IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 7, 2021, accepted July 18, 2021, date of publication July 26, 2021, date of current version August 6, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3100540

A Survey on the Security of Wired, Wireless, and 3D Network-on-Chips

AMIN SARIHI[1], (Graduate Student Member, IEEE), AHMAD PATOOGHY[2], AHMED KHALID[3], MAHDI HASANZADEH[4], MOSTAFA SAID[5], AND ABDEL-HAMEED A. BADAWY[1], (Senior Member, IEEE)

[1]Klipsch School of Electrical and Computer Engineering, New Mexico State University, Las Cruces, NM 88003, USA
[2]Department of Computer Science, University of Central Arkansas, Conway, AR 72035, USA
[3]Department of Computer Engineering, Assiut University, Assiut 71515, Egypt
[4]Independent Researcher, Tehran 25529, Iran
[5]Department of Computer and Electrical Engineering and Computer Science, California State University, Bakersfield, CA 93311, USA

Corresponding author: Amin Sarihi (sarihi@nmsu.edu)

ABSTRACT Network-on-Chips (NoCs) have been widely used as a scalable communication solution in the design of multiprocessor system-on-chips (MPSoCs). NoCs enable communications between on-chip Intellectual Property (IP) cores and allow processing cores to achieve higher performance by outsourcing their communication tasks. NoC paradigm is based on the idea of resource sharing in which hardware resources, including buffers, communication links, routers, etc., are shared between all IPs of the MPSoC. In fact, the data being routed by each NoC router might not be related to the router's local core. Such a utilization-centric design approach can raise security issues in NoC-based designs, e.g., integrity and confidentiality of the data being routed in an NoC might be compromised by unauthorized accesses/modifications of intermediate routers. Many papers in the literature have discovered and addressed security holes of NoCs, aiming at improving the security of the NoC paradigm. However, to the best of our knowledge, there is no solid survey study on the security vulnerabilities and countermeasures for NoCs. This paper will review security threats and countermeasures proposed so far for wired NoCs, wireless NoCs, and 3D NoCs. The paper aims at giving the readers an insight into the attacks and weaknesses/strengths of countermeasures.

INDEX TERMS Network-on-chip, threat model, hardware security, hardware trojan, DoS attack.

I. INTRODUCTION

Ever increasing VLSI technology shrinkage has enabled Multi-Processor System-on-Chips (MPSoCs) to accommodate tens of Intellectual Properties (IPs), e.g., processing cores, memory modules, and various I/O components. This technology shift urges the necessity for an efficient communication architecture to enable fast, yet energy-efficient data exchange across the chip. Network-on-Chips (NoCs) were first introduced in 2004 [1] as a scalable communication architecture and later widely used in the design and fabrication of many chips (e.g., Tilera TILE64 [2] and Kalaray's MPAA-256 [3]). As most of the modern MPSoCs use an on-chip network as their backbone communication architecture, the industry has already started offering NoC IPs (for instance, FlexNoC IP from Arteris company [4]) to facilitate the design process.

The associate editor coordinating the review of this manuscript and approving it for publication was Yanjiao Chen.

The main idea of on-chip networks is to share resources to boost resource utilization, i.e., a number of on-chip components are interconnected via a shared network that is governed by a set of structural, routing, switching, and flow-control rules. While attaining a satisfactory bandwidth, the structured architecture of NoCs helps to avoid long communicating wires (also known as long interconnects) that are significant contributors to dynamic power consumption [5] as well as reliability issues [6]. NoCs offer high resource utilization, design modularity, and support for parallel communications with moderate performance/energy efficiency [7]. NoCs' moderate efficiency has its roots in i) having communications between far cores in which messages have to be forwarded over a long chain of adjacent routers, and ii) one-to-many message broadcast situations that have to be handled sequentially. To address these shortcomings, researchers have introduced the idea of adding wireless communications through wireless routers. In a WNoC (Wireless Network-on-Chip), far-distant messages and one-to-many

messages are broadcasted in a one-hop fashion through wireless links/transceivers, boosting the overall performance and energy efficiency [8].

While performance aspects of NoC have been addressed over the years in different proposals/papers, security, nevertheless, has remained a serious challenge for designers. The idea of an on-chip network that easily accesses/forwards messages of different IP cores chip-wide may complicate the security issues, i.e., the benefit gained by resource sharing in NoC fabrics might be counter-productive when data integrity and confidentiality are essential.

Utilization of 3rd-party IPs (3PIPs) to avoid the exorbitant cost and design time of MPSoCs is common among SoC designers [9]. The IPs could range from processing cores and DSP units [10] to even the NoC itself [4]. NoC IPs are widely used in different devices such as tablets, mobile phones, and autonomous vehicles as a part of the MPSoC. 80% of the top five Chinese fabless companies are using the 3rd-party Arteris FlexNoC interconnect [11]. In fact, MPSoC designers prefer to use the 3PIP (third-party IP) NoC due to the lower time-to-market and production cost. However, the utilization of 3PIPs products in the design of MPSoCs, may ultimately introduce new security vulnerabilities, e.g., security holes or threats. Some of the 3PIPs could be infected by some type of Hardware Trojan (HT) [11]. This includes IPs designed in-house using a trusted design team and trusted CAD tools such as Synopsys and Cadence. They might be infected by a foundry during the post-design stage [12]. HT-infected IPs use specific trigger conditions, making them substantially harder to detect [13], so they mostly bypass verification and manufacturing testing procedures. Upon activation, HTs could take a severe toll on the functionality of the chip. A successful attack could lead to irrecoverable economic and social losses that might not be easily compensated [14].

Generally, a malicious activity targets at least one of the major security requirements including *confidentiality*, *integrity*, and *availability* which are called the CIA triad [15] and are defined as follows:

- **Confidentiality**: sensitive information should be only available to authorized agents.
- **Integrity**: unauthorized agents are not allowed to modify the contents of a message.
- **Availability**: network resources remain available during its operation.

Attackers try to undermine these three security aspects, and SoC designers must obtain and keep the security goals to safeguard the system. While NoC security has been surveyed briefly in some previous works [16]–[19], to the best of our knowledge, there is no solid work that surveys the recent security solutions for various NoC technologies. This paper covers both the different attack models and the proposed countermeasures for wired, wireless, and 3D NoCs.

The rest of this paper is organized as follows. In section II, we explain the architecture of wired, wireless, and 3D NoC. In section III, the attacks and the related taxonomy in the literature are discussed. Proposed countermeasures in wired, wireless, and 3D NoC is discussed in details in sections IV, V, VI, respectively based on the threat model. Lastly, In section VII, a guideline for future research in NoC security is provided.

II. PRELIMINARIES

NoCs were introduced to implement the idea of separating communications and computations inside modern multi-core chips. Every communicating agent on the chip is equipped with the required tools to interact with the on-chip network. The network stack (shown in Figure 1) is widely used in the design of on-chip networks. According to this figure, layers of the network stack are accommodated using some layers in the hardware. The application layer provides communications between applications running on different processing/DSP cores and/or applications accessing memory/IO module. The application layer is, in fact, a communication layer between an application running on a processing/DSP core and another processing core or a memory/IO module. The transport layer is the SoC firmware that offers system-level services to send/receive messages between IPs/modules. The data link layer, which makes data chunks ready for transmission, is hosted on the hardware component known as the network interface (NI). The NI is the gateway that connects every local IP to the global (chip-wide) network. Finally, the network and physical layers of the stack are implemented as the on-chip routers and channels. Routers are responsible for storing, routing, and forwarding data units over the channels. They are interconnected using channels following a predefined topology structure such as 2D/3D mesh, butterfly, fat-tree, etc.

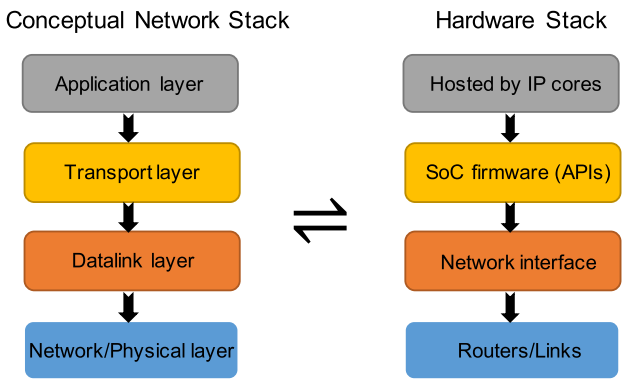


FIGURE 1. The network stack model.

Figure 2 shows the architecture of a typical router used in NoCs, known as a virtual channel (VC) router. It consists of the following components:

- VC buffers: store ingress/egress data units (so-called flits) for each port.
- Ports or channels: pass data between adjacent routers.
- Routing computation unit: computes the appropriate outgoing port for packets based on the routing policy used and the network congestion situation.

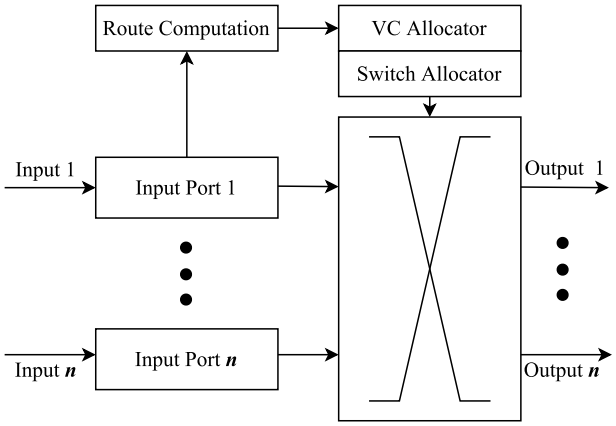


FIGURE 2. Architecture of a typical VC-enabled router for NoCs.

- VC allocator: finds an available VC buffer for the outgoing packets.
- Switch allocator: responsible for the arbitration of the input/output ports of the crossbar switch.

Routers/channels of on-chip networks are treated as shared resources to achieve better utilization, i.e., every router is responsible for storing, routing, and even forwarding packets that might not be issued by the router's local core. In fact, several on-chip routers may contribute to forwarding a packet that might not be directly related to them. The benefit gained by resource sharing in NoCs may be counter-productive when data integrity and confidentiality are considered. This raises the chances for malicious routers to access, sniff or modify packets of data. This is especially true if the NoC is integrated as a 3PIP in a bigger design [11]. While cores and NIs deal with local data only, routers and channels are in charge of forwarding local and global data.

Looking at the components of an NoC-based MPSoC, vulnerability points of such chips can be listed as follows:

- Processing cores/IPs are bought from various IP providers and can contain HTs injected by the designer/manufacturer to either implement or facilitate malicious activities. They can also be infected by software malware [20], [21].
- One or more of the working NIs in the network might be infected by an HT to interrupt expected services of a clean NI.
- One or more routers or links of the network might be infected by HTs to conduct malicious activities.

In Section III of the paper, we discuss how each of the infections mentioned above may result in a security threat. There might be additional components in the NI or the router logic if the NoC is a more advanced one, e.g., wireless NoC, 3D NoC, or a custom NoC architecture. Considering two major classes of advanced NoCs, we describe other components that can be found in wireless NoCs and 3D NoCs and their corresponding security threats in the next subsections.

A. 3D NETWORK-ON-CHIPS (3D NoCs)

The implementation of NoCs in 3D stacked ICs creates what is called 3D NoCs. This newborn communication architecture allows for significant performance, area, and power improvements over traditional 2D NoCs [22]. One major architectural difference observed in 3D NoCs is the use of vertical channels to realize inter-layer communications. Vertical channels add two more ports to the router and crossbar switch to communicate with upper and lower layers. Due to the different fabrication processes of vertical connections, mainly based on Through Silicon Vias (TSVs), the following changes compared to 2D NoCs are worth mentioning. 1) TSV channels introduce several advantages, including lower signal delay, smaller chip form factor, and higher integration density. 2) TSVs help 3D NoCs mitigate inter-layer traffic load and use fewer signal drivers and repeaters [23]. 3) Due to their high fabrication costs, TSV channels are implemented only in a handful of routers in some designs. This makes TSV channels security/reliability hot-spots in the 3D NoCs [24].

B. WIRELESS NETWORK-ON-CHIPS (WiNoCs)

Packets in wireless NoCs (WiNoCs) can pass through two paths: one is the conventional wired NIs, routers, and channels; the other is the shared wireless medium by using wireless interfaces and wireless signal transceivers of the NI. The wireless hub is basically a conventional NoC router with an extra port connected to the wireless interface. Depending on the application needs, some researchers assume all routers are connected to wireless interfaces [8] and many others assume that the network is divided into clusters where each cluster is connected to a single wireless hub to minimize the design cost [25], [26], Figure 3 shows an example of such a clustered network. The most significant advantage of WiNoCs lies in its low delay in broadcast and multicast transmissions. Thanks to the millimeter-wave omnidirectional antennas that are usually adopted in WiNoCs [27], applications such as cache coherency protocols are performed tremendously fast

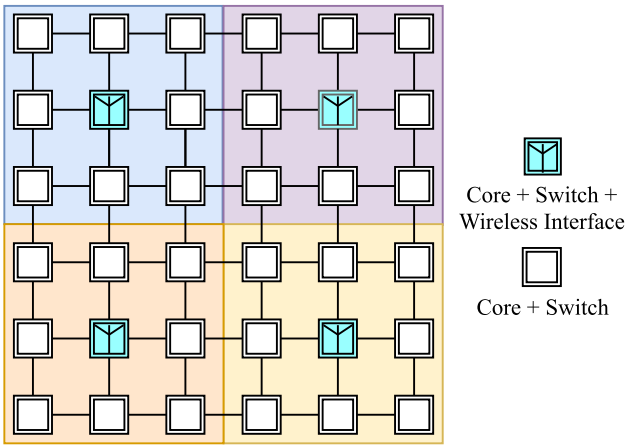


FIGURE 3. A simple example of a WiNoC with four clusters.

using WiNoCs. For instance, WiNoCs are able to transfer a 64B cache line anywhere within 5–15 cycles [28].

III. TAXONOMY AND ATTACKS

Figure 4 represents a high-level taxonomy of the attacks on MPSoCs. The so-called *physical attacks* require physical access to the chip to conduct the intended malicious activity, i.e., read some internal/external signals of the chip, access NoC channels, or monitor the chip's power profile. Physical attacks can either be: 1) invasive, in which the packaging of the victim chip is dissected by the attacker for more detailed analysis, e.g., probing attacks; however, the chip must remain functional after decapsulation, or 2) non-invasive, which do not impose physical modifications to the chip. Non-invasive physical attacks are relatively cheaper and easier to implement, e.g., power side-channel attacks.

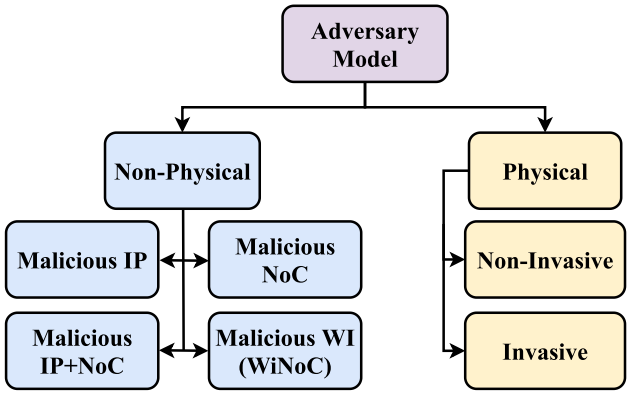


FIGURE 4. High-level taxonomy of the attacks on MPSoCs.

The need for physical access limits the applicability of physical attacks. In contrast, non-physical attacks do not rely on having physical access to the chip that eases their application in MPSoCs. In the rest of this section, we assess and classify non-physical security attacks. In Section III-A, attacks that are common for traditional 2D NoCs, 3D NoCs, and WiNoCs are discussed. In sections III-B and III-C, attacks that are specifically applicable on WiNoCs, and 3D NoCs are discussed, respectively.

In MPSoCs, *Malware Injections* and *Hardware Trojans* are the two major sources of non-physical attacks by introducing malicious IP, malicious NoC, or a combination of both. IPs infected by malware contribute to 80% of the total attacks on embedded systems [29]. Malware infections mostly happen at the device firmware/software update/patching process [30] where the software accesses bare-metal hardware.

HTs are tiny circuits that start their malicious activity after being triggered. The triggering part of HTs seeks for very rare conditions (mostly a set of signals acquiring their rare values). Subsequently, most traditional logic testing methods mostly fail in detecting HTs [31]. As HTs show negligible power and area footprints, side-channel analysis methods would also exhibit deficiencies in detecting them. Many researchers have addressed HT-induced malfunctions in NoC-based MPSoCs. HTs can be inserted by different agents at various stages of the design manufacturing process. Some insertion scenarios are as follows:

- HTs can be inserted through gate-level manipulation of the NoC netlist by an adversary designer [32], [33].
- EDA tools can also target companies' products to insert HTs for defamation purposes.
- Designs layout might be modified at the fabrication stage [34].
- 3PIPs used to expedite MPSoC design might be pre-infected by HTs [33].

Researchers have proposed various HT circuits for processing/NoC IPs. HTs proposed for processing IPs are beyond the scope of this paper, we only review NoC HTs in the rest of this section.

A malicious NoC comprises situations of having at least a malicious Network Interface (NI) or a malicious router in the NoC fabric. Either the NoC vendor or the fabrication factory can insert HT circuitry into the clean NoC design to infect the NoC. HT insertions target i) the network interface to alter packetizing/de-packetizing or flow control of data, or ii) the router logic to negatively impact route computation which leads to packet misroute/loss/duplication, or to inject low-priority packets to discover the timing information of high-priority packets [35].

Ancajas *et al.* [11] have explored the consequences of inserting an HT in a cloud MPSoC system. The proposed HT initiates a duplication attack once inserted in the router. The area and power overheads of the proposed HT are 4.62% and 0.28%, respectively. The HT proposed in [14] targets allocator modules of NoC routers by de-prioritizing arbiters. It prolongs the crossbar traversal delay by denying fair crossbar allocation to the packets destined to or originated from a victim node. The HT's overhead is nearly 4% compared to the baseline router. Another study by Daoud and Rafla [36] has proposed an HT to be inserted within the NoC routers. This HT is capable of misrouting packets and eventually causing DoS while having less than 1% area overhead compared to the baseline router. Daoud and Rafla [37] have introduced a DoS attack using a *black hole router* that drops packets passing through it with less than 2% power and area overhead. Raparti and Pasricha [38] have implemented an HT in the NI that facilitates information stealing. The HT works by manipulating the FIFO header pointer in the NI FIFO queue that eventually leads to duplication attacks. The HT yields a 1.3% overhead compared to the baseline NI. Table 1 summarizes the HTs area and power costs in the previous work.

Malicious IP+NoC is, of course, a more serious threat as it is capable of carrying out a wider spectrum of threats; however, it is harder to achieve by adversaries. As the SoC designers order IPs from different IP providers, the activation chances of malicious NoC and IPs at the same time to conduct a collaborative attack are slim.

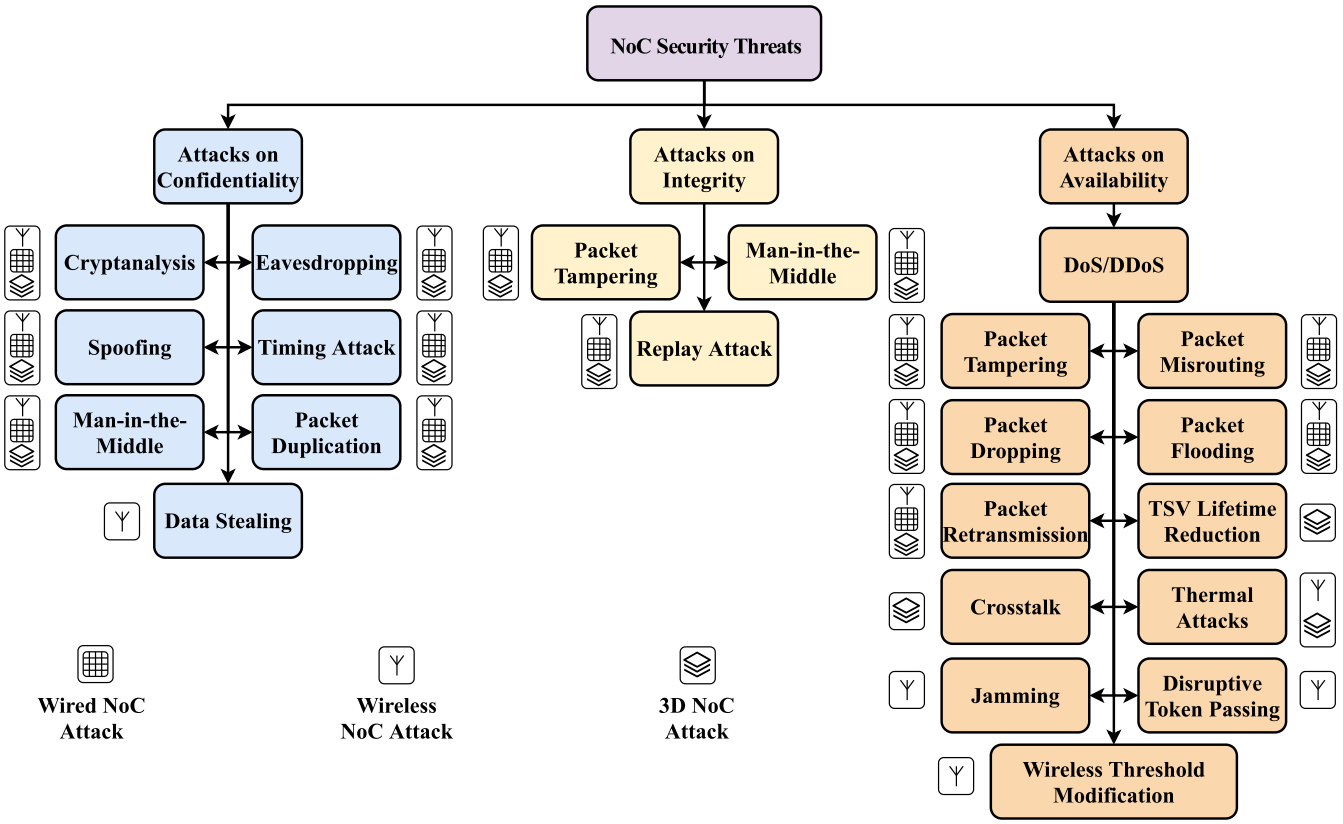


FIGURE 5. Classification of the addressed attacks based on the attacker's target security goal.

TABLE 1. HT implementations costs.

Reference	Area	Power	WRT*
Ancajas et al. [11]	4.62%	0.28%	Router
Rajesh et al. [14]	4.32%	0.014%	Router
Daoud and Rafla [36]	0.2%	–	Router
Daoud and Rafla [37]	1.98%	0.74%	Router
Raparti and Pasricha [38]	1.3%	–	NI

*WRT $\triangleq$ With Respect To.

A. COMMON ATTACKS ON 2D, 3D, AND WIRELESS NoCs

Figure 5 shows our proposed attack classification based on the attacker's target security goal. In this figure, it is considered that an attacker aims to undermine at least one of the three security bases, i.e., *confidentiality*, *integrity* and *availability*. Following is an explanation of the bases and the attacks proposed for each one in the context of NoC-enabled MPSoCs.

Confidentiality, as one of the major factors of security, denotes to protect assets of a system from any unauthorized access. In the NoC context, this is equivalent to guaranteeing that the messages/packets traversing the network are kept private between the sender and receiver nodes. For this aim, packet encryption is considered as one of the solutions to ensure data confidentiality in NoCs. Based on assessments we have done in our studies, attacks that alter data confidentiality In the NoC context can be summarized as follows:

- **Eavesdropping:** data communication between a source and a destination node is being sniffed by an unauthorized adversary, e.g., a malicious router, or a router along with an IP. The data may contain sensitive information such as passwords or encryption keys [32].
- **Differential Cryptanalysis:** analogous to differential power analysis attacks, the attacker tries to infer secret information by analyzing the transmitted data over a channel and guessing the encryption key [11]. In most cases, the attack needs collaboration between a malicious router and a malicious IP.
- **Timing Attack:** intentional collisions between the attacker's data and others' sensitive data on a specific path are made to release valuable information regarding the timing and the volume of the sensitive information to the attacker [39]. The key idea is that the attacker measures the timing delay of their own data that is proportional to the existence or non-existence of the sensitive information on the same path.
- **Spoofing:** gaining unauthorized access to data by using a counterfeit identity [39]. The accessed data could be a portion of the shared memory that is, by default, not accessible [40].
- **Man-in-the-middle (MITM)**: an unapproved node meddles in the communication between a pair of source and destination nodes to monitor the passed data.

Figure 6 shows the threat model used in the MITM attacks. Node A will send a communication request to Node B and node B replies. The malicious node M intercepts the connection between A and B through impersonating as the other party to A and B, respectively. It serves both nodes by fraudulently tampering with the connection between them to infer secret information. The MITM node has to keep the connection flowing between A and B and for the continuity of the attack. MITM attack can alter confidentiality and/or integrity of NoCs.

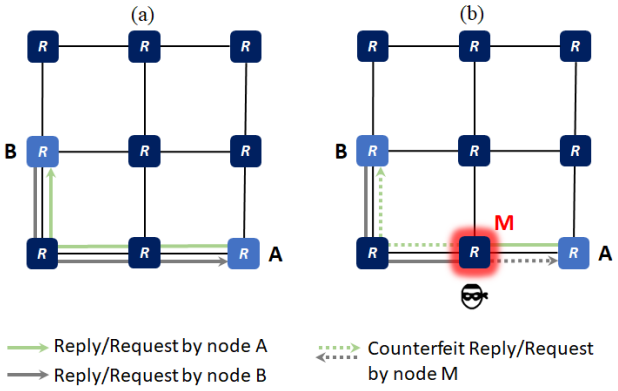


FIGURE 6. The threat model used in MITM attacks.

- **Duplication:** A malicious router duplicates incoming packets and forwards them to a new destination for further processing, e.g., cryptanalysis or timing analysis. Due to the fact that the router still forwards the packets to the intended destinations, detecting this activity as a part of a multi-agent attack might not be easy.

Integrity, as another major security factor, is also subject to various attacks by adversaries. In the security context, integrity refers to preventing any unauthorized agent from modifying security assets. In the NoC context, this translates to preventing unauthorized changes in any field of packets, including data or control fields. We review attacks on integrity in the following:

- **Tampering:** the content of a packet can be modified by a malicious node on different levels such as source corruption, payload corruption, and tag corruption [21]. This can lead to various network-level problems, including packet misrouting [41], packet dropping, and packet retransmission.
- **Replay Attack:** an intercepted packet is replayed by an adversary to the same destination to ask for unauthorized resources. As the original and replayed packets are the same, the receiver treats them as two valid packets [42].

The last type of attacks in our classification targets the *Availability* of an NoC fabric. Traditionally, availability is defined as having assets accessible to legitimate agents at any time. Assets may include routers, buffers, channels, and other resources in the network. The goal of such attacks is to ultimately stop the chip from working by rendering some of its resources unavailable. The attacks classified in this category are as follows:

- **Denial of Service (DoS):** an adversary disrupts the system's functionality by malicious activities such as flooding the network with junk packets or misrouting packets to artificially-made congested paths [39]. As the latency degradation (network-level impacts of the attack) translates to application performance degradation, DoS attacks can alter the functionality of real-time MPSoCs without even completely stopping them [14].
- **Distributed Denial of Service (DDoS):** if a DoS attack is collaboratively carried out by multiple adversaries on a common victim [43], it is referred to as a distributed DoS attack.
- **Retransmission Attack:** when a tampered packet fails the error checking mechanism (done either hop-by-hop or only at the destination node), a retransmission request is sent to the source node. Repeating this process over and over will eventually lead to denial of service [44].
- **Packet Dropping:** when some packets of the network are discarded. This can be done on either random packets or packets carrying certain information in the header or body [45].
- **Packet Misrouting:** attackers misroute packets of the network with the aim of creating livelocks or deadlocks that may halt the operation of the whole chip [36].
- **Packet Flooding:** attackers flood the system with arbitrary data to delay or drop the legitimate flow of data [46].

B. ATTACKS ON WIRELESS NoCs

The following attacks exploit features/specifications of wireless connectivity to launch attacks on wireless NoCs.

- **Jamming-based DoS:** transmitting junk packets on the same frequency band that legitimate wireless routers use, causes collisions and prevents the legitimate packets from being received properly. If the junk packets are sent using higher transmission power than the legitimate ones, even more distortion will happen.
- **Eavesdropping:** taking advantage of the broadcasting nature of the wireless medium, a malicious eavesdropper can passively listen to all packets being transmitted over the wireless medium. In some cases, an external attacker (a device located outside the chip packaging) with an antenna tuned to the working frequency of the chip can listen to the chip's communications without any of the internal nodes ever knowing.
- **Packet Tampering:** If a malicious wireless interface (WI) is malicious, it can tamper with the content of the messages sent through it. This may cause the entire chip to malfunction or experience degraded performance.
- **Broadcast Data Stealing:** a smart packet tampering attack launched by converting unicast packets (destined to a specific receiver) to broadcast ones. When broadcast

packets are sent over the wireless medium, this allows external eavesdroppers to steal the packets.

- **Threshold Modification Attack:** the adversary tampers with the message-size threshold in the router to have fewer/more messages sent wirelessly. If the adversary sets the threshold to zero, all messages will be directed to the wireless interfaces leading to QoS and over-utilization issues. In contrast, setting the threshold to a high value leads to under-utilization of the wireless interfaces.
- **Token Tampering:** attackers tamper with the communication time limit (known as token) in contention-free wireless NoCs. By setting the token time to i) zero: no node will ever transmit over the wireless channel, or ii) maximum: all nodes start transmitting at the same time without any actual receivers or vice versa. Such over-utilization of the wireless hubs leads to a massive power drain.
- **Thermal Attack:** as a side effect of over-utilization of resources, attackers can overheat the chip that eventually leads to performance degradation and/or data loss. To tackle the heat issues, the operating system may apply thermal throttling by reducing the operating frequency, thus degrading the performance.

C. ATTACKS ON 3D NoCs

Adversaries can attack 3D NoCs in some unique ways targeting the stacked architecture of the chips. We review these attacks next.

- **TSV Crosstalk Attack:** crosstalk happens due to electrical coupling between wires of a TSV, and can be used to invalidate packets while passing through the TSVs. The attacker will inject some bait packets to pre-charge the TSV wires in a way that maximizes the tampered bits in the packet. Tampering happens by delaying or accelerating signal transitions on the victim wires of the TSV. It has been shown in [47] that this type of tampering may have a wide range of consequences, including packet misrouting that can lead to a global deadlock over the network.
- **TSV Lifetime Degradation:** TSVs degrade over time depending on the intensity of the workload passing through them. An attacker can shorten the lifespan of the chip by over-utilizing a specific TSV. For this attack, extra packets are being intentionally forwarded toward a specific vertical link. This attack can have drastic impacts on the chip's lifetime [48].
- **Thermal Attack:** having multiple dies stacked vertically results in trapping heat between them due to the relatively long distance to the heat sink. This creates opportunities to leverage heat to either stealthily trigger an HT or generate excess heat. Excess heat can degrade the performance due to thermal throttling or even shorten the lifespan of the chip.

IV. WIRED NoCs COUNTERMEASURES

A key factor of an NoC-security countermeasure is the unit in which the countermeasure is implemented. If the NIs are assumed trusted (in-house design), the countermeasure(s) can be integrated within the NI units. Many researchers have integrated their security solutions in the NI units, e.g., bulky modules for symmetric and asymmetric cryptography [40], [49], [50]. Also, most countermeasures that guarantee secure memory access are implemented in NIs. These countermeasures can profoundly enhance data confidentiality, while they might not be able to address DoS attacks such as packet misrouting, packet dropping, and packet tampering.

Routers are literally the first line of defense, so researchers have also used routers to integrate their security solutions [32], [35], [36], [51], [52]. Suppose a packet injected by a malicious task is heading towards a secured zone of the network to carry out an attack, e.g., timing attack, spoofing attack, etc. Security-enhanced routers can easily prevent such attacks. However, built-in countermeasures (mostly DoS protection countermeasures) of 3PIP routers are susceptible to reverse-engineering, so the functionality and security services of these routers might not be highly reliable. NoC designers must consider these trade-offs in the design stage to make optimal decisions based on the characteristics of the target applications of the MPSoC.

A. DATA CONFIDENTIALITY COUNTERMEASURES

Figure 7 depicts a situation where a malicious router makes copies of sensitive packets with high-security requirements and redirects them to an unauthorized IP to infer secret information. The details of this attack are explained in section III-A. Data encryption and data scrambling methods can significantly prevent these threats, and they are widely used to achieve confidentiality in NoC-enabled MPSoCs. By obfuscating ciphertext relation with plaintext and key (also known as confusion and diffusion), adversaries cannot extract secret information from the ciphertext.

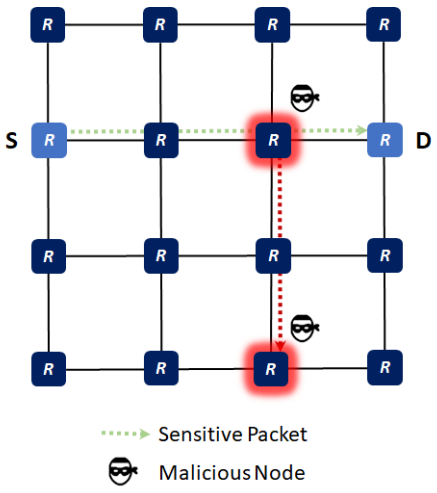


FIGURE 7. The threat model used in packet duplication attacks.

Researchers have used various types of encryption methods to address security threats covered in section III-A. In the rest of this subsection, we first briefly review various types of encryption methods as a preliminary and then review the literature.

Symmetric-key encryption algorithms use the same cryptographic keys for both encryption and decryption processes [53], [54]. Consequently, they would need to agree on keys ahead of time, i.e., a process known as key exchange. The key is exchanged either by sending unencrypted data over a secure channel or through key exchange mechanisms, such as Diffie-Hellman, in an insecure environment. In addition to the key exchange, symmetric-key encryption method can either work on i) data blocks obtained by dividing the plaintext into fixed-size blocks and encrypting each block of data (so-called Block ciphers), or ii) data streams obtained by dividing the plaintext into single bits, and each bit is encrypted individually (known as stream ciphers). The asymmetric-key (public-key) encryption algorithms, on the other hand, use different keys for encryption and decryption processes. The encryption key (public key) is visible to everyone; however, each user's decryption key (private key) is kept private [55], [56]. In general, the symmetric-key encryption methods (and in particular block ciphers) have been used more frequently in the NoC security context. This is due to the lower hardware requirements of these methods (less memory, less logic, etc) as well as being more predictable in terms of establishing a secure connection.

Assuming malicious NoC+IP model, Sepúlveda *et al.* [21] proposed security countermeasures integrated into the NI. Authors argue that the NI is built in-house and, therefore, can be trusted. The so-called tunnel-based NI encrypts all portions of the packet except the destination address. The encryption used is straightforward, i.e., XOR of the packet payload with a random key (AES in counter mode). A hash function is also applied to the packet for authentication and tamper detection. However, the AES encryption leads to a relatively large area overhead when compared to the baseline MPSoC.

Oliveira *et al.* [49] aimed at measuring the latency overhead of AES encryption when used between the NI and the router. The proposed architecture consists of a manager element that generates unique random keys for each communication session. A firewall is placed between the IP and NI to monitor and manage the IP's incoming/outgoing traffic. Encryption keys are sent through the firewall that decides whether the data must be encrypted or decrypted. The ASIC implementation shows a 193.7% increase in area, while hardware simulation shows a 395.92% surge in worst-case latency, making this architecture an unrealistic solution. The paper has failed to provide a key exchange algorithm that remains one of the most critical problems in the security of today's MPSoCs.

Charles *et al.* [57] proposed a key exchange mechanism along with an anonymous routing approach that hides the source/destination of a packet while traversing the network. The anonymous routing has two phases: route discovery and data transfer. In the route discovery phase, a packet is broadcasted from the source node that contains three security fields: i) the source's one-time public key in plaintext, ii) the encrypted version of the source's one-time public key, and a random number using the destination's public key, iii) the temporary public key of the sender node. Every node that receives this packet tries to decrypt the encrypted portion of the message and compare it with the first part; if it succeeds, it is the intended receiver; otherwise, the node only forwards that packet. Once the packet gets to the intended destination, the first and second fields will match, and the packet will not be forwarded anymore. At this point, the receiver generates a symmetric key (using a random number embedded in the encrypted part of the packet) and returns an encrypted packet that contains a nonce for establishing an anonymous path and a key for symmetric-key encryption. In this method, each node only knows its previous and next neighbor and is unaware of the final source and destination. Although the method is claimed to be highly secure, an attacker can break the route discovery phase by tampering with the route-confirmation packets. Also, the method imposes significant area and power overheads as it requires hop by hop decryption during the key exchange phase. Finally, this method needs to have keys pre-loaded in the routers.

Sant'Ana *et al.* [58] have noted the shortcomings of secure zones, encryption, and firewalls. They argue that these remedies limit MPSoC utilization and incur a substantial hardware cost. They have proposed two encryption schemes (AES and SIMON [59]) to be embedded into the network with stark area overhead differences. The high-security achievement by AES comes with significant power and area overhead. Simon, on the other hand, offers security on constrained devices. Its area is almost one-fifth of AES's area, and its power overhead is 25 times less; however, it is nearly seven times slower than AES. There is no information provided regarding what percentage of the packets were encrypted.

To reduce the cost of AES and still add high level of security, the authors of [50] adopted the Hummingbird-2 [60] block cipher scheme. Hummingbird-2 is a lightweight encryption which is mostly implemented in RFID tags. The authors assumed malicious NoC+IP adversary model, where sniffed packets could be sent to a malicious IP for further analysis. They used incremental cryptography, which outperforms other encryption algorithms, to guarantee secure communications between IPs. As depicted in Figure 8, the encryption module is placed in the NI. The proposed incremental encryption is claimed to be suitable for specific data types such as images where chunks of data are fetched from consecutive memory locations. In this method, packets take advantage of the previous encrypted memory requests and the corresponding decrypted memory responses to partially reduce the required encryption/decryption computations. Indeed, new packets are encrypted with fewer computations since the whole encryption process is no longer needed. Although the authors have claimed 80% similarity in packets

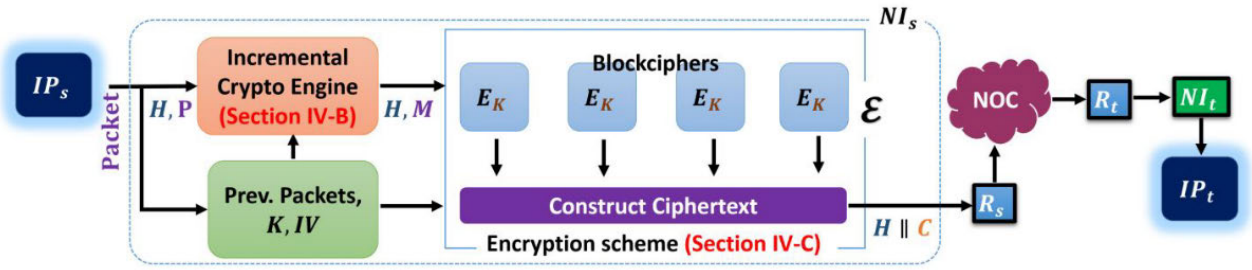


FIGURE 8. The incremental encryption module is placed within the NI [50]. The proposed architecture can encrypt four blocks of data simultaneously.

of specific applications such as image-processing, they did not address the actual chance of similar data block positions as well as the same cryptographic nonces (per requirements of the proposed incremental encryption). Moreover, packet headers should be kept as plaintext to enable routing, which may result in data-stealing attacks. The key exchange scheme between two IPs is not addressed in the paper, too.

Unlike the former mentioned works in this section, [11] has not addressed confidentiality with standard encryption methods. Instead, the authors have developed/used their own lightweight data scrambling methods. The authors have proposed a security mechanism to attain data confidentiality and thwart packet duplication. Data is scrambled by the SoC firmware at the first layer using *XOR* cipher encryption (one-time pad) to lower the chance of HT activation. While effective in stopping functional HTs, the security depends on the key distribution and generation mechanisms that were not explained. To cope with other types of HTs like always-on and internally triggered HTs, authors also used encryption in other layers to guarantee NoC confidentiality. However, the authors have not studied the impacts of the used encryption methods on the activity of the SoC signals. As the encryption keys should be generated randomly, the *XOR* operation used in the encryption algorithm can increase the signal activity[1] when the ciphertext is being routed over the network. This acts against the paper's goal of not triggering possible HTs of the chip.

In [38] Raparti and Pasricha have addressed snooping attacks (duplication) in their work where a malicious IP and an HT in the NI can cooperate to steal information. The paper aims at detecting HTs during runtime and pinpointing the software task that initiates the snooping attack. The authors implemented their own version of an HT in the NI's FIFO queue. A lightweight HT mitigation mechanism is also implemented in the NI to ensure that the flits are not reproduced with different destination IDs. Also, an analog-based HT detection mechanism is utilized. It is based on the observation that the ratio of incoming/outgoing messages in a trusted node is less than 1. The detection mechanism is not fast-acting and may need up to two days to alert the system about the attack.

[1]Signal activity denotes the rate of having signal transitions ($0 \Leftrightarrow 1$) on a net of a digital circuit.

Moreover, the proposed countermeasure is unable to detect HTs that copy data in the router instead of the NI.

Table 2 summarizes confidentiality countermeasures and compares them in terms of the encryption method, area/power overheads, and performance impact.

B. TIMING-ATTACK COUNTERMEASURES

Figure 9 illustrates an example of the timing attack scenario. Node S sends a sensitive packet to node D. The attacker node injects its packets with the same destination to monitor the sensitive packets from S to D. Consequently, both packet types follow the same route. Since both the attacker's data and sensitive data request the same output ports in the routing path, arbiters decide which dataflow to grant first. Degradation of the attacker's throughput stems from the injection of the sensitive traffic, thereby leaking information about the attacker's sensitive traffic.

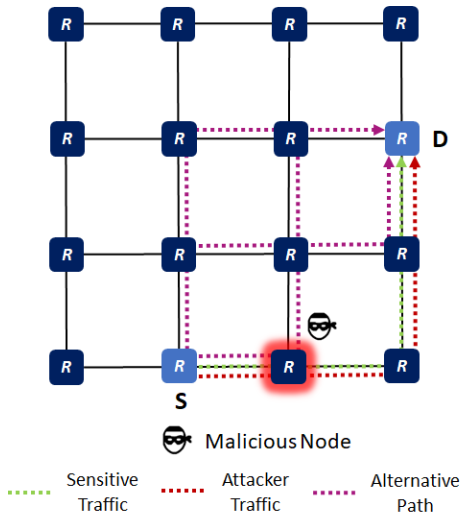


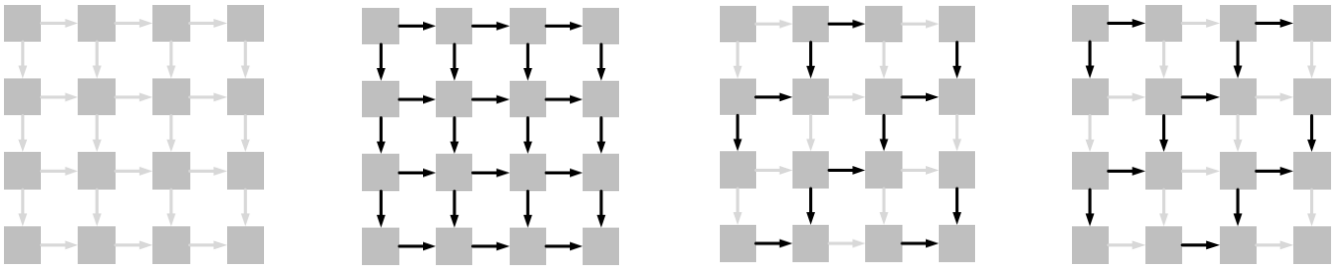
FIGURE 9. The threat model used in timing attacks.

The implementation of timing attacks requires ample information about NoC topology, sensitive and non-sensitive information/tasks mapping, and the used routing algorithm. The impact of timing attacks can be significant, e.g., in [64], a malicious task observes AES encryption sensitive traffic and recovers 12 out of 16 bytes of the 128-bit key. This makes

TABLE 2. Summary of countermeasures addressing confidentiality of NoCs.

Reference	Adversary	Encryption Method	Area / Power Overhead	WRT	Performance Impact
Sepúlveda et al. [21]	IP+NoC	AES	3.4% / 2.6%	MPSoC	2.1% gain
Oliveira et al. [49]	IP+NoC	AES	193.7% / *N/A*	Router	395.9% loss
Charles et al. [57]	IP+NoC	Authenticated Encryption [61] (AES + Galois hash [62])	*N/A*	*N/A*	4% loss
Sant'Ana et al. [58]	IP+NoC	AES SIMON	177.7% / *N/A* 37.7% / *N/A*	Router Router	2.5% loss * 46.5% loss **
Charles and Mishra [50]	IP+NoC	Hummingbird-2	2% / *N/A*	NoC	30% gain *W.R.T* traditional encryption
Ancajas et al. [11]	IP+NoC	One-time pad	9.5%, / 5.1%	SoC OCP Interface [63]	3.8% loss
Raparti and Pasricha [38]	IP+NoC (NI)	*N/A*	5.5% / 2.1%	NI	48.4% gain *W.R.T* HT-infected NI

*For real traffic
**For synthetic traffic



(a) TDM odd cycles (b) TDM even cycles (c) SurfNoC odd cyles (d) SurfNoC even cycles

FIGURE 10. Time-division multiplexing scheduling for 16 nodes in a 2D mesh vs. the proposed SurfNoC architecture [65].

cryptanalysis attacks easier by hugely reducing the keyspace. To address timing attacks, a variety of approaches such as static (e.g. allocating links in time using time-division multiplexing and dynamic (e.g. task migration) resource allocation have been proposed. These works are reviewed in the rest of this subsection.

The idea of resource allocation to counter timing attacks started in [51] by Wang and Suh as a static scheme. The authors employ temporal network partitioning to thwart timing attacks. In this paper, a set of applications with a specific security requirement are called a domain, e.g., high-security and low-security domains. The goal of the countermeasure is to decouple the high-security domain from the low-security domain. This goal can be achieved by prioritizing low-security domains over high-security domains. The twofold countermeasure consists of a static allocation of input VCs to each domain, and a priority-based arbitration for router resources, e.g, the router crossbar. To prevent DoS attacks caused by the low-priority domain's traffic, the network enforces a static bandwidth limit on the low-security traffic. While the proposed security countermeasure offers negligible performance impact under a specific traffic pattern, imposing a static threshold will contribute to performance degradation on highly-sensitive domain traffic.

Wassel *et al.* [65] borrowed the static domain concept from the previous study in [51]. The authors have explored two approaches to achieve domain non-interference as depicted in Figure 10. Time-division multiplexing (TDM) is illustrated in Figures 10.a and 10.b where packets in each black and grey domains must wait for their turn to advance in the network. The whole network is divided into time slices that are dedicated to each application domain. In contrast, Figures 10.c and 10.d depict a scenario where domains are washed over the network as waves. In any given cycle, packets of both domains are being served in the network. Thus, the packets do not have to wait an extra cycle to move throughout the network. The so-called SurfNoC architecture notably enhances performance over TDM; however, it needs more buffering space and bigger switch allocators.

Although static allocation provides implementation simplicity, its static behavior can be guessed/outperformed by the attacker. Hence the achieved security is fragile. That motivation leads to dynamic schemes in [11], [66], [67], where they yield higher security as compared to the static-based architectures.

The authors in [11] used node obfuscation to tackle timing attack. Applications are migrated (task migration) to other nodes to provide more path diversity. It is worth mentioning that the new nodes should be compatible with the mapped application. The migrations are managed by the SoC firmware, where it keeps a list of PEs that match. Nevertheless, node obfuscation poses a substantial performance overhead and is subject to resource limitations on the chip.

Sepúlveda *et al.* [66] applied dynamic resource allocation but in a different way. The authors proposed a router architecture to address timing attacks by dynamically allocating several virtual channels to each input stream according to the communication and security requirements. The authors used this to keep the attacker's traffic independent of the sensitive traffic. A pool of virtual channels is available in the router, along with a pseudo-random number generator that randomly allocates a virtual channel to each input. While the approach produces promising results in terms of decoupling the sensitive and malicious traffic, it shows a relatively large area overhead compared to previous works [51]. Additionally, the authors have ignored the presence of HTs in the NoC.

Sepúlveda *et al.* [67] appended their work in [66] by proposing another dynamic allocation scheme. They proposed random arbitration and adaptive routing to address timing attacks. Generally, arbitration schemes are mostly deterministic, e.g., round-robin arbitration. In this study, the random arbiter uses pseudo-random number generators and physically unclonable functions (PUF) to achieve arbitration randomness. Moreover, the West-First routing algorithm provides route randomization to tackle both the security and performance problems.

To avoid the performance penalty of resource allocation, researchers in [52], [68] [35] opted for another approach. They diversify the routes from source to destination in various ways to avoid bottlenecks that are susceptible to DoS attackers. Indrusiak *et al.* [52] have factored in hard real-time performance constraints in addition to security. Authors have used a combination of source and distributed routing techniques to randomize the sensitive path. In source routing, the path selection is made within the local IP or the NI, in which packet latency and performance constraints are considered. In distributed routing, the path is randomly chosen according to the relative position of the packet and a set of rules derived by the turn-model. Also, to balance the security-performance trade-off, an evolutionary algorithm for task mapping has been used with the following inputs: NoC parameters, the security level, and the routing randomization approach. In this study, random route selection leads to higher packet latency.

Boraten and Kodi [68] have suggested routing traffic through under-utilized routers for security and performance gains. The proposed approach addresses both DoS and timing attacks. Applications are assigned with a security domain before data transmission. When there is a conflict between the high-security and low-security data domains, precedence is given to the low-security domain. If backpressure in the high-security domain is detected, the domain can request a routing change. A set of four routing algorithms with different levels of routing flexibility are used. Extra virtual channels are used to allow packets to switch their routing without having a deadlock. The proposed approach obtains a good security-performance trade-off; however, utilizing extra VCs and different routing mechanisms lead to a notable area overhead.

Reinbrecht *et al.* [35] introduced distributed timing attack involving at least two malicious routers/IPs. The two types of infected nodes are *Injectors* and *Observers*. In this scenario, a sensitive path carrying packets sent by the main memory is monitored by the *Observers*. The goal is to congest the sensitive path by *Injectors* and observe the throughput of the *Observer* node and detect sensitive packets. The nodes are constantly monitoring link bandwidth, and if the bandwidth threshold is exceeded, the routers send an alert to the neighboring routers. Upon detecting an attack, the routing algorithm switches from XY to YX to avoid using the same path for sensitive information. It is worth mentioning that this routing randomization strategy will fail to provide security if the attacker knows about the alternative routing strategy. Based on the previous assumptions, it is highly possible. The same threat model and countermeasure are used in [64] to recover AES key bits.

Table 3 compares the previous work in this section based on route randomization, resource allocation, area/power overhead, and performance impact.

C. PRESERVING DATA INTEGRITY & AUTHENTICITY

Given that the NoC fabric can attain data integrity, a message recipient can verify whether the received message has been tampered with or not. As in many cases where data integrity can be achieved through sender authentication, researchers have jointly addressed the integrity and authenticity in some works. In general, there are three major approaches to guarantee data integrity and authenticity in NoC-based MPSoCs. i) The application of error detection/correction codes and/or unkeyed hash functions that only addresses data integrity. ii) Joint use of keyed hash functions and message authentication codes to address both data integrity and authenticity. iii) Incorporating physically unclonable functions for authentication-only purposes. In this section, we review papers addressing data integrity/authenticity after a quick review of the preliminaries of the mentioned approaches.

A cryptographic hash function is a one-way mathematical function that creates a fixed-length message digest regardless of the input message size. The output of the hash function is called a message digest. The one-way property guarantees that the input data cannot be extracted based on the message digest. When the hash of the input data is computed, it is then appended to the original message and sent to the receiver. The receiver evaluates data integrity by running the hash algorithm on the message's body and comparing the result with the received tag. Since the message space could be far greater than the hash digest space, used hash functions must be collision-resistant, meaning that no two similar messages can be found with the same hash digest.

Boraten and Kodi [69] have proposed a combination of algebraic manipulation detection (AMD) and cyclic redundancy check (CRC) codes to address the integrity of NoCs. The authors have assumed that the HTs are smart enough to tamper with a packet while keeping its CRC correct. To address this threat model, they use AMD codes for

TABLE 3. Details of the previous works addressing timing attacks.

Reference	Adversary	Route Randomization	Resource Allocation	Area/Power Overhead	WRT	Performance Impact	WRT
Wang and Suh [51]	IP	No	Static	≈0%	NoC	*N/A*	*N/A*
Wassel et al. [65]	IP	No	Static	162% - 316% / 146% - 310%	Buffer, Crossbar	75% gain	TDMA
Ancajas et al. [11]	IP+NoC	No	Dynamic	*N/A*	*N/A*	Software-level	*N/A*
Sepúlveda et al. [66]	IP	No	Dynamic	9% / 8%	Router	≈25% gain	SurfNoC [65]
Sepúlveda et al. [67]	IP	West-First	Dynamic	11% / 5% for Random arbitration 9% / 8% for Adaptive routing	NoC	Up to 90% gain	Unprotected NoC
Indrusiak et al. [52]	IP	XY/YX, West-First	*N/A*	*N/A*	*N/A*	*N/A*	*N/A*
Boraten and Kodi [68]	IP	XY, 01TURN *, ROMM **	*N/A*	*N/A* / 1.84%	NoC	2 - 20% gain	NoC
Reinbrecht et al. [35]	IP	XY/YX	*N/A*	21.1% / 16.2%	Router	*N/A*	*N/A*

*Orthogonal one turn routing
**Randomized oblivious minimal multi-phase routing

sensitive packets and prioritize them against regular packets. The AMD code mechanism embeds the path information into the packet header. As a result, the packet integrity is preserved, and unauthorized duplicated packets can be detected/dropped at the destination router. As the sensitive packets (coded with AMD) are given a higher priority than normal packets (coded with CRC), the method is vulnerable to DoS attacks through the injection of junk sensitive traffic. The proposed approach is not immune to any HTs inserted in the NI as all mechanisms are being applied/checked at NoC routers. Overall, AMD yields significant area overhead compared to CRC. Also, the header flit is left unprotected, which could lead to the revelation of sensitive information according to [35].

Authors of [70] have tried to expand the idea of error detection/correction codes by utilizing a model checking method for NoC integrity. They have proposed a model checking approach to check computations of the router's pipeline stages. The checker's hardware is distributed over the router stages to perform model checking at the exact pipeline stage. The authors have claimed that the rules used in [71] are not enough to pinpoint HTs, so they have extended the ruleset. This model checker tries to detect more transient faults (through functional correctness checking) with the hope of activating/detecting probable HTs at the router. The proposed method assumes that HT characteristics are analogous to transient faults. HTs in this work are assumed to be capable of influencing resource allocations and corrupting data. Also, NoC buffers and status registers are protected from fault injection attacks. The hardware and the power overhead are 1.1% and 1.5%, respectively, compared to the baseline router. Since the behavior of HTs and transient faults is not the same in all cases, the proposed model checker fails in detecting some HTs, e.g., non-functional HTs trying to perform thermal attacks. Also, the model completely ignores any attacks that are not altering routers' functionality. This may include many confidentiality and integrity-related attacks.

Fort-NoCs [11] architecture proposes a hash-based packet certification to guarantee source integrity. Based on a lookup table located at the IP, a fixed tag generated by the SoC firmware is appended to the data. The data is then passed to the NoC and is routed to the destination. The SoC firmware checks this tag at the destination to ensure that a legitimate source IP issued the packet. As tags are updated only after the system boot-up, the system is not highly secure, i.e., it is susceptible to replay attacks and all analysis-based attacks. Moreover, it is not clear how the method should distribute the tags after every boot-up. The area, power, and performance overheads posed by packet certification are negligible.

The message digest of unkeyed hash functions only depends on the input data, whereas keyed hash functions additionally utilize a secret key to generate the message digest. Keyed hash functions are mostly used where the authenticity and integrity of data are both considered, whereas unkeyed hash functions only guarantee integrity. Data authentication is a process in which the receiver ensures that the intended party sent the data. The message authentication codes (MACs) generate a digest by using a private key shared between the source and the destination. The message and the tag are sent to the receiver to verify the message's security properties. Authentication will not be compromised as long as the key is not revealed to a third party. Keyed hash function and block ciphers are used as the MAC backbone. HMAC, SipHash, and cipher blocks in CBC and GCM modes are examples of used techniques. Figure 11 illustrates the MAC process. Although, authentication is achieved even with sending plaintext messages, authenticated encryption (AE) and authenticated encryption with associated data (AEAD) have been proposed to avoid information leakage. In AE, the integrity and authenticity of the ciphertext are evaluated at the destination. On the other hand, AEAD adds the ability to check the integrity and authenticity of some associated data (AD) in the plaintext appended to the ciphertext.

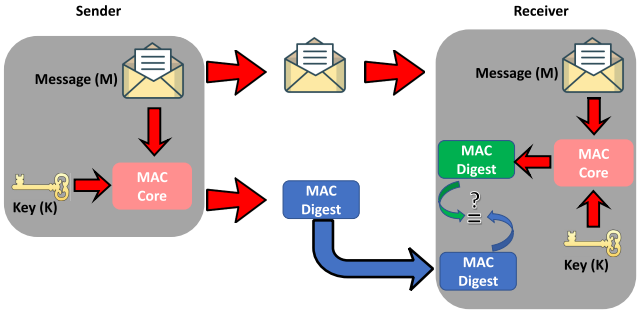


FIGURE 11. Message authentication process in the source and destination nodes.

The associated data can be the header information which is kept in plaintext for proper routing mechanism.

An authenticated encryption approach is used in the study by Sajeesh and Kapoor [61]. IPs are dynamically divided into secure and non-secure. Non-secure IPs are prevented from communicating with secure ones. Subsequently, malicious code injection attacks such as buffer overflow are prevented. The Authenticated encryption module is implemented within the secure IPs to verify the packet's source/integrity. The authors have used Galois counter mode (GCM) with GHASH [62] for encrypted authentication. There is a nearly 20% increase in slice LUT utilization and less than 15% increase in slice register utilization in FPGA implementation. The most notable shortcoming is that the encryption key remains constant throughout the lifetime of NoC. Also, the header information is left in plaintext, which could lead to cryptanalysis attacks [72].

Sepúlveda *et al.* [21] have used SipHash, a keyed hash function for message authentication [73] to address both integrity and authentication in the network. SipHash iteratively performs a series of add, rotation, and XOR operations to achieve fast MAC computation for short messages. The proposed scheme incurs hardware overhead of as big as 2% when compared to the entire baseline MPSoC.

Moriam *et al.* [74] have proposed an approach to send a linear combination of data packets (also known as network coding) for forwarding data packets. The method helps to ensure integrity, boost availability, and enhance the efficiency and robustness of NoCs. Additionally, this method can address replay attacks. The assumed threat model of the paper consists of malicious routers capable of packet dropping and tampering. The paper assumes that other components of the MPSoC, including NIs and IPs, are secure since they were designed in-house. However, due to design parameters like time-to-market and cost, these components have, in fact, higher chances to be designed by 3rd parties. A block cipher called mCrypton in CBC mode has been used as a lightweight solution for authentication without addressing any key exchange mechanism. The MAC computation needs up to 39 cycles at the sender/receiver while providing more security aspects. The area overhead compared to the baseline MPSoC is 2.7%. Overall, the devised approach incurs significant performance overhead.

Charles and Mishra [72] devised a trust-aware routing mechanism to circumvent the malicious nodes. The assumed threat model consists of malicious IPs that can modify packets' content to fail the authentication process, which will increase network congestion. A trust value ranging from -1 (untrusted) to $+1$ (trusted) is defined in the paper to utilize secure paths. Each node observes the trust values of its 1-hop and 2-hop neighbors to choose a trusted path. Nodes continually update their trust values based on a sigmoid function (depicted in Figure 13) and their recent communications history. Trust values diminish either if a packet is lost or the sender does not receive the ACK packet. In contrast, successful secure communications that deliver ACK packets to the sender will help boost the trust values. The proposed approach demonstrates significant performance improvements with only 6% area overhead compared to the baseline router. It is noteworthy that this method requires an end-to-end flow control mechanism to let secure packets return their ACK packets to the senders, which incurs performance overhead.

The concept of Physically Unclonable Function (PUF) was coined by Pappu *et al.* [75]. PUFs are one-way lightweight hardware security primitive which produce a unique output called a *Response* for a given input called a *Challenge*. The unique response will act as the entity identifier and can be used for device authentication and key generation. Unlike the encryption algorithms that integrate confusion and diffusion as sources of entropy, PUFs leverage manufacturing process variation [76]. Despite all the advantages, PUFs should be utilized with care due to their reliability issues caused by temperature and voltage variations [77].

The study by [78] has embedded two delay-based PUFs, namely arbiter PUF (APUF) and ring-oscillator PUF (ROPUF), in NoC. Since the PUF structure is intertwined in the router's architecture, it is extremely difficult for an attacker to initiate PUF removal/replacement attacks. The proposed PUFs use the available pool of multiplexers of the router's crossbar switch in its architecture; however, their architecture is reorganized to a cascaded form. The router can either work in its normal mode or switch to PUF mode. Pass transistors have been used [79] to enable these features. To produce random challenges, the buffer occupancy of the input ports under the *dynamic adaptive deterministic* (DyAD) routing algorithm [80] is used as a source of randomness. The hardware evaluations show an 11% and 7% increase in the area overhead compared to the baseline router for the APUF and ROPUF, respectively. Moreover, the PUF impact on the router's critical path is negligible, and the performance overhead is less than 0.1%; however, no network-level simulation is provided. Despite the promising result and thorough security analysis of the design, the functionality of the design relies on an adaptive routing algorithm.

One form of packet tampering is malicious modifications of the header flit, e.g., changing the flit type. This may compromise the flit's data integrity and result in a misrouted packet, deadlock, livelock, or flit loss. Frey and Yu [33]

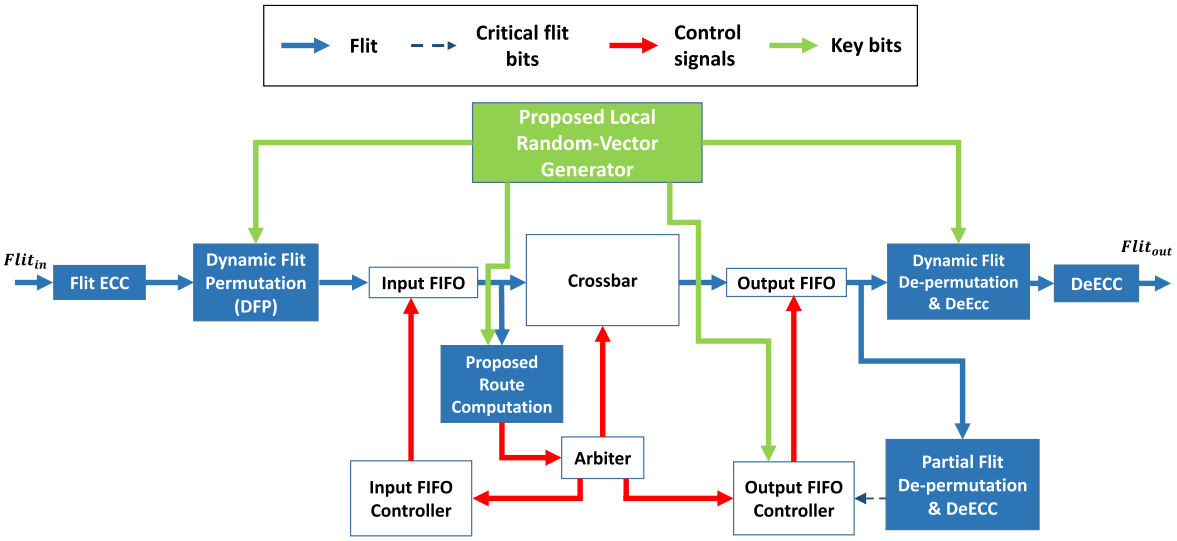


FIGURE 12. The proposed router architecture in [33] for HT mitigation and detection.

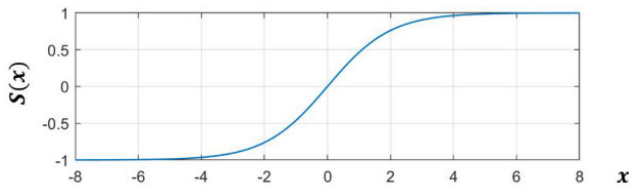


FIGURE 13. $S(x+\delta)$ and $S(x-\delta)$ are computed to update the trust values where S is the sigmoid function and δ is a small positive number.

have targeted packet tampering prevention by conducting flit integrity check and permutation of flit contents inside the router. The paper's assumed threat model is malicious routers capable of modifying the flit type or changing the packet address to give access to unauthorized IPs (spoofing attack). To tackle the problem, the ingress packets' critical fields are first encoded before entering the input FIFO of a router using Error Control Code (ECC). Next, they are scrambled using dynamic flit permutation. The whole process is shown in Figure 12. A PUF structure is implemented within each router to ensure the randomness of the permutation function. Extra modules for flit de-permutation and ECC decoding must also be implemented in the router to allow flits to be forwarded to the next node. The area and power overheads are 39% and 13%, respectively, with respect to the baseline router.

Table 4 compares the previous work in this section in terms of the integrity scheme, area, power, and performance overheads.

D. PRESERVING AVAILABILITY

In this sub-section, we review the DoS attacks in NoC-enabled MPSoCs that can be conducted by either malicious IPs or malicious NoC. Malicious IPs mostly use packet flooding to introduce congested areas to violate the real-time constraints of the chip. Malicious NoCs, on the other hand, perform packet tampering to cause packet retransmission, packet dropping and/or packet misrouting. Most of the papers that explore DoS attacks have assumed that the 3PIP NoC is the main suspect. Both software-level and hardware-level solutions have been provided to mitigate DoS in MPSoCs; however, hardware-level approaches introduce lower performance impact.

An attack model that has been repeatedly addressed in the literature is packet misrouting carried out by HT-infected NoC routers. The following steps should be taken to tackle this attack model:

- The HT-infected NoC router must be pinpointed with a proper detection mechanism.
- The router should be isolated from the rest of the network.
- Proper routing algorithms should be utilized to bypass the isolated router.

The countermeasures are generally composed of two main stages: 1) HT or DoS detection and 2) isolating the adversary or avoiding it by routing the packets in alternative paths. The HT/DoS detection is done in a variety of ways, mostly using machine learning techniques. On the other hand, the routing part is usually done using partially dynamic routing, as will be seen in the following.

Addressing the adverse performance impact of HT isolation techniques in [65], [68] is the primary motivation of the study by Wang *et al.* [34]. The authors have proposed using an artificial neural network (ANN) for HT detection. The model is trained offline using feature sets consisting of link and buffer utilization of each input port, local operation temperature, and the last epoch's transient error rate. In the detection phase, the trained model will label the routers as

TABLE 4. Summary of the previous works addressing data integrity.

Reference	Adversary	Integrity Scheme	Area / Power Overhead	WRT	Performance Impact
Ancajas et al. [11]	IP+NoC	Packet Certification	0.3% / 0%	SoC OCP Interface [63]	2% loss
Sepúlveda et al. [21]	IP+NoC	SipHash	1.1% / 1.2%	MPSoC	1.8%loss
Frey and Yu [33]	NoC	Dynamic Flit Permutation	39% / 17%	NI	*N/A*
Charles and Mishra [72]	NoC	Trust-Aware Routing	(6% / 28.3% Real Traffic 67.6% Synthetic Traffic)*	Router	43.6% gain compared to the HT-infected NoC
Sajeesh and Kapoor [61]	IP	AES (GCM mode), GHASH	20%** / *N/A*	NI	≈0%
Moriam et al. [74]	NoC	mCrypton (CBC Mode)	2.7% / *N/A*	MPSoC	26-39 cycles
Boraten and Kodi [69]	IP+NoC	AMD, CRC	*N/A*	*N/A*	1% loss

*Energy improvement compared to the HT-infected SoC
**FPGA resoruces

HT-infected or HT-free. This phase is followed by an HT mitigation stage in which the predicted labels are fed to a smart routing module to choose between one of the three available routing algorithms: O1TURN, West-First, and Negative-First. Packets are also labeled as high-security or low-security packets if any of the source or destination routers are HT-free and HT-infected, respectively. A bypass channel is integrated into each router to pass the high-security packets to avoid the HT-infected routers. The router's routing decisions are based on a deep reinforcement learning (DRL) controller that selects the routing algorithms with the highest expected long-term return in terms of network performance and energy efficiency. The results show higher HT detection accuracy and lower latency and energy consumption compared to the previous works with only 3% area overhead compared to the baseline router.

Madden *et al.* [32] proposed a remedy for DoS attacks using spiking neural networks (SNN). SNN is adopted because it is usually used in applications that require minor changes in the neurons' weights as a result of minor changes in the input training set, such as a moving object in a static environment in video applications. In fact, this is very similar to the anomaly in network traffic caused by HT to congest the network. The attack is carried out by flooding the network with unnecessary packets. The detection scheme tries to detect the anomalies in the NoC traffic patterns to pinpoint the attack. The spiking network identifies the temporal patterns within the data. It observes the total number of request-to-send signals in a certain time interval by a router to detect potential attacks. While achieving 86% HT detection accuracy in different attack scenarios, the area overhead was not reported; however, it seems to be significantly high. Moreover, this method does not support credit-based flow control NoCs, and the accuracy depends on the length of the attacks.

Another application of machine learning techniques is used to address misrouting as well. Firstly, a model has to be trained based on a dataset. Then the trained model is used to detect anomalies at runtime. The training phase of supervised learning algorithms requires significant time/energy, and due to the limited power budget and timing constraints in the NoC context, it is done offline. On the other hand, unsupervised learning algorithms do not require any training. Authors of [81] have proposed supervised and unsupervised machine learning frameworks to detect real-time anomalies such as packet retransmission, packet misrouting, and tampering in NoC-based many-core architectures. The feature extraction is narrowed down to the packet source and destination addresses, transfer path, and transfer distance. To reduce the hardware complexity, the model is trained offline using a ''Golden Dataset'' (an HT-free network), and anomalies are injected randomly in the routers. Four supervised learning algorithms have been used: K-Nearest Neighbor (KNN), Support Vector Machine (SVM), Linear Regressor, and Decision Tree. Moreover, four unsupervised learning algorithms are used: Simple K-Means, Farthest First, Estimation Maximization, and Hierarchical Clustering. While being more costly, supervised learning techniques outperform unsupervised techniques in accuracy, which is above 90%. In a quad-core router, the area and the power overhead are 3% and 9%. The latency overhead of the proposed architecture is 18% of the total execution time. Also, the unsupervised learning techniques are not as effective as supervised algorithms against the detection of spoofing attacks.

To compensate for the fact that the model could lack training on unseen attacks during the initial training phase, the same authors devised an approach to update the trained model by utilizing a modified balanced window (MBW) online machine learning algorithm. In other words, it is a mistake-driven learning model for detecting unexpected attacks at runtime. The prediction model is updated if its prediction was wrong [82]. Overall, using MBW leads to a lower area overhead, lower detection latency, and higher detection accuracy.

Rather than costly ML models, Daoud and Rafla in [36] addressed the same threat model. The HT circuit is first inserted into the router's logic with less than 1% area overhead. The HT detection scheme is implemented within the router and locates the HT-infected router based on the packet input port and its destination. The malicious router's address will be propagated to the adjacent routers, and a routing algorithm borrowed from the fault-tolerant context is employed to detour the infected node with less than 1% area overhead. Since it is unknown that at what router's pipeline stage the packet was misrouted, the proposed countermeasure in [70]

can be utilized to pinpoint the HT's exact location. In the presence of more complex HTs, the detection messages from the HT-infected router to the neighboring node can also be dropped.

In [37], the same authors addressed the packet dropping attack caused by an HT-infected router in which a black-hole router drops any packet that passes through. Subsequently, the infected node will not forward the received packet and, therefore, will cause DoS. The HT model poses less than 2% power and area overhead compared to the baseline router. The number of black-hole routers and their spatial distribution over the network has a drastic impact on the attack's effectiveness. To remedy the problem, the same security-aware routing used in [36] was proposed.

Trojan-aware routing was also adopted in [83] to address DoS attacks started by malicious routers. The assumed HT takes control of the router to misroute any packet heading to destinations located at the same column as the malicious router. The proposed routing consists of three stages: detection, shielding (isolation), and HT detouring. Each router uses a combination of the source ID, input port, and destination ID for each packet to check if the XY routing was violated. In case a router detects a routing violation, it alerts its neighbors to bypass the malicious router. The proposed method achieves acceptable latency and throughput results while having area and power overheads less than 3% compared to the baseline router. However, the application of the proposed method is limited only to the NoCs using XY routing algorithms.

Other ways to detect anomalies in traffic can be achieved through traffic monitoring. For example, Rajesh *et al.* [14] introduced a case in which a third-party NoC causes bandwidth denial. The HT in the NoC can suppress the crossbar allocation requests and de-prioritize arbiters to impose latency. The proposed runtime latency auditor scheme detects latency anomalies in packets and utilizes them to identify the malicious node. The security solution compares packets' latency at a given node with adjacent nodes due to their spatial and temporal similarities. The latency computations are done in the SoC firmware, placed between the NI and the local processing IP. The area and power overheads are 12.73% and 9.34%, respectively, when compared to the SoC OCP (open core protocol) interface [63]. There are critical problems with this HT detection scheme. First, latency is influenced by the network's workload, leading to false positives [83]. Second, using static thresholds may also lead to more false positives and false negatives in HT detection.

Another traffic monitoring technique is introduced by Charles *et al.* [9]. In this work, a different type of bandwidth denial attack is addressed which DoS is caused by flooding the network with useless packets. The authors argue that an NoC-based solution must be lightweight and real-time to adhere to the NoC constraints, so they introduced a real-time traffic monitoring scheme to address the attack. The authors assumed that a malicious IP could target a memory controller by flooding the network with unnecessary packets. After analyzing the network's communication patterns, packet arrival curves and destination packet latency curves are constructed. In other words, the traffic behavior is statically stored within the routers. Packets failing to adhere to the curves are subject to a DoS attack. Lastly, a broadcasting detection mechanism is used to localize the attacking node. By receiving alert messages from the neighboring node, the flag values within each router are updated, and they will eventually pinpoint the attack source. The method yields 5.93% and 3.87% area and power overhead compared to the baseline router, respectively.

Frey and Yu [84] address the NI's security. The *FSM (finite-state machine) control* module is an attractive target for HT implementation since it is the main control logic of the transmitter. Tampering with the functionality of the FSM control will change its behavior and eventually lead to performance degradation. The authors have added key bits and dummy states in the FSM to prevent and detect attacks. As shown in Figure 14, without knowing the key, the attacker will jump to the dummy states and is not able to return to the legal states. This trap will eventually lead to HT detection. Moreover, the previous and current states of the FSM are constantly monitored to detect any illegal state transition by the HT. The downside, however, is that the key may be inferred due to the limited number of states. Also, this approach cannot detect the duplication attack introduced in [38]. The power and area footprints of the countermeasure are 1.7% and 3.2% respectively compared to the baseline NI using OCP (open-core protocol) [85].

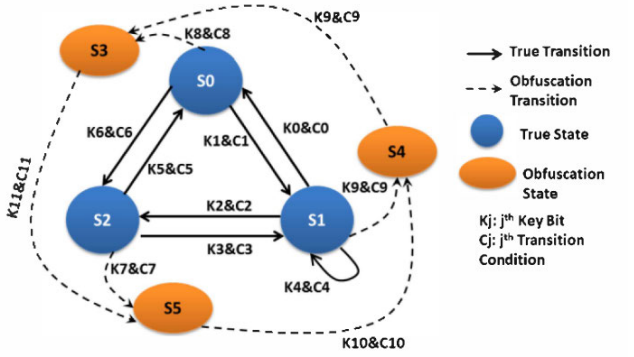


FIGURE 14. The obfuscated FSM design in [84] that uses key bits to mitigate and detect attacks.

In the proposed approach by Hussain *et al.* [44], each core has an E2E (end-to-end) HT detection module. In the E2E scheme, authentication is performed only at the destination node, which imposes less power and performance impact. The goal of the so-called energy-efficient HT detection design (EETD) is to localize HTs by dispatching searching agents from the destination node. In case no HT was detected at the destination, the localization units (LUs) will be power-gated. The effectiveness of this approach highly depends on the threshold setting defined by the designer. If it is not fine-tuned, it could lead to false positives and waste

TABLE 5. Summary of the previous works addressing availability.

Reference	Adversary	Countermeasure	Area / Power Overhead	WRT	Performance Impact	WRT
Wang et al. [34]	NoC (Router)	ANN (HT detection) and DRL (Dynamic routing)	3% / *N/A*	Router	29% gain	[65] [68] [88]
Madden et al. [32]	IP	Spiking NN	*N/A*	*N/A*	*N/A*	*N/A*
Kulkarni et al. [81]	NoC (Router)	Supervised/Unsupervised ML (HT Detection)	2% - 12% / 1% - 3%	MPSoC	18% - 25% loss	HT-Free MPSoC
Kulkarni et al. [82]	NoC (Router)	MBW online ML (HT Detection)	56% / –	MPSoC [89]	50% gain	[89]
Daoud and Rafla [36]	NoC (Router)	Trojan-Aware Routing	0.4% / 0.6%	Router	*N/A*	*N/A*
Daoud and Rafla [37]	NoC (Router)	Trojan-Aware Routing	[36]	[36]	[36]	[36]
Manju et al. [83]	NoC (Router)	Trojan-Aware Routing	2.78% / 3%	Router	38% - 48% gain	HT-Infected NoC
Rajesh et al. [14]	NoC (Router)	RLAN	12.73% / 9.84	SoC OCP [63]	5.4% loss	NoC
Charles et al. [9]	IP	Packet Latency Curves	6% / 4%	Router	0%	*N/A*
Frey et al. [84]	NoC	NI FSM Obfuscation	3.2% / 1.7%	Baseline NI	*N/A*	*N/A*
Hussain et al. [44]	NoC (Router)	Verification Units	*N/A* / 38%*	[33]	40% gain	[33]
JYV et al. [86]	NoC (Router)	Bit Shuffling (Proactive)	21.2% / -	Router	20% gain	HT-Infected NoC
Boarten and Kodi [87]	NoC (Link)	Link Obfuscator (Proactive)	2% / 6%	NoC	1-3 cycles loss per node	NoC

*Energy consumption is reported.

of power. These static threshold settings can be compromised by an adversary through reverse-engineering [38]. Energy overhead and performance overhead are reduced by 38% and 40% compared to [33]. The area overhead was not reported; however, it should be significant due to the use of both end-to-end and hop-to-hop HT detection modules. Additionally, it is not clear how the proposed approach can locate the source of packet flooding attacks.

Different from the earlier works, the techniques of [86] and [87] proactively prevent the HT from activation and causing DoS attacks. In [86], JYV *et al.* implemented an HT in the NoC router that targets sensitive fields (packet source, address, flit quantity, and sequence number) of the flits to suppress network performance. To thwart the HT, a bit shuffling per-router technique is used to reduce the HT activation probability. Input bits are shuffled before entering the input FIFO, and the shuffling pattern (key) is extracted from the input message itself. There is also an address extractor module to partially reverse-shuffle the flit fields and extract the destination address. This stage seems redundant as the route computation can be done before shuffling. The performance impact of the countermeasure is low, and the area overhead is 21.2% compared to the baseline router.

Boraten and Kodi [87] implemented a link HT that injects faults. These faults are beyond the correction capability of ECCs (error correction code), and will cause packet retransmission and launching DoS attacks. The target-activated sequential-payload (TASP) HT impersonates itself as a transient fault by changing the fault location using an FSM (finite state machine). The FSM decides when and where to activate the HT payload. To get past the single-error correction double-error detection (SECDED) ECC module, the attacker only flips two bits of the output. The authors have used a switch-to-switch mitigation technique to prevent the HTs from activation. This technique helps to locate the HT in which flits are obfuscated by shuffling, inverting, and scrambling data. The detection module analyzes and keeps

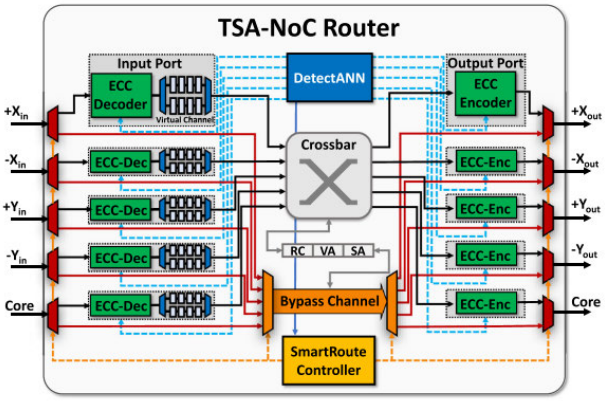


FIGURE 15. The proposed router architecture in [34]. The DetectANN component is used for HT detection, the Bypass Channel is used for passing high-security packets, and SmartRoute Controller chooses one of the three available routing algorithms.

a history of the passed flits to use different obfuscation techniques for the retransmitted flits. The proposed switch-to-switch link obfuscator will lead to 2% and 6% power and area overhead, respectively, compared to the entire NoC. The high power consumption is due to the switch-to-switch obfuscation method.

Table 5 summarizes the previous works addressing availability in terms of HT location, the proposed countermeasure, area, power, and performance overheads.

E. ACCESS AUTHORIZATION

In many cases, the identity of the requesting router/IP must be verified prior to granting data access. Consequently, a set of rules known as access control can be enforced to limit the access of certain IPs. In the authorization phase, a malicious requesting IP/router, so-called an *Initiator*, targets the valuable data assets to achieve the following goals [90]:

- Extracting secret information by reading from restricted memory addresses.

- Changing system's configuration by writing into restricted addresses.
- Reducing system's bandwidth by flooding the network with unnecessary memory requests. (This case is discussed in Subsection IV-D.)

Firewalls have been widely used [19], [91]–[93] to realize/enforce the access control rules. A firewall contains lookup tables that store the access rights. The access decision is made based on the following criteria [92]: i) initiator's source ID (might be a task ID or an IP ID), ii) address requested by the initiator, iii) length of the requested data, iv) whether the operation is a load/store, v) whether the operation is accessing data/instructions, and/or vi) role of the initiator (user/supervisor). To address the dynamic workload of NoCs, firewalls need to support programmability at runtime; otherwise, the impact of the firewalls in terms of latency could be significant. As illustrated in Figure 16, firewalls can either be placed in the NI of the target nodes or the NI of the initiators, and this choice can be made during the design time based on prior knowledge about the characteristics of the network's workload. When the distributed firewall is adopted, bandwidth is better utilized because packets will be rejected at the initiators before reaching the target NI. The firewalls could also be placed between the routers [94]. Security wrappers have been employed instead of firewalls in secure zones. Wrappers do not require lookup tables and hence lead to less overheads. As we discuss in Section IV-F, firewalls and wrappers can be used in the implementation of secure zones as well.

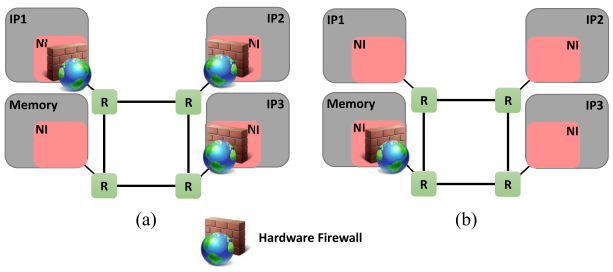


FIGURE 16. Firewall can be placed at the (a) initiators and (b) the target NI.

Although static firewalls impose lower overheads, they are not efficient when the workload has dynamic characteristics. Among the static firewalls we discuss [40], [91], [93]–[95]. Fiorin *et al.* [93] are among the first to propose a secure platform for NoCs based on firewalls. Data is protected from unauthorized accesses using a set of *Data Protection Units*. The units are implemented within the NIs, and lookup tables are used to grant/refuse accesses. The method uses a predefined format for flits/packets that contains information about memory accesses, including the source/destination IPs, memory address requested, and other detailed information. The authors have missed the point that the used packet format itself can be utilized for facilitating timing attacks. Also, the extra information that is carried by packets imposes performance overhead on the network. Later, the dynamic runtime configuration of the units was proposed in [92] through a Network Security Manager architecture. To limit the number of nodes that can update the access policies, only trusted and supervisor nodes can communicate with the security manager.

Grammatikatis *et al.* [91] proposed a static firewall to protect the shared memory from malicious code and viruses. The firewall is located in the NI of the initiator side and thwarts information leakage and DDoS attacks. As mentioned earlier, the firewall architecture at the initiator prevents the NoC from early saturation since it stops memory accesses before entering the network. The proposed firewall protects memory segments (with variable segment size) instead of memory entries using a segment-level rule-checking module that monitors the issued memory accesses. Accordingly, DDoS attacks carrying out with multiple processes will be detected and denied.

Hu *et al.* [95] placed a firewall in an application-specific NoC. Unlike general purpose NoCs, the topology is modified in specific-purpose NoC to yield the best performance results. The knowledge about the application at the design time enables the designer to adopt static security policies. The authors have assumed secure domains in which a group of initiators and targets are secure, and firewalls are not needed within the domain. Instead, inter-domain firewalls between the routers are used to deny the malicious flow. Additionally, bandwidth is more preserved since additional header information is not needed anymore. Nevertheless, finding the optimal firewall location is challenging due to the irregular topology of NoC. Integer linear programming (ILP) is used to solve this problem. The proposed approach leads to a substantial bandwidth enhancement compared to when the firewall is located at either initiators or targets.

Achballah *et al.* [94] proposed a firewall that is placed between the NoC routers and separates the secured and non-secured zones on NoC. The firewall serves two purposes. i) forcing access control to prevent secure data extraction, and ii) verifying physical links' occupation time to thwart DoS. Each initiator's access restrictions are defined based on the source and destination address of the requests, and these rules are then programmed into the firewalls. Additionally, the occupation time of physical links is monitored to address DoS. A counter within the FIFO buffer is employed to count the link's occupation time in the number of clock cycles. The IP request will be denied if the counter exceeds a specific threshold. As the firewall is implemented with fully combinational logic, the latency overhead is minimized. Since the proposed firewall is not utilizing authentication schemes, it fails to address packet tampering and spoofing attacks.

In [40], initiators' ID along with access codes are used to guarantee that nodes cannot access forbidden memory addresses. IPs are divided into four virtual groups, namely highly trusted, trusted, non-trusted and unknown islands.

Memory is divided into regions, and each island can only access its dedicated address range.

Few dynamic firewalls have been proposed in the literature [90], [92], [96]. Cioranesco *et al.* [90] argue that authenticated encryption between the initiator and the target imposes a significant latency overhead. The authors proposed a security scheme called CSAC (cryptographically secure access control) to authenticate the firewall's programming agent and secure the programming sequence to address this problem. To achieve this goal, HMAC (Keyed-Hash Message Authentication Code) is used. It takes the session key, incoming programming sequence, and state variable (tracker of the session's history) as inputs. The key is programmed for each session, and the SoC's hardware root-of-trust is responsible for the key exchange mechanism. HMAC guarantees that only the parties who share the key can verify the authenticity of the message. The proposed method is resilient against injection/glitching techniques and replay attacks. It also guarantees that the access policies can be securely maintained. This method has been used in the industry (FlexNoC) as well [19].

Fernandes *et al.* [96] have proposed a firewall placed between the NI and the routers' local ports. The firewall offers two security services: i) restricts unauthorized accesses issued by malicious initiators, and ii) compares the NI-produced packet's header-flit with an internal register within the firewall to detect any malicious tampering with the header information. Although the firewall can address spoofing attacks, malicious routers can still tamper with the header information if the NoC is considered insecure.

Table 6 summarizes the previous works in this section by firewall location and type.

TABLE 6. Details of the previous works addressing access authorization.

Reference	Firewall Location	Firewall Type
Fiorin et al. [93]	Initiator / Target	Static
Grammatikatis et al. [91]	Initiator	Static
Hu et al. [95]	Between Routers	Static
Achballah et al. [94]	Between Routers	Static
Kinsy et al. [40]	Target	Static
Fiorin et al. [92]	Initiator / Target	Dynamic
Cioranesco et al. [90]	Target	Dynamic
Fernandes et al. [96]	Initiator / Target	Dynamic

F. SECURE ZONES

When it comes to secure communications among a group of IPs, *secure zones* (SZs) could be an effective solution [97]. For example, when a multi-task application is mapped on multiple cores of an MPSoC for optimal performance/energy purposes, providing secure data exchange among the tasks may need numerous parallel encrypted communications. It will impose the complexity of $O(n^2)$ where n is the number of tasks requiring secure communication. SZs are, in fact, network-level facilities to ease intra-group secure communications at the system/application level. It is worth mentioning that the concept of SZ is used in some real-world applications that employ MPSoCs [98].

In an SZ, IPs with the same security requirements are grouped and treated in the same way. SZs might be used to protect MPSoCs from malicious components and/or data from various attacks. In terms of shape, a zone can be rectangular, non-rectangular, or disjoint. In a rectangular zone, all minimal paths that connect zone members are located inside the zone. This mitigates the exposure of the zone's data to untrusted parts of the NoC. However, in a non-rectangular zone, some minimal paths in the zone have links located outside the zone. Finally, in a disjoint zone, members form multiple islands are not physically connected. Consequently, zone members will have to rely on untrusted NoC resources to communicate. Figure 17 shows possible arrangements of SZs. As can be seen, both rectangular and non-rectangular zones can share zone members as well. In this case, we have to ensure that the shared IP does not leak information between the zones. Figure 17.e illustrates how logical zones are formed when the zone members are not physically connected. Members of a logical zone cooperate in running parts of a shared application.

SZs might be used to fulfill either of the following security goals. 1) To protect MPSoCs against unauthorized accesses to resources such as sensitive memory/cache located at some cores. 2) To protect sensitive data/traffic flowing the NoC fabric of MPSoCs. SZs can be used to address a wide range of security attacks, including DoS/DDoS, packet dropping, spoofing, tampering and eavesdropping in MPSoCs. These protections are achieved since members of an SZ are considered trusted. In most continuous SZs, data can even be sent in plaintext for intra-zone communications [29] as the data will not meet any untrusted NoC components. However, having communications between non-physically connected SZs will be challenging. That is why most researchers have proposed to encrypt the data for inter-zone communications between multiple (Figure 17.a) or discontinuous zones (Figure 17.e). In the rest of this section, we will review previous works that used SZs to address the secure execution of applications.

1) STATIC SECURE ZONES

Depending on the assumed threat model and the characteristics of the target application(s), SZs can be defined either at design time (static zones) [40], or at runtime (dynamic zones) [40], [61], [99]. Unlike dynamic SZs that can support various applications at runtime, static SZs are only limited to known applications at design time. In addition, static SZs might be easily outperformed by other static security solutions since the characteristics of the target application is available at the design time. For example, one may opt to design custom NoCs that avoid unnecessary channels to force routing data in predetermined paths. This solution prevents exposing packets to untrusted routers/IPs and is applicable in cases that the target applications are known.

In [100], authors have proposed an obfuscation module to add random delay to packets generated inside the static

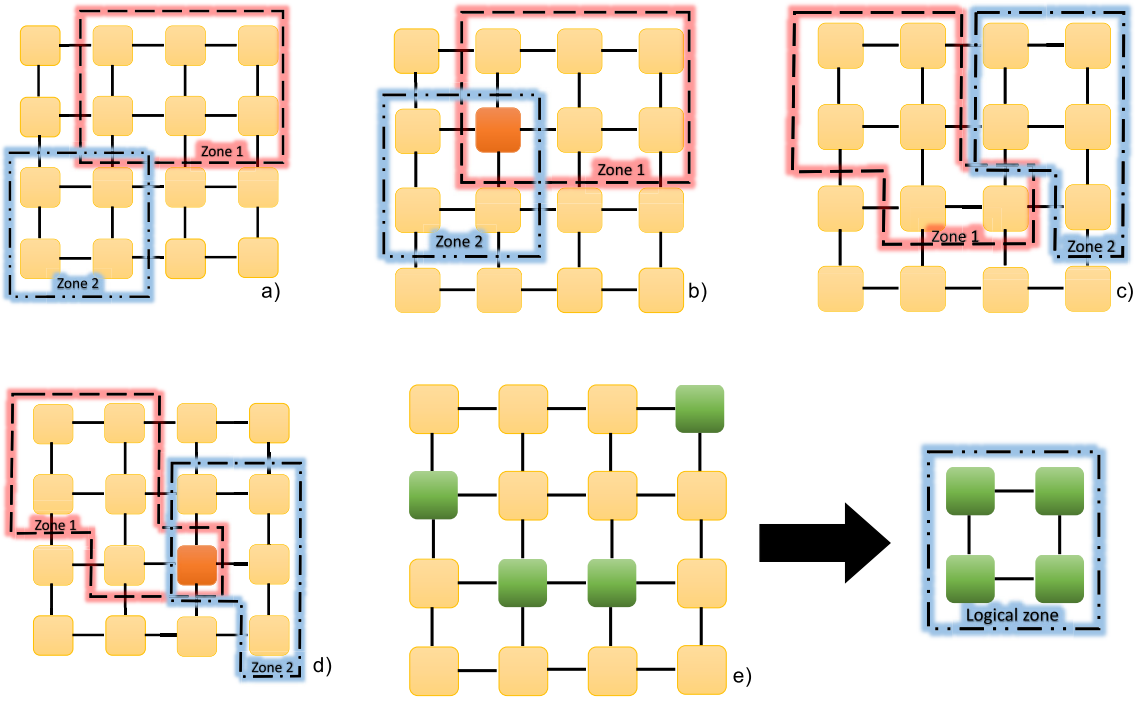


FIGURE 17. Possible arrangements of secure zones, a) rectangular non-overlapping, b) rectangular overlapping, c) non-rectangular non-overlapping, d) rectangular overlapping, e) logical zone.

zone to protect them against timing attacks and probing. Consequently, malicious IPs are no longer able to monitor the computation time and/or cache hit/miss rate of the target application. The obfuscation module, placed in the NI, can impede all packets by a constant (so-called blinding) or add a random delay before allowing the packets to leave the NI (called masking in the paper). The resources like the AES encryption component can be secured inside the zone by utilizing the proposed obfuscation. The paper uses circuit and packet switching mechanisms to transmit normal and secure packets, respectively. This is done with the aim of revealing smaller chunks of sensitive information to the attacker as in the packet switching flits can be stored at different routers. Security-wise, it is not clear from the paper why circuit switching is involved. Overall, the discussions on dual-switching are not complete and need more elaborations. The masking and blinding methods respectively impose 12.61% and 26.94% performance overhead. Compared to the baseline router, the method's power and area overheads are 18% and 16%, respectively.

The authors of [101] have proposed the idea of runtime-controlled security wrappers to form static SZ, which are segmented from other parts of the MPSoC. After the security wrappers are set, the traffic originating from outside the zone is no longer allowed to enter it, so alternative paths will be selected to bypass the SZ. As IPs inside the zone are all dedicated to the secure execution of the application, non-secure applications must be migrated to non-SZ IP cores after defining a proper continuous shape for the zone. A configuration controller is used to update SZ policies based on the MPSoC's task mapping information. Firewalls implemented in the NIs are used as a security mechanism to protect the nodes against malicious requests. Due to the need for task migrations, this method requires a significant time (up to 100K cycles) to form an SZ, which might not be acceptable in applications with real-time requirements. Additionally, the performance degradation of the method is tightly dependent on the SZ's shape, i.e., having a wide SZ in the center of the network imposes a long bypassing path to non-secure packets.

Due to the shortcomings of static SZs, most researchers have tried to add some levels of dynamics to static SZs. This is achieved by allowing the user to control static SZ [40], or reshaping the static SZ with task migration [102] and similar methods. In [40], Kinsy *et al.* have proposed a design-time SZ which can be controlled by the user. The SZ consists of four islands of IPs: highly-trusted, trusted, unknown, and non-trusted. The trust tag is based on the IP's source of origin. Applications are further grouped into trusted and untrusted as well. Cores ID along with access codes are used to guarantee that each core is not accessing forbidden memory addresses. Additionally, a public key exchange mechanism is used where a lead node in each secure island stores other islands' public keys for communication. Since each island can access a specific portion of the memory, any given node that needs access asks the lead node for permission. Access will be granted or denied based on the trust tag. Although the zones are static, cores can be added/removed from the

islands dynamically based on the user requirements and some defined policies. Lastly, a routing algorithm is introduced in which the traversal of zones by non-member-generated traffic is limited. Performance degradation is less than 9% in different benchmarks, while the area overhead is 17%. While stated as a low-cost solution, it is not the right conclusion.

The SZ proposed in [102] is a static one that can be reshaped dynamically at the runtime. Upon detecting a malicious IP, the corresponding zone is responsible for blocking the IP and finding alternative paths by changing the routing algorithm. Two distinct routing algorithms are used for intra-zone and inter-zone communications. The intra-zone routing algorithm is a fixed routing algorithm based on restricting some turns, while the inter-zone algorithm is more flexible, i.e., a non-minimal Odd-Even routing that offers higher routing adaptivity. Since the paper relies on a firewall for detecting malicious IPs, the achieved security is limited to that of the firewall. Also, attacks that can bypass the firewall can easily compromise the security of the system. The authors have reported the power, area, and latency overheads of 5.8%, 7.2%, and 7.8%, respectively, with respect to the whole MPSoC.

2) DYNAMIC SECURE ZONES

Sajeesh and Kapoor [61] is among the first papers to address dynamic SZs. The authors have proposed separating MPSoC into secure IPs, non-secure IPs, and link IPs. While secure IPs are trustworthy and non-secure ones are not, link IPs act as firewalls between the two groups to filter unauthorized accesses to secure IPs. The SZ in this work is dynamic that may result in constructing disjoint zones. Authors have used authenticated encryption to protect sensitive IPs from DoS, extraction of secret information, and hijacking attacks at the cost of 15-20% extra hardware requirements. As one of the first papers in the field, many details of constructing, attaining, and security of the zone are not discussed in the paper.

In [99], the authors have devised dynamic continuous SZs backed up by a manager IP per zone. They have proposed the CEASAR-MPSoC architecture to exchange encrypted and authenticated packets for protecting/configuring the network firewalls. The method defines firewall tables at each IP to implement user-defined SZs. A new NI architecture is proposed to perform firewall management, including rule update and enforcing packet drop in a failed authentication case. The authors have used the AEGIS and ASCON algorithms for the authentication and encryption of firewall management packets. Upon receiving a firewall management request by a manager, the manager's NI starts generating an encryption packet to implement the request and sends the packet to corresponding zone members. Accordingly, receiving members update their firewall tables after decrypting and authenticating the firewall management packet. The encryption unit is shared between the packetizer and de-packetizer units preventing the same NI from sending and receiving firewall management packets simultaneously. AEGIS and ASCON cores have respectively added 12% and 23% latency and 277.6% and 18% area overheads compared to baseline NoC.

Other researchers have used SZs to protect data/applications on an MPSoC. Watcher *et al.* [103] have proposed an architecture that supports application-level SZ to protect applications against DoS attacks, timing attacks, spoofing attacks, and malicious applications. To protect a specific application, the proposed architecture implements hardware wrappers that allow the manager IP to construct a secure region around the IP cores that host the application. To start a new zone, the manager broadcasts a message containing the upper right (UP) and lower left (LL) corner addresses of the zone, so that boundary nodes start creating their wrappers. Consequently, all packets coming from IPs outside the zone will be blocked, i.e., they are not allowed to enter the zone. Also, if other applications are running on the zone IPs, they will be suspended until zone termination. The study only reported the area overhead, and it is 18.6% with respect to the baseline router.

Fernandes *et al.* [30] have defined three communication scenarios for routing sensitive information inside and outside of an SZ: 1) full intra-zone communication in which both the source and destination IPs are located inside the same zone, 2) partial intra-zone communication for source/destination IPs in the same zone having their communication path partially outside the zone, and 3) inter-zone communication, where the source and destination IPs are located in different zones. The paper has used routing policies to guarantee secure transmission of packets while deadlock is prevented. The authors have modeled the network with the weighted graph at which nodes are IPs and edges are network channels with security weights for each edge of the graph. The security weights are used along with the Dijkstra algorithm to find the shortest secure path for packets. This approach aims to route the packet inside the zone as much as possible, while packets traversing through insecure zones are encrypted. The authors have not addressed how to measure/compute the security weights, i.e., the weightings are assumed to be known at the design time. Moreover, this method is vulnerable to most of the attacks introduced by malicious routers, since routers are assumed trusted.

Sepùlveda *et al.* also proposed an architecture in [104] that uses hierarchical group key distribution protocols for MPSoC protection. In this method, members of an SZ first discover a public partial group key through pre-loaded keys in their local key buffers. Then, a hierarchical Diffie-Hellman protocol is used to obtain a secret group key that enables members of the SZ to start secure communications. The proposed method needs the network's mapping information to accomplish the first step, i.e., key discovery. However, this may not be available in all situations as the applications running on an MPSoC may change over time resulting in a time-variant mapping. Also, the local key buffers potentially impose high security/reliability risks to the system.

Sepùlveda *et al.* [105] proposed an architecture that provides authentication, access control, and confidentiality

TABLE 7. Secure zone formation, architecture, and security target.

Reference	Protection Target	Zone Arrangement	Zone Creation	Zone Status	Zone Shape	Methods
Reinbrecht et al. [100]	Node / Traffic	*N/A*	Design-time	Static	*N/A*	Dual Packet Switching / Obfuscation function
Sepúlveda et al. [101]	Node / Traffic	Continuous	Design-time	Static	*N/A*	Monitoring / Firewall
Kinsy et al. [40]	Node	Discontinuous	Design-time / Runtime	Static / Dynamic	Logical	Diffie-Hellman / MAC
Sepúlveda et al. [102]	Traffic	Continuous	Design-time / Runtime	Static / Dynamic	*N/A*	Firewall / Region Based and Non-minimal Odd-Even Routing
Sajeesh and Kapoor [61]	Node	Disjoint	Runtime	Dynamic	Logical	GCM Authenticated Encryption
Azad et al. [99]	Node	*N/A*	Runtime	Dynamic	Regular / Irregular / Rectilinear	Firewall / Authentication / Encryption
Wachter et al. [103]	Traffic	Continuous	Runtime	Dynamic	Regular	BroadcastNOC architecture for security purpose / Wrapper
Fernandes et al. [30]	Traffic	Continuou / Discontinuous	Runtime	Dynamic	All	Segment-Based Routing Algorithm
Sepúlveda et al. [104]	Traffic	Continuous / Discontinuous	Runtime	Dynamic	All	Authentication / Diffie-Hellman Group-Key Agreement
Sepúlveda et al. [105]	Node / Traffic	Continuous / Discontinuous	Runtime	Dynamic	Rectilinear / Logical / Regular	Authentication / Encryption / Secure Routing
Caimi et al. [39]	Node / Traffic	Continuous	Runtime	Dynamic	Rectalinear	Security Wrappers / Optional Routing Algorithms
Sepúlveda et al. [29]	Node / Traffic	*N/A*	Runtime	Dynamic	*N/A*	Firewall / Authenticated Packets

services through creating SZs. SZs follow two rules i) exterior packets are not allowed to enter the zone, and ii) only IPs at the zone's borders contribute to generating the group keys based on Diffie-Hellman protocol. Communication among the IP members of the SZ is encrypted through the secret group key. By establishing a secret group key among the IP members of a zone, NoCs can isolate sensitive traffic and prevent data leakage. In this work, the penalty of the SZ creation is reduced up to 35% by reducing the number of IPs contributing to the key generation process. However, the authors have not reported the overall performance impact of the proposed approach in NoC. The architecture's area and power overheads are reported as 9.2% and 4.1% of the NoC, respectively.

The authors of [39] have used a global manager processor (GMP), multiple local manager processors (LMPs), and slave processors for creating application-level SZs. GMP sends sensitive applications to one of the LMPs, creating an SZ with a rectangular shape. The SZ's rectangular shape guarantees secure communication of zone members via secure links (links inside the same zone). Processors do not share any resources in SZs. When an SZ is created, only packets destined to one of the secure applications are allowed to enter the SZ, and other packets will be deflected. The fixed locations of GMP and LMPs in this method make the architecture prone to DoS and cryptanalysis attacks.

In [29], Sepúlveda *et al.* have proposed a secure 3D-NoC protected against software attacks by using dynamic, distributed, and agile SZs and firewalls. All components inside the same SZ are considered trusted, and therefore, transactions inside the SZ are unencrypted. Based on the paper, an elastic SZ is able to change its shape according to the security requirements of the mapped applications on the 3D-MPSoC. A reconfiguration and security manager module is defined to reconfigure firewalls of the elastic SZ based on the security policies. The method requires multiple firewalls to implement and update the security policies and reshape the zone. Subsequently, the area, power, and performance overheads of the proposed architecture are 5%, 2%, and 2% with respect to the whole MPSoC chip.

Table 7 summarizes the previous works, SZ parameters, and used security countermeasures.

V. WIRELESS NoCs COUNTERMEASURES

In this section, we review the countermeasures proposed to address wireless NoCs security threats that were introduced in section III-B.

In [106], the author takes advantage of the static nature of the wireless medium to limit physical parameters such as humidity, temperature, and losses that can affect the wireless operation. The major assumption of having a metallic heat sink covering the chip eliminates external spoofing attacks. The proposed countermeasure addresses internal spoofing attacks. This method requires a setup phase to fill in the *Address Conversion Table* on each node. Each node broadcasts a test message to inform other nodes of its unique signal power at this phase. During runtime, the power of each received message is compared with that of the corresponding sender entry in the table. If a mismatch is found, all nodes will receive an alert message asking to ignore any incoming wireless messages until the attacking wireless interface is addressed. An issue could arise from having multiple nodes

at an equal distance, so their received power would be very similar. This is solved by having 4 evaluating modules at the corners of the chip and using trilateration to detect the rogue node.

In [8], the authors tried to leverage their previous paper on spoofing attacks [106] by adding two additional modules to combat DoS and eavesdropping attacks, the so-called Prometheus architecture. The DoS module can handle attacks in contention-free or contention-based traffic; however, the authors have not considered the cases in which an HT accesses the physical layer to launch a jamming attack right after a WI starts transmitting data. In contention-free traffic, if a collision is detected, Prometheus assigns a node to monitor collisions to stop the currently transmitting WI from sending data. Prometheus identifies the attacker ID through the source address (in the header flits) or the power ID. However, the monitoring node selection method is not clear in this study. As for contention-based traffic, the authors have derived a metric called the *Unfairness Ratio* that considers injection throughput, reception throughput, and back-off delay for a given WI. If the ratio passes a configurable threshold, the node is declared rogue, and the OS would turn it off. This threshold needs to be fine-tuned for the specific network that Prometheus would be deployed on.

As for Eavesdropping detection, after comparing multiple relatively lightweight encryption algorithms, the authors chose Py that is only susceptible to linear distinguishing attacks [107]. The authors mentioned that Py would not satisfy latency restrictions when used with high injection rates in the future, but it is a reasonable option in the light to mid traffic.

In [108]–[110] the authors have targeted jamming and eavesdropping attacks from both internal and external sources. The jammer node is assumed to inject packets without permission, i.e., with no valid token. The junk traffic produced by the jammer node causes a high bit error rate in the system (up to 50%) which is much higher than the typical error rates of these systems (not higher than 10^{-5}). The authors used a machine learning classifier to detect jamming errors. The classifier works in conjunction with a burst error control unit (BEU) and a defense unit (DU) to detect/prevent jamming attacks (Figure 18). To protect the chip against possible external eavesdroppers, the authors used a simple data scrambling approach in the form of XORing the flits with the same (periodically changing) key. For internal eavesdropping attacks, though, they have equipped the input port with a low complexity checker unit to check if the WI is passing down any unauthorized data flits. Violating WIs will be shut down by the power management unit to turn off that malicious WI. However, this mechanism would fail to protect the system in case of broadcast packets [108], [109].

In [111] the authors have exploited a similar technique to the one utilized in [108], [109] to counter jamming attacks on a different architecture called Network-in-Package (NiP). More than one multi-core chip are considered to communicate wirelessly in this architecture, while cores in each chip

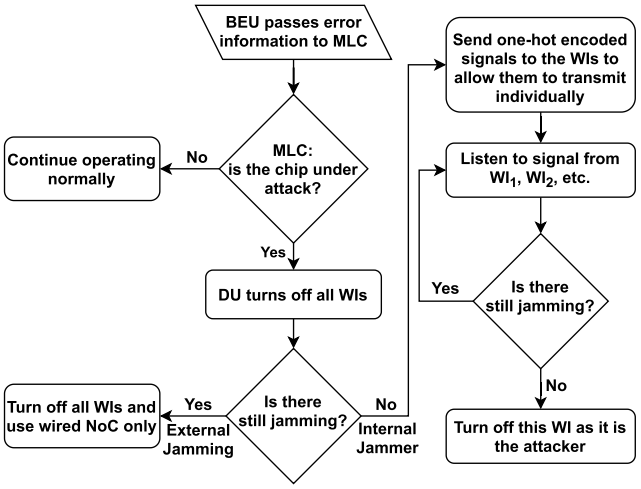


FIGURE 18. Simplified flowchart for jamming detection [108], [109].

are connected using a wired mesh. The authors have assumed a minimum of two wireless interfaces and a maximum of one internal jammer per chip, and only one external attacker is assumed. The attacker can conduct adversarial attacks after reverse-engineering the attack detection unit, then adding minor carefully crafted noise to the attack so that the classifier labels the attack as a normal operation. The authors have used the same approach from [108], [109] to detect internal and external jammers as in Figure 18.

However, a smarter way is adopted to handle external jammers; instead of powering off all wireless interfaces as in [108], [109], CDMA encoder/decoder circuits are deployed. This way, the jamming signals appear as white noise and can be removed easily at any receiver. The CDMA keys are stored in a tamper-proof memory at each interface and are periodically changed to avoid being detected by brute-force search.

In [112], the authors have utilized a distributed channel access mechanism (CAM). Each CAM controller determines the access time needed for its corresponding WI according to the local load level [113]. The header flit is modified to include source and destination WIs' addresses as well as the access time. Initially, each WI broadcasts its needed access time to construct a ranking table for all of the WIs in descending order (Figure 19 shows an example table). Transmission starts according to the table while listening WIs monitor how much access time is spent. If a rouge WI tries to illegitimately hold the channel for a longer time than its previously announced access time, other WIs would send *DoS attack flags* to a *majority voter* unit that eventually decides/disables the violating WI. Spoofing attacks can be launched by modifying the WI source address to point to the WI right before the currently transmitting WI. In this case, when this node finishes transmission, other WIs will deduce that the violating WI is the one now allowed to use the channel. To combat this, any WI that uses the channel for an amount of time greater than a threshold

Rank	WI	Access Time
1	WI_A	T_A
2	WI_X	T_X
3	WI_B	T_B
4	WI_Y	T_Y

Dos Condition: WI_A takes $T_A + 1$

Spoofing Condition: WI_A takes $T_A + T_B + T_Y + 1$, WI_X raises the flag

$$T_A > T_X > T_B > T_Y$$

FIGURE 19. An example of the ranking table used in [112] where WI_A is the attacker.

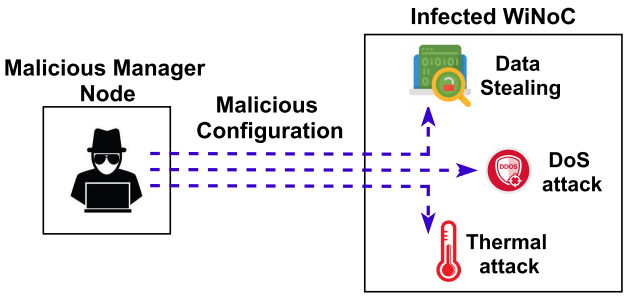


FIGURE 20. Introducing different attacks through malicious configuration from the adversary manager node.

(maximum possible idle time of the channel) is assumed as a spoofing WI. The next node in the ranking table raises a spoofing attack flag that results in disabling the violating WI.

Despite the minimal reported overhead, sending the WI source address and the access time seems redundant as all other WIs already have this information. Additionally, the spoofing detection could be simply based on comparing the incoming source address with the legitimate source address.

The malicious configuration attacks discussed in Section III-B are addressed in [114]. The authors have used reconfigurable routers, NIs, and wireless hubs. The router has a register called *threshold configuration register* (TCR) to store the critical packet size parameter. The router may or may not direct the incoming packets to the corresponding wireless hub based on TCR's value and packet size. Each wireless hub has two configuration registers; *token start count register* (TSCR) and *token end count register* (TECR), which are configured at the start of the operation by the *Configuration Module*.

The used threat model in this paper is sending tampered configuration data by a malicious manager node to cause multiple security threats (see Figure 20). Firstly, the attacker can set the threshold in the TCR register to its maximum value so that all packets are routed through the wired network. On the other hand, setting it to its minimum value would cause all packets to be routed wirelessly, creating a bottleneck at the wireless hubs. Another possible attack is to configure TSCR and TECR registers to make the token duration zero causing DoS. Setting TSCR and TECR to their maximum token time (the entire time of the running application) leads to overutilization of the hubs as they will transmit messages non-stop. The authors have studied the combination of two attacks together and found that combined overutilization attacks may lead to thermal threshold violations as well. Finally, the attacker can also apply spoofing attacks by having security keys sent over the wireless network instead of the wired network. To do this, the key packets are maliciously declared as broadcast packets to be sent over the wireless network; an eavesdropper then can simply compromise these messages.

The authors implemented countermeasures for the previously mentioned attacks. Firstly, they adopted a distance check in the router to decide whether to send packets wirelessly or on the wired mesh. Routers compare the wired and wireless distances and pick the shortest. The authors have also implemented a token wait counter (TWC) in each wireless hub which counts transmissions without legitimate token as a sign of DoS attack. If the most significant bit is 1, the hub is detected to be under DoS attack because TWC counted for many clock cycles without a token. On the other hand, they have implemented two packet transmission counters, one at the transmitter (called PTC) and another at the receiver (called PRC) sections of the hub, to detect the disruptive token passing attack. Whenever a hub transmits or receives, it respectively increments the PTC or PRC by one. If the PTC overflows while PRC is still zero, the hub is detected to be under attack and will be switched off.

In [42], the authors have addressed the security of cache coherence messages in MESI-based ECONO protocol [116]. The system has 64 tiles, including 16 shared L3 banks and 48 cores, such that each processing core can communicate concurrently with all of the shared banks using 16 different frequency channels. The proposed system aims at preventing the flooding replay attacks, modification, and impersonation for cache coherency messages.

Each tile is equipped with a counter register per each L3 bank. The counters count the number of messages sent toward each L3 bank. To access L3 banks, corresponding key and counter value are sent on the secure wired mesh to the L3 bank to fetch any missing data stored in an L2 bank. To ensure authenticity, the L3 bank checks the key and compares it with the associated key of the sender in its database. To guarantee freshness, the sender/receiver uses the counter to check if it matches the currently expected message count. This helps to prevent replay attacks at which the attacker forwards old messages as new messages. To ensure integrity, the counter and key are used to perform hashing on the message, and the resulting hash would be concatenated to the message. If any modification occurs, the receiver will generate a different hash value and recognize the attack. Despite the success of the countermeasures in dealing with such attacks, the presented scheme can only detect false messages. However, the adversary node can still use such weakness to flood the network with unwanted messages.

TABLE 8. List of WiNoC references and their adversary model, attack type, countermeasure and penalty.

Reference	Adversary	Attack Type	Countermeasure	Attack Target	Attack Penalty	Countermeasure Penalty
Lebiednik et al. [8]	IP, WI	Spoofing, DoS, Eavesdropping	Power profiling, Classification, Encryption	Availability, Confidentiality	Performance, Information leak	Power, Area, Performance
Garcia et al. [42]	IP	Modification, Spoofing	Message counters, Hashing, Unique Key per L2 core	Confidentiality, Integrity	Not specified	Performance, Information leak
Lebiednik et al. [106]	IP, WI	Spoofing	Power profiling	Availability, Confidentiality	Performance, Information leak	Power, Area
Vashist et al. [108], [109]	WI	DoS (Jamming), Eavesdropping	Machine learning classifier, Data Scrambling	Availability, Confidentiality	Performance, Information leak	Power, Area
Ahmed et al. [111]	WI	DoS (Jamming)	Machine learning classifier	Availability	Performance	Power, Area, Performance
Rout et al. [112]	WI	DoS, Spoofing	Access time monitoring	Availability	Performance	Power, Area
Biswas et al. [114]	IP	DoS, Thermal, Eavesdropping	Proactive distance measure, Token time counter, message counters	Availability, Confidentiality	Performance, High temperature	Performance, area, power
Ganguly et al. [115]	Not specified	DoS	Offline creation of less sensitive links to DoS attack	Availability	Performance	Not specified

The authors of [115] have recommended the use of the small-world NoC (smNoC) [117] because of its resilience to DoS attacks [118]. The links of the NoC are created based on inverse law distribution where the distance between two cores and their communication rate determine the probability of creating a link between them. A wireless NoC with smNoC backbone has wireless shortcuts not wired as in the regular ones. The creation process needs an optimization framework and that was the key for the authors to tailor this process to get to the final smNoC so-called secure smNoC (ssmNoC). The authors have considered a situation when a hardware Trojan in a core can trigger excessive injections of garbage traffic into the network. Following the initial network setup, simulated annealing (SA) heuristics is used to minimize the impact-spread of DoS attacks by deciding where to have NoC channels. The authors have defined a metric, which reflects the impact-spread of DoS over the NoC. The average hop-distance between the switches in the ssmNoC, μ indicates the interconnectedness of the NoC. The metric to be optimized should therefore decrease μ as the DoS spreads over the network via multiple levels, l, of victim nodes. Consequently, the optimization metric, ρ, for the SA algorithm is given in Eq. 1

$$\rho = \frac{\Delta\mu}{l} \tag{1}$$

The algorithm will produce a new network every iteration, and when the error in ρ becomes approximately zero, the algorithm stops, and the resulted network is the one with minimal performance impact in the presence of a DoS attack. The authors have shown a gain in performance and reduced packet energy. While seeming creative, the solution is not applicable for general MPSoCs since it is tightly application-dependent.

VI. 3D NoCs COUNTERMEASURES

Likewise wireless NoCs, the existence of more than one type of communication channel in a 3D NoC raises specific security issues. In a 3D NoC, traditional wired channels are used for horizontal or intra-layer communications, while vertical TSV channels are utilized for inter-layer communications due to their higher bandwidth. In the rest of this section, we review the countermeasures specifically proposed for 3D NoCs.

In [29], [119], security vulnerabilities in 3D NoCs were first introduced. The proposed attack prevention strategy is based on applying a different security policy at the software level. The authors have added new information to packets to determine if the packet is sent from a trusted IP or not. The added information includes the size of the payload, the deadline for the transaction to be performed, the signatures of the routers and bus arbiters used by the packet on the path between the initiator and the destination, and an ID that counts the number of transactions between an initiator/destination pair. As the initiator/destination nodes only know the actual values, different attacks can be addressed. For example, in a replay attack, if the attacker resends the intercepted packet, the firewall hardware would detect it since the received ID does not have the anticipated value. If the attacker could guess the ID correctly, another way to detect the replay attack is to look at the deadline where the replay attack messages must be reached before the transaction deadline. The authors have also proposed a reconfigurable hardware security firewall that can be updated when new applications are loaded.

Although addressing some of the security challenges of 3D NoCs, the proposed method may not be practical due to its overheads. The network delay has increased about $\approx 3\times$ of its baseline value at the saturation injection rate. This is mainly because the proposed method is an application-level method that needs support from the local cores. As reported in the paper, the area overhead ranges between 0.2% to 1.2% and the power overhead from 2.5% to 10.4%.

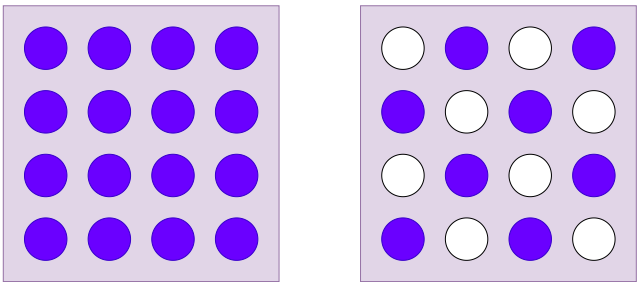
In [120] and its extended study [121], Sepúlveda *et al.* have introduced the special attacks that can take place in a 3D NoC with TSVs. 3D chip designers always try to pack wires of a TSV channel in a smaller cross-sectional area to save silicon. However, the closer proximity between

TABLE 9. List of 3D NoC references and their adversary model, attack type, countermeasure and penalty.

Reference	Adversary	Attack Type	Countermeasure	Attack Target	Attack Penalty	Countermeasure Penalty
Sepulveda et al. [29]	IP	Not specified	Proactive reconfigurable firewall	General attack	Not specified	Performance, power, area
Patooghy et al. [47]	IP	Crosstalk	Not specified	Availability	Performance loss, Deadlock	–
Das et al. [48]	IP	Over-utilization of TSVs	Not specified	Availability	Shorter chip lifetime	–
Sepulveda et al. [119]	IP	Not specified	Proactive reconfigurable firewall	General attack	Not specified	Performance, power, area
Sepulveda et al. [120]	IP	Crosstalk, Lifetime	Proactive reconfigurable firewall + Interleaved TSVs	Availability	Performance loss, Deadlock	Performance, power, area
Sepulveda et al. [121]	IP	Crosstalk, Lifetime	Proactive reconfigurable firewall + Interleaved TSVs	Availability	Performance loss, Deadlock	Performance, power, area

TABLE 10. Analogy between the biology immune system and the security in 3D NoCs [120].

Immune system feature	Analogous security feature
Antigen	Packets Attack
Antibody	Countermeasure
Recognizers	Attack detectors
Lymphocyte for antibody mutation	Applying new security rules for new loaded applications



(a) Regular TSVs. (b) Interleaved TSVs.

FIGURE 21. In (a) regular TSVs are used for carrying secure packets while interleaved TSVs in (b) (white TSVs do not carry data) are used to carry malicious packets from untrusted IPs.

wires of TSVs makes stronger parasitic capacitances of TSV wires and in turn stronger electrical coupling. The coupling alters information passing the centered TSV wires, so-called the victim TSV by its surrounding TSVs (aggressor TSVs). Malicious software executed on the 3D-MPSoC can exploit this natural phenomenon to manipulate the data on some TSV wires. The attacks can be in the form of i) modifying data on the victim-TSV through the coupling effect of adjacent aggressor TSVs [122], ii) reduce the lifetime of the victim TSV by increasing the chance of stress cracks [123], [124] or through electromigration effects [23], and iii) corrupting transmissions by delaying or speeding up signal transitions [123].

The authors have built what is called a ''3D-LeukoNoC'' based on the analogy between a security system in a 3D NoC and the biological immune system in the human body. A biological human immune system can identify attackers (antigens) and produce a suitable defense (antibodies). As shown in Table 10, 3D-LeukoNoC emulates the same behavior of the immune system to defend the 3D NoC system. The authors have introduced two hardware modifications to defend against the attacks mentioned above. 1) A Recognizer that inspects the source address of each packet and retrieves the appropriate security policy for the source IP. 2) A Lymphocyte responsible for updating the security policy (antibody generation) when a new application is mapped. The Lymphocyte also decides whether to use normal or interleaved TSV communication in forwarding the application data over the network. As shown in Figure 21, interleaving results in less crosstalk due to the larger distances between neighboring TSVs so that secure communications are mapped to interleaved TSVs.

Despite the reasonable overhead of this approach, 3D-LeukoNoC cannot detect attacks, but rather relies on assumptions about applications (being malicious or not) that might be violated by the application at run-time.

The authors in [47] have proposed a novel crosstalk attack that relies on introducing as much crosstalk as possible between TSVs. As Figure 22 shows possible transition patterns between TSVs, the higher the capacitance of the pattern, the more severe effect it has on the victim TSV. With the goal of increasing the probability of crosstalk on victim wires, the attacker application sends a stream of ''0101..01'' to its receiver side. This pattern, so-called *bait flits* will have the four aggressors always carrying a signal different than the victim wire. As the bait flits pre-charge TSV wires, they have a high probability of corrupting the normal flits being transmitted on the TSVs right after them. Induced bit flips by the bait flits can lead to a variety of issues at the network level, such as (1) Packet Loss: changing a header flit into a data flit, making the routers unable to route it, (2) Packet Mis-delivery: modifying the address field of a header flit, leading the infected packet to a random receiver, (3) Data Error: messing up data in a data flit and/or (4) Fake Packets: changing a data flit into a header flit, which would result in two incomplete random packets. If the attacker sends enough bait flit, the attack could cause a global deadlock when a sufficient number of lost or fake packets are abandoned in the buffers of the routers. There are several works targeting

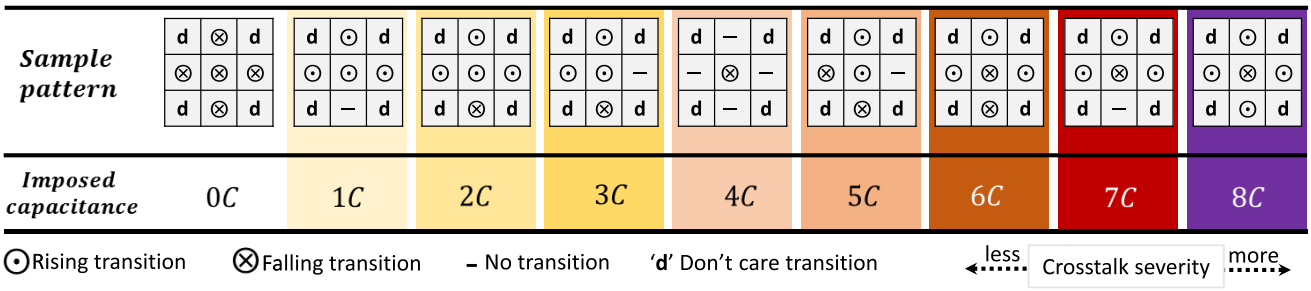


FIGURE 22. Examples of the different corsstalk patterns ranging from 0C (lowest) to 8C (highest) [47].

the reduction of crosstalk. Some utilize design layout changes [125] while others rely on encoding the transmitted data [126]. The author demonstrated that even encoding techniques could not fully prevent a successful attack. The author tested the effectiveness of the attack against the ITCM encoding technique [127], as it is a recent work specifically proposed for crosstalk tolerance. The results showed that ITCM could reduce the occurrence of more aggressive transition patterns while increasing less aggressive ones. Being a completely software-based attack, the attacker does not need any special access to the hardware of the NoC. This makes it easy to deploy.

The authors in [48] have analyzed the over-utilization of TSVs called a lifetime attack. The manufacturer of the chip can abuse its knowledge of the chip to launch such an attack through software updates to accelerate the aging of the devices to push customers for upgrading to newer models [128]. Excess use of a TSV increases the internal resistance that degrades TSV's performance and makes it unusable for fast communications. The attack generates excess traffic passing through the TSVs to accelerate their degradation. This would bottleneck the alternative TSVs and haste their lifetime reduction (cascade effect [48]). The authors have used a small world 3D NoC (sw3DNoC) and compared it with mesh networks in terms of reliability and performance [129].

The authors have explored three possible attacks that the manufacturer can launch with differing levels of severity: (1) Uniform Random Attack: All routers in the NoC are subject to a random increase in their message injection rates. This would result in uniform wear out across all routers, so this attack has the least effect on the MTTF, (2) Critical Region Attack: the most contested region of the chip is targeted until it is worn out. The traffic increase in the abutting channels expedites their wear out in turn, and (3) Critical Vertical Links Attack: the most contested TSVs are targeted. Depending on the load, the critical TSVs are spread over a broader region than the critical region. The attacker needs to inject less additional traffic to reach similar MTTF values to the previous attacks. The results show that the NoC lifetime can be reduced by 11%-26% by injecting only 3%-10% additional traffic.

VII. FUTURE DIRECTIONS AND CONCLUSIONS

In this review paper, we reviewed most of the papers related to the security of NoC-based MPSoCs published since 2015. The main goal of the paper is to give insight to the researchers of the field to easier assess/compare the state of the art of secure MPSoCs. This review paper also sheds light on the areas that have not been addressed. In the rest of this section, we review the unaddressed topics in the design of secure MPSoCs along with our proposals to address them. We believe this section would serve as a useful roadmap to shape future research in the design of secure MPSoCs.

A. OVERLAPPING SECURE ZONES

Among the research gaps that we have detected in the secure MPSoCs, *Overlapping Secure Zones* is one. Many papers have addressed various aspects of secure zones, including zone formation, working with logical zones, and key management over the zone members. However, the need for having overlapping secure zones is not addressed yet. The major research question with overlapping zones is making sure that data will not leak from a zone to another at the routers that belong to multiple zones. Considering shared hardware resources of NoC routers, e.g., key memory to store encryption keys, encryption/decryption modules, and flit buffers with access to the data belong to different zones, it would not be easy to fulfill the security requirements of overlapping zones. In this context, new threat models may be defined to conduct malicious activities, e.g., tampering, spoofing, side-channel, etc. We believe that secure overlapping zones can only be achieved through isolation at the hardware level such that data of different zones can be kept separate. Wrapping the mentioned hardware components of NoC routers and having a hardware root of trust at the router can be utilized to address this security concern.

B. SECURING EMERGING NOC ARCHITECTURES

Emerging architectures for MPSoCs such as *Router-less NoCs* demand extensive research to address these architectures. For example, in a router-less NoC, communications are carried out through cascaded links that form chip-wide circuit-switched communication loops. Every node that

receives a packet either ejects the packet or forwards it to the next node on the loop. Such a simplified mechanism eliminates NoC routers that consume/take noticeable energy/area in MPSoCs. Several of such loops are required to keep the whole network connected, i.e., some nodes belong to multiple loops to allow inter-loop data exchanges. Although performance-wise efficient, router-less NoCs are easy targets for attackers as a malicious node can easily access all loop packets due to lack of path diversity.

C. ADOPTING ANONYMOUS ROUTING

According to the pipelined implementation of NoC routers, upon receiving a new packet, the router starts the route-computation stage to compute the appropriate outgoing link for the packets. This information is required at virtual-channel allocation and cross-bar allocation stages. To keep the router's critical path as short as possible, almost all of the previous works in the field of NoC security assume that the header flit of packets are unencrypted (sent as plaintext). Although this choice enables NoC routers to perform the routing computation in less than a cycle, it opens doors to many security attacks such as spoofing, side-channel, and packet misrouting. This is a serious question that requires the research community's attention. The research challenge here is to secure the header flits while not imposing tens of cycles of delay to the routing process. In fact, the straightforward application of encryption algorithms to scramble the header flits is not a feasible solution as it will impose high performance/energy overhead to the NoC. We believe that a type of anonymous routing in which the header is obfuscated or does not carry exact destination information might be a potential solution. For example, source routing algorithms in which the path is computed at the source node and a scrambled version of the path embedded into the packet might be an option for further investigations.

D. ENCRYPTION VERSUS DATA SCRAMBLING

There are obvious trade-offs between the levels of security versus the implementation costs. It is well known that encryption is more secure than data scrambling but imposes much more performance, area, and power overheads, especially for sophisticated encryption algorithms like AES. On the other hand, the main challenge for scrambling algorithms is to provide true random numbers. As we find some NoC security works adopted encryption techniques [8], [106] for security purposes, others utilized data scrambling [108], [109] to manage the overheads. To the best of the authors' knowledge, there is no solid study with clear judge/justification so far to help MPSoC designers choose one technique over the other. In conducting such a study, researchers should consider a wide variety of attacking models to make it sound.

E. PUF-BASED AUTHENTICATION IN NoCs

An attacker in an MITM attack takes control of the connection between two parties and makes them believe that they are communicating with each other. It can intercept packets from two communicating entities, modify the content and resend them (as shown in 6). One particular threat by a MITM is to send his public keys to a requesting party and decrypt secret messages. In the computer security domain, a certification authority (CA) is used to verify the ownership of public keys by signing them with CA's private keys. CA's public key is then used to verify that the public keys are genuine and authenticated. This task is more complicated in tiny resource-constrained devices [130] such as NoC IPs. One potential future direction is leveraging PUF-based (physically unclonable function) authentication schemes (details explained in section IV-C). PUFs advantages in NoC context were limited to bit permutation [33], and random arbitration [67]. Despite the advantages, PUFs should be used with care since their response reliability can be impacted by voltage and temperature variations [78].

F. SECURING WIRELESS NOCS AGAINST CONCURRENT JAMMING ATTACKS

There are also research gaps in wireless NoCs as well. For example, *multiple concurrent internal jamming attack* is one to pinpoint. The pioneering work of [108], [109], [111] assumes only a single jammer in the chip that could be easily detected as pointed out in Section V. However, the approach presented would eventually fail if there are multiple concurrent jammers. In that case, the detection of jamming attacks is still possible, but determining the source of the attack with the presented approach would fail.

G. THERMAL ATTACKS IN 3D NoCs

The work of [131] provides a temperature distribution comparison among 2D, 2-layer 3D and 4-layer 3D NoCs. That work indicates that the temperature gradient increases with the increasing number of 3D layers because only one layer is attached to the heat-sink. The standard deviation of temperature for the 4-layer 3D chip is approximately 40 times higher than that of the 1-layer 2D chip. A Trojan can use that weakness and flood the top-most layer with fake and/or legitimate traffic to increase the probability of exceeding the temperature threshold to induce thermal throttling. Thermal throttling usually comes in the form of reducing frequency which degrades performance. In spite of the severity of such an attack, no countermeasures were introduced to either detect or eliminate this potential security threat.

H. CROSSTALK ATTACKS COUNTERMEASURES IN 3D NoCs

As pointed out in Section VI, crosstalk attacks may lead to a deadlock and yet are very simple to implement by a malicious IP. Regardless of these facts, the authors of [29], [47], [121] did not provide a framework to detect such attacks. While [47] studied and analyzed the attack consequences, the other works proactively make cautious assumptions about the running software and send the packets on interleaved TSVs if any software is assumed malicious. However, this software might not be adversarial, and in that case, valuable resources are wasted due to false assumptions.

REFERENCES

[1] W. J. Dally and B. P. Towles, *Principles and Practices of Interconnection Networks*. Amsterdam, The Netherlands: Elsevier, 2004.

[2] S. Bell *et al.*, "TILE64-processor: A 64-core SoC with mesh interconnect," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2008, pp. 88–598.

[3] B. D. de Dinechin, P. G. de Massas, G. Lager, C. Léger, B. Orgogozo, J. Reybert, and T. Strudel, "A distributed run-time environment for the Kalray MPPA-256 integrated manycore processor," *Procedia Comput. Sci.*, vol. 18, pp. 1654–1663, Jan. 2013.

[4] J.-J. Lecler and G. Baillieu, "Application driven network-on-chip architecture exploration & refinement for a complex SoC," *Design Autom. Embedded Syst.*, vol. 15, no. 2, pp. 133–158, 2011.

[5] H. G. Lee, N. Chang, U. Y. Ogras, and R. Marculescu, "On-chip communication architecture exploration: A quantitative evaluation of point-to-point, bus, and network-on-chip approaches," *ACM Trans. Design Autom. Electron. Syst.*, vol. 12, no. 3, pp. 1–20, Aug. 2007.

[6] A. E. Zonouz, M. Seyrafi, A. Asad, M. Soryani, M. Fathy, and R. Berangi, "A fault tolerant NoC architecture for reliability improvement and latency reduction," in *Proc. 12th Euromicro Conf. Digit. Syst. Design, Archit., Methods Tools*, Aug. 2009, pp. 473–480.

[7] M. A. J. Sethi, F. A. Hussin, and N. H. Hamid, "Ss," *Sci. Int.*, vol. 5, no. 27, pp. 4133–4144, 2015.

[8] B. Lebiednik, S. Abadal, H. Kwon, and T. Krishna, "Architecting a secure wireless network-on-chip," in *Proc. 12th IEEE/ACM Int. Symp. Netw.-Chip (NOCS)*, Oct. 2018, pp. 1–8.

[9] S. Charles, Y. Lyu, and P. Mishra, "Real-time detection and localization of DoS attacks in NoC based SoCs," in *Proc. Design, Automat. Test Eur. Conf. Exhib. (DATE)*, Mar. 2019, pp. 1160–1165.

[10] S. Murali and G. De Micheli, "Bandwidth-constrained mapping of cores onto NoC architectures," in *Proc. Design, Automat. Test Eur. Conf. Exhib.*, vol. 2, 2004, pp. 896–901.

[11] D. M. Ancajas, K. Chakraborty, and S. Roy, "Fort-NoCs: Mitigating the threat of a compromised NoC," in *Proc. 51st Annu. Design Automat. Conf. Design Automat. Conf. (DAC)*, 2014, pp. 1–6.

[12] M. Tehranipoor and F. Koushanfar, "A survey of hardware Trojan taxonomy and detection," *IEEE Des. Test. Comput.*, vol. 27, no. 1, pp. 10–25, Jan. 2010.

[13] R. S. Chakraborty, S. Narasimhan, and S. Bhunia, "Hardware Trojan: Threats and emerging solutions," in *Proc. IEEE Int. High Level Design Validation Test Workshop*, Nov. 2009, pp. 166–171.

[14] J. S. Rajesh, D. M. Ancajas, K. Chakraborty, and S. Roy, "Runtime detection of a bandwidth denial attack from a rogue network-on-chip," in *Proc. 9th Int. Symp. Netw.-Chip*, Sep. 2015, pp. 1–8.

[15] J. Ramachandran, *Designing Security Architecture Solutions*. Hoboken, NJ, USA: Wiley, 2002.

[16] Z. Huang, Q. Wang, Y. Chen, and X. Jiang, "A survey on machine learning against hardware Trojan attacks: Recent advances and challenges," *IEEE Access*, vol. 8, pp. 10796–10826, 2020.

[17] L. Daoud, "Secure network-on-chip architectures for MPSoC: Overview and challenges," in *Proc. IEEE 61st Int. Midwest Symp. Circuits Syst. (MWSCAS)*, Aug. 2018, pp. 542–543.

[18] L. Fiorin, C. Silvano, and M. Sami, "Security aspects in networks-on-chips: Overview and proposals for secure implementations," in *Proc. 10th Euromicro Conf. Digit. Syst. Design Archit., Methods Tools (DSD)*, Aug. 2007, pp. 539–542.

[19] A. B. Achballah, S. B. Othman, and S. B. Saoud, "Problems and challenges of emerging technology networks-on-chip: A review," *Microprocessors Microsyst.*, vol. 53, pp. 1–20, Aug. 2017.

[20] J.-P. Diguet, S. Evain, R. Vaslin, G. Gogniat, and E. Juin, "NoC-centric security of reconfigurable SoC," in *Proc. 1st Int. Symp. Netw.-Chip (NOCS)*, May 2007, pp. 223–232.

[21] J. Sepúlveda, A. Zankl, D. Flórez, and G. Sigl, "Towards protected MPSoC communication for information protection against a malicious NoC," *Procedia Comput. Sci.*, vol. 108, pp. 1103–1112, Jan. 2017.

[22] V. Pavlidis and E. Friedman, "Physical analysis of NoC topologies for 3-D integrated systems," Springer, New York, NY, USA, Tech. Rep., Nov. 2011.

[23] A. Sheibanyrad, F. Pétrot, and A. Jantsch, *3D Integration for NoC-Based SoC Architectures* (Integrated Circuits and Systems), 1st ed. New York, NY, USA: Springer-Verlag, 2011.

[24] E. Taheri, M. Isakov, A. Patooghy, and M. A. Kinsy, "Addressing a new class of reliability threats in 3-D network-on-chips," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 7, pp. 1358–1371, Jul. 2020.

[25] C. Wang, W.-H. Hu, and N. Bagherzadeh, "A wireless network-on-chip design for multicore platforms," in *Proc. 19th Int. Euromicro Conf. Parallel, Distrib. Netw.-Based Process.*, Feb. 2011, pp. 409–416.

[26] A. K. Kodi, M. A. I. Sikder, D. DiTomaso, S. Kaya, S. Laha, D. Matolak, and W. Rayess, "Kilo-core wireless network-on-chips (NoCs) architectures," in *Proc. NANOCOM*. New York, NY, USA: Association for Computing Machinery, 2015, pp. 1–6, doi: 10.1145/2800795.2800797.

[27] S. Deb, A. Ganguly, K. Chang, P. Pande, B. Beizer, and D. Heo, "Enhancing performance of network-on-chip architectures with millimeter-wave wireless interconnects," in *Proc. 21st IEEE Int. Conf. Appl.-Specific Syst., Archit. Processors (ASAP)*, Jul. 2010, pp. 73–80.

[28] S. Abadal, A. Mestres, J. Torrellas, E. Alarcon, and A. Cabellos-Aparicio, "Medium access control in wireless network-on-chip: A context analysis," *IEEE Commun. Mag.*, vol. 56, no. 6, pp. 172–178, Jun. 2018.

[29] J. Sepúlveda, G. Gogniat, D. Florez, J.-P. Diguet, C. Zeferino, and M. Strum, "Elastic security zones for NoC-based 3D-MPSoCs," in *Proc. 21st IEEE Int. Conf. Electron., Circuits Syst. (ICECS)*, Dec. 2014, pp. 506–509.

[30] R. Fernandes, C. Marcon, R. Cataldo, J. Silveira, G. Sigl, and J. Sepúlveda, "A security aware routing approach for NoC-based MPSoCs," in *Proc. 29th Symp. Integr. Circuits Syst. Design (SBCCI)*, Aug. 2016, pp. 1–6.

[31] S. M. Sebt, A. Patooghy, H. Beitollahi, and M. Kinsy, "Circuit enclaves susceptible to hardware Trojans insertion at gate-level designs," *IET Comput. Digit. Techn.*, vol. 12, no. 6, pp. 251–257, Nov. 2018.

[32] K. Madden, J. Harkin, L. McDaid, and C. Nugent, "Adding security to networks-on-chip using neural networks," in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, Nov. 2018, pp. 1299–1306.

[33] J. Frey and Q. Yu, "A hardened network-on-chip design using runtime hardware Trojan mitigation methods," *Integration*, vol. 56, pp. 15–31, Jan. 2017.

[34] K. Wang, H. Zheng, and A. Louri, "TSA-NoC: Learning-based threat detection and mitigation for secure network-on-chip architecture," *IEEE Micro*, vol. 40, no. 5, pp. 56–63, Sep. 2020.

[35] C. Reinbrecht, A. Susin, L. Bossuet, and J. Sepúlveda, "Gossip NoC-avoiding timing side-channel attacks through traffic management," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI (ISVLSI)*, Jul. 2016, pp. 601–606.

[36] L. Daoud and N. Rafla, "Routing aware and runtime detection for infected network-on-chip routers," in *Proc. IEEE 61st Int. Midwest Symp. Circuits Syst. (MWSCAS)*, Aug. 2018, pp. 775–778.

[37] L. Daoud and N. Rafla, "Analysis of black hole router attack in network-on-chip," in *Proc. IEEE 62nd Int. Midwest Symp. Circuits Syst. (MWSCAS)*, Aug. 2019, pp. 69–72.

[38] V. Y. Raparti and S. Pasricha, "Lightweight mitigation of hardware Trojan attacks in NoC-based manycore computing," in *Proc. 56th Annu. Design Automat. Conf.*, Jun. 2019, pp. 1–6.

[39] L. L. Caimi, V. Fochi, E. Wachter, D. Munhoz, and F. G. Moraes, "Activation of secure zones in many-core systems with dynamic rerouting," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2017, pp. 1–4.

[40] M. A. Kinsy, S. Khadka, M. Isakov, and A. Farrukh, "Hermes: Secure heterogeneous multicore architecture design," in *Proc. IEEE Int. Symp. Hardw. Oriented Secur. Trust (HOST)*, May 2017, pp. 14–20.

[41] M. S. Obaidat, I. Woungang, S. K. Dhurandher, and V. Koo, "A cryptography-based protocol against packet dropping and message tampering attacks on mobile ad hoc networks," *Secur. Commun. Netw.*, vol. 7, no. 2, pp. 376–384, Feb. 2014.

[42] F. P. García and J. L. Abellán, "Secure communications in wireless network-on-chips," in *Proc. 2nd Int. Workshop Adv. Interconnect Solutions Technol. Emerg. Comput. Syst. (AISTECS)*. New York, NY, USA: Association for Computing Machinery, 2017, pp. 27–32, doi: 10.1145/3073763.3073768.

[43] N. Hoque, D. K. Bhattacharyya, and J. K. Kalita, "Botnet in DDoS attacks: Trends and challenges," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2242–2270, 4th Quart., 2015.

[44] M. Hussain, A. Malekpour, H. Guo, and S. Parameswaran, "EETD: An energy efficient design for runtime hardware Trojan detection in untrusted network-on-chip," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI (ISVLSI)*, Jul. 2018, pp. 345–350.

[45] S. Charles and P. Mishra, "Reconfigurable network-on-chip security architecture," *ACM Trans. Design Autom. Electron. Syst.*, vol. 25, no. 6, pp. 1–25, 2020.

[46] Q. Shafi, "Cyber physical systems security: A brief survey," in *Proc. 12th Int. Conf. Comput. Sci. Appl.*, Jun. 2012, pp. 146–150.

[47] A. Patooghy, M. F. Torkaman, and M. Elahi, "Your hardware is all wired up! Attacking network-on-chips via crosstalk channel," in *Proc. 12th Int. Workshop Netw. Chip Archit. (NoCArc)*. New York, NY, USA: Association for Computing Machinery, 2019, pp. 1–6, doi: 10.1145/3356045.3360716.
[48] S. Das, K. Basu, J. R. Doppa, P. P. Pande, R. Karri, and K. Chakrabarty, "Abetting planned obsolescence by aging 3D networks-on-chip," in *Proc. 12th IEEE/ACM Int. Symp. Netw.-Chip (NOCS)*, Oct. 2018, pp. 1–8.
[49] B. Oliveira, R. Reusch, H. Medina, and F. Moraes, "Evaluating the cost to cipher the NoC communication," in *Proc. IEEE 9th Latin Amer. Symp. Circuits Syst. (LASCAS)*, Feb. 2018, pp. 1–4.
[50] S. Charles and P. Mishra, "Securing network-on-chip using incremental cryptography," in *Proc. ISVLSI*, 2020, pp. 168–175.
[51] Y. Wang and G. E. Suh, "Efficient timing channel protection for on-chip networks," in *Proc. IEEE/ACM 6th Int. Symp. Netw.-Chip*, May 2012, pp. 142–151.
[52] L. S. Indrusiak, J. Harbin, and M. J. Sepúlveda, "Side-channel attack resilience through route randomisation in secure real-time networks-on-chip," in *Proc. 12th Int. Symp. Reconfigurable Commun.-Centric Syst.-Chip (ReCoSoC)*, Jul. 2017, pp. 1–8.
[53] J. Daemen and V. Rijmen, "Aes proposal: Rijndael," Tech. Rep., 1999.
[54] D. Coppersmith, "The data encryption standard (DES) and its strength against attacks," *IBM J. Res. Develop.*, vol. 38, no. 3, pp. 243–250, May 1994.
[55] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978.
[56] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theory*, vol. IT-22, no. 6, pp. 644–654, Nov. 1976.
[57] S. Charles, M. Logan, and P. Mishra, "Lightweight anonymous routing in NoC based SoCs," in *Proc. Design, Automat. Test Eur. Conf. Exhib. (DATE)*. San Jose, CA, USA: EDA Consortium, Mar. 2020, pp. 334–337.
[58] A. C. Sant'Ana, H. M. Medina, K. B. Fiorentin, and F. G. Moraes, "Lightweight security mechanisms for MPSoCs," in *Proc. 32nd Symp. Integr. Circuits Syst. Design (SBCCI)*, 2019, pp. 1–6.
[59] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "The SIMON and SPECK lightweight block ciphers," in *Proc. 52nd Annu. Design Automat. Conf.*, Jun. 2015, pp. 1–6.
[60] D. Engels, X. Fan, G. Gong, H. Hu, and E. M. Smith, "Ultra-lightweight cryptography for low-cost RFID tags: Hummingbird algorithm and protocol," Centre Appl. Cryptograph. Res., Tech. Rep., 2009, vol. 29, no. 1.
[61] K. Sajeesh and H. K. Kapoor, "An authenticated encryption based security framework for NoC architectures," in *Proc. Int. Symp. Electron. Syst. Design*, Dec. 2011, pp. 134–139.
[62] D. McGrew and J. Viega, "The galois/counter mode of operation (GCM)," NIST, Gaithersburg, MD, USA, Tech. Rep., 2005.
[63] R. Gudla and K. Stevens, "Design and implementation of clocked OCP interfaces between IP cores and on-chip network fabric," Ph.D. dissertation, Dept. Elect. Comput. Eng., Univ. Utah, Salt Lake City, UT, USA, 2011.
[64] C. Reinbrecht, A. Susin, L. Bossuet, G. Sigl, and J. Sepúlveda, "Side channel attack on NoC-based MPSoCs are practical: NoC prime+probe attack," in *Proc. 29th Symp. Integr. Circuits Syst. Design (SBCCI)*, Aug. 2016, pp. 1–6.
[65] H. M. Wassel, Y. Gao, J. K. Oberg, T. Huffmire, R. Kastner, F. T. Chong, and T. Sherwood, "SurfNoC: A low latency and provably non-interfering approach to secure networks-on-chip," *ACM SIGARCH Comput. Archit. News*, vol. 41, pp. 583–594, Jun. 2013.
[66] J. Sepúlveda, D. Flórez, M. Soeken, J. Diguet, and G. Gogniat, "Dynamic NoC buffer allocation for MPSoC timing side channel attack protection," in *Proc. IEEE 7th Latin Amer. Symp. Circuits Syst. (LASCAS)*, Feb. 2016, pp. 91–94.
[67] M. J. Sepúlveda, J. P. Diguet, M. Strum, and G. Gogniat, "NoC-based protection for SoC time-driven attacks," *IEEE Embedded Syst. Lett.*, vol. 7, no. 1, pp. 7–10, Mar. 2015.
[68] T. H. Boraten and A. K. Kodi, "Securing NoCs against timing attacks with non-interference based adaptive routing," in *Proc. 12th IEEE/ACM Int. Symp. Netw.-Chip (NOCS)*, Oct. 2018, pp. 1–8.
[69] T. Boraten and A. K. Kodi, "Packet security with path sensitization for NoCs," in *Proc. Design, Automat. Test Eur. Conf. Exhib. (DATE)*, 2016, pp. 1136–1139.
[70] T. Boraten, D. DiTomaso, and A. K. Kodi, "Secure model checkers for network-on-chip (NoC) architectures," in *Proc. 26th Ed. Great Lakes Symp. VLSI*, May 2016, pp. 45–50.
[71] A. Prodromou, A. Panteli, C. Nicopoulos, and Y. Sazeides, "NoCAlert: An on-line and real-time fault detection mechanism for network-on-chip architectures," in *Proc. 45th Annu. IEEE/ACM Int. Symp. Microarchitecture*, Dec. 2012, pp. 60–71.
[72] S. Charles and P. Mishra, "Lightweight and trust-aware routing in NoC-based SoCs," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI (ISVLSI)*, Jul. 2020, pp. 160–167.
[73] J.-P. Aumasson and D. J. Bernstein, "SipHash: A fast short-input PRF," in *Proc. Int. Conf. Cryptol. India*. Berlin, Germany: Springer, 2012, pp. 489–508.
[74] S. Moriam, E. Franz, P. Walther, A. Kumar, T. Strufe, and G. Fettweis, "Protecting communication in many-core systems against active attackers," in *Proc. Great Lakes Symp. VLSI*, May 2018, pp. 45–50.
[75] R. Pappu, B. Recht, J. Taylor, and N. Gershenfeld, "Physical one-way functions," *Science*, vol. 297, no. 5589, pp. 2026–2030, Sep. 2002.
[76] J. Shi, Y. Lu, and J. Zhang, "Approximation attacks on strong PUFs," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 10, pp. 2138–2151, Oct. 2019.
[77] J. W. Lee, D. Lim, B. Gassend, G. E. Suh, M. van Dijk, and S. Devadas, "A technique to build a secret key in integrated circuits for identification and authentication applications," in *Symp. VLSI Circuits Dig. Tech. Papers*, 2004, pp. 176–179.
[78] P. Nagabhushanamgari, V. Sehwag, I. Chakrabarti, and S. Chattopadhyay, "Embedding delay-based physical unclonable functions in networks-on-chip," *IET Circuits, Devices Syst.*, vol. 15, no. 1, pp. 27–41, Jan. 2021.
[79] M. T. Rahman, F. Rahman, D. Forte, and M. Tehranipoor, "An aging-resistant RO-PUF for reliable key generation," *IEEE Trans. Emerg. Topics Comput.*, vol. 4, no. 3, pp. 335–348, Jul./Sep. 2016.
[80] J. Hu and R. Marculescu, "Dyad—Smart routing for networks-on-chip," in *Proc. 41st Design Automat. Conf.*, 2004, pp. 260–263.
[81] A. Kulkarni, Y. Pino, M. French, and T. Mohsenin, "Real-time anomaly detection framework for many-core router through machine-learning techniques," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 13, no. 1, pp. 1–22, Dec. 2016.
[82] A. Kulkarni, Y. Pino, and T. Mohsenin, "Adaptive real-time Trojan detection framework through machine learning," in *Proc. IEEE Int. Symp. Hardw. Oriented Secur. Trust (HOST)*, May 2016, pp. 120–123.
[83] R. Manju, A. Das, J. Jose, and P. Mishra, "SECTAR: Secure NoC using Trojan aware routing," in *Proc. 14th IEEE/ACM Int. Symp. Netw.-Chip (NOCS)*, Sep. 2020, pp. 1–8.
[84] J. Frey and Q. Yu, "Exploiting state obfuscation to detect hardware Trojans in NoC network interfaces," in *Proc. IEEE 58th Int. Midwest Symp. Circuits Syst. (MWSCAS)*, Aug. 2015, pp. 1–4.
[85] P. D. Karandikar, "Open core protocol (OCP) an introduction to interface specification," in *Proc. 1st Workshop SoC Archit., Accel. Workloads*, vol. 10, Jan. 2010, pp. 1–26.
[86] J. Y. V. M. Kumar, A. K. Swain, S. Kumar, S. R. Sahoo, and K. Mahapatra, "Run time mitigation of performance degradation hardware Trojan attacks in network on chip," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI (ISVLSI)*, Jul. 2018, pp. 738–743.
[87] T. Boraten and A. K. Kodi, "Mitigation of denial of service attack with hardware Trojans in NoC architectures," in *Proc. IEEE Int. Parallel Distrib. Process. Symp. (IPDPS)*, May 2016, pp. 1091–1100.
[88] H. Salmani, "COTD: Reference-free hardware Trojan detection and recovery based on controllability and observability in gate-level netlist," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 2, pp. 338–350, Feb. 2017.
[89] P. Cotret, J. Crenne, G. Gogniat, and J.-P. Diguet, "Bus-based MPSoC security through communication protection: A latency-efficient alternative," in *Proc. IEEE 20th Int. Symp. Field-Program. Custom Comput. Mach.*, Apr. 2012, pp. 200–207.
[90] J.-M. Cioranesco, C. Hampel, G. O. de Almeida, and R. P. do Canto, "Cryptographically secure on-chip firewalling," in *Proc. Int. Conf. Netw. Syst. Secur.* Cham, Switzerland: Springer, 2015, pp. 428–438.
[91] M. D. Grammatikakis, K. Papadimitriou, P. Petrakis, A. Papagrigoriou, G. Kornaros, I. Christoforakis, O. Tomoutzoglou, G. Tsamis, and M. Coppola, "Security in MPSoCs: A NoC firewall and an evaluation framework," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 8, pp. 1344–1357, Aug. 2015.
[92] L. Fiorin, G. Palermo, S. Lukovic, V. Catalano, and C. Silvano, "Secure memory accesses on networks-on-chip," *IEEE Trans. Comput.*, vol. 57, no. 9, pp. 1216–1229, Sep. 2008.

[93] L. Fiorin, G. Palermo, S. Lukovic, and C. Silvano, "A data protection unit for NoC-based architectures," in *Proc. 5th IEEE/ACM Int. Conf. Hardw./Softw. Codesign Syst. Synth. (CODES+ISSS)*, Sep. 2007, pp. 167–172.

[94] A. B. Achballah, S. B. Othman, and S. B. Saoud, "Toward on hardware firewalling of networks-on-chip based systems," in *Proc. Int. Conf. Adv. Syst. Electric Technol. (IC_ASET)*, Jan. 2017, pp. 7–13.

[95] Y. Hu, D. Müller-Gritschneder, M. J. Sepúlveda, G. Gogniat, and U. Schlichtmann, "Automatic ILP-based firewall insertion for secure application-specific networks-on-chip," in *Proc. 9th Int. Workshop Interconnection Netw. Archit., Chip, Multi-Chip*, 2015, pp. 9–12.

[96] R. Fernandes, B. Oliveira, J. Sepúlveda, C. Marcon, and F. G. Moraes, "A non-intrusive and reconfigurable access control to secure NoCs," in *Proc. IEEE Int. Conf. Electron., Circuits, Syst. (ICECS)*, Dec. 2015, pp. 316–319.

[97] L. L. Caimi, V. Fochi, E. Wachter, and F. G. Moraes, "Runtime creation of continuous secure zones in many-core systems for secure applications," in *Proc. IEEE 9th Latin Amer. Symp. Circuits Syst. (LASCAS)*, Feb. 2018, pp. 1–4.

[98] C. Reinbrecht, B. Forlin, A. Zankl, and J. Sepúlveda, "Earthquake—A NoC-based optimized differential cache-collision attack for MPSoCs," in *Proc. Design, Automat. Test Eur. Conf. Exhib. (DATE)*, 2018, pp. 648–653.

[99] S. P. Azad, M. Tempelmeier, G. Jervan, and J. Sepúlveda, "CAESAR-MPSoC: Dynamic and efficient MPSoC security zones," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI (ISVLSI)*, Jul. 2019, pp. 477–482.

[100] C. Reinbrecht, A. Aljuffri, S. Hamdioui, M. Taouil, B. Forlin, and J. Sepúlveda, "Guard-NoC: A protection against side-channel attacks for MPSoCs," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI (ISVLSI)*, Jul. 2020, pp. 536–541.

[101] J. Sepúlveda, G. Gogniat, C. Pedraza, R. Pires, W. J. Chau, and M. Strum, "Hierarchical NoC-based security for MP-SoC dynamic protection," in *Proc. IEEE 3rd Latin Amer. Symp. Circuits Syst. (LASCAS)*, Feb. 2012, pp. 1–4.

[102] J. Sepúlveda, R. Fernandes, D. Florez, C. Marcon, and G. Sigl, "Dynamic security-aware routing for zone-based data protection in multi-processor system-on-chips," Tech. Rep., 2016.

[103] E. Wachter, L. L. Caimi, V. Fochi, D. Munhoz, and F. G. Moraes, "BrNoC: A broadcast NoC for control messages in many-core systems," *Microelectron. J.*, vol. 68, pp. 69–77, Oct. 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0026269217301593

[104] J. Sepúlveda, D. Flórez, V. Immler, G. Gogniat, and G. Sigl, "Efficient security zones implementation through hierarchical group key management at NoC-based MPSoCs," *Microprocessors Microsyst.*, vol. 50, pp. 164–174, May 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0141933117301424

[105] J. Sepúlveda, D. Flórez, and G. Gogniat, "Efficient and flexible NoC-based group communication for secure MPSoCs," in *Proc. Int. Conf. ReConFigurable Comput. FPGAs (ReConFig)*, Dec. 2015, pp. 1–6.

[106] B. Lebiednik, S. Abadal, H. Kwon, and T. Krishna, "Spoofing prevention via RF power profiling in wireless network-on-chip," in *Proc. 3rd Int. Workshop Adv. Interconnect Solutions Technol. Emerg. Comput. Syst. (AISTECS)*. New York, NY, USA: Association for Computing Machinery, 2018, doi: 10.1145/3186608.3186610.

[107] A. Maximov, "Two linear distinguishing attacks on VMPC and RC4A and weakness of RC4 family of stream ciphers," in *Fast Software Encryption*, H. Gilbert and H. Handschuh, Eds. Berlin, Germany: Springer, 2005, pp. 342–358.

[108] A. Vashist, A. Keats, S. M. P. Dinakarrao, and A. Ganguly, "Securing a wireless network-on-chip against jamming-based denial-of-service and eavesdropping attacks," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 27, no. 12, pp. 2781–2791, Dec. 2019.

[109] A. Vashist, A. Keats, S. M. P. Dinakarrao, and A. Ganguly, "Unified testing and security framework for wireless network-on-chip enabled multi-core chips," *ACM Trans. Embedded Comput. Syst.*, vol. 18, no. 5s, pp. 1–20, Oct. 2019, doi: 10.1145/3358212.

[110] A. Vashist, A. Keats, S. M. Pudukotai Dinakarrao, and A. Ganguly, "Securing a wireless network-on-chip against jamming based denial-of-service attacks," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI (ISVLSI)*, Jul. 2019, pp. 320–325.

[111] M. M. Ahmed, A. Ganguly, A. Vashist, and S. M. P. Dinakarrao, "AWARe-Wi: A jamming-aware reconfigurable wireless interconnection using adversarial learning for multichip systems," *Sustain. Comput., Informat. Syst.*, vol. 29, Mar. 2021, Art. no. 100470. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2210537920301943

[112] S. S. Rout, A. Singh, S. B. Patil, M. Sinha, and S. Deb, "Security threats in channel access mechanism of wireless NoC and efficient countermeasures," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Oct. 2020, pp. 1–5.

[113] S. H. Gade, S. S. Rout, M. Sinha, H. K. Mondal, W. Singh, and S. Deb, "A utilization aware robust channel access mechanism for wireless NoCs," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2018, pp. 1–5.

[114] A. K. Biswas, N. Chatterjee, H. K. Mondal, G. Gogniat, and J.-P. Diguet, "Attacks toward wireless network-on-chip and countermeasures," *IEEE Trans. Emerg. Topics Comput.*, vol. 9, no. 2, pp. 692–706, Apr. 2021.

[115] A. Ganguly, M. Y. Ahmed, and A. Vidapalapati, "A denial-of-service resilient wireless NoC architecture," in *Proc. Great Lakes Symp. VLSI (GLSVLSI)*. New York, NY, USA: Association for Computing Machinery, 2012, pp. 259–262, doi: 10.1145/2206781.2206844.

[116] J. L. Abellán, A. Ros, J. Fernandez, and M. E. Acacio, "ECONO: Express coherence notifications for efficient cache coherency in many-core CMPs," in *Proc. Int. Conf. Embedded Comput. Syst., Archit., Modeling, Simulation (SAMOS)*, Jul. 2013, pp. 237–244.

[117] U. Y. Ogras and R. Marculescu, "'It's a small world after all': NoC performance optimization via long-range link insertion," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, no. 7, pp. 693–706, Jul. 2006.

[118] R. Albert, H. Jeong, and A.-L. Barabási, "Error and attack tolerance of complex networks," *Nature*, vol. 406, pp. 378–382, Jul. 2000, doi: 10.1038/35019019.

[119] J. Sepúlveda, G. Gogniat, R. Pires, W. Chau, and M. Strum, "Security-enhanced 3D communication structure for dynamic 3D-MPSoCs protection," in *Proc. 26th Symp. Integr. Circuits Syst. Design (SBCCI)*, Sep. 2013, pp. 1–6.

[120] J. Sepúlveda, G. Gogniat, D. Florez, J.-P. Diguet, C. Pedraza, and M. Strum, "3D-LeukoNoC: A dynamic NoC protection," in *Proc. Int. Conf. ReConFigurable Comput. FPGAs (ReConFig)*, Dec. 2014, pp. 1–6.

[121] J. Sepúlveda, G. Gogniat, D. Flórez, J.-P. Diguet, R. Pires, and M. Strum, "TSV protection: Towards secure 3D-MPSoC," in *Proc. IEEE 6th Latin Amer. Symp. Circuits Syst. (LASCAS)*, Feb. 2015, pp. 1–4.

[122] D. Z. Pan, S. K. Lim, K. Athikulwongse, M. Jung, J. Mitra, J. Pak, M. Pathak, and J.-S. Yang, "Design for manufacturability and reliability for TSV-based 3D ICs," in *Proc. 17th Asia South Pacific Design Automat. Conf.*, Jan. 2012, pp. 750–755.

[123] E. J. Marinissen, "Testing 3D stacked ICs containing through-silicon vias," in *3D Integration for NoC-Based SoC Architectures* (Integrated Circuits and Systems). New York, NY, USA: Springer, Nov. 2010, pp. 47–74, doi: 10.1007/978-1-4419-7618-5_3.

[124] W.-W. Shen and K.-N. Chen, "Three-dimensional integrated circuit (3D IC) key technology: Through-silicon via (TSV)," *Nanosc. Res. Lett.*, vol. 12, no. 1, pp. 1–9, Jan. 2017, doi: 10.1186/s11671-017-1831-4.

[125] S. Mondal, S.-B. Cho, and B. Kim, "Modeling and crosstalk evaluation of 3-D TSV-based inductor with ground TSV shielding," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 1, pp. 308–318, Jan. 2017.

[126] Z. Shirmohammadi, H. Z. Sabzi, and S. G. Miremadi, "3D-DyCAC: Dynamic numerical-based mechanism for reducing crosstalk faults in 3D ICs," in *Proc. IEEE Int. High Level Design Validation Test Workshop (HLDVT)*, Oct. 2017, pp. 87–90.

[127] Q. Zou, D. Niu, Y. Cao, and Y. Xie, "3DLAT: TSV-based 3D ICs crosstalk minimization utilizing less adjacent transition code," in *Proc. 19th Asia South Pacific Design Automat. Conf. (ASP-DAC)*, Jan. 2014, pp. 762–767.

[128] H. Wieser, "Beyond planned obsolescence: Product lifespans and the challenges to a circular economy," *Gaia, Okologische Perspektiven Natur-, Geistes- und Wirtschaftswissenschaften*, vol. 25, pp. 156–160, Oct. 2016.

[129] S. Das, J. R. Doppa, P. P. Pande, and K. Chakrabarty, "Design-space exploration and optimization of an energy-efficient and reliable 3-D small-world network-on-chip," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 36, no. 5, pp. 719–732, May 2017.

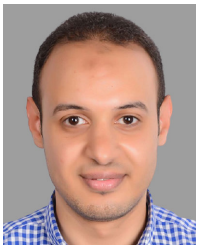
[130] G. Sharma, V. Kuchta, R. A. Sahu, S. Ellinidou, S. Bala, O. Markowitch, and J.-M. Dricot, "A twofold group key agreement protocol for NoC-based MPSoCs," *Trans. Emerg. Telecommun. Technol.*, vol. 30, no. 6, p. e3633, 2019.
[131] D.-C. Juan, S. Garg, and D. Marculescu, "Statistical thermal evaluation and mitigation techniques for 3D chip-multiprocessors in the presence of process variations," in *Proc. Design, Automat. Test Eur.*, Mar. 2011, pp. 1–6.

AMIN SARIHI (Graduate Student Member, IEEE) received the B.Sc. degree in electrical engineering from Shiraz University, Shiraz, Iran, in 2015, and the M.Sc. degree in electrical engineering from Iran University Science and Technology, Tehran, Iran, with a focus on digital systems. He is currently pursuing the Ph.D. degree in electrical and computer engineering with New Mexico State University, NM, USA. His research interests include hardware security and trust, network-on-chip security, and embedded systems security.

AHMAD PATOOGHY received the Ph.D. degree in computer engineering from the Sharif University of Technology, Tehran, Iran, in 2011. From 2011 to 2017, he was an Assistant Professor with Iran University of Science and Technology, Tehran. From 2017 to 2018, he worked as a Senior Researcher with Boston University, Boston, MA, USA. He has published more than 80 conferences and journal articles and served as a reviewer for a wide variety of IEEE, ACM, Elsevier, and Springer journals. He has also served as a Panelist of the National Science Foundation for reviewing grant proposals. He is currently leading the Intelligent and Embedded Systems Laboratory, University of Central Arkansas. His research interests include security and reliability of multi-processor system-on-chips (MPSoC), cyber-physical systems, hardware design, and acceleration for deep/spiking neural networks.

AHMED KHALID received the B.S. degree (Hons.) in computer engineering from Assiut University, Egypt, in 2020. His current research interests include network-on-chips security, deep learning, reinforcement learning, and robotics.

MAHDI HASANZADEH received the B.Sc. degree in computer hardware engineering from Azad University, Mashhad, Iran, in 2014, and the M.Sc. degree in computer systems architecture from the Science and Research Branch, Azad University, Tehran, Iran, in 2018. His research interests include hardware security, design of hardware accelerators for deep learning, deploying machine learning on embedded/edge platforms, and network-on-chips.



MOSTAFA SAID received the M.Sc. and Ph.D. degrees in computer engineering from Egypt-Japan University, Egypt, in 2012 and 2015, respectively. From 2015 to 2017, he was an Assistant Professor with Assiut University, Assiut, Egypt. He held various other academic positions throughout his career in USA, such as a Postdoctoral Research Associate with Brown University, a Research Assistant Professor with New Mexico State University, and a Research and Development Intern with Samsung. He is currently a Tenure Track Faculty Member with California State University, Bakersfield. He has published more than 20 conferences and journal articles and served as a reviewer for a wide variety of IEEE, ACM, and Elsevier journals. He has also served as a Panelist of the National Science Foundation for reviewing grant proposals. His main research interests include network-on-chips, power and thermal modeling/management of SoCs, and 3D integration.

ABDEL-HAMEED A. BADAWY (Senior Member, IEEE) received the B.Sc. degree (Hons.) in electronics engineering from Mansoura University, Egypt, in 1996, with a focus on computers and control systems, and the M.Sc. and Ph.D. degrees in computer engineering from the University of Maryland, in 2002 and 2013, respectively. He is currently an Associate Professor with the Klipsch School of Electrical and Computer Engineering, New Mexico State University, Las Cruces, NM, USA. He is also a Los Alamos Joint Faculty with the New Mexico Consortium. He has been a Visiting Research Scientist with the New Mexico Consortium. He was a Lead Research Scientist with the High-Performance Computing Laboratory, George Washington University. His research interests include computer architecture, high-performance computing, performance modeling and prediction, green computing, hardware security, and optical computing. He has more than 80 publications in conferences and journals. He has served as a PC member and a reviewer for a wide variety of IEEE, ACM, Elsevier, and Springer journals. He has also served as a Panelist at the National Science Foundation and Department of Energy for reviewing grant proposals. He is a Senior Member of the IEEE Computer Society, a Professional Member of the ACM, and a Dean's Member of the ASEE. He served as the Vice-Chair of the Arkansas River Valley IEEE Section, in 2014 and 2016. His research has been awarded best paper/poster awards.