Fortifying NoC-based Manycores: Distributed Monitoring to Detect Security Threats

Rafael Follmann Faccenda*, Gustavo Comarú*, Ney Calazans‡, Luciano Lores Caimi†, Fernando Gehm Moraes*
*School of Technology, Pontifical Catholic University of Rio Grande do Sul – PUCRS – Porto Alegre, Brazil
{gustavo.comaru, rafael.faccenda}@edu.pucrs.br, fernando.moraes@pucrs.br
‡Computing Department - DEC, Federal University of Santa Catarina – UFSC – Araranguá, Brazil – nlvcalazans@gmail.com
†Federal University of Fronteira Sul – UFFS – Chapecó, Brazil – lcaimi@uffs.edu.br

***Abstract*—As computing technology advances, manycore systems have become increasingly used due to their performance and parallel processing capabilities. However, these systems also present significant security risks, including threats from hardware Trojans, malicious applications, and peripherals capable of executing various attacks, such as denial-of-service, spoofing, and eavesdropping. Researchers have proposed several security enhancements to address these challenges, including establishing secure zones, authentication protocols, and security-aware routing algorithms. This paper introduces a distributed monitoring mechanism that can detect suspicious activities within NoC links, peripheral interfaces, and packet reception protocols, explicitly focusing on detecting DoS, spoofing, and eavesdropping attacks. We conducted a comprehensive attack campaign to evaluate the effectiveness of our monitoring mechanisms and test the platform's resilience. The results were promising, with the system successfully detecting all attacks and continuing to execute applications correctly. Although there was an average execution time penalty of 8.83%, which included the time taken to generate attack warnings from the monitoring mechanisms and apply countermeasures, this mechanism significantly enhanced the security of manycore systems.**

***Index Terms*—Security, NoC-based Manycores, Security Threat, Monitoring, Security Evaluation.**

I. Introduction

Manycore systems achieve enhanced performance by leveraging parallel processing, meeting the growing needs of embedded devices that require efficient power usage and constrained communication capabilities. Such systems contain Processing Elements (PEs) interconnected by a Network-on-Chip (NoC). The NoC architecture, which includes routers and links, facilitates the transmission of data and control messages among the PEs. The Network Interfaces (NIs) connect PEs to the NoC routers.

As the adoption and complexity of manycore systems increases, data protection concerns appear as a design requirement [1]. A manycore may be employed in scenarios where availability is critical, and downtimes must be minimized. These systems may also handle sensitive information; thus, it is necessary to protect this data from unauthorized access [2]. According to [3], data protection, unauthorized access, and availability are major concerns on the manycore design.

Adversaries can exploit manycore vulnerabilities and execute several attacks using malicious implants (Hardware Trojans – HT) or via third-party IPs [4]. For this reason, it is necessary to have monitoring features that can observe the system activity and detect suspicious behavior. Then, based on the reported irregularity, apply the most suited countermeasure.

This paper addresses the security vulnerabilities inherent in NoC-based manycores by proposing a distributed monitoring infrastructure capable of identifying and responding to various attack types. The threat model considers Denial-of-Service, Spoofing, Data Corruption, and Eavesdropping (Section II). Each type of attack is paired with specific countermeasures designed to protect the manycore. The *primary objective* is to enhance the security of these systems through a distributed monitoring and the dynamic deployment of countermeasures tailored to the nature of detected threats.

The *original contribution* of this paper is the development of a distributed monitoring infrastructure. The distributed monitoring infrastructure requires a set of architectural features, which are discussed in Section III. These features are not platform-specific; thus, different manycores can adopt them to enhance their security.

II. Threat Model and Countermeasures

This section discusses harmful behaviors that may affect manycores, considering threats reported in the literature and the applied countermeasures adopted for each attack.

This paper considers three types of attacks: malicious applications ($MalApp$), malicious hardware (HTs), and malicious peripherals ($MalPeriph$). $MalApp$ refers to harmful software loaded into the platform. HTs are intrusive hardware elements placed into the integrated circuit during design or fabrication [5]. $MalPeriphs$ represents external devices connected to the platform that can harm applications with IO communication.

Below, we present the types of attacks that can be carried out on a manycore.

Denial-of-Service (DoS). DoS includes attacks that deny or hinder the operation of a given system function. Flooding, packet loss, and misrouting are common attacks executed at the communication level. Zhang et al. [6] explore DoS attacks known as Blackhole and Sinkhole, executed by HTs inside routers. The Distributed Denial-of-Service (DDoS) [7] consists of several compromised IPs performing DoS attacks simultaneously. Yao et al. [8] describe a DoS attack targeting the arbiters of input ports within a router's switch allocator to disrupt packet transmission.

Side-Channel Attacks (SCA). SCA gathers information through direct interference and eavesdropping. The attacker may discover important information about the manycore based on the retrieved data. The SCA is the primary type of attack associated with leaking or stealing sensitive information,

This work was financed in part by CAPES (Finance Code 001), CNPq (grants 309605/2020-2, 407829/2022-9 and 311587/2022-4), and FAPERGS (grant 21/2551-0002047-4).

which can be from different natures, such as current [9], power dissipation, electromagnetic irradiation, or timing information.
Spoofing. Attack characterized by a malicious source (hardware, IO, or software) that falsifies its identity to obtain unauthorized privileges [10].
Data Corruption Gondal et al. [11] present an HT that corrupts messages to disturb communication. Based on the effects of this HT, they propose an error correction mechanism.
Eavesdropping Eavesdropping is the direct stealing of information from applications. Raparti and Pasricha [12] present a data-snooping promoted through message duplication and redirection. The attack is performed by a router infected with an HT. Kulkarni et al. [13] implement an HT as a small circuit installed in the router's input buffers. The HT attack, which is randomly triggered, manipulates the header flit, changing the packet target and redirecting it to another PE.

State-of-the-art shows that a manycore must provide a diverse suite of countermeasures tailored to the nature of the attack. Thus, this research contributes to manycore security by proposing a distributed monitoring infrastructure to identify the attack category and fire the corresponding countermeasure.

III. Monitoring Mechanisms

Monitoring is the first step in detecting an attack. Therefore, distributed monitoring mechanisms that can supervise the NoC and PEs, sending warning reports upon detecting suspicious behaviors, are mandatory. The distributed monitoring uses six general mechanisms presented below, alongside the specific proposals adopted in this work.

1) **Secure Zones (SZ)**: regions of the manycore with computation and communication resources reserved for applications with security constraints. This work adopts Opaque Secure Zones (OSZ) [14], which establishes rectangular SZs at runtime that isolate computation and communication using a method that blocks incoming and outgoing traffic at the OSZ borders.
2) **Link Control (LC)**: configurable hardware module that blocks traffic in the NoC links to isolate PEs and create the OSZs. In the OSZ method, the LCs block all incoming and outgoing traffic, meaning that packets cannot enter or leave the isolated region. LCs are AND/OR gates connected to control flow signals of the links, configured by memory-mapped registers, representing a negligible increase in area.
3) **Access Points (AP)**: configurable hardware module that controls one opening in the OSZ border to enable the communication between the application running inside the OSZ with peripherals. In this work, the AP adopts an authentication protocol [15], that uses two authentication flits in the packet to validate communication. APs are simple mechanisms controlling access to the routers' links, differently from firewalls, which are complex mechanisms used to manage security policies n normally inserted at the network interfaces.
4) **Peripheral Network Interface**: a hardware module between the NoC and a peripheral that authenticates traffic to and from peripherals. This work utilizes a Secure Network Interface for Peripherals (SNIP) [16], that controls access to and from peripherals at application granularity, only accepting requests from applications registered by the system manager.
5) **Packet arrival confirmation**: a mechanism that can acknowledge the correct packet reception by the target, being able to indicate if the packet did not reach the destination. The adopted mechanism responsible for this monitoring is the "session protocol" [17]. This protocol creates a virtual connection between the communicating tasks and sends control messages through a control NoC in parallel with data messages to monitor packet arrivals. Packets are accepted upon simultaneous reception of the corresponding control and data messages that verify the packet's source and target.
6) **Master-worker communication protocol**: all IO communications must start from the application (*master*), with the peripheral answering the requests (*worker*). Such communication protocol prevents $MalPeriph$ from flooding the manycore with unexpected packets.

These monitoring elements generate warnings and send them to the security manager. Note that the presented security features are not platform-specific. The designer can select any Secure Zone or Peripheral Network Interface proposal if they send warnings or reports related to suspicious activity.

Monitoring mechanisms 1, 2, and 5 are related to the communication inside the secure zone. Such mechanisms may fire four warnings: *missing packets* (**W1**), *unexpected data* (**W2**), *suspicious route* (**W3**), *access attempts* (**W4**).

Monitoring mechanisms 3, 4, and 6 focus on the communication with peripherals, sending warnings related to authentication: *wrong key* (**W5**), and also warnings related to issues with peripheral packets: *missing packet* (**W1**), *unexpected data* (**W2**). Besides the warning codes (W1-W5), the security manager registers additional parameters (e.g., source, timestamp, and hop count) that help to identify attacks.

A. *Attack Detection*

The presented monitoring mechanisms can detect DoS, Spoofing, and Eavesdropping attacks. Data corruption is detected when deploying an application into the system with a MAC (Message Authentication Code) attached to the tasks' binaries. SCA detection is out of the scope of the current monitoring mechanisms.

DoS attacks based on flooding cause congestion in the NoC by filling the links and buffers with invalid packets. When a flooding attack occurs, if the flow of malicious packets tries to cross an OSZ, the LCs of the OSZ border report invalid *access attempts*, sending a **W4** warning and discarding the packets. As these packets do not have a correct authentication key, if they arrive at an AP, they are also discarded, and the AP sends a *wrong key* warning (**W5**).

Another type of DoS is packet dropping, which causes routers to remove packets from the NoC so they never reach the final destination. Packet exchange between tasks is monitored in our platform using the session protocol. Whenever a packet does not arrive in the expected time, the PE emits a *missing packet* warning (**W1**). In the case of missing a peripheral communication packet (also **W1**), the master-worker protocol detects, considering that an attacker dropped either the request or response from a peripheral.

Spoofing is the falsification of identity to access system resources. Malicious tasks or peripherals can attack sensitive applications by sending packets with wrong data to disrupt the application computation. These attacks can be detected by the AP, when packets attempt to enter the OSZ with incorrect key (**W5**), or by the master-worker protocol when data arrives at the PEs without being requested (**W2** - unexpected data).

Eavesdropping attacks happen in the NoC by misrouting sensitive packets of a secure application or malicious tasks attempting to send sensitive data to malicious agents. Since the secure application computation and communication are contained inside a OSZ, any attempt to send data to elements outside the OSZ will be blocked by the LCs and needs to be reported. However, in the case of packets trying to leave the OSZ (different from the flooding that was attempting to enter), the LC emits a *suspicious route* warning (**W3**), alerting for an attempt of information leakage.

IV. Experimental Setup

Experiments use a manycore modeled at the RTL level (SystemC and VHDL). The manycore has two NoCs: one for data and one for control. The data-NoC is a packet-switching network with two physical channels, supporting XY (default) and source routing. The control-NoC is a broadcast NoC with single-flit packets and a search path mechanism, which allows path discovery for source routing.

To evaluate the effectiveness of the distributed monitoring mechanisms, we simulated the scenario depicted on Figure 1, with the following features:

- manycore size: 64 PEs (8 × 8);
- two peripherals connected on routers (2,7) and (5,7);
- MPE (system and security manager) on the top-left PE;
- application injector peripheral at (0,7), responsible for deploying applications into the manycore;
- six applications running in secure mode, mapped in 6 OSZs. Four $appSec$ communicate with IO devices (those with AP).

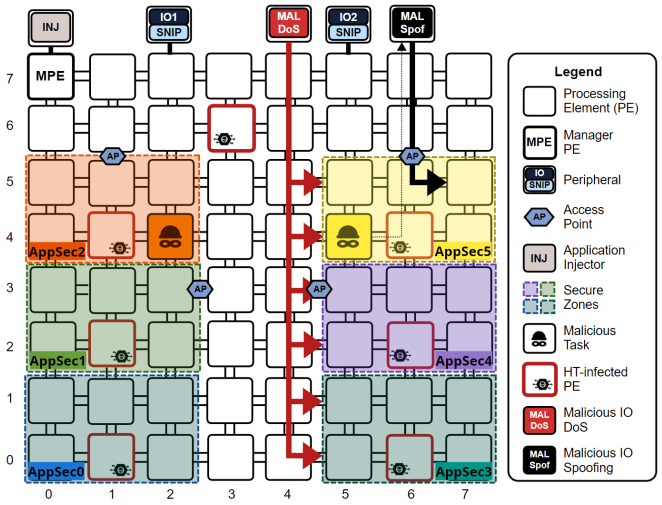

Fig. 1. Experimental setup to evaluate the security framework. The PEs are identified by their coordinates on the x-axis and y-axis as in a Cartesian plane with the origin in the bottom-left corner PE.

We performed the attack campaign to this scenario using the following threats:

HT that drop packets - one inside each OSZ: (1,0), (1,2), (1,4), (6,0), (6,2), (6,4), and one in the path for peripherals: (3,6).

MalDoS peripheral (4,7) - malicious peripheral injecting packets at a high rate to flood the NoC (one packet each 2,000 clock cycles). The packets target PEs inside the three right OSZs, selected randomly.

MalSpoof peripheral (6,7) - malicious peripheral that receives keys and tries to enter the OSZ. This peripheral receives the keys from a malicious task running on PE (5,4) and sends packets to the task on PE (7,5).

MalTask0 in PE (2,4) - malicious task that tries to send packets to PEs outside the OSZ. On each iteration of the application, *MalTask0* sends packets to PE (0,0).

MalTask1 in PE (5,4) - malicious task with a task that leaks key to *MalSpoof*. Every two application iterations, *MalTask1* sends a packet leaking the keys to the *MalSpoof* in (6,7).

V. Results

This section presents two analyses using the scenario presented in Figure 1. It starts with the warnings reported to the MPE during the attack campaign and then discusses the applications' execution time overhead.

Table I displays the number of warnings detected by the distributed monitoring infrastructure (2nd column). The 3rd column specifies the monitoring mechanism that generated each warning, while the 4th column identifies the threat responsible for each warning.

TABLE I
WARNINGS REPORTED BY THE MONITORING INFRASTRUCTURE.

Warning	Occurence	Detection	Threat
Missing Packet (W1)	95	Session Protocol (75) Master-Worker (20)	Hardware Trojans
Unexpected Data (W2)	4	Access Point (2) Master-Worker (2)	MalSpoof
Suspicious Route (W3)	19	Link Control (19)	MalTask0
Access Attempt (W4)	248	Link Control (248)	MalDoS
Wrong Key (W5)	101	Access Point (101)	MalDoS MalSpoof
TOTAL	**467**		

Missing Packet (**W1**) occurred 95 times. The HTs inside the OSZ dropped 75 packets, which were detected by the packet arrival confirmation mechanism (session protocol). Outside the OSZ, another 20 packets were dropped, detected by the master-worker protocol. Upon detecting the attack, the MPE selected a *rerouting* countermeasure to establish a new communication path for resending the lost packets.

The four Unexpected Data warnings (**W2**) were generated by an AP (2 cases) or PEs (2 cases). Four of the packets sent by the *MalSpoof* peripheral had the correct keys, with the AP blocking two, and the other two passed through it, reaching the PE that was not expecting data (master-worker protocol). Since, in these four cases, the attack packets had the correct keys, the MPE triggered a *key renewal* countermeasure.

Suspicious Route warnings (**W3**) are generated by the LCs reporting that packets are trying to leave the OSZ. These packets are sent by the malicious task on PE (2,4), *MalTask0*, being discarded by the LCs.

Access Attempt (**W4**), which signalizes DoS attacks, records 248 attempts to enter the OSZ. These were packets from the injector *MalDoS*, targeting random PEs within the three OSZs. All packets were immediately discarded.

Wrong Key (**W5**), warnings from forged packets by the *MalSpoof* injector. These packets, which attempted to access OSZs, have incorrect keys and thus failed the authentication process. As a result, the AP discarded these packets.

The attack campaign comprised DoS, spoofing, and eavesdropping. The monitoring infrastructure successfully detected all initiated attacks, generating the necessary warnings to implement countermeasures such as packet dropping, rerouting and retransmission, key renewal, and remapping of the APs. Without the security mechanisms, applications crashed shortly after the of the first attacks. However, with the detection of attacks, generation of warnings, and execution of countermeasures, all applications completed their execution successfully.

To analyze the impact of such attacks and their respective countermeasures in terms of execution time, Table II shows the execution time in terms of absolute values (2nd and 3rd columns) and the relative overhead (4th column). The baseline column corresponds to the execution without attacks. When simulated in the baseline scenario, the malicious applications have their malicious behavior deactivated.

TABLE II
EXECUTION TIME OF APPLICATIONS RUNNING IN THE OSZ(IN MS)

Apps	Baseline	Attack Campaign	Overhead
AppSec0	9.74	11.23	15.26%
AppSec1	9.94	10.28	3.36%
AppSec2	5.06	5.53	9.21%
AppSec3	9.78	11.21	14.68%
AppSec4	9.92	10.53	6.15%
AppSec5	9.93	10.05	1.15%
Scenario	**13.69**	**14.30**	**4.47%**

Note that the applications' runtime overhead varies between 1.15% and 15.26%, averaging 8.3%. The overall execution time overhead, i.e., the time required to execute all applications in parallel, was 4.47%. This overhead is due to the number of attacks to which applications are subjected and the more time-consuming countermeasures, such as rerouting-retransmission and key renewal. The impact on each application varies depending on the specific attack generated, highlighting the importance of conducting a severe attack campaign.

Results on Table II emphasize the efficiency of the proposed countermeasures, demonstrating that the cost of securing applications in terms of execution time is remarkably low.

VI. CONCLUSION

The security framework, enhanced by the monitoring mechanisms discussed in this paper, establishes a comprehensive set of rules and procedures to protect manycore systems. These mechanisms reduce vulnerabilities and aid in recovery from attacks. Immediate countermeasures such as packet discarding are activated upon attack detection, while others depend on warnings transmitted to the system manager.

The attack campaign not only demonstrated the robustness of these security methods but also exposed limitations within the platform that restricted the scope of possible attacks, such as the inability of NIs and (SNIPs) to process cropped packets (an HT may drop part of a given packet, for example).

The set of warnings the system manager receives may enable the development of system-level heuristics based on security to protect sensitive resources in manycores. These countermeasures could potentially include terminating an attack by shutting down an application that is injecting malicious packets. Furthermore, enhancements to the Session Protocol could enable the location of HTs.

REFERENCES

[1] S. Baron *et al.*, "Security mechanisms to improve the availability of a Network-on-Chip," in *IEEE International Conference on Electronics, Circuits, and Systems (ICECS)*, 2013, pp. 609–612, https://doi.org/10.1109/icecs.2013.6815488.

[2] J. M. Kumar *et al.*, "Fortified-noc: A robust approach for trojan-resilient network-on-chips to fortify multicore-based consumer electronics," *IEEE Transactions on Consumer Electronics*, vol. 68, no. 1, pp. 57–68, 2021, https://doi.org/10.1109/TCE.2021.3129155.

[3] J. Ramachandran, *Designing Security Architecture Solutions*. John Wiley & Sons, Inc., 483p, 2002, 483p.

[4] S. Charles and P. Mishra, "A Survey of Network-on-Chip Security Attacks and Countermeasures," *ACM Comput. Surv*, vol. 54, no. 5, pp. 101:1–101:36, 2022, https://doi.org/10.1145/3450964.

[5] S. Bhunia *et al.*, "Hardware Trojan attacks: Threat analysis and countermeasures," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1229–1247, 2014, https://doi.org/10.1109/JPROC.2014.2334493.

[6] L. Zhang *et al.*, "Effectiveness of HT-assisted sinkhole and blackhole denial of service attacks targeting mesh networks-on-chip," *Journal of Systems Architecture*, vol. 89, pp. 84–94, 2018, https://doi.org/10.1016/j.sysarc.2018.07.005.

[7] S. Charles *et al.*, "Real-Time Detection and Localization of Distributed DoS Attacks in NoC-Based SoCs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 12, pp. 4510–4523, 2020, https://doi.org/10.23919/date.2019.8715009.

[8] J. Yao *et al.*, "Spotlight: An Impairing Packet Transmission Attack Targeting Specific Node in NoC-based TCMP," in *IEEE European Test Symposium (ETS)*, 2023, pp. 1–4, https://doi.org/10.1109/ETS56758.2023.10174197.

[9] D. Mesquita *et al.*, "Current mask generation: a transistor level security against DPA attacks," in *SBCCI*, 2005, pp. 115–120, https://doi.org/10.1145/1081081.1081114.

[10] B. Bisht and S. Das, "BHT-NoC: Blaming Hardware Trojans in NoC Routers," *IEEE Design & Test*, vol. 39, no. 6, pp. 39–47, 2022, https://doi.org/10.1109/MDAT.2022.3202998.

[11] H. Gondal *et al.*, "A method to detect and avoid hardware trojan for network-on-chip architecture based on error correction code and junction router (ECCJR)," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 4, pp. 581–586, 2020, https://doi.org/10.14569/ijacsa.2020.0110476.

[12] V. Y. Raparti and S. Pasricha, "Lightweight Mitigation of Hardware Trojan Attacks in NoC-based Manycore Computing," in *ACM/IEEE Design Automation Conference (DAC)*, 2019, pp. 1–6, https://doi.org/10.1145/3316781.3317851.

[13] V. J. Kulkarni *et al.*, "Packet header attack by hardware trojan in NoC based TCMP and its impact analysis," in *IEEE/ACM International Symposium on Networks-on-Chip (NOCS)*, 2021, pp. 21–28, https://doi.org/10.1145/3479876.3481597.

[14] L. L. Caimi and F. Moraes, "Security in Many-Core SoCs Leveraged by Opaque Secure Zones," in *ISVLSI*, 2019, pp. 471–476.

[15] R. F. Faccenda *et al.*, "Lightweight authentication for secure IO communication in NoC-based many-cores," in *2023 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2023, pp. 1–5.

[16] G. Comarú *et al.*, "Secure Network Interface for Protecting IO Communication in Many-cores," in *Symposium on Integrated Circuits and Systems Design (SBCCI)*, 2023, pp. 1–6, https://doi.org/10.1109/SBCCI60457.2023.10261655.

[17] R. F. Faccenda *et al.*, "Detection and Countermeasures of Security Attacks and Faults on NoC-Based Many-Cores," *IEEE Access*, vol. 9, pp. 153 142–153 152, 2021, https://doi.org/10.1109/access.2021.3127468.