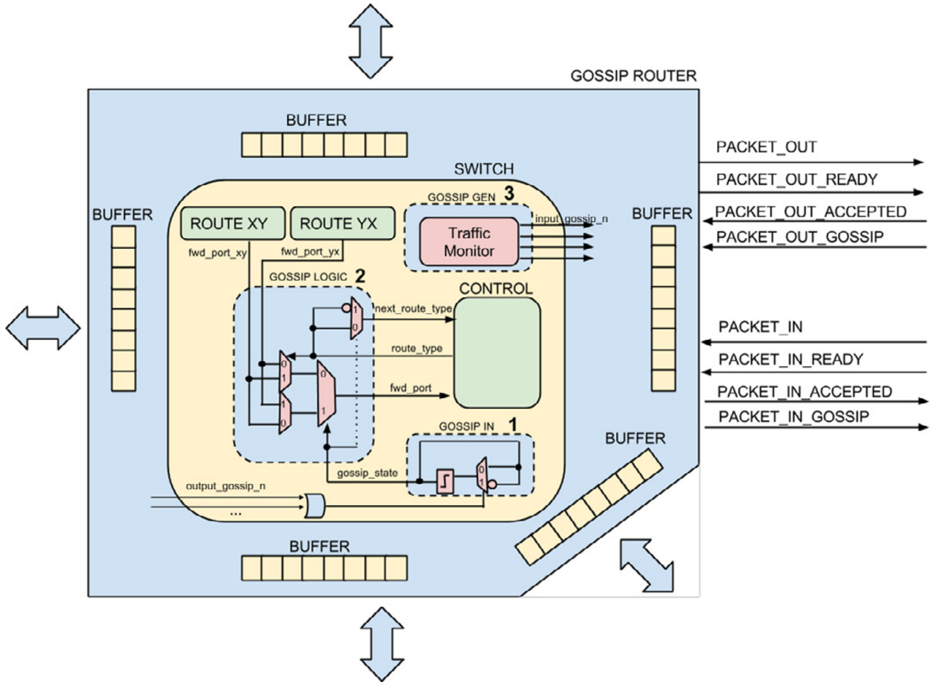


Fig. 5. *Gossip router* microarchitecture: (1) Gossip in block; (2) Gossip logic; (3) Gossip generator.

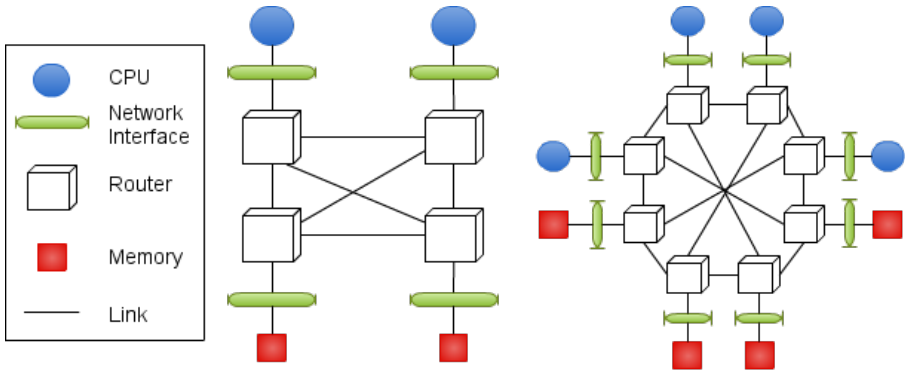
5. Protecting Ext. Communication

Secure Admission | Appsec Execution | Protected I/O access

PUCRS

Grammatikakis et al. (2015)

- **firewall** at the NI which, by checking the physical address against a set of rules, rejects untrusted CPU requests to the on-chip memory
- firewall has three modules:
  - operating mode controller (OMC), that accepts, decodes and dispatches NoC firewall commands;
  - segment-level rule-checking (SLRC), processes incoming memory accesses and configuration commands;
  - the interrupt unit (INTU) that accepts interrupt requests from the OMC and SLRC modules and reports interrupt contexts to the CPU



(a) 4-node STNoC (b) 8-node STNoC

Few works in the literature concerned with IO accesses

6. Security Methods Proposals

6.1 Lightweight security mechanisms

Security Vulnerabilities and Countermeasures in MPSoCs
SANTANA, Anderson ; MEDINA, Henrique ; MORAES, Fernando Gehm.
IEEE Design & Test, January 2021.

6.2 SDN – Software Defined Networking

SDN-Based Secure Application Admission and Execution for Many-Cores
RUARO, Marcelo; CAIMI, Luciano; MORAES, Fernando Gehm
IEEE Access, v.8, pp. 177296-177306, September 2020.

6.3 OSZ – Opaque Secure Zones

Security in Many-Core SoCs Leveraged by Opaque Secure Zones
CAIMI, Luciano; MORAES, Fernando Gehm
In: ISVLSI, 2019, pp. 471-476.

ISCAS'17, ICECS'18, LASCAS'18

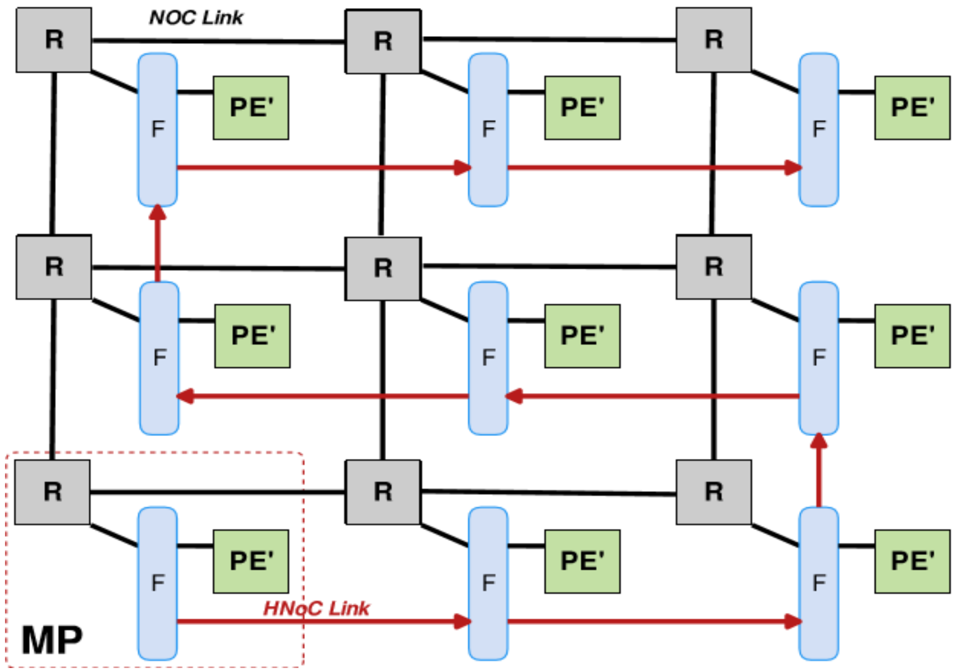
6.1 Lightweight security mechanisms

D&T 2021

- dedicated network to send sensitive data
- spatial isolation of applications
- filters to block malicious traffic (simple firewall)
- AES or lightweight cryptography

Dedicated network:

- loosely connected to the MPSOC
- serial Hamiltonian path that runs through all PEs
- small area footprint: 2 ports instead of 5 of a standard 2D-mesh, no need to add input buffers
- only the MP may inject data into it (root-of-trust)
- MP injects cryptographic keys and application/task identifiers



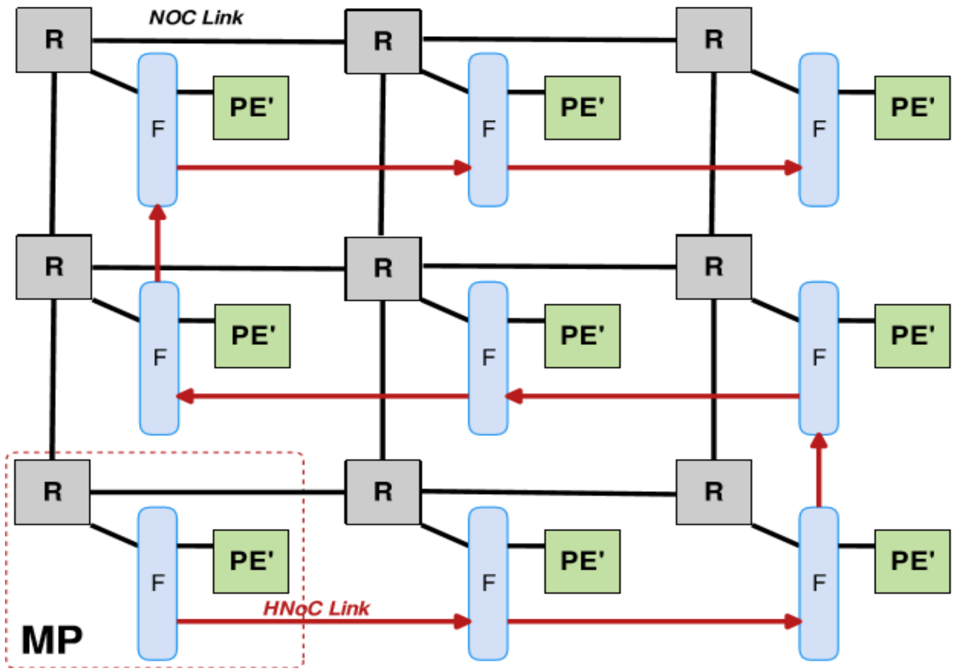
6.1 Lightweight security mechanisms

D&T 2021

- dedicated network to send sensitive data
- spatial isolation of applications
- filters to block malicious traffic (simple firewall)
- AES or lightweight cryptography

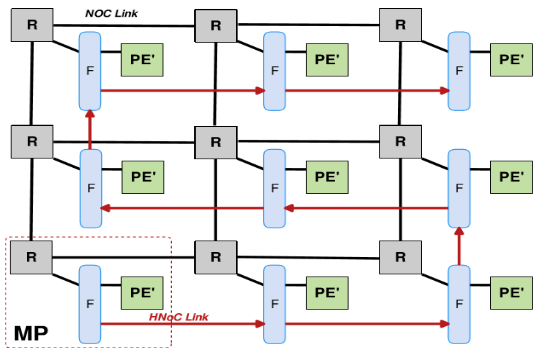
Spatial isolation of applications:

- new restriction in the task mapping: *tasks belonging to different applications cannot share the same processor*
- restricting task mapping prevents malicious tasks from running on the same processor, thus preventing a malicious task from accessing sensitive data, ensuring security at the computation level



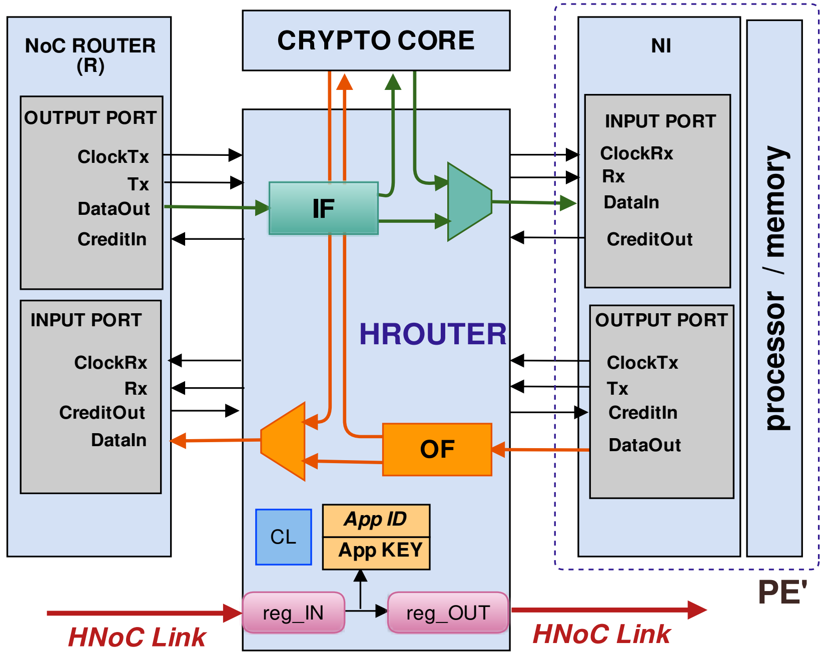
6.1 Lightweight security mechanisms

D&T 2021



Filters to block malicious traffic:

- MP configures the filters during mapping
- **OF** (output filter) tags the packets entering the NoC with the correct application identifier, dropping all other packets
  - prevents tasks from forging an App_ID, avoiding the execution of attacks
- **IF** (input filter) admits of two packet types: packets that match the App_ID or management packets
  - IF discards all other packets.
  - avoids attacks as DOS



6.1 Lightweight security mechanisms

D&T 2021

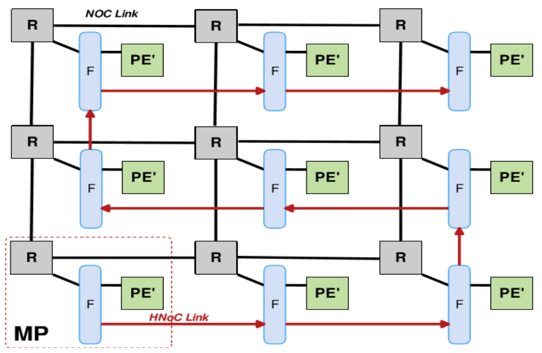
AES or lightweight cryptography:

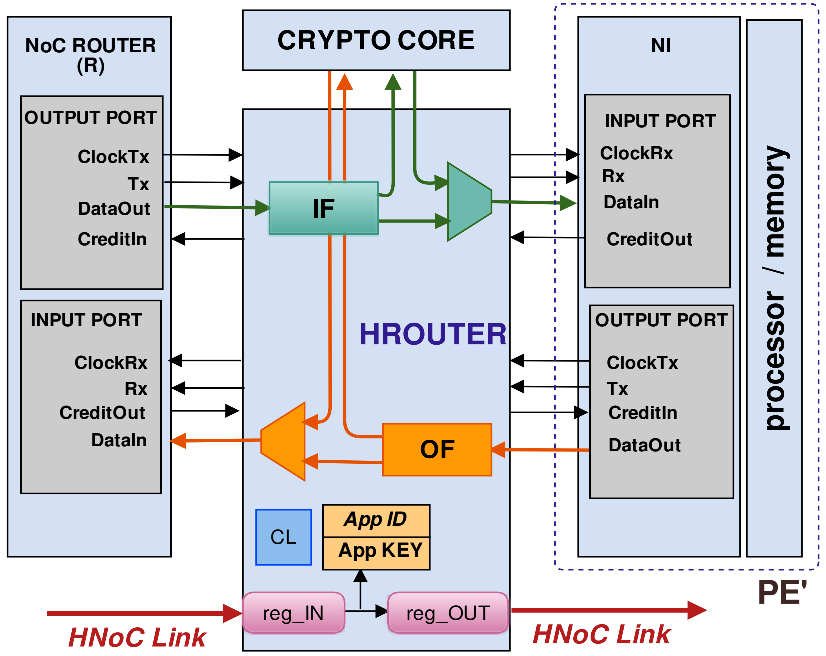
- crypto core – design choice

Table 1 - AES and SIMON comparison – 65 nm technology.

	SIMON	AES
Latency (clock cycles)	70	19
Area (μm²)	22,371	105,316
Cell Count	4,076	20,5634
Power (μW)	16,033	399,233

Evaluating the Cost to Cipher the NoC Communication
OLIVEIRA, Bruno; REUSCH, Rafael; MEDINA, Henrique; MORAES, Fernando Gehm
In: LASCAS, 2018





- ➢ Non-intrusiveness is the **keyword** of this work
- ➢ HNoC: is generic, with a small area footprint
- ➢ Software level: restrictions in the task mapping heuristic and distributing sensitive data to HNoC

Application is protected, but traffic shared in the NOC → DoS, SCA is possible

6.2 SDN – Software Defined Networking

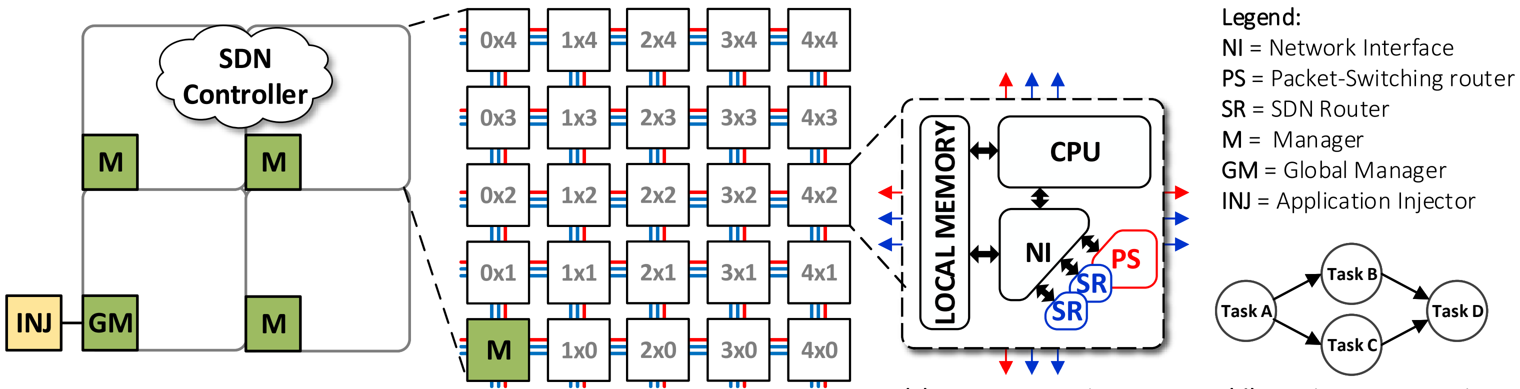
IEEE Acess 2020

What is SDN :

- Software Defined Networking (SDN): simplify network management and reduce routers' cost
- Reduced hardware complexity
- Flexible management to support different objectives

Architecture

- MPN – multiple physical networks
- 1 PS subnet
- *n* SDN subnets – circuit switching
- SDN configures paths



(a) MCSoC: Cluster-based Manag. (b) Cluster Architecture (c) Processing Element (d) Application Graph

PS subnet

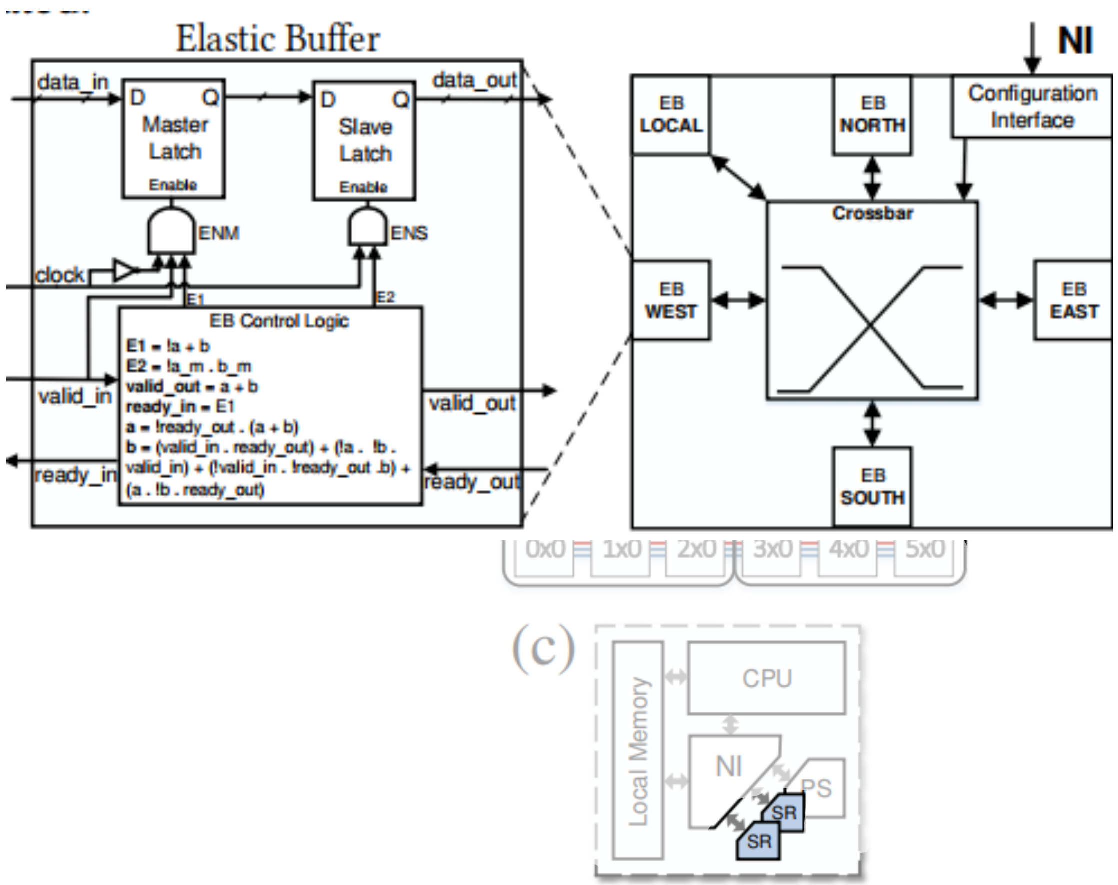
HERMES: an infrastructure for low area overhead packet-switching networks on chip — 768 — 2004
F Moraes, N Calazans, A Mello, L Möller, L Ost
Integration 38 (1), 69-93

6.2 SDN – Software Defined Networking

IEEE Acess 2020

SDN Router (SR)

- Simple forwarding unit
- Connects a given *inport* to a given *outport*
- Use **Elastic-Buffers** instead input buffers (low area overhead – 20% of PS)
- **Configuration interface**
- Network Interface (NI) programs the SR routers according to configuration packets sent by the SDN Controller

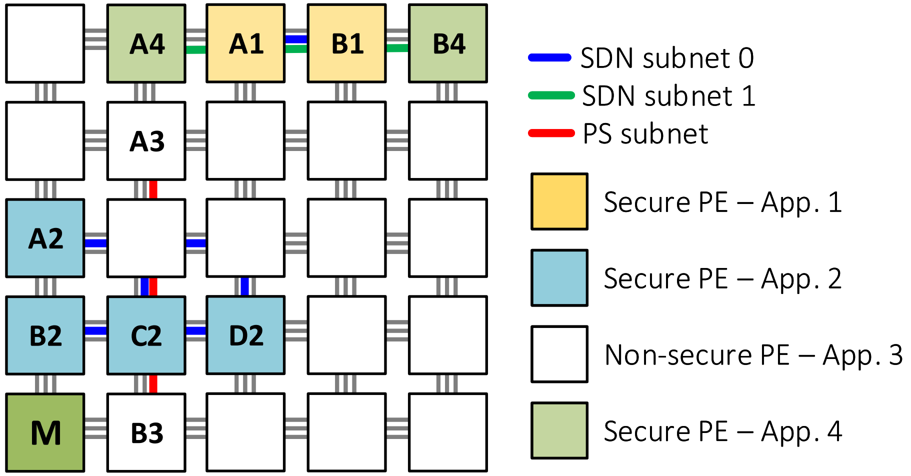


6.2 SDN – Software Defined Networking

IEEE Acess 2020

Method:

- Disjoint SZ with circuit-switching communication



5 Steps

1. Initialization

- executes once, at system startup
- Elliptic Curve Diffie–Hellman Key Exchange (ECDH) protocol ➔ K_e

2. Application Admission

- Request of a new application, authenticated by K_e

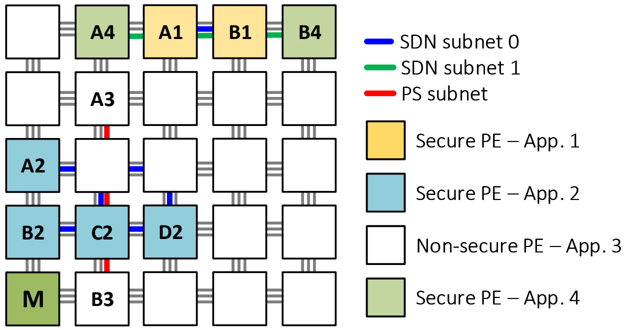
6.2 SDN – Software Defined Networking

IEEE Acess 2020

Steps

3. SDN-based secure task mapping

- mapping with spatial isolation
- SDN controller must ensure the availability of CS paths
- complex protocol with security ensured at all steps

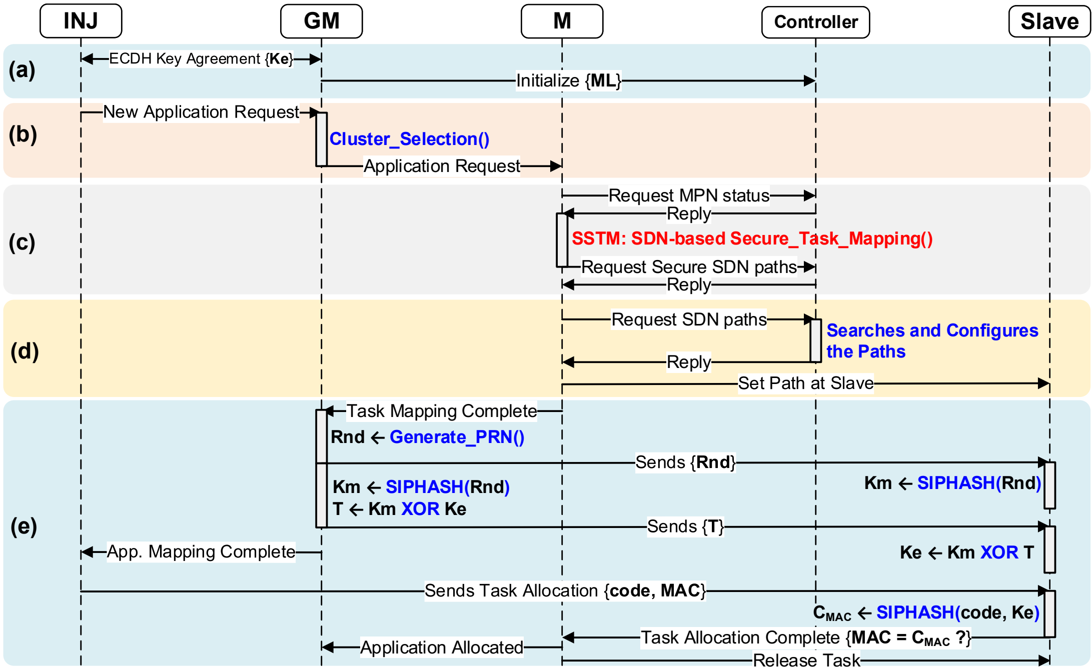


4. SDN connections establishment

- SDN controller configures the the SR using the PS subnet

5. Secure task loading

- MAC verification for each task



6.2 SDN – Software Defined Networking

IEEE Acess 2020

Originality

- communication and computation protected
- provide security to applications by dynamically establishing circuit switching using SDN
- better system utilization due to non-continuous regions
- offers communication integrity, leading to data transmission without the overhead of encryption, arbitration, and routing required in PS NoCs

Avoided threats

- DoS attacks, are prevented due to the resources' isolation at the application communication level
- timing attacks are prevented since no time inferences can be taken from packets in CS channels

Cost

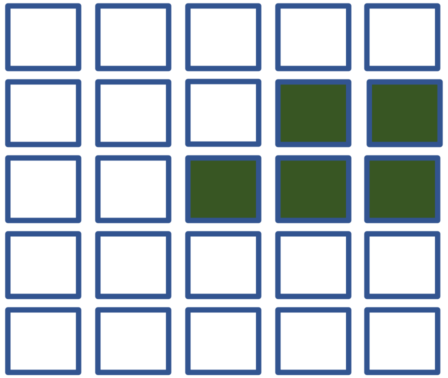
- application admission latency due to the SDN execution for finding paths between communicating tasks, but it is negligible for the end-user (below 1 ms)

Open issues

- protection of the packet switching network to prevent DoS attacks
- definition of a method for safely communicate with peripherals

6.3 OSZ - Opaque Secure Zones

ISVLSI 2019



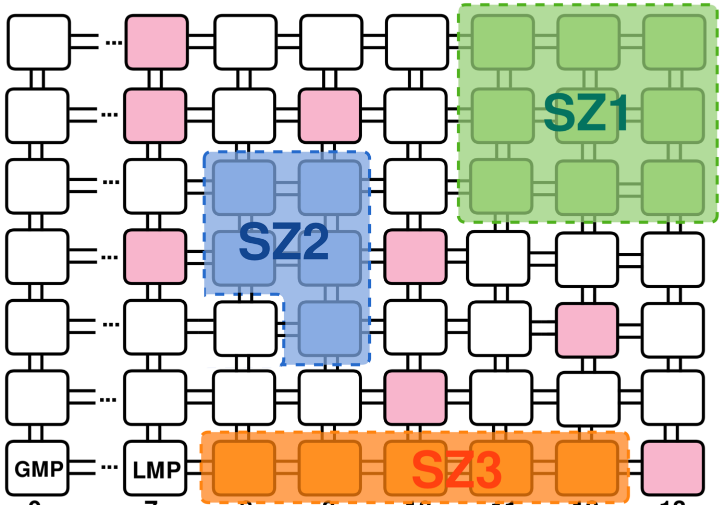
Creation time: runtime

Shape: continuous rectilinear

Communication sharing: avoided

Computation sharing: avoided

Methods: temporal-spatial isolation and rerouting

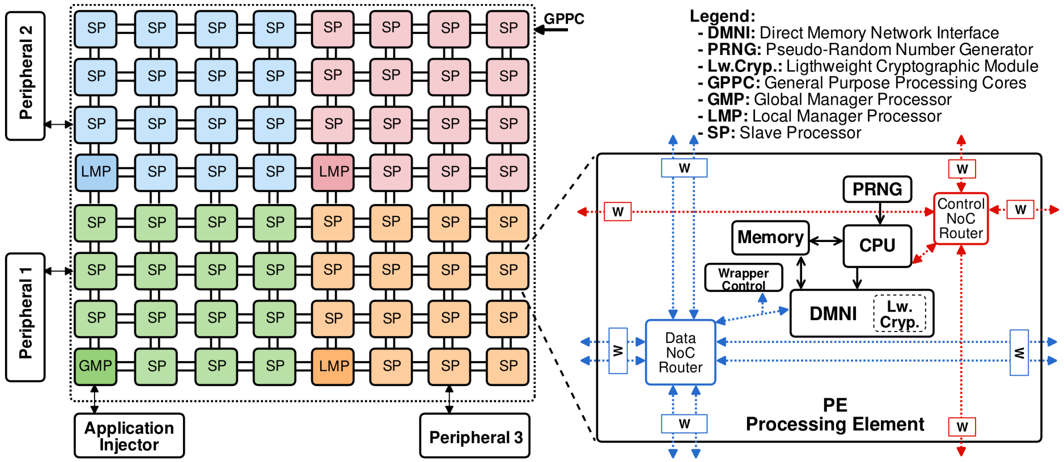


Multiple OZs coexist simultaneously

6.3 OSZ - System Architecture

ISVLSI 2019

NoC-based many-core system with peripheral support



(a) (b)

PE

- 32 bits MIPS-like Processor
- DMNI module
- Local dual port memory
- Data NoC router
- Control NoC router
- Wrappers

Data NoC

- Duplicated physical channels
- Wormhole packet switch
- Support to XY and source routing
- Input buffer, 8-flit depth
- 16 bits flit length

Control NoC - *BrNoC*

- Broadcast as default transmission mode
- Small area footprint: centralized buffer using an 8-entry CAM (content-addressable memory) memory

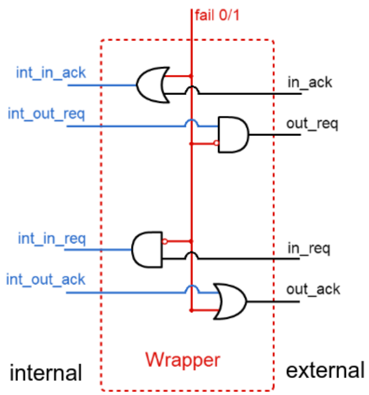
BrNoC: a Broadcast NoC for Control Messages in Many-core Systems
WACHTER, Eduardo; CAIMI, Luciano; FOCHI, Vinicius; MUNHOZ, Daniel; MORAES, Fernando Gehm
Microelectronics Journal, Volume 68, October 2017, Pages 69–77.

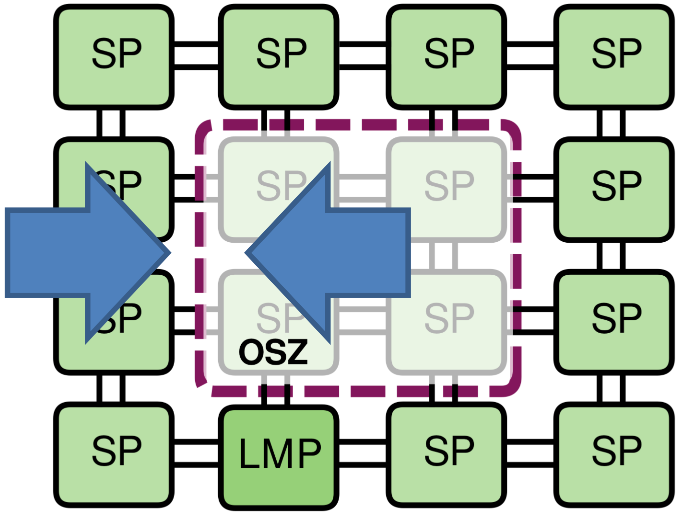
6.3 OSZ - General View of the Method

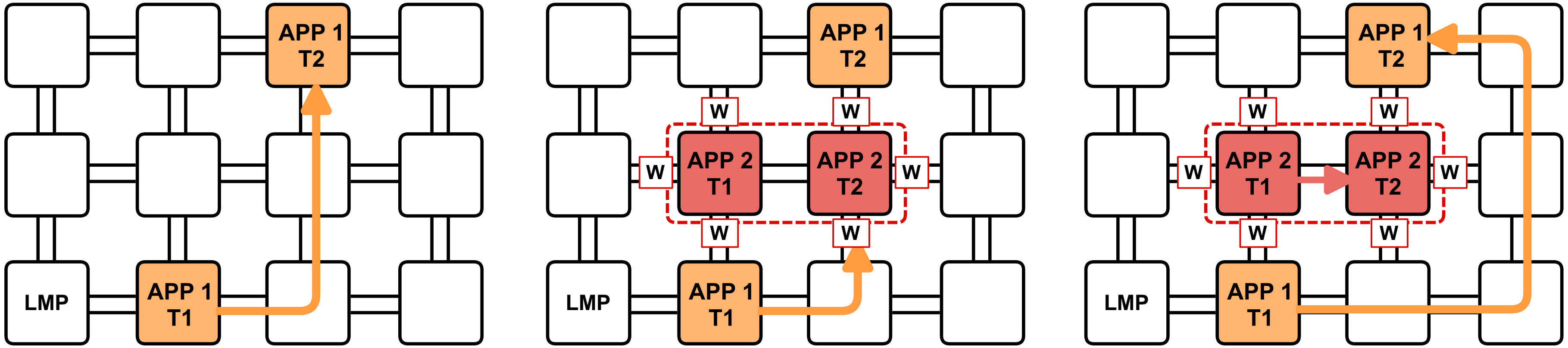
ISVLSI 2019

Method

- Secure Application admission (ECDH)
- **Create the OSZ with wrappers**
- Launch App
- **Reroute packets outside OSZ**
- Notify ended tasks to LMP (manager processor)
- Clear memories of PEs and open the OSZ







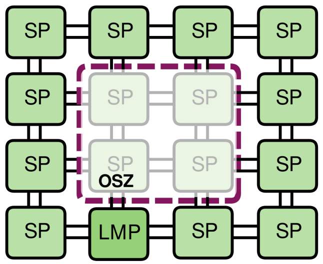
(a) (b) (c)

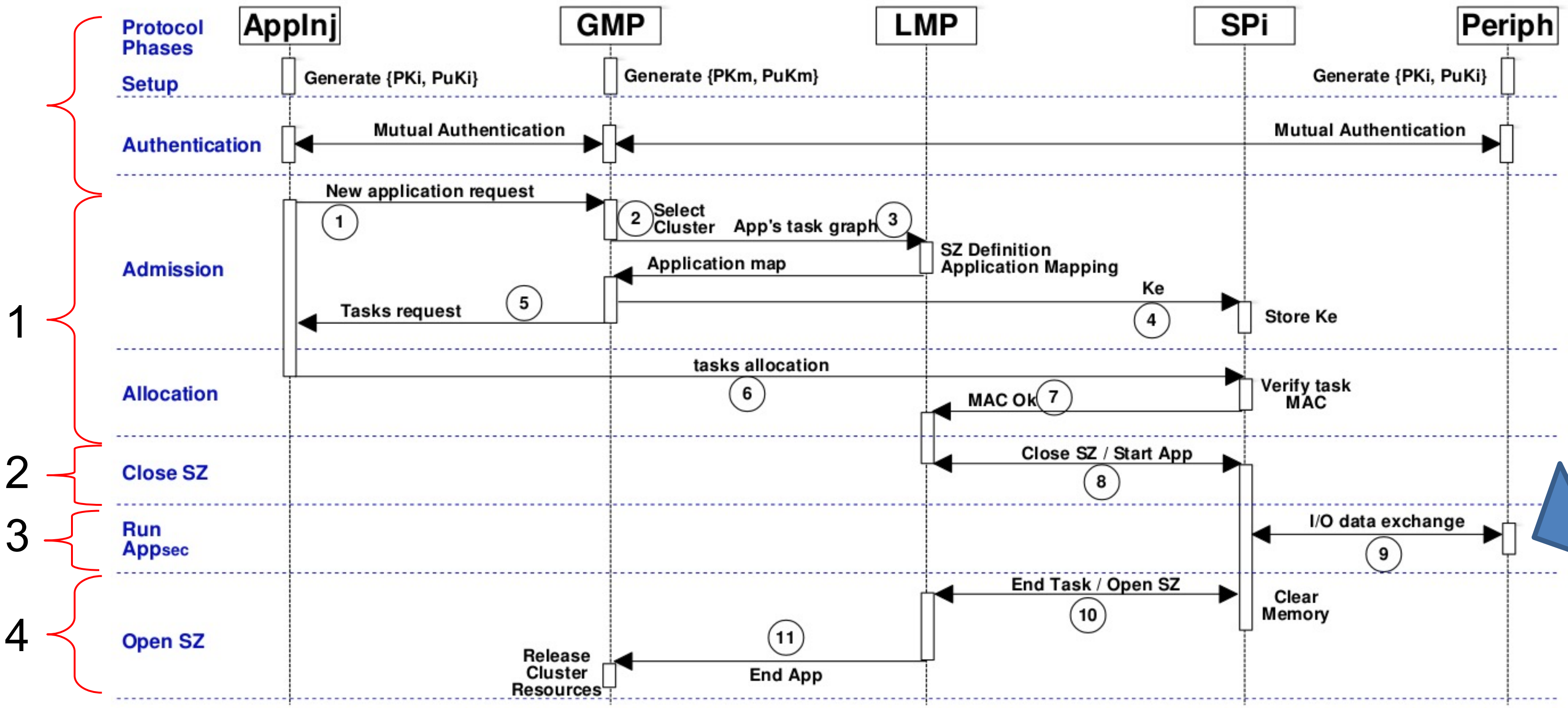
6.3 OSZ - General View of the Method

ISVLSI 2019

Advantages

- No need to encrypt the application data
- All attacks related to communication and computation sharing are avoided
- Small hardware cost: brNoC + wrappers





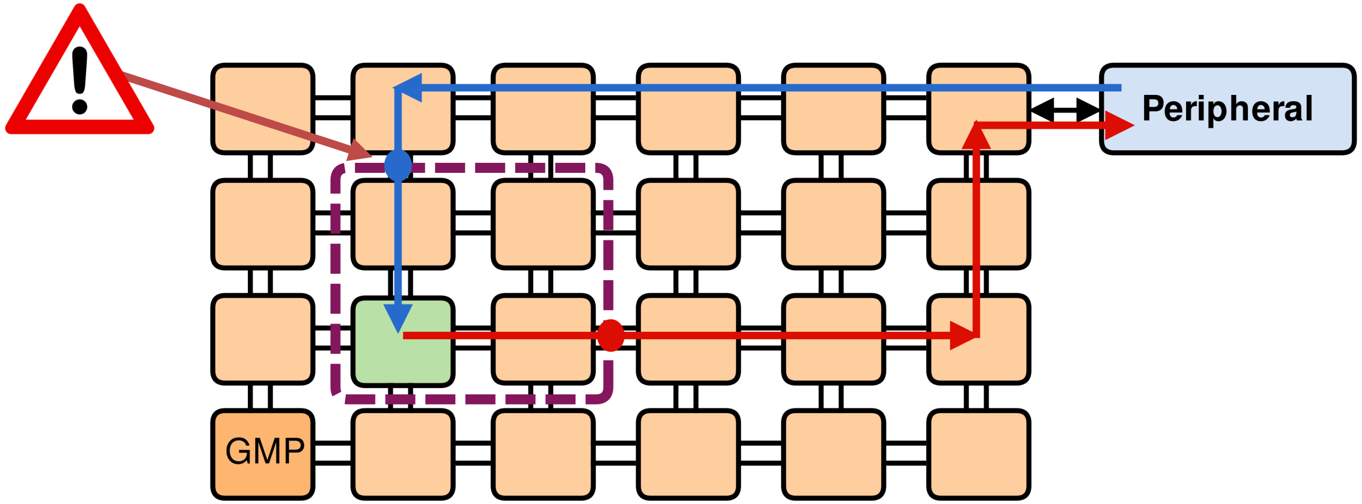
Secure IO communication

6.3 OSZ - Opaque Secure Zones

ISVLSI 2019

Communication with peripherals: selective opening of access points

- Communication with peripherals uses master/slave approach
- API differentiate inter-task messages from I/O messages: *IO_Send()* or *IO_Receive()* – packets protected with a MAC
- Opened wrapper to send data: no security issue to app inside OSZ. But threat to I/O message outside OSZ
- Opened wrapper to receive data: security issue
- I/O messages can be encrypted (confidentiality)



6.3 OSZ - Opaque Secure Zones

ISVLSI 2019

- OSZs: original procedure to mitigate resource sharing
  - runtime execution with several SZs co-existing in parallel
  - internal OSZ communication **without** cryptography, not penalizing the execution time of the secure application
- Robust method to enable OSZs to communicate with I/O devices
- Issues:
  - **Attacks from HTs**
  - Key exchange with peripherals (NI?)
  - Standard NI with lightweight cryptography

Open-source NoC-based Many-Core for Evaluating Hardware Trojan Detection Methods
WEBER, Iaçanã ; MARCHEZAN, Geaninne ; CAIMI, LUCIANO L. ; MARCON, CESAR A. ; MORAES, Fernando Gehm
doi In: ISCAS, 2020

microeletrônica
sistemas embarcados
arquitetura de microprocessadores embarcados
NoCs
many-cores
Segurança
IoT
circuitos e sistemas assíncronos/GALS
aplicações em telecomunicações

Thanks!

Fernando Gehm Moraes
fernando.moraes@pucrs.br
https://www.inf.pucrs.br/moraes